CSCE 236 Embedded Systems, Spring 2014
Homework 2

Started: Monday, January 27, 2014
Due: Beginning of class Weds, February 5, 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. *Arduino Setup and Programming (*Note you only need the Arduino for this assignment, you will not need the breadboard or any of the other components.*)*
Instructor sign off: *Before the start of class on the due date, you must show the instructor or TA the functional code running on your Arduino for this part of the assignment. You can get signed off at any of the office hours of the instructor or TAs (see online for posted times).* **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. Failing to get signed off before the due date will result in a zero for the programming portion of this assignment.*
You must also turn in your code by visiting `http://cse.unl.edu/~cse236/handin/`**.** *Failing to electronically turn in your code will result in a 10 point penalty on this assignment. Points may also be deducted for coding errors, poor style, or poor commenting.* Note, you do not need to turn in a printout of your code for this assignment.

a) (10pts)**.** *For this problem, you should write a program that will monitor the keyboard from the* **Serial Monitor** *(located in the tools menu) and will turn on the LED if you press the 'l' key, off if you press the 'o' key, and will blink the number of times that the LED has been toggled (turned from off to on) when the 'b' key is pressed. To get started with this, look at the example code (the communication section is a good place to start) that comes with the Arduino sketch environment and their online help. Do not make the code any more complex than it needs to be (e.g. do not use serialEvent()), but also remember that there should be appropriate comments in your code.*
Sign off:

b) (5pts)**.** *What voltage does the main processor on the Arduino run at? If it draws 15mA, what is the power usage in Watts?*

c) (5pts)**.** *What clock speed does the Arduino run at? What is the maximum clock speed for the ATmega328 used on the Arduino if there is no external oscillator?*

Problem 3. *Operation Timing*

It is often useful to time how long a particular operation takes on an embedded system to ensure proper functionality. Most operations happen very quickly, but by executing the same operation many times (by putting it in a loop) you can determine how long an operation takes by simply using an LED and stopwatch. The problem is that the loop itself will incur some overhead. To further complicate this problem on an 8-bit processor, such as our Atmel, this overhead will be dependent on whether the loop counter is 8-, 16-, or 32-bits.

In this section, you should determine how long each iteration of a for loop takes when using an 8-, 16-, and 32-bit unsigned integer as the loop counter (`uint8_t`*,* `uint16_t`*, and* `uint32_t`*, respectively). To do this, turn on an LED before entering the for loop and then turn it off once the for loop is complete. The compiler will optimize out any code that does not do anything, so we need to add an instruction that prevents this. You can do this by adding an assembly command (which the compiler will not remove) that does nothing. The easiest is to use the "nop" (no operation) command. To do this in* `c` *do (for the 8-bit version):*

```
//Declare this globally to prevent compiler optimizations
//This is especially important for 32 bit types
uint8_t cntr8 = 0xff;

for(cntr8 = 0; cntr8 < 255; cntr8++){
  asm volatile("nop");
}
```

You should then experiment until you come up with values that make the LED stay on for somewhere between 5 and 15 seconds. This is sufficiently long that you can get a good estimate of how long each iteration takes by timing this overall time with a stopwatch. Note that for the 8-bit and 16-bit variables you may need to nest loops to delay for a sufficiently long period. In this case you can ignore any overhead that the nesting may cause. In addition to enabling you to time operation overhead, this will also give you an alternative, albeit less accurate, way to delay for a specific period of time as opposed to using the Arduino `delay(...)` *function.*

Note: *The compiler likes to optimize variables. To avoid problems associated with this, declare your counters gobally and assign them values that can only be represented with the specified type. For instance, for 32 bit counters, declare them as* `uint32_t cntr32 = 0xffffffff;`*. Also, if you get the same values for 16- and 32-bit counters it is likely a compiler optimization causing issues.*

a) (10pts)**.** *Approximately how many clock cycles does an 8-bit, 16-bit, and 32-bit loop iteration as described above take? How did you determine this? Hint: you can do this good enough with just a stop watch and LED as described above.*

b) (10pts)**.** *Show a program that turns on the red LED for 1 second using 8-bit loops, the green for 1 second using a 16-bit loop, and the blue for 1 second using a 32-bit loop (turn each off after it is on so only one is on at a time). Get this program signed off by showing the instructor. Also include a copy of this program in the code you turn in electronically.*

Sign off:

Problem 4. *Digital I/O. Refer to the Arduino Uno R3 schematic posted on the course website for this question. For these questions, make sure to give* `C` *code where appropriate.*

a) (5 pts)**.** *On the Arduino, how would you configure pin* `10` *on the output headers to be an input and how would determine if the pin is set* `HIGH`*? For this question, use the Arduino commands.*

b) (5 pts)**.** *Which pin on the Atmel does this pin correspond to?*

c) (5 pts)**.** *Which* registers *(there are three) control this pin and how would you set this pin to an input and determine if it is set to high? By register, I mean those you find in the datasheet (e.g.* `PORTD`*).*

d) (5 pts)**.** *How would you set this pin to an output and set it low? Give both the Arduino-version and the Atmel register version.*

Problem 5. *Resistors*

a) (5pts)**.** *If the current flowing through R2 is 0.5mA, what is the value of R4?*

b) (5pts)**.** *In the schematic above (and the specified current), what is the voltage at points A, B, and C?*

c) (5pts)**.** *What is the equivalent series resistance of all of these resistors? What is the equivalent resistance if they were all placed in parallel instead?*

Problem 6. *ATmega328 Datasheet*

a) (5pts)**.** *What memory address is the* `PINC` *register located at?*

b) (5pts)**.** *How would you set (give the* `C` *code) bits 4 and 5 in the* `DDRB` *register to the lowest two bits in the variable* `var`*? Make sure you set these bits at the same time and do so in a single line of code.*

c) (5pts)**.** *Which assembly instructions take the most clock cycles to complete? How many clock cycles do they take?*

d) (5pts)**.** *What is the starting memory address for the Internal SRAM on the Atmel on the Arduino??*

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