**Timer0 Interrupt**

*Final lab project*

**Phillip May,** 1639294

**ABSTRACT**

*In this lab I programmed and wired the PIC16F887 to be a hypothetical traffic tracking device.*

Experiment Performed on **5 December 2017**

Report Submitted on **12 December 2017**



Department of Computer

Engineering Technology

**INTRODUCTION TO MICROPROCESSORS (247-302)**

*Day Yann Fong*

Table of Contents

[Introduction 3](#_Toc500850378)

[Equipement 3](#_Toc500850379)

[Results and Discussion 4](#_Toc500850380)

[Conclusion 6](#_Toc500850381)

[Additional 7](#_Toc500850382)

[References 7](#_Toc500850383)

# Introduction

For the final lab project of introduction to microprocessors I implemented a hypothetical traffic monitoring system using the PIC16F887. The prototype system uses two buttons as hypothetical inputs from sensors and a third button to switch to EEPROM viewing mode in order to prove that it is correctly saving the EEPROM data as the debugger does not work for this.

# Equipement

• PIC16F887

• Pickit3

• 2 100nF capacitors

• 4 LED

• 7 270ohm resistors (current limiting)

•3 10Kohm resistor (pull up resistors)

•3 push button

•Connecting wires

• 3 7 Segment display

# Techinical Details

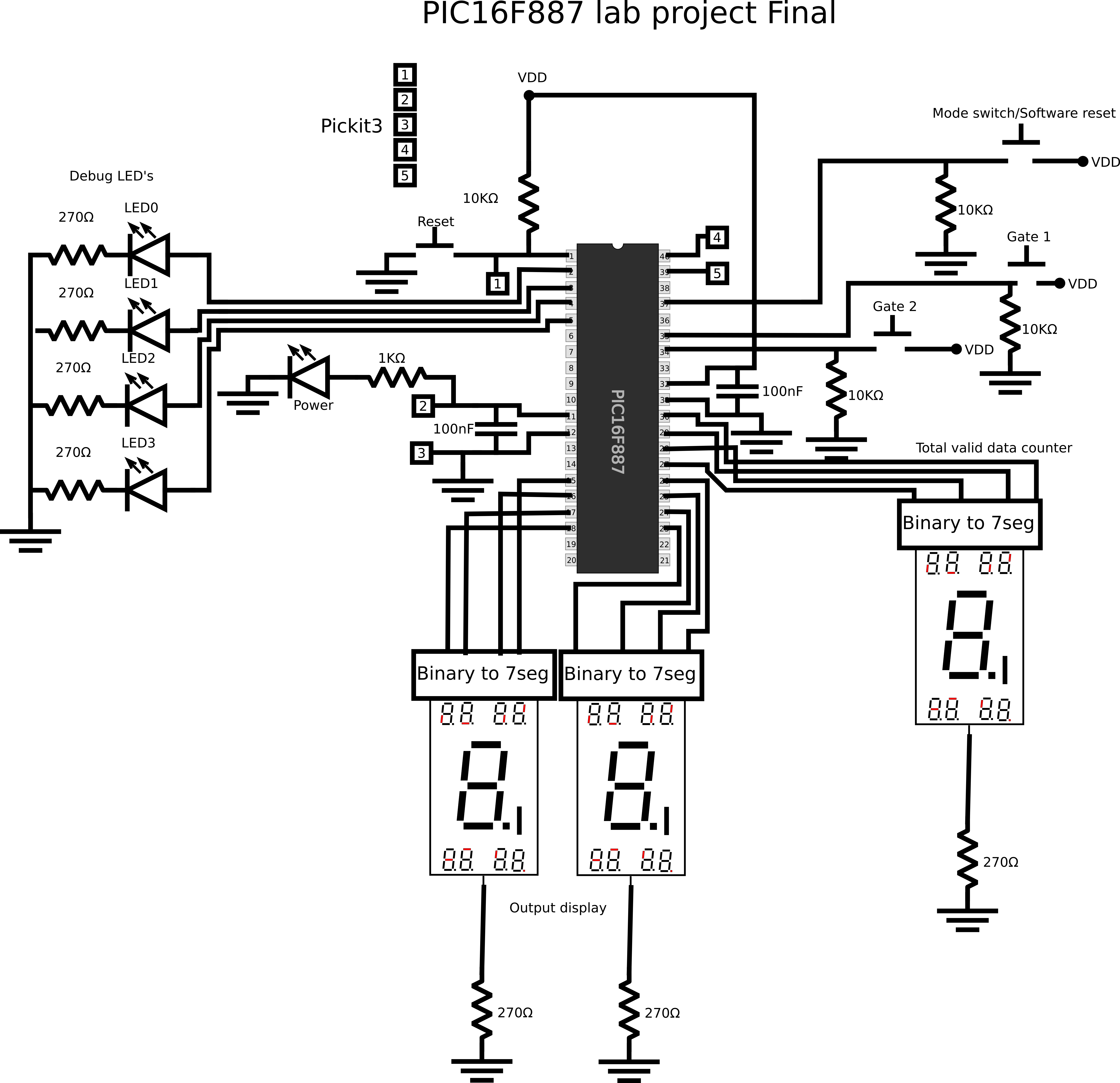
Technical features and design specifications:

Two user inputs that for timing and one for additional features.

