Micromint, Inc

AN813

MicroBolt

Using the Watchdog Timer on the MicroBolt

10/7/2005

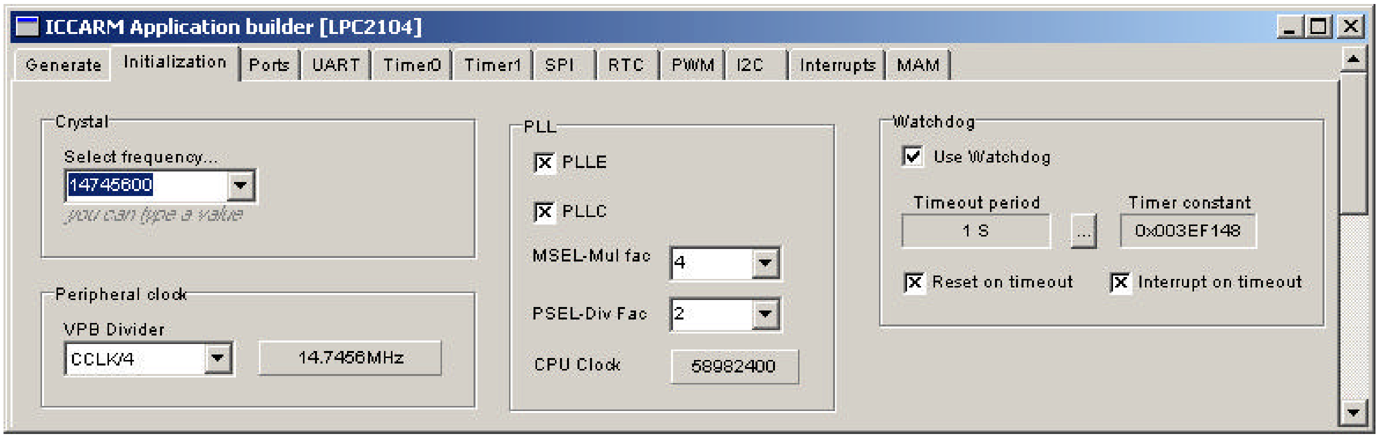
Introduction:
This application notes demonstrates how to use the watchdog timer on the MicroBolt.

Background:
The MicroBolt contains a watchdog timer to protect and reset the system during fatal software crashes.

How it works:
This ImageCraft ICCARM demo project simply blinks the Micromint MicroBolt's LED on and off every second. In addition to the LED, the internal watchdog timer is utilized. The watchdog is reset or "kicked" via the KickTheDog(); function call. This function disables interrupts, writes the proper sequence to the watchdog for reset, then re-enables the interrupts. This sequence must be written via back to back writes to reset the watchdog, and this is why interrupts are disabled.

If you comment out the KickTheDog(); function call and then download the program, you'll notice the MicroBolt LED is turned off after 1 second. This is due to the watchdog timer timing out. You can easily adjust the watchdog timeout by using the ImageCraft Appbuilder utility (see below screenshot). It will allow you to enter in a value that is then converted to a hexadecimal value. This code can then be generated and pasted into your application.

Screenshot:



Crystal
Select frequency... 14745600

Peripheral clock
VPB Divider CCLK/4 14.7456MHz

PLL
☒ PLLE
☒ PLLC
MSEL-Mul fac 4
PSEL-Div Fac 2
CPU Clock 58982400

Watchdog
☑ Use Watchdog
Timeout period 1 S
Timer constant 0x003EF148
☒ Reset on timeout
☒ Interrupt on timeout

