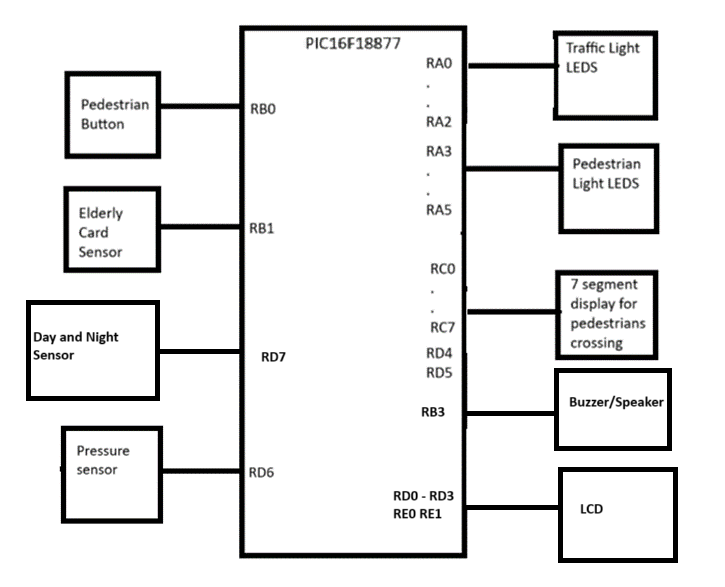
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Technical Requirement** | **Task** | **Technique** | **Component** | **Assign To** | **Due Date** |
| 1. | Traffic light system for cars which include the green, yellow and red light. We will make use of delays to set lights, red with a default timer of 20s, green 5s and yellow 5s. | 1. Connect LEDS TB3 to PORTA 2. Connect RA0 and RA2 to LED1 – LED3 respectively | Digital Output  Timers | LED1 for red  LED2 for yellow  LED3 for green | Aryan | 10/6/24 |
| 2. | Pedestrian light system which includes green man and red man. We will make use of delays to set Green Man with a default timer of 20s and Red Man to the total duration of Green Car Light and Yellow Car Light. | 1. Connect LEDS TB3 to PORTA 2. Connect RA3 and RA5 to LED4 & LED6 respectively | Digital Output  Timers | LED4 for red  LED6 for green | Famous | 10/6/24 |
| 3. | Pedestrian button for pedestrians to cross the road. We will use polling method to constantly check for the pedestrian button. When the button is pressed, the 5s countdown for the Green Car Light begins. | 1. Connect Pushbuttons TB4 to PORTB 2. Connect S7 to RB0 | Digital Input  Timers | Push button S7 | Aryan | 13/6/24 |
| 4. | Elderly button for elderly to cross the road with additional time of 10s added to the default green man timer and red light timer. We will make use of external interrupts to implement the button. | 1. Connect Pushbuttons TB4 to PORTB 2. Connect S8 to RB1 | Digital Input  External Interrupt  Timers | Push button S8 | Famous | 13/6/24 |
