CSCE 236 Embedded Systems, Spring 2014
Homework 4

Started: Wednesday, Feburary 26, 2014
Due: Beginning of class Friday, March 7, 2014

Instructions: This homework is an individual assignment, collaboration is not allowed. If you discuss any problems with others, please note this on the assignment as described in the syllabus. Also note any materials outside of lecture notes, course textbooks, and datasheets that you used. Show your work and describe your reasoning to get partial credit if your solution is incorrect. This homework is due on the date listed above before the start of class.

Name:

Problem 1 (5pts)**.** *(To be completed at end of assignment) Approximately how much time did the total assignment take? Which problem took longest and how much time did it take?*

Problem 2. *Timers and PWM.* **For this problem assume a CPU frequency of 12MHz.**

a) (5pts)**.** *Give the C code to configure the registers (e.g.* `TCCR0A`*, etc.) to set the 8-bit* `Timer0` *in Fast PWM mode with a frequency as close to 1KHz as possible. Comment each line of code to indicate how you are configuring it. Remember to assume a 12MHz clock frequency.* **Hint:** *You should try all of the possible clock prescalers to determine which gives you the best value.*

b) (5pts)**.** *What is the actual frequency that the timer will run at?*

c) (5pts)**.** *How would you configure it to a 25% duty cycle (on 25% of the time)? Give the C code setting the Atmel registers to do this.*

d) (5pts)**.** *If you used* `Timer1` *instead, could you get closer to an actual frequency of 1KHz? Configure the registers for* `Timer1` *for 1KHz and make sure to show and comment your code. What is the actual frequency you achieve with your configuration?*

e) (5pts)**.** *What would the frequency be if you switched from Fast PWM to Phase Correct PWM mode without changing other settings?*

f) (5pts)**.** *Describe the differences between Fast PWM and Phase Correct PWM mode and why you may want to use Phase Correct PWM in some circumstances. You may want to include a picture of the signals to help with your description.*

Problem 3. *Analog to Digital Converters (ADC)*

a) (5pts)**.** *On the Arduino, if the ADC reports a value of 127, what is the voltage on the ADC pin?*

b) (5pts)**.** *Write the Arduion C code to setup and then read the analog value on pin* `A1`*.*

c) (5pts)**.** *How could you use the ADC on the Arduino to measure a battery that has a maximum voltage of 12V? What is the resolution that you can measure it at?* **Hint:** *Use a voltage divider.*

d) (5pts)**.** *On a different processor (not an Arduino) with a 14-bit ADC and a 3.3V reference voltage, what is the voltage resolution?*

e) (5pts)**.** *What is the maximum number of conversions per second you can perform on the Atmel ADC with a 20MHz clock while still maintaining full accuracy? Explain your answer.* **Hint:** *Make sure to take into account the ADC prescaler.*

Problem 4. *Debugging: For this problem you should download the code from the course website for this assignment. This code is from a developer who was trying to keep his job safe by writing very hard to understand code. Unfortunately, the author died in a hang gliding accident (which he as doing while he should have been at work!). Now we are left with code that is nearly impossible to understand, but is critical to our company. Your job is to use the debugging techniques we have learned about in class to figure out how to use this code.* **You do not need to turn in your code online for this assignment.**

a) (10pts)**. Before** *modifying any of the code, you should compile the program* **as is** *and determine how much memory and flash the program is using using the* `avr-objdump -d -t -h -S file.cpp.elf` *(replace* `file.cpp.elf` *with the proper filename). How much RAM and flash are being used? Make sure to explain how you got this and include relevant lines from your objdump.*

b) (10pts)**.** *In order to start the program the functions* `startOne()`*,* `startTwo()`*,* `startThree()`*, and* `startFour()` *must all be called. Unfortunately, these must be called in a particular order and the proper order is unknown (and is not one, two, three, four). Further, it seems that calling them in the wrong order will cause the Arduino to freeze* **and** *some of these function calls mess up the serial printing. Use the debugging techniques discussed in class to figure out the proper order to call these functions. What is the order and briefly describe how you figured it out.*

c) (10pts). *There is another function* `setMem(char i)` *that takes a character and writes it to a particular location in memory. After much examiniation, we were able to determine that the value passed to* `setMem` *is being stored at a memory location at or greater than* $0x40000800$. *Determine which memory locations the first character of your first name and first character of your last name are stored. Give the memory location (in hex), the initials you used (case sensitive), and briefly describe how you figured this out.* **Hint:** `setMem(char i)` *stores it as an offset in an integer array starting at location* $0x40000800$*, so you can access it as* `*((volatile int *)0x40000800)`.

d) (10pts). *There is a final function,* `runLoop()`*, that should be called from the main loop function. This performs the key computation for our system whenever the button is pressed (attached to Arduino pin 5*[1]*). Time how long this function takes when the button is pressed and when it is not pressed. Answer in milliseconds and briefly describe how you figured this out.*

[1]Fortunately the developer left a properly wired board behind.
Do not forget to fill in the amount of time you spent on this assignment in Question 1.