EENG 3910: Project V – Digital Signal Processing System Design
Assignment 2
Due 8 February 2016
Nathan Ruprecht
UNT – Electrical Engineering

Introduction

Assignment 2 is an introduction to interrupts, UART, and SysTick. The first 2 problems show the difference between using the SysTick timer and Timer0 peripheral while using interrupts to toggle the LED at a varying frequency. Problem 3 is the introduction to using putty to use the board UART capabilities. And then problem 4 is summarizing what we read in certain technical documents.

Results and Discussion

Problem 1: Blink LEDs with Timer0
Code breakdown:

-Setting up the clock at 40MHz

-Calling a function to initialize the LEDs. This includes enabling port f where the LED pins are located, then defining which pins will be used: PF1, PF2, PF3.

-Calling a function to initialize the timer. Includes enabling the peripheral, Timer0, as a 32-bit timer in periodic mode. So it's a 32-bit number that counts down to 0, then restarts the countdown. When the interrupt happens, the code calls the function Timer0_ISR.

-Timer0_ISR clears the flag for the interrupt so it is reset, toggles the Blue LED, and then returns to the previous function.

Whatever the value of TIMER0_FREQ, the frequency of the signal is half of that. So for a signal frequency of 1Hz, 2Hz, 5Hz, and 10Hz, TIMER0_FREQ is 2Hz, 4Hz, 10Hz, and 20Hz, respectively.

Problem 2: Blink LEDs with SysTick Timer
Code Breakdown:

-Setting up the clock at 80MHz

-Like Lab2_1, this code calls a function to initialize the LEDs by enabling port f then the 3 pins for each color.

-Then calls a function to initialize the system tick. The difference here is that the timer is a 24-bit instead of a 32-bit. Again, it sets up a function to call when the interrupt flag is triggered: SysTick_ISR.

-SysTick_ISR just toggles the green LED

TIMER0_FREQ	Signal Freq
10	5
8	4
5	2.5
2	6.207
1	3.103

4.8Hz works fine, but 4.7Hz is the smallest frequency before it breaks down. I couldn't say one way or another, but I thought the change in timer size (24-bit vs 32-bit) is why there is a breakdown in signal frequency as time0_freq went below 4.8Hz.

Problem 3: Serial Communication with UART
Code Breakdown:

-Set system clock at 40 MHz
-The usual initialization of LEDs, UART, and timer that we've seen
-Enabling the interrupt and timer0
-Display the "instructions" text to the screen
-Check for user input and run functions depending on switch statement. Toggle red, green, or blue LED if r, g, or b key is hit. Displaying the "instructions" text again if h is hit.

Problem 4: Summary of documents
SW-TM4C-DRL-UG-2.1.1.71 – Chapter 18 is the interrupt controller, Chapter 29 is System Tick (SysTick), 30 is the Timer, and 31 is UART. For all of these chapters, the main point is all the functions we can use in order to control different aspects of the board. Again, it goes into greater detail by breaking down and explain each function so we can better analyze code we're given as well as writing code to achieve what we want.

SW-TM4C-UTILS-UG-2.1.1.71 – Since this assignment introduced UART, it makes sense that one of the technical documents assigned includes it. Chapter 20 talked about the UART Standard IO Module. The chapter covered the different functions at our disposal that use the UART and breaks down the functions for a better understanding of how to manipulate them.

tm4c123gh6pm – Chapter 2 had a lot of information. The chapter was talking about the processor our board uses, the Cortex-M4F processor. It started with general statistics of the processor, then a block diagram of how it is setup. The chapter went into great detail covering a lot of technical topics such as stacking, registers, pointers, memory, etc.

<u>Summary and Conclusion</u>

This assignment was less of coding and more of seeing the effects of signal frequency change by changing timer0 frequency. I understand the big picture use of interrupts and their importance, but implementation is still a foreign concept to me. Not coding anything made it difficult so practicing in future labs will be important. Problem 3 was big because we now have a way for user input and board responses. The only code I ever changed was the TIMER0_FREQ at the beginning of each program.