Bank Security System Represented via Circuit Boards and Subsystems

Modules Representing Security Protocols for a Bank

Module: **Extended System Design**

Focus: **Designing an Analogue System**

A report originally created to achieve an

A Level in Electronics

Submitted 5/5/2017

Abstract

This paper outlines the design of a bank security system, represented through electronic subsystems that each demonstrate a particular security feature. The system demonstrated in this report has two set of inputs, the first being an array of panic buttons, to be used in the event that a staff member is in danger. The second input is a 16-bit binary code, which is used to open the vault door. If a wrong code is inputted, or if any of the panic buttons are pressed, then the bank will go into lockdown. This is represented by several LED's that will flash intermediately and indefinitely, until the threat is dealt with accordingly. The circuit also has another LED that flashes periodically, signifying to the security guards when it is time to do another routine time check and inspect the vault, furthering increasing the security of the bank. The report then goes on to outline the specifications of each subsystem.

Contents

Initial Specification

- The system should have an LED that has a mark to space ratio as close as possible to 14/1, which is then inverted to give a brief 1 second flash every 15 seconds.
- The system should have an alarm made out of two LED's, both with time periods of 1 second and mark to space ratios of 1/1, flashing out of phase with each other.
- The system should have multiple panic buttons that activate a warning LED when pressed.
- The system should have a pre-set code that is used to lock the vault that is easily accessible and changeable, without having to change the wiring of the circuit.
- The system should trigger the alarm if a wrong code combination is inputted.
- The system should have a way to reset the circuit without being turned off.
- The system should have a way to turn the rest of the circuit off, even when there is a power supply that is turned on and connected to the circuit.

Circuits Required

Master Switch and Reset Button

For any of the subsystems to receive power, the master switch must be turned on. Otherwise, even if power is being delivered to the circuit, no power will be directed to any of the subsystems. If the reset button is pressed at any time, then the entire circuit will stop having power flowing through it.

Panic Button Input

Once the *Master Switch* is turned on, a 5V DC supply will be directed to the sets of buttons in the circuit. If both buttons in the same set are pressed the voltage will be sent through an AND gate. This causes a warning LED to light up. Only one set of buttons will need to be pressed for this to occur.

Dual Flashing LEDs

This part of the circuit consists of 2 470Ω resistors, 2 22kΩ resistors, 2 100uF capacitors and an NPN transistor. As long as the *Master Switch* has been turned on, the circuit will have two alternating flashing LEDs, until the *Reset Button* is pressed. This will stop the 5V DC supply from flowing around the subsystem, stopping the LEDs from flashing until the button has been released.

16-Bit Binary Code Input

After the *Master Switch* is turned on, a 5V DC supply will be sent through all of the DIP switches that are turned on. Once the user has entered a code into the *16-Bit Binary Input*, the circuit will compare it to the *Present Code* via 4 7486 XOR gates, to check if the DIP switches produce the same output. If the *16-Bit Binary Code Input* is the same as the *Present Code*, a voltage will be sent to an LED, to inform the user they have inputted the correct code to open the vault door.

555 Timer (Astable)

Once there is a 5V DC supply provided to the circuit by the *Master Switch* being turned on, the *555 Timer* starts to cause a capacitor to charge up. The capacitor will continue to do so for 14 seconds, giving out an output the entire time. At the end of this, the capacitor will dissipate all of its stored charge for 1 second. The *555 Timer* will not produce an output during this phase. The output of this is then fed into a NOT gate to invert the signal, resulting in 1 second pulse every 15 seconds. This is then sent to an LED, causing it to turn on for 1 second, before turning back off for 14 seconds.

Subsystem #1 - 555 Timer

Circuit Diagram of the Subsystem

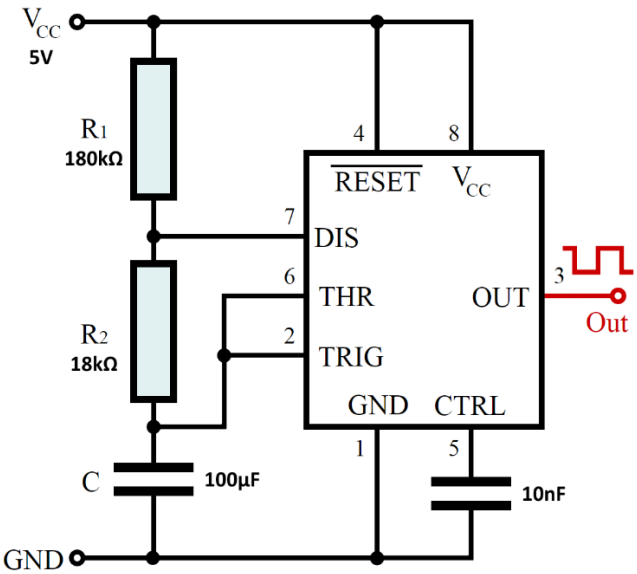

Figure 1: *The circuit diagram of Subsystem 1.*

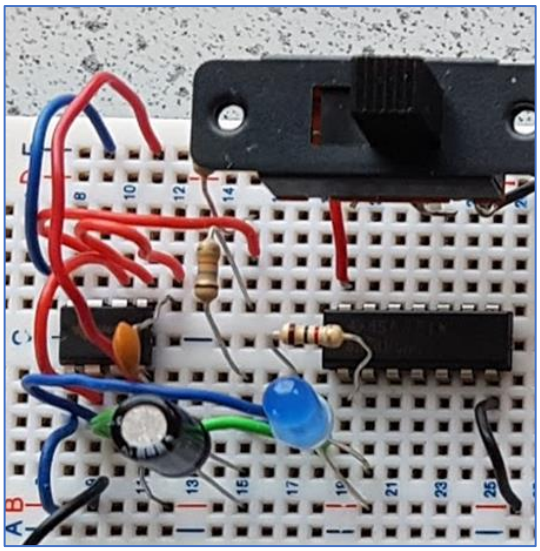

Figure 2: *Subsystem 1 built onto the circuit.*

Initial Subsystem Specification

- The subsystem should be powered by a voltage supply close to 5V DC.
- The subsystem should be built as an astable, providing an alternating positive output.
- The subsystem should last for 15 seconds, outputting a mark space ratio around 14/1.
- The subsystem should then be inverted, to give a brief 1 second flash every 15 seconds.
- The clock output can have any frequency, as it is irrelevant.

Function of the Subsystem

The subsystem contains a 555 timer that turns on for a second once every 15 seconds representing the time (with each second representing a minute) that it takes to reach the quarter intervals on the clock. I chose to use a 555 timer because I needed to have a different mark and space in my mark space ratio. The 555-timer is used to make an astable, with the output of the clock feed into a NOT gate to invert the mark to space ratio for later use, from 14:1 to 1:14. In order to build the 555-timer for my system I needed to first research how to wire up each the pins, all of which are stated below.

