EE445M – Lab 1: Graphics, LCD, Timer and Interpreter

Kristian Doggett and Akshar Patel

1/13/15

A. Objectives

Goals:
Refreshing knowledge of TM4C123 Launchpad
Interrupt driven UART
TFT driver for two logical displays
Multi-channel ADC driver
Periodic interrupts with GPTimer
Build command line interpreter

Summary:

In lab one, we are refamiliarizing ourselves with our development board, the TM4C123 Lanchpad, by designing basic uController functions such as ADC, UART, and timers. Additionally, we are laying a framework for future labs by designing a command line interpreter and adding drivers for our TFT screen. We are enhancing our prior ADC drivers to allow for multiple channels and specifying sampling rates and number of samples dynamically. Instead of busy-wait, our UART driver now uses interrupts, which allows for future development of data exchanges using FIFOs. We measured the ISR times of our general purpose timer so we can get an idea of the overhead for ISRs in general. Finally, we separated our TFT into two logical displays in anticipation of more advanced graphics capabilities and built the foundation of a CLI to interact with future systems.

B. Hardware Design

None for Lab 1.

C. Software Design

1. LCD Driver
2. ADC Driver
3. Timer Driver
4. Interpreter

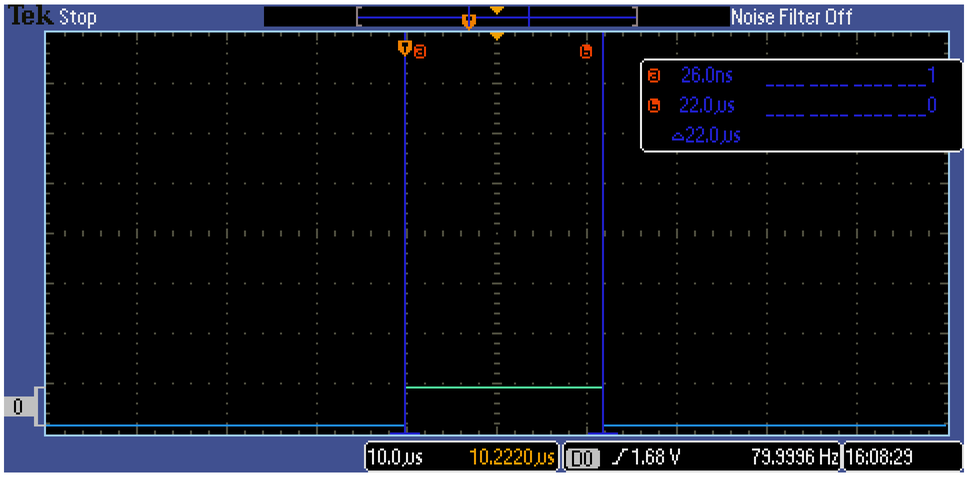
All code is included at end of report.

D. Measurement Data

1. Estimated time of periodic interrupt:

(# of instructions in ISR handler) * 12.5 ns = 16 * 12.5 ns = .2 us

2. Measured time to run periodic interrupt: 22 us



E. Analysis and Discussion

1. ADC Range/Resolution/Precision

   a. Range (max/min ADC inputs): 0 to 3 V
   b. Precision (number of distinguishable inputs): 12 bits
   c. Resolution (smallest distinguishable change): 1 mV

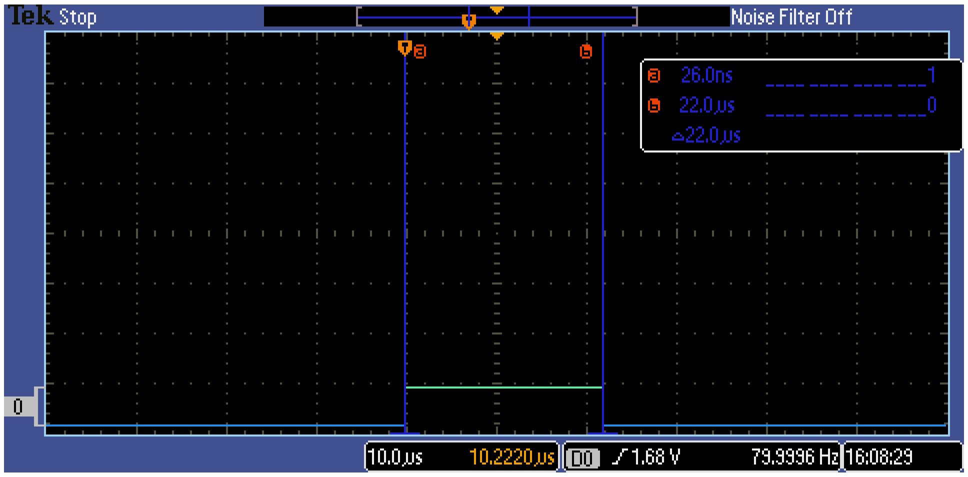
2. List the ways the ADC conversion can be started. Explain why you choose the way you did.

   Sample triggering for each sample sequencer is defined in the ADC Event Multiplexer Select
   (ADCEMUX) register.

   Trigger sources include:
   1. processor (default)
   2. analog comparators
   3. external signal on a GPIO
   4. GP Timer
   5. PWM generator
   6. continuous sampling

3. Measured time to run periodic interrupt.

We used the method of toggling a bit within the ISR. Out measured time to run the ISR was 22 us. We did not use the alternate method of calculating time lost because it was simpler to calculate ISR time.



4. There is a large discrepancy in our measured time versus estimated time.

We measured 22 us and assuming the ISR takes 16 instructions:

time to execute ISR/# of instructions = 22us/16 = 1 us bus speed, which is clearly incorrect.

Assuming clock is executing an instruction every 12.5 ns:

# of instructions = time to execute ISR/system clock = 22 us/12.6 ns = 1,760 instructions, which is clearly not correct.

This means we are either measuring the ISR time incorrectly or estimating the number of instructions in correctly. Most likely measuring the ISR time incorrectly since the calculated instruction number of 1,760 makes our measured ISR time off by two orders of magnitude.

If the ISR uses 16 instructions with a bus speed of 12.5 ns we would anticipate the ISR time as

ISR time = bus speed * # of instructions = 12.5 ns * 16 = .2 us

Estimated instructions: 16
Calculated instructions: 1,760

Measured ISR time: 22 us

Calculated ISR time based on 16 instructions: .2 us

We were discussing this on Piazza (https://piazza.com/class/i4ye1tm4y9s4ve?cid=15) but never resolved it by the time of the report.

5. What is range, resolution, and precision of SysTick timer?

   Range: 0 to 2^24
   Resolution: 12.5 ns
   Precision: 24 bits