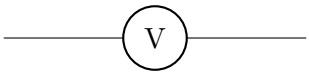
A-Level Electronics Project

Task 2 - Electronic System

Voltmeter



Thomas Boxall

march 2022

BEXHILL COLLEGE: 56635
CANDIDATE NO: 5377

Contents

Chapter 1

Introduction

1.1 Introduction To The Project

I chose this project with the idea of measuring the voltage of batteries up to 5V. The inspiration for this project came from seeing a battery tester and wondering how it works. This project also sounded like a challenge as it is combining lots of the different subsystems I learnt about in the first year of the A level course into one big system. The system will begin when the power to the circuit is turned on; it will work continuously, displaying the voltage connected until the system is powered down. If no voltage is connected, it will display 0V.

1.2 Definition Of A Voltmeter

Before researching the different types of voltmeter and how they worked, I thought it would be useful to find out the definition of a voltmeter. Wikipedia[1] defines a Voltmeter as:

> A voltmeter is an instrument used for measuring electric potential difference between two points in an electric circuit. It is connected in parallel. It usually has a high resistance so that it takes negligible current from the circuit.

[1] https://en.wikipedia.org/wiki/Voltmeter

Chapter 2

System Planning

2.1 Research

Before I start to plan my design, I will need to start by researching existing designs.

2.1.1 Existing Designs

I started by researching existing designs of voltmeter. I learnt that there were a variety of designs, each doing fundamentally the same thing. They would all take an analogue voltage as an input and display a numerical value as the output. The exact method which they used to display differed, this is where the main differences can be found between the designs.

Analogue Voltmeter

These work by having a coil of wire suspended in a strong magnetic field. The current of the signal who's voltage needs to be measured is passed through this wire, the magnetic field and coil interact and the coil rotates, moving the needle which is attached to it. The needle points to a scale which can be read by the operator[1].

Digital Voltmeter

These work by taking the analogue input voltage then converting it to a digital signal then outputting that signal in numerical form. They usually use an integrating ADC. Digital voltmeters' accuracy can be affected by many factors including temperature, input impedance and power supply variation. The input impedance issues can be overcome by having input resistance $> 1M\Omega$. Digital voltmeters also have to be calibrated frequently, using a known voltage source[2].

Multimeter

Multimeters can measure many different values from an electrical circuit. They have a digital voltmeter built into them as one of their options; this works as described above[3].

2.2 Components Of Subsystems

I then began to work out what subsystems I would need to use to convert the input voltage to seven-segment displays. I worked out that first the voltage would need to be converted into a binary number which then could be inputted into a memory lookup table. This can then output BCD which can go into the display subsystem.

[1] https://en.wikipedia.org/wiki/Voltmeter

[2] https://en.wikipedia.org/wiki/Voltmeter

[3] https://www.electronicdesign.com/technologies/test-measurement/article/21146730/multimeter-measurements-explained

2.3 Specification

- The voltmeter will measure in volts and range from 0V to 5V, in 0.1V increments.
- The voltmeter will have a sample rate of 1KHz $\pm 400Hz$ times per second. This will be produced by Schmitt Astable with a frequency of 1KHz.
- The voltmeter is accurate to within $\pm$0.1V.
- The voltmeter takes less than 0.5 seconds to adjust its output once a change in voltage is detected.
- The voltmeter should take a +5V, 0V and -5V (within $\pm$0.5V) input.
- The voltmeter will be easy to use.
- The voltage will output to two 7-segment displays; one for the units and one for the tenths.
- The entire circuit will only require human input to connect the analogue voltage.
- The voltmeter should be as efficient as possible, reducing heat dissipated to the environment.

Chapter 3

Subsystem Design

I began by working out the very simple block diagram of my voltmeter.
Initially, I broke the voltmeter down into 3 systems: an ADC to convert the analogue voltage from the input to a digital signal; a memory lookup table using an EEPROM to reference the voltage and convert from binary to BCD; and a display to show the readout of the voltmeter.

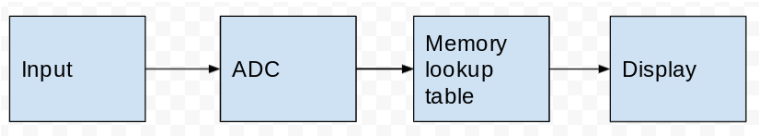


Figure 3.1: The overall simple block diagram of my voltmeter

I then worked out what components I would need to use for each subsystem and how they would be connected together.

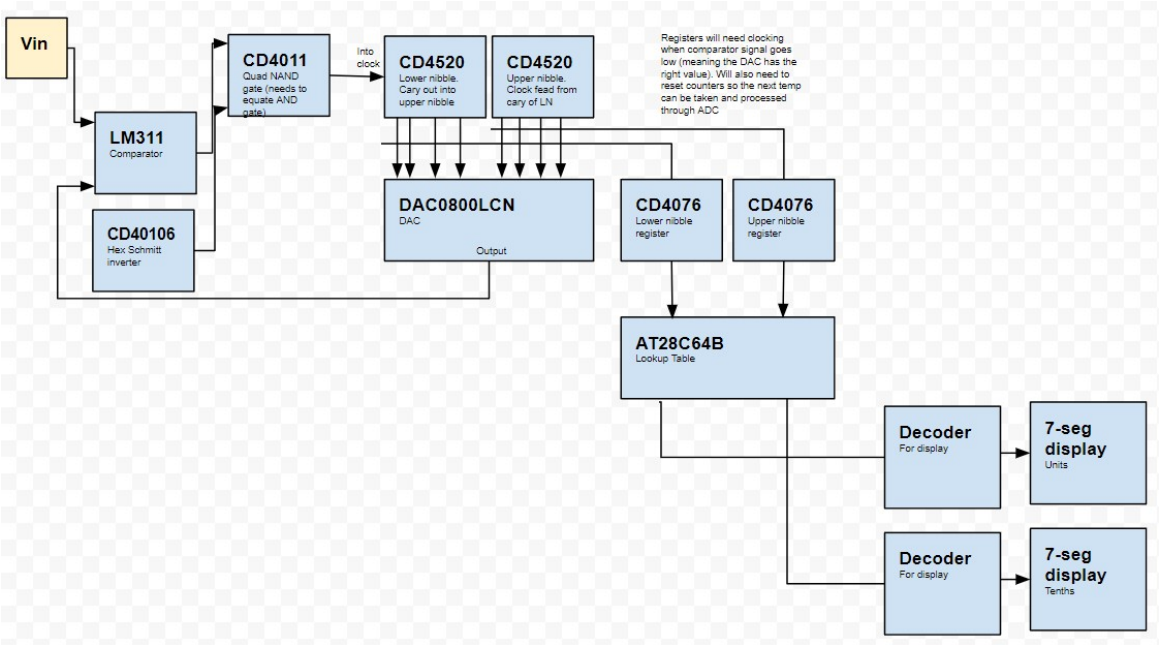


Figure 3.2: Diagram showing how chips interact

NB: Some wires have been omitted from the diagram above for clarity.
Also at this stage, I decided on the colours of wires that I would be using. These can be found in the appendix.

3.1 Clock Subsystem

The clock is a fundamental component of the voltmeter as it is used to trigger the counter within the ADC to count, ramping the voltage. This means that the frequency of the clock has to be very fast, around 1KHz; however if it isn't this fast, the clock will still be within acceptable parameters (±400Hz). I decided that I would use a Schmitt Inverter Astable.

Design

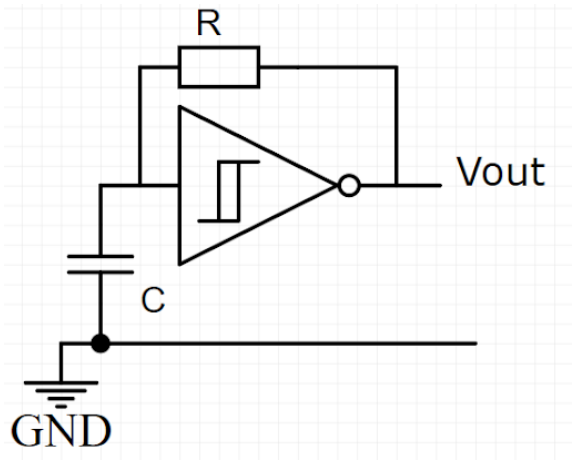


Figure 3.3: Circuit diagram of clock subsystem

I then calculated the required component values using the equation $f = \frac{1}{RC}$
I knew we had capacitors with capacitance $1\mu F$, so I substituted that along with my desired frequency of 1KHz into the equation then rearranged to calculate the value of the resistor.

$$1 \times 10^3 = \frac{1}{R \times (1 \times 10^{-6})}$$

$$R = \frac{1}{1 \times 10^3 \times (1 \times 10^{-6})}$$

$$R = 1000$$

The resultant component values are:
$C = 1\mu F$
$R = 1K\Omega$

Building & Testing

The clock subsystem was the first subsystem that I constructed. I tested this by connecting the output of the clock to the oscilloscope probe and using its digital readout to tell me the frequency. During testing, I found out that the clock's frequency isn't 1KHz, however it is 1.36KHz which is acceptable as it is less than 1.4KHz which is the upper parameter. This is because as long as the clock is relatively fast the functionality of my voltmeter won't be damaged.

