Hongyu Zeng

Professor Beichel

Lab2 Report

1. Introduction

The goal is to implement a 7-segment display with two buttons using Arduino UNO programed in AVR Assembly. One button increases the digit, and one decreases the digit. The number ranges from 0 to 19 where the carry after 10 is shown by the decimal point.

1. Schematic

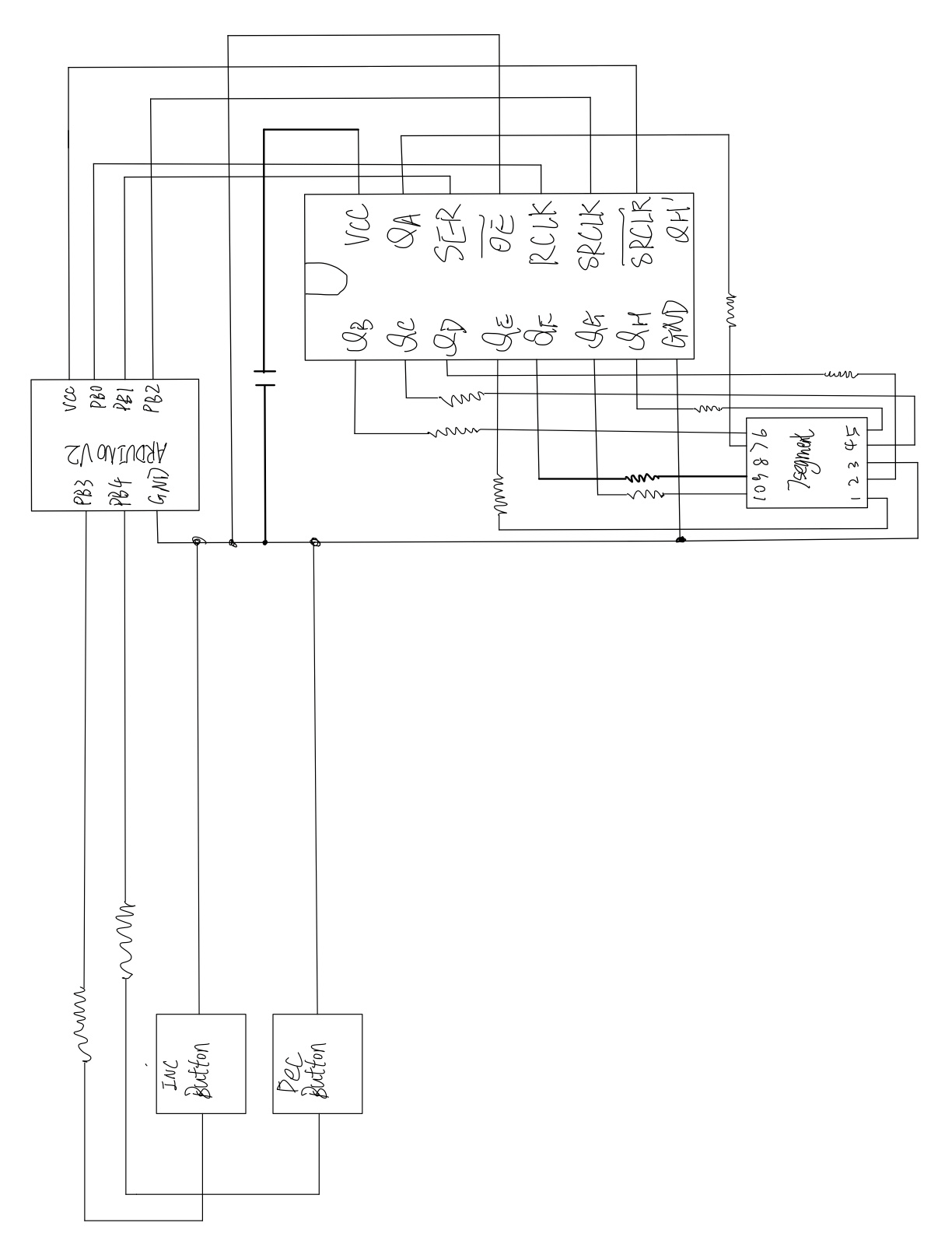
Input: PB3 PB4

Output: PB0 PB1 PB2

Resistors: 1K ohm

Capacitor: 10^4 pF

VCC = 5 V



1. Discussion

Things verified

|  |  |
| --- | --- |
| Digit increased/decreased when correponding button got pressed | check |
| Digit only change once when long pressed | check |
| Upper limit is 19 | check |
| Lower limit is 0 | check |

We choose 1k resistor as the maximum forward continuous current for 7-segment display is 30mA (Vcc = 5V).

The circuit has a clock speed of 16MHz. The idea is to have a main loop looping infinitely. Inside the main loop we have button check that implements software debouncing, which checks the number of lows (in our case, button pressed) 9 times with an interval of 10ms. Then it compares if the number of lows is higher or equal than 5, which suggest that the button is pressed. It then checks if the digit is on the upper limit 19 or lower limit 0 and increase/decrease it if not. Note that in order to make the digit only increase/decrease once for each press (no matter how long you press it, it only changes one time), we added a register R29 storing the state of the button being pressed or not. So when the “maybeincrease” subroutine is called, it checks the previous state of the button and determine whether to increase it or not. When we increase/decrease the digit, we set the state of that button on the R29 to 1 (for increment button is the MSB and for decrement button is the next bit to MSB).

Note that the increment button has higher priority as the increment branch always got check first. Thus if we hold the increment button then press the decrement button, it won’t do anything. But if we hold the decrement button and press increment button, it will go up.

For displaying and updating digit, we first create 20 variables that stores the bit map for each number. The “load\_pattern” checkst the content of the R22 which stores the decimal value of the number and load the corresponding bit map to R30 (our bit map regiester). For displaying the digit, we rotate through carry on the MSB on the R16 (updated by R30) 8 times. Then the corresponding bit map shifts to the shift register.

1. Conclusion

From this lab I learned how to implement loops and if-else in AVR Assembly. I’m also more comfortable with the AVR instructions.

1. Appendix A: Source Code

