Freescale Semiconductor
Application Note

Document Number: AN3541
Rev. 0, 1/2008

Multichannel Communication Controller HDLC Superchannel Mode on the MPC8560

by *Michael Johnston*
Networking and Multimedia Group Applications
Freescale Semiconcuctor, Inc.
East Kilbride, Scotland

This software package is an example exercising the multichannel communication controller (MCC) of the MPC8560 PowerQUICC™ III. It should serve as a reference that illustrates how to initialize the MCC itself, the interrupt controller, the serial interface, and other related steps in order to educate the user and serve as a starting point for design. This software is very low-level in nature, in regards to the OSI reference model, only addressing the data link layer and some of the physical layer. It uses no memory management or application level I/O functions. It is not intended to represent actual driver software, but rather provide a starting point from which a software engineer may educate themselves on proper initialization of this functionality.

Contents

© Freescale Semiconductor, Inc., 2008. All rights reserved.

freescale™ semiconductor

1 Software Description

Before using this code, it is highly recommended that you familiarize yourself with the *MPC8560 PowerQUICC III™ Integrated Communications Processor Reference Manual*, and the *PowerPC™ e500 Core Family Reference Manual*.

This example configures MCC1 in HDLC mode and demonstrates 8 superchannels, each composed of 16 MCC channels. Each superchannel has a ring of eight transmit buffer descriptors (TxBDs) worth of data each one with a different test pattern, with each BD being treated as a separate HDLC frame. All data is received via internal loopback at the TDM level.

Each superchannel also has a ring of eight receive buffer descriptors (RxBDs). The filling of each receive buffer is indicated in the receive buffer descriptor, and generates a new entry in the Receive Interrupt Circular Queue. Once the queue entry threshold has been reached, an external interrupt is triggered to the PowerPC core. The interrupt handler verifies that the RxBDs are error free and compares transmitted data to received data. More information on the interrupt handling process can be found in Section 7, “Software Interrupt Handling.”

If reception was successful a message will be printed on the CodeWarrior console window informing the user that the test has passed. If there was some error recorded an error message will be printed to the screen and the program will spin in a while loop at the point of failure.

It is important to note that this software does not perform complete initialization of the part. Initial device set-up is done using the 8560ADS_RevA_init.cfg configuration file.

2 Test Configuration and Code Modifications

2.1 Channel Numbers and Modes

This example uses all 128 HDLC channels on MCC1, TDMA as follows:

Superchannel 0 => 0-16 MCC channels

Superchannel 16 => 16-31 MCC channels

Superchannel 32 => 32-47 MCC channels

Superchannel 48 => 48-63 MCC channels

Superchannel 64 => 64-79 MCC channels

Superchannel 80 => 80-95 MCC channels

Superchannel 96 => 96-111 MCC channels

Superchannel 112 => 112-127 MCC channels

2.2 Selection of Serial Interface and TDM Bus

The code is written to use MCC1 as the default controller but this can be easily changed to MCC2 by setting ‘MCC’ to equal ‘MCC2’ in main() of MCC_HDLC.c. Note however that the user is responsible for correct configuration of the parallel I/O ports and external wiring of those signals, as well as any SIRAM programming changes, when changing to MCC2. The same is true when changing the ‘TDM’ setting from

the default of ‘TDMA’. Please refer to the Serial Interface, MCC, and Parallel IO chapters of the 8560 Reference Manual for additional information.

2.3 Continuous or "ONE-SHOT" Testing

The test can either run once or can perform a continuous testing loop. This is achieved by setting testmode at the top of MCC_HDLC.c to either RUN_ONCE or CONTINUOUS. If set to CONTINUOUS, the test will still disable the TDM and issue stop commands to superchannels at the end of each loop of the test. An interrupt is taken once all expected receive buffers have been filled for a given test loop. The user may keep track of how many times the interrupt handler has been called (and thus, how many test loops have been performed) by looking at the IntCounter global variable.

