Department of Electronic and Telecommunication Engineering

University of Moratuwa

Extension Power Socket with a Timer

Group EN-2

Amarasinghe A.M.V.M.	200027R
Amarasinghe Y.E.	200029B
Anuradha A.K.	200041E
Arukgoda A.M.O.	200051J

This report is submitted as a partial fulfilment of module EN1190

07 October 2022

Content

Abstract

As our project we have designed a s an enhanced multiplug which compatible for turn on a device by a specified time past and turning off a device after a specified time. Users can activate the periodical mode of the device to supply the current to the equipment periodically. Also, we developed a model for the user to manually operate the equipment connected to the multi-plug. An LCD display is included in the product and that shows all settings and remaining time details to the user. We have designed this product as a solution to the energy waste in Sri Lanka where energy has a high demand. Through this product, user can limit the time that an equipment is connected to power and if needed, they can switch this power socket on off periodically as required.

1 Introduction

We live in an era in which energy has a high demand. Therefore, cost of the energy is increasing day by day. In Sri Lanka, this problem is severe because of the current economic crisis and shortage of fuels and energy sources. Most of the things we do in day-to-day life need electricity therefore electricity is essential for living in this era. People are very busy with their work therefore sometimes they forgot to switch off the power supply or device. It causes energy waste. This causes people to pay more than they are supposed to pay for electricity bills. Equally, it is not good to waste energy. Survey results provide evidence for this There is Electrical and Electronic equipment which can be damaged due to supplying power more than the required time. This can be caused sometimes device to be fully damaged or partially damaged.

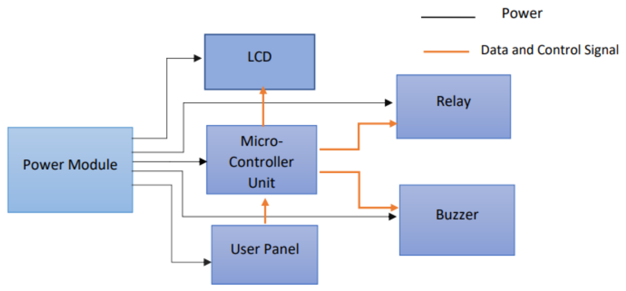
Our solution for this concern is an extension power socket with a timer. This product is an enhanced multiplug which compatible for turn on a device by a specified time past and turning off a device after a specified time. Users can activate the periodical mode of the device to supply the current to the equipment periodically. Also, we developed a model for the user to manually operate the equipment connected to the multi-plug. An LCD display is included in the product and that shows all settings and remaining time details to the user. Therefore, the user can feel the functioning of the device while giving the inputs and the process of the system. We make our best effort to make a simple UI and great UX for the product so the user can control the device easily by using only four buttons.

This product is powered by a PIC16F628A microcontroller, some common electronic components, and an AC-DC step-down module. All the components we need are available in the local market and are cheap and common. The microcontroller, display, and other electronic components are powered by 5V voltage which is given by 230V AC to 5V DC converter. We use the PIC16F628A microcontroller instead of Atmega328p because PIC Microcontroller is cheaper than Atmega328p and it is capable enough for our purpose. We flash the PIC microcontroller with the .hex file code using an Arduino board as a programmer and use the programmed IC to design the product. The PCB of the device was designed by Altium Designer and the enclosure was designed by Solid works.

This product can turn on and off a device in a time intervals of 99 hours 99 minutes 99 seconds with an accuracy of 1 second. Also, the user can end the timer and turn the device on or off immediately using just one switch. This product beeps when the appliance is turned on and off after the timer counts down to the time set by the user. The power consumption of the device is very low, and the user can disable the timer and display for unnecessary times or save power. A power outage or turning off the timer does not affect the device's available memory, so the user can restart the timer from pause time or reset the timer and create a new countdown time after turning the device back on. There are approximately 2m of wire between the product's wall plug and the appliance's power socket. The product has only one power socket with a fuse that can handle up to 7A current. We use square pins for wall power plugs for safety, and a square and round socket for multi-plug for greater appliance compatibility for timer multi-plug design. The overall design is logically complex but physically simple for the user.

2 Method

This extension power socket was divided into two main parts, power socket and LCD display with buttons. The project was started by implementing every part of extension power socket using proteus simulation software. We flash the PIC microcontroller with the .hex file code using an Arduino board as a programmer and use the programmed IC to design the product. The PCB of the device was designed by Altium Designer and the enclosure was designed by Solid works.



2.1 Components

2.1.1 Microcontroller

The microcontroller we used was PIC16F628A microcontroller. PIC16F628A is a CMOS FLASH-based mid-range 8-bit microcontroller that comes with an 18-Pin package, out of which, 16 pins can be used as I/O pins. This microcontroller has 4 MHz of internal oscillator with 128 bytes of EEPROM data memory, packed with a single Capture/Compare/PWM, and a USART module with 2 comparators. Low voltage programming is supported by this microcontroller unit.

The key points we considered to select this microcontroller are its low power consumption and low cost. When we were in initial steps, an Arduino board was used to program the microcontroller. The outer appearance and the pin diagram of PIC16F628A are shown in figure 1 and figure 2 respectively.

Figure 1- PIC16F628A package

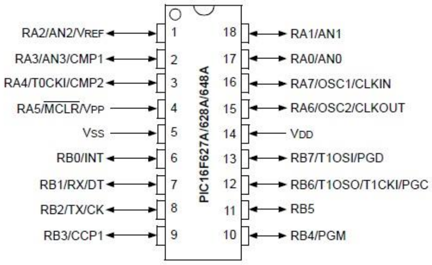


Figure 2- pin diagram

2.1.2 LCD Display

This display was selected due to its compliance to our requirements. The 16x2 LCD display was used in the 4-bit mode. LCD Display acts as the user interface and shows the process status, current option, and time remaining to end the current operation. An image is shown in figure 3.

Figure 3 – 16x2 LCD Display

2.1.3 AC-DC step-down module

We used AC-DC step-down module to convert supply AC voltage to DC and reduce the value. Step down module gives regulated 5V DC power to the circuit from the main line. (230V AC to 5V DC) It's maximum current is 700 mA and maximum power output is 3.5W isolated switching power supply. An image is shown in figure 4.

Figure 4 – AC-DC step down module

2.1.4 Relay Module

A 5V relay module is used to Switch off or switch on the power supply to the socket when needed. A 5v relay is an automatic switch that is commonly used in an automatic control circuit and to control a high-current using a low-current signal. The input voltage of the relay signal ranges from 0 to 5V. An image is shown in figure 5.

Figure 5- 5V Relay module

2.1.5 Buzzer

A buzzer is used to give user the signal that the time that has been set is over. Buzzer is a sounding device that can convert audio signals into sound signals which is powered by DC voltage. An image is shown in figure 6.

Figure 6 – Buzzer

2.1.6 Other Components

Table 1 listed the other components we have used except the components listed below.

Component	Value	Quantity
3 core wires	16/0.20 mm	2m
Plug top		1
Push buttons		6
Switch		1
18 pin bases		1
Resistor	10 KΩ	7
Preset resistor	10 KΩ	1
2N2222 transistor		2
Crystal	4 MHz	1
Capacitor	22pF	2

Table 1- Other components

2.2 Display and Menu

As the first step, the menu and the display were implemented using Proteus simulation. Liquid Crystal library was used for this purpose. Our design has a socket which we can set the time to start and to stop. There are buttons to set the time and to control the operation. We decided to add 5 buttons to operate this extension power socket with a timer.5 buttons are power button, Menu/+ button, select button, enter button and Start/Stop. The purpose of each button is listed below.

2.2.1. Power Button

When we press this button, the timer circuit gets powered, and the LCD will display a message saying, “Extension Power Cord with Timer” and then the LCD will display the 1st option of the menu (set on time with the previous set time) automatically.

2.2.2. Menu /+ Button

In the display, we can navigate through the three options of the menu (set ON time, set OFF time, Timer Cycle) After selecting a digit place in the time using the select button pressing the menu button will add an increment to the current value.

2.2.3. Select Button

Used to select a digit place when setting time, after setting the desired time using the menu button, pressing this button will shift the digit place to the right.

2.2.4. Enter Button

After setting the required time in any option (set ON time, set OFF time) or after setting value (1 or 0) to the timer cycle, pressing Enter button will save the changes we made.

2.2.5. Start/ Stop Button

After setting the desired set OFF time, set ON time, and timer cycle option, we use this button to start the operation or to terminate a currently ongoing operation instantly.

2.3 Power circuit

Operating voltage is 230V 50Hz AC voltage and an AC-DC Step down converter is used to convert 230V AC Voltage to 5V DC Voltage. Maximum Current (With appliance Current) that this product could handle is 13A and maximum device Current drain is 500mA. Power Consumption of this extension power socket is 2W.

2.4 PCB Design

The schematic (Appendix 1) of the circuit was drawn first and the PCB (Appendix 2 and 3) was designed accordingly using Altium. The PCB was made small as much as possible. Lines were routed in the top and the bottom layer. 5V power lines were routed at the thickness of 1mm. 230V power lines were routed at the thickness of 2mm.

2.5 3D Enclosure

The PCB designed using Altium was exported to Solid works as a 3D step file. Then external parts were imported to the 3D assembly. Next the bottom part of the enclosure and the top part of the enclosure were designed. The top part and the bottom part were connected using fasteners (Appendix 4 and 5).

3 Results

The LCD display and the menu worked properly without any failure. The buttons did not have any bouncing effect. The expected scrolling effect of the menu worked properly. The above results were observed using the breadboard prototype. Buzzer worked properly as expected and the final expected result were there without any failures.

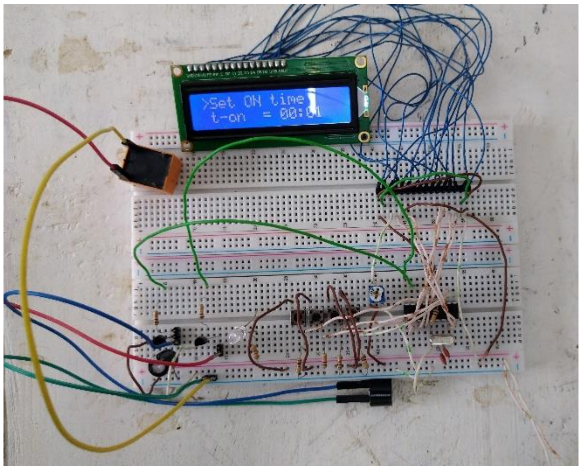


Figure 7- Breadboard implementation

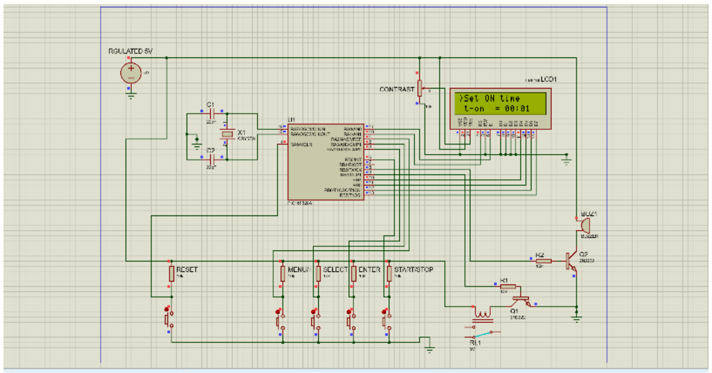


Figure 8- Proteus Simulation

4 Discussion

Each part of the product was separately implemented and then it was combined. Initially PIC16F628A was selected for combining the code. The prototype was successful in the proteus simulation. Next, we implemented the circuit in breadboard prototype. Every component of the circuit worked properly expect buzzer. After connecting a capacitor parallel to the buzzer, buzzer worked properly, and breadboard prototype was complete. At first our plan was to use 6 buttons to control the power socket with display. But after we understood that 5 buttons would be sufficient for the design and changed the buttons by removing reset button. After completing designing the solid works enclosure and PCB design, we printed the enclosure by 3D printing and printed the PCB. Circuit components were successfully soldered to PCB and circuit was placed inside the enclosure. After that we connected each component in the design using wires. LCD display and menu worked properly, and buzzer also worked properly in the prototype. Final expected outcome was there without any failure.

4 Project Budget with BOQ

Component	**Price (Rs.)**
PIC16F628A	850
HD44780 LCD	760
3 core wire (16/0.20mm) 2m	560
PCB designing	400
Enclosure 3D printing	1000
Plug Top	650
SPMC Power Module	350
5V-5pin relay	120
Push Button(x5)	50
Switch	10
Buzzer	60
18 Pin Base	30
1/4W 10 KΩ Resistor(x7)	7
Preset Resistor 10KΩ	10
2N2222 Transistor(x2)	16
4MHz Crystal	35
Ceramic Capacitor 22pF(x2)	4
Total Cost	**4912**

Table 2 – Project Budget

5 Acknowledgement

We would like to thank each person who has helped us even in a very small manner in order to achieve good results in this project. We would like to specially thank Dr. Ajith Pasqual. He motivated us to learn the extra subjects required.

