

RJ assessment final

by Rhys Jones

Submission date: 05-Dec-2025 02:43PM (UTC+0000)

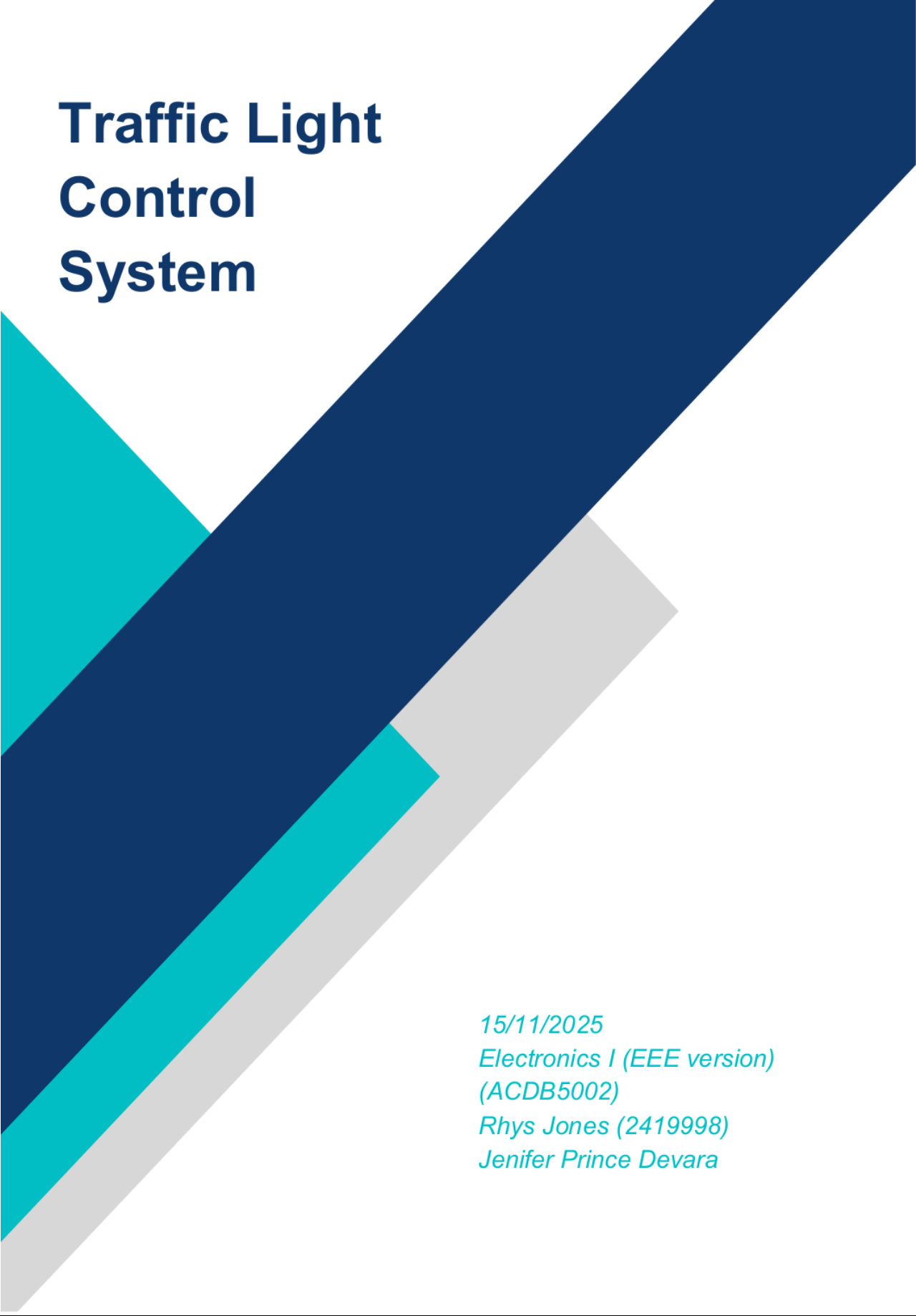
Submission ID: 269757073

File name: 58084_Rhys_Jones_RJ_assessment_final_634321_80471519.docx (1.23M)

