



EMBEDDED C

ACDA 4013

Jenifer Prince Devaraj

Dafydd Adams

2320108

Table of Contents

List of Figures	ii
Introduction	1
Background Research.....	1
Circuit Design	2
Flowchart	3
Screen Dumps	4
Testing Details.....	7
Conclusion.....	8
References	9

List of Figures

Figure 1 - Proteus Circuit Design	2
Figure 2 - Flowchart of Engine System Program	3
Figure 3 - System Ready	4
Figure 4 - Engine Started	4
Figure 5 - Coolant OK	5
Figure 6 - Fan Activated	5
Figure 7 - Alert	6
Figure 8 - Shutdown Done	6
Figure 9 - Reset	7
Figure 10 - While Loop	7

Introduction

This project uses a PIC 8 bit Microcontroller (PIC16F877A) and Proteus, the task is to design an automotive cooling system status simulator using three buttons to simulate Engine Start (SW1), Coolant Sensor Trigger (SW2) and Fan Activation (SW3) with each switch triggering the next stage of operation when pressed in the correct order. Use three LEDs to indicate the system status with LED1 for Start Signal, LED2 for Coolant Confirmation and LED3 as an Alert Indicator. A motor is used to simulate a radiator fan and should be controlled via 2 output pins to simulate forward operation. Once the fan operation cycle is complete blink the alert LED three times with a 100ms interval indicating the system is moving to shutdown. All system messages must be displayed on an LCD Module (connected to PORTD in 4-bit mode) such as:

- System Ready
- Engine Started
- Coolant OK
- Fan Activated
- Motor Running
- ALERT!
- Shutdown Done

Each message must also be sent via UART to a Virtual Terminal within Proteus for real-time system monitoring.

Background Research

The components used within this system include the PIC16F877A microcontroller which is an 8 bit controller with 40 or 44 pins, this project uses the 40 pin version. [1] This microcontroller operates at 20MHz and has 5 Input / Output (I/O) Ports A-E. It supports serial communication via Universal Synchronous Asynchronous Receiver Transmitter (USART) and Universal Asynchronous Receiver Transmitter (UART) [2] which is used for the Virtual terminal in Proteus but can also support Inter Integrated Circuit (I²C) and Serial Peripheral Interface (SPI). This microcontroller is widely used for Embedded Projects due to the low power consumption and ease of use and can be used for motor control [3], robotic arms [4] and Water Quality Monitoring Systems [5] to name a few uses. Sensors can be used in conjunction with the PIC with temperature sensors, ultrasonic sensor and light dependant resistors which can be used to create a smart system that can adapt light output dependant on the light levels within a room [6]. Another use that aligns with the current project is using the PIC to control windscreen wipers, headlight automation and smart window features [7] [8]. Another paper looked at using a PIC for a cooling system utilising LM35 temperature sensors to control motors using pulse width modulation (PWM) to improve power management, feedback was provided through LED's and an LCD display which is similar to this project [9].

The LCD Module is used within this circuit to display the messages; to minimise the number of wires required it is connected in 4-bit mode using 4 pins as data transfer as opposed to 8 and 3 pins for Register Select (RS), Read / Write (R/W) and Enable (E). The remaining pins are used for power and backlight. A variable resistor (10k) is used on the power connections to control the contrast pin (V_0) by acting as a voltage divider allowing the contrast of the LCD to be changed as required to aid readability [10]. A crystal oscillator is used in conjunction with 2 capacitors (22pF) to provide a highly accurate clock for the PIC [11]. This is done by using the mechanical resonance of the crystal and measuring the frequency produces and using it as a time reference, the two capacitors must be chosen carefully to ensure correct operation and the desired frequency is produced [12]. To

replicate forward operation the motor is controlled via a normally open relay which controls the power supply to the motor as is widely done in industry due to the current required to power motors being much larger compared to the control loop. The remaining components are 4 push buttons and 3 LED's, 2 Green and 1 Red which are used to provide feedback during system operation.

