Experiment no: 01

Name of the experiment: Blinking LED using PIC Microcontroller.

Objectives:

1. Designing a LED blinking circuit.
2. Understanding the circuit diagram of PIC Microcontroller.

Theory: LED's are small, powerful lights that are used in many different applications. It is as simple as light ~~turing~~ turning on and off. LED is an electronic device, which emits light when the current passes through its terminals.

An LED is a two-terminal device. The two terminals are called as Cathode and Anode.

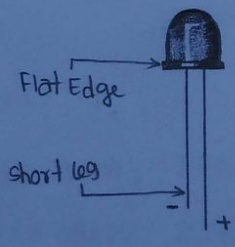
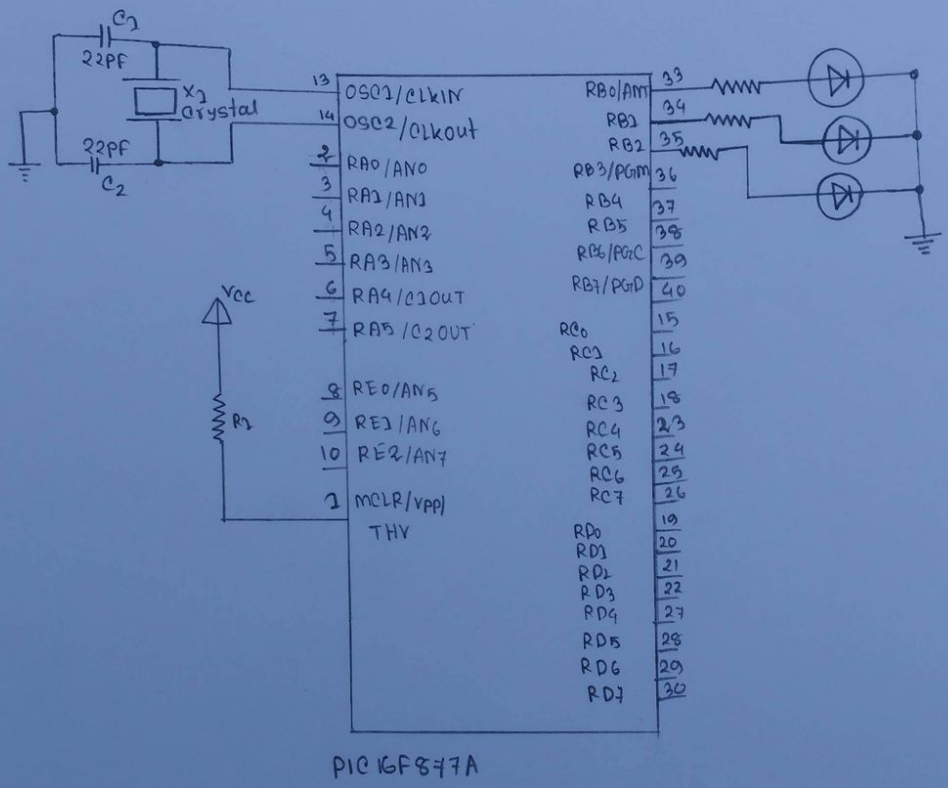


Fig: a structure of LED

Apparatus Required: LED, PIC16F877A, Power supply, resistor, crystal oscillator, capacitor.