Pin 1 - Ground

The ground pin is connected to the 0V rail - commonly called the negative rail or earth rail.

Pin 2 - Trigger

This pin connects to the lower comparator and is used to set the control flip flop. When it is taken low, the output goes high. If it is kept low, the output remains high and the Threshold (Pin 6) will not force the output low. This is because the trigger pin is the dominant pin.

Pin 3 - Output

To make the output high, the Trigger (Pin 2) is momentarily taken from a high to a low. This causes the output to go high. The output can be returned to a low by making the Threshold (Pin 6) go from a low to a high, or by taking the reset pin to a low state.

Pin 4 - Reset

This pin is used to make the Output (Pin 3) low. The reset pin must go below 0.7 volts and it needs at least 0.1mA to reset the chip. The reset pin is an overriding function. It will force the Output (Pin 3) to go low regardless of the state of the Trigger (Pin 2). The pin is tied to the positive rail when not in use to avoid the possibility of false resetting.

Pin 5 - Control Voltage

By applying a voltage to this pin, it is possible to vary the timing of the chip independently of the RC network, producing a frequency modulated output whilst in astable mode. If the pin is not used, it should be bypassed to ground connected to a 10nF capacitor, to prevent noise entering the chip.

Pin 6 - Threshold

This pin is one input to the upper comparator (the other being Pin 5). It makes the Output (Pin 3) go low, by taking the Threshold pin from low to a voltage above 2/3 of the rail voltage.

Pin 7 - Discharge

This pin is connected to the open collector of an NPN transistor. When the transistor is turned "on", the pin is shorted to ground. The timing capacitor is connected to ground and is discharged when the transistor turns "on". The conduction state of this transistor is identical in timing to that of the output stage. It is "on" when the output is low and "off" when the output is high.

Pin 8 - Rail

This pin is the positive supply voltage pin for the 555. Supply-voltage operating range is 4.5V to 16V DC, which is why I have chosen to use a 5V DC power supply.

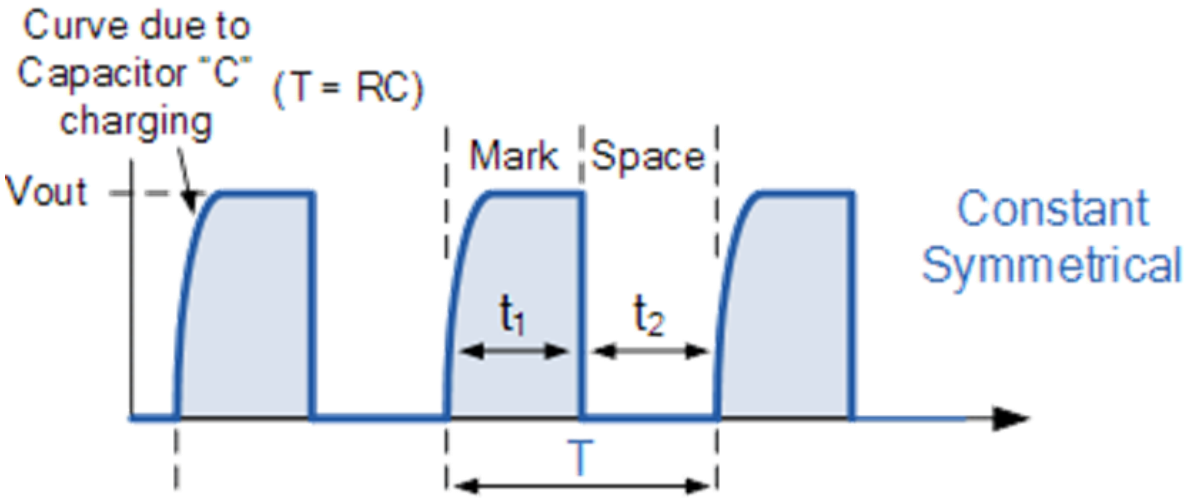

Figure 3: *A basic diagram that demonstrates how the mark to space ratio is calculated* (Storr, 2017).

Developing the Subsystem

Open further research, I found out that the mark time period cannot be less than the space time period. This was overcome however by inverting the output of the astable. I also discovered that the 0.69 used in most time period calculations is only correct to 2 significant figures; ln(2) was actually a more accurate number to use. However, because ln(2) is an irrational number, this resulted in me being unable to get an exact result of 14 seconds off and 1 second on by using the E-24 standard resistors and E-8 capacitors. Due to this, I prioritised that the time period be as close to 15 seconds as possible, with the mark to space ratio only being 1 second and 14 seconds if at all possible.

The sum of these periods gives an overall periodic time (T) of

$$T = (C_1R_{B1} + C_2R_{B2}) \ln 2. \qquad (6.33)$$

For a mark-to-space ratio of 1 : 1, i.e. when each transistor conducts for an equal period, $C = C_1 = C_2$, and $R = R_{B1} = R_{B2}$.

Figure 4: *The calculations needed in order to determine a desired mark to space ratio* (Storr, 2017).

Test Procedures

In order to get the most precise time period of 15 seconds, I had to calculate which resistor and capacitor values I would be able to use based on the calculation in **Figure 4**. The problem I had however was that due to the mark to space ratio I needed, I knew I would have to use values that weren't on the E-24 resistor list, meaning I would have to add multiple resistors in parallel. In order to make sure the values I calculated worked to produce a 14:1 mark space ratio, I used a stopwatch to measure the time it took for the LED to turn on and the back off again. I repeated this procedure until I had done it 5 times, so that I could average out the results in order to gain a more accurate result. I also chose to bypass the NOT gate while recording the sets of data, so that I could record the original mark and space time periods produced by the 555-timer before they were inverted. I also set up a voltmeter like in my circuit diagram to test the voltage supplied to the subsystem when the subsystem was both turned on and off.

The Mark Space Ratio of Subsystem 1				
Data Set	**Mark (Secs)**	**Space (Secs)**	**Period (Secs)**	**M/S Ratio**
Set 1	13.58	1.30	14.88	13.58:1.30
Set 2	13.47	1.12	14.59	13.47:1.12
Set 3	13.36	1.31	14.67	13.36:1.31
Set 4	13.40	1.37	14.77	13.40:1.37
Set 5	13.29	1.20	14.49	13.29:1.20
Average	13.42	1.26	14.68	13.42:1.26

Figure 5: *Testing the mark space ratio of the subsystem to make sure that it meets the specification.*

Test Results

As you can see in in **Figure 5**, the actual total time period and mark space ratio is very close to the 15 seconds that I stated I wanted in my initial subsystem specification. The frequency of the 555-timer was also measured to be 0.094 Hz, although it is irrelevant to the purpose of the subsystem as it was designed around the total time period. I also used a voltmeter to test the voltage of the subsystem, which gave me a reading of 4.86V when on and 0V when off, matching my initial specification.

