Smart Locker Project

Final Report by Group #8

EST. 1785 UNB
UNIVERSITY OF NEW BRUNSWICK

Project by

Sharan Nagarajan

Aali Q Alotaibi

Tanishq Tanmay

Table of contents:

Participation

Section	Sharan Nagarajan	Aali Alotaibi	Tanishq Tanmay
Background information	60%		40%
User manual	80%		20%
Hardware Block diagram	33%	33%	33%
Software block diagram	50%		50%
Keypad interfacing and testing	100%		
Lcd interfacing and testing	100%		
Password algorithm	100%		
LCD, keypad and password algorithm integration	100%		
Subsystem Specification			40%
Schematics			100%
Light system			100%
Light system testing (LDR, ADC and the LEDs)			100%
System testing - test plan and test log	60%	10%	30%

Overall contribution:

Sharan Nagarajan : 50%

Aali Q Alotaibi: 15%

Tanishq Tanmay: 35%

Background Information

Locks are of great importance and are used everywhere like jail doors, house doors, shelf lockers etc. Our project is a better version of a locker system that is being used everywhere. The cost of electrical items is cheaper than ever before. We are able to add more and more features to a particular product with almost no increase in the cost of manufacturing because of electronics.
Password Based Door Lock System is a simple product where a secure password will act as a door unlocking system. Traditional lock systems using mechanical lock and key mechanism are being replaced by new advanced techniques. These are an integration of mechanical and electronic devices, and are highly intelligent. One of the prominent features of these innovative systems is their simplicity and high efficiency. This project is proposed to be a better version of a traditional locker. The control of the locking system shall be easy to use and it should be easy to interface so that customers from any background face very minimal difficulty while setting it up across their doors.
This kind of locks are better than traditional locks due to multiple reasons, a few are mentioned down:
The keypad cover protects the lock system from atmosphere, impacts and accidental damages when the keypad is not in use.
The alarm system (in case of attempted breakage) helps in additional security which is a huge advantage over traditional lock
No need to carry keys, as the keypad is digital.

These kind of locker systems can be used as an alternative and replace the traditional locker systems with minimal increase in price.
By completing this project we hope to gain an overall understanding of using a microprocessor and how to program a microprocessor and interface it with different components. We learned how to navigate through any datasheet to know the particular requirements of a component.

User Manual

User instructions:

- Approach the keypad. Once you are close enough for the sensor to detect your presence, the flap will open giving access to the keypad.
- Now enter the passcode.
- Once you enter the correct password the system will automatically accept it and verify it,
- if the password is correct, then the door lock is opened for some amount of time, after a few seconds the door lock will be closed automatically
- If the password is incorrect, then the door lock will remain closed and a message appears on the screen “password incorrect” indicating that the password is incorrect and the user has to try again
- You will get 5 attempts to enter the right passcode. The display shall accordingly acknowledge your attempts.
- On failure to do so, the keypad shall be disabled and an alarm will go off indicating a forced intrusion. (see “What to do in case of an alarm trigger?” section)

What to do in case of an alarm trigger?:

- The alarm trigger will be activated if the user has entered the wrong password for more than 5 times. In order to confirm the safety of the locker, the alarm will not be turned off by the user.
- A button will be set up near the alarm which when pressed will reset the whole system thereby, enabling the keypad to take inputs again and silencing the alarm.
- In order to disable the alarm and reset the whole system, there is a hidden button in the system. The presence of the hidden button is not to be mentioned to the general users and must be known only to the authorized personnel who are responsible for managing the locker.
- Once the reset button is pressed the user has to wait for sometime, then the system is returned to its initial state, the user can enter his password again and the user and the user can have 5 tries again.

Troubleshooting common user problems:

- If the keypad is non responsive check its connection to the system
- If the LED display is not working try the following actions:
- The power supply might be cut off, check if the power supply connection is proper.
- If you are using batteries then try changing the batteries

Safety warnings:

- The prototype is not designed to withstand high physical stress, therefore subjecting the keypad and LCD to high physical stress must be avoided
- The photoresistor must be cleaned at a regular basis. The dust accumulation over the photoresistor may cause the illuminating system to malfunction