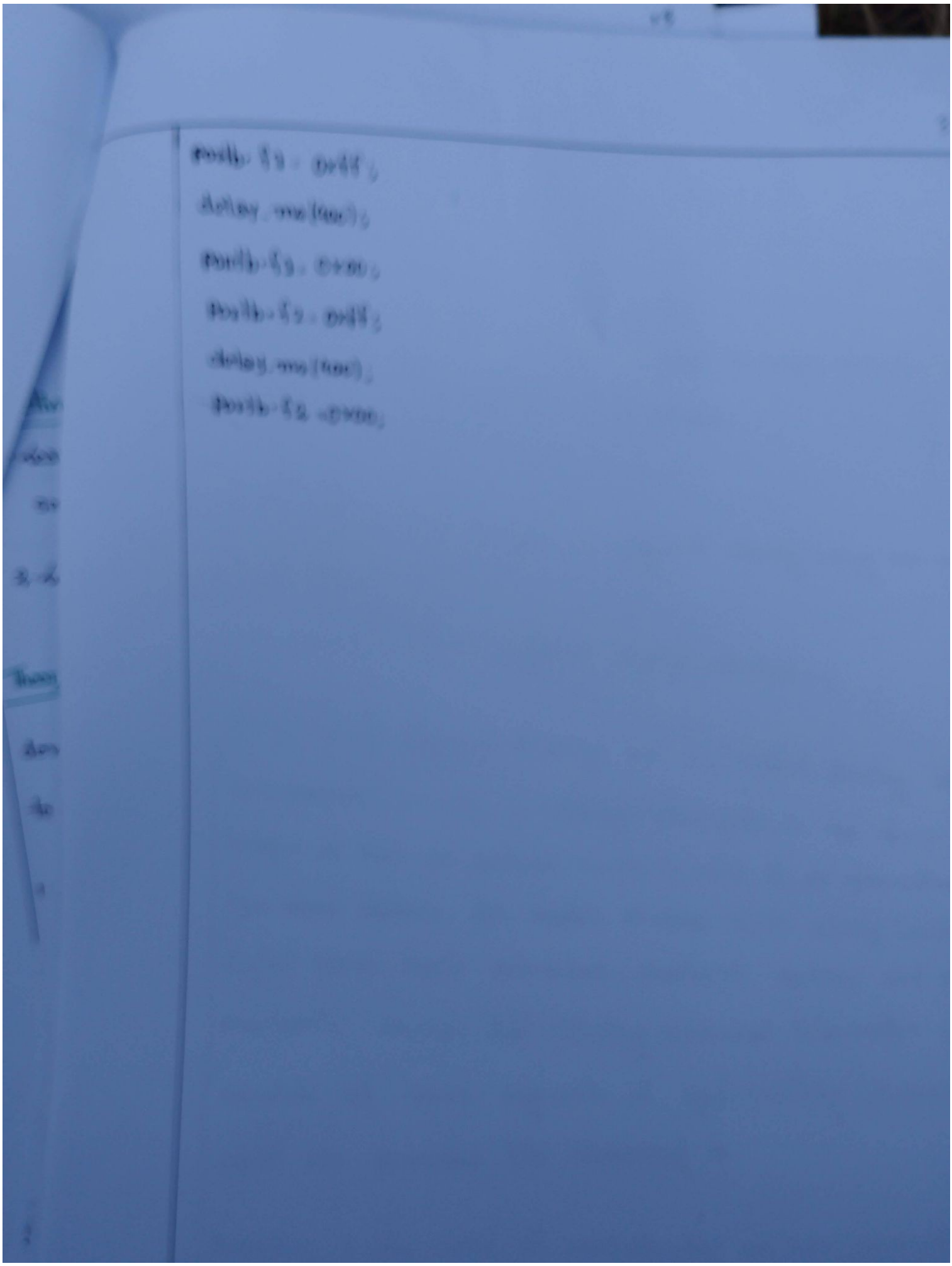
Circuit Diagram:



PIC16F877A

Program:

```
void main()
 {
   TRISB = 0x00;  //set portb as output.
   PORTB = 0x00;  //Initialize all portb as off state.
   while(1)
   {
      portb.fo = 0xff;
      delay_ms(400);
      portb.fo = 0x00;
```



Experiment no: 02

Name of the experiment: Write a program to count 0 to 9 in 7 segment display using PIC microcontroller.

Objectives:

1. Learning how to design a 7 segment display using PIC microcontroller.
2. Understanding 7 segment display's principle.

Theory: Seven segment displays are the output display device that provides a way to display information in the form of images or text or decimal numbers which is an alternative to the more complex dot matrix displays. It is widely used in digital clocks, basic calculators, electronic meters and other electronic devices that display numerical information. It consists of seven segments of light-emitting diodes (LEDs) which are assembled like numerical 8.

According to the type of application, there are two types of configurations of seven-segment display.

i) Common anode display

ii) Common cathode display

1) In common cathode seven segment displays, all the cathode connections of LED segments are connected together to logic 0 or ground. we use logic 1 through a current limiting resistor to forward bias the individual anode terminals a to g.