Circuit Design

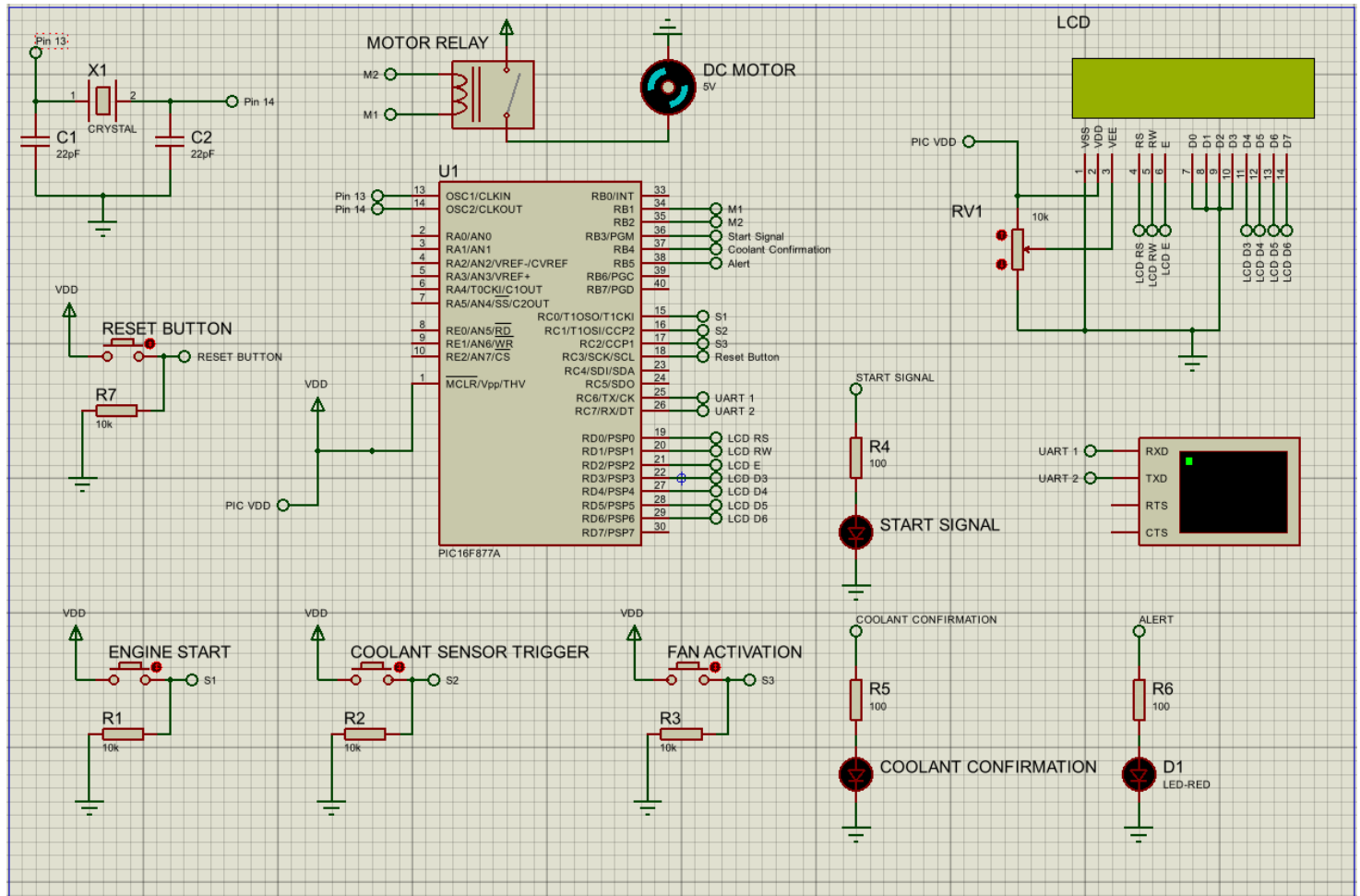


Figure 1 - Proteus Circuit Design

The above is the circuit designed on Proteus with the supply voltage, usually 5V for a PIC 16F877A, connected to port 1. Above this the Crystal Oscillator, connected to pins 13 and 14 along with the 2 capacitors. Ports RB1 and RB2 are used to connect the relay controlling the motor. Ports RB3, 4 and 5 control the 3 LEDs in the following order, Start Signal, Coolant information and Alert, all LEDs have a resistor in series to avoid damage. Inputs are then connected to RC0, 1, 2 and 3 in the following order. Switch 1, 2, 3 and Alert Switch which also have a resistor to avoid damaging