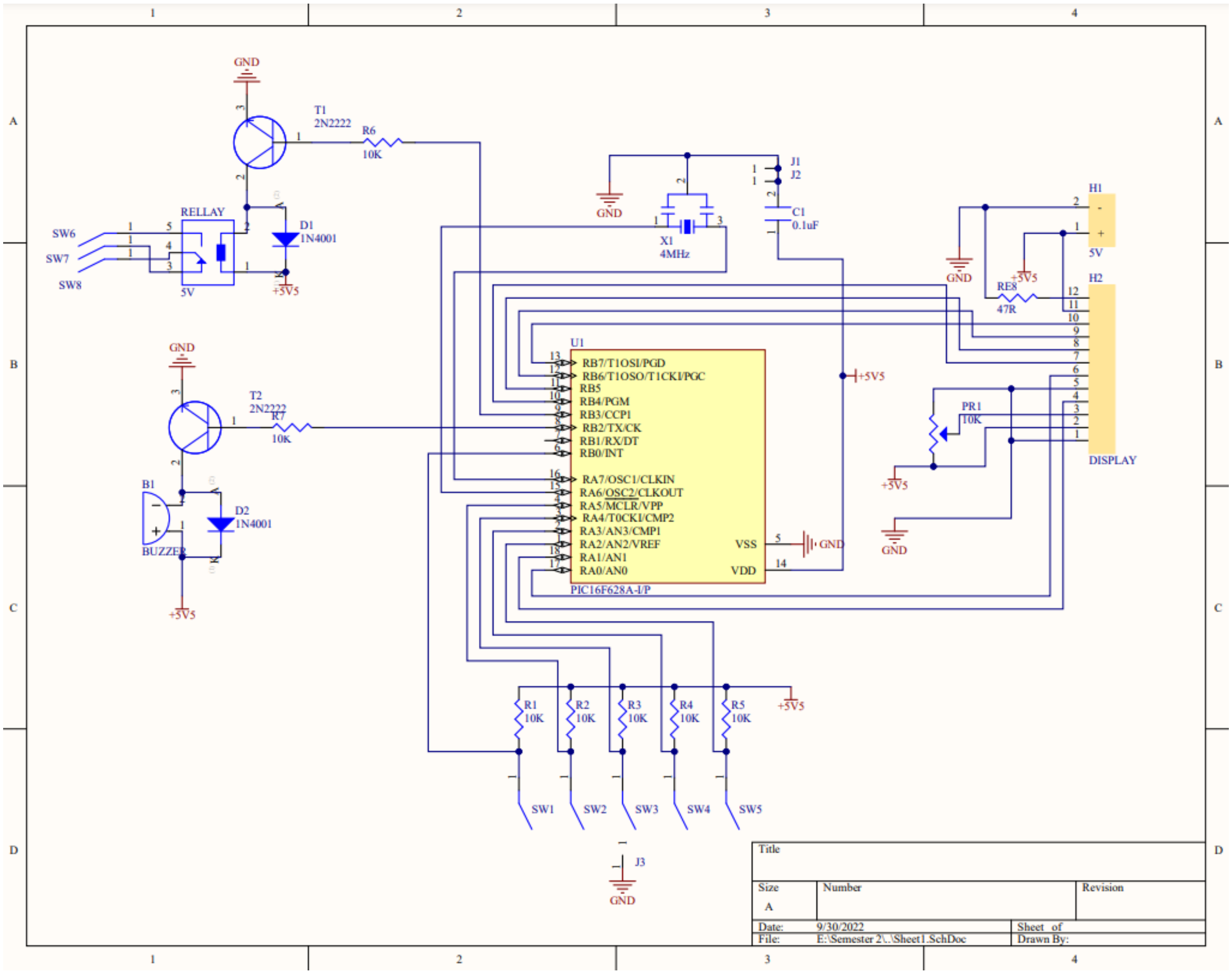
References

https://www.electronics-lab.com/project/pic16f628a-programmable-digital-timer/

https://ww1.microchip.com/downloads/en/DeviceDoc/40044G.pdf

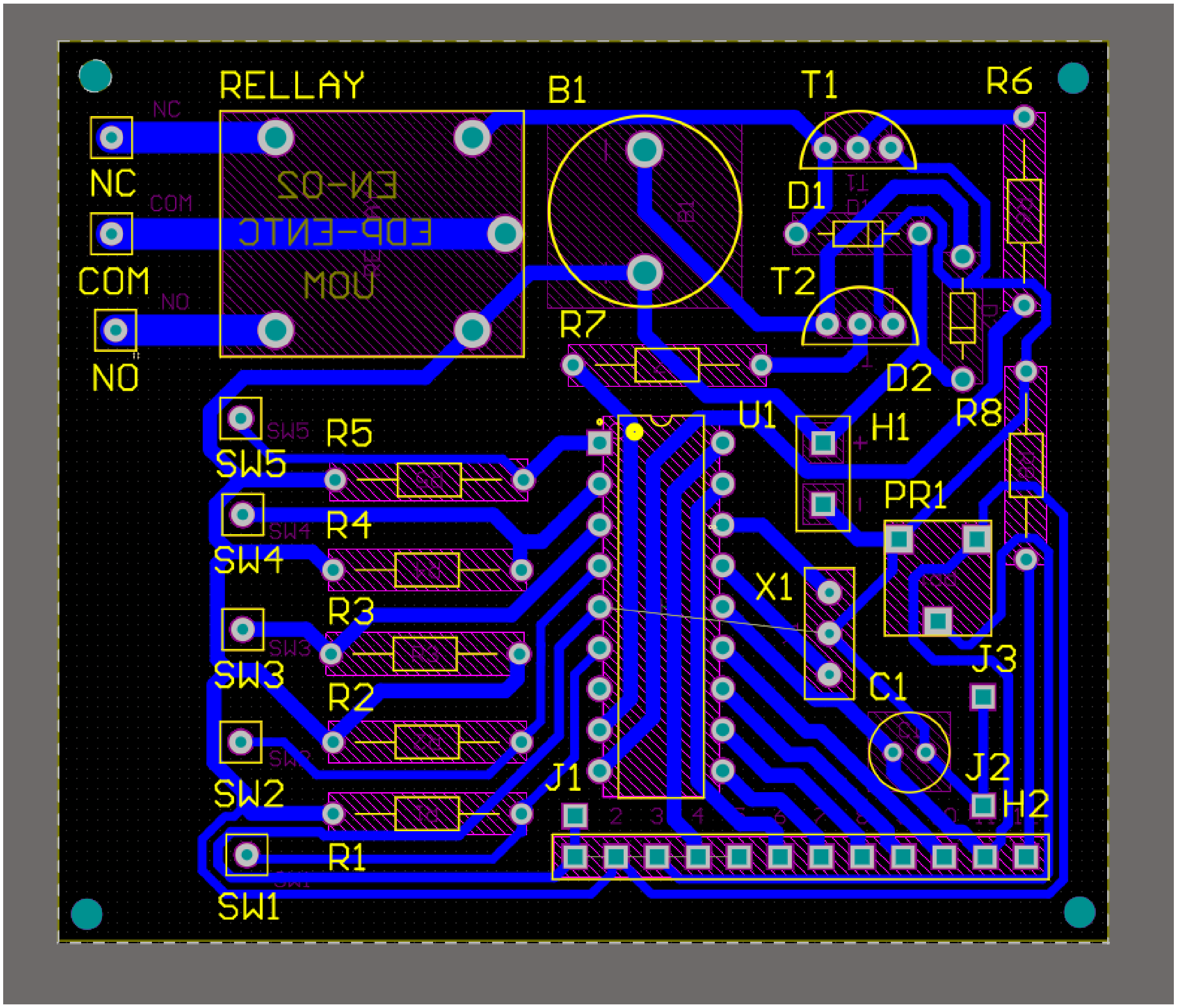
6 Appendices

6.1 Appendix 1 - PCB Schematic

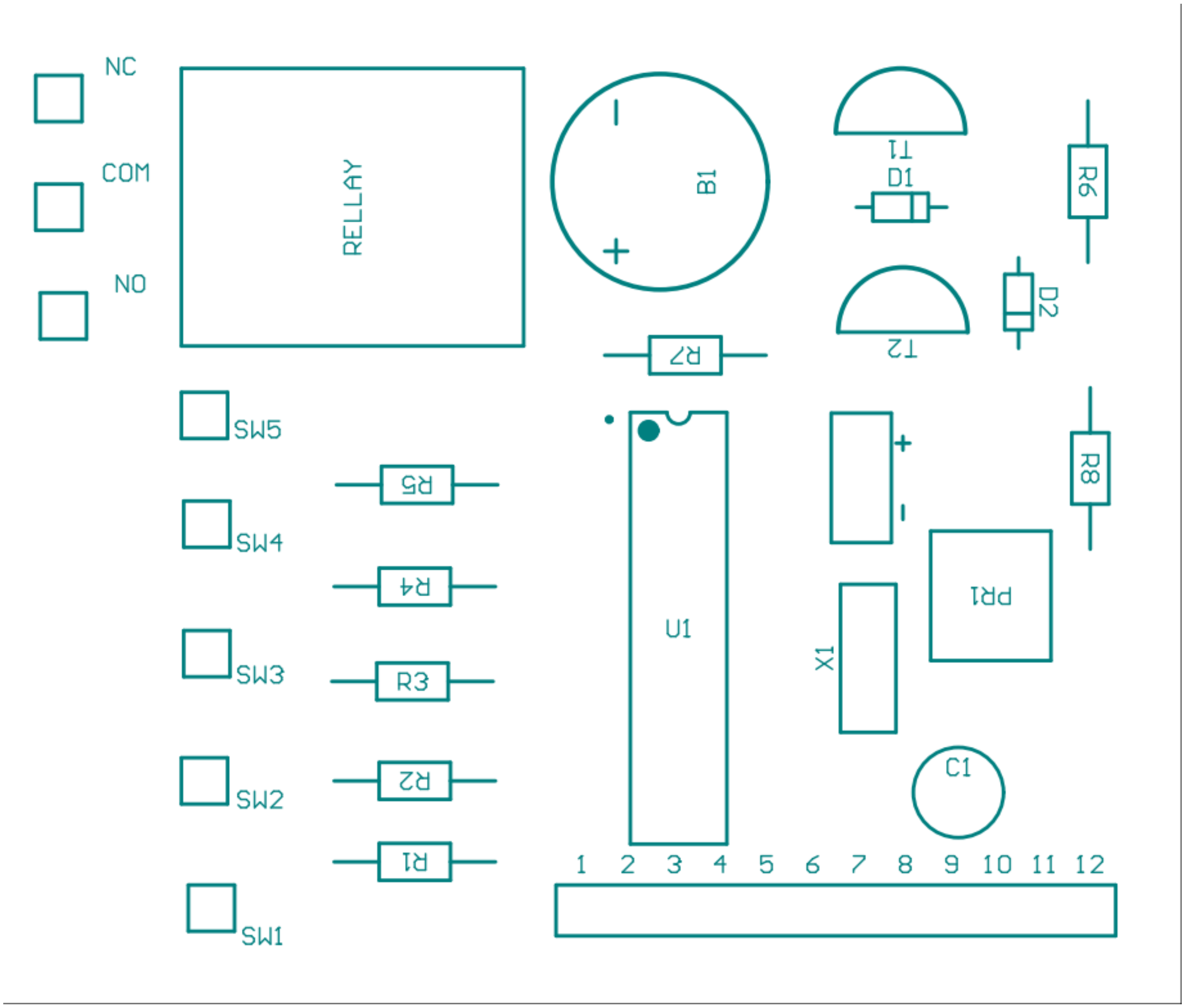


Appendix 1- PCB Schematic

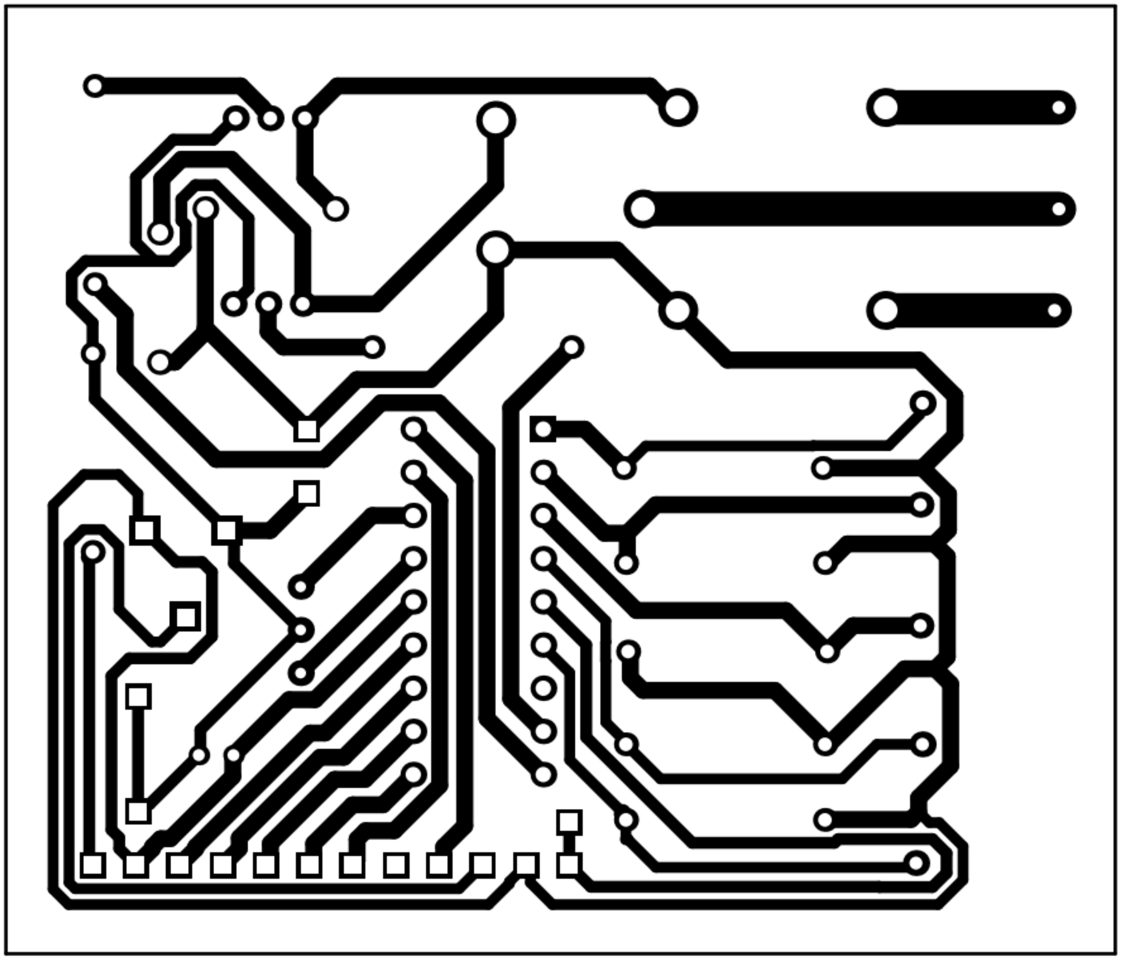
6.2 Appendix 2 - PCB Routing



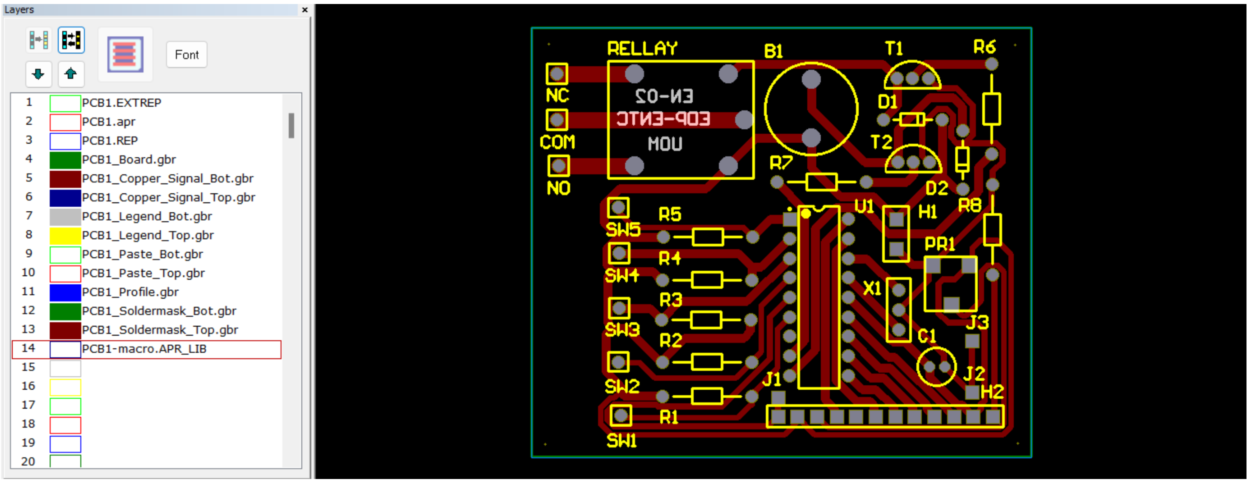
PCB Routing



PCB Assembly Drawing



PCB Copper Path Layer



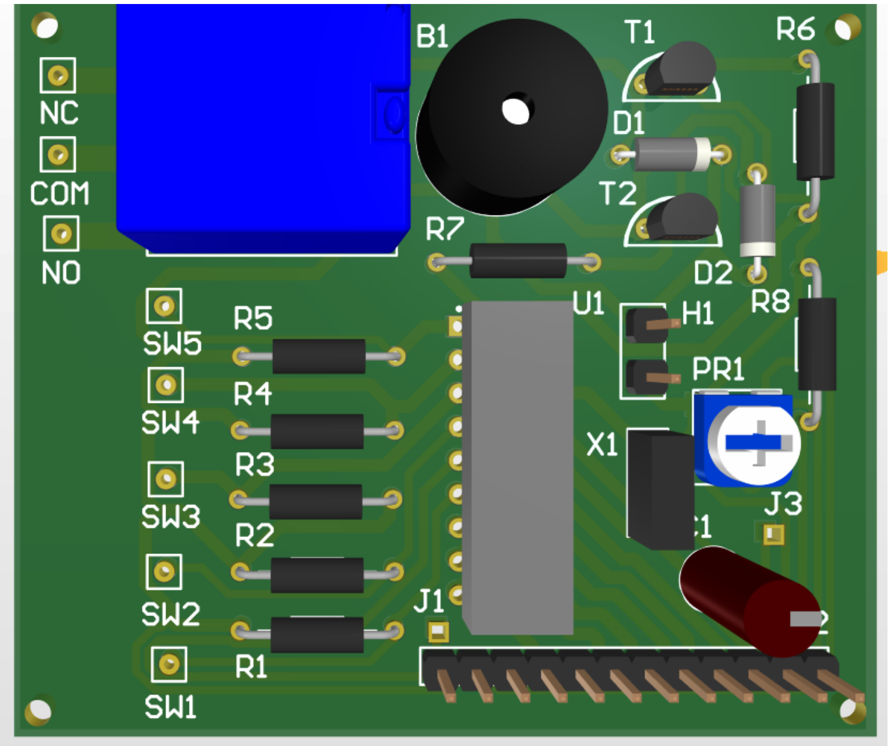
Layers

Font

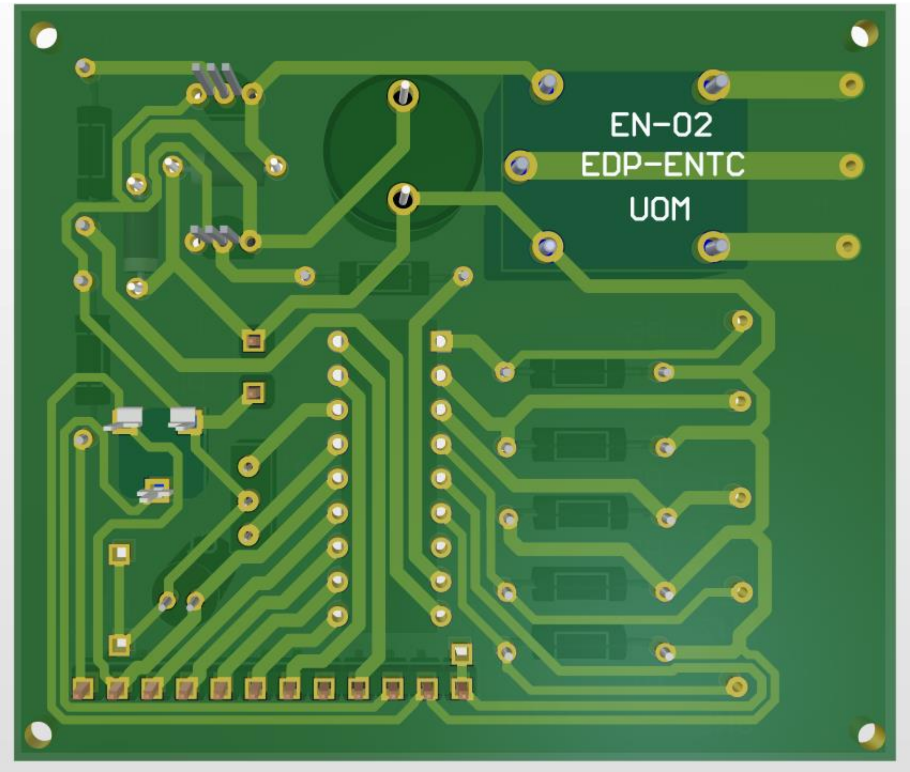
1 PCB1.EXTREP
2 PCB1.apr
3 PCB1.REP
4 PCB1_Board.gbr
5 PCB1_Copper_Signal_Bot.gbr
