Jose Capestany

Sayaf Almeri

EE215

3/10/2016

LAB5 Report

The purpose of this lab was to use the timer module to make a red LED flash on and off in Code Composer Studio. Before we were allowed to code we had to answer questions on specific topics that we had to answer. The following sections and questions are what we had to answer, followed by our response:

**Preliminary Concepts:**

1. Explain the code used to set, clear and toggle a single bit in a register while leaving the other bits unchanged.

**You can set a bit by ORing it with a 1. You can clear a bit by ANDing it with a 0. You can toggle a bit by XORing it with a 1.**

1. What information is available in the header file? What useful definitions are made in the header file will we be using for this lab?

**A header file contains function declarations and macro definitions that can be shared between source files. Our header file contains the pin and ports for the programming board.**

1. Why does each version of the MSP430 have its own header file?

**It is because each version of MSP430 is slightly different and needs to be correct in its pin and port numbers.**

1. On the development board that we are using, what are the physical connections that are provided between the MSP430 and the rest of the devices?

**The physical connections that are provided are the PINs.**

1. What is the watchdog timer and why does it need to be put on hold for this application? (Our text “MSP430 Basics” has a good explanation of this)

**The watchdog timer resets the MSP430 if is not constantly pinged. We are disabling it so our program can run indefinitely.**

**Timer Module:**

1. Give a detailed explanation of each of the functions of the bits in the TA1CTL register. Go into the header file for our chip “msp430f5438a.h” and identify the useful definitions that will be used to configure the Timer A module.

**Bit 0 is the Timer A interrupt flag. Bit 1 is Timer\_A interrupt enable, this enables an interrupt request. Bit 2 is Timer\_A clear, and resets the count direction. Bit 3 is unused. Bits 4 and 5 are for mode control (up, continuous, up/down, stop). Bits 6 and 7 is for the input divider for the input clock. Bits 8 and 9 is for the Timer\_A clock source. Bits 10-15 are unused.**

1. Explain in words how the Timer A module will be configured using the definitions in the header file.

**It will be configured with and interrupt flag and with interrupt enable. It will clear and be in up-mode. The input clock is SMCLK and the input divider is 4.**

1. We will use “SMCLK” and “Up-Mode”. Explain these options in detail.

**The SMCLK is what are programming board will use for the input clock source. Up-mode means that the timer will count up to a specific point and then restart at 0.**

**Digital I/O Ports:**

1. What GPIO pin is this LED connected to?

**It is connected to the 1.0 pin.**

1. Scan through the sections in the user’s guide and the device data sheet on I/O ports. Explain what port control registers we need to set for the LED1 to blink.

**The port control registers we need to use are ports 1 and 2 for inputs and 3-8 for outputs.**

**Flowchart:**

The following flowchart is a simple representation of what intend our code to follow. The LED is off and then waits a specified amount of time before toggling the LED. The counter is reset and the process keeps going indefinitely since we are disabling the watchdog timer.

C:\Users\Jose Capestany\Downloads\LAB5.png

**Conclusion:**

This program was very hard to figure out as we had little C programming experience before. Once we realized how the Timer\_A module worked, then it became a whole lot simpler. We realized that we just needed to toggle one bit and make sure the Timer\_A module was setup correctly. We will most likely prepare slightly more before labs from now on.

**APPENDIX: CODE**