| 5. | To display the amount of time left to cross the road, we will make use of the 7 segment display to count down the time left. | 1. Connect 7 segment display TB1 to PORTC 2. Connect SL1 and SL2 to RD4 and RD5 respectively as a 2-4 decoder for the 7 segment display | Digital Output | 7-segment-display D1 and D2 | Aryan | 16/6/24 |
| 6. | A pressure sensor that detects how many people are on each side of the pedestrian crossing in order to add 5s to the default green man timer and red light timer, using ADC and a potentiometer. | 1. Connect an External Potentiometer to PORTD 2. Connect the Input of the Potentiometer to RD6 as an Analog Input. | Analog Input | External Potentiometer | Famous | 16/6/24 |
| 7. | A Speaker/Buzzer that sounds whenever it is time for pedestrians to cross and it will keep ticking till the end of the Green Man Light and Red Car Light duration. | 1. Connect the jumper across the pins 2 and 3 2. Connect the pin INT SPK to RB3 | Digital Output | Speaker | Aryan | 1/8/24 |
| 8. | A Day and Night sensor replaced by an external potentiometer to represent, will detect whether it is day or night. If it is day then the traffic light sequence will continue as normal but if it is night then there would be a LED/Lightbulb turned on while having a decreased walking time since usually there isnt that many people walking around at night so the cars wouldn’t need to for so long. | 1. Connect an External Potentiometer to PORTD 2. Connect the input of the Potentiometer to RD7 as an Analog Input. | Analog Input | External Potentiometer | Aryan | 1/8/24 |