2.4 Buffers, Buffer Descriptors, Interrupt Queues and Their Respective Buses

The user may select which bus the data buffers are on, as well as which bus the BDs and interrupt queues are on (BDs and interrupt queues must always be on the same bus). These parameters are chosen by setting the BD_BUS and BUFFER_BUS constants to either PPC or LOCAL, and this define is included in mcc.h. The space used for receive and transmit data buffers is called the buffer pool. The starting memory location of this space is set through the BUFFERPOOLBASE parameter in mcc.h. Likewise, the buffer descriptors are located via BDRING_BASE in mcc.h. The starting addresses for the transmit and receive interrupt queues are selected by the TX_INTCQ and RX_INTCQ declarations, also in mcc.h.

If the user wishes to change these locations to addresses on a different bus, the user is responsible for properly setting the aforementioned bus constant declarations to reflect the new address settings. This code package already handles changing the TSTATE and RSTATE registers to indicate the correct buses for data and BDs/interrupt queues. Note that TSTATE and RSTATE are set in MCCChanSpecInit().

The example uses the constant BUFFER_SIZE, which is set at 256 bytes at the top of mcc.h. In InitBDs() in MCC_HDLC.c, the receive buffers are set to BUFFER_SIZE and the size of transmit buffers is set to (BUFFER_SIZE - 4). For the purposes of this example, HDLC transmit BD length fields are kept 4 bytes less than the receive buffer to leave room in the Rx buffer for 32 bit CRC.

Only one set of transmit data buffers is used, and each channel has its own set of TxBDs that point to this same area. Each frame will be received in its own buffer.

The transmit buffers contain the following test patterns:

1. Buffer #0—0x55
2. Buffer #1—0xAA
3. Buffer #2—0x00
4. Buffer #3—0xFF
5. Buffer #4—Incrementing "Walking Ones"
6. Buffer #5—Decrementing "Walking Ones"
7. Buffer #6—Increment from 0 to 255
8. Buffer #7—Decrement from 255 to 0

If the user wishes to change the data patterns used in the example, they may do so in the LoadTxBuffers() routine.

3 Development Environment

Please note that this project was specifically designed to run on an MPC8560ADS board (revA of the board was used during development with CPU/CCB frequencies of 833/333MHz). It was tested using CodeWarrior development studio for Power Architecture v8.7 (Build 61218) and USBTAP.

In order to run the code it requires some external connections to be made on the CPM expansion connector of the MPC8560ADS (P30). The code sets up BRG5 and TMR1 to be the clock and sync respectively for TDMA1. This requires the following steps:

1. Connect D9 to D1 and D4. This is BRG5 (PC23) to CLK1 (PC31, the clock input for the TDM) and TIN1 (PC28)
2. Connect D2 and B26. This is TOUT1 (PC30) and L1RSYNC (PA6).

For more information on the clocking scheme used please refer to Section 6, “Clocking and Timer Configuration.”

3.1 Useful Addresses

For the particular compiled version enclosed in this package, the user may find the following addresses useful to look at (assuming PPC bus used for BDs and buffers):

0x0000B9D0—global variable "IntCounter," counts how many test loops have been run

0x00110000—base of Rx buffer descriptor ring

0x00110200—base of Tx buffer descriptor ring

0x00120000—base of Tx interrupt circular queue

0x00120600—base of Rx interrupt circular queue

0x00400000—base of data buffer pool

0x40000000—CCSRBAR address

0x40080000—channel-specific parameter ram base

0x40089000—channel extra parameter ram base

0x40081900—superchannel table

0x40088700—MCC1 global parameter

4 Files Included in this Package

The following files are included in this example:

- mcc_hdlc.mcp—CodeWarrior project file
- mcc.h—Header file containing structure and constant definitions specific to this application
- MCC_HDLC.c—Main source code file

- MPC8560.h—This header file contains the Internal Memory Map structure declarations for the MPC8560.
- netcomm.h—This header file contains global data type definitions.
- 8560ADS_RevA_init.cfg—CodeWarrior configuration file for 8560ADS revA.