2) whereas all the anode connections of the LED segments are connected together to logic 1 in a common anode seven segment display. we use logic 0 through a current limiting resistor to the cathode of a particular segment a to g.

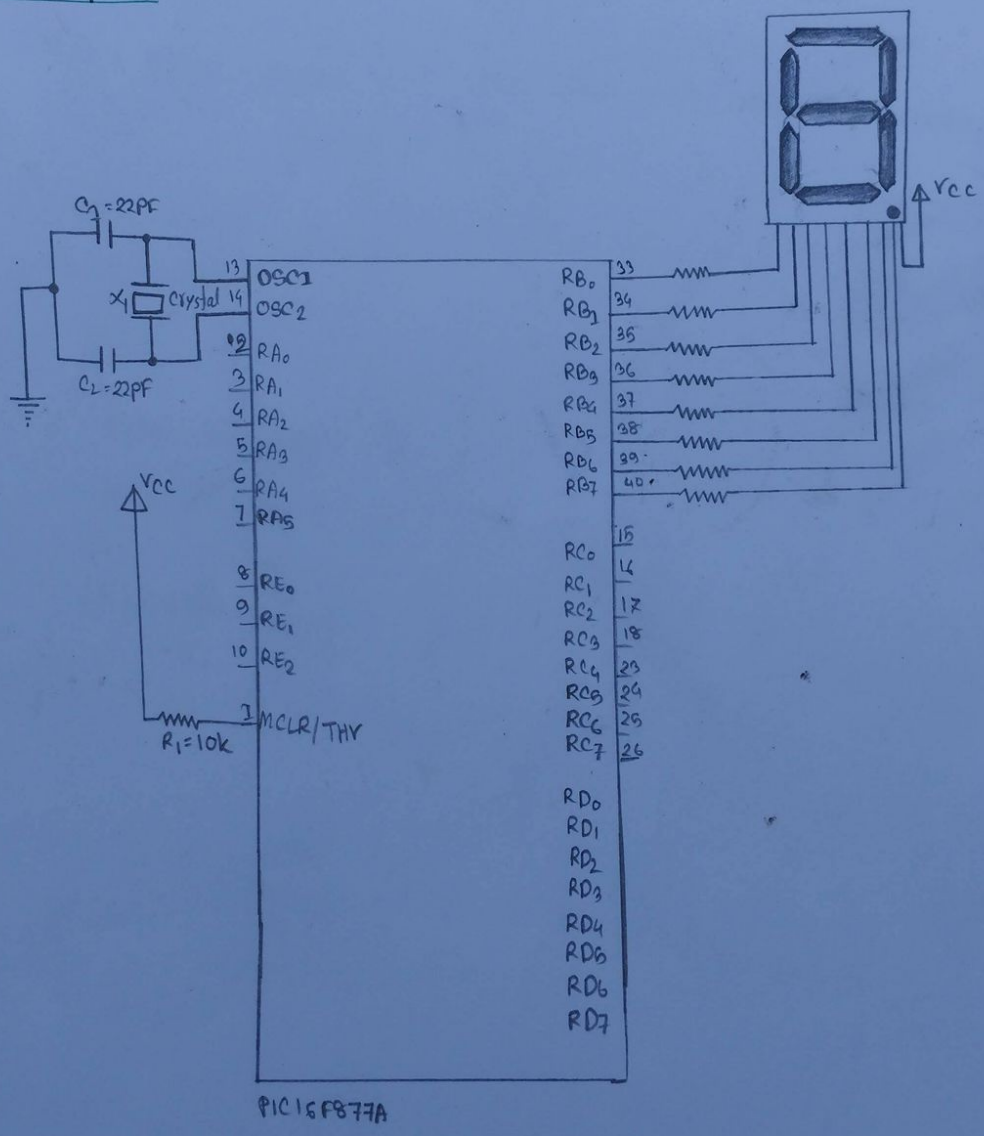
Common anode seven segment displays are more popular than cathode seven segment displays because logic ~~ser~~ circuits can sink more current than they can source and it is the same as connecting LEDs in reverse.