Considered Alternative Design

An Alternative Design for Subsystem 1		
Option	**Main Advantage**	**Main Disadvantage**
Using a Schmitt trigger to create a mark space ratio.	Not susceptible to noise whilst within the voltage threshold.	Unable to have a different mark and space time period.

Figure 6: *A possible alternative design that I initially considered when first designing Subsystem 1.*

Due to the Schmitt trigger not being able to produce a different mark and space time period, I decided to use the 555-timer. This allowed me to send a quick pulse at the end of every clock cycle, rather than a long drawn out pulse which would not have met my initial specification.

Evaluation of the Performance

This subsystem works reliably, with no major flaws at all. Compared to the initial specification for this subsystem, the final performance is as required by the initial specification. This is because the astable 555 timer produces a mark space ratio close to my desired 14:1 that is 13.42:1.26, which is the inverted using the NOT gate. One thing that could be improved is to switch around the resistors until a closer mark and space time period is achieved. This would take a long time however, as the same values of resistors would still be being used, as the only thing changing would be the slight tolerance difference in the subsystem.

Final Subsystem Specification

- The subsystem is powered by a voltage close to 5V DC, as seen by my recorded 4.86V.
- The subsystem is an astable, and provides an alternating positive output voltage, as seen by my recorded voltage values of 4.86V and 0V.
- The subsystem does output a mark to space ratio that is very close to 14:1, with a mark to space ratio of 13.42:1.26.
- The subsystem does invert the output, to give a brief flash for around 1 second (1.26 seconds) that is very close to every 15 seconds (14.68 seconds).
- The clock output has a frequency of 0.094 Hz, but it is still irrelevant to the project.

Subsystem #2 - Dual Flashing LEDs

Circuit Diagram of the Subsystem

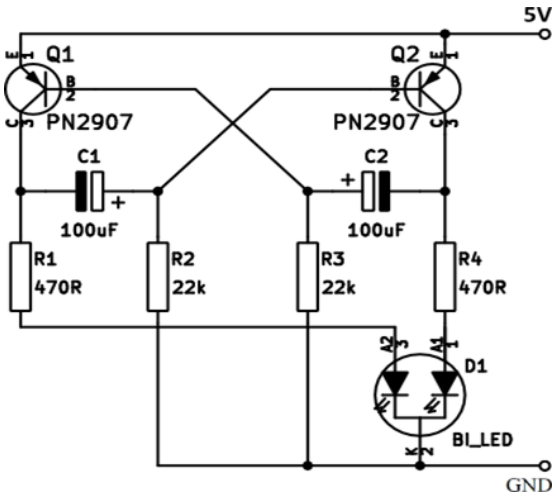

Figure 7: *The circuit diagram of Subsystem 2.*

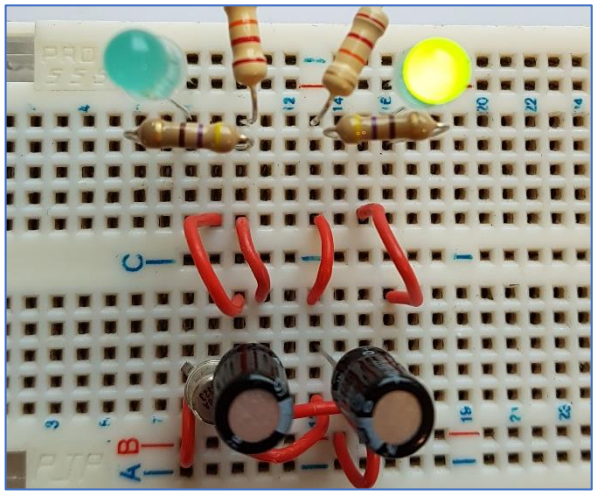

Figure 8: *An example output of Subsystem 2.*

Initial Subsystem Specification

- The subsystem should be powered by a voltage supply close to 5V DC.
- The subsystem should output the same mark and space time in the mark space ratio.
- The subsystem should have a total time period as close to 1 second as possible.
- The subsystem should have only one LED that is on at a time.

Reasons for Using Dual Flashing LEDs

My original design was to use two buzzers, however I later decided to use transistors, resistors, and capacitors in order to make dual flashing LEDs. This is because we had plenty of each of these in the department, whereas I would have had to order buzzers if I wanted to use it in my circuit. This would have reduced the amount of time at the end to test the subsystem to make sure it works reliably.

Function of the Subsystem

The dual flashing LED's represent the security alarm for the bank. Once a power supply is provided to the subsystem, the LED's will flash on and off alternatively with the same time period and will not stop until the user turns the master switch off (demonstrated later on in Subsystem 5).

Test Procedures

In order to obtain a total time period of 1 second for my dual flashing LED's, I used the previously detailed **Figure 4** to work out the component values. The product of my capacitor and my resistance equalled 0.721 to 3 significant figures, so I chose a value of 88 uF (4 x 22uF capacitors) and 8.2 kΩ. I measured the time it took for the LED's to turn on and off again with a stopwatch. I repeated the procedure until I had done it 3 times so that I could gain a more accurate result, and I also set up a voltmeter to test the voltage supplied to the subsystem. Due to the tolerance of the components in the circuit, the actual total time period and mark space ratio is detailed in **Figure 9**.

Test Results

The Time Period of Subsystem 2			
8.2k (Ohms)	**Time Period For 10 Cycles (Seconds)**		
Result 1	9.64	9.62	9.65
Result 2	9.65	9.63	9.64
Result 3	9.63	9.66	9.63
Set Average (3.s.f)	9.643	9.637	9.640
Overall Average (3.s.f)	9.642		
1 Cycle (Avg. / 10, 3.s.f)	0.964		

Figure 9: *Testing the time period of the subsystem to make sure that it meets the specification.*

Considered Alternative Design

An Alternative Design for Subsystem 2		
Option	**Main Advantage**	**Main Disadvantage**
Using buzzers in order to simulate the bank alarm.	The pitch can be changed to become more notable.	None on site, requiring time and money to buy equipment.

Figure 10: *A possible alternative design that I initially considered when first designing Subsystem 2.*

Due to the previously mentioned time constraints with ordering buzzers, I decided to settle on using LED's. Since my alarm is the main focus of my project, I wanted the LEDs to be as notable as possible. If I were to use the previous 555 timer setup for this subsystem, you would not see the LEDs were lit up due to the resistor values needed to make the time period one second. Due to this, I decided to not use a 555 timer, which resulted in brighter LEDs as a result of needing lower resistor values.