6 PCB1_Copper_Signal_Top.gbr
7 PCB1_Legend_Bot.gbr
8 PCB1_Legend_Top.gbr
9 PCB1_Paste_Bot.gbr
10 PCB1_Paste_Top.gbr
11 PCB1_Profile.gbr
12 PCB1_Soldermask_Bot.gbr
13 PCB1_Soldermask_Top.gbr
14 PCB1-macro.APR_LIB
15
16
17
18
19
20

PCB Gerber Output

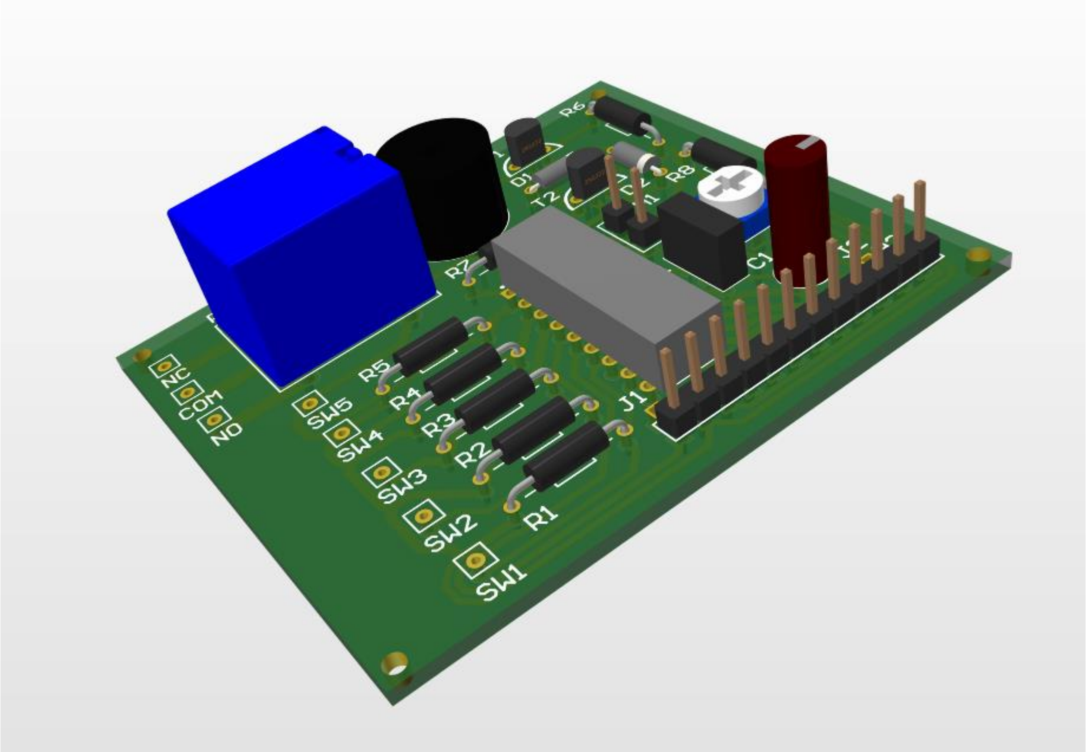
6.3 Appendix 3 - PCB 3D View



PCB 3D View Front

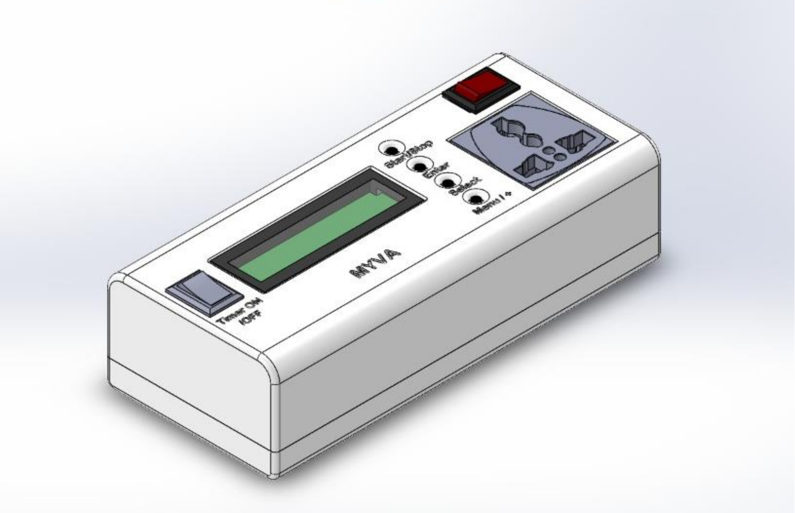


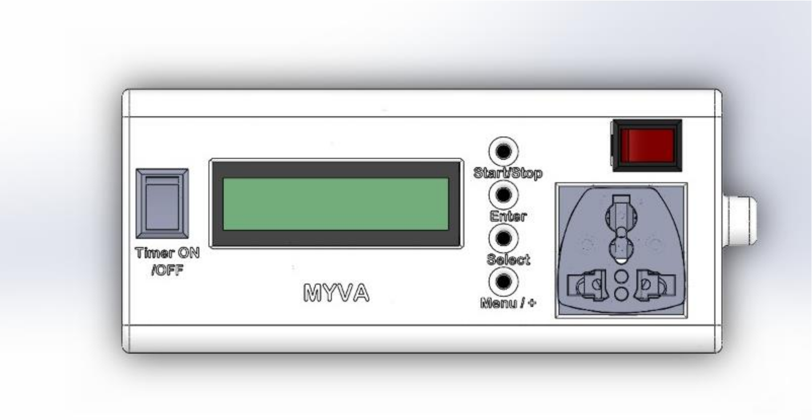
PCB 3D View Back

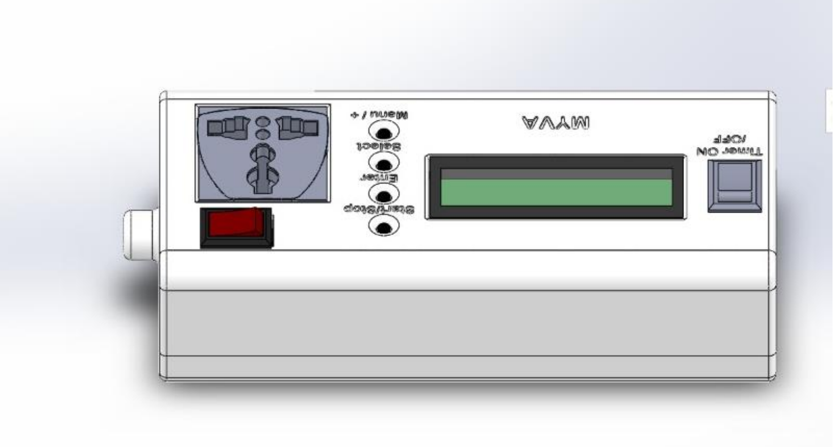


PCB 3D View Angle

6.4 Appendix 4 - Enclosure Design





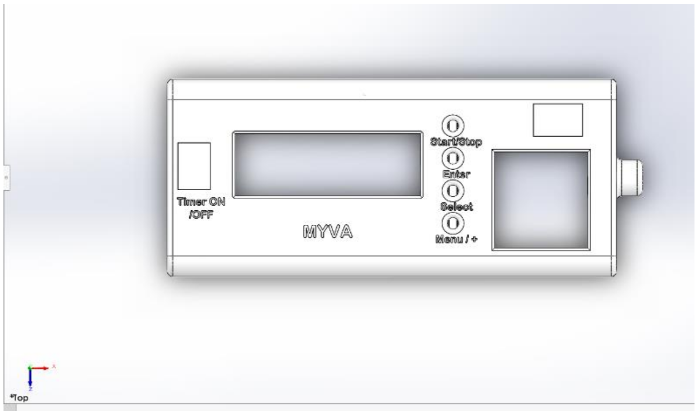


6.5 Appendix 5 - Separated Parts

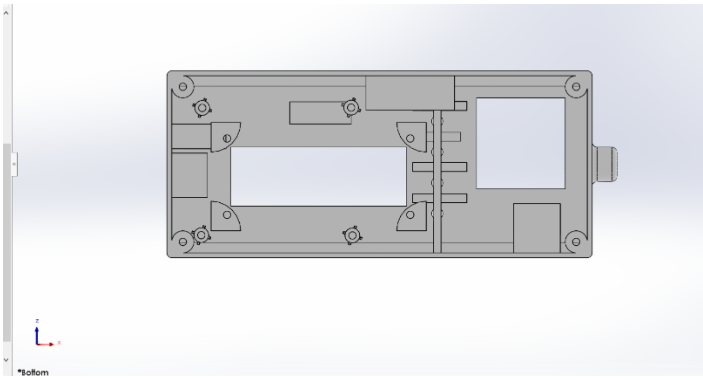
Consist of 2 part as upper part and lower part.