Hardware Block Diagram

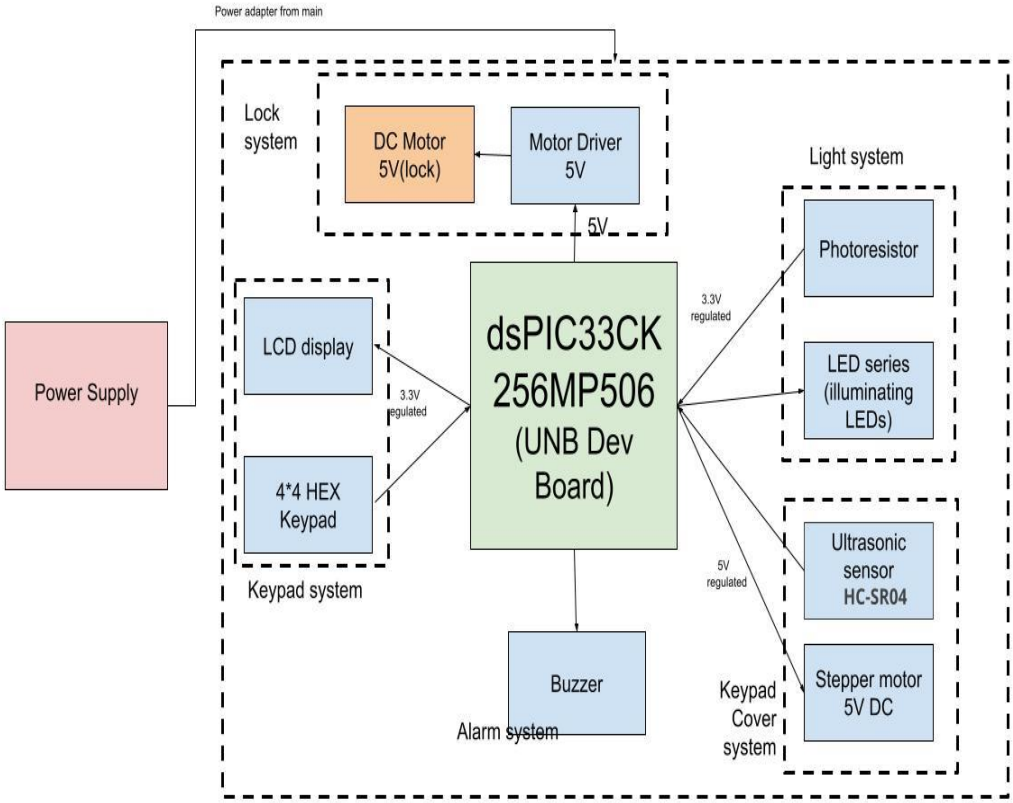

Figure: 1

The following components are used in our project:

1. Power system: can be a 9/12v battery or a from a12v power output adapter (MODEL: SOY-1200200US)
2. LCD display -1602A
3. Keypad: 4*4 hex keypad
4. Active buzzer
5. Ultrasonic sensor - HCSR04
6. Stepper motor 5V DC
7. Photoresistor
8. LED series
9. 5V motor driver

Software Block Diagram:

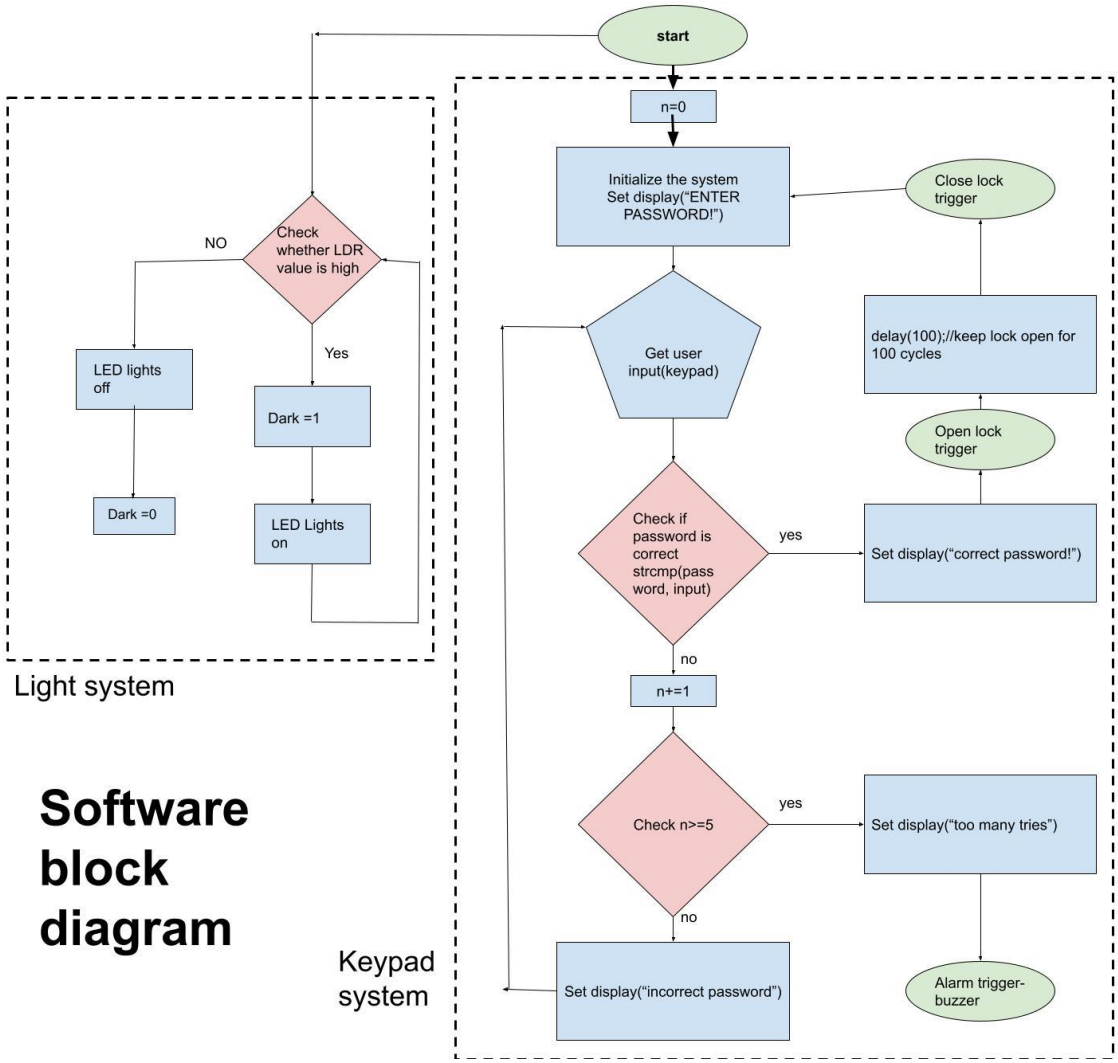

Figure: 2

System Analysis and description

Detailed Block Design

The system consists of 5 block components, namely:

1. dsPIC33CK256MP506
2. LCD display
3. Keypad
4. Alarm system
5. Lock system

dsPIC33CK256MP506

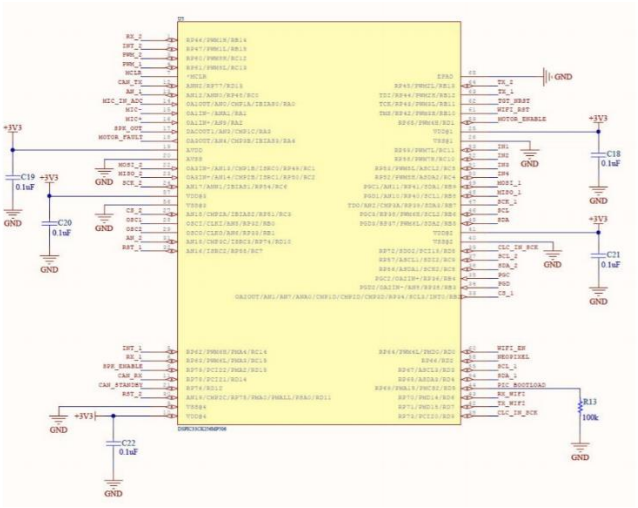

Figure: 3

The UNB Dev board will act as the central hub and will connect all other systems. It contains dsPIC33CK256MP506 microcontroller which is the heart of the Dev board. This will be the main controller which we will be using for our controller system.