components. The UART connection is then attached to RC6 (TX) and 7 (RX). Finally, the LCD is connected in 4 bit mode to RD0 – RD6. As opposed to having wires connecting all relevant ports and components together for the simulation small pins have been used and labelled ensuring the diagram is easy to read.

Flowchart

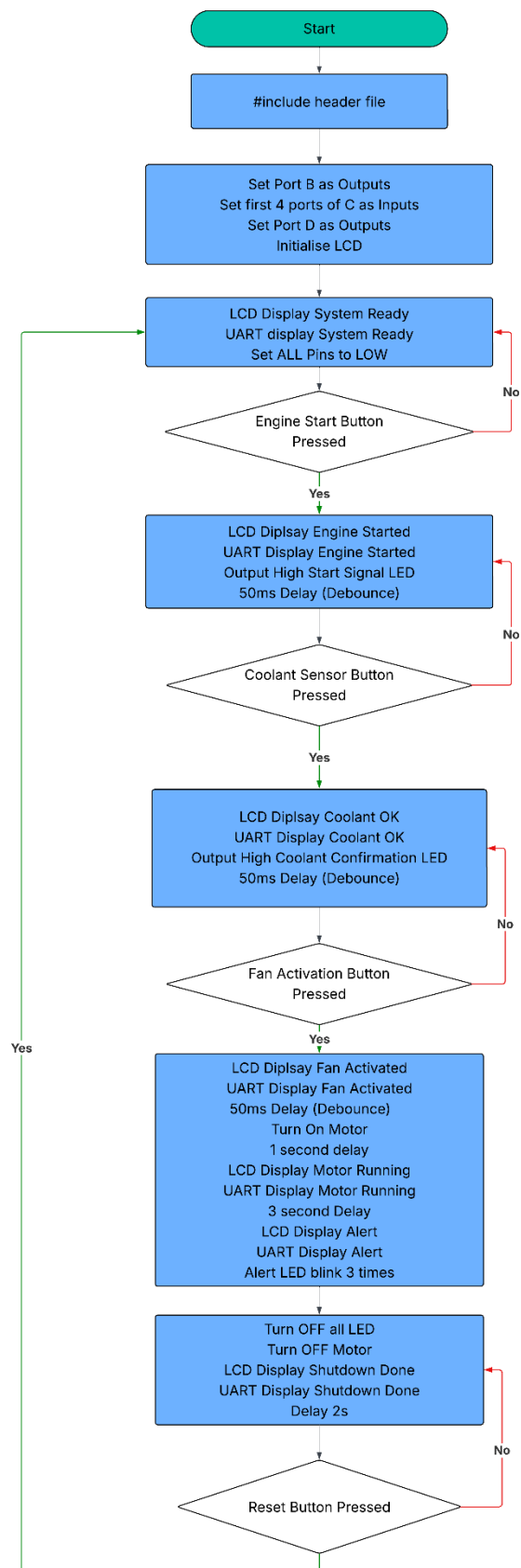


Figure 2 - Flowchart of Engine System Program

Screen Dumps

After building the system in Proteus and writing the program as the flowchart does there were slight changes made to the code discussed in the testing section. The first screenshot below shows the system ready with both the LCD display and the UART shown.