Upper part

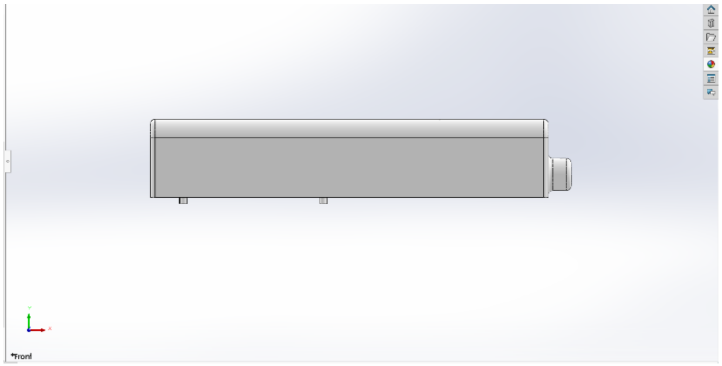
Top view



Bottom view

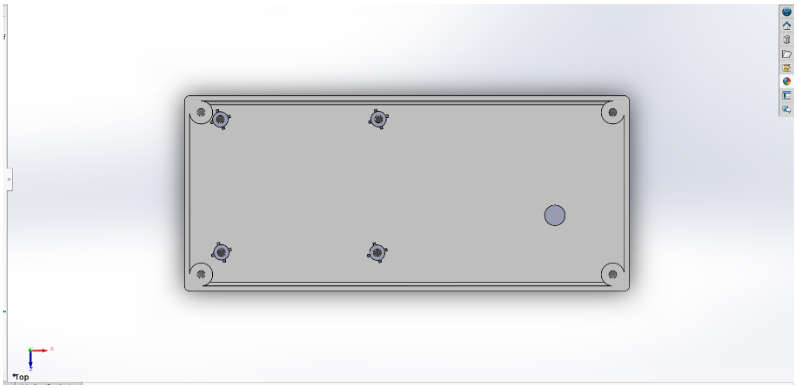


Front view

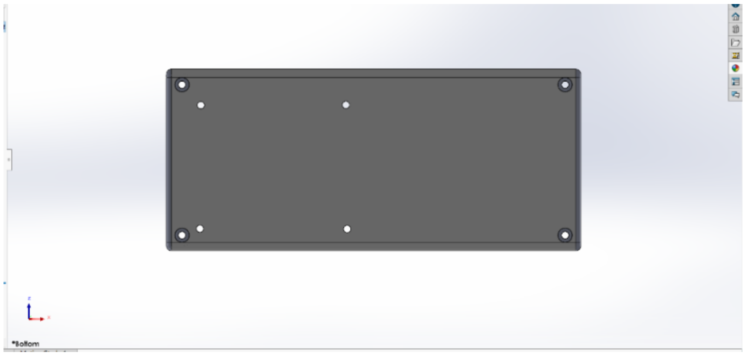


Lower part

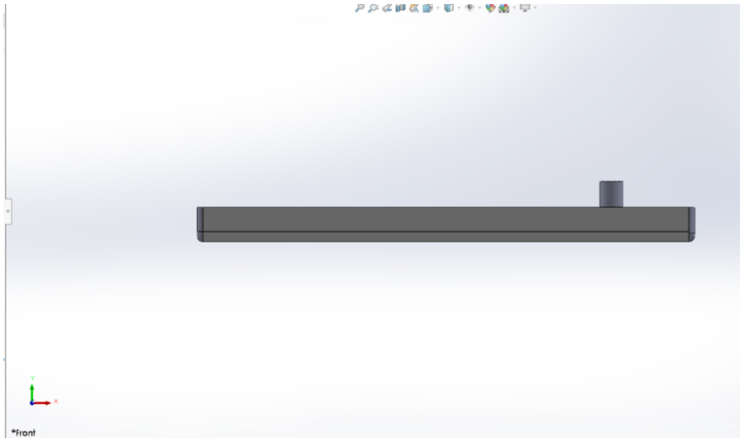
Top view



Bottom view

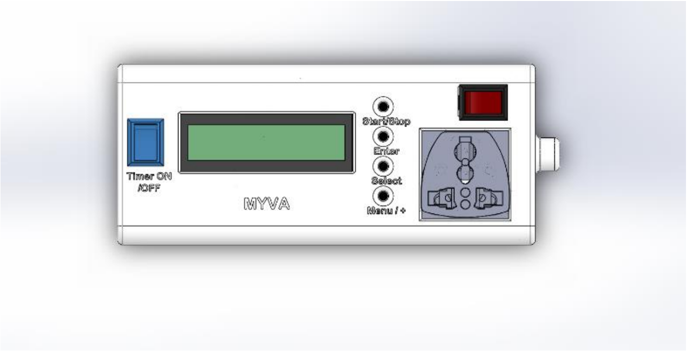


Front view

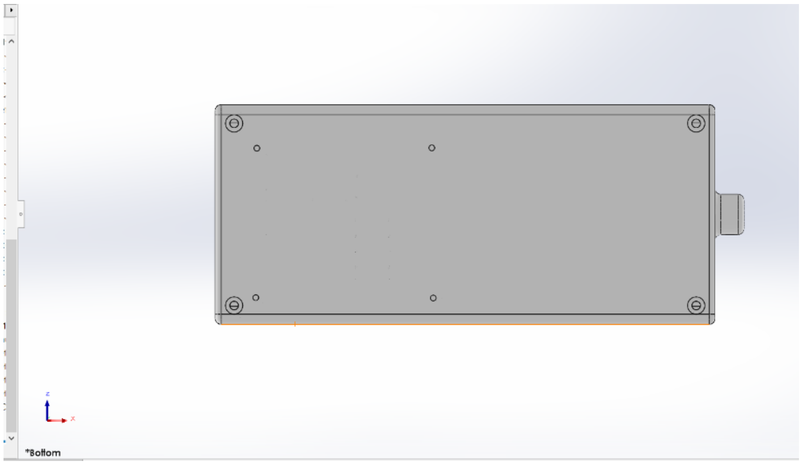


Assemble

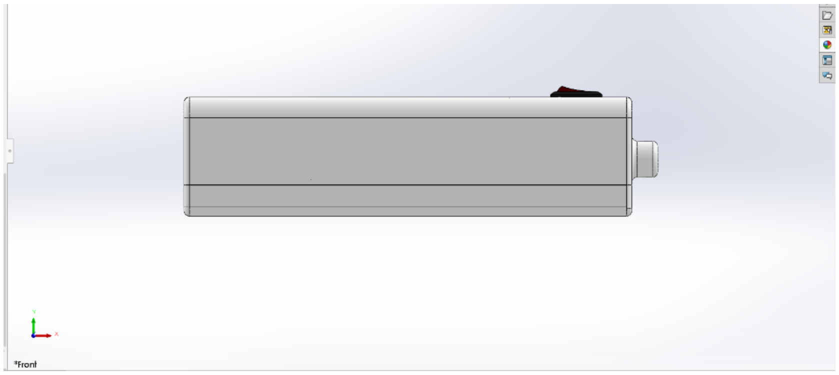
top view



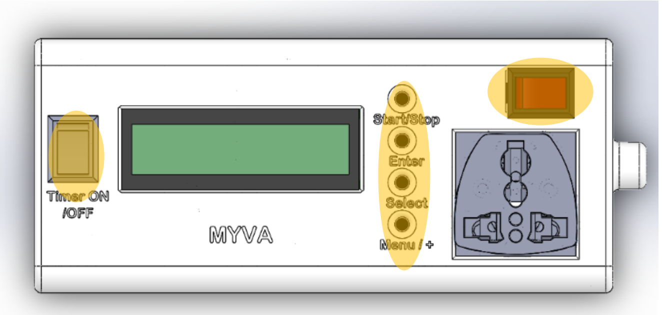
Bottom view



Front view

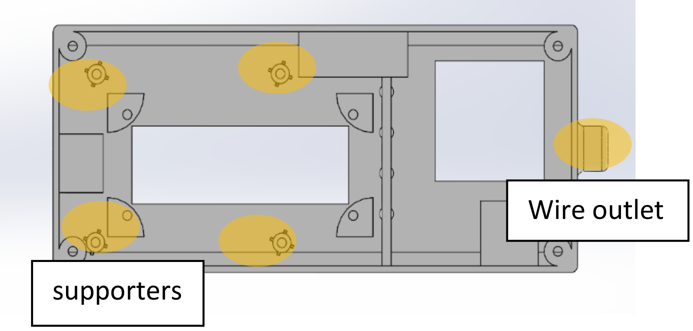


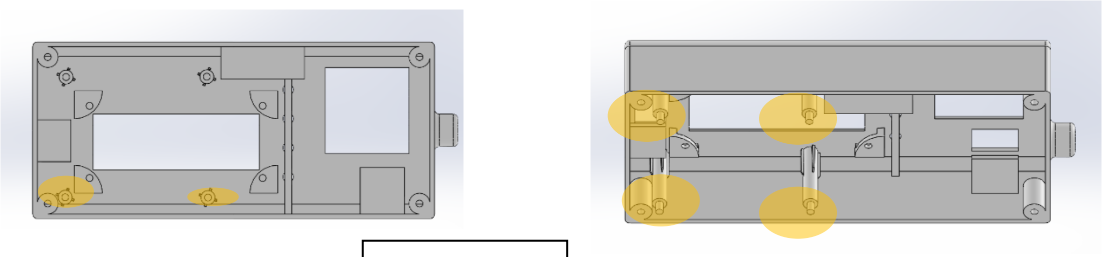
used 2mm wall thickness after considering the material cost and the strength. As the 1st stage we have only included one socket. we removed the reset button from the user interface as there is no use of a reset button to the user. So, we have used 4 mini push buttons instead of 5 push buttons and place them between the socket and lcd for easy use (can operate by the right hand). we include 2 one-way switches, one on the top of socket which uses to power ON /OFF the socket and other one near the lcd display which uses to ON/OFF the timer.



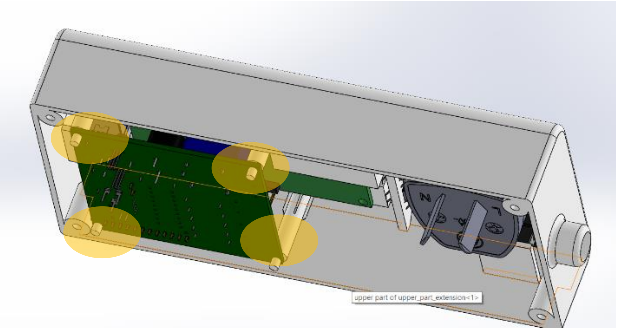
We made an outlet to the 7A wire to prevent the loss of connectivity by unnecessary motions and it is made near the socket to reduce the length of 13A wire inside the product. Lcd display will be connected to the upper part by 4 3mm nuts and bolts using the 4 supporters.

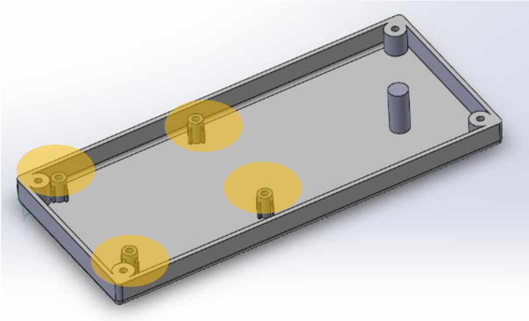
PCB is mounted using mounting bosses (4 pin bosses). its pin side is on the upper part and hole side is on the lower part as we can connect all the components on the upper part and use the lower part as a covering part.