Evaluation of the Performance

Comparing the final performance to the initial specification, the subsystem worked perfectly. I produced a time period close to my desired 1 second with a recorded value of 0.964 seconds, and only one LED is ever on at time due to the correct wiring in the subsystem. One thing that could be added to further improve the subsystem is a loud buzzer alternating between two different pitches. This is so that it is more noticeable than just two small LEDs turning off and on. With both LEDs and buzzers, the alarm would be even more noticeable.

Final Subsystem Specification

- The subsystem is powered by a voltage close to 5V DC, as seen by my recorded 4.91V.
- The subsystem has a total time period close to 1 second, measured at 0.964 seconds.
- The subsystem does have only one active LED on at a time, as seen in my testing above.
- The subsystem does output the same mark and space time period in the mark space ratio, shown in my testing above.

Subsystem #3 - Logic Gates Input

Circuit Diagram of the Subsystem

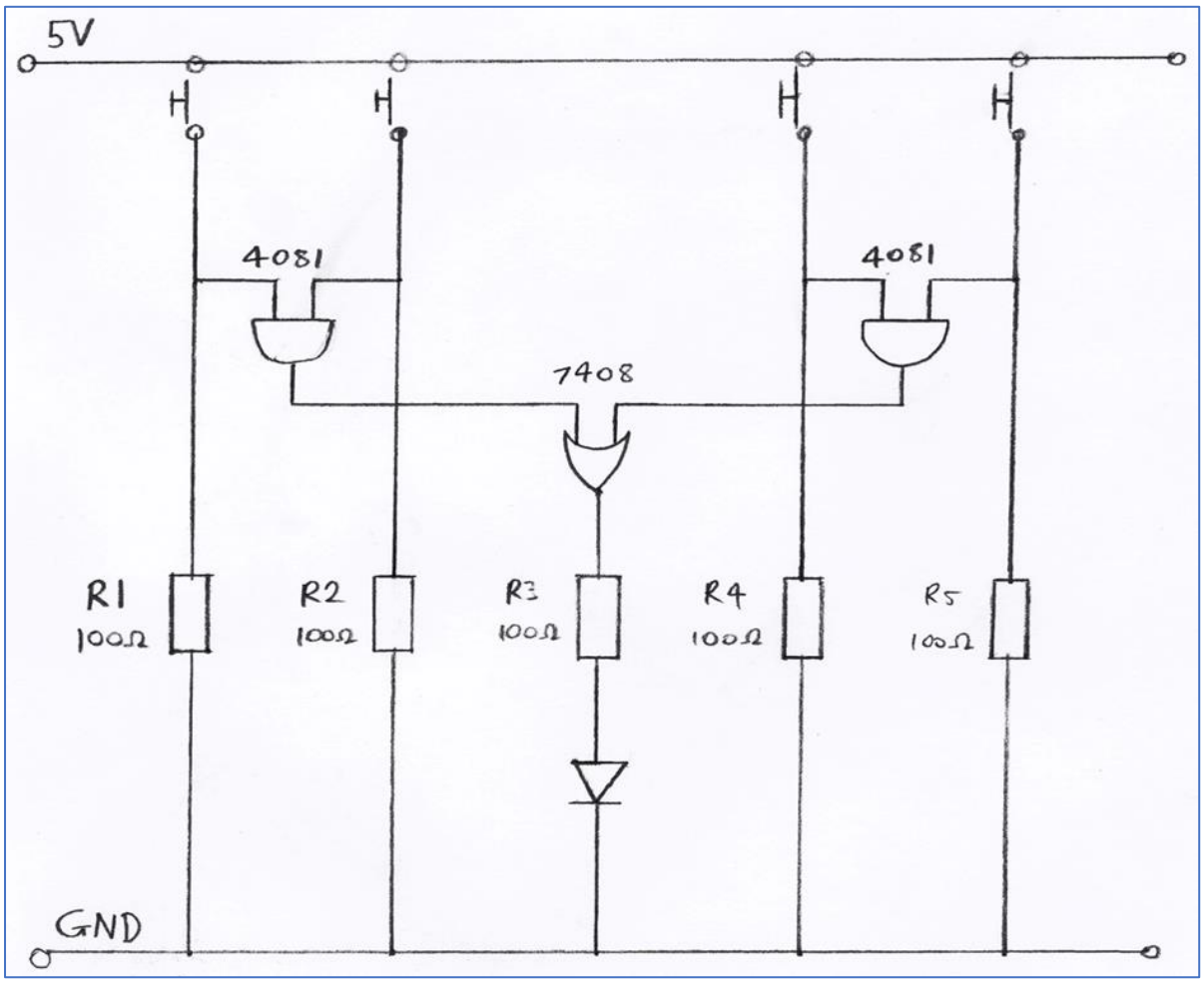

Figure 11: *The circuit diagram of Subsystem 3, representing two separate sets of two panic buttons.*

Initial Subsystem Specification

- Both the logic gates in the subsystem should be supplied with a voltage close to 5V.
- Each set of buttons in the subsystem must turn on the output LED when all pressed.
- Only one set of buttons in the subsystem needs to all be pushed to produce the output.

Reasons for Using Logics Gates and Buttons

Switches are very easy to turn on, and since the subsystem is representing panic buttons in a bank, I wanted the inputs to be easy to activate. I did not want to make it too easy however in case of an accidental press when there is no real threat. To combat this, I made the subsystem require one set of two side by side buttons to be pressed at the same time. I chose to use 1 4081 AND gate and 1 7408 OR gate for this subsystem, since we had plenty of those particular chips in the department already. This meant that I would not have had to spend extra time and money ordering any other chips in for the subsystem, allowing me to focus on the rest of the system in the meantime.

Test Procedures

In order to make sure the subsystem was working perfectly, I wanted to make sure that I had tested every possible combination of button presses. Rather than just pressing random combinations of buttons, I created a table of all 16 combinations, to make sure I did not miss any of them out. I wrote down what output I expected to see before started testing, in order to see if I received the same output as I wanted when I pressed each button. I also set up a voltmeter as seen in my circuit diagram to test the voltage supplied to the subsystem.

Test Results

The Outputs of Subsystem 3					
Button Set A		**Button Set B**		**Results**	
Button 1	**Button 2**	**Button 3**	**Button 4**	**LED Output**	**Expected**
-	-	-	-	Off	Off
-	-	-	Pressed	Off	Off
-	-	Pressed	-	Off	Off
-	-	Pressed	Pressed	On	On
-	Pressed	-	-	Off	Off
-	Pressed	-	Pressed	Off	Off
-	Pressed	Pressed	-	Off	Off
-	Pressed	Pressed	Pressed	On	On
Pressed	-	-	-	Off	Off
Pressed	-	-	Pressed	Off	Off
Pressed	-	Pressed	-	Off	Off
Pressed	-	Pressed	Pressed	On	On
Pressed	Pressed	-	-	On	On
Pressed	Pressed	-	Pressed	On	On
Pressed	Pressed	Pressed	-	On	On
Pressed	Pressed	Pressed	Pressed	On	On

Figure 12: *Testing the outputs of Subsystem 3 to make sure that the LED's match the expected result.*

I tested all 16 combinations of button presses, all of which produced the output that I needed. I also received a voltage reading of 4.87V by using a voltmeter as seen in my circuit diagram. Both of these mean that my initial specifications for the subsystem have been met.