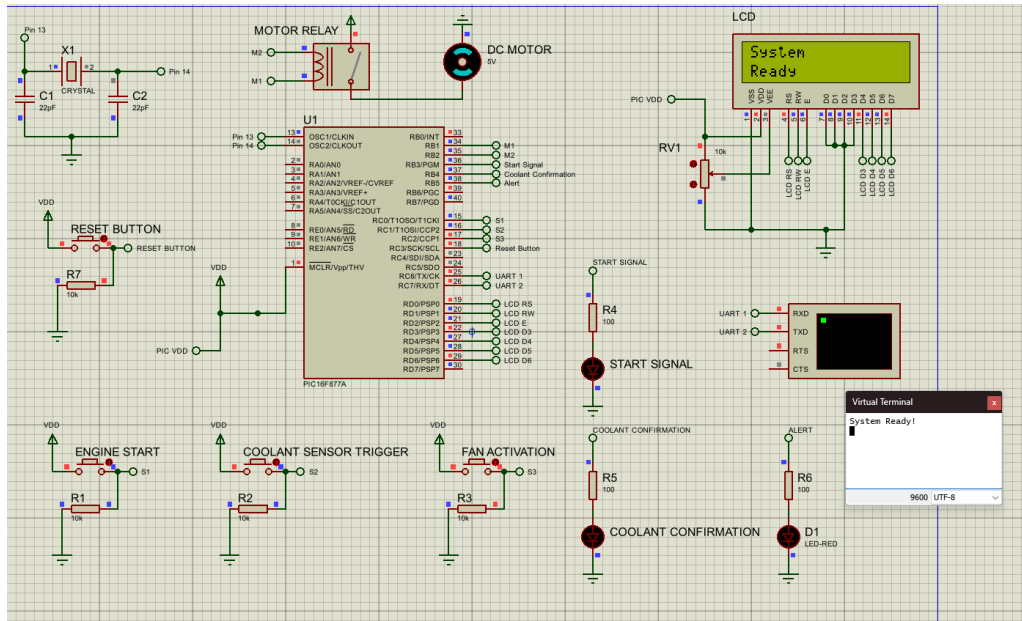


Figure 3 - System Ready

The next screenshot, figure 4, shows after the engine start button has been pressed with the LED illuminated and the message updated to say engine started.

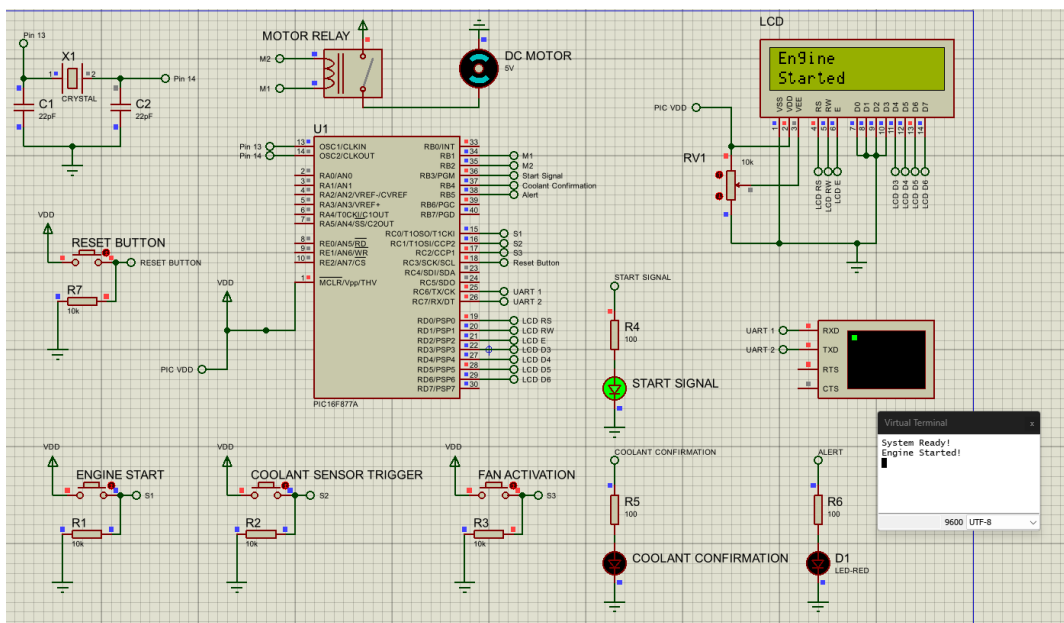


Figure 4 - Engine Started

Once the coolant sensor trigger is pressed the LED illuminates and the LCD and UART display are updated accordingly.

