

## CSC211 Laboratory 7

### Background Information

#### AVR microcontroller

We have already said quite a bit about this in class. If you would like to read more, you can take a look at the materials I have posted under “Course Resources” on the course website.

#### STK500 development board

This board provides a working environment for embedded systems (ie, microcontroller) code development. Yes, it is used in the “real world” for this purpose.

Note the following features:

- a row of 8 (small red) push-button switches for input
- a row of 8 (very small, but very bright) LEDs for output, located next to each switch
- ATMEGA8515L microcontroller chip (about 2 x 0.5)
- power switch: in a corner on the far side of the board from the push-button switches
- power jack: next to the power switch
- serial ports: we’ll use the one labelled RS232 CTRL to program the AVR
- User Guide

#### Static electricity:

The chips used in this lab are based on CMOS technology, and as such, they are more sensitive to small static shocks than are the TTL logic chips we used previously. Even a shock that you don’t feel can damage the chip, so it is advisable to avoid touching the AVR chip with your fingers. That said, I sometimes forget, and I have not experienced any difficulties yet.

Please also be careful not to drop anything metal (ie, rings or paperclips) onto the development board as this could also damage the board or the chip by causing a short-circuit.

### Laboratory Exercises

1. Take a good look at the development board to find all the feature described above. Also note the three ribbon cables:
  - One connects a set of pins labeled PORTB with a header labeled LEDS. This connection causes any output you write through PortB of the AVR to be sent to the board’s LEDs.
  - One connects a set of pins labeled PORTD with a header labeled SWITCHES. This connection causes the signals created by pressing the board’s push-button switches to be sent to PortD of the AVR.
  - One connects a header labeled ISP6PIN to another header labeled SPROG3. This cable sends data from the serial cable to the correct AVR pins during ISP (in-system) programming.
2. Download the following two files from the course website (under Course Resources): **Makefile**, and **m8515def.inc**. Read through the makefile to see what it does. Skim through the definitions file too, if you like. Place these files in the directory you will use for this laboratory.

3. Write an assembly program called `blink.asm` that blinks the 8 LEDs attached to PortB of the AVR on and off. To do this, you will need to alternate the output signal sent to PortB between high and low (1 and 0) on each output pin. However, if you do this too quickly, the LED will appear to be on all the time. Thus, your program should include delays that cause each output value to be held steady for a considerable length of time before changing to the next value. You may determine the blink speed you want, but I suggest you start with a delay loop that runs for approximately  $2^{17}$  iterations. How can you make that happen if the counter value used to control your loop is only an 8-bit value?

When you are ready to test your program on the AVR:

- Check that the Makefile includes the following line near the top: `FILE=blink`  
This indicates that your source file is named `blink.asm`. (Note that for future programs you will want to modify the makefile to indicate the correct file name for each program.)
  - Assemble your source program by typing `make` at the bash command line. *Be sure to look at the output avra provides. For example, look for the words “assembly complete with no errors” toward the bottom of the output.*
  - Plug the serial cable into the STK500 serial port labeled `RS232 CTRL`. Plug the power cord into the STK500 power jack and turn on the STK500 power switch.
  - Program the AVR (i.e., copy the executable code to the AVR’s flash memory) by typing “`make program`” at the bash command line. *Be sure to look at the output avrdude provides. For example, look for the words “verifying... flash verified” toward the bottom of the output.* If all works correctly, you should see the LEDs flash 3 or 4 times very quickly (indicating that the AVR is being programmed), and then the AVR should start running your program and flashing the LEDs at a different speed.
4. Take a look at the file named `blink.hex` that is created by the assembler. It contains the machine instructions of your program, represented in hexadecimal characters.
  5. Modify your program to produce a different pattern of blinking lights. For example, you could have half the LEDs on and the other half off at all times. You could also have only one LED on at any time, but cause the light to travel down the row of LEDs.
  6. **For those with extra time:**  
Read or skim through the STK500 User Guide.