The two timing inputs (Gate 1 and 2) are triggered in sequence and the time between the two presses is recorded. If the timer waits for more than 9.9 seconds without response it will go back to idle and wait for a new input. The third input switches to a memory viewer mode that allows you to look at the EEPROM containing previous sequences time.

Detailed schematic and high level flow charts:

Schematic:



The wiring of decoders of 7 segments display are not listed as they were nonstandard and I used both anode and cathode 7 segment displays. A release version would standardize all the parts to cheapest ones obtainable.

High level flowcharts:

Included in the .zip for size reasons

ISR flowchart omits context saving.

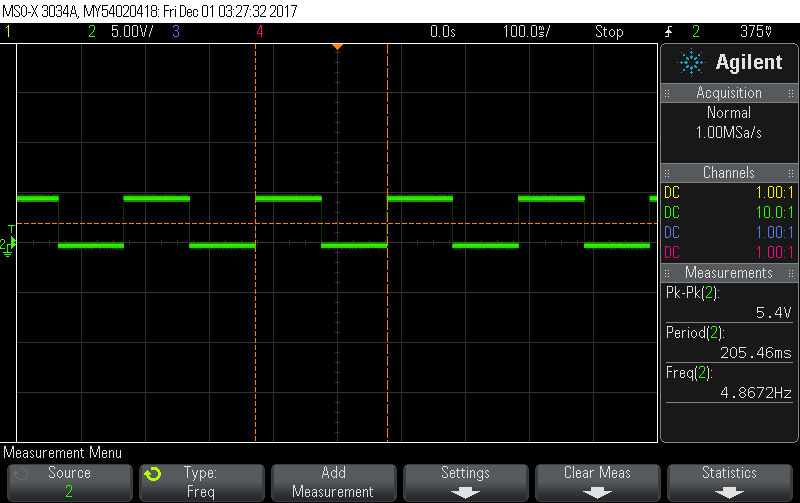
Technical details of firmware implementation

The timing is done via interrupts in order to achieve a reliable way to get very close to 1 second delay. The clock was kept close to default speed in order to have be as reliable as possible. The calculations are as follows:

The prescaler was set to 1:256, the constant was 200 and the clock was set to 2Mhz

(1/2Mhz)\*4\*200\*256=102.4mS

Because I don‘t trust my calculations I also checked it on the oscilloscope to see how accurate it was and it was within a 5% margin of 100mS. This could probably be fine-tuned more.



The counters are reset when the user presses the start button. The three user input buttons are polled in main along with some overflow checks. The 7 segment displays are driven with decoders so those need to be run on external power. EEPROM mode is completely separate from the normal main loop.

# Discussion and Analysis

Problems encountered:

I encountered numerous problems while implementing the system. It could probably be its own report to itself to list literally every single one I had. Ignoring the common problem of writing code that doesn’t work and then fixing it the problems I encountered with their respective solutions are as follows.

* Wiring was quite complex and slow work, further impeded by random internal shorts and having no good way to route wires to physical lack of space for the wires.

Testing often was the best solution to this.

* The bits of portc are hooked up in reverse to the double 7 segment display.

Rewiring was a possible solution, however in the interest of time and my sanity I opted to quickly fix in software as a function.

Debugging was mostly avoided by programing incrementally and fixing each step individually to avoid more sources of error than absolutely needed. That being said I still encountered numerous problem while programming including the debugger reporting false data about what the EEPROM was storing among other issues. I resolved all the issues as they came up and the final version works as intended.

Technical knowledge learned/enhanced:

This was by far the most complex program I have ever single handedly written and to add further complication it was all done in pic assembly. Thanks to this I have an even greater appreciation for flowcharts even when they are just made for self-reference and not for submission. It’s very easy to get lost in what parts of complex programs do and flowcharts simplify this considerably. Besides simply enhancing my knowledge of using general digital IO on the PIC16F887 I also learned about using the EEPROM and about an architecture limitation related to switching interrupts off.

Potential enhancements to project:

Potential enhancements include optimizing the code. Code optimizations would be moving all the user variables out of shared memory so that other programs or other parts of a further improved version of this can use it. The EEPROM viewer mode uses software delay instead of waiting for a button release, which could be fixed. I am sure at least some of my BANKSELs are completely useless and redundant, that could be fixed. There are probably other improvements of which I am unaware of that could also further improve the project. The debug LED’s could be removed from the final release version.

# Conclusion

In conclusion, the project worked as expected. The project presented some challenges however I feel it probably could have been more challenging. Given that we were the one that suggested the project I suggested something I knew I could reasonably do with a normal amount of effort. Had I been given something outside of what I know is possible I would have learned to do things I would have had no knowledge of. I would have done the impossible instead of simply doing the possible. Other details of the project where discussed in other parts of this lab report.

# References

|  |  |
| --- | --- |
| [1] | Microchip Technology Inc., "MPASM Assembler, MPLINK Object Linker, MPLIB Object Librarian User's Guide," Microchip Technology Inc., 2013. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/33014L.pdf. [Accessed 20 9 2017]. |