Word count: 2409

Character count: 11274

Traffic Light Control System



15/11/2025
Electronics I (EEE version)
(ACDB5002)
Rhys Jones (2419998)
Jenifer Prince Devara

1 INTRODUCTION

1.1 Description of the assignment

The task given was to make a traffic light control system and would also need a timer display that operates with precise timing intervals. These traffic lights would be on a T-Junction, and the traffic lights would need to be efficient and prioritize safety. There needs to be a pedestrian crossing section that lets pedestrians cross the road safely.

1.2 Breaking down the problem

Breaking down the assignment further I would need to pick a microchip capable of handling all of the traffic lights and would be able to handle the 7 segments as well. As shown in figure 1 the assignment asks for 7 segment displays on each side of the traffic lights and because it is a 2-digit number there needs to be six 7 segments displays. There needs to be 3 traffic lights in the circuit as this is on a T-Junction which has 6 lanes and 3 of these are going to need the traffic lights on them. The pedestrian crossing is going to need 4 lights per crossing because there needs to be a green and a red signal each side. With 4 lights on each crossing there is going to be 12 lights on the Pedestrian crossing alone. If every light uses 1 I/O on the Microchip the total is 63 (see Fig 2) which would make our Microchip quite large in size. Since some of the lights would be doing the same action e.g. the 7 segments and the pedestrian lights then it can be cut down to 25 I/O pins (see Fig 3).

Design a traffic light control system for a T-Junction. This system design aims to enhance traffic management efficiency while prioritizing safety, making it suitable for a T-Junction.

- (a) Install three traffic signal lights on each side of the road for comprehensive traffic management.
- (b) Include a set of 7-segment displays on each side to count down from 40 to 0 seconds.
- (c) Configure the system to change traffic signals every 10 seconds for predictable and smooth traffic flow.
- (d) Reserve the final 10 seconds of the signal cycle for a dedicated pedestrian crossing phase. Install LEDs to display Pedestrian Green and Red signals.
- (e) During the pedestrian crossing phase, program the traffic signals to either turn off or display a red signal for all three roads, ensuring pedestrian safety.

Fig 1

Fig 2

	Pin to micro	number of	Total		
7 seg	7	6	42		
ped lights	1	12	12		
Traffic lights	3	3	9		
			63	Total of I/O pins	

Fig 3

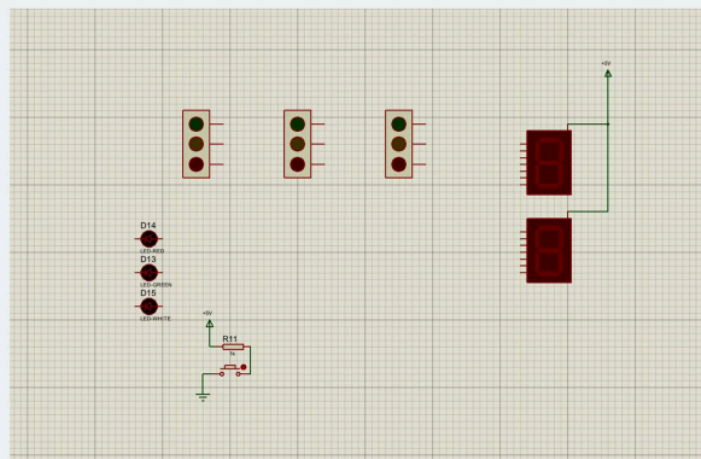
	Pin to micro	number of	Total		
7 seg	7	2	14		
ped lights	1	2	2		
Traffic lights	3	3	9		
			25	Total of I/O pins	

2 BACKGROUND RESEACH

2.1 THE COMPONENTS WHEN WORKING WITH EACH OTHER

As shown in Fig4 these are how many pins will be used in the circuit. There is a pull-down button included the circuit as I decided to use an external interrupt to make a pedestrian wait light signal display when the button is pushed. Seeing all of the components next to each other made it easier to visualise how it will all work together such as having separate Ports for the 7 segments and then what pins the components can attach to and still work without interfering with others.