| 9. | A LCD that displays various texts and figures to tell users of the road what to do and not do. | 1. Connect the 4bit data buses to RD0-RD3, Enable and Register Select to RE0 and RE1 respectively. | Digital Output  Timers  Bit Manipulation | LCD | Famous | 1/8/24 |

# Port Pins Listing and Block Diagram

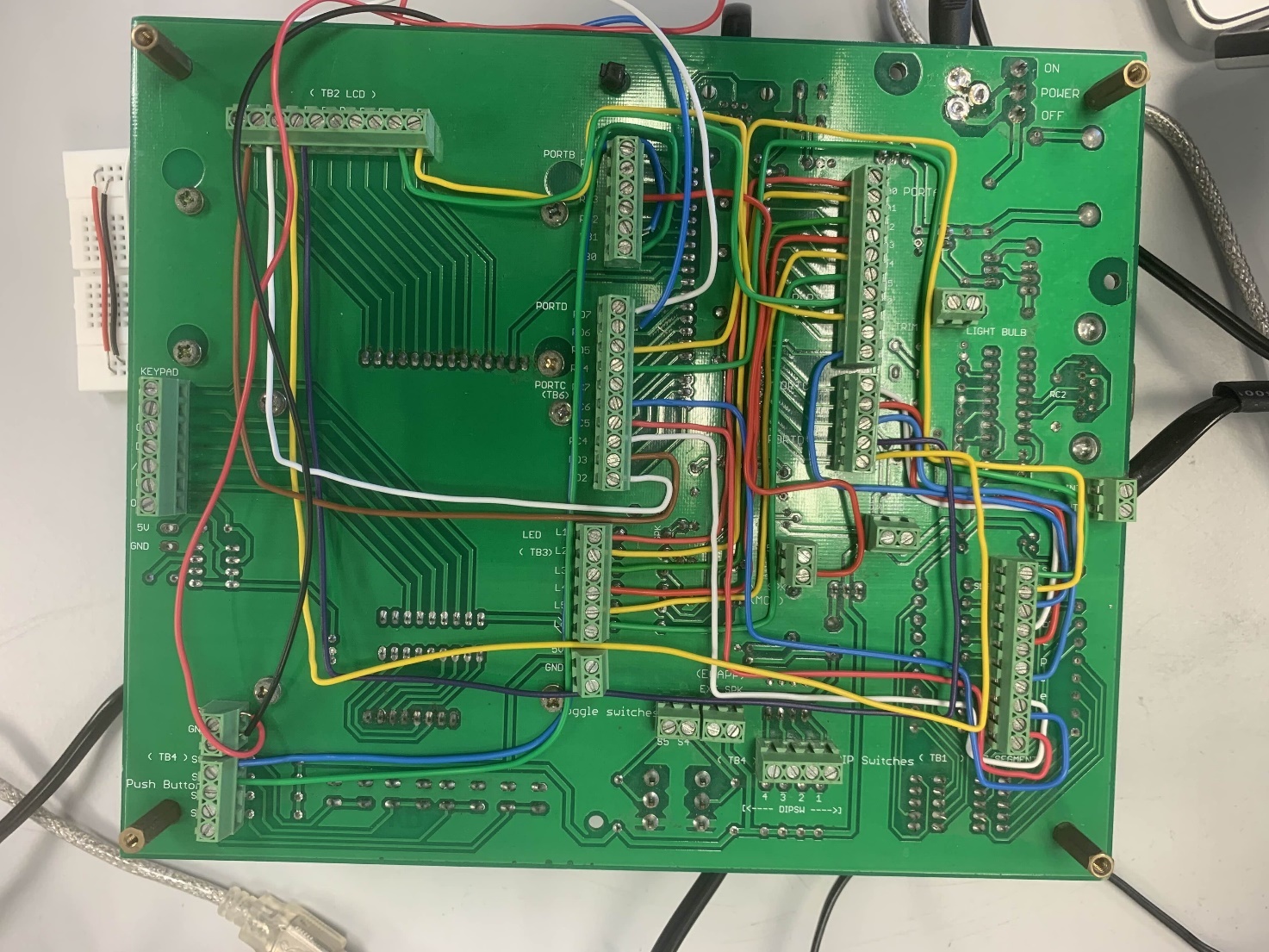
* 1. **Port Pins Table for PIC16F18877**
* *Tabulate the list of port pins used in your project.*
* *State the configuration of ANSELx and TRISx registers.*

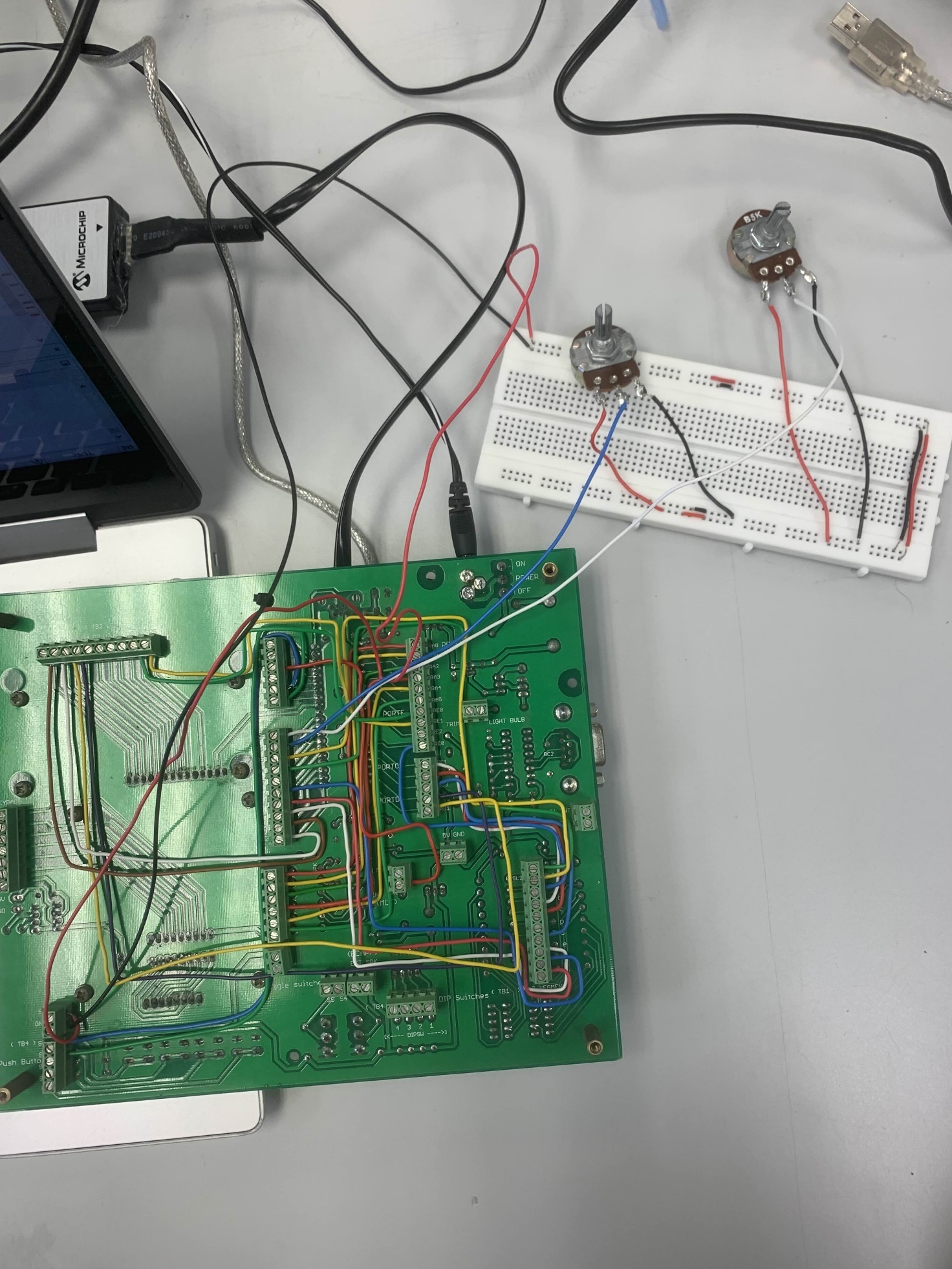
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pin** | **ANSELx**  Is pin used as:  • Digital,  • Analog, or  • Unused (Digital) | **TRISx**  Is pin used as:  • Input,  • Output, or  • Unused (Input) | **Device name: PIN name** eg. LCD: RS or  7-segment: a | **Remarks** |
| RA0 | Digital | Output | LED 1 | Red Traffic Light for Cars |
| RA1 | Digital | Output | LED 2 | Yellow Traffic Light for Cars |
| RA2 | Digital | Output | LED 3 | Green Traffic Light for Cars |
| RA3 | Digital | Output | LED 4 | Red Man Light for Pedestrians |
| RA4 | Digital | Output | LED 5 | LED acting as Light Bulb for Night Time |
| RA5 | Digital | Output | LED 6 | Green Man Light for Pedestrians |
| RA6 | Digital | Input | Already used by oscillator | NA |
| RA7 | Digital | Input | Already used by oscillator | NA |
| ANSELA = 0b00000000; | | | | |
| TRISA= 0b11010000; | | | | |
|  | | | | |
| RB0 | Digital | Input | Push Button Switch S7  GREEN WIRE | Pedestrian traffic light button |
| RB1 | Digital | Input | Push Button Switch S8  BLUE WIRE | Elderly card sensor |
| RB2 | Unused (Digital) | Unused (Input) | Unused | NA |
| RB3 | Digital | Output | Speaker | Red Light Speaker |