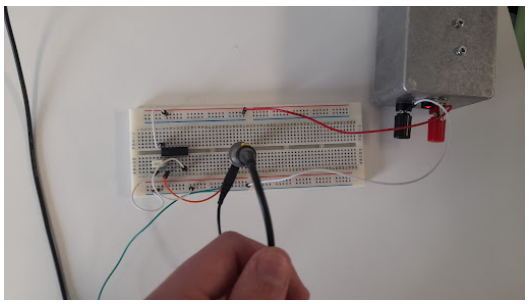


Figure 3.4: Top-down shot of the clock on a breadboard connected to the power supply with an oscilloscope test probe connected to the output of the clock

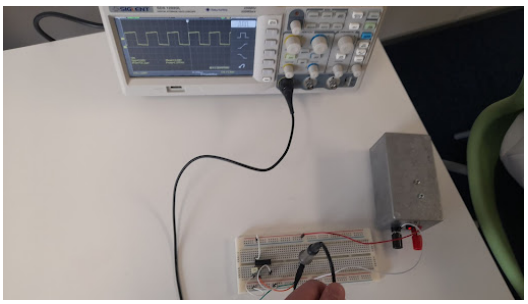


Figure 3.5: View of the oscilloscope and breadboard

Figure 3.6: Output of clock shown on an oscilloscope

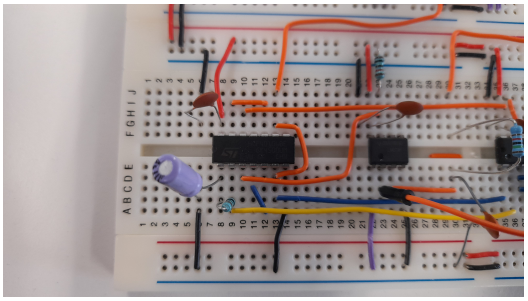


Figure 3.7: Neatened clock subsystem

After building this, I evaluated some of the different subsystems I could have used instead of the Schmitt astable. I could have used a 555 timer astable, however this would have resulted in me having an extra chip on my breadboards, as I would still need to use Schmitt inverters in a delay line later in the building process. Alternatively, I could have used a Crystal Oscillators. The additional accuracy which this provides would not be necessary.
This subsystem is within the acceptable parameters I set in my design and I am happy with it.

3.2 Ramp Analogue to Digital Converter(ADC)

I decided to use a ramp ADC because my output wouldn't need an extremely high sample rate and it uses fewer components than the flash ADC. If I was to use a flash ADC in my project where I need 8-bit outputs, I would need 256 comparators & 256 resistors as well as a priority encoder large enough to handle 256 inputs. Using a ramp ADC means I only need 6 ICs as well as a handful of other components.
The ramp analogue to digital converter is a very complex system with lots of different components and subsystems within it. For this reason, within my report, it will be broken down into the following subsystems (this is also the order in which I constructed them):

- Digital to Analogue converter;
- Counter;
- Comparator;
- Delay Line;
- Latch;

- Low Pass Filter.

3.2.1 Digital to Analogue Converter

Design

As part of my research, I found out that digital to analogue converters come in DIP packages and opted to use a DAC0800LCN. This DAC requires a +5V, 0V and -5V supply.
I used the datasheet (https://www.ti.com/lit/ds/symlink/dac0800.pdf) of the DAC0800LCN to work out how it is connected.

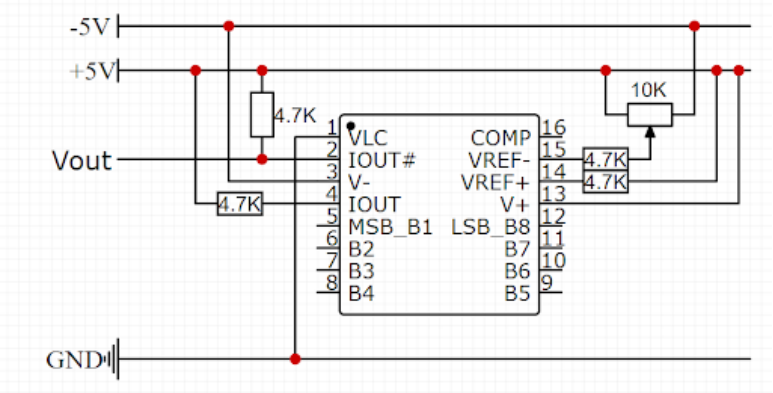


Figure 3.8: Circuit diagram of the DAC

NB: The inputs (B1 through B8) will be connected as shown in the counter circuit diagram

Building & Testing

I connected the DAC on its own first to ensure that it worked as expected before I combined it with the rest of the ADC. To test it, I fed a binary sequence into it (using the +5V rail and 0V rail to provide logic 1 and 0 respectively) and measured the output of the DAC using the voltmeter function of a multimeter.
The output of the DAC was extremely noisy. At the time of building the DAC, I disregarded this and assumed that it would be fine. However, as I constructed more of the ADC, I realised that the noisy output was causing lots of problems. Initially to rectify this, I tried adding decoupling capacitors to the +5V input of all chips on my breadboards. This should have reduced the noise from the power rails. This was unsuccessful, so I then added a low lass filter to the output of the DAC (this goes into the inverting input of the comparator). This is because the output of the DAC was at a frequency of 4MHz. As this was so high, I was able to select a break frequency which would filter the noise out and leave the ramp unaffected. I set the break frequency of the LPF to 10KHz and this solved my problems. The filtered output from the DAC was now clean, meaning it was no longer interfering with the Comparator. See Low Pass Filter section for the circuit diagram as well as the notes from testing the filter.

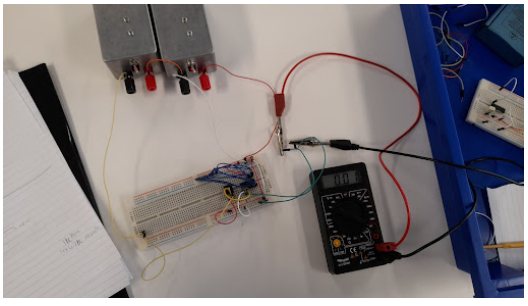


Figure 3.9: The DAC connected to 00000000, giving Vout of 0.01V

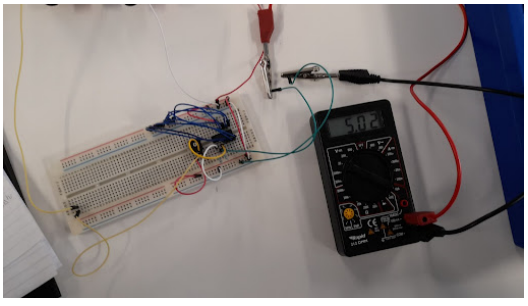


Figure 3.10: The DAC connected to 11111111, giving Vout of 5.02V

The input value 00000000 should have given me exactly 0V and the input value 11111111 should have give me exactly 5V. To obtain these values, I adjusted the potentiometer on pin 15 until the values were correct. The potentiometer is used to provide a reference voltage of 0V to the DAC chip.