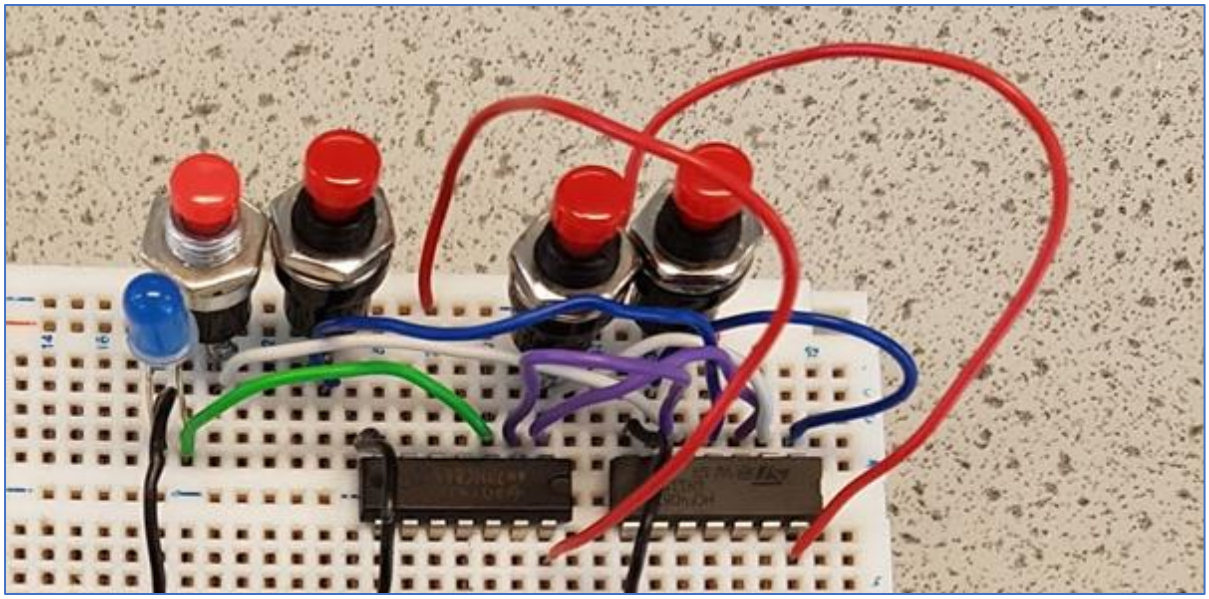

Figure 13: *Subsystem 3 built onto the circuit, utilising 1 4081 AND logic chip and 1 7408 OR logic chip.*

Considered Alternative Design

An Alternative Design for Subsystem 3		
Option	**Main Advantage**	**Main Disadvantage**
Wiring the panic buttons directly to the LED output.	The buttons turn the LED on as soon as one button is pressed.	An accidental button press is much easier to occur this way.

Figure 14: *A possible alternative design that I initially considered when first designing Subsystem 3.*

Panic buttons in real life are designed to require two buttons to be pressed in order to avoid an accidental lockdown, so I have incorporated this into my circuit in order to remain as true to my initial specification. I believed the disadvantage of a few extra nanoseconds this resulted in was insignificant compared to the advantage of greatly reducing the chance of an accidently press.

Evaluation of the Performance

This subsystem works completely as intended, and the final performance is as required by the initial specification. When both buttons in either set of buttons are pressed, the warning LED will turn on as desired. Improvements that could be made include adding a set reset latch, causing the LED to stay on after the buttons have been released. This would show the user that the police have already been called, without the buttons needing to be pressed again.

Final Subsystem Specification

- Both the AND and OR gates in the subsystem are supplied with 5V, as seen by my voltage reading of 4.87V.
- Each set of buttons in the subsystem turns on the output LED when all pressed, as seen by my testing results above.
- Only one set of buttons in the subsystem needs to all be pushed to produce the output, as seen by my testing results above.