Fig 4



2.2 CHOOSING A MICROCHIP

A Microchip is not something you can choose lightly because it needs to be suitable for purpose when inserted into a circuit. Microchip the company offers a data base of all of their chips [1] where you can set the parameters and choose a suitable Microchip easily based on what kind is needed in the circuit. In Figure5 it shows the parameter used to choose the Microchip. The specific microchip chosen was a PIC16F877A because it was a Microchip I was familiar with and had used before but any of the other Microchip could have been used as well.

Fig 5

Product			CPU Speed Max MHz (megahertz)	I/O Pins Max
pic16F			1 - 20	4 - 160
☐	PIC16F914	  	20	36
☐	PIC16F917	  	20	36
☐	PIC16F74	  	20	33
☐	PIC16F747	  	20	36
☐	PIC16F77	  	20	33
☐	PIC16F777	  	20	36
☐	PIC16F877A	  	20	33

2.3 HOW TRAFFIC LIGHTS WORK

Traffic lights are something everyone uses everyday and even though I drive I still wanted to double check that I had the right sequence setup for when I was writing my code. Using the Governments [2] website I checked how the traffic signals worked just to double check that it was the same as what I remember. The Traffic light sequence goes as follows and is shown in Figure 6

1. Red
2. Red and Amber
3. Green
4. Amber
5. Red

This all happens at different times in the real world as different lights might have different rates of traffic. In the circuit this sequence will happen every 10 seconds on the different lights just for convenience and to stick within the 40 second time limit on the assignment.

Fig 6



3 CIRCUIT DESIGN

3.1 CIRCUIT DIAGRAM AND COMPONENTS NEEDED

This was the final Circuit diagram (Figure 7) Which includes all of the resistor blocks and the transistors needed to run the LEDs as 5V is too small a voltage to run all of the voltage of LEDs. There is a crystal in the top left set to 8MHz which is my chosen clock speed to run the PIC16F877A as there is no internal clock on the Microchip itself to set the speed. There is a +5V connection on the master clear pin as there is no option to disable it on the PIC. The interrupt button needs to go on Pin_B0 as it is the only pin that accepts external interrupts. The 7 segments work by grounding the 7 segment to the microchip when the light needs to be turned on because they are common anode. To turn off a segment on the 7 segment display a voltage is applied to the lights that are wanted to be turned off as this pushes the same voltage in both directions so no current can flow.