LCD display

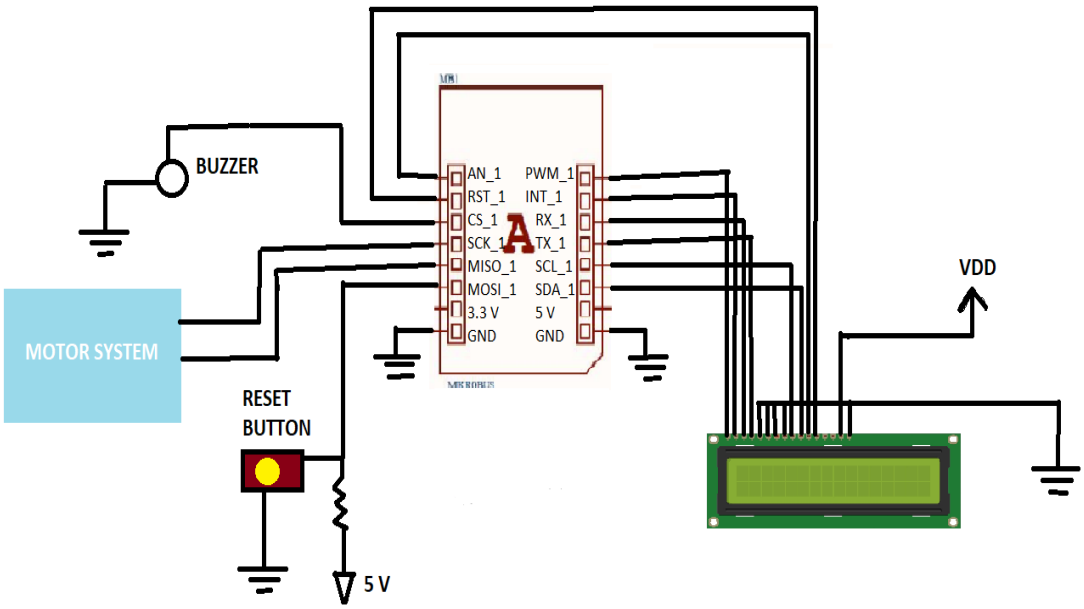

Figure: 4

The keypad system is the user interface for visual representation. LCD display will be used to interact with the user. The LCD will first display “Enter the password” and if the user enters the incorrect password it will display “You have entered the wrong password. Please try again” and then it will go back to displaying “Enter the password”. And if the user enters the right password it will send the signal to the Lock system. If the user enters the wrong password more than 5 times then a signal will be sent to the alarm system.
The LCD display is operated in a 4 bit mode.

Keypad

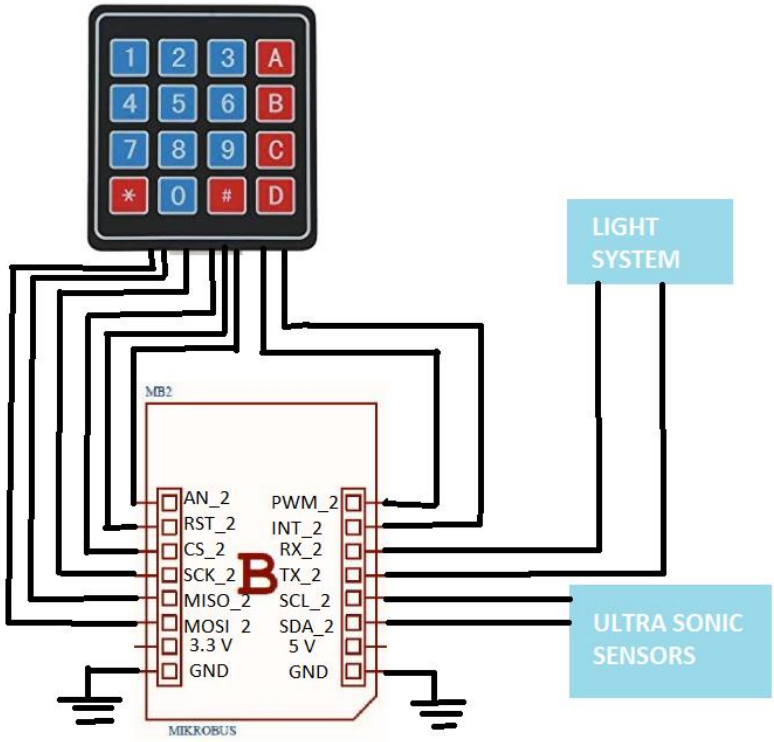

Figure: 5

It is a 4* 4 Hex keypad which contains 16 buttons a user can press, it is a means by which the user can enter his password. The keypad works on the basis of allotting 4 rows as a write pin and the 4 columns as the read pin. In this way, we will know which button is being pressed at a particular time. In order to avoid the “bounce back” nature of the buttons a delay() is function is used to take in the values of the

Alarm system

Figure: 6

The alarm system consists of an active buzzer which is triggered whenever the number of incorrect attempts reaches 5. It also contains a reset button in it, the button is to reset the entire system so that the user can continue to use this digital lock as it is before the intruder alarm is activated. This button is present in the alarm system as it must be away from the intruder and must be hidden in a secure location.

Lock system

If the correct password is entered in the keypad, the UNB board will verify the password and the lock will be opened. This contains a 5V step motor DC which will rotate 180 degrees and signifies the lock is opened. And if left idle for a long time, it will rotate 180 degrees anticlockwise and go back to the original position which will indicate that the lock remains closed.

