Whitlatch, Robert

rmw2634

Lab 1

A) Objectives

The objectives of this lab were to familiarize ourselves with the low level device drivers needed on the Tiva TM4C123GH6PM Launchpad. Additionally, we were to design and implement a command interpreter, and the communication protocols necessary to access it. We were also to learn about interrupts, and their implications (i.e. time required, general implementation, etc.). Finally, we began to write a few fundamental function for our OS.

B) Software

```
// ************** ST7735_Message **************** //

// Sends message to screen partition indicated.

// Input: device to be used, which line to print to,

//       string to be printed, a number that will be printed last,

//       and whether the number should be printed.

// Output: None

// ********************************************** //


 // -----------ADC0_Open-----------

// Prepares an ADC0 channel to execute conversions, in a busy-wait fashion

// Input: Channel to be opened

// Output: None


//------------ADC0_In------------

// Busy-wait Analog to digital conversion

// Input: none

// Output: 12-bit result of ADC conversion
```

```
//------------ADC0_Collect--------
// Collect a specificed amount of ADC data
// Input: Channel to be collected from, frequency to collect at,
//       Buffer to store data in, and number of samples to take
// Output: None

//-----------ADC0_Status-----------
// Return status of the previous ADC0_Collect task.
// Input: None
// Output: Number of samples remaining, 0 indicates completion

// ***************** OS_AddPeriodicThread ****************
// Submits a task to be preformed periodically
// Inputs:  task is a pointer to a user function,
//         period in usec, priority of the task
// Outputs: none

// ***************** OS_ClearPeriodicTime ****************
// Resets periodic timer
// Inputs:  task is a pointer to a user function,
//         period in usec, priority of the task
// Outputs: none

// ***************** OS_ReadPeriodicTime ****************
// Return value of periodic timer
// Inputs:  task is a pointer to a user function,
//         period in usec, priority of the task
// Outputs: none
// -------------processCommand------------------
// Processes and executes entered command
// Input: Command string, and its length
// Output: None
```

D) Measurement Data

1) Estimated number of cycles to execute dummy() ISR = 9

    9 = 1 [dummy()] + 2 [Push/Pop] + 3 [Ack ISR] + 3 [call dummy()]

    9*12.5ns = 112.5ns

2) Measured number of cycles to execute dummy() ISR = 59

    59*12.5ns = 737.5ns

E) Analysis and Discussion

1) Range = [4095:0] or 3.3v to 0v.

   Resolution = 0.8mv is smallest change detectable.

   Precision = 2^12 or 4096 unique values.

2) An ADC conversion can be trigger by software, timers, the analog comparators, pwm, and gpio. I used software and timer triggered. Software triggered is used for the adc command in the interpreter, and timer triggered is used for batch adc collection.

3) I measure the time the ISR took to run by having the main WaitForInterrupts(), then recording the missing time from the timer once the interrupt returned. This method seems optimal as you have an exact point for when the ISR was triggered, which the other methods lack.

4) 737.5ns/9 = ~82ns per instruction for this empty ISR.

5) SysTick Range = [0xFFFFFF:0x000000] or [(2^24)-1:0].
   SysTick Resolution = 1 cycle = 12.5ns.
   SysTick Precision = 2^24.