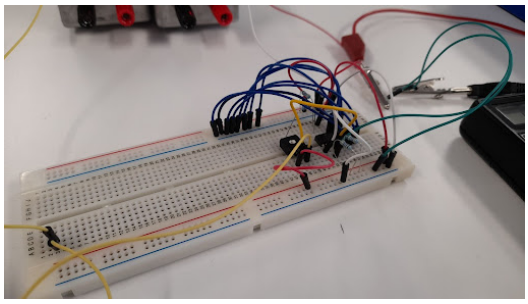


Figure 3.11: The DAC connected with temporary jumper wires on the breadboard

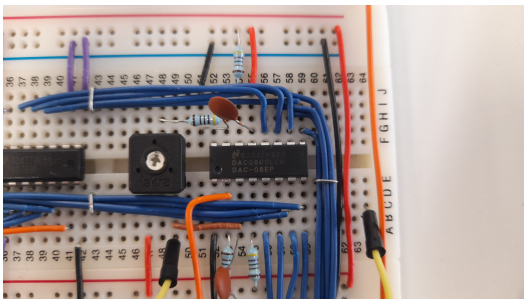


Figure 3.12: Close-up of neatened DAC chip

This subsystem will have to be tested further when I have constructed the counter, as this is what will drive it.
An alternative subsystem I could have used is a resistor ladder DAC or a binary weighted DAC. Overall, the most compact method which I could have used is the fully integrated DAC chip. The binary weighted DAC would have required an additional 12 resistors as well as two more op amps. The resistor ladder DAC would have required more than 16 resistors.

3.2.2 Counter

The counter will be used to provide the ramping input which goes into the DAC.

Design

To begin designing this subsystem, I looked at the datasheet for the CD4520 and understood how to operate the dual counters in cascaded up counter mode. I then designed the circuit diagram.

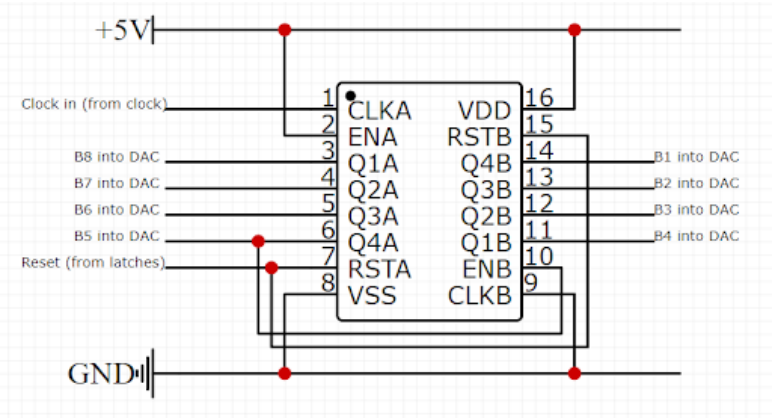


Figure 3.13: Circuit diagram of the counter

Building & Testing

I began by connecting the counter together as I specified in my circuit diagram. As I had already built the clock, I connected that to the clock input of counter A, and cascaded into counter B. After constructing the DAC, I was able to test the counters and didn't get the response I expected.
After some troubleshooting, I realised that I hadn't connected the reset pins to 0V, meaning they were floating, causing the counter to reset randomly, giving me random readings on the output of the DAC. Once I had rectified the mistake, the counter worked as expected.

I then neatened the wires which carried the binary data between the counter and DAC. Before progressing further, I tested this again to ensure it still worked, which it did. The clock wires were still jumper wires as in the final circuit, they wouldn't be connected like this.
I was now able to test the DAC further. By connecting the probe of the oscilloscope to the output of the DAC, I was able to confirm that the counter was correctly outputting a sequentially increasing binary number and that the DAC was correctly converting this to an analogue signal. I could see this by the fact that the output of the DAC was a ramp waveform with amplitude 5V.

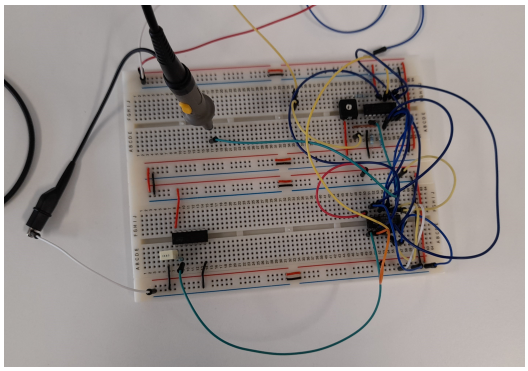


Figure 3.14: The counter, clock and DAC connected together with temporary jumper wires

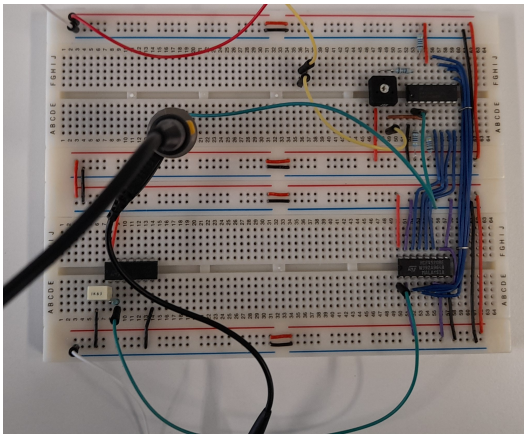


Figure 3.15: Post wire neatening

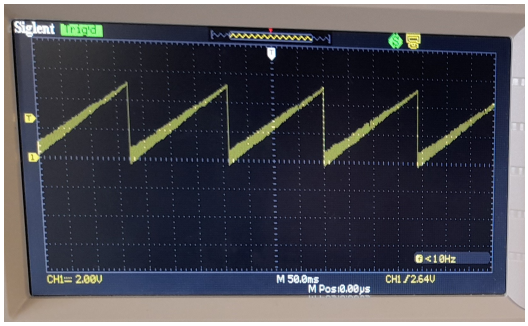


Figure 3.16: Ramp voltage output of the DAC

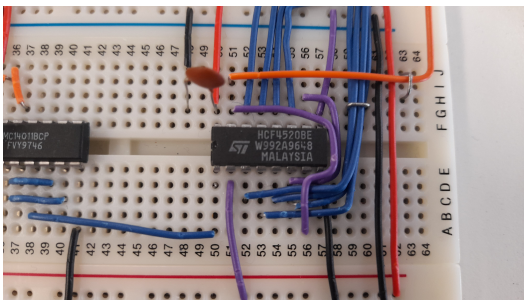


Figure 3.17: Close-up of neatened counter chip

The noisy output from the DAC can be seen in Figure 3.16, especially at the lower part of the ramp. An alternative subsystem I could have used is to manually construct the counters out of D-Flip-Flops however this would have been an unnecessary level of complexity for no benefit, as well as using lots more components.

3.2.3 Comparator

Design

I designed this based on a LM311 comparator and CD4011 IC NAND chip. First I drew out the circuit diagram, using the datasheets of both of the chips.

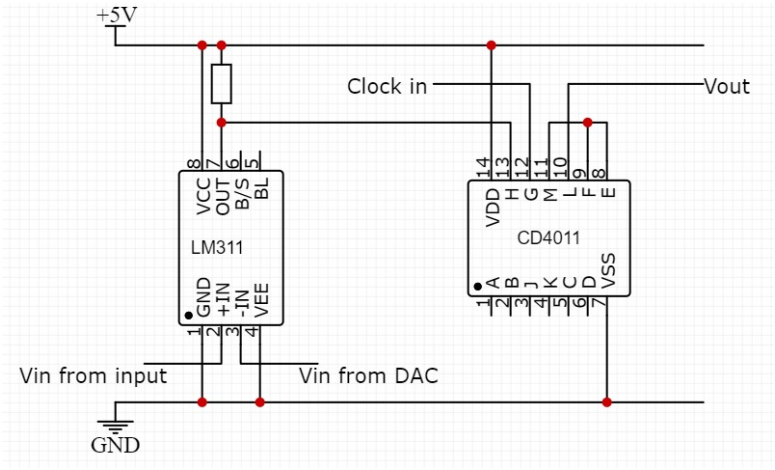


Figure 3.18: The circuit diagram for the Comparator

NB: There were some additional Schmitt inverters used to regulate the signal outputted from the comparator before it was AND'ed with the clock signal. These have been omitted from this diagram and can be seen in the full schematic in the appendix.

Building & Testing

I connected the comparator according to the design. To test the comparator, I used two rotary potentiometers, each connected as a potential divider to allow me to adjust the voltages, simulating the ramping output of the DAC and the varying voltage output from the input. I used a multimeter to check the input voltages and a logic probe to view if the output logic level was high or low.