Light System

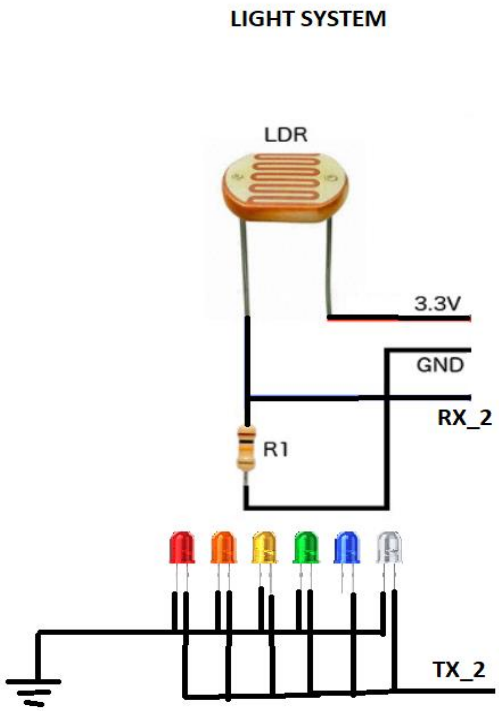

Figure: 7

Photoresistors, also known as light dependent resistors (LDR), are light sensitive devices most often used to indicate the presence or absence of light, or to measure the light intensity. In the dark, their resistance is very high, sometimes up to 1MΩ, but when the LDR sensor is exposed to light, the resistance drops dramatically, even down to a few ohms, depending on the light intensity. LDRs have a sensitivity that varies with the wavelength of the light applied and are nonlinear devices.

The Photoresitors will detect the intensity of the light and will send an Analog Signal to the microcontroller. The Microcontroller will convert the analog signal into the digital signal using the built-in ADC module and this will give a signal to the sequence of the LEDs. If the light intensity detected is low or the resistance value of LDR is very high then the LEDs will glow which is kept near the Keypad. This will make the keypad visible even in dark.

Subsystem testing:

LCD display – 1602A

Testing: the programming commands for the LCD interfacing is checked and the expected output is verified

s.no	Program command	observation	result
1.	Lcd_Cmd(0x01)	Clearing Display Screen	Working properly
2.	Lcd_Cmd(0x02)	4bit mode for LCd-Screen	Working properly
3.	Lcd_Cmd(0xc0)	Cursor beginning of second line	Working properly
4.	Lcd_Print_String("Wrong Password");	Message displayed on the screen	Working properly
5.	Lcd_Cmd(0x0e)	Display cursor blinking	Working properly
6.	Lcd_Cmd(0x38)	lcd initialize matrix 8bit	Working properly

Table: 1

4*4 hex keypad

Testing: working of each button is checked and the keypad is tested with physical stresses over the buttons

s.no	Button title	action	result
1	1	Pressed firmly	Working properly
2	2	Pressed firmly	Working properly
3	3	Pressed firmly	Working properly
4	4	Pressed firmly	Working properly
5	5	Pressed firmly	Working properly
6	6	Pressed firmly	Working properly
7	7	Pressed firmly	Working properly
8	8	Pressed firmly	Working properly
9	9	Pressed firmly	Working properly
10	A	Pressed firmly	Working properly
11	B	Pressed firmly	Working properly
12	C	Pressed firmly	Working properly
13	D	Pressed firmly	Working properly
14	*	Pressed firmly	Working properly
15	0	Pressed firmly	Working properly
16	#	Pressed firmly	Working properly

Table: 2

Buzzer

Testing:

1. The capability of the buzzer produces the right amount of sound is verified.
2. Capability to produce the total number of beeps (20 beeps) in the desired time is verified.

Reset button mechanism

Testing:

1. To check if the reset button is working properly
2. When the button is pressed, the system reads the input as 0.
3. When the button is not pressed the system reads the input as 1, which is the expected result. Therefore, the result is verified.

Light system

S no.	Task done	Observation	Result
1	LDR connected with microcontroller	LDR did not detect the light intensity	Did not work properly
2	Test run connecting with a single LED	Led did not glow when resistance is high	Did not work properly
3	Test run 2 for single LED	LED glows when the light is dim	Worked
4	Test run 3 for multiple LEDs	LED glows when the light intensity is low	Worked

Table: 3

Alarm System

We tried using ESP32 module which was built-in the UNB Dev board but we were not able to send the signal through any wireless medium to the Alarm

system so we went with a wired connection as it will be more reliable than a wireless source as the noise interference would be less and also the wired connection will take less time compared to the wireless connection.

System Testing

Testing:

After integrating the whole system, it is essential to test the system to check if the whole system is working as intended

Test plan:

1. Performance of the password algorithm to work properly in multiple scenarios
    a. User enters complete password
    b. User enters wrong password
    c. User enters incomplete password
    d. User enters at delayed time intervals
2. Working of the system if incorrect password is entered more than 5 times
    a. Working of alarm
    b. Checking if the reset button is working
    c. Checking if the reset button is working
    d. Checking if the alarm is capable of ringing for the required time
3. Working of the illumination system is checked

Test log:

s.no:	Testing action:	Observation	result
1	User enters complete password	"Password Matched" is displayed on the LCD screen and the door lock is unlocked. after sometime the door is locked	Working As expected
2	User enters wrong password	“Wrong Password” is displayed and the buzzer is activated for a certain amount of time, then the buzzer is deactivated and then the system is returned back to normal	Working As expected