| RB4 | Unused (Digital) | Unused (Input) | Unused | NA |
| RB5 | Unused (Digital) | Unused (Input) | Unused | NA |
| RB6 | Digital | Input | Already used by PICkit-4 | Set as input |
| RB7 | Digital | Input | Already used by PICkit-4 | Set as input |
| ANSELB = 0b00000000; | | | | |
| TRISB= 0b11111111; | | | | |
|  | | | | |
| RC0 | Digital | Output | 7-Segment Display a  BLUE WIRE | a |
| RC1 | Digital | Output | 7-Segment Display b  WHITE WIRE | b |
| RC2 | Digital | Output | 7-Segment Display c  RED WIRE | c |
| RC3 | Digital | Output | 7-Segment Display d  BLUE WIRE | d |
| RC4 | Digital | Output | 7-Segment Display e  WHITE WIRE | e |
| RC5 | Digital | Output | 7-Segment Display f  RED WIRE | f |
| RC6 | Digital | Output | 7-Segment Display g  BLUE WIRE | g |
| RC7 | Unused (Digital) | Unused (Input) | 7-Segment Display . | Unused Decimal Place |
| ANSELC = 0b00000000; | | | | |
| TRISC= 0b10000000; | | | | |
|  | | | | |
| RD0 | Digital | Output | LCD : D4  PURPLE WIRE | 4-bit Data Bus |
| RD1 | Digital | Output | LCD : D5  YELLOW WIRE | 4-bit Data Bus |
| RD2 | Digital | Output | LCD : D6  WHITE WIRE | 4-bit Data Bus |
| RD3 | Digital | Output | LCD : D7  BROWN WIRE | 4-bit Data Bus |
| RD4 | Digital | Output | 7-Segment Display : SL1  GREEN WIRE | Lowest Bit 2-4 Decoder |
| RD5 | Digital | Output | 7-Segment Display : SL2  YELLOW WIRE | Highest Bit 2-4 Decoder |
| RD6 | Analog | Input | External Potentiometer | Pressure Sensor |
| RD7 | Analog | Input | External Potentiometer | Day & Night Sensor |
| ANSELD = 0b01000000; | | | | |
| TRISD= 0b11000000; | | | | |
|  | | | | |
