NITTE EDUCATION TRUST | NMAM INSTITUTE OF TECHNOLOGY

Course : Microprocessor and Microcotroller Lab.

A Mini Project Report on

SMART SECURITY SYSTEM

Submitted By

NAME	USN
LISHA RODRIGUES	4NM20EC054
MANNA MOHAMMED ZISHAN	4NM20EC055
MANOJ BHAT	4NM20EC056
MIHIR MAHESH NAYAK	4NM20EC057

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

N.M.A.M. INSTITUTE OF TECHNOLOGY, NITTE -574110

2022 – 2023

ABSTARCT

Safety is the most crucial concern of humans. We always try to keep our things between ourselves. For this reason, we still use various methods to lock our precious items like a locked diary. And when it comes to our daily life, we are more serious. In the modern age, there are so many ways to lock the door; one of them is a password-based lock system a system where you are the only one to know how to access it. It saves our daily life from various malicious problems like a thief. This system will give us the security that we want. To make our life more secure, we are going to build a password-based door lock system. This system is easy to assemble and very easy to use in our daily life. Anyone can use it to secure themselves.

As the world grows to be digitally dependent, even doors can now be designed to be secure and safe. The digital version of the lock and key holds a lot of promising features. The digital door lock is securely placed with passwords. In this electronics project, we have created a digital door lock that is controlled using a basic controller, the 8051 microcontroller. The aim of this project is to understand the fundamental working principle behind a digital door lock.

Digital Code Lock project is an electronic number lock system or an electronic combination lock using 8051 which has a preset 5-digit password stored inside the program. The system collects 5-digit user input, compares the user input with the preset password inside the program, and if the user input and stored password matches, access will be granted (by opening the door with the help of motor driver for a few seconds and closing it automatically after the stipulated time). If there is a mismatch between user input and stored password, access will be denied (by not opening the closed door that is by keeping the motor in OFF position).

II. OBJECTIVES OF THE PROJECT

a. Interfacing 4x4 keypad with 8051 microcontroller.

b. Interfacing LCD (Liquid Crystal Display) with 8051 microcontroller.

c. Interfacing L293D motor driver with 8051 microcotroller.

d. Interfacing LED with 8051 microcontroller.

e. Interfacing buzzer with 8051 microcontroller for alerting system.