3	User enters incomplete password	The system waits until the password is completed. The system does not do anything until it received more characters from the user as an input	Working As expected
4	User enters passwords at delayed time intervals	The system receives the characters from the users even incase of large time delays between entering the characters on the hex keypad	Working As expected
5	User enters more than 5 incorrect tries	“too many tries” is displayed and the buzzer is activated	
6	Working of alarm	The alarm is working properly upto the required sound and with the required time delay in between the buzzer beeps as programmed	Working As expected
7	Reset button is pressed	The buzzer is turned off and after some time delay the system is reset back to its original state and the user again has 5 tries remaining	Working As expected
8	Checking if the alarm is capable of ringing for the required time	The alarm continues to ring indefinitely unless the reset button is not pressed	Working As expected

Table: 4

Conclusion

The overall project was a very new experience for all of us. We gained practical knowledge and hands on experience while working with the PIC microcontroller and other components. We got to know about the real world scenario about the Engineering Design Process and kind of experienced it in our project in Milestone 2. We got to know how the Engineering firms and the customers deal. The smart clock system was not as easy as we initially thought it would be. We had great difficulty while trying to make the LCD work. And interfacing the LCD with the password algorithm was also a very hard part. We also searched and studied a lot on how to use a wireless connection using ESP32 built-in the UNB Dev board for the alarm system. The project gave each one of us an experience of working as a team and was a very new experience.

Reference:

- https://ww1.microchip.com/downloads/en/DeviceDoc/dsPIC33CK256MP508-Family-Data-Sheet-DS70005349H.pdf

https://youtu.be/RXQ01zkJKyI

- https://circuitdigest.com/microcontroller-projects/4x4-keypad-interfacing-with-pic16f877a/
- https://diy.waziup.io/sensors/light/photoresistor.html

