CSCE 236 Embedded Systems, Spring 2012
Homework 3

Started: Thursday, February 2, 2012
Due: Beginning of class Thursday, February 9, 2012

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.

Instructor sign off: Where an instructor sign-off is specified you must show the instructor or TA the functional code running on your Arduino *before the due date*. Look at the course website for posted office hours where you can be checked off. **Plan ahead** and email the instructor or TA to make alternative arrangements if none of these times work, but do not leave this until the last moment. For these sign offs please bring electronic or printed versions of your code **and** the assignment sheet for the instructor to sign.

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. *Timing operations. You may want to refer to Section 5 of Lab 1 for parts of this problem.*

a) (5pts). *How long does each iteration of a basic for loop, shown below take? What about when the loop counter is a* `uint16_t` *and* `uint32_t`*? Answer in micro seconds.*

```
for(uint8_t i = 0; i < 255; i++){
  asm volatile("nop");
}
```

b) (5pts). *How many CPU clock cycles does the each iteration of the loop take for* `uint8_t`, `uint16_t`*, and* `uint32_t` *types?*

c) (5pts). *Now insert a multiplication command of a variable times an integer constant into the body of this loop for each of the three types. Note, you may need to declare your variable as* `volatile` *so the compiler does not optimize it out. How many clock cycles does a multiplication take based on your measurements? Include the relevant line of code here and indicate if this time is just for the multiplication or if other operations are included in this timing (e.g. memory operations).*

d) (5pts). *Now repeat the above, except multiply by a floating point number (e.g. 3.141) instead of an integer value. How many clock cycles does this take?*

e) (5pts). *Examine the assembly code generated for the above code using the* `avr-objdump` *command discussed in class. What assembly instructions or functions are actually called to perform the two different types of multiplication?*

Problem 3. *Digital Signal Processors (DSP). In class that we discussed DSPs and one of the common tasks implemented on them is a finite impulse response (FIR) filter which is defined as* $y(n) = \sum_{i=0}^{N-1} a_i x_i$ *for an input* x_i*, weight* a_i*, and filter length* N*.*

a) (5pts). *How many addition and multiplication operations are required to implement this filter? Answer in terms of* N*.*

b) (5pts). *If you store the previous* $y(n)$*, how can you reduce the overall number of operations? How many operations do you need per update?*

c) (5pts). *Most DSP processors have a dedicated multiply-and-add instruction. For a FIR filter, how many instructions are required for each new reading when implemented on a DSP with a dedicated multiply-and-add instruction.*

Problem 4. *Schematics. You may want to refer to Section 5 of Lab 1 for parts of this problem.*

a) (5pts). *In Lab 1, Figure 1, the circuit for connecting the LEDs is shown. The LEDs "drop" a fixed voltage and the resistors serve to limit the amount of current that flows through the LEDs. The drops for each of these are RED 2.0V, GREEN 3.2V, and BLUE 3.2V*[1]*. How much current flows through each of these resistors (and therefore the same amount flows through the corresponding LED, which determines the brightness)?*

b) (5pts). *How many LEDs could be connected and controlled by a single pin on the Arduino (hint: look at the Atmel datasheet).*

c) (5pts). *If nothing else is using any current, how many LEDs could be controlled by processor?*

d) (5pts). *Draw a schematic showing how you would connect 3 LEDs to a single output pin while keeping the brightness of each individual LED the same as in the original case.*

[1] As an example, for the BLUE LED, which drops 3.2V the voltage across the corresponding resistor will be 1.8V (since $1.8V + 3.2V = 5.0V$).

Problem 5. *Input*

a) (5pts). *In Lab 1, you configured the button by using a pullup resistor. However, as you learned in class, the input pins also have the ability to activate an internal pullup resistor. Write the* `C` *code below to activate the internal pullup resistor of pin* `PC2`*.* **Instructor sign off required:** *You should also implement this and show the instructor the functionality of using your button in Problem 6 without the resistor.*

b) (5pts). *Using the internal pullup means you can remove resistor R4 in Figure 3, Lab 1. What happens if you do not have this resistor and do not activate the internal pullup?*

Problem 6. *For this problem you should complete the sections in* `morse.c` *where STUDENT CODE is indicated. To do this problem, you will need to include* `morse.c` *and* `morse.h` *in your sketch (use the menu* `Sketch->Add File`*. To call functions from* `morse.c`*, you will need to put* `#include "morse.h"` *in your main sketch file.*

Complete the `morse.c` *code so you will be able to send Morse code blinks. You should make sure that you are able to specify any of* `LED_RED`*,* `LED_GREEN`*, or* `LED_BLUE` *as the LED to output blinks or any combination of them (e.g.* `LED_BLUE | LED_RED`*). I would recommend creating helper functions that turn on or off LEDs so if you switch the pin that controls the LEDs, you only have to change code in one or two places. All of the code described here should be in a single program that runs at the same time.*

a) (5pts). *In* `morse.c`*, the Morse blink pattern (dots and dashes) for each character are stored in a single byte. Read the code and describe how this is done and what the meaning of each bit is.*

b) (5pts). *Now write the* `C` *code to turn on and off the LEDs by setting the registers. What pins did you connect the LEDs to? How did you configure these pins as output? How do you turn the LED on and off? Make sure to include the relevant code here.*

c) (5pts). *Write code that will blink "Hi W" when the board starts with "Hi" blinked with the red LED and "W" blinked using the red and blue together.* **Instructor sign off required, no written answer needed.**

d) (10pts). *Now implement code that will output in Morse code "less" (in green) if the button is pressed for less than one second and "more" (in red and green together) if it is pressed for more than one second. Describe how you implemented this and watch out for bouncy buttons.* **Instructor sign off required.**

Problem 7. For this homework you must turn in your code by visiting `http://cse.unl.edu/~cse236/handin/`. *If you have not used cse handin previously, you will need to register using the registration link on the right of the webpage. Failing to turn in your code will result in up to a 20 point penalty on this assignment. Points may also be deducted for coding errors, poor style, or commenting.*

Do not forget to fill in the amount of time you spent on this assignment in Question 1.