III.BLOCK DIAGRAM

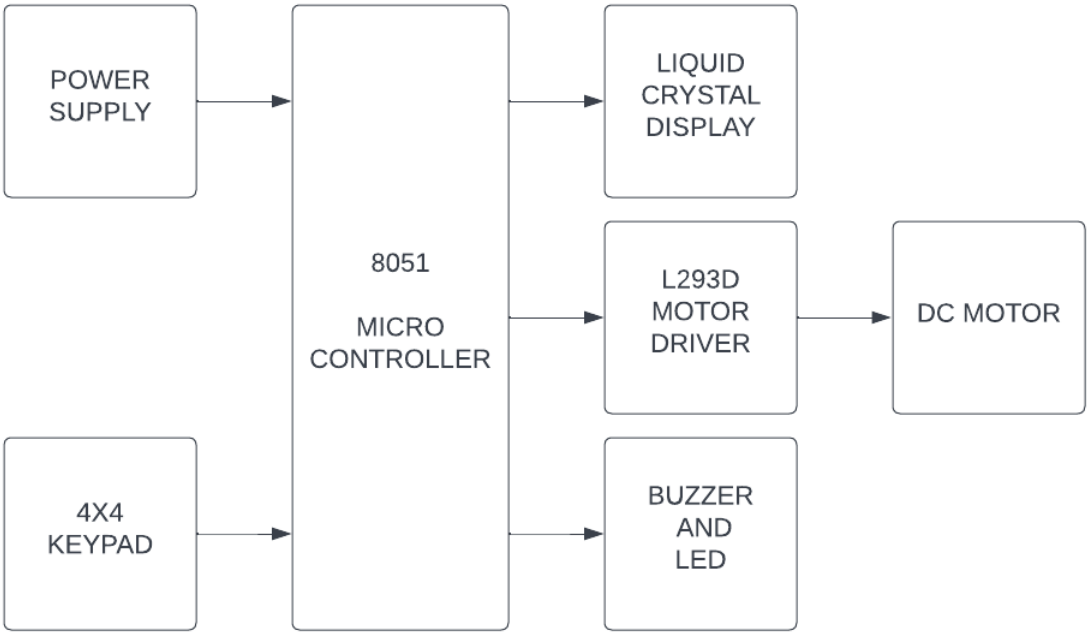


Figure 3.1 : Block diagram

This system mainly contains microcontroller, keypad module, LCD, motor driver, buzzer and LED. Microcontroller controls the complete processes like taking password form keypad module, comparing passwords with predefined password, driving motor driver, buzzer and send status to LCD display. Keypad is used for inserting password into the microcontroller. Buzzer is used for indication of wrong password, motor driver is used to indicate the opening of the door and LCD is used for displaying status or messages on it.

IV. CIRCUIT DIAGRAM

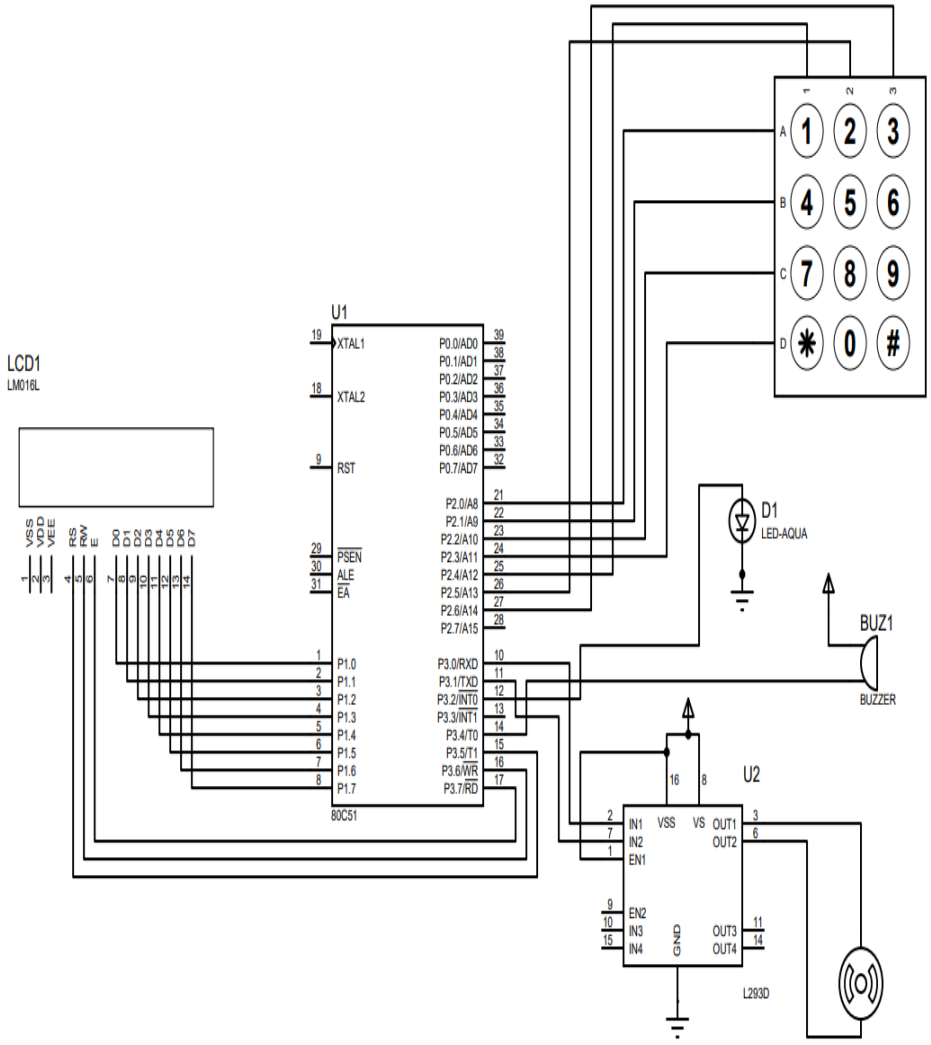


Figure 4.1 : Circuit diagram

V. Method

a. Interfacing 4x4 keypad with 8051 microcontroller.