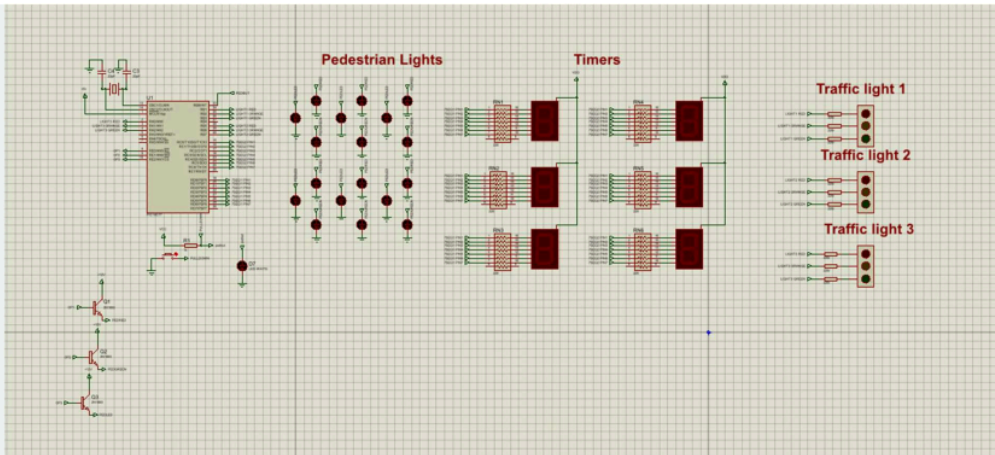


FIG7

4 SIMULATION BUILD

4.1 Proteus design

This was the graphic made on proteus to display where all of the timers and lights would be in the T-Junction (Figure 8). Also, this graph shows the external interrupt buttons for the pedestrian wait light. Together this system works to keep people safe by having an easy-to-understand sequence where it goes in a clockwise motion on the traffic lights and then when it finishes it lets the pedestrians cross. The zebra crossings are there to make sure the drivers don't block the way and make it harder for pedestrians to cross I chose values of 220 ohm resistance on the LEDs and the 7 segments as to when it came to building it physically I wouldn't accidentally put to much current through any of the components.

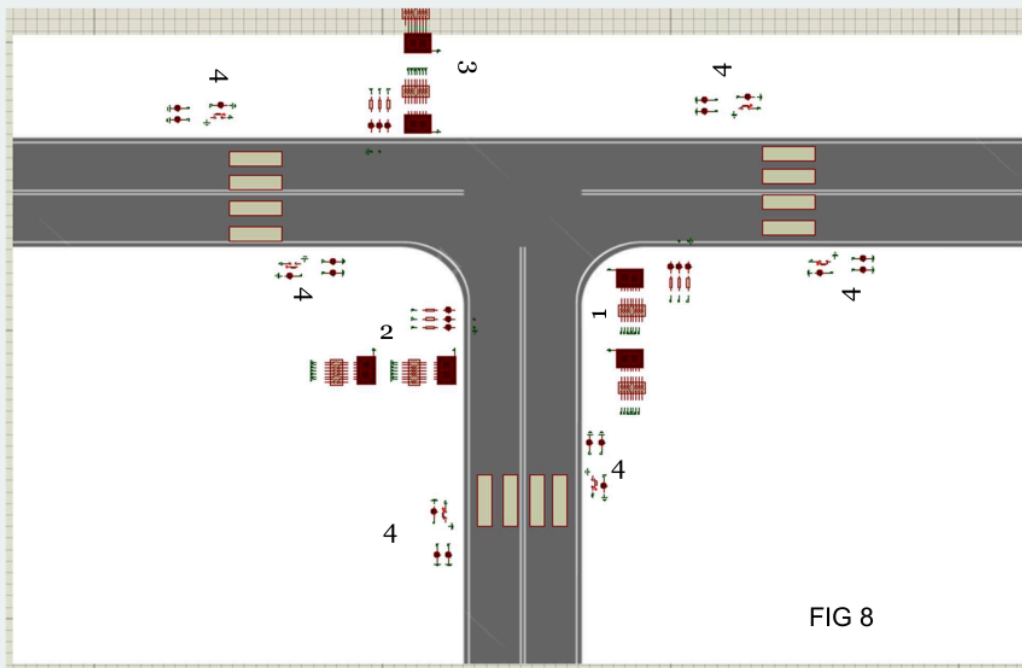


FIG 8

programming due to less bugs and a better architecture in general. The next step in putting the PIC onto the breadboard was to program the PIC using a pickit 3 as shown in (Figure 10).