Subsystem #4 - 16-Bit Binary Code Input

Circuit Diagram of the Subsystem

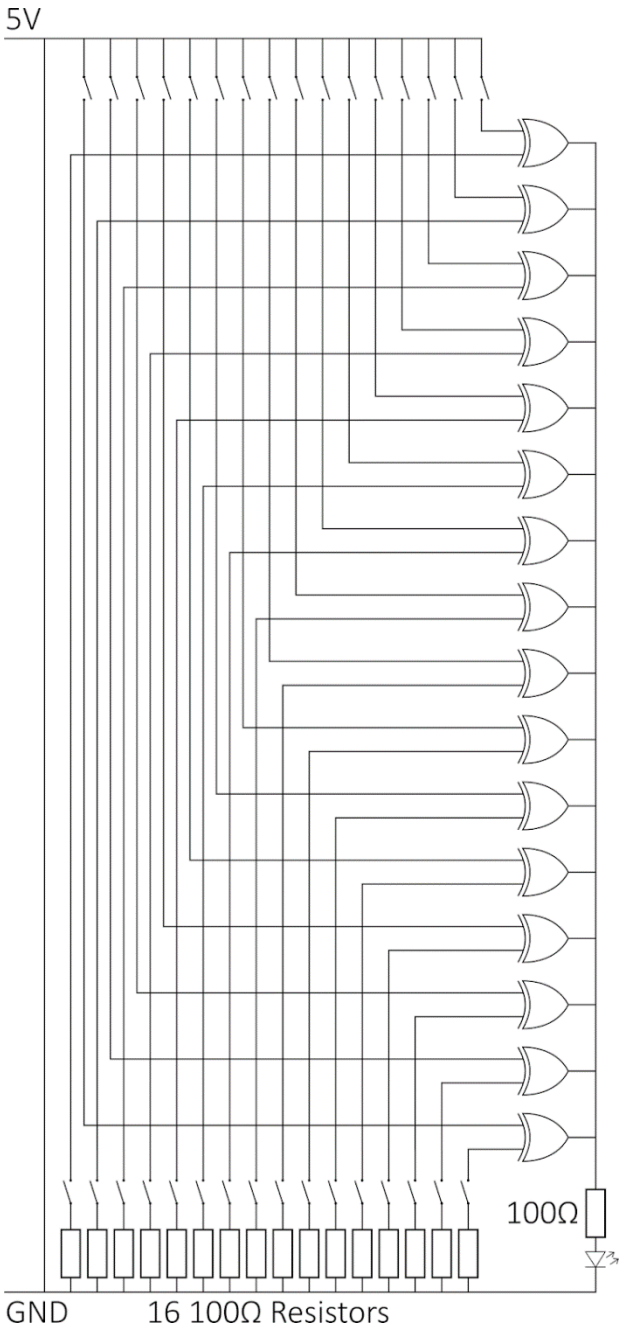

Figure 15: *The circuit diagram of Subsystem 4.*

Initial Subsystem Specification

- The subsystem should be powered by a voltage supply close to 5V DC.
- The subsystem should have a pre-set 16-Bit code, that is easily changeable by the user.
- The subsystem should allow the user to input a 16-Bit binary code.
- The subsystem should compare the pre-set code with the user's inputted code, causing an LED to turn on if the two codes are different.

Reasons for Using a 16-Bit Binary Code

For my project to be as secure as possible, I wanted a system that could easily be expanded upon, with a pre-set code that could effortlessly be changed at any time. Due to this, I decided to use two sets of switches that are then compared with each other via the use of XOR gates. If both the user's input switches and the pre-set switches are all matching, then no signal will be sent to the warning LED. I decided to use XOR instead of XNOR gates, so if just one of the user's input switches is wrong then a signal with be sent to the warning LED, rather than having to check each switch individually.

Pictures of Subsystem

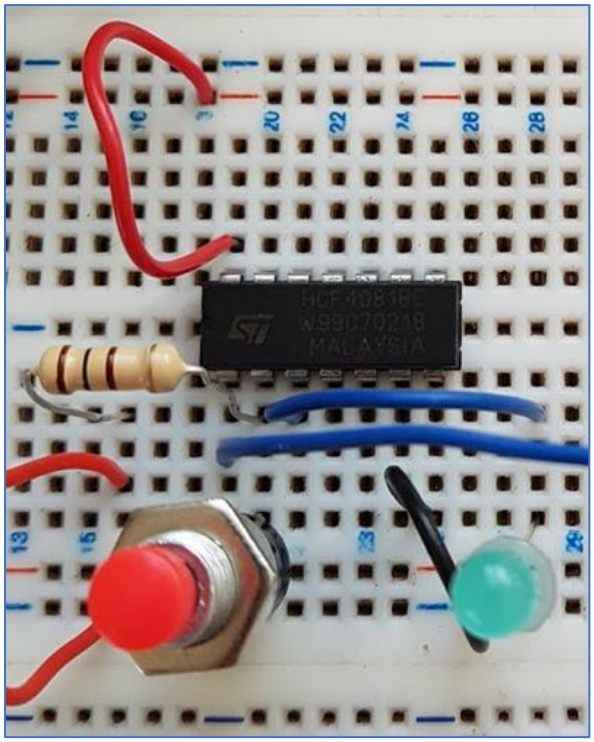

Figure 16: *The final LED output of Subsystem 4.*

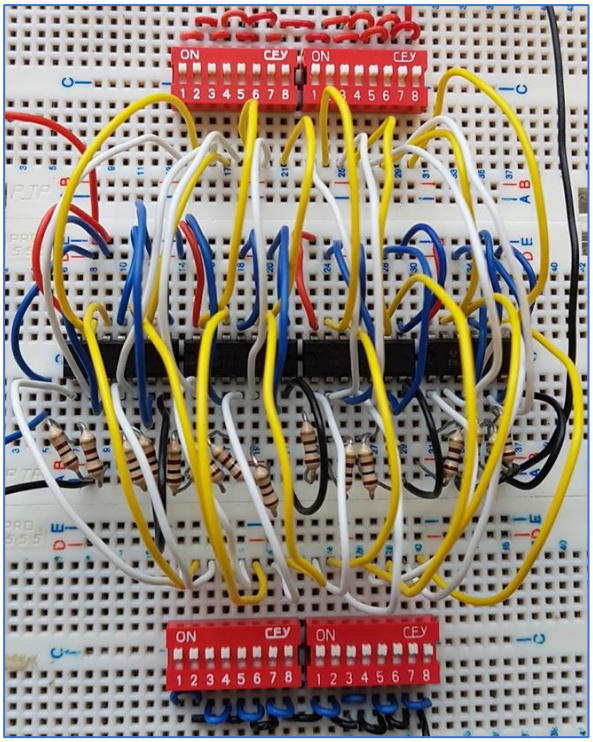

Figure 17: *The two 16-bit codes being compared.*

If the bottom 16-Bit user input does not match the top pre-set 16-Bit code as seen in **Figure 17**, then a warning LED turns on once the check button in **Figure 16** is pressed. The pre-set code can easily be altered by changing the values to 0 or 1, allowing a total possible combination of 65,536 (2^{16}) codes.

Test Procedures

Due to there being 65,536 different possible combinations, it would take far too long to test each of them individually. Therefore, I decided to test a specific sample of 36 chosen combinations, making sure all of the switches were tested. To start off, I tested the LED output when all 32 switches were off, as well as when they were all turned on. Next, I tested what happened when all 16 of the user's switches were turned off whilst all 16 of the pre-set switches were on and vice versa. Afterwards, I tested the output of the LED when only 1 matching switch was turned off for both the user's code and the pre-set code for all 16 switches. Lastly, I tested the LED when only 1 of the user's switches was turned off. Doing these specific tests would show if there were any problems that could occur in any of the combinations. I also set up a voltmeter to test the voltage supplied to the subsystem. All of my tests outputted the expected result, successfully achieving my initial specification.

Test Results

The Outputs of Subsystem 4					
User's 16-bit Code		**Pre-set 16-bit Code**		**LED Output**	
Switches On	**Switches Off**	**Switches On**	**Switches Off**	**Expected**	**Actual**
None	All	None	All	Off	Off
All	None	All	None	On	On
None	All	All	None	Off	Off
All	None	None	All	Off	Off
All except #1	Switch #1	All except #1	Switch #1	On	On
…	…	…	…	On	On
All except #16	Switch #16	All except #16	Switch #16	On	On
All except #1	Switch #1	All	None	Off	Off
…	Switch #1	All	None	Off	Off
All except #16	Switch #1	All	None	Off	Off

Figure 18: *Testing the outputs of Subsystem 4 to make sure that the LED's match the expected result.*

Considered Alternative Design

An Alternative Design for Subsystem 4		
Option	**Main Advantage**	**Main Disadvantage**
Using a combination lock with 4 buttons for the vault input.	More user friendly looking layout in comparison.	Only has 256 (4^4) possible codes to choose from.