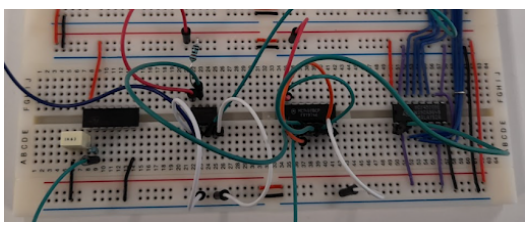


Figure 3.19: The comparator(2nd from left) & NAND(3rd from left) chip setup before any test equipment attached

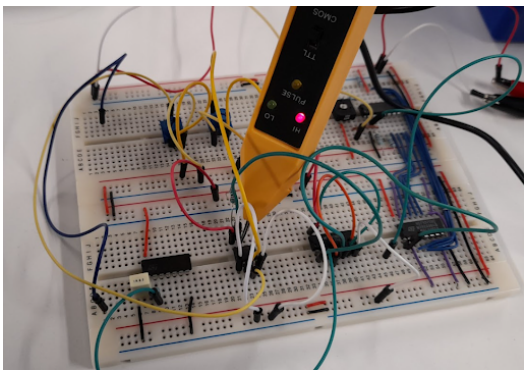


Figure 3.20: The comparator, rotary potentiometers and a logic probe showing a high output

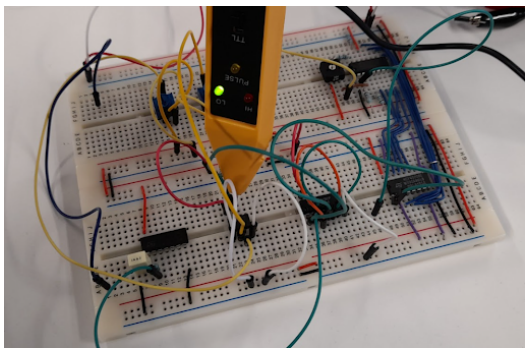


Figure 3.21: The comparator, rotary potentiometers and a logic probe showing a low output

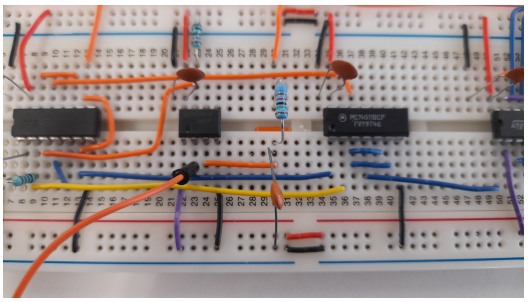


Figure 3.22: The neatened comparator subsystem (also shows LPF)

The output signal from the comparator wasn't clean; this lead to false triggering of the reset to the counter. To resolve this, I put the signal through 2 Schmitt inverters to make the signal into more of a perfect logic 0 pulse. This worked and I was happy with the progress so far.

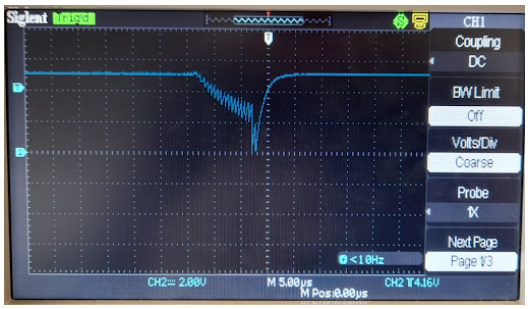


Figure 3.23: The comparator output before filtering with 2 Schmitt inverters

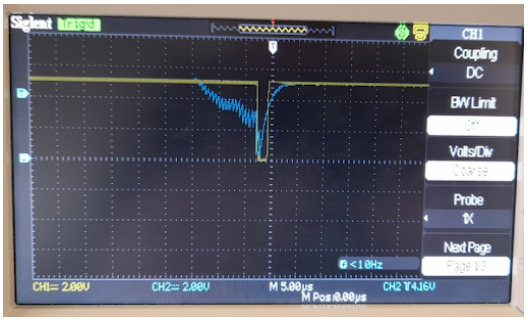


Figure 3.24: The before (blue) and after (yellow) filtering the comparator output with 2 Schmitt inverters

3.2.4 Delay Line

Delays are needed after the comparator has temporarily sent a low pulse to ensure that the correct things happen in the correct order. First the latches have to latch, then after some delay, the counter has to reset. It is important that these happen in this order so that the latches have latched before their signal is changed.

Design

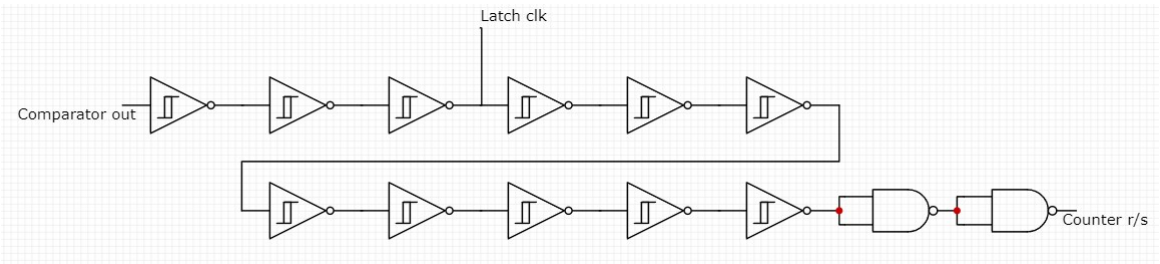


Figure 3.25: The circuit diagram for the delay line

Building & Testing

This delay line caused a lot of problems during the building and testing phase, especially during the construction of the latches. The final iteration of the design upon which I have settled, is by no means perfect, however it does work and the subsystems interact as I expect them to. It is not perfect as there is a seemingly unnecessary number of Schmitt inverters used, I discuss this and some possible solutions further in the testing and evaluation chapters.

3.2.5 Latch

This will save the binary signal outputted from the counter when the correct value has been found. This signal will then be sent to the EEPROM.

Design

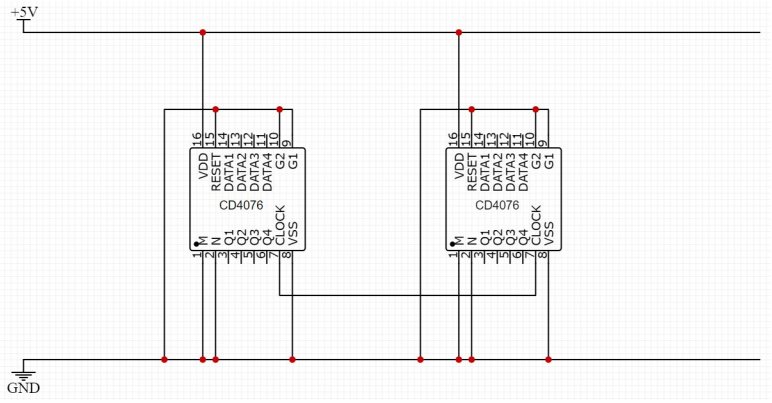


Figure 3.26: Circuit diagram of the latch

Pins 3 through 6 inclusive for both chips will be connected to the EEPROM and pins 11 through 14 inclusive will be connected to the counter output lines. These have been omitted from this diagram for simplicity and can be seen in the full schematic.

Building & Testing

I connected the latches as specified in my design. This didn't work initially, so I troubleshooted and found that the noise on the output of the DAC was causing me problems. I then added the Low Pass Filter to that line which solved my problems. *(See DAC design and testing for more information on this & circuit diagram)*

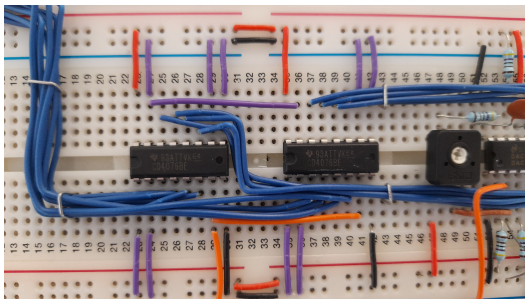


Figure 3.27: Neatened latch system

3.2.6 Low Pass Filter

This subsystem is needed to filter out the noise from the output of the DAC. The DAC's output has a frequency of about 4MHz which is extremely high. The break frequency I will be using is 10KHz,

this should be high enough to still allow the ramp waveform through.

Design

I first drew the circuit diagram for this circuit.