Figure 5.1.1 : 4X4 Keypad

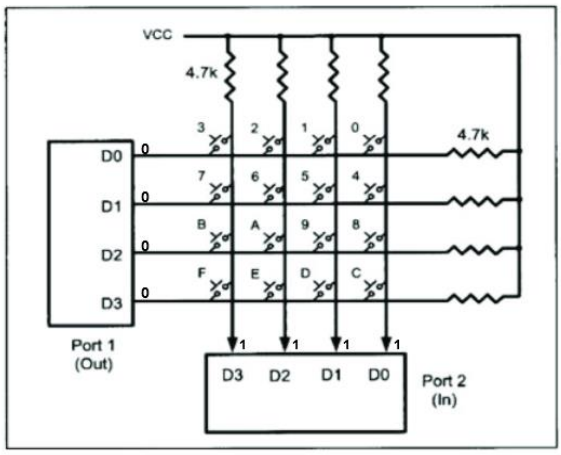
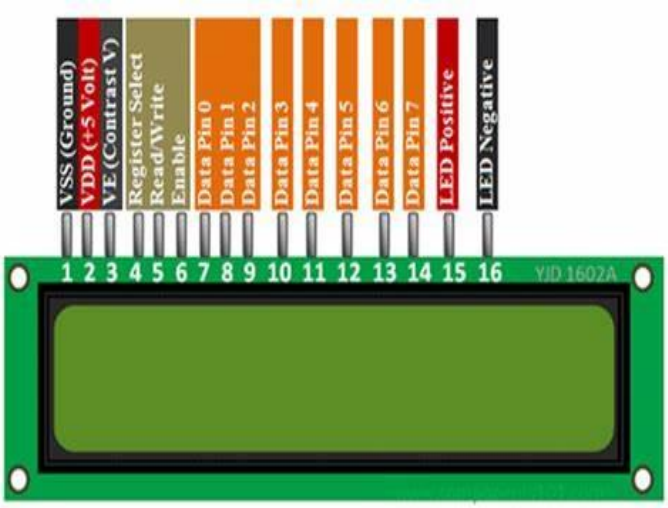


Figure 5.1.2 : Internal structure of keypad

In a keypad, push button switches are arranged in rows and columns. For a 4×4 keypad 16 switches are used and to connect to microcontroller we need 16 inputs pins. But the arrangement is changed by connecting switches in a special way. Now we need only 8 pins of microcontroller to connect keypad to it. The status of each key/switch is determined by scanning the rows or columns. The column pins (Col1–Col4) are connected to the microcontroller as the input pins and the rows pins (Row1–Row 4) are connected to the output pins of the microcontroller. Normally, all the column pins are pulled high by internal or external pull up resistors. Now we can read the status of each switch through scanning.

To detect the key that's pressed, port pins of the microcontroller are connected to the rows and columns of the matrix respectively. The port pins which are connected to the rows of the matrix is configured as an output, hence making each row logic 0. On the contrary, the port pins which are connected to the columns is configured as an input, making the column at logic 1. When a button is pressed, it changes the logic of that particular column to 0 because the button causes a short circuit between the row and the column. As an example, let us say the key with the number 1 is pressed. This causes the D1 column to go to logic 0 as it shorts row D0 with it. To find the row which has the pressed key, the microcontroller grounds the rows one by one and checks in which case the column is logic zero. This helps the microcontroller to find which key was pressed.

b. Interfacing LCD with 8051 microcontroller.



No	HEX Value	COMMAND TO LCD
1	0x01	Clear Display Screen
2	0x30	Function Set: 8-bit, 1 Line, 5x7 Dots
3	0x38	Function Set: 8-bit, 2 Line, 5x7 Dots
4	0x20	Function Set: 4-bit, 1 Line, 5x7 Dots
5	0x28	Function Set: 4-bit, 2 Line, 5x7 Dots
6	0x06	Entry Mode
7	0x08	Display off, Cursor off
8	0x0E	Display on, Cursor on
9	0x0C	Display on, Cursor off
10	0x0F	Display on, Cursor blinking
11	0x18	Shift entire display left
12	0x1C	Shift entire display right
13	0x10	Move cursor left by one character
14	0x14	Move cursor right by one character
15	0x80	Force cursor to beginning of 1st row
16	0xC0	Force cursor to beginning of 2nd row

Figure 5.2.1 : LCD

Figure 5.2.2 : Commands

16×2 Liquid Crystal Display which will display the 32 characters at a time in two rows (16 characters in one row). Each character in the display is of size 5×7 pixel matrix.