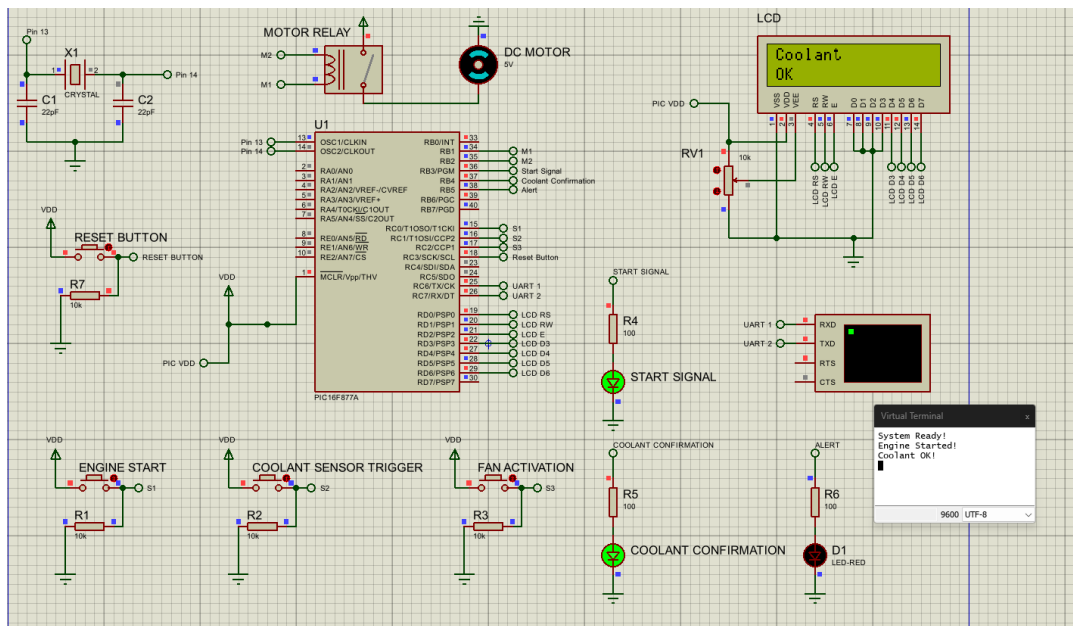


Figure 5 - Coolant OK

Once the fan is activated the motor can be seen turning, the outputs on the relay also show that there would be current flowing and pulling in the contact to activate the fan.