| RE0 | Digital | Output | LCD : E  GREEN WIRE | Enable |
| RE1 | Digital | Output | LCD : RS  YELLOW WIRE | Register Select |
| RE2 | Unused (Digital) | Unused(Input) | Unused | NA |
| RE3 | Digital | Input | Already used as reset pin (MCLR) | NA |
| RE4 | Unused (Digital) | Unused (Input) | Pin is not present | Set as unused |
| RE5 |
| RE6 |
| RE7 |
| ANSELE = 0b00000000; | | | | |
| TRISE= 0b11111100; | | | | |

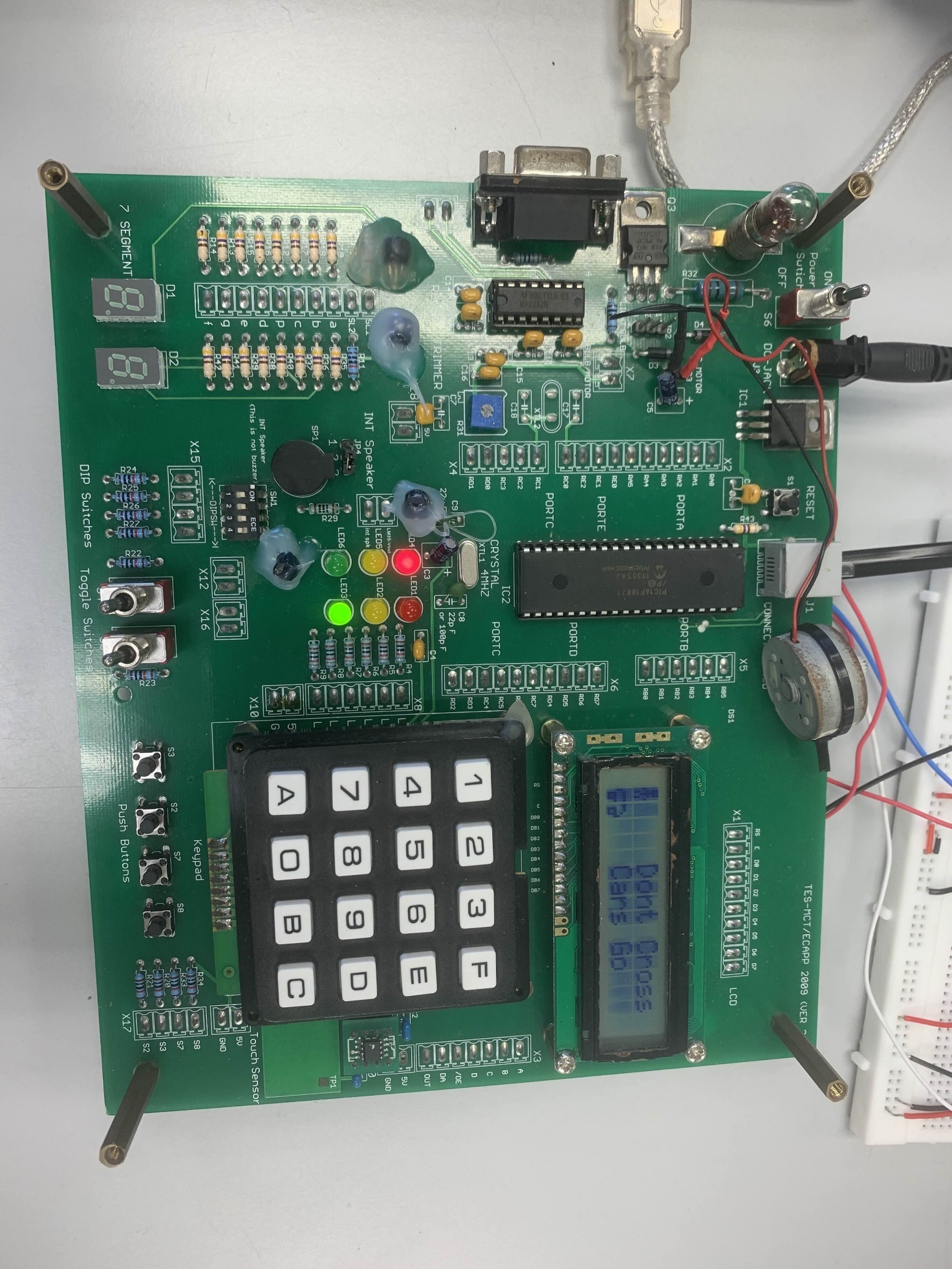
* 1. **Block Diagram**

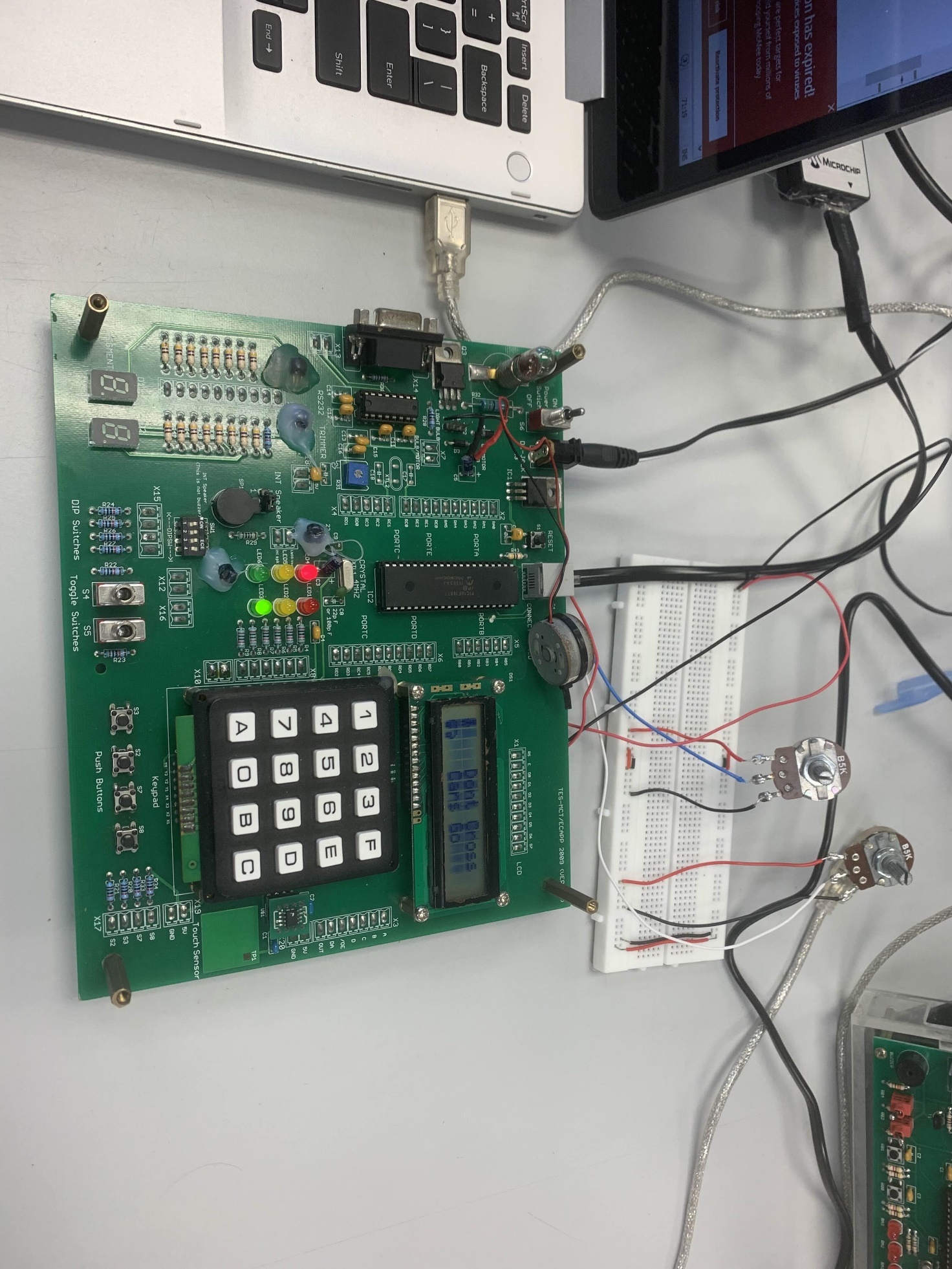
****

## Pictures of Board Set Up









# Program Design Map

**Program Design Map**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Source File** | **What is the name of the function you have designed and resides in this source file?** | **What is the purpose of this function?** |
| 1 | main.c | void initSysPins(void) | To configure and initialise the port pins as analog/digital, inputs/outputs |
| 2 | usrTask.c | void usrTask\_PedDeduct(void)  void detPed(void) | **void usrTask\_PedDeduct(void):**  To handle the reading of push button S7, to determine whether a pedestrian wants to cross the road. If pressed, 5 seconds will be deducted from the Green Light and Red Man default timer provided there is more than 10s remaining in the default timer.  **void detPed(void):**  To apply the polling method for the push button S7. |
| 3 | adc.c | void initADC(void)  unsigned int adc\_GetConversion(void)  unsigned int adcDayNight\_GetConversion(void) | **void initADC(void):**  To configure and initialise the ADC using ADREF, ADCLK, ADACQ, ADCON0 to implement the use of ADC operation.  **unsigned int adc\_GetConversion(void):**  To start ADC conversion and get the result of the Total Pressure.  **unsigned int adcDayNight\_GetConversion(void):**  To start ADC conversion and get the result to see whether is Day or Night |