Figure 19: *A possible alternative design that I initially considered when first designing Subsystem 4.*

Since I want my bank to be as secure as possible, I chose to go with my the 16-bit inputs, as I believe the layout cleanness is heavily outweighed by the greater number of possible combinations allowed.

Evaluation of the Performance

Comparing the final performance to the initial specification, the subsystem worked as intended. When the two codes matched, the warning LED would stay off, but when the codes were different, the warning LED would turn on as intended. One improvement could be a counter that counts all the user's inputs, and after 3 failed attempts, the subsystem would become unusable until being reset.

Final Subsystem Specification

- The subsystem is powered by a voltage close to 5V DC, with a recorded value of 4.59V.
- The subsystem has an easily changeable pre-set 16-Bit code, as seen in my circuit diagram.
- The subsystem allows the user to input a 16-Bit binary code, as seen in my diagram.
- The subsystem compares the pre-set code with the user's input, as seen in my testing.

Subsystem #5 - Set Switch and Reset Button

Circuit Diagram of Subsystem

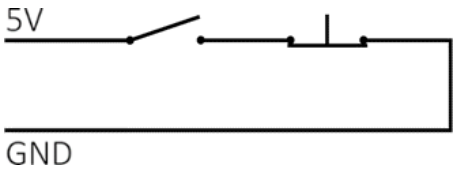

Figure 20: *The circuit of Subsystem 5.*

This final subsystem connects up all of the others in the circuit, delivering power to them all, representing the banks main power supply. Without the master switch turned on, none of the other subsystems are able to function. If the push to break button is pressed at any time, this subsystem will then stop sending power around the entire circuit.

Initial Subsystem Specification

- All of the subsystems should function as intended if the master switch is on.
- None of the other subsystems should function if the master switch is turned off.
- Power should stop being delivered around the circuit if the reset button is pressed.

Reasons for Using a Set Switch and Reset Button

I decided to use a switch that controlled all of the subsystems, in the event that a change had to be made to one of the security systems. For example, if this system were reproduced for a real bank, there would need a way to turn off the alarm even when the building has power supplied. I decided to use a switch for the main power supply, so that the user would not need to continually press it.

Picture of Subsystem (Top Left Corner, Supplies Power to the Entire System)

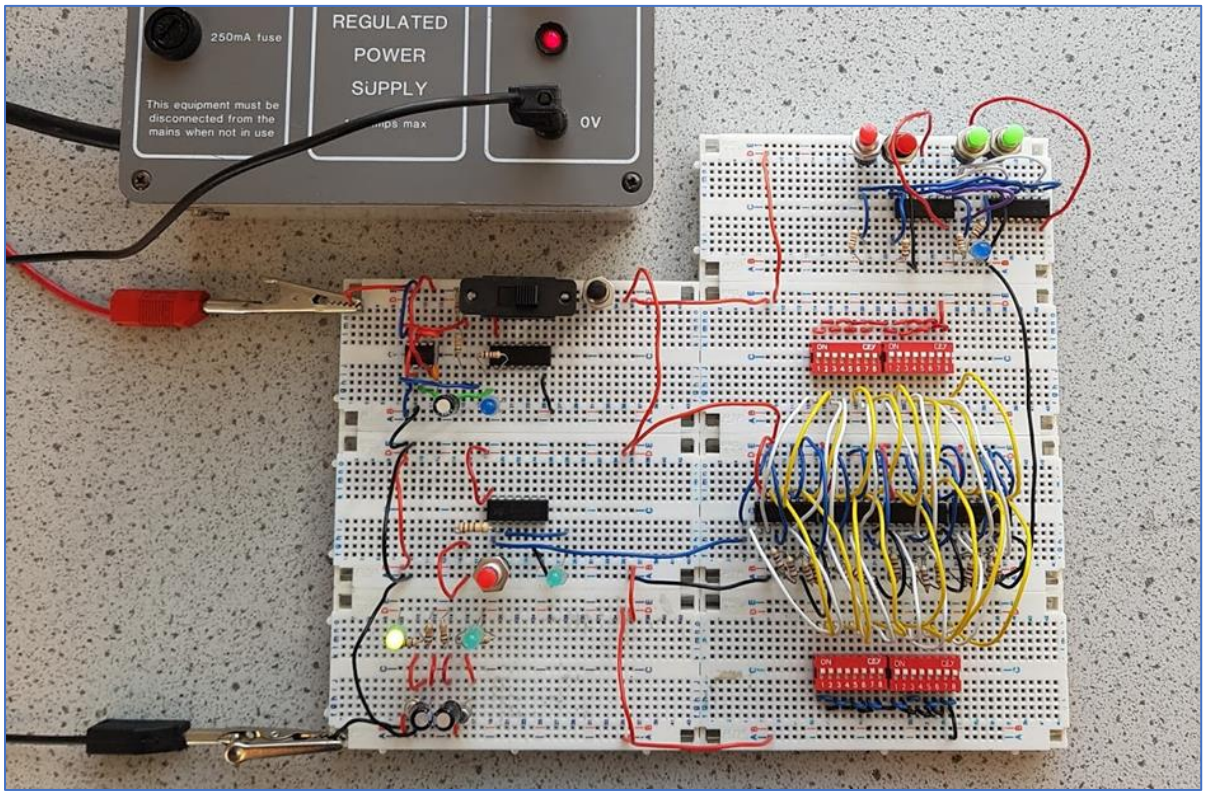

Figure 21: *The final subsystem in the circuit, located in the top left, powering the other subsystems.*

Test Procedures

Once I had connected my master switch up to the other subsystems, I supplied power to the entire circuit, to see if the other subsystems would still work and behave as intended. I also tested the push to break button, to make sure it stopped any power from reaching any of the subsystems. Similar to my other subsystems, I set up a voltmeter as seen in my circuit diagram. I did this to test the voltage supplied to each subsystem. As seen in **Figure 22** below, each subsystem still worked as intended, matching my initial specification of having approximately 5V when powered, and 0V when not.

Test Results

The Outputs of Subsystem 5				
Subsystem Tested	**Master Switch On**		**Master Switch Off**	
555 Timer	4.86V	0.00V	0.00V	0.00V
Dual Flashing LEDs	4.91V	0.00V	0.00V	0.00V
Logic Gates Input	4.87V	0.00V	0.00V	0.00V
16-Bit Code Input	4.59V	0.00V	0.00V	0.00V

Figure 22: *Testing the outputs of Subsystem 5, and how it affects the other subsystems in the circuit.*

Considered Alternative Design

An Alternative Design for Subsystem 5		
Option	**Main Advantage**	**Main Disadvantage**
Using multiple buttons inputs to control power delivery.	Could allow greater control when a user is present.	Requires a user to keep the circuit supplying power.

