Freescale Semiconductor
Application Note

Document Number: AN3400
Rev. 1, 09/2007

Using the Programmable Interrupt Timer on the MCF5213 ColdFire Microcontroller

by: Alfonso Gonzalez
Daniel Torres
RTAC Americas

1 Introduction

This document explains how to configure the programmable interrupt timer (PIT) module for the ColdFire™ MCF5213 microcontroller (MCU). Basic knowledge of the function and configuration options help you understand on how the PIT module works. The examples in this application note may be modified to suit the specific needs of any application.

2 Overview

Each PIT is a 16-bit timer that provides precise interrupts at regular intervals with minimal processor intervention. The timer can count down from the value written in the modulus register or it can be a free-running down-counter.

Contents

freescale™ semiconductor

3 Description

The PIT generates a periodic interrupt request to the core when its counter reaches the 0 value. Each PIT channel uses two values to calculate the timeout value of the counter. Those values are:

- PIT prescaler — Located in the PIT control and status register, this is a four-bit field with values that can vary between 0x0 and 0xF. This value divides the system clock by 2 (Prescaler Value) (1 – 32768).
- PIT modulus value — This value is stored on the PIT modulus register. This value is the counter value reloaded to the PIT's counter register every time it reaches 0 value.

The PIT's timeout value is given here:

1/(Internal Bus clock) * 2(Prescaler Value+1) * Module Value = Time (seg). ***Eqn. 1***

4 Case Study for the Programmable Interrupt Timer

The example code is intended to show how the PIT works, generating an interrupt request to turn on and off an LED using the two PIT channels and a GPIO port. For this example, the M5213EVB MCU was used, so the LEDs 0 and 1 of the port C were used. The LED0 is turned off/on every second and the LED1 is turned on/off every two seconds. Each interrupt request has a different interrupt level and priority (for more information on the interrupt controller, refer to Chapter 12, "Interrupt Controller Module," of the MCF5213 Reference Manual). According to the level and priority, one interrupt service routine (ISR) interrupts in the middle of the other with lower priority ISR. For this demo, the PIT1 ISR has greater priority than PIT0 ISR.

The IntConfig function clears the PIT0's and PIT1's vectors bit (one at the time) in the interrupt mask register (IMR) to enable the interrupt request from the PIT's vectors; it also configures the interrupt level and priority for the designed vectors and assigns the handler function to the PIT0 and PIT1 vectors. The function can be used to configure any user interrupt vector of the MCF5213 MCU. Figure 1 shows the flow chart of the function.

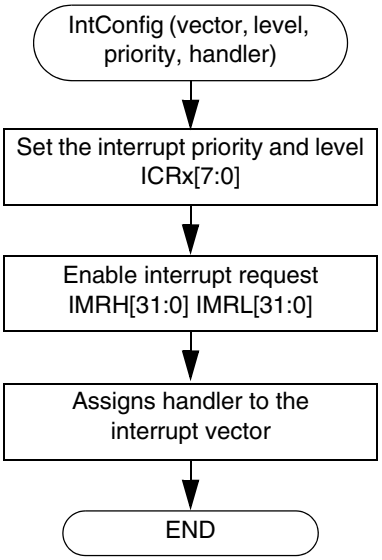

Figure 1. Interrupt Enabler Flowchart

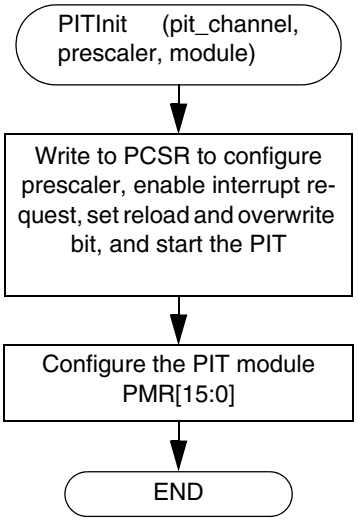

Figure 2. PIT Initialization Flowchart

The PIT initialization function configures the PIT channels. One to generate an interrupt request every second; and the second PIT to generate an interrupt every two seconds. Figure 2 follows the process of this function.

The functionality of the ISR for both PITs is the same (see Figure 3). It turns on the LED 0 or the LED 1 (PIT0 manages LED 0 and PIT1 manages LED 1) if the LED is off, or it turns the LED off if it is on. The frequency of the signal driven by the LED is twice the PIT's timeout value.