| 4 | sev\_seg.c | void seg\_DispAll(unsigned int result) | To break down the countdown timer of the pedestrian crossing into its tenth and oneth place and to display the countdown of the timer to pedestrians. |
| 5 | extint.c | void initSysExtint(void)  void extint\_setElderlyDetect(void) | **void initSysExtint(void):**  To configure and initialise the peripherals like enabling/disabling the Global Interrupt, clear External Interrupt flag, enable external interrupt and configure for Rising edge/Falling edge to make use of the External Interrupt.  **void extint\_setElderlyDetect(void):**  Determine whether an elderly wants to cross the road. If elderly taps their card, 5 seconds will be added to the Red Light and Green Man default timer. |
| 6 | isr.c | void \_\_interrupt() isr(void) | Whenever the external interrupt is detected, the ISR will call the elderlyAdd() to add on 10 seconds to the Red Light and Green Man default timer. |
| 7 | Buzzer.c | void playInitialTune(void)  void updateToneAndDisplay(void) | **void playInitialTune(void):**  To play the initial tune of the traffic light when it first turns into Red Car Light and Green Man Light.  **void updateToneAndDisplay(void):**  To play the constant ticking for the remaining duration. |
| 8 | lcd.c | void initLCD(void)  void lcdCreateCustomChars(void)  void displayStandingStickman(void)  void displayRightWalkStickman(void)  void displayLeftWalkStickman(void)  void displayGreenLightText(void)  void displayYellowLightText(void)  void displayRedLightText(void)  void clearLCD(void) | **void initLCD(void):**  To initialize the LCD.  **void lcdCreateCustomChars(void):**  To create custom characters in the CGRAM of the LCD.  **void displayStandingStickman(void):**  To display the standing stickman character on the LCD.  **void displayRightWalkStickman(void):**  To display the right walking stickman character on the LCD.  **void displayLeftWalkStickman(void):**  To display the left walking stickman character on the LCD.  **void displayGreenLightText(void):**  To display the "Dont Cross" and "Cars Go" messages during the green light phase.  **void displayYellowLightText(void):**  To display the "Dont Cross" and "Cars Slow" messages during the yellow light phase.  **void displayRedLightText(void):**  To display the "Cross" and "Cars Stop" messages during the red light phase.  **void clearLCD(void):**  To clear the LCD display. |
| 9 | dspTask,c | void greenLight(void)  void yellowLight(void)  void redLightGreenMan(void)  void resetLights(void)  void dspTask\_OnSevSeg(void) | **void greenLight(void):**  To handle the timing and state changes for the green light.  **void yellowLight(void):**  To handle the timing and state changes for the yellow light.  **void redLightGreenMan(void):**  To handle the timing and state changes for the red light and green pedestrian signal.  **void resetLights(void):**  To reset all light states and timers.  **void dspTask\_OnSevSeg(void):**  To display the remaining red light time on a seven-segment display. |
|  |  |  |  |

# Flow Charts