VEE pin is meant for adjusting the contrast of the LCD display and the contrast can be adjusted by varying the voltage at this pin. This is done by connecting one end of a POT to the Vcc (5V), other end to the Ground and connecting the center terminal of the POT to the VEE pin. The 16X2 LCD module has two built in registers namely data register and command register. Data register is for placing the data to be displayed, and the command register is to place the commands. The 16×2 LCD module has a set of commands each meant for doing a particular job with the display.

High logic at the RS pin will select the data register and low logic at the RS pin will select the command register. If we make the RS pin high and the put a data in the 8 bit data line (DB0 to DB7), the LCD module will recognize it as a data to be displayed. If we make RS pin low and put a data on the data line, the module will recognize it as a command. R/W pin is meant for selecting between read and write modes. High level at this pin enables read mode and low level at this pin enables write mode E pin is for enabling the module. A high to low transition at this pin will enable the module. DB0 to DB7 are the data pins. The data to be displayed and the command instructions are placed on these pins.

c. Interfacing L293D motor driver with 8051 microcotroller.

Figure 5.3.1 : L293D motor driver IC

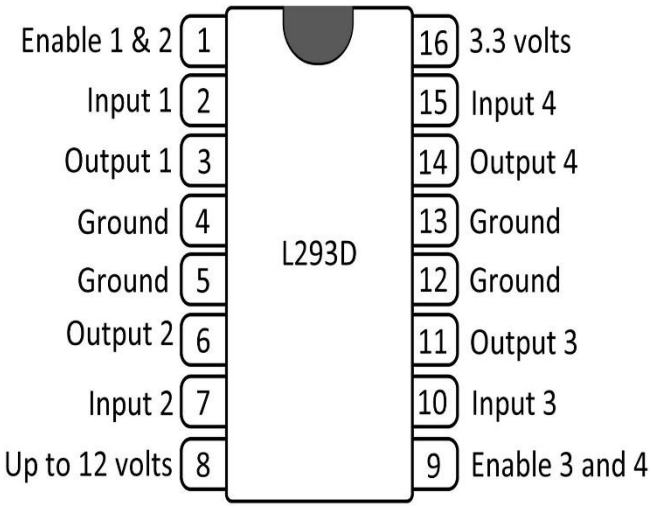


Figure 5.3.2 : Pin diagram

The maximum current that can be sourced or sunk from a 8051 microcontroller is 15 mA at 5v. But a DC Motor need currents very much more than that and it need voltages 6v, 12v, 24v etc depending upon the type of motor used. Another problem is that the back emf produced by the motor may affect the proper functioning of the microcontroller. Due to these reasons we can't connect a DC Motor directly to a microcontroller.

Interfacing 8051 with DC motor requires a motor driver. There are various types of driver ICs among which L293D is typically used for interfacing DC motor with 8051. L293D is an IC with 16 pins which are represented in the figure above. L293D IC is having ratings of 600mA per channel and DC supply voltage in the range of 4.5V to 36V. These ICs can be protected from inductive spikes by connecting higher speed clamp diodes internally. This 16 pin L293D IC can be used for controlling the direction of two DC motors. The IC L293D works based on the H-bridge concept. The voltage can be made to flow in either direction using this circuit (H-bridge) such that by changing the voltage direction the motor direction can be changed.