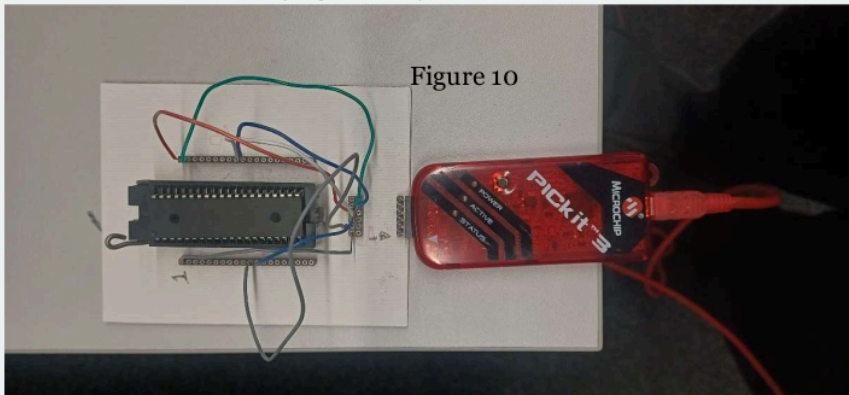


Figure 10

The final step was to connect all the components to the PIC16F877 (figure 11). It doesn't look very nice but the traffic light system worked but not the timers. I later looked at the specification sheet again for the PIC16F877A which is the same one for the PIC16F877 [4] and found on page 173 that the chip can only support 200mA on PORT C and PORT D which is where my 7 segments are. The two 7 segment displays were sinking between each other 318mA because of the 220ohm resistors. Increasing the resistance of the 7 segments to 470 ohms was the solution which then gave me a current of 148mA which the PIC16F877 was able to handle and then made it all run smoothly.

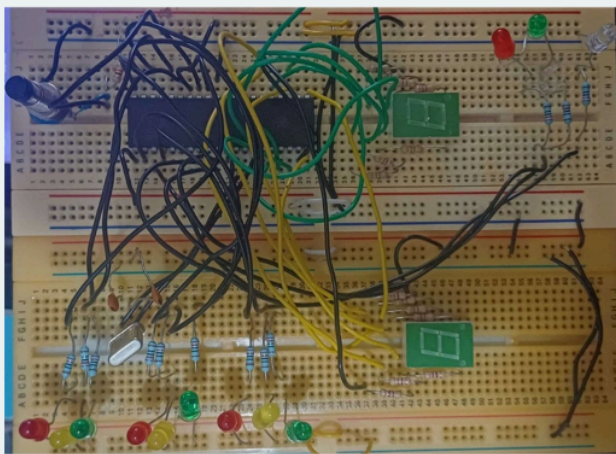


FIG 11

6 PSUEDO CODE

Firstly, the code does “#include <Main.h>” which then sets up the clock speed of 20MHZ and includes the specific PIC used in this case PIC16F877

Then the code sets up the routines which will be called upon later by main

```
void traffic_light1(void);  
void traffic_light2(void);  
void traffic_light3(void);  
void Peds(void);
```

Next, I set an unsigned integer called lights to equal 1

After that I used #INT_EXT which sets up the external interrupt pin(B0) on the PIC

Then is the routine for the external interrupt which checks to see if the variable lights is == 2 which checks if its equal to 2 because if you only put 1 equals sign it sets it to 2.

If the check comes back and “lights” is equal to 2 then the program will set Pin_e1(the pedestrian wait) light to High.

Else if the check is not equal to 2 the light remains low until prompted otherwise.

Next is the main part of the program called “Main” this is the most Important part of the coding because it is the starting point. The first thing in the main program is to check if the Microchip is running and has initialized correctly. To test this the program starts by setting Pin_E2 high and setting the 7 segments high as well. Then there is a 500-millisecond delay and then everything set high goes low. This is repeated twice so that a flashing occurs in the circuit to make it obvious for the user.

After that we set the external interrupt to work when the pin goes from high to low using the external interrupt button. Then we clear the interrupt on the external and enable global interrupts and any interrupts coming to the external interrupt flag.

Next the code populates the 7 segment displays with 4 and 0 to give the start of the timer.

Then the code sets Ports A B and E low to stop them from floating.

After that is the while(true) loop which is an infinite loop keeping the code going forever until there is a loss of power or an error

The while loops first step is to do the traffic_light1 routine and then continue to do the other 3 until it loops back to the first routine again.

Now we move onto the main coding section which is the 4 routines controlling the lights.