Appendix

Keypad password system code:

```
//#include<reg51.h>
#include "xc.h"
#include<string.h>
#include <libpic30.h>

#define FCY 16000000UL //Crystal Frequency, used in delay

//Micro Define
#define D4 LATBbits.LATB12
#define D5 LATCbits.LATC15
#define D6 LATCbits.LATC14
#define D7 LATCbits.LATC13

//Pin configuaration for lcd-screen
#define RS LATCbits.LATC0
#define RW LATDbits.LATD4
#define EN LATDbits.LATD3

//Pin configuaration for servo-motor
#define m1 LATBbits.LATB8
#define m2 LATBbits.LATB7

//Pin configuaration for Keypad: row
#define X_1    LATCbits.LATC1
#define X_2    LATCbits.LATC2
#define X_3    LATCbits.LATC6
#define X_4    LATCbits.LATC3

//Pin configuaration for Keypad: col
#define Y_1    PORTDbits.RD11
#define Y_2    PORTDbits.RD10
#define Y_3    PORTCbits.RC12
#define Y_4    PORTBbits.RB15

//Pin configuaration for Buzzer
#define buzzer LATBbits.LATB2

//Pin configuration for reset button
#define rst   PORTBbits.RB9

//Globar Array Declaration for Storing Password
char masterpass[]="12345";
char id[5];
```

```
//Function declaration
void port_init(); //function to set the pin configuration
//void lcdint(); // Function to initialize Lcd-Screen
//void lcddis(char *); //Function to Display a string
void delay(int); //Delay Function
//void lcdcmd(char); // Function to command Lcd
//void lcddata(char); // Function to display a single character in screen
//char lcdkey(); // Lcd Keyboard function
//char scan_key(); //Function to take single key input
void door_open(); //Function to open door
void door_close(); //Function to close Door
void sounder(); // Function to Buzz sounder

void Lcd_SetBit(char data_bit); //Based on the Hex value Set the Bits of the Data Lines
void Lcd_Clear();
void Lcd_Set_Cursor(char a, char b);
void Lcd_Start();
void Lcd_Cmd(char a);
void Lcd_Print_String(char *a);
void Lcd_Print_Char(char data);  //Send 8-bits through 4-bit mode
char keypad_scanner(void);

void main(){
        port_init();
        int n;
        char key;
   int rf = 0;
   int tries =0;


   Lcd_Clear(); //Intitializing LCD-Screen
   Lcd_Print_String("Door is Locked");
     //Lcd_Cmd(0xc0);
     //void Lcd_Print_String("Enter Passcode:");
   delay(100);
   while(1){
     Lcd_Cmd(0x01);  //Clearing Display Screen
     Lcd_Cmd(0x02); //4bit mode for LCd-Screen
     Lcd_Print_String("Enter Password:");
     Lcd_Cmd(0xc0); // Cursor beggining of second line
     n=0;
     while(n<5){
       key=keypad_scanner();
       id[n]=key;
       //Lcd_Print_Char(key);
       delay(100);
       n++;
     }
```

```
      Lcd_Cmd(0x01); //ClearingLCD-Screen
      Lcd_Cmd(0x02); // 4bit mode for LCD-Screen


      if(strcmp(masterpass,id)==0){
         Lcd_Print_String("Password Matched");
         delay(200);
         door_open(); //opening door
         delay(6000);
         door_close(); //closing door
         rf=1;
         tries=0;
         Lcd_Cmd(0x01); //Clearing LCD-Screen
         Lcd_Cmd(0x02);        //4bit mode for LCD-Screen

      }
      else{
         Lcd_Print_String("Wrong Password");
         sounder();       //Buzzing Sounds
         delay(200);
         tries+=1;
         Lcd_Cmd(0x01); //Clearing LCD-Screen
         Lcd_Cmd(0x02);        //4bit mode for LCD-Screen
         Lcd_Print_String("Please Try Again");
         delay(200);
         Lcd_Cmd(0x01); //Clearing LCD-Screen
         Lcd_Cmd(0x02);        //4bit mode for LCD-Screen
      }
      if (tries>=5){
         Lcd_Print_String("Wrong Password");
         while(rst==1){
         sounder();       //Buzzing Sounds
         delay(200);
         }
         Lcd_Cmd(0x01); //Clearing LCD-Screen
         Lcd_Cmd(0x02);        //4bit mode for LCD-Screen
         tries=0;
      }

   }
}

void port_init(){
   TRISBbits.TRISB12 = 0;
   TRISCbits.TRISC15 = 0;
   TRISCbits.TRISC14 = 0;
   TRISCbits.TRISC13 = 0;

   //Pin configuaration for lcd-screen
```

```
    TRISCbits.TRISC0 = 0;//RS
    TRISDbits.TRISD4 = 0;//RW
    TRISDbits.TRISD3 = 0;//EN

    //Pin configuaration for servo-motor
    TRISBbits.TRISB8 = 0; //m1
    TRISBbits.TRISB7 = 0;//m2

    //Pin configuaration for Keypad: row
    TRISCbits.TRISC1 = 0;//X1
    TRISCbits.TRISC2 = 0;//X2
    TRISCbits.TRISC6 = 0;//X3
    TRISCbits.TRISC3 = 0;//X4

    //Pin configuaration for Keypad: col
    TRISDbits.TRISD11 = 1; //Y1
    TRISDbits.TRISD10 = 1; //Y2
    TRISCbits.TRISC12 = 1; //Y3
    TRISBbits.TRISB15 = 1; //Y4

    //Pin configuaration for Buzzer
    TRISBbits.TRISB2 = 0;//BUZZER

    TRISBbits.TRISB9 = 1; //reset
}


void Lcd_SetBit(char data_bit) //Based on the Hex value Set the Bits of the Data Lines