5 Expected Output

The following output should be seen on the CodeWarrior console window if the test is successful:

MCC HDLC superchannel example software

EPIC: Disable External Interrupts

EPIC: Reset in progress...

EPIC: Reset completed...

EPIC configured in mixed mode...

EPIC: CPM Configure...

Connect ISR...

Enable

EPIC: Enable External Interrupts in MSR

EXCEPTION 500: An External Interrupt

TEST LOOP COMPLETED SUCCESSFULLY

NOTE

If continuous mode is enabled, the last two print-outs will not occur. The test will run until the user issues a stop command.

6 Clocking and Timer Configuration

In order to supply the clock and sync required for correct TDM operation the following registers are configured:

1. cmxsi1cr = 0x00000000
   => Configures TDMA1 RX clock as CLK1, TDMA1 TX clock as CLK2
2. brgc5 = 0x0001000A
   => Configures and enables BRG5
3. tgcr1 = 0x09, tmr1 = 0x1f0e, trr1 = 0x00ff
   => Configures timer TOUT1 to have TIN1 as an input and to reset immediately after the reference value in trr1 is reached.

The register settings above combined with the external connections given in Section 3, “Development Environment,” provide the following sources for TDMA clock and sync:

- BRG5 provides the clock input to the CLK1 pin which is used as TDMA RX clock. Since the CRT bit is set in SIXMR (see SIinit()), RX and TX have same clock and sync.

- BRG5 also provides the clock input to TIN1 which is used as input to generate TOUT1. TOUT1 is connected externally to L1RSYNC.

7 Software Interrupt Handling

The software is configured to use the Programmable Interrupt Controller (PIC) of the MPC8560 to handle receive frame interrupts on MCC1. The PIC is initialized using the epicInit() function. epicDefault() in epicInit() is responsible for enabling CPM interrupts to the PIC. The MCC receive interrupt will occur when the GRFCNT value in the MCC PRAM reaches 0, this should occur after all 8 buffers are received for each of the 8 superchannels, i.e. after 64 frames are received. When an MCC receive frame interrupt occurs a CPM interrupt is signalled to the core and an external interrupt exception (0x500) is taken. The interrupt handler in interrupt.c is then responsible for calling epicISR() and this calls the function linked to the CPM interrupt, in this case cpmIsr(). cpmIsr() is then responsible for verifying the interrupt is for an RX event on MCC1. At this point the RxBDs are checked for errors and the received data compared to the data transmitted. Assuming the test passes ‘NotDone’ is set to FALSE and this allows the main flow in main() to continue.

8 MCC CPM Performance Calculator

If developers wish to obtain a CPM performance estimation for their MCC configuartion this can be achieved using the CPM performance calculator (MPC8260CALC2) available from the MPC8560 website on http://www.freescale.com.

9 Revision History

Table 1 provides a revision history for this application note.

Table 1. Document Revision History

Rev. Number	Date	Substantive Change(s)
0	01/29/2008	Initial release.

THIS PAGE INTENTIONALLY LEFT BLANK

How to Reach Us:

Home Page:
www.freescale.com

Web Support:
http://www.freescale.com/support

USA/Europe or Locations Not Listed:
Freescale Semiconductor, Inc.
Technical Information Center, EL516
2100 East Elliot Road
Tempe, Arizona 85284
+1-800-521-6274 or
+1-480-768-2130
www.freescale.com/support

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
www.freescale.com/support

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku
Tokyo 153-0064
Japan
0120 191014 or
+81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor
Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
+1-800 441-2447 or
+1-303-675-2140
Fax: +1-303-675-2150
LDCForFreescaleSemiconductor
@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org. The PowerPC name is a trademark of IBM Corp. and is used under license. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc., 2008. Printed in the United States of America. All rights reserved.

Document Number: AN3541
Rev. 0
1/2008