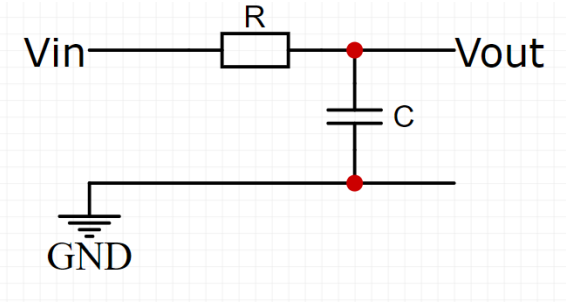


Figure 3.28: The circuit Diagram for the low pass filter

I then calculated the values for the resistor and capacitor using the break frequency formula with a break frequency of 10KHz.

$$f_b = \frac{1}{2\pi RC}$$

$$10 \times 10^3 = \frac{1}{2\pi RC}$$

$$R = \frac{1}{2\pi(10 \times 10^3)C}$$

$$R = \frac{1}{2\pi(10 \times 10^3)(10 \times 1^{-9})}$$

$$R = 15915.49431\Omega$$

$$R \approx 1.59K\Omega$$

This means the resultant component values are:

$R = 16K\Omega$
$C = 1nF$

Building & Testing

I built this as I had specified in my design. This worked extremely well; reducing the amount of noise on the analogue output of the DAC.

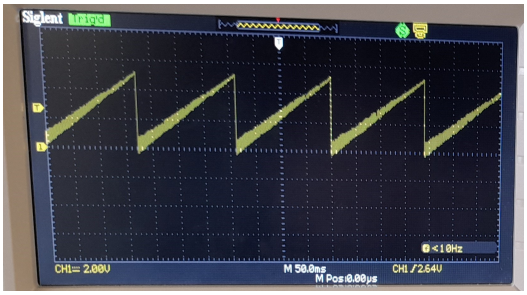


Figure 3.29: The input to the Low Pass Filter

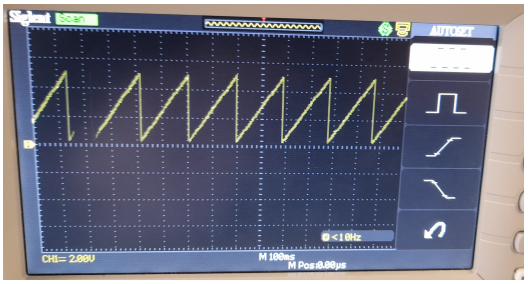


Figure 3.30: The output of the Low Pass Filter

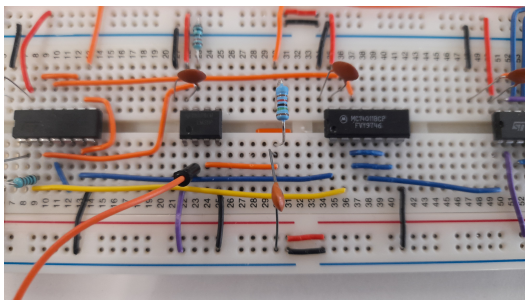


Figure 3.31: The physical layout of the Low Pass Filter on the breadboard

An alternative subsystem I could have used is an active filter, this would have increased the complexity however as it would require an additional resistor and op-amp. Alternatively, I could have used a band pass filter, however this would also require an additional component - the inductor. Both of these options would have added unnecessary complexity as the simple passive low pass filter works fine.

3.3 Display Subsystem

There will be two seven-segment displays, one for the units and one for the tenths. The one for the units will have a decimal point which will be permanently tied high.

Design

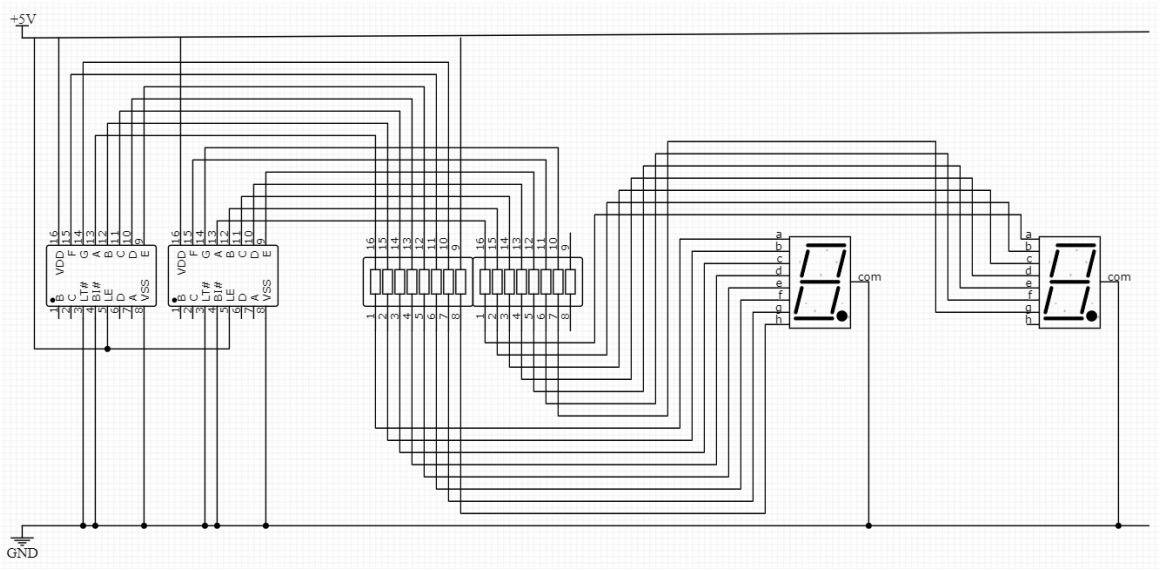


Figure 3.32: The circuit diagram for the display subsystem

The inputs A, B, C and D for each of the display drivers will be connected to the outputs of the EEPROM. These connections can be seen in the full circuit diagram in the appendix.

Building & Testing

First I connected this up as specified in my design, except from pins 3, 4 and 5 which I left unconnected. After I had connected these, the design worked. To confirm that the design worked, I fed a binary signal into the display subsystem. This can be seen in Figures 3.33 & 3.34.

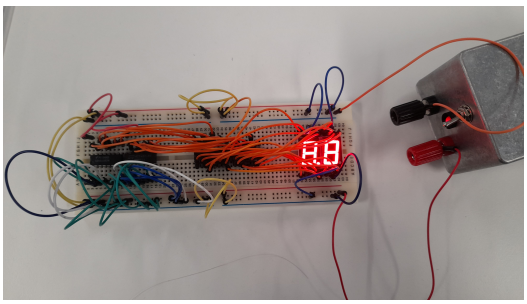


Figure 3.33: Pre-neatened subsystem showing 8.8

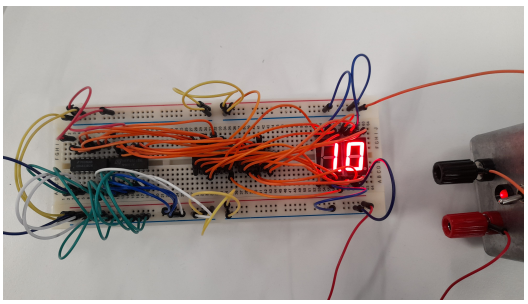


Figure 3.34: Pre-neatened subsystem showing 1.0

I then neatened the subsystem. To test that I hadn't broken anything during the neatening process, I used temporary jumper wires to act as the bits being fed into the display drivers. During testing, I found that I hadn't broken anything and the subsystem was working as I wanted it to.

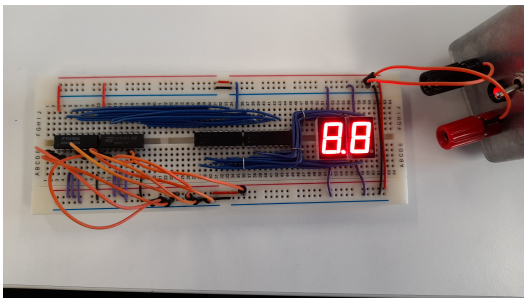


Figure 3.35: Neatened subsystem showing 8.8

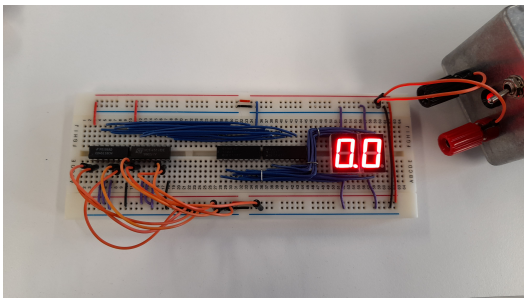


Figure 3.36: Neatened subsystem showing 0.0

3.4 Memory Lookup Table Subsystem

The memory subsystem is based off of an Electronically Erasable Programmable Read-Only Memory (EEPROM) chip. The memory will work by using the value which has been latched onto the latches from the ADC subsystem when the ADC has converted the analogue value from the input voltage. The purpose of the EEPROM is to take a input value and convert it to the corresponding BCD value which has been programmed into it. This will then be fed into the display subsystem which will display the values.