The practical application of L293D IC is (L293D acts as an interfacing device) DC motor interfacing with 8051 microcontroller by which we can control the speed and direction of the motor.

d. Interfacing LED with 8051 microcontroller

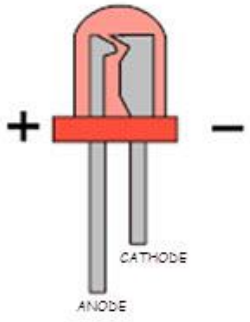


Figure 5.4.1 : LED

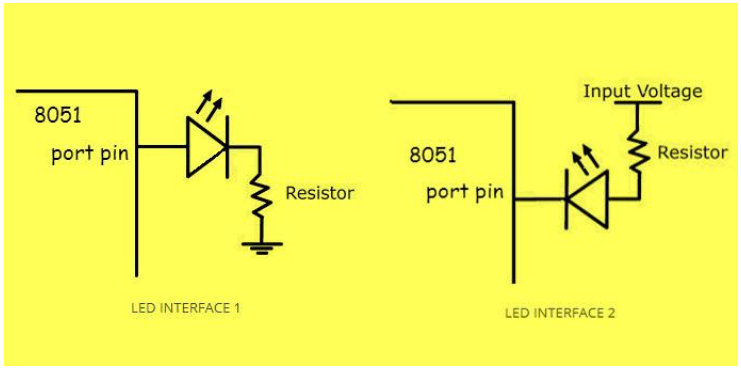


Figure 5.4.2 : Interfacing LED with MCU

A light-emitting diode (LED) is essentially a PN junction that emits a monochromatic (single color) light when operated in a forward-biased direction. LED convert electrical energy into light energy. It has two terminals positive and negative as shown in the figure.

The only way to know polarity is either to test it with a multimeter or by carefully observing inside the LED. The larger end inside the led is positive (anode) and the shorter one is -ve (cathode), that is how we find out the polarity of the LED. Another way to recognize the polarity is, connecting leads, positive terminal has more length than negative terminal.

There are two ways which we can interface LED to the Microcontroller 8051. But the connections and programming techniques will be different. The interface LED 2 is in forward biased because the input voltage of 5v connected to the positive terminal of the LED, So here the Microcontroller pin should be at LOW level. And vice versa with the interface 1 connections. LED Interface 1 will glow LED, only if the PIN value of the MC is HIGH as current flows towards the ground. LED Interface 2 will glow LED, only if the PIN value of the MC is LOW as current flows towards PIN due to its lower potential.

e. Interfacing buzzer with 8051 for alerting system.

Figure 5.5.1 : Buzzer

Buzzer is an electronic device that converts the electronic signal into buzzing noise. It can be used as electronic bell or as quiz buzzer in many applications around us. This project uses a type of buzzer known as active buzzers.Active buzzers are the simplest one to use and they are typically available in voltages from 1.5V to 24V. All you need to do is apply a DC voltage to the pins and it will make a sound typically buzzers will have polarity and the polarity is the same as an LED and a capacitor – the longer pin goes to positive. One downside of active buzzers is that the frequency of the sound is fixed and cannot be adjusted.

Positive pin of the buzzer is connected to 5V and the other pin is connected to microcontroller port. This type of connection is possible, if the current requirements of the buzzer is not more than 20mA. The output is in current source mode so that buzzer will turn ON when the output of the port is logic LOW and turns OFF when the output of the port is login HIGH.