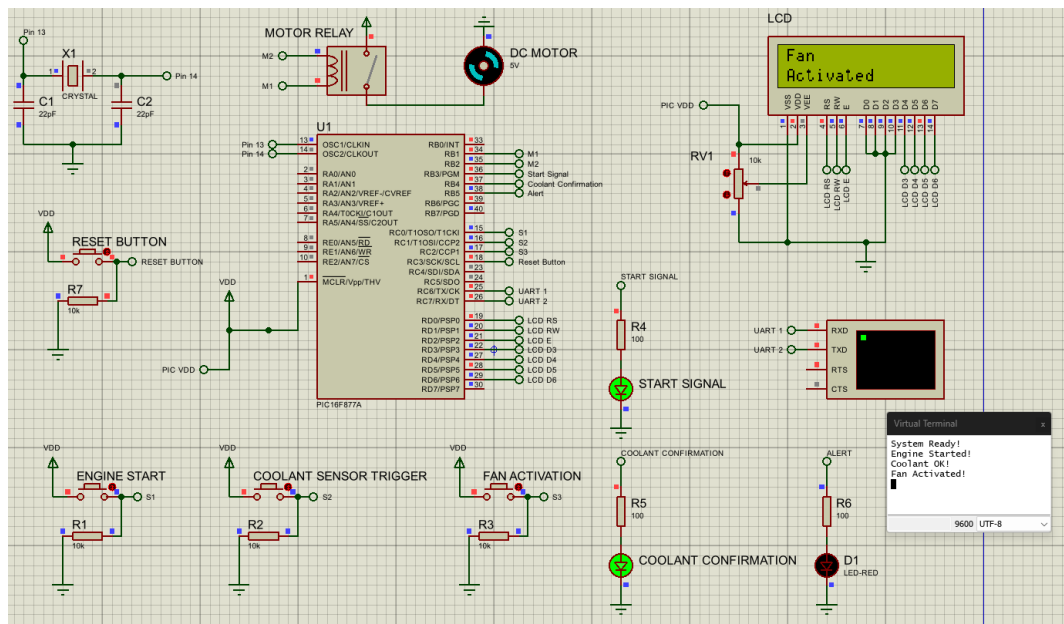


Figure 6 - Fan Activated

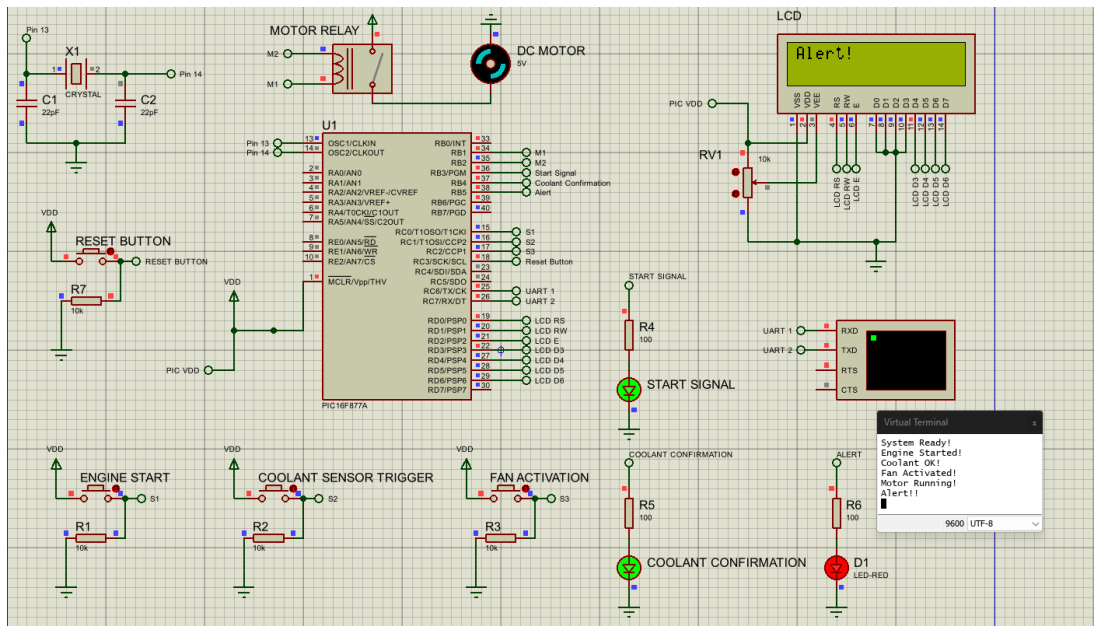


Figure 7 - Alert

The above figure shows the alert LED flashing and the LCD and UART showing Alert, after this shutdown is complete and the corresponding display capture can be seen in figure 8 with all LED's off and both displays showing shutdown complete.