Design

To begin designing this, I worked out the values which the EEPROM would take and what it would output. I used a Google Sheets spreadsheet to do this. This was then inputted into the programming software which will burn the data onto the chip. To calculate the values I used the following steps.

1. List all values from 0 to 255 inclusive

2. Calculate what exact voltage this value would correspond to using the equation $Voltage = \left(\frac{5}{256}\right) \times Value$

3. Multiply this new value by 10, then round to zero decimal places.

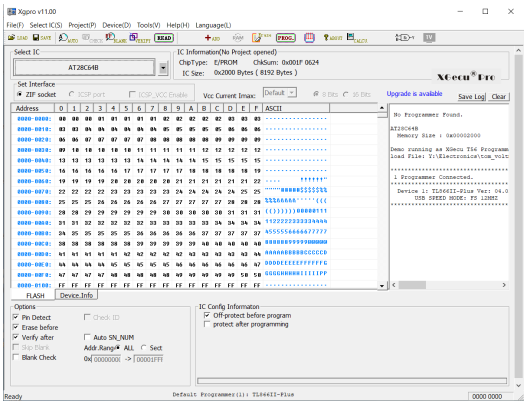


Figure 3.37: The data ready to be burnt to the chip

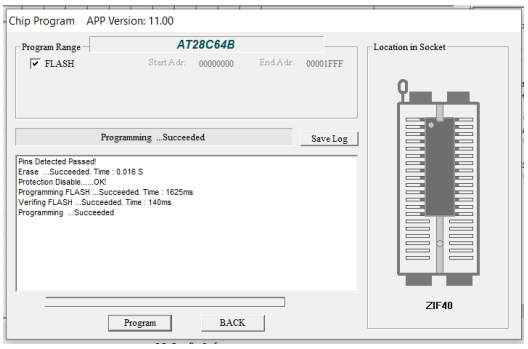


Figure 3.38: The log output from the burning software

I then worked out the schematic of the EEPROM.

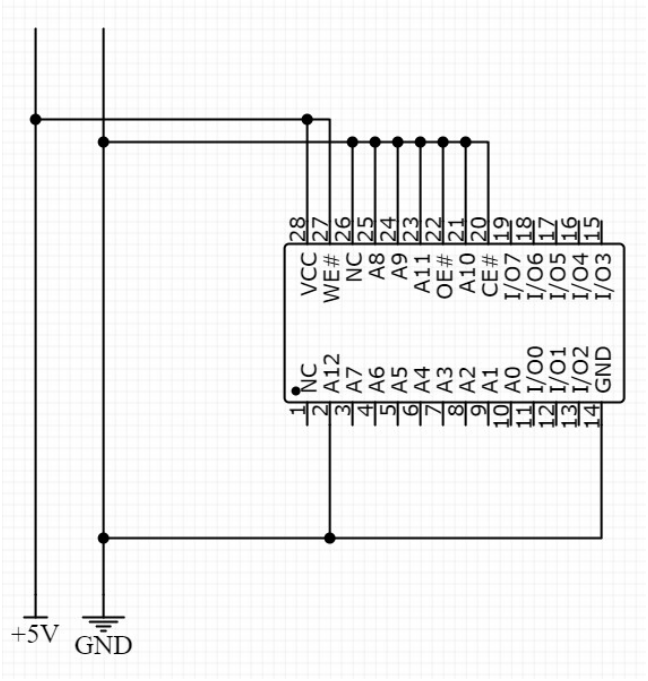


Figure 3.39: Circuit diagram of the memory lookup table subsystem

Pins 3 through 10 inclusive will be connected to the output of the latches and pins 12, 13, 14 and 15 through 19 inclusive will be connected to the inputs of the display subsystem. These have been omitted from the diagram above for clarity and can be seen in the full schematic.
On the EEPROM chip, there are a number of pins which have to be tied either high or low, to allow it to output and to prevent overwriting the data which has been burnt to the chip. Pin 27 is write enable, this is active low so has to be tied high as we don't want to accidentally overwrite the data on the EEPROM. Pin 22 is output enable (again, active low), this has to be tied low as we want to output the data on the EEPROM. Pin 20 is chip enable (again, active low), this has to be tied low as we want to enable the chip to work.

Building & Testing

To begin building this, I connected the outputs to inputs of the display subsystem, this would allow me to test if the EEPROM has been programmed correctly. I then connected the functional and power pins. Finally, I connected the inputs which I won't be using (as the EEPROM chip has a 14-bit input and I only need an 8-bit input) to 0V to ensure that they won't affect the address which the EEPROM

is drawing data from. The inputs which I will be using, I initially connected to 0V (00000000). Now I am able to use jumper wires to imitate the binary sequence from the latches to ensure that the data has correctly been stored on the EEPROM and it is outputting as expected.

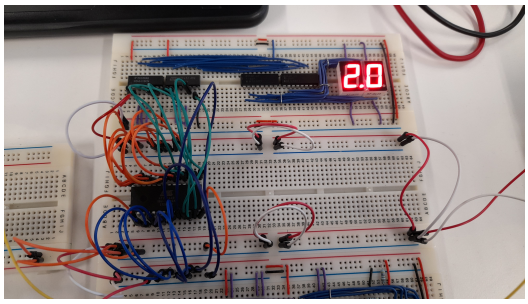


Figure 3.40: Input: 01100110 and Output: 2.0

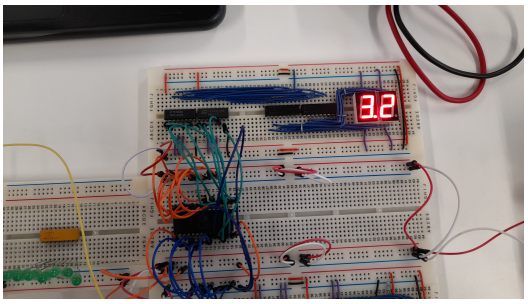


Figure 3.41: Input: 10100010 and Output: 3.2

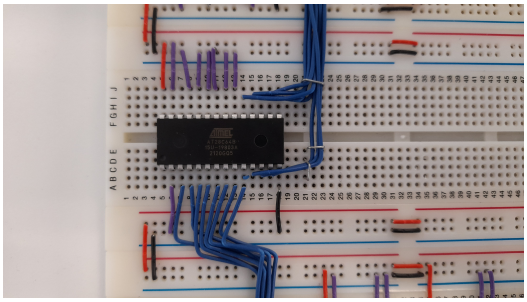


Figure 3.42: Neatened EEPROM subsystem

Now I know that the data has correctly been stored on my EEPROM, I am able to connect the input to the EEPROM to the latch outputs; using a potentiometer for input to the circuit. Initially when I did this, the output wasn't displaying what I was expecting it to. At this point, I was running out of time with the project, so I decided that due to the fact that the EEPROM was working on its own, I would be able to just neaten up the wires then test and troubleshoot the whole system together.
An alternative subsystem I could have used instead of the EEPROM is a microprocessor. This would have been overkill for this project.
I can now test the full system together. This can be found in chapter 5.

Chapter 4

User Guide

Operation of the voltmeter is extremely simple.

1. Connect the power supply to the system using the +5v, 0v and -5V connections
2. Connect the input voltage to the input connector on the breadboard
3. Read the voltage from the seven segment displays

Chapter 5

Final System Testing

After completing construction of the system, I then could test the entirety of the system.
To do this, I used a potentiometer to provide the input voltage. I can measure this using a multimeter and compare it to the output of my voltmeter. This was unsuccessful as the voltages were different between the multimeter and my voltmeter's output voltage.
After using the oscilloscope, I found out that the latches weren't clocking at the right time, resulting in the displays not updating. This was most probably because, despite the delay line, there was not enough delay before clocking the latches.
To attempt to resolve this, I added in a button in and removed the delay line ending in the clock for the latches. The button is used to trigger the clock of the latch. This didn't work.
At this point I ran out of time to complete the project. If I had more time I would continue troubleshooting. The next troubleshooting step I would be trying is to add a system controller which would trigger the latches to latch and counters to reset at frequent intervals. This would hopefully improve the reliability of the system.

