**Lab 5 7-Segment LED**

**Joseph Tomalewski**

**Date: 10/18/2022**

**Purpose of Lab**

Utilize AVR assembly and a combination of a C function to light up the 4-digit 7 segment LED light that I connect to the breadboard and use pins 2-12 to output the machine code. For this lab there is C code to demonstrate on the LED what I must do in assembly code. Which is to flash numbers on certain digits in a certain order. I will be using the data tables provided to me to utilize a list to compare and indirectly address my registers with.

**Description of Solution(s)**

To program the solution for the morse coder, I would break up my assembly code into 3 major parts. A setup ports function, select digit function, and a driver code function, (Such as the main loop). I would first program to make all the pins of portB and portD output, since I will not be using input, yet I will just be sending data to the 7-segment LED.

Next, to program the select digit function I would have it be the first function to be called as it will select the correct digit on the LED segment to output the corresponding number on that digit. This function would include a loop that would check the position of the data in the digit\_table. If the correct position is found corresponding to the data\_table. I will use that value to select the digit on the segment and output it to portB.

Lastly, I will have this driver code function to run first, which is the show\_number() function. This function will run stack macro first, then the select digit function, to finally return to loading the X and Y registers with the PortB, and PortD data tables. We would first check the position of the number variable that was passed from the C function and with this position value we would increment according with the data tables. Finally selecting the correct segments to output to portB and portD. PortB in this case will focus on if the number needs a crossbar which is segment G. If the position selected does not need a crossbar it will output to portB without setting the portB.0 bit.

**Test Results**

For this lab I noticed that I would have to program the same thing that is demonstrated in C which made me have a better understanding of what I should need to program and how, I would first program the first part in assembly which is setting the ports and selecting the digit, once I was able to finish programming that part, which was relatively easy in my opinion. I would then move on to programming the driver code that would output the corresponding number to the selected digit that were both provided by the C code. The results end up having my 7-segment LED output 0 on digit 0, 2 on digit 1, 4 on digit 2, 6 on digit 3, 8 on digit 0, and then output the odd numbers from 0 – 9. This program will output the even numbers then odds from 0-9, moving ahead a digit selected for each number.

**Answers to Questions**

None

**Discussion**

In my opinion this lab was not at all too difficult other than making sure I have the correct ports and pins selected and connected properly. I could imagine I would have a way more difficult time programming without the videos provided as they would show a foundation of how I needed to program in my assembly file. I enjoyed this lab; I feel that it would be a great addition to add this program to another real-life sensor and have them interact with each other.

**Contribution to Teamwork**

This was a solo lab.

**References**

For this lab I mainly used the resources provided in the lab5 assignment. I realized that the videos provided in this lab were extremely helpful in programming the Arduino. I have also used a stack macro to push and pop registers onto and off the stack. This algorithm was very useful as it seems slightly tedious and complicated to make my own.