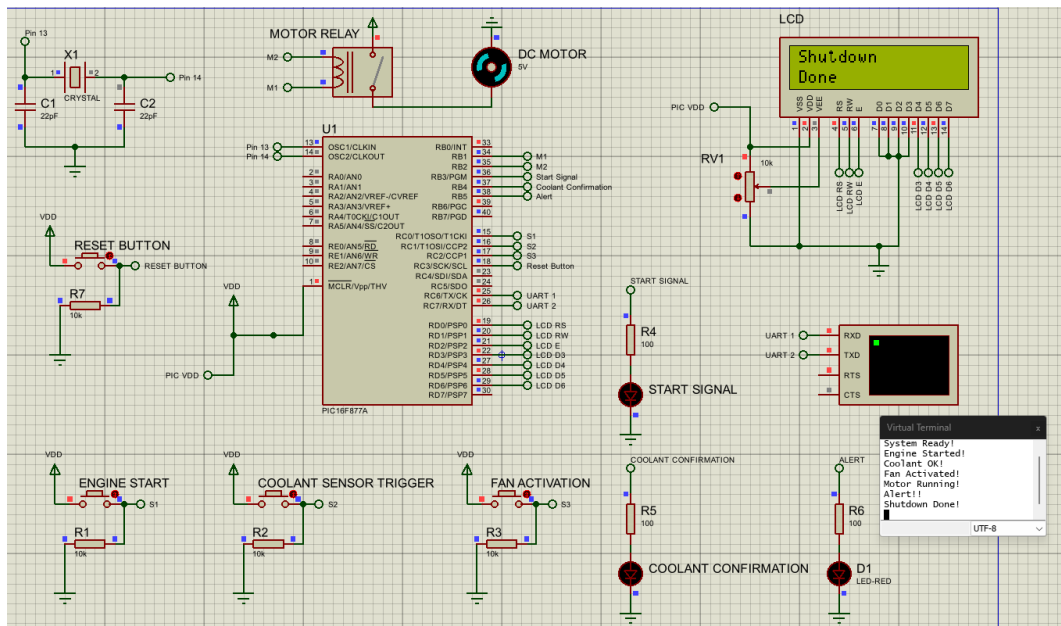


Figure 8 - Shutdown Done

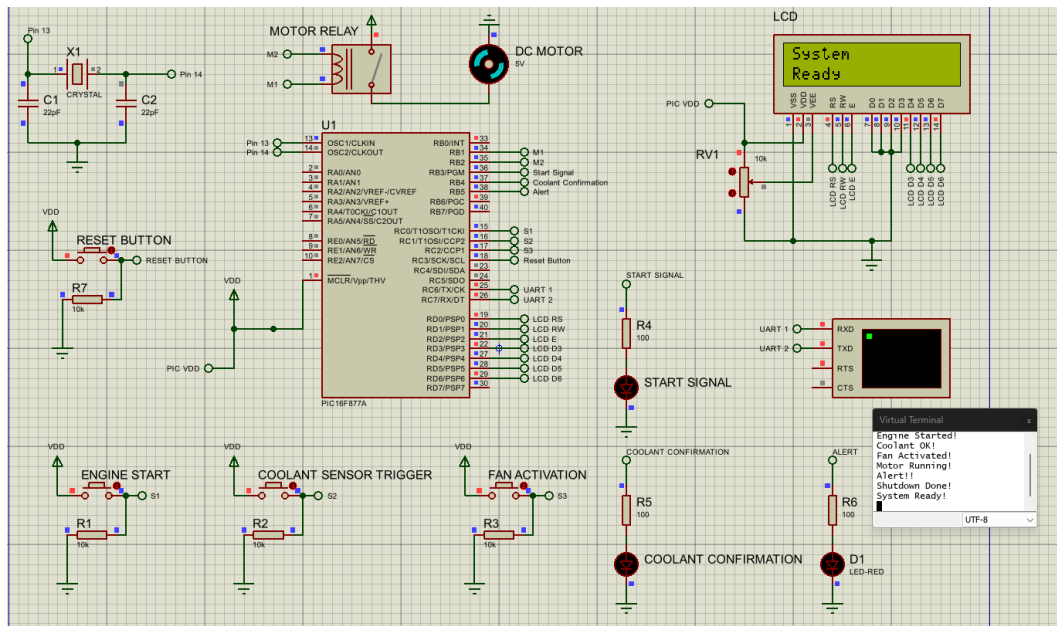


Figure 9 - Reset

A reset button was added extra to the circuit so when pressed it would reset the system to the beginning and the system would state 'System Ready'.