Chapter 6

Evaluation

Once I had fully constructed my circuit, I am able to connect it to the power supply and evaluate what I have produced against my success criteria.

6.1 Evaluating against Specification

- **The voltmeter will measure in volts and range from 0V to 5V, in 0.1V increments.**
  When the voltmeter works, this is achieved. However as the voltmeter doesn't work reliably this point cannot be marked as achieved.

- **The voltmeter will have a sample rate of 1KHz $\pm 400Hz$ times per second. This will be produced by Schmitt Astable with a frequency of 1KHz.**
  From testing the clock subsystem, I know that the frequency of the clock is $1.36KHz$. This is a suitable deviation from the original success criteria (as it is an increase of less than 400Hz), meaning the clock pulses 1360 times per second. Therefore this point can be marked as achieved.

- **The voltmeter is accurate to within $\pm 0.1V$.**
  From testing the final circuit, I have found out that when it works, it is accurate within my specified bounds but when it doesn't work it is extremely inaccurate. Therefore this point can be marked as partially achieved.

- **The voltmeter takes less than 0.5 seconds to adjust its output once a change in voltage is detected.**
  Despite the fact that the voltmeter was unreliable, I was able to measure the frequency of the clock. This frequency would mean that the clock could complete a full ramp in less than 0.5 seconds. Therefore this point marked as achieved.

- **The voltmeter should take a +5v, 0V and -5V (within ±0.5V) input.**
  These are the inputs required for the DAC subsystem, therefore this point can be marked as achieved.

- **The voltmeter will be easy to use.**
  From testing and asking the opinions of those who have tested my voltmeter for me, I have found that the voltmeter is easy to use. Therefore this point can be marked as achieved.

- **The voltage will output to two 7-segment displays; one for the units and one for the tenths.**
  As seen in my full layout image, there are two 7-segment displays, one of which is used for units and the other for tenths. Therefore this point can be marked as achieved.

- **The entire circuit will only require human input to connect the analogue voltage.**
  When the circuit is working, this point is achieved. However as the circuit doesn't work all of the time, this point can be marked as partially complete.

- **The voltmeter should be as efficient as possible, reducing heat dissipated to the environment.**
  As my circuit is using a low voltage and current input, there isn't much energy to be dissipated. This low energy input requirement is partially due to the fact that I chose to use CMOS chips rather than TTL. TTL require a much higher operating voltage and current therefore they dissipate a much higher amount of energy. Therefore this point can be marked as achieved.

From evaluating against my specification, I can conclude that this project was not a complete success as only 66% of the specification points were met.
For my project to have been a success, I would have needed an 75% achievement rate of my success criteria. This would have meant 7 of the points would have to be achieved.

6.2 Improvements

To improve this project, I would troubleshoot further into the cause of the latches not clocking at the right time, then work to fix that issue. Assuming that this is the only issue I come into, then that would allow me to mark the remaining 4 success criteria points as achieved.
A further improvement I could make is to take my layout on breadboards, and use a digital tool to convert it into a PCB. This would reduce the overall physical layout of the circuit as well as reduce the chance for a component to be knocked out. Furthermore, this would improve the noise performance of the circuit and allow for a better grounding arrangement. This would also allow me to implement a better timing system as it would be simpler to construct.

6.3 Conclusion

In conclusion, my system did not meet all of the specification points I put together before constructing it. This means it is not a functioning voltmeter. This was an extremely challenging project however once I understood how the subsystems worked, I was able to design and build them using the knowledge I gained from the theory modules of the course. The most challenging aspect was troubleshooting the many issues I ran into, especially the noise on the output from the DAC.

Appendix A

Location of chips on breadboard

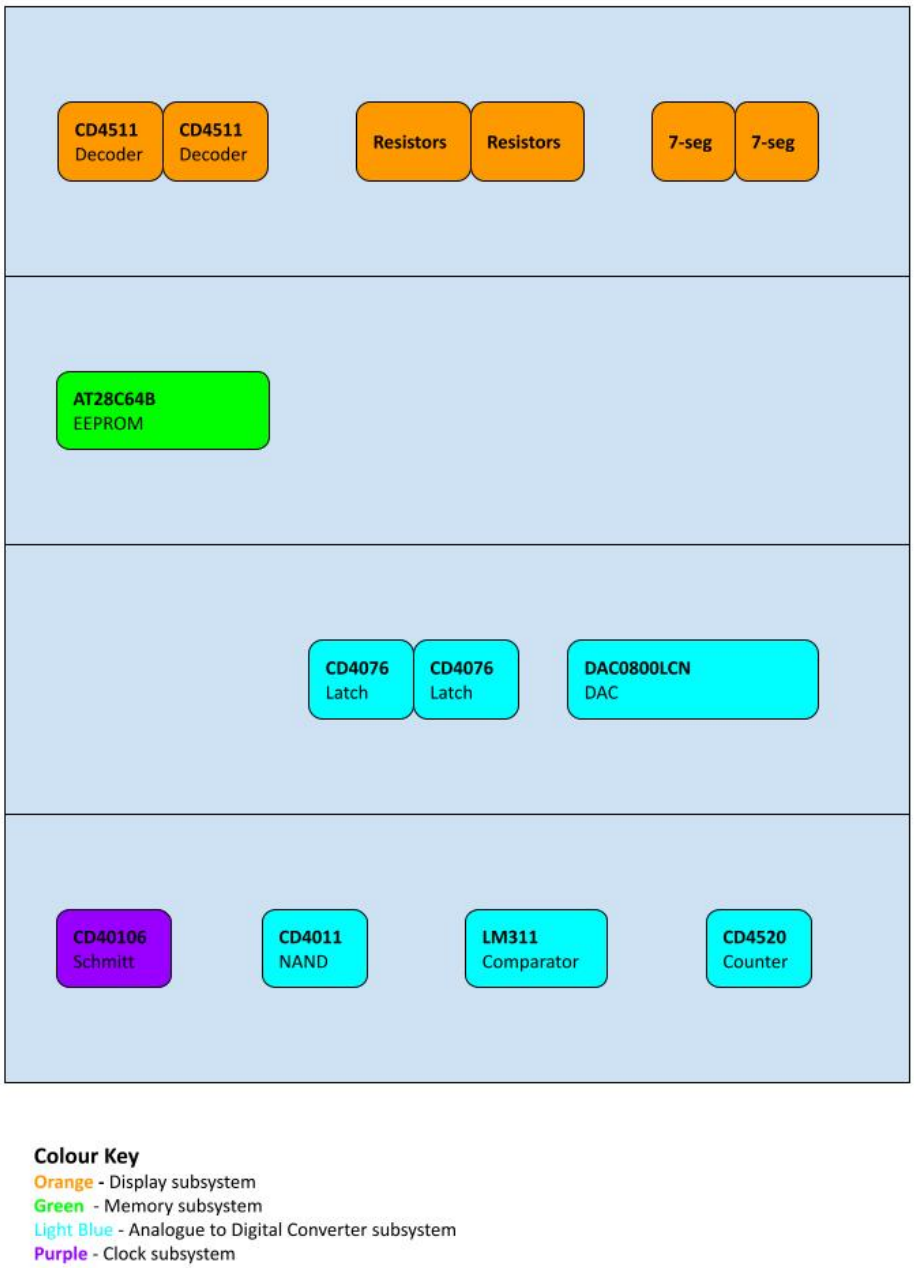


Figure A.1: Location of chips on the breadboard

Appendix B

Colours of wires used

Colour of wire	Signal wire carrying
Black	0V
Blue	Binary data
Brown	-5V
Orange	Analogue Data / Control
Purple	Functional pins (eg, enable / reset)
Red	+5V
Yellow	Clock