VI. Result

Experimental setup

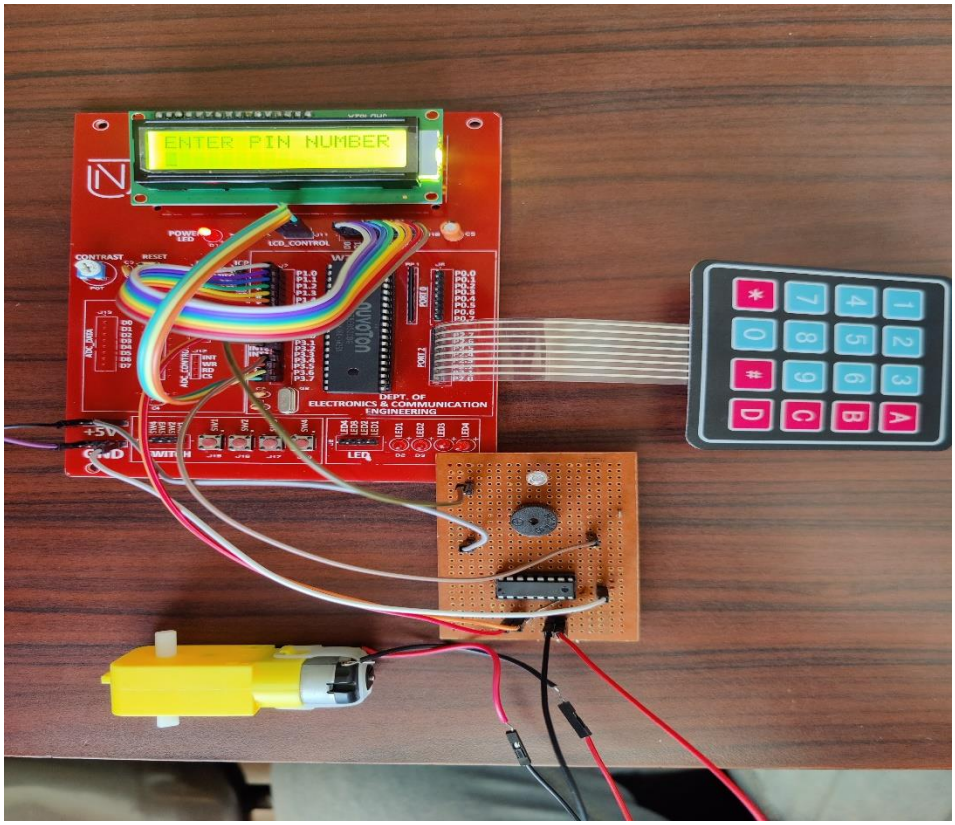


Figure 6.1.1 : Experimental setup

Discussion:

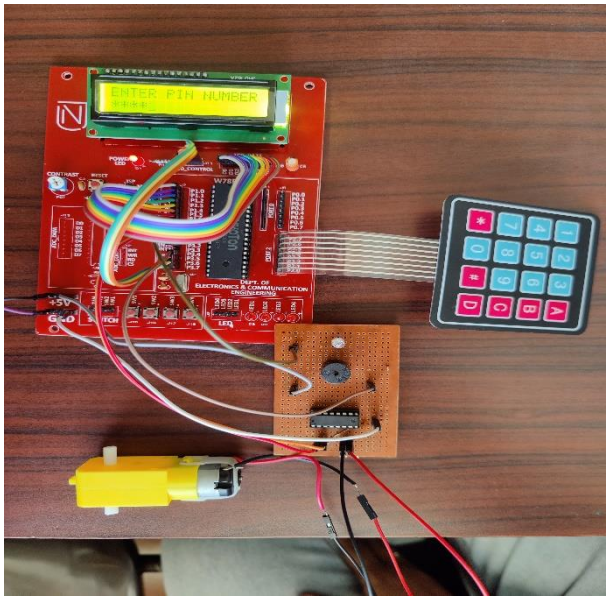


Figure 6.2.1 : Pressing keypad

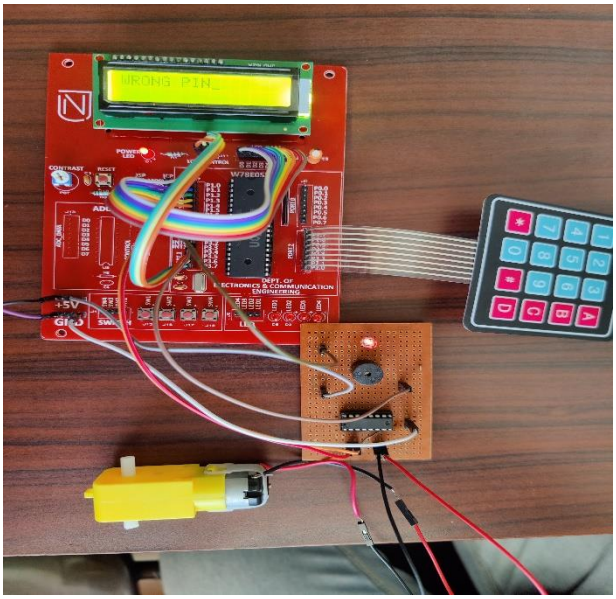


Figure 6.2.2 : Wrong pin

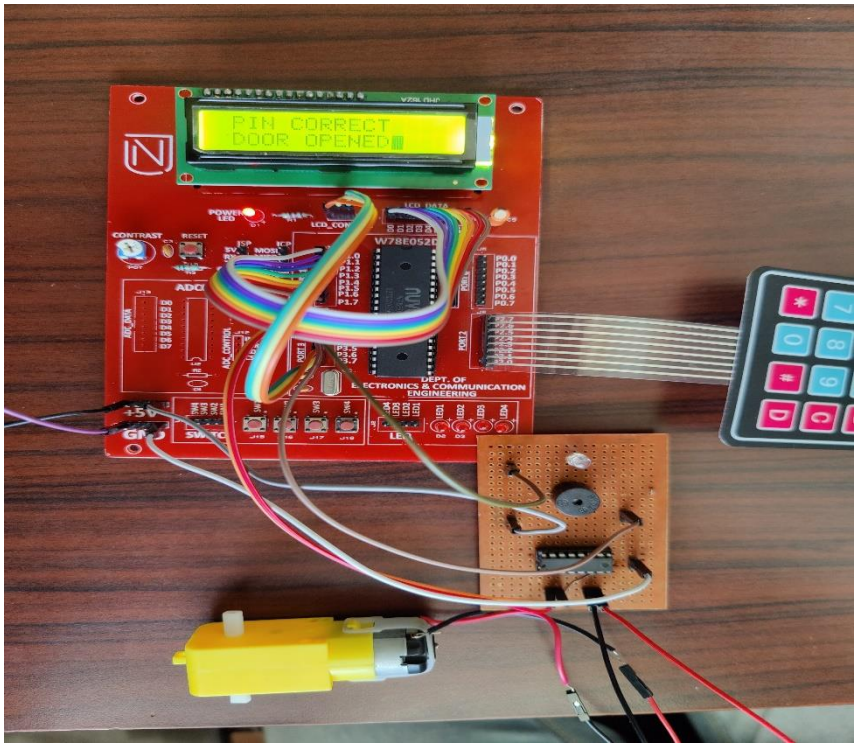


Figure 6.2.3 : Correct pin

The Fig 6.1.1 Experimental Setup shows the circuit connections of the mini – project. It consists of 16x2 LCD display, 4x4 keypad, L293D with DC motor, buzzer and LED.

Microcontroller scans the number pressed in 4x4 keypad and compares it with preset password. “PIN CORRECT” is displayed in the LCD display if the entered password matches with predefined password and L293D drives motor in clockwise direction. The motor is brought back to original position after some amount of time. Buzzer beeps and LED glows if the entered password does not matches with the predefined password.