Seven segment displays are used in,

Digital clocks, clock radios, calculators, wristwatches, speedometers etc.

Apparatus required: PIC16F877A, ~~cap, res~~ capacitor, resistor, crystal, common anode 7 segment.

circuit diagram:



Code:

```
void main()
  {
    int i=0;
    char arra[] = {0x40, 0x79, 0x24, 0x30, 0x19, 0x12, 0x02, 0x78,
                   0x00, 0x10};
    TRISB = 0x00;
    portb = 0xff;
    while(1)
    {
      portb = arra[i];
      delay_ms(500);
      i++;
      if (i==10)
        i=0;
    }
}
```

Experiment no: 03

Name of the experiment: Analog signal input in the microcontroller or display ADC value in the virtual terminal.

Objectives:

1) To design a circuit that display ADC value in the virtual terminal.
2) To learn about the display of ADC value in the virtual terminal.

Theory: The role of ADC converter is to convert analog voltage values to digital values. The ADT converter converts analog voltage to binary numbers. These binary numbers can be in different length - 2, 4, 8, 10 bit. The more bits the binary number has, the higher the resolution of the A/D.

With two bits, we can only display 4 different options:

00	01	10	11

We can show the changes from 0 to 5 volt with 4 levels.

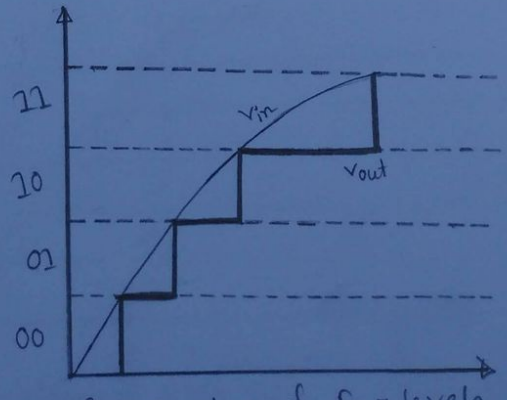


fig: Display of four levels

we can see from fig:a Vout is not close enough to the original analog input voltage values. Thus, we can say that A/D with the binary number of two-bits has a low resolution and there is a large gap between the real value of the analog input voltage and the values represented by the A/D.

Now, let us consider that the voltage that supplied to the A/D converter is still varies from 0 to 5 volt, however, the A/D converter converts the input voltage to a binary number of three bits.

with three bits we can get 8 different options:

000	001	010	011	100	101	110	111

we can now see the eight levels in the following illustration.

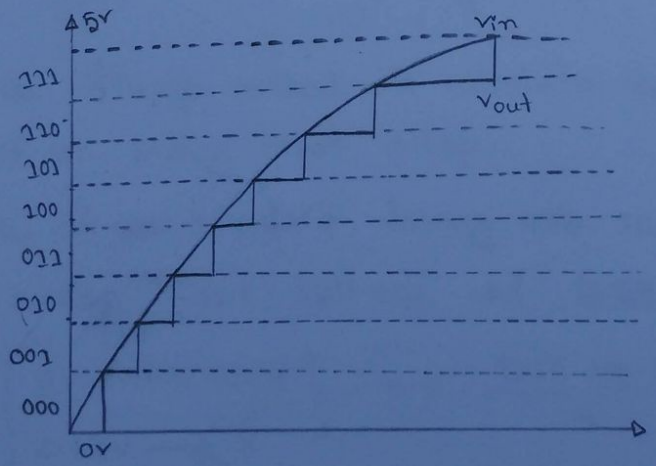


fig:b - Display of eight levels

From fig-b, we can see that the gap between the analog signal and the digital signal is smaller compared to the previous graph.

Based on the "good" results that we received, we can say that current A/D converter has a high-resolution compare to previous case.

Therefore we can say that the A/D of the microcontroller with a larger amount of bits has a higher resolution A/D takes less time than the conversion time of the high resolution A/D.

The ADCON module located within the PIC microcontroller has a resolution of ten-bit length. Therefore, the converter can devide the analog input voltage between 0v to 5v to 2^10 levels, which are 1024 levels. we can say that the resolution of this component is very high.

we can use the triangle method to calculate the binary representation of an analog input voltage.