Appendix C

Full schematic

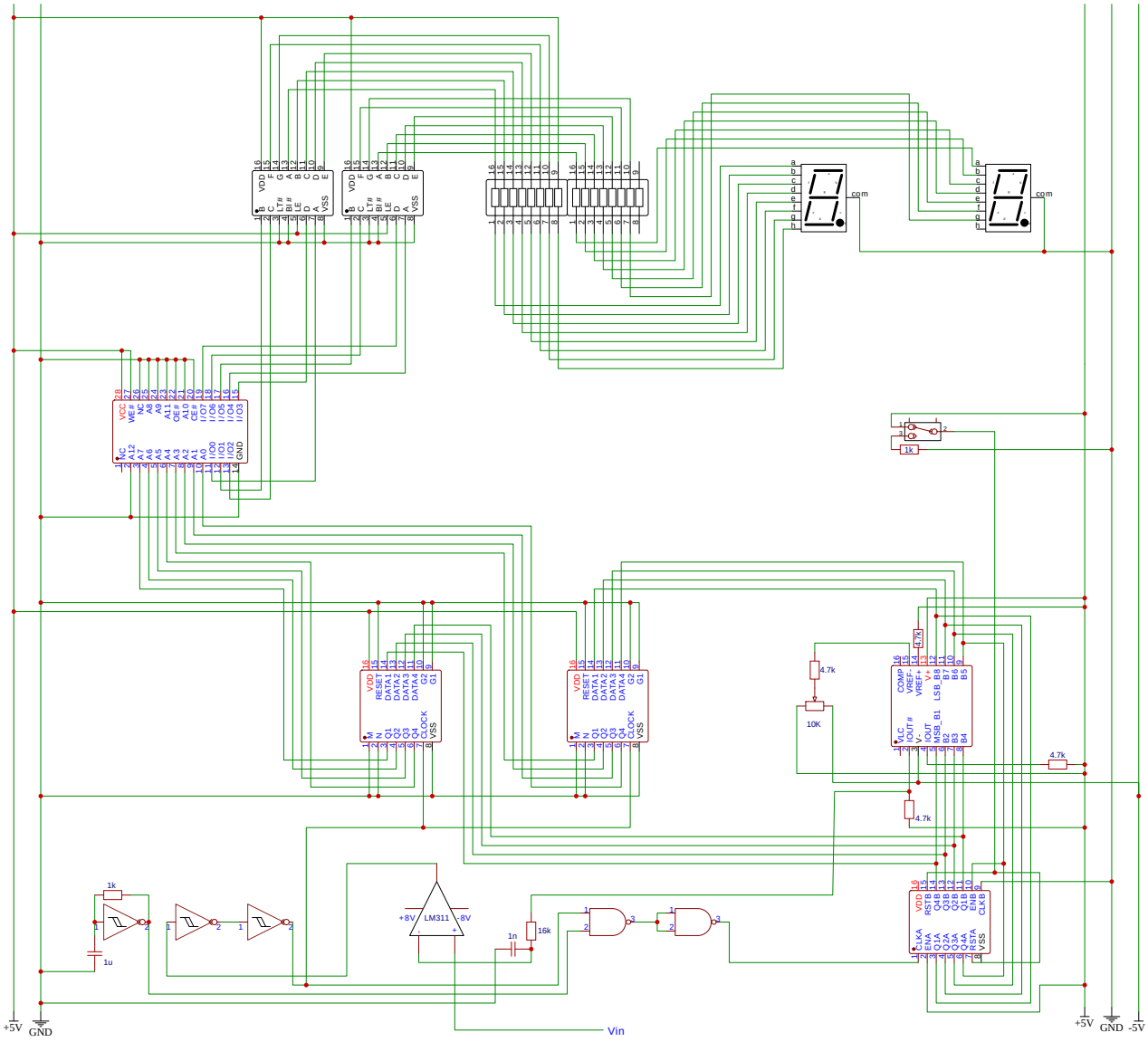


Figure C.1: Full Schematic

Appendix D

Final Physical Layout

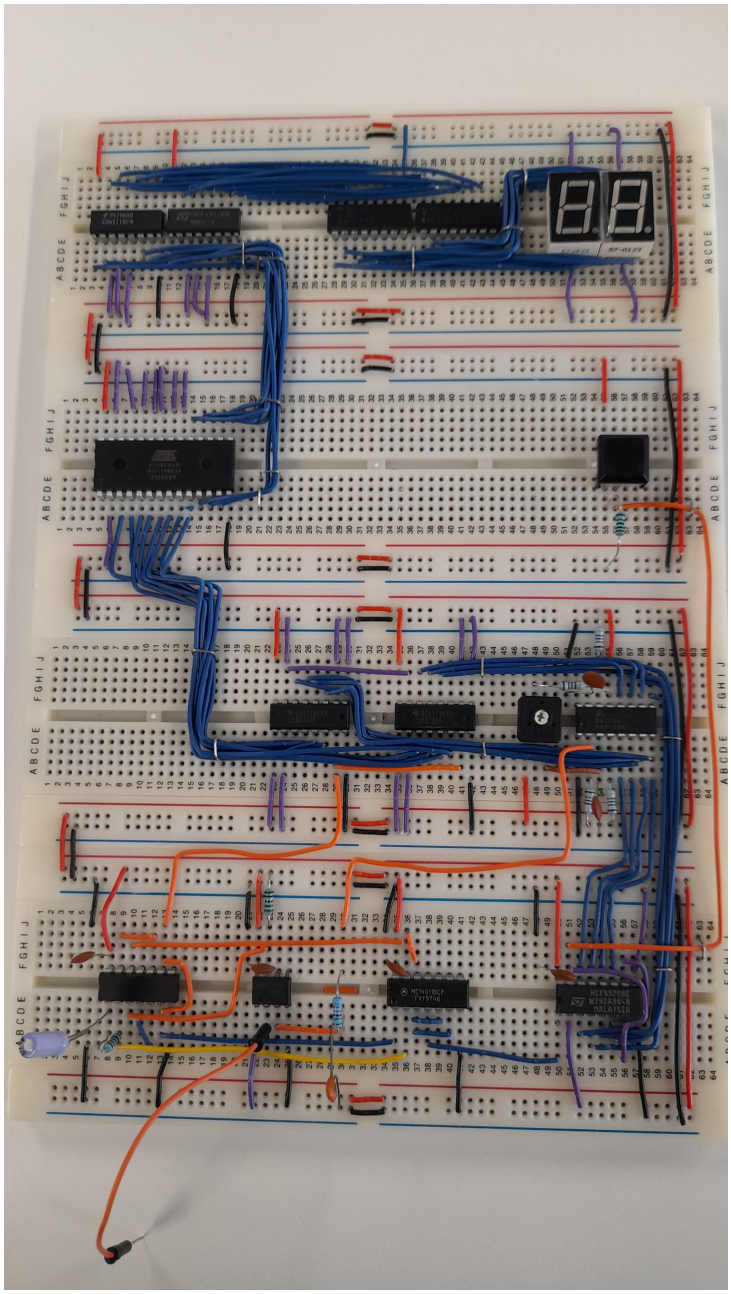


Figure D.1: Final layout of the circuit

Appendix E

Full Components list

E.1 Resistors

- 2 220Ω 8-way resistor packs
- 1 380Ω
- 1 $1K\Omega$
- 4 $4.7K\Omega$
- 1 $10K\Omega$ rotary potentiometer
- 1 $10K\Omega$
- 1 $16K\Omega$

E.2 Capacitors

- 7 $1\mu F$
- 1 $1nF$

E.3 Integrated Circuit Chips

- 1 CD40106, Hex Schmitt Inverter, https://www.ti.com/lit/ds/symlink/cd40106b.pdf
- 1 LM311, Comparator, https://www.ti.com/lit/ds/symlink/lm311.pdf
- 1 CD4011, Quad NAND Gates, https://www.ti.com/lit/ds/symlink/cd4011b.pdf
- 1 CD4520, Dual Binary Up-Down Counter, https://www.ti.com/lit/ds/symlink/cd4520b.pdf
- 1 DAC0800LCN, 8-Bit Digital To Analogue Converter, https://www.ti.com/lit/ds/symlink/dac0800.pdf
- 2 CD40706, 4-Bit D-Type Registers, https://www.ti.com/lit/ds/symlink/cd4076b.pdf
- 2 CD4511, BCD-To-7-Segment Latch Decoder Drivers, https://www.ti.com/lit/ds/symlink/cd4511b.pdf
- 1 AT28C64B, 64K (8K x 8) Parallel EEPROM with Page Write and Software Data Protection, https://ww1.microchip.com/downloads/en/DeviceDoc/doc0270.pdf

E.4 Other

- 1 Push-To-Make button