Code snippet

```
// password based door lock system in 8051 microprocessor
#include <reg51.h>
// connected pins
// keypad rows
sbit keyrow1 = P2 ^ 7;
sbit keyrow2 = P2 ^ 6;
sbit keyrow3 = P2 ^ 5;
sbit keyrow4 = P2 ^ 4;
//keypad column
sbit keycolumn1 = P2 ^ 3;
sbit keycolumn2 = P2 ^ 2;
sbit keycolumn3 = P2 ^ 1;
sbit keycolumn4 = P2 ^ 0;
// motor pins
sbit motorpin1 = P3 ^ 0;
sbit motorpin2 = P3 ^ 1;
//buzzer and led
sbit buzzer=P3^4;
sbit LED = P3^2;
// lcd pins
sbit rs = P3 ^ 5;
sbit rw = P3 ^ 6;
sbit en = P3 ^ 7;

//functions
void lcdcmd(unsigned char);
void lcddat(unsigned char);
void lcddisplay(unsigned char *q);
char keypad();
void check();
void delay(unsigned int);
unsigned char pin[] = {"123AB"};
unsigned char Epin[5];

// main function
void main()
{
   lcdcmd(0x0F); //decimal value: 15
   lcdcmd(0x38); //decimal value: 56
   lcdcmd(0x01); //decimal value: 1
   LED = 0 ;
   while (1)
   {
      unsigned int i = 0;
      lcdcmd(0x80); //decimal value: 128
      lcddisplay("ENTER PIN NUMBER");
      delay(1000);
      lcdcmd(0xc0); //decimal value: 192
      while (pin[i] != '\0')
      {
         Epin[i] = keypad();
         delay(1000);
         i++;
      }
      check();
   }
}

//delay function
void delay(unsigned int j)
{
   int a, b;
   for (a = 0; a < j; a++)
   {
      for (b = 0; b < 10; b++)
         ;
   }
}

// lcd commands functions
void lcdcmd(unsigned char A)
{
   P1 = A;
   rs = 0;
   rw = 0;
   en = 1;
   delay(1000);
   en = 0;
}

//lcd data function
void lcddat(unsigned char i)
{
   P1 = i;
   rs = 1;
   rw = 0;
   en = 1;
   delay(1000);
   en = 0;
}

//lcd display charecters function
void lcddisplay(unsigned char *q)
{
   int k;
   for (k = 0; q[k] != '\0'; k++)
   {
      lcddat(q[k]);
   }
```

```
    delay(10000);
}

// assign keypad character value function
char keypad()
{
    int x = 0;
    while (x == 0)
    {
        // assign values for first row
        keyrow1 = 0;
        keyrow2 = 1;
        keyrow3 = 1;
        keyrow4 = 1;
        if (keycolumn1 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '1';
        }
        if (keycolumn2 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '2';
        }
        if (keycolumn3 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '3';
        }
        if (keycolumn4 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return 'A';
        }
        // assign values for second row
        keyrow1 = 1;
        keyrow2 = 0;
        keyrow3 = 1;
        keyrow4 = 1;
        if (keycolumn1 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '4';
        }
        if (keycolumn2 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '5';
        }
        if (keycolumn3 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '6';
        }
        if (keycolumn4 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return 'B';
        }
        // assign values for third row
        keyrow1 = 1;
        keyrow2 = 1;
        keyrow3 = 0;
        keyrow4 = 1;
        if (keycolumn1 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '7';
        }
        if (keycolumn2 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '8';
        }
        if (keycolumn3 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '9';
        }
        if (keycolumn4 == 0)
        {
            lcddat('*');
```

```
            delay(1000);
            x = 1;
            return 'C';
        }
        // assign values for forth row
        keyrow1 = 1;
        keyrow2 = 1;
        keyrow3 = 1;
        keyrow4 = 0;
        if (keycolumn1 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '*';
        }
        if (keycolumn2 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '0';
        }
        if (keycolumn3 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return '#';
        }
        if (keycolumn4 == 0)
        {
            lcddat('*');
            delay(1000);
            x = 1;
            return 'D';
        }
    }
}

// password check function and run the door motor
void check()
{
    //  compare the input value with the assign password value
    if (pin[0] == Epin[0] && pin[1] == Epin[1] && pin[2] == Epin[2] && pin[3] == Epin[3] && pin[4] == Epin[4])
    {
        delay(1000);
        lcdcmd(0x01); //decimal value: 1
        lcdcmd(0x81); //decimal value: 129

        // show pin is correct
        lcddisplay("PIN CORRECT");
        delay(1000);

        // door motor will run
        motorpin1 = 1;
        motorpin2 = 0;
                                delay(3100);
                                motorpin1 = 0;
        motorpin2 = 0;
                                delay(10000);

        lcdcmd(0xc1); //decimal value: 193
        // show the door is unlocked
        lcddisplay("DOOR OPENED");
        lcdcmd(0x01); //decimal value: 1
                                delay(1000);

                                        delay(300000);
                                motorpin1 = 0;
        motorpin2 = 1;
                                delay(3100);
                                motorpin1 = 0;
        motorpin2 = 0;
    }
    else
    {
        lcdcmd(0x01); //decimal value: 1
        lcdcmd(0x80); //decimal value: 128
        lcddisplay("WRONG PIN");
        delay(1000);

                               buzzer=0;
                                LED = 1;
                                delay(30000);
                                buzzer=1;
                                LED = 0 ;
                                delay(1000);

        lcdcmd(0x01); //decimal value: 1

                                motorpin1=0;
                                motorpin2=0;
                                delay(1000);
    }
}

// end
```