For example, let us calculate binary value representation on the analog input voltage of 3.65 volt:

$$\left\{\begin{array}{l} 5V \longrightarrow 1024 \\ 3.65V \longrightarrow X \end{array}\right\} \rightarrow X = \frac{1024 * 3.65}{5} = 747.52 \approx 748$$

The analog input voltage of 3.65v will be represented by decimal number 748 or by binary number 1011101100. Using similar way we can find a binary representation for any desired level of the analog input voltage.

Circuit Diagram:

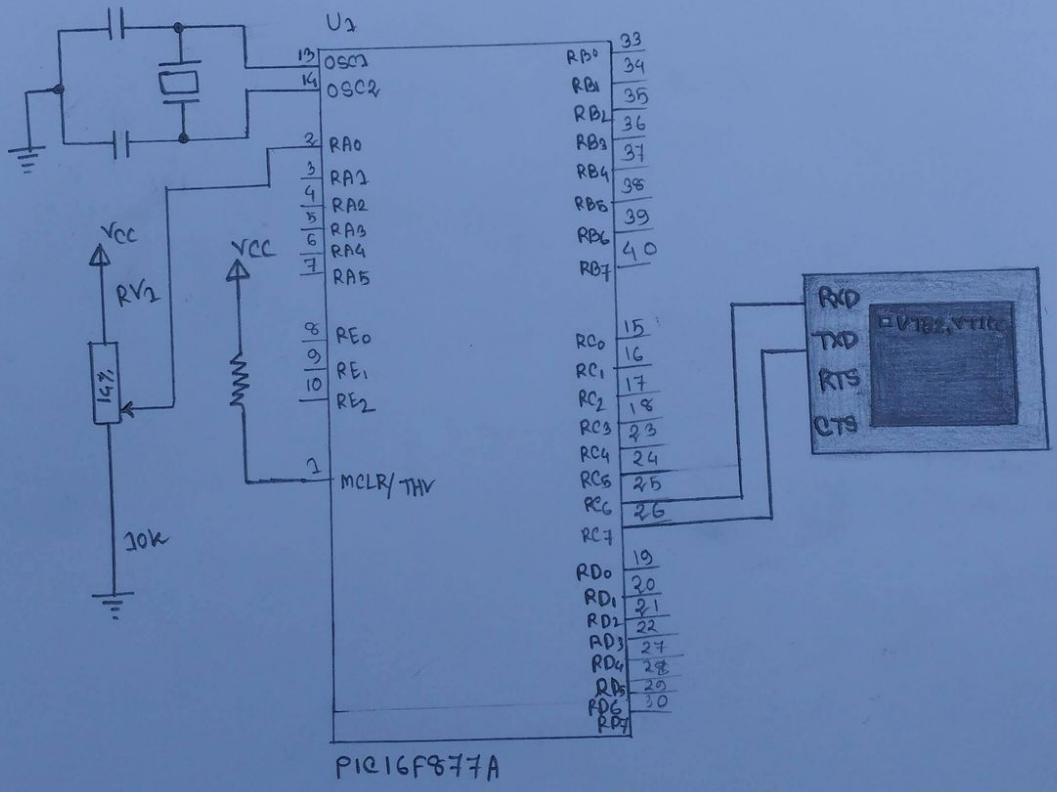


figure: Displaying ADC value in Virtual Terminal

program:

```
int valADC; //create a variable that will hold the ADC value.
char x[4]; //create a char array
void main()
{
```

```
//initialize UART
UART1_Init(9600);
//initialize ADC
ADC_Init();
while(1)   //create a loop
  {
    valADC = ADC_Read(0); // Read ADC value in RA0
    IntToStr(valADC, x); // convert into string/ char array
    UART1_write_Text("Analog value = "); //print
    UART1_write_Text(x);
    strcpy(x, "    "); //clear char array
    UART1_write(13);
    Delay_ms(1000);
  }
}
```

Experiment no: 04

Name of the experiment: write a program to display 2 digit number using 7 segment multiplexing technique.

Objectives:

1. Learning how to design a display of 2 digit number using 7 segment multiplexing technique.
2. Learning and understanding about multiplexing.