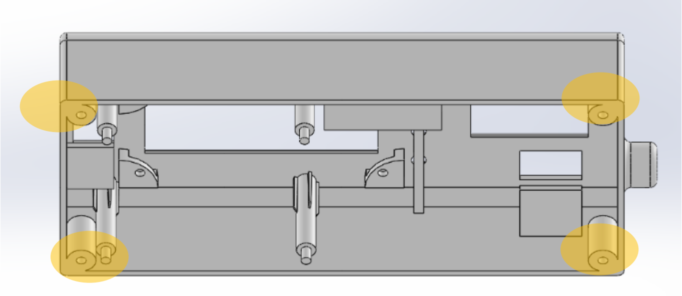
Upper part



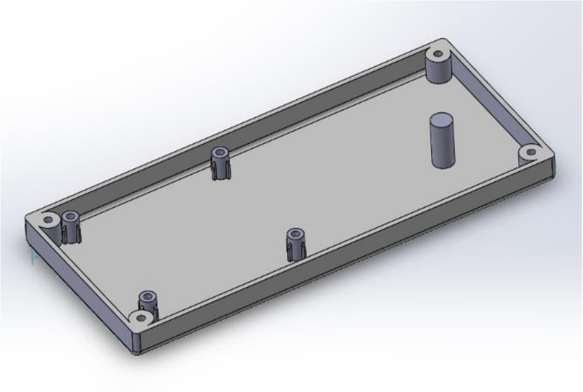


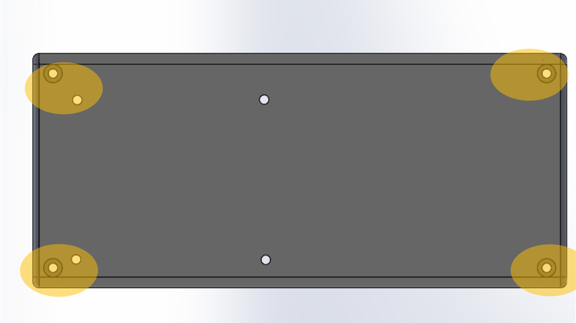
Lower part

Upper part of the product and the lower part of the product is connected using 4 nut and bolts (length= ¾). Bolts are placed in the upper part using 4 supporters.

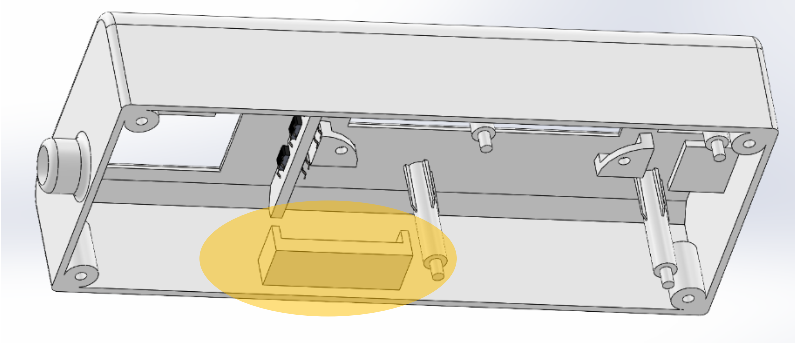


holes to insert the nuts are in the lower part.

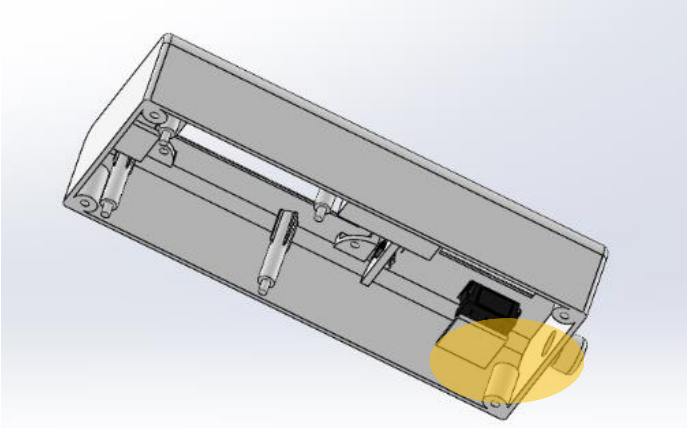




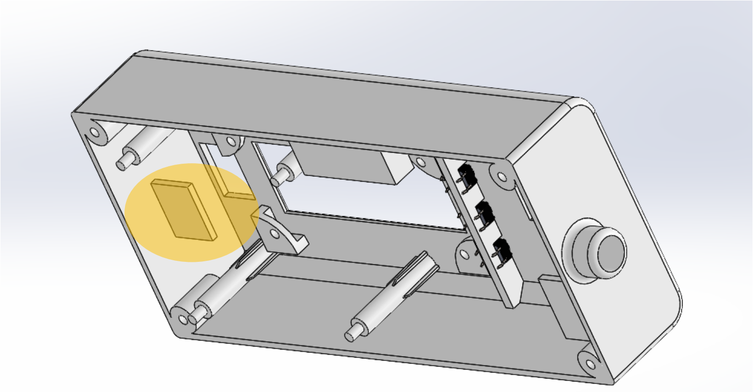
We included a part to place the power module



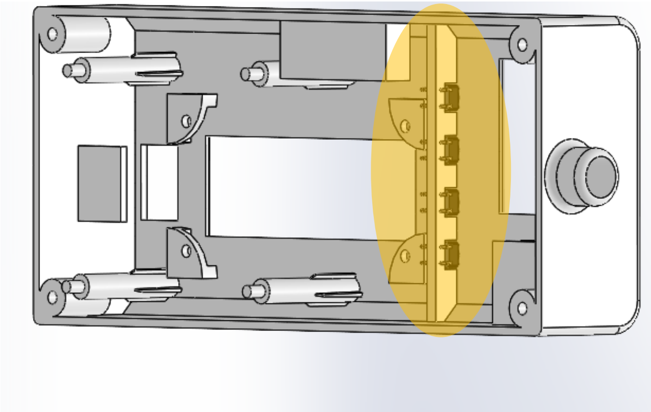
A support to socket switch



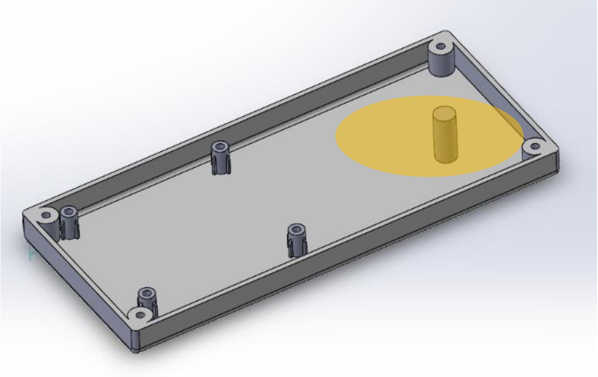
A support to timer switch



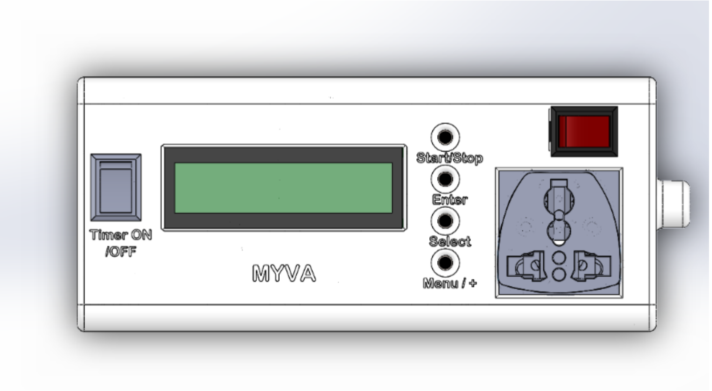
a part to place 4 buttons

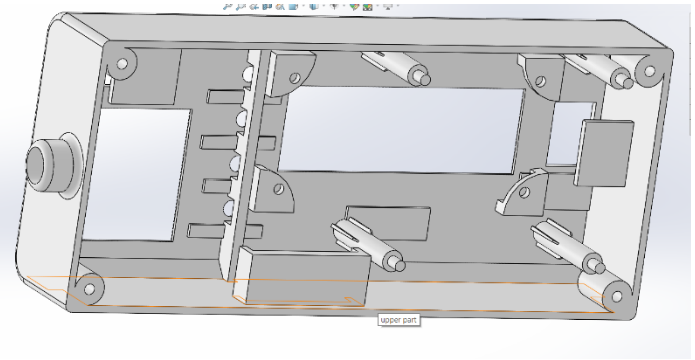


Supporter to the socket

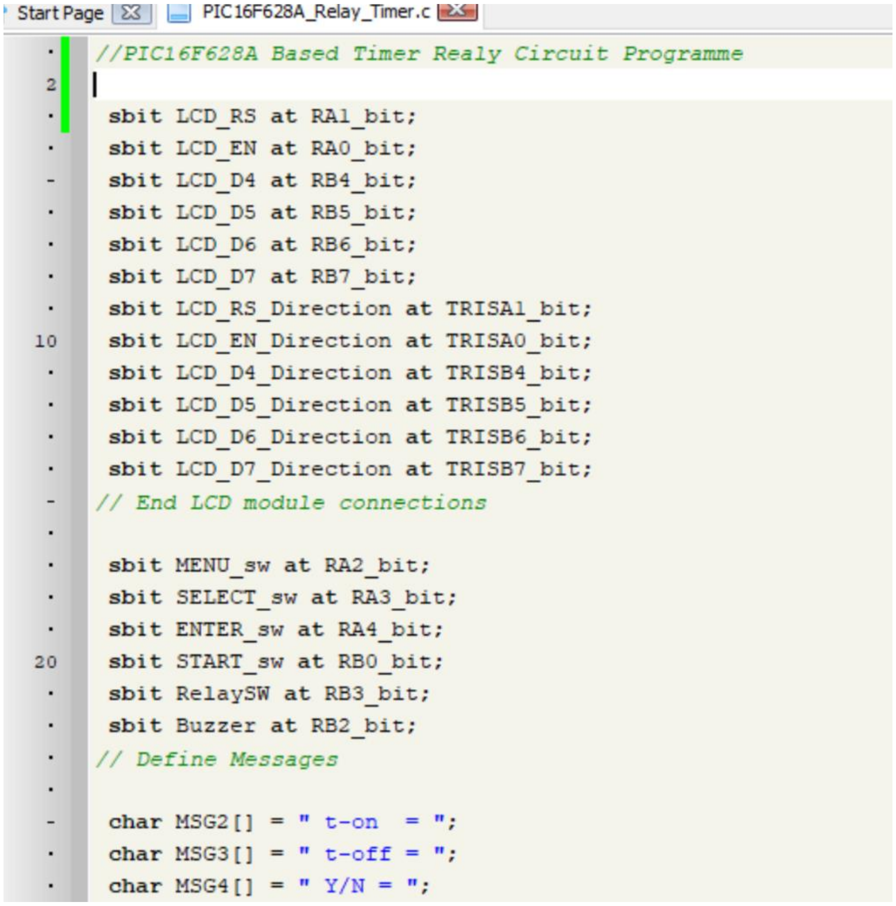


We add the labels of 4 buttons and the timer ON/OFF switch and the product brand “MYVA” as extruded cuts and add an extruded boss to ensure the strength in that area