{
    if(data_bit& 1)
        D4 = 1;
    else
        D4 = 0;

    if(data_bit& 2)
        D5 = 1;
    else
        D5 = 0;

    if(data_bit& 4)
        D6 = 1;
    else
        D6 = 0;

    if(data_bit& 8)
        D7 = 1;
    else
        D7 = 0;
}
```

```
//Lcd-Screen Initialization Function

void Lcd_Clear(){
        Lcd_Cmd(0x38);      //  lcd initialize matrix 8bit
        delay(2);
        Lcd_Cmd(0x01);      // Clearing Lcd-Screen
        delay(2);
        Lcd_Cmd(0x80);      // lcd screen row-0 col-0
        delay(2);
        Lcd_Cmd(0x0e);      //Display cursor blinking
        delay(2);
}
//setting cursor function-new
void Lcd_Set_Cursor(char a, char b)
{
   char temp,z,y;
   if(a== 1)
   {
     temp = 0x80 + b - 1; //80H is used to move the curser
      z = temp>>4; //Lower 8-bits
      y = temp & 0x0F; //Upper 8-bits
      Lcd_Cmd(z); //Set Row
      Lcd_Cmd(y); //Set Column
   }
   else if(a== 2)
   {
      temp = 0xC0 + b - 1;
      z = temp>>4; //Lower 8-bits
      y = temp & 0x0F; //Upper 8-bits
      Lcd_Cmd(z); //Set Row
      Lcd_Cmd(y); //Set Column
   }
}

//start lcd
void Lcd_Start()
{
  Lcd_SetBit(0x00);
  //for(int i=1065244; i<=0; i--)  NOP();
  delay(1000);
  Lcd_Cmd(0x03);
    __delay_ms(5);
  Lcd_Cmd(0x03);
    __delay_ms(11);
  Lcd_Cmd(0x03);
  Lcd_Cmd(0x02); //02H is used for Return home -> Clears the RAM and initializes the LCD
  Lcd_Cmd(0x08); //Select Row 1
  Lcd_Cmd(0x00); //Clear Row 1 Display
```

```
  Lcd_Cmd(0x0C); //Select Row 2
  Lcd_Cmd(0x00); //Clear Row 2 Display
  Lcd_Cmd(0x06);
}

//Delay Function
void delay(int x)
{
        int i,j;
        for(i=0;i<x;i++)
        for(j=0;j<1275;j++);
}
//Lcd-Command Function
void Lcd_Cmd(char a)
{
   RS = 0;
   Lcd_SetBit(a); //Incoming Hex value
   EN  = 1;
      __delay_ms(4);
      EN  = 0;
}
//Lcd-Display String Function
void Lcd_Print_String(char *a)
{
   int i;
   for(i=0;a[i]!='\0';i++)
     Lcd_Print_Char(a[i]);  //Split the string using pointers and call the Char function
}

//Lcd-Data Function to Send Char to Lcd Data port
void Lcd_Print_Char(char data)  //Send 8-bits through 4-bit mode
{
  char Lower_Nibble,Upper_Nibble;
  Lower_Nibble = data&0x0F;
  Upper_Nibble = data&0xF0;
  RS = 1;           // => RS = 1
  Lcd_SetBit(Upper_Nibble>>4);           //Send upper half by shifting by 4
  EN = 1;
  //for(int i=2130483; i<=0; i--)  NOP();
  delay(1000)
  EN = 0;
  Lcd_SetBit(Lower_Nibble); //Send Lower half
  EN = 1;
  //for(int i=2130483; i<=0; i--)  NOP();
  delay(1000);
  EN = 0;
}

// Buzzer Function
```

```
void sounder(){
	int i;
	for(i=0;i<20;i++){
		buzzer=1; //buzzer on
		delay(100);
		buzzer=0; //buzzer off
	}
}
//Function to Open Door
void door_open(){
	Lcd_Cmd(0x01); //Clearing LCD-Screen
	Lcd_Cmd(0x02); //4bit mode for LCD-Screen
	Lcd_Print_String("Opening Door...");
	m1=1; //Motor on
	m2=0; //motor off
}
//Function to Close Door
void door_close(){
	Lcd_Cmd(0x01);
	Lcd_Cmd(0x02);
	Lcd_Print_String("Closing Door...");
	m1=0;
	m2=0;
	delay(20);
	//m1=1;
	//m2=0;
	m1=0;
	m2=1;
	delay(500);
	m1=0;
	m2=0;
}
//Function to scan single Key and return it to main Function
char keypad_scanner(void)
{
	X_1 = 0; X_2 = 1; X_3 = 1; X_4 = 1;
	if (Y_1 == 0) { __delay_ms(100); while (Y_1==0); return '1';
	}
	if (Y_2 == 0) { __delay_ms(100); while (Y_2==0); return '2';
	}
	if (Y_3 == 0) { __delay_ms(100); while (Y_3==0); return '3';
	}
	if (Y_4 == 0) { __delay_ms(100); while (Y_4==0); return 'A';
	}

	X_1 = 1; X_2 = 0; X_3 = 1; X_4 = 1;
	if (Y_1 == 0) { __delay_ms(100); while (Y_1==0); return '4';
	}
	if (Y_2 == 0) { __delay_ms(100); while (Y_2==0); return '5';
```

```
        }
        if (Y_3 == 0) { __delay_ms(100); while (Y_3==0); return '6';
        }
        if (Y_4 == 0) { __delay_ms(100); while (Y_4==0); return 'B';
        }

        X_1 = 1; X_2 = 1; X_3 = 0; X_4 = 1;
        if (Y_1 == 0) { __delay_ms(100); while (Y_1==0); return '7';
        }
        if (Y_2 == 0) { __delay_ms(100); while (Y_2==0); return '8';
        }
        if (Y_3 == 0) { __delay_ms(100); while (Y_3==0); return '9';
        }
        if (Y_4 == 0) { __delay_ms(100); while (Y_4==0); return 'C';
        }

        X_1 = 1; X_2 = 1; X_3 = 1; X_4 = 0;
        if (Y_1 == 0) { __delay_ms(100); while (Y_1==0); return '*';
        }
        if (Y_2 == 0) { __delay_ms(100); while (Y_2==0); return '0';
        }
        if (Y_3 == 0) { __delay_ms(100); while (Y_3==0); return '#';
        }
        if (Y_4 == 0) { __delay_ms(100); while (Y_4==0); return 'D';
        }

    return 'n';
}
```