The first second and third routine are identical except for the output on Port D.

The routines set Port D to the desired 10second digit then set Port C to display 9.

Then it sets the specific amber light to the routine to turn on for the first routine it would be the amber light for the first sequence.

Next there is a delay of 800 milliseconds

After that the delay the output goes high for the green light

Followed by another delay of 200ms

Then the code displays the value 8 onto Port C.

This continues with 1000 milli seconds delay with the values counting down on Port C until Port C displays 2.

Next the Green light is set to turn off and the amber light turns back on. Then there is a delay of 1000 milliseconds and the numeric value of 1 is put on Port C.

After that there is another delay of 1000 milliseconds and then the light goes back to red

Then the numeric value 0 is displayed on Port C and another 1000 millisecond delay happens and then the next routine and traffic sequence starts.

This happens 2 more times and then the Pedestrian routine happens.

Next is the last routine where the pedestrian lights start.

Followed by that the integer "lights" is set to 2 so that if the external interrupt is pushed then the wait light will not turn on.

Port D is now displaying the value 0 and Port c goes through its final countdown

After that the green pedestrian light is turned on and the red pedestrian light is turned off.

Port C count downs until the value of 1 is displayed on the 7 segment

Next the green light turns off and the red light turns back on for the pedestrians and port C hits 0 which then has a delay of 1000ms.

Finally, everything has hit 0 so the cycle restarts by setting Port D to display 4 and lights is set to 1 so that when the loop restarts the external interrupt button works.

The loop then continues until there is either power loss or until the circuit is turned off.

7 DISCUSSION

I found this project to be very interesting and helpful in learning about data sheets and how to turn a simulation into a physical project. I need to make sure that I am not relying on the simulation to calculate if all of my resistor values are correct and to double check the currents I am putting into the Microchips in future projects. If I had to redo this project again, I would like to try and build the entire circuit with all 6 of the 7 segments and all of the pedestrian lights to see how it affects the current draw of the circuit and ways to combat having that many components needing voltage. I would also like to try and use Microchips I am unfamiliar with as to be able to learn how to read data sheets better and not have some of the information memorised. The results from this project made me very happy as I have very little errors and if there were any its just from silly mistakes.

8 CONCLUSION

In conclusion the project was a success and going forwards I know where to improve and what to be more careful on. Improving on what I know is a step going forward and I learnt a lot from this assignment. Next time using a PCB would be easier than using breadboard but that will require more time on easy pc and using different software to get the PCB to be made.

References

- [Microchip, "Parametric search," Microchip, [Online]. Available: <https://www.microchip.com/en-us/parametric-search/980?filters=JTdCJTlyY2F0ZWdvcnlkcm9wZG93biUyMiUzQSU1QiUyMk1pY3JvY29udHJvbGxlcuMIMjBBbmQIMjBQcm9jZXNzb3JzJTlyJTJDJTlyQWxsJTlwTWljcm9jb250cm9sbGVycyUyMGFuZCUyME1pY3JvcHJvY2Vzc29ycyUyMiUyQyUyMiUyMiU1RCU3RA==>. [Accessed 24 October 2025].
- [Unknown, "GOV.UK," Department for Transport, 22 October 2025. [Online]. Available: <https://www.gov.uk/guidance/the-highway-code/light-signals-controlling-traffic>. [Accessed 24 October 2025].
- [A. / . Hewlett-Packard, "5082-7730 - Agilent / Hewlett-Packard - Datasheet.Live," [Online]. Available: <https://www.datasheet.live/pdfviewer?url=https%3A%2F%2Fpdf.datasheet.live%2F45dcd5df%2Fagilent.com%2F5082-7730.pdf>. [Accessed 17 11 2025].
- [M. T. Incorporated, "PIC16F877A," [Online]. Available: <https://ww1.microchip.com/downloads/aemDocuments/documents/MCU08/ProductDocuments/DataSheets/39582C.pdf>. [Accessed 17 11 2025].