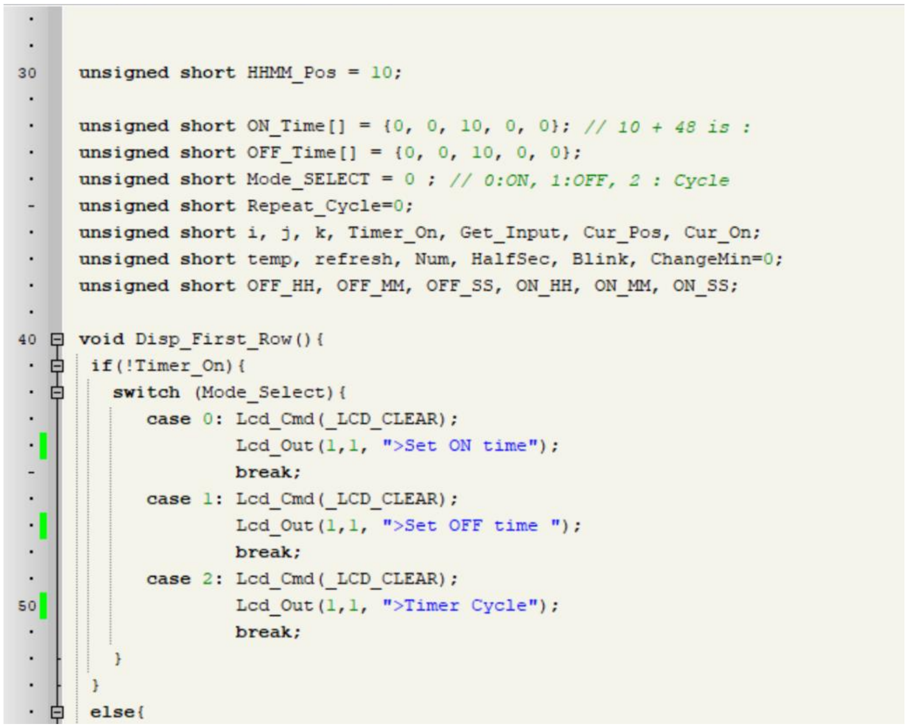




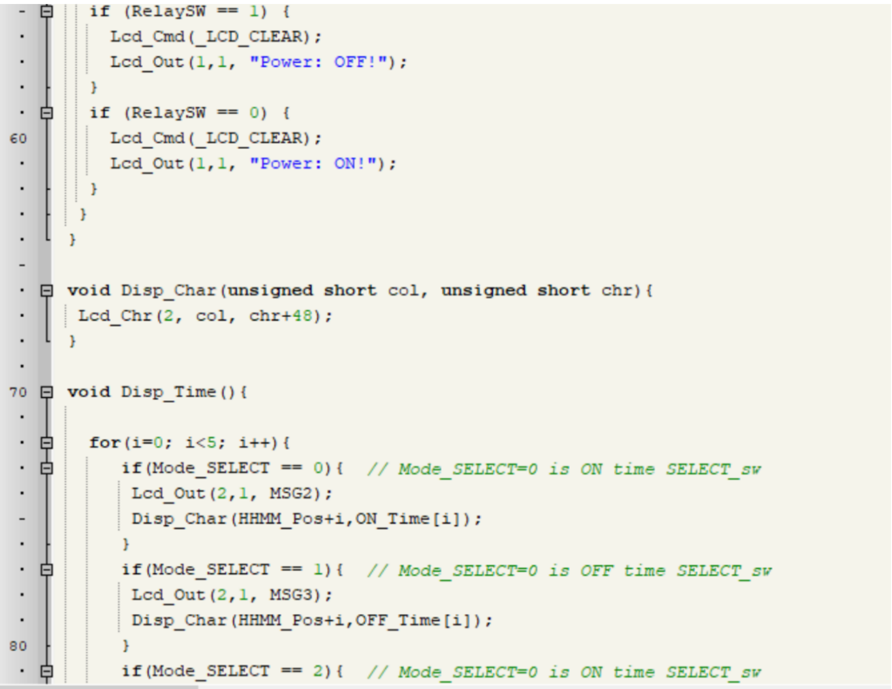
6.6 Appendix 8 - MikroC Code



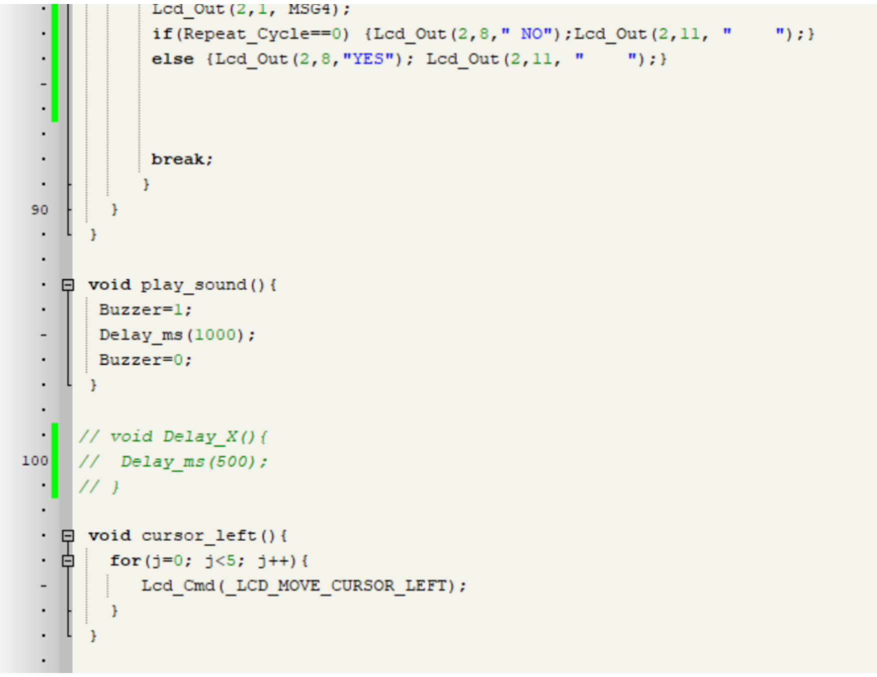
```
//PIC16F628A Based Timer Realy Circuit Programme

 sbit LCD_RS at RA1_bit;
 sbit LCD_EN at RA0_bit;
 sbit LCD_D4 at RB4_bit;
 sbit LCD_D5 at RB5_bit;
 sbit LCD_D6 at RB6_bit;
 sbit LCD_D7 at RB7_bit;
 sbit LCD_RS_Direction at TRISA1_bit;
 sbit LCD_EN_Direction at TRISA0_bit;
 sbit LCD_D4_Direction at TRISB4_bit;
 sbit LCD_D5_Direction at TRISB5_bit;
 sbit LCD_D6_Direction at TRISB6_bit;
 sbit LCD_D7_Direction at TRISB7_bit;
// End LCD module connections

 sbit MENU_sw at RA2_bit;
 sbit SELECT_sw at RA3_bit;
 sbit ENTER_sw at RA4_bit;
 sbit START_sw at RB0_bit;
 sbit RelaySW at RB3_bit;
 sbit Buzzer at RB2_bit;
// Define Messages

 char MSG2[] = " t-on  = ";
 char MSG3[] = " t-off = ";
 char MSG4[] = " Y/N = ";
```



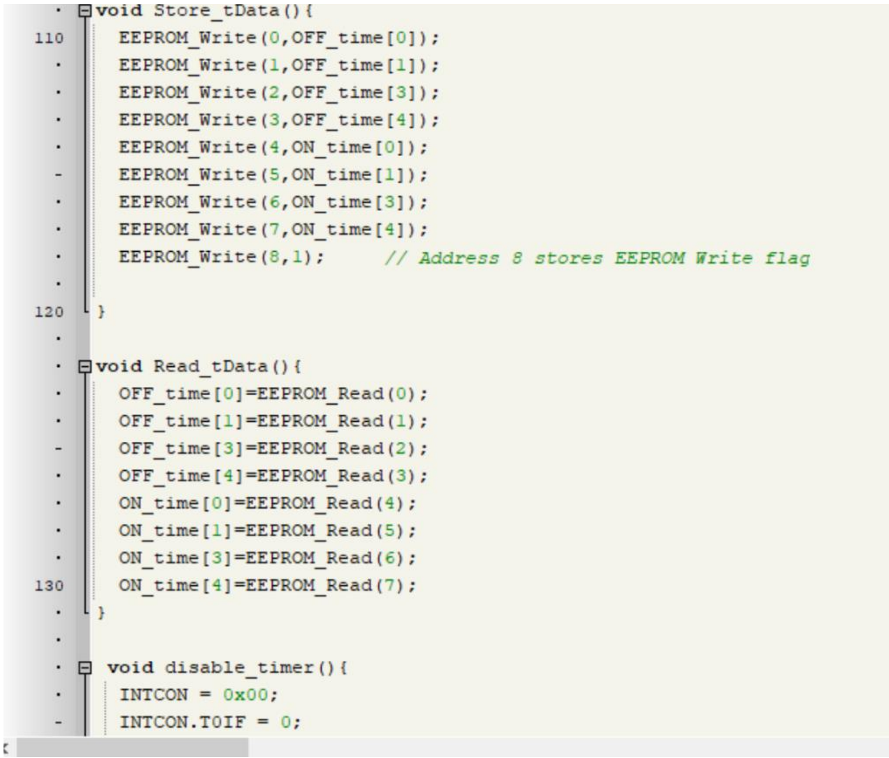
```
unsigned short HHMM_Pos = 10;

unsigned short ON_Time[] = {0, 0, 10, 0, 0}; // 10 + 48 is :
unsigned short OFF_Time[] = {0, 0, 10, 0, 0};
unsigned short Mode_SELECT = 0 ; // 0:ON, 1:OFF, 2 : Cycle
unsigned short Repeat_Cycle=0;
unsigned short i, j, k, Timer_On, Get_Input, Cur_Pos, Cur_On;
unsigned short temp, refresh, Num, HalfSec, Blink, ChangeMin=0;
unsigned short OFF_HH, OFF_MM, OFF_SS, ON_HH, ON_MM, ON_SS;

void Disp_First_Row(){
 if(!Timer_On){
   switch (Mode_Select){
      case 0: Lcd_Cmd(_LCD_CLEAR);
              Lcd_Out(1,1, ">Set ON time");
              break;
      case 1: Lcd_Cmd(_LCD_CLEAR);
              Lcd_Out(1,1, ">Set OFF time ");
              break;
      case 2: Lcd_Cmd(_LCD_CLEAR);
              Lcd_Out(1,1, ">Timer Cycle");
              break;
   }
 }
 else{
```



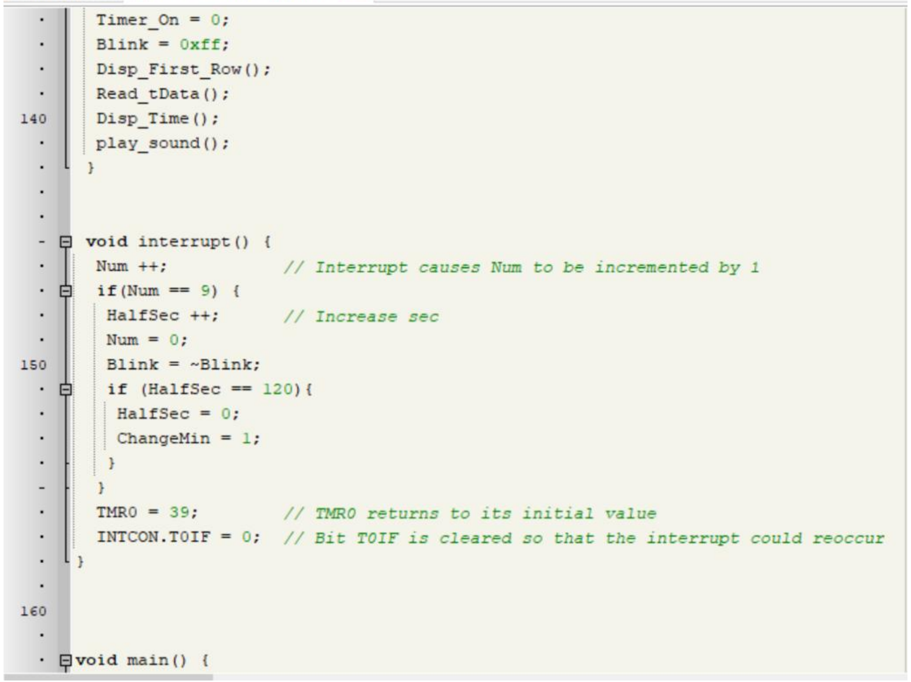
```
   if (RelaySW == 1) {
     Lcd_Cmd(_LCD_CLEAR);
     Lcd_Out(1,1, "Power: OFF!");
   }
   if (RelaySW == 0) {
     Lcd_Cmd(_LCD_CLEAR);
     Lcd_Out(1,1, "Power: ON!");
   }
 }
}

void Disp_Char(unsigned short col, unsigned short chr){
 Lcd_Chr(2, col, chr+48);
}

void Disp_Time(){

   for(i=0; i<5; i++){
      if(Mode_SELECT == 0){  // Mode_SELECT=0 is ON time SELECT_sw
       Lcd_Out(2,1, MSG2);
       Disp_Char(HHMM_Pos+i,ON_Time[i]);
      }
      if(Mode_SELECT == 1){  // Mode_SELECT=0 is OFF time SELECT_sw
       Lcd_Out(2,1, MSG3);
       Disp_Char(HHMM_Pos+i,OFF_Time[i]);
      }
      if(Mode_SELECT == 2){  // Mode_SELECT=0 is ON time SELECT_sw
```



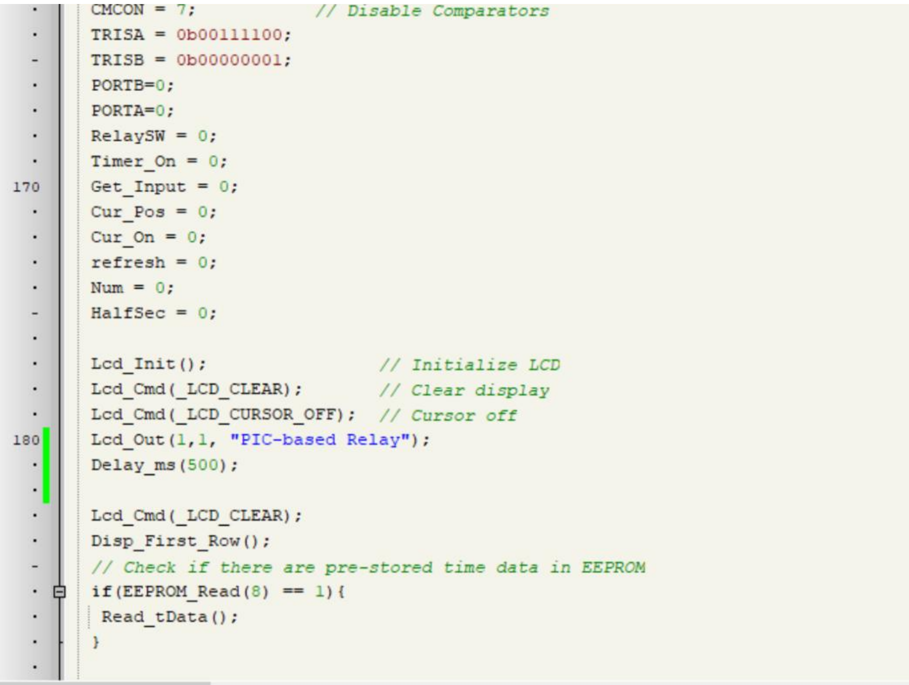
```
                Lcd_Out(2,1, MSG4);
                if(Repeat_Cycle==0) {Lcd_Out(2,8," NO");Lcd_Out(2,11, "    ");}
                else {Lcd_Out(2,8,"YES"); Lcd_Out(2,11, "    ");}



                break;
               }
           }
   }

void play_sound(){
 Buzzer=1;
 Delay_ms(1000);
 Buzzer=0;
 }

// void Delay_X(){
//  Delay_ms(500);
// }

void cursor_left(){
   for(j=0; j<5; j++){
      Lcd_Cmd(_LCD_MOVE_CURSOR_LEFT);
   }
 }
```



