SENG 5831 Software Development for Real-Time Systems (sec 001) Spring 2014

Lab Assignment 1 : Counters, Timers, and Scheduling

Due : Sunday, March 30th at 11:59pm

Submit electronically via email or repo

Introduction

The purpose of this lab is to introduce you to embedded programming and some common hardware systems. The program you will produce will take in user input and blink LEDs at a user-specified frequency using interrupts and timers. You will be provided with the entire framework of the system, some subroutines, and some code fragments to get you started. Consider each blinking LED to be a different task that needs to be scheduled. Notice how each task is scheduled in a different way. The experiments that you will run once your code is functional will demonstrate the advantages and disadvantages of these various "scheduling" methods.

The Program

You will write a program to blink the three separate LEDs at various user-defined frequencies using 4 different methods. A toggle counter will be used to keep track of the number of toggles for each color.

1. Using WCET static or dynamic analysis, determine the number of iterations required in a for-loop to occupy the CPU for 10ms. Use this loop to blink the red LED at 1HZ (i.e. a period of 1000ms).

2. Create a software timer (~~16-~~8-bit) with 1ms resolution, then blink the red LED inside a cyclic executive at a user-specified rate using your software timer. Essentially, the ISR is releasing the red LED task.

3. Create another software timer (~~8-~~16-bit) with 100ms resolution (10Hz), and blink the yellow LED inside the ISR for the timer interrupt. In this

case, the system is being polled at a specific frequency to determine the readiness of a task.

4. Create a Compare Match Interrupt with a frequency equal to the user-specified frequency for blinking the green LED (use timer1). Generate a PWM pulse on OC1A (aka Port D, pin 5) to toggle green.

The LEDs should be connected to the following port pins:

Green : Port D, pin 5. Look on the bottom of the board for the SPWM port.
Yellow: Port A, pin 0. This could be any pin, but chose this for consistency across all your projects.
Red: Port A, pin 2. This could be any pin, but chose this for consistency.

The above will be controlled with a simple serial-based user interface. The communication between the PC and the microcontroller is handled using interrupts and buffers. The buffer is polled inside of the cyclic executive to check for input. This is an event-triggered task, unlike the blinking of LEDs, which are time-triggered (although you could argue that timer interrupts are events). This code has been provided for you. The menu options are as follows:

{Z/z} <color>: Zero the counter for LED <color>

{P/p} <color>: Print the coutner for LED <color>

{T/t} <color> <int> : Toggle LED <color> every <int> ms.

<int> = {0, 100, 200, ... }

<color> = {R, r, G, g, Y, y, A, a}

Examples:

t R 250 : the red LED should toggle at a frequency of 4Hz.

Ta 2000 : all LEDs should toggle at a frequency of .5Hz.

t Y 0   : this turns the yellow LED off

Zr : zero the toggle counter for the red LED

Experiments and Report (to hand in)

Run a series of experiments as described below and answer the following questions. For each experiment,

- Zero all toggle counters (>za).
- Toggle the LEDs for approximately 1 minute.
- Record the number of toggles for all LEDs (>pa).

1. Use your original version of toggling the red LED that uses for-loops. Toggle all 3 at 1Hz. (Do not type in any menu options while you are toggling until the 1 minute is up). How good was your WCET analysis of the for loop? If it is very far off, adjust it. Why did I not want you to use the menu while running the experiment?

2. Use your software timer to toggle the red LED. Toggle all 3 at 1Hz. Simply observe the final toggle count. All should be about 60 (maybe the red is off by 1). If this is not the case, you probably set something up wrong, and you should fix it.

3. Set all LEDs to toggle at 2Hz (500ms). Place a 90ms busy-wait for-loop into the ISR for the green LED. Toggle for 1 minute and record results. Now place a 90ms busy-wait for-loop into the ISR for the yellow LED. Toggle for 1 minute and record results. What did you observe? Did the busy-wait disrupt any of the LEDs? Explain your results.

4. Repeat #3, except use a 110ms busy-wait. You probably won't be able to use the menu functions. If not, report that, and discuss what you observed from the blinking. Explain your results.

5. Repeat #3, except use a 510ms busy-wait. Explain your results.

6. Repeat #5 (i.e. 2Hz toggle with 510ms busy-wait), except place an sei() at the top of the ISR with the for-loop in it. Explain your results.