Lab 1

Name: Dylan Christopherson

CEE-345 Microprocessor System Design

Introduction:

The goal of this lab was to introduce us to Assembly, C, and the STK-600 board. We were supposed to design and implement assembly code that would blink LEDs in different patterns. We were then supposed to complete the same process in C. The machine code that the board is flashed can only be so large. Assembly code is more difficult to write, however, it translates to a small hex file. C on the other hand, is easier to write, but it ends in a larger hex file. Learning both is important so we can be dynamic as we can as engineers when using this board.

LDI – Stands for Load Immediate. This loads an 8 bit constant directly to register 16 to 31.

SPL – Stack Pointer Low

SPH – Stack Pointer High

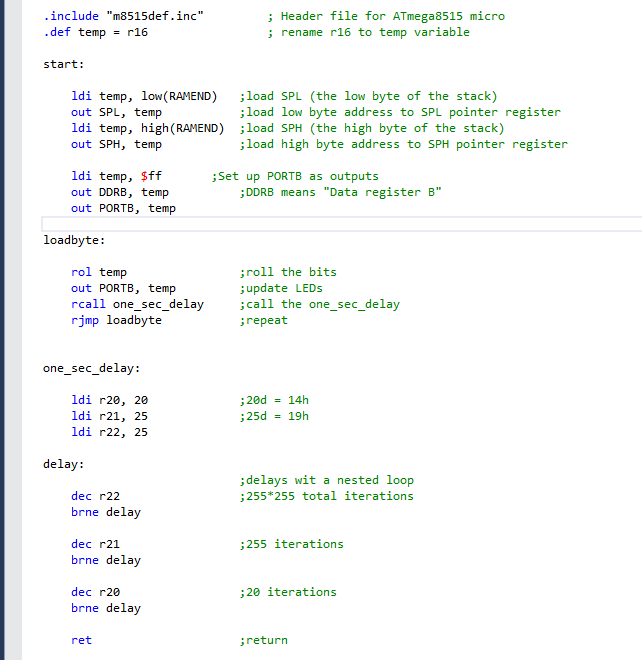
RAMEND – A label that represents the last memory address in SRAM. It’s a 16 bit word so we use the functions to split 16 bits into 8 bits so it can be handled.

Using low() and high(), we can return the low byte and high byte respectively of a 16-bit word

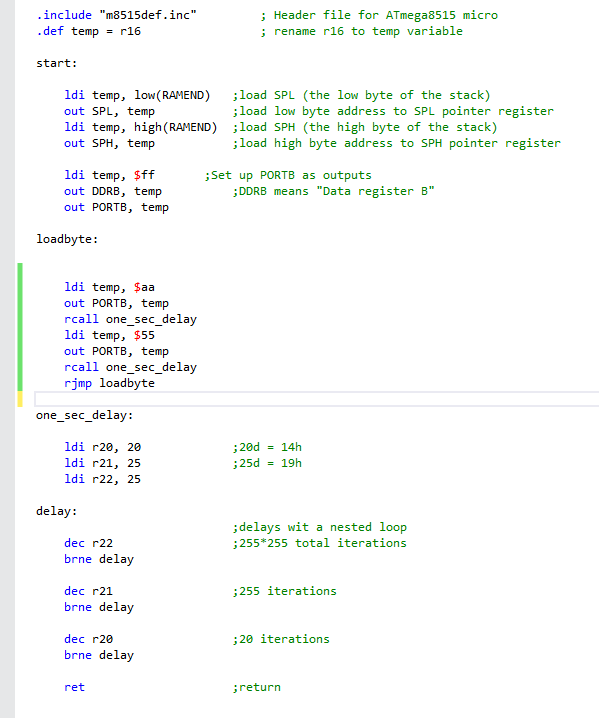
rol (Rotate left): The MSB is rotated to the carry flag, the carry flag is rotated to the lsb, all other bits are shifted left.

Part 1: Our goal for part 1 was to toggle multiple LEDs on and off on the STK-600 board with a blinking pattern. This first section was written in assembly.

* Loading registers, setting up the ports, outputting value to ports, creating LED pattern, creating a delay, and creating iterations.



Part 2: This section consisted of modifying code from Part 1. We wanted to generates a new LED blinking pattern. LED flashing at an alternate pattern with two LEDs on at a time. This was also written in Assembly.

* Loading registers, creating two different LED patterns, creating a delay for both, setting up the ports, and setting up the iterations.

Part 3: Finally, we had to write Part 1 using C.

Pattern starts initially as: 00000001

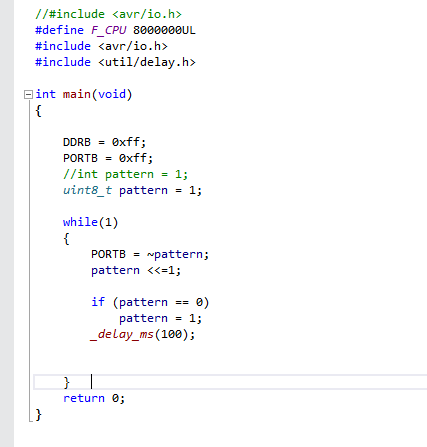
The inverse of pattern gets sent to PORTB: 11111110 (1 means the LED is off)

Then patter is shifted one bit left: 00000010

Then the inverse is taken and the pattern is continued.

Once pattern gets to 00000000, pattern is reset to 00000001.

* Clock speed for controller, creating LED pattern, creating a time delay, and goes through the sequence 8 times.



There were some struggles from this lab. First off, this was one of my first opportunities to dive into assembly. This is also the been the first time in a couple of years I’ve coded in C. So, I think I’m going to enjoy diving into, and learning/refamiliarizing myself with these languages and their uses. Although this lab was just an introduction, I look forward to where this class is headed.