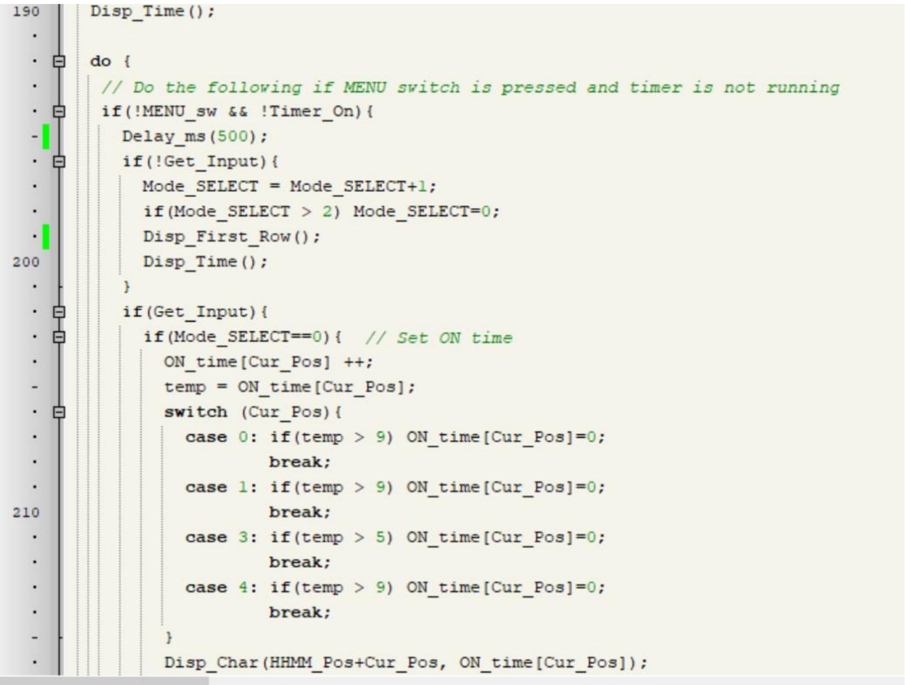
```
void Store_tData(){
  EEPROM_Write(0,OFF_time[0]);
  EEPROM_Write(1,OFF_time[1]);
  EEPROM_Write(2,OFF_time[3]);
  EEPROM_Write(3,OFF_time[4]);
  EEPROM_Write(4,ON_time[0]);
  EEPROM_Write(5,ON_time[1]);
  EEPROM_Write(6,ON_time[3]);
  EEPROM_Write(7,ON_time[4]);
  EEPROM_Write(8,1);       // Address 8 stores EEPROM Write flag

}

void Read_tData(){
  OFF_time[0]=EEPROM_Read(0);
  OFF_time[1]=EEPROM_Read(1);
  OFF_time[3]=EEPROM_Read(2);
  OFF_time[4]=EEPROM_Read(3);
  ON_time[0]=EEPROM_Read(4);
  ON_time[1]=EEPROM_Read(5);
  ON_time[3]=EEPROM_Read(6);
  ON_time[4]=EEPROM_Read(7);
}

 void disable_timer(){
  INTCON = 0x00;
  INTCON.T0IF = 0;
```



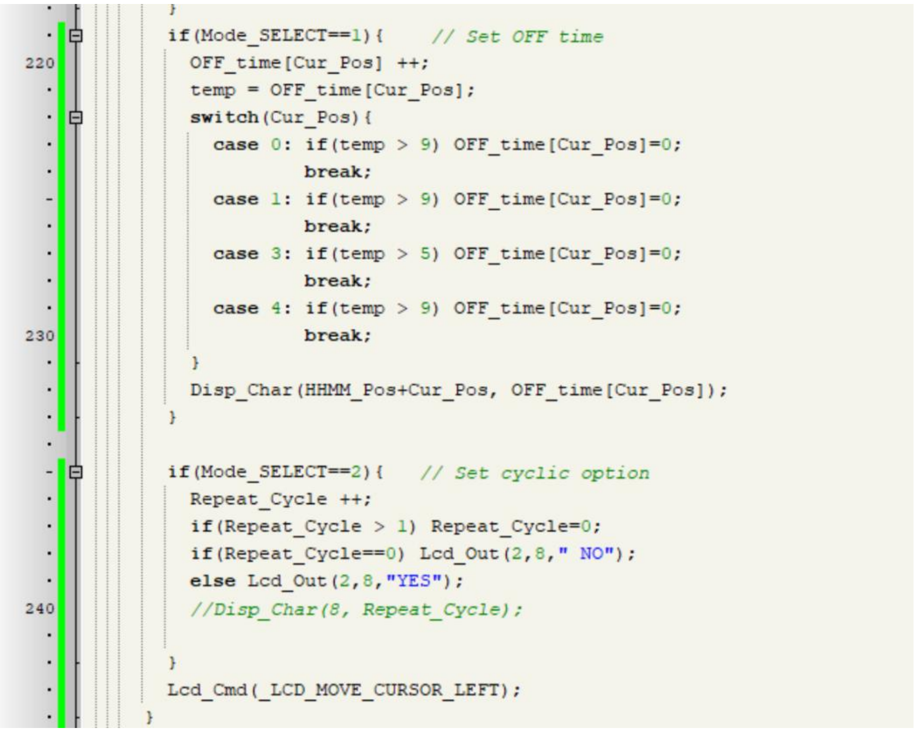
```
    Timer_On = 0;
    Blink = 0xff;
    Disp_First_Row();
    Read_tData();
    Disp_Time();
    play_sound();
  }



void interrupt() {
    Num ++;              // Interrupt causes Num to be incremented by 1
    if(Num == 9) {
      HalfSec ++;        // Increase sec
      Num = 0;
      Blink = ~Blink;
      if (HalfSec == 120){
        HalfSec = 0;
        ChangeMin = 1;
      }
    }
    TMR0 = 39;           // TMR0 returns to its initial value
    INTCON.T0IF = 0;     // Bit T0IF is cleared so that the interrupt could reoccur
}



void main() {
```



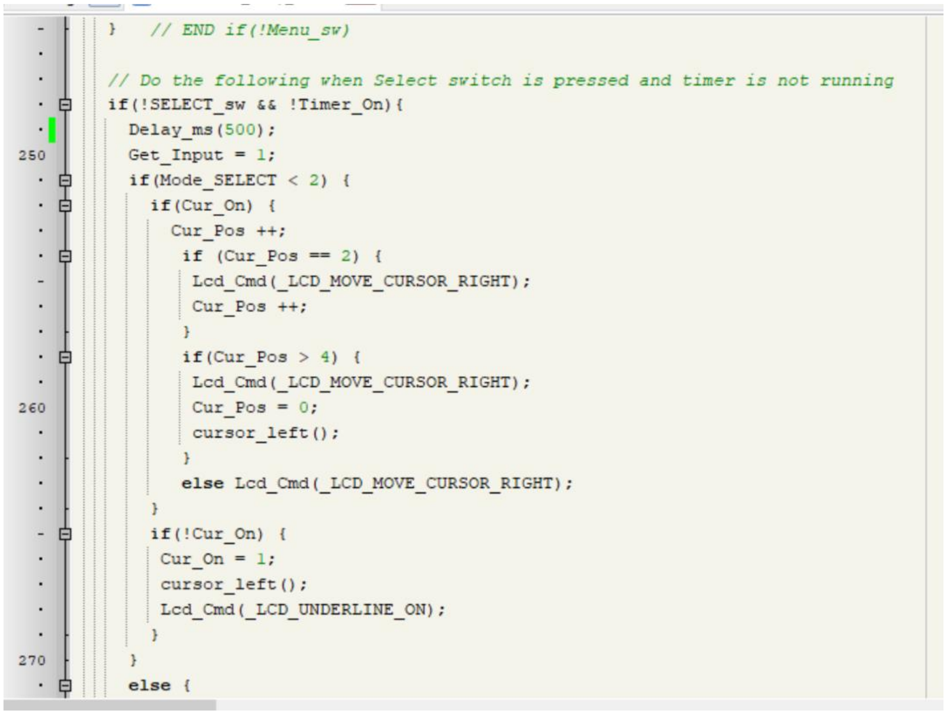
```
    CMCON = 7;                 // Disable Comparators
    TRISA = 0b00111100;
    TRISB = 0b00000001;
    PORTB=0;
    PORTA=0;
    RelaySW = 0;
    Timer_On = 0;
    Get_Input = 0;
    Cur_Pos = 0;
    Cur_On = 0;
    refresh = 0;
    Num = 0;
    HalfSec = 0;

    Lcd_Init();                     // Initialize LCD
    Lcd_Cmd(_LCD_CLEAR);            // Clear display
    Lcd_Cmd(_LCD_CURSOR_OFF);       // Cursor off
    Lcd_Out(1,1, "PIC-based Relay");
    Delay_ms(500);

    Lcd_Cmd(_LCD_CLEAR);
    Disp_First_Row();
    // Check if there are pre-stored time data in EEPROM
    if(EEPROM_Read(8) == 1){
      Read_tData();
    }
```



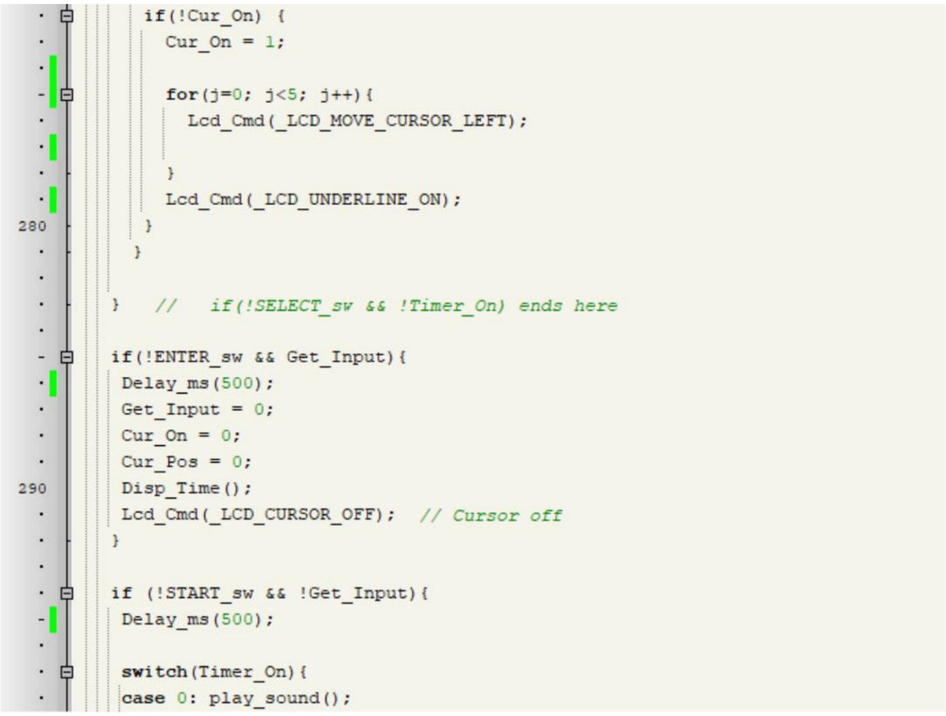
```
Disp_Time();

do {
  // Do the following if MENU switch is pressed and timer is not running
  if(!MENU_sw && !Timer_On){
    Delay_ms(500);
    if(!Get_Input){
      Mode_SELECT = Mode_SELECT+1;
      if(Mode_SELECT > 2) Mode_SELECT=0;
      Disp_First_Row();
      Disp_Time();
    }
    if(Get_Input){
      if(Mode_SELECT==0){  // Set ON time
        ON_time[Cur_Pos] ++;
        temp = ON_time[Cur_Pos];
        switch (Cur_Pos){
          case 0: if(temp > 9) ON_time[Cur_Pos]=0;
                  break;
          case 1: if(temp > 9) ON_time[Cur_Pos]=0;
                  break;
          case 3: if(temp > 5) ON_time[Cur_Pos]=0;
                  break;
          case 4: if(temp > 9) ON_time[Cur_Pos]=0;
                  break;
        }
        Disp_Char(HHMM_Pos+Cur_Pos, ON_time[Cur_Pos]);
```



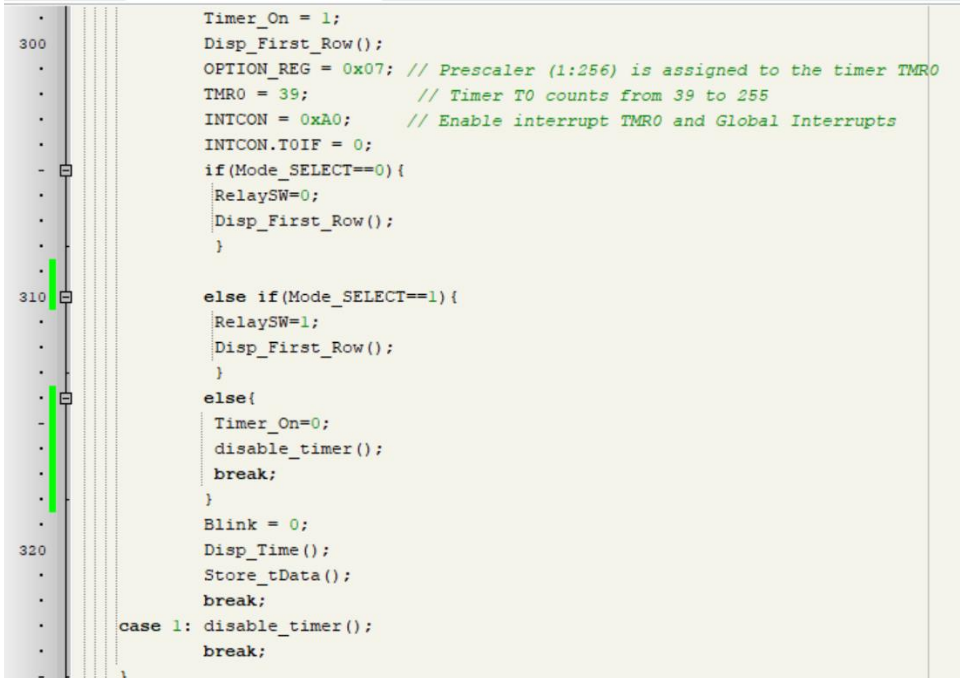
```
      }
      if(Mode_SELECT==1){     // Set OFF time
        OFF_time[Cur_Pos] ++;
        temp = OFF_time[Cur_Pos];
        switch(Cur_Pos){
          case 0: if(temp > 9) OFF_time[Cur_Pos]=0;
                  break;
          case 1: if(temp > 9) OFF_time[Cur_Pos]=0;
                  break;
          case 3: if(temp > 5) OFF_time[Cur_Pos]=0;
                  break;
          case 4: if(temp > 9) OFF_time[Cur_Pos]=0;
                  break;
        }
        Disp_Char(HHMM_Pos+Cur_Pos, OFF_time[Cur_Pos]);
      }

      if(Mode_SELECT==2){    // Set cyclic option
        Repeat_Cycle ++;
        if(Repeat_Cycle > 1) Repeat_Cycle=0;
        if(Repeat_Cycle==0) Lcd_Out(2,8," NO");
        else Lcd_Out(2,8,"YES");
        //Disp_Char(8, Repeat_Cycle);

      }
      Lcd_Cmd(_LCD_MOVE_CURSOR_LEFT);
    }
```