Program Listing:

```
/*
|------------------------------------------------------------------------------------------
|------------------------------------------------------------------------------------------
| File Name                      : MicroBoltWatchdog.c
| Author                         : Micromint, Inc.
| Copyright                      : Copyright © 2005, Micromint, Inc.
| Creation Date                  : 9/30/05
| Version                        : 1.00
| Spaces per tab                 : 2
| Description                    : Main C file
| Revision                       : Initial
|------------------------------------------------------------------------------------------
|------------------------------------------------------------------------------------------
*/

/*
|----------------------------------------------------
|     Includes
|----------------------------------------------------
*/

#include <ARM/philips/lpc210x.h>
#include <arm_macros.h>

#include "MicroBoltWatchdog.h"

/*
|------------------------------------------------------------------------------------------
|     Function        :   main
|     Inputs          :   None
|     Outputs         :   None
|     Purpose         :   Main function for system
|     Author          :   Micromint, Inc.
|------------------------------------------------------------------------------------------
*/

void main(void)
  {
        unsigned int Delay;

/*
|----------------------------------------------------
|     MicroBolt hardware setup
|----------------------------------------------------
*/

  __DISABLE_INTERRUPT();                        // Disable all interrupts

/*
|-------------------------
| Configure MAM
|-------------------------
*/
  MAM_CR = 0x00;                                // Turn MAM off (default)
  MAM_TIM = 0x04;                               // Set flash timing to 4 clock cycles

  MAM_CR = 0x02;                                // Fully enable the Memory Accleration Module

/*
|-------------------------
| Configure VIC
|-------------------------
*/
  VICVectAddr0 = (unsigned)pll_isr;             // Assign the PLL lock ISR vector address
  VICVectCntl0 = INTERRUPT_CHANNEL_FOR_PLL;     // Assign the VIC address to the actual interrupt
  VICIntEnable = INTERRUPT_ENABLE_FOR_PLL;      // Enable the interrupt

/*
|-------------------------
| Configure PLL and CCLK
|-------------------------
*/
```

```
  SCB_PLLCFG |= 0x23;                            // Set to 59 MHz (0x03 is multiply value of 4)
  SCB_PLLCON |= 0x01;                            // Enable the PLL
  SCB_PLLFEED = 0xAA;                            // Shadow register copy to enable changes
  SCB_PLLFEED = 0x55;                            // in PLLCON and PLLCFG

/*
|-------------------------
| Configure PCLK
|-------------------------
*/
  SCB_VPBDIV = 0;                    // Peripheral clock is 1/4th Processor clock which equals 14.7456 MHz

/*
|-------------------------
| Configure GPIO
|-------------------------
*/
  PCB_PINSEL0=0x00000000;              // JTAG is via secondary port (Configured via Imagecraft App builder)
  PCB_PINSEL1=0x55400000;

  GPIO_IODIR=(0x00000000<<16)|                   // Make all inputs to start with
             0x00000000;

  GPIO_IODIR |= MICROBOLT_LED;                   // Setup MicroBolt LED as output

/*
|-------------------------
| Configure Watchdog
|-------------------------
*/
  WD_WDTC=0x003EF148;                            // Watchdog timeout set to 1 second
  WD_WDMOD=0x00000003;                           // Watchdog interrupts and causes reset
  WD_WDFEED=0xaa;                                // Start the watchdog
  WD_WDFEED=0x55;

  __ENABLE_INTERRUPT();                          // Enable all interrupts

  GPIO_IOCLR = MICROBOLT_LED;                    // MicroBolt LED Off

  for (Delay = 0; Delay < 3000000; Delay++);     // Delay for a few (Shows watchdog reset if applicable)

/*
|----------------------------------------------------
|    Start of application
|----------------------------------------------------
*/

  while(1)                                       // Do this forever
    {
    KickTheDog();         // Kick the watchdog (Comment this line out to see watchdog reset every second)

    GPIO_IOSET = MICROBOLT_LED;                  // MicroBolt LED On
    for (Delay = 0; Delay < 500000; Delay++);    // Delay for a few
    GPIO_IOCLR = MICROBOLT_LED;                  // MicroBolt LED Off
    for (Delay = 0; Delay < 500000; Delay++);    // Delay for a few
    }
  }

/*
|----------------------------------------------------
|    Functions
|----------------------------------------------------
*/

/*
|-------------------------------------------------------------------------------------------
|    Function         :   KickTheDog
|    Inputs           :   None
|    Outputs          :   None
|    Purpose          :   Reset the watchdog timer
|    Author           :   Micromint, Inc.
|-------------------------------------------------------------------------------------------
*/
```

```
void KickTheDog(void)
  {
  __DISABLE_INTERRUPT();                        // Disable all interrupts

  WD_WDFEED=0xaa;                               // Reset the watchdog with back to back writes
  WD_WDFEED=0x55;

  __ENABLE_INTERRUPT();                         // Enable all interrupts
  }

/*
|------------------------------------------------------------------------------------------
|     Function        :   pll_isr
|     Inputs          :   None
|     Outputs         :   None
|     Purpose         :   Once PLL has locked, connect it and use for system clock
|     Author          :   Micromint, Inc.
|------------------------------------------------------------------------------------------
*/

#pragma interrupt_handler pll_isr

void pll_isr(void)
  {
  SCB_PLLCON |= 0x02;                           // Connect the PLL
  SCB_PLLFEED = 0xAA;                          // Shadow register copy to enable changes
  SCB_PLLFEED = 0x55;                           // in PLLCON and PLLCFG
  VICIntEnClear = PLL_CLR;                      // Disable the PLL interrupt, not needed anymore
  VICVectAddr = VIC_ACK;                        // Acknowledge Interrupt
  }
```