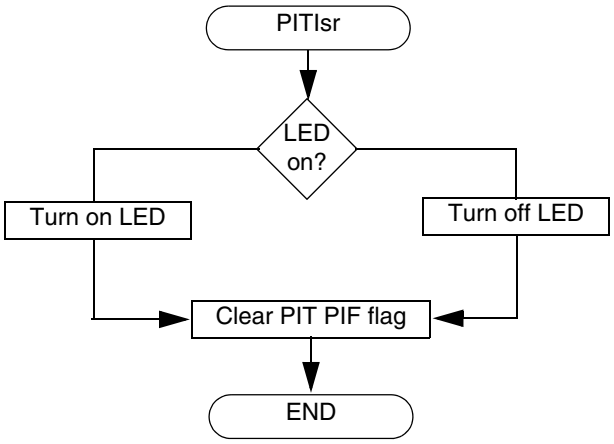

Figure 3. PIT Interrupt Service Routine Flowchart

4.1 Register Configuration

The following sections explain how the PITInit function initializes the registers.

4.1.1 PIT Control and Status Register (PCSR)

Figure 4 shows the configured PCSR. The following code lines load the initialization value of the register:

```
MCF_PIT_PCSR(pit) |= MCF_PIT_PCSR_PIF;             /*clear PIT interrupt flag*/
MCF_PIT_PCSR(pit) =(MCF_PIT_PCSR_PRE(prescaler)    /*defines prescaler*/
                   /*interrup enable*/
                   |MCF_PIT_PCSR_PIE
                   /*reset counter when write to PMR*/
                   |MCF_PIT_PCSR_OVW
                   /*reload module value when counter reaches 0*/
                   |MCF_PIT_PCSR_RLD
                   /*enable PIT*/
                   |MCF_PIT_PCSR_EN);
```

IPSBAR 0x15_0000 (PCSR0) Access: Supervisor
Offset 0x16_0000 (PCSR1) read/write

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	PRE				0	DOZE	DBG	OW	PIE	PIF	RLD	EN
W														w1c		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Written	0	0	0	0	1	1	1	1	0	0	0	1	1	1	1	1

Figure 4. PIT Control and Status Register Configuration

4.1.2 PIT Modulus Register (PMR)

In the code line below the module value is written to the PMR. Figure 5 shows the module value for this example.

```
MCF_PIT_PMR(pit) = MCF_PIT_PMR_PM(module);          /*write module to PMR*/
```

IPSBAR 0x15_0002 (PMR0) Access: Supervisor
Offset 0x16_0002 (PMR1) read/write

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

Figure 5. PIT Modulus Register Configuration

4.2 Configuration Summary

The following steps are needed to configure the PIT module:

- Calculate the modulus value to generate the desire interrupt frequency
- Configure the interrupt control module to enable the PIT's interrupt requests
- Configure the PCSR to set all the desired features of the PIT and enable the PIT
- Write the modulus value to the PMR to reload the PIT counter register (PCNTR) after it reaches zero

5 Configuration Notes

The following details are important when configuring the PIT:

- The PCNTR cannot be written. It can only be modified by writing to the PMR when the OVW bit is set.
- The order of the initialization process is important if a desired modulus value is needed. First, configure the PCSR and set the overwrite (OVW) bit. Then, write the PMR. This causes the PCNTR to load the PMR value to the PCNTR.
- The PCNTR's reset value is 0xFFFF.

6 Conclusion

The PIT is the easiest way to generate a periodic interrupt on the Coldfire MCF5213. It could be used to perform many routines, from periodic signals to scheduling functions. Also, the PIT has only two registers that can be configured and a very easy to calculate timeout value.

7 References

Find the newest software updates and configuration files for the MCF5213 at www.freescale.com.

The M5213EVB development board employs PIT software demo.

For more information on the programmable interrupt timer module, refer to MCF5213 ColdFire Integrated Microcontroller Reference Manual at www.freescale.com.

The PITSoftwareDemo software was developed in CodeWarrior for ColdFire V6.3.

Download the source files for PITSoftwareDemo software (PITSoftwareDemo.zip) from www.freescale.com.

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 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Document Number: AN3400
Rev. 1
09/2007

freescale™
semiconductor