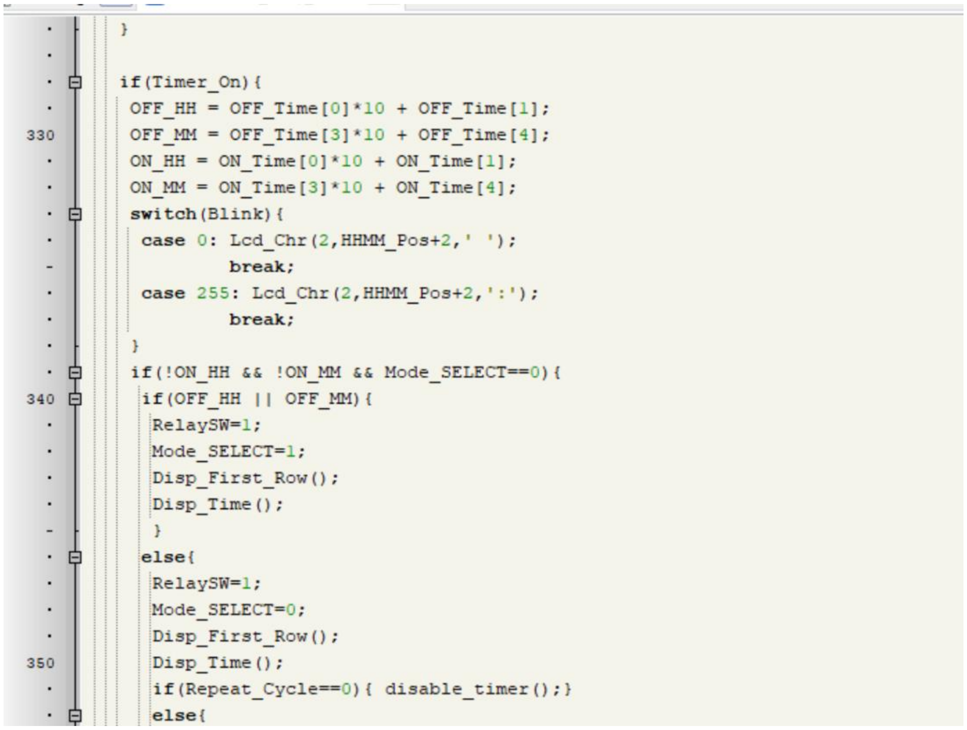
```
}   // END if(!Menu_sw)

// Do the following when Select switch is pressed and timer is not running
if(!SELECT_sw && !Timer_On){
  Delay_ms(500);
  Get_Input = 1;
  if(Mode_SELECT < 2) {
    if(Cur_On) {
      Cur_Pos ++;
       if (Cur_Pos == 2) {
        Lcd_Cmd(_LCD_MOVE_CURSOR_RIGHT);
        Cur_Pos ++;
       }
       if(Cur_Pos > 4) {
        Lcd_Cmd(_LCD_MOVE_CURSOR_RIGHT);
        Cur_Pos = 0;
        cursor_left();
       }
       else Lcd_Cmd(_LCD_MOVE_CURSOR_RIGHT);
    }
    if(!Cur_On) {
     Cur_On = 1;
     cursor_left();
     Lcd_Cmd(_LCD_UNDERLINE_ON);
    }
  }
  else {
```



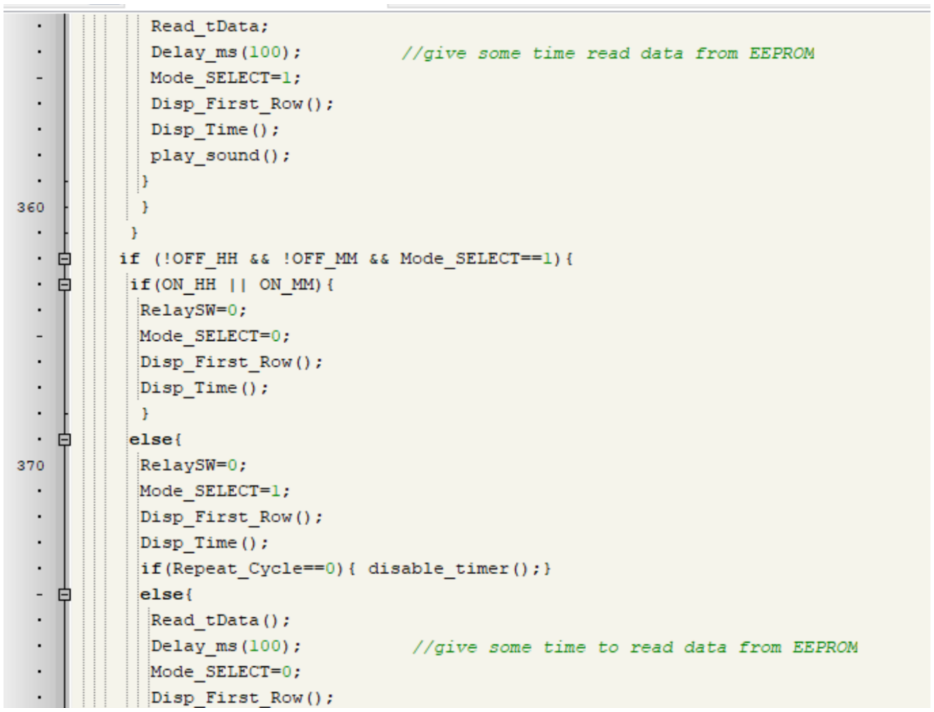
```
    if(!Cur_On) {
      Cur_On = 1;

      for(j=0; j<5; j++){
        Lcd_Cmd(_LCD_MOVE_CURSOR_LEFT);

      }
      Lcd_Cmd(_LCD_UNDERLINE_ON);
    }
   }

}  //   if(!SELECT_sw && !Timer_On) ends here

if(!ENTER_sw && Get_Input){
 Delay_ms(500);
 Get_Input = 0;
 Cur_On = 0;
 Cur_Pos = 0;
 Disp_Time();
 Lcd_Cmd(_LCD_CURSOR_OFF);  // Cursor off
}

if (!START_sw && !Get_Input){
 Delay_ms(500);

 switch(Timer_On){
 case 0: play_sound();
```



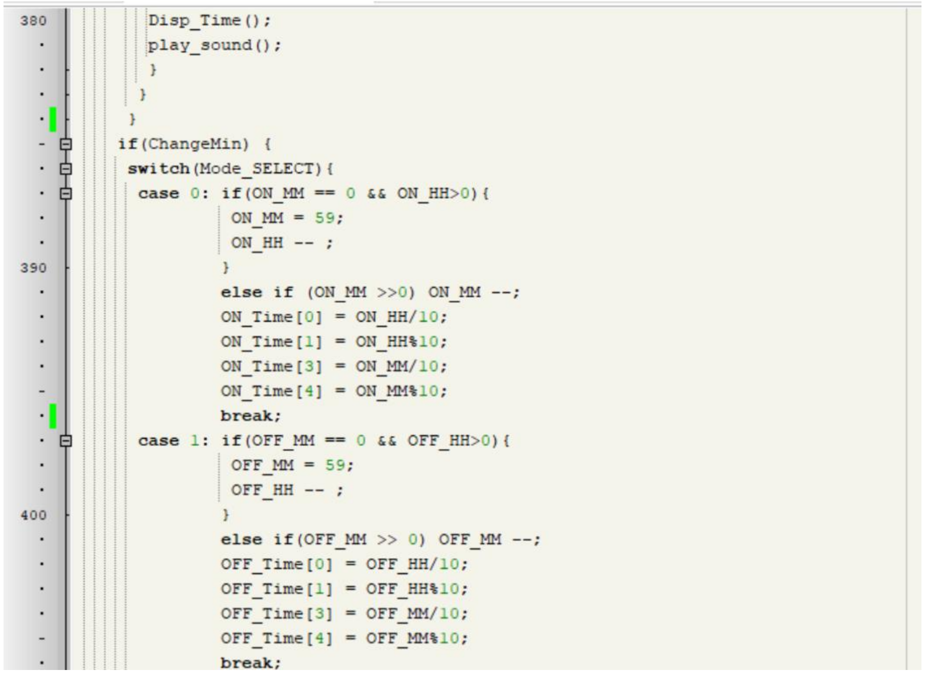
```
        Timer_On = 1;
        Disp_First_Row();
        OPTION_REG = 0x07; // Prescaler (1:256) is assigned to the timer TMR0
        TMR0 = 39;          // Timer T0 counts from 39 to 255
        INTCON = 0xA0;     // Enable interrupt TMR0 and Global Interrupts
        INTCON.T0IF = 0;
        if(Mode_SELECT==0){
         RelaySW=0;
         Disp_First_Row();
         }

        else if(Mode_SELECT==1){
         RelaySW=1;
         Disp_First_Row();
         }
        else{
         Timer_On=0;
         disable_timer();
         break;
        }
        Blink = 0;
        Disp_Time();
        Store_tData();
        break;
case 1: disable_timer();
        break;
}
```



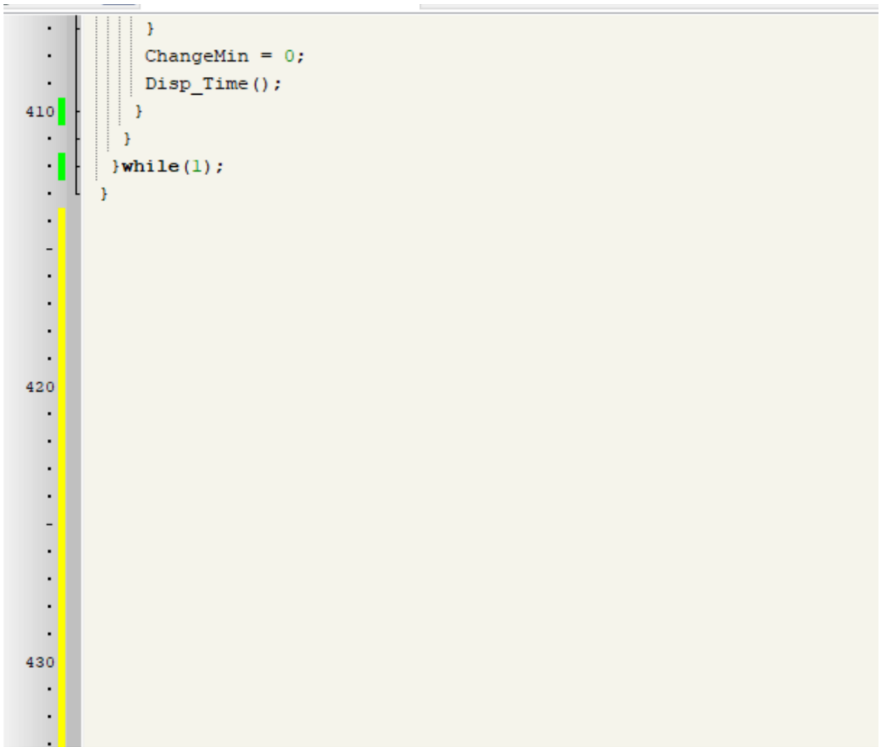
```
}

if(Timer_On){
 OFF_HH = OFF_Time[0]*10 + OFF_Time[1];
 OFF_MM = OFF_Time[3]*10 + OFF_Time[4];
 ON_HH = ON_Time[0]*10 + ON_Time[1];
 ON_MM = ON_Time[3]*10 + ON_Time[4];
 switch(Blink){
  case 0: Lcd_Chr(2,HHMM_Pos+2,' ');
          break;
  case 255: Lcd_Chr(2,HHMM_Pos+2,':');
          break;
 }
 if(!ON_HH && !ON_MM && Mode_SELECT==0){
  if(OFF_HH || OFF_MM){
   RelaySW=1;
   Mode_SELECT=1;
   Disp_First_Row();
   Disp_Time();
   }
  else{
   RelaySW=1;
   Mode_SELECT=0;
   Disp_First_Row();
   Disp_Time();
   if(Repeat_Cycle==0){ disable_timer();}
   else{
```



```
        Read_tData;
        Delay_ms(100);          //give some time read data from EEPROM
        Mode_SELECT=1;
        Disp_First_Row();
        Disp_Time();
        play_sound();
       }
      }
     }
    if (!OFF_HH && !OFF_MM && Mode_SELECT==1){
     if(ON_HH || ON_MM){
      RelaySW=0;
      Mode_SELECT=0;
      Disp_First_Row();
      Disp_Time();
      }
     else{
      RelaySW=0;
      Mode_SELECT=1;
      Disp_First_Row();
      Disp_Time();
      if(Repeat_Cycle==0){ disable_timer();}
      else{
       Read_tData();
       Delay_ms(100);          //give some time to read data from EEPROM
       Mode_SELECT=0;
       Disp_First_Row();
```



```
       Disp_Time();
       play_sound();
       }
      }
     }
    if(ChangeMin) {
     switch(Mode_SELECT){
      case 0: if(ON_MM == 0 && ON_HH>0){
                ON_MM = 59;
                ON_HH -- ;
              }
              else if (ON_MM >>0) ON_MM --;
              ON_Time[0] = ON_HH/10;
              ON_Time[1] = ON_HH%10;
              ON_Time[3] = ON_MM/10;
              ON_Time[4] = ON_MM%10;
              break;
      case 1: if(OFF_MM == 0 && OFF_HH>0){
                OFF_MM = 59;
                OFF_HH -- ;
              }
              else if(OFF_MM >> 0) OFF_MM --;
              OFF_Time[0] = OFF_HH/10;
              OFF_Time[1] = OFF_HH%10;
              OFF_Time[3] = OFF_MM/10;
              OFF_Time[4] = OFF_MM%10;
              break;
```



```
            }
            ChangeMin = 0;
            Disp_Time();
          }
        }
    }while(1);
  }
```