Figure 23: *A possible alternative design that I initially considered when first designing Subsystem 5.*

Having to hold down a button in order to use the rest of the circuit would have been very tiring, as a single release of it would stop the entire circuit from working. Therefore, I chose to use a switch so that once the power was turned on it stayed on without any human interaction needed, until the user either turns the switch off or presses the push to break button.

Evaluation of the Performance

The final performance is as required by the initial specification, as the subsystem supplies power to all of the other subsystems. When the switch is turned on, then 4.59V to 4.91V is supplied to all the various subsystems. When the switch is turned off, or when the reset button is pressed, then power will no longer be delivered to the circuit, as I intended to occur.

Final Subsystem Specification

- All of the subsystems function as intended when the switch is on, as seen in my tests.
- None of the other subsystems function with the master switch off, as seen in test results.
- Power stops being delivered when the reset button is pressed, as seen in my test results.

Evaluation of the Entire System

Overall, I am very satisfied with the performance of my system as I have met my initial design brief. The performance of the entire system is very reliable, as seen in all the previous tests, matching each initial specification set for each subsystem. All of my subsystems work both individually and when integrated together, as seen by my testing below. When power is delivered from Subsystem 5, the rest of the circuit behaves as intended. The 555 timer begins to tick, the dual flashing LEDs begin to flash alternately, the panic buttons produce an output to an LED when a set of buttons are pressed, and another LED turns on if the 16-bit user input is incorrect. I have also met my initial specifications, adapting several along the way when necessary to do so to make the system more secure.

Calculations for the Entire System

First of all, I tested to see how much of the 5V power supply was actually being supplied, which was 4.98V. I then tested to see how much current was being provided, which I recorded to be 2.10A. This means that I could calculate my total power usage of my circuit, which is 10.458W. This is very low for security system, which means that more power would likely need to be supplied if more advanced security checks were added. Furthermore, I tested to see if the circuit or power supply increased in temperature after turning in on, and after leaving it on for 20 minutes I saw no noticeable difference. This is a good result as the circuit would not be very reliable if it got hot after a long period of time, as a security system needs to always be on.

Test Procedures for the Entire System

In order to see if all my subsystems worked together, I set up multiple voltmeters at the same time as shown in my previous test procedures, and turned on the master switch (Subsystem 5) to see if the voltages matched the same readings when the subsystems were separate. I also decided to test the time periods of the 555 timer and the dual flashing LEDs with a stopwatch to see if they were the same as when I tested them separately. Finally, I decided to test if the panic buttons and the 16-bit binary code worked when connected to the other subsystems.

Test Results for the Entire System

The Outputs of Subsystem 6	
Subsystem Tested	**Voltage Received**
555 Timer	4.86V
Dual Flashing LEDs	4.91V
Logic Gates Input	4.87V
16-Bit Code Input	4.59V

Figure 24: *Testing the voltage output of each subsequent subsystem whilst Subsystem 5 is activated.*

When the 555 timer receives power, it begins counting for 13.42 seconds. Afterwards, the capacitor starts to discharge for 1.26 seconds, sending 4.86V to an LED to tell the user it is time for a routine security check, before the cycle starts once again. Once the second subsystem receives power, it turns on two alternating flashing LEDs, each with a mark and space time period of 0.964 seconds.

The panic buttons can be pressed to call for backup in the event of an emergency, represented by a warning LED, so long as the master switch is turned on and delivering 4.87V. These can be pressed while the rest of the circuit is running, and it has no negative impact on the voltage being received by the rest of the subsystems. If both the master switch is switched on and the 16-bit user input is incorrect, a warning will be sent to security if the button input is pressed, represented by an LED.

Comparing Initial Specifications and Final Performance

The initial and final specifications agree that:

- The system has an LED with a mark to space ratio that is extremely close to 14:1, tested to be 13.42:1.26, which is then inverted to give a brief 1 second flash every 15 seconds, as seen in my circuit diagram for Subsystem 1.
- The system has an alarm made out of two LED's, both with time periods of 1 second and mark to space ratios of 1:1, with a measured value of 0.964 seconds. flashing out of phase with each other.
- The system has a secure way to lock the vault, that triggers the dual flashing LED's representing the alarm if the wrong combination is inputted, as seen in my circuit diagram for Subsystem 3.
- The system has a pre-set code that is easily accessible and changeable, without having to change the wiring of the circuit, as seen in my circuit diagram for Subsystem 3.
- The system can be reset the circuit without being turned off, as seen in Subsystem 5.
- The system has a way to turn the rest of the circuit off, even when there is a connected power supply that is turned on, as seen by my testing for Subsystem 5.

The initial and final specifications did not agree that:

- The system should activate the alarm when only one panic button is pressed:
    - This was changed to 2 sets of buttons, both containing 2 buttons. Only one set needs to be pressed but both buttons are required to be pushed together. I decided to go with this alternative design because panic buttons in real life are designed to require two buttons next to each other to be pressed at the same time, in order to avoid an accidental lockdown, which is a problem I stated earlier in Subsystem 3.

Future Improvements

To improve my circuit even further, one possible change I could make is to make my panic buttons activate the dual flashing LEDs instead of the power supply. Another change I could make to my project is to add more switches to the 16-bit binary code, to make it even more secure than it already is. I could also add a voltage amplifier to the output of the 16-bit binary code, so that the LED lights up brighter and is easier to see by the user. Furthermore, I could have replaced my master switch with a key lock, so that only an authorised person with a key is able to turn the security system on and off. Another improvement would be to make my time periods closer to the initial specification, as I could have used multiple resistors in series in order to gain a more accurate value. For example, in my 555 timer I aimed to have a mark space ratio of 14:1 but ended up with a value of 13.42:1.26. While this was a good result to get with the resistor values available to me, I could have achieved an even closer value by using this method.

Reflection

Upon reflection of my work, I believe I have expanded on my knowledge of how data is transferred. While I was able to understand the logic behind the Start and Stop protocol itself, I had some trouble utilizing it in my code to begin with; developing the flow charts however made this far easier. I made sure to test my code in order to make sure that no bugs creeped up, using the flowcharts to outline the logic of my program to aid me in visualising the steps when coding the task. If I were to attempt a similar assignment in the future, I would definitely create flowcharts again, as they helped me to visualise my code. I would also research further how an Arduino could be utilised, in order to further my understanding of utilising WI-FI adapters, potentially finding a way to connect it to a Raspberry Pi for example. Overall, I am happy with the knowledge I have gained from this assignment.

References

Storr, W. (2017, March). *Astable Multivibrator*. Retrieved from Electronics Tutorials: https://www.electronics-tutorials.ws/waveforms/astable.html

Appendices

Appendix A: The Full Circuit Diagram