Testing Details

During initial testing the engine system worked as intended with all buttons producing the expected outcome and only in the correct order except for the reset switch. The way the program was initially written had not used a while loop for the reset button only an if statement and therefore when pressed it would reset the system but then after the first use it stopped working and the program ended. The use of a loop the section was used and the button sequence was moved to the beginning and named start process enabling that section of code to be called whenever required and minimise the need to repeat code. The Start Process section is a long piece of code so calling it as a function reduced coding significantly.

```
68 while (True)
69 if (input(Reset_Button)) // Reset Button Pressed
70 {
71     StartProcess(); //Goes to StartProcess
72 }
73 else
74 {
75     output_low(Motor_Ini1);
76     output_low(Motor_Ini2);
77     output_low(LED_StartSignal);
78     output_low(LED_CoolantConfirmation);
79     output_low(LED_Alert);
80     //Above sets all outputs as Low
```

Figure 10 - While Loop

Conclusion

To conclude the engine coolant system and button sequence run as expected. A potential change that would be used next time would be the addition of a 7805 voltage regulator to ensure the 5V supply was maintained to reduce the risk of damage to the PIC from high voltage. There would also be a slight change to the code with the section of code setting all outputs to low being placed into a function and then called whenever required thus reducing the amount of code being re-written unnecessarily.

References

- [1] Microchip, "PIC16F877A," 29 03 2026. [Online]. Available: <https://www.microchip.com/en-us/product/pic16f877a>. [Accessed 29 03 2026].
- [2] Microchip, "Universal Asynchronous Receiver and Transmitter (UART) Peripherals," Microchip, 29 March 2026. [Online]. Available: <https://www.microchip.com/en-us/products/microcontrollers/8-bit-mcus/peripherals/communication-connectivity/uart>. [Accessed 29 March 2026].
- [3] T. A. Arun, T. S. Madhukar and J. Deka, "Speed Control of DC Motor using," *Multidisciplinary Journal of Research in Engineering and Technology*, vol. 1, no. 2, pp. 223-234, 2014.
- [4] M. Y. I. A. S. M. Buji AB, "Model Design, Simulation and Control of a Robotic Arm using PIC 16F877A Microcontroller," *The Arid Zone Journal of Engineering, Technology and Environment*, vol. 15, no. 1, pp. 67-76, 2019.
- [5] S. A. A. D. M. M. A. B. W. P. S. P. T. K. & T. S. K. Pote, "Designing of PIC 16F877A based pH Node for the Development of Water," *International Research Journal of Engineering and Technology (IRJET)*, vol. 8, no. 3, p. 1447, 2021.
- [6] H. A. Al Issa, S. Thuneibat and M. Abdesalam, "Sensors Application Using PIC16F877A Microcontroller," *American Journal of Remote Sensing*, vol. 4, no. 3, pp. 13-18, 2016.
- [7] B. M. Guruswamy and P. Nagaraj, "Automotive car body module control using PIC16F877A," in *International Conference on Circuits, Communication, Control and Computing*, Bangalore, 2014.
- [8] MicroChip, "Embedded Control Solutions for Demanding Automtive Applications," MicroChip, Chandler, 2026.
- [9] A. Tiwari, P. Bhide and V. Gohil, "Adaptive Thermal Control using Dual Motors and PIC16F877A Microcontroller in Multi-Zone Environments," in *8th International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, 2025.
- [10] E. M. Schwartz and J. A. Martiney, "4-bit LCD Interface Notes," University of Florida, Florida, 25.
- [11] R. M. Cerda, *Understanding Quartz Crystals and Oscillators*, Boston: Artech House, 2014.
- [12] J. R. Vig, "Introduction to Quartz Frequency," US Army Laboratory Command, Fort Monomuth, 1992.