Theory: A seven-segment display is a form of electronic display device for displaying decimal numerals that is an alternative to the more complex dot matrix display.

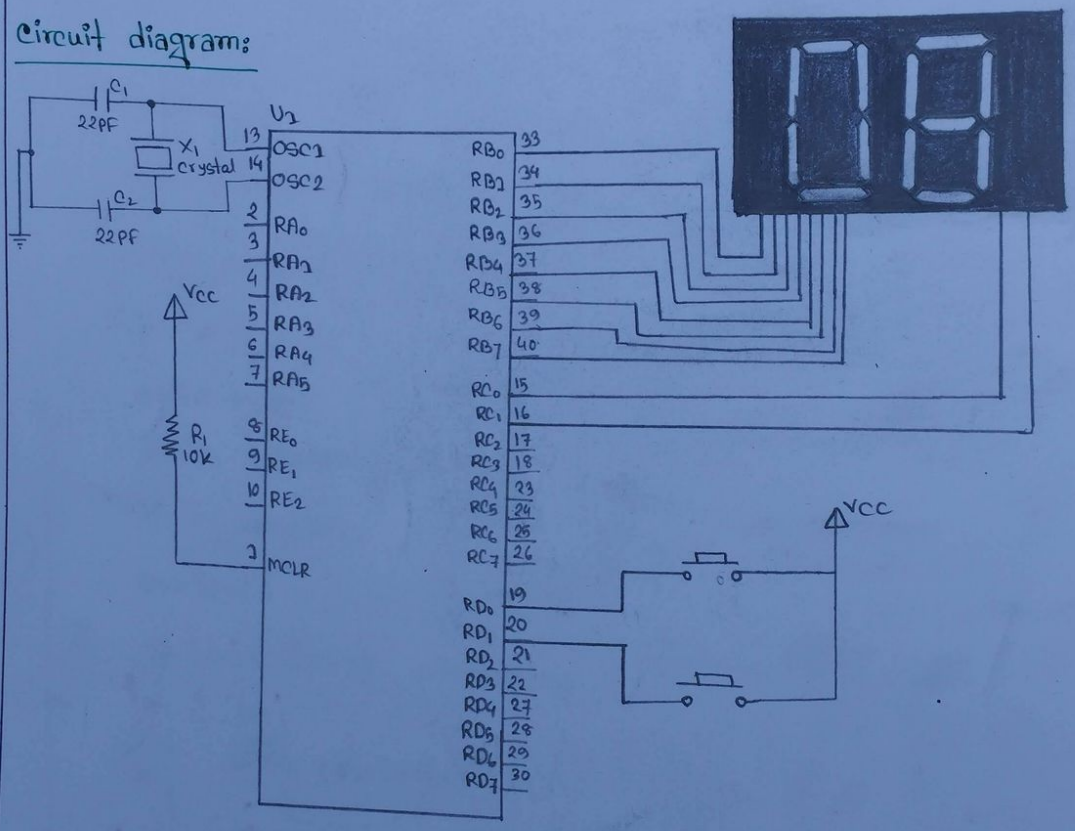
Multiplexing technique is used for driving multiple seven segment displays. The theory of 7 segment display is simple. All the similar segments of multiple LED displays are connected together and driven through a single IO pin.

Multiplexing is necessary to interface two or more seven segment displays to a microcontroller. Sofware program can control these multiplexed seven segments to ON/OFF in a cyclical fashion.

This also helps to reduce power in battery operated systems.

Apparatus Required: pic16f877A, capacitor, resistor, crystal, button, ~~common~~ 2 digit 7 segment.

Circuit diagram:



Mikro C code:

```
char segment[] = { 0x3F, 0x06, 0x5B, 0x4F
int i=0;
void main()
  {
    trisb = 0x00;
    trisc = 0x00;
```

```
void main()
 {
   TRISC = 0x00;
   TRISD = 0x00;
   while(1)
    {
      PORTD = 0x80;
      PORTC = 0x00;
      MSDelay(10);
      PORTD = 0x40;
      PORTC = 0xff;
      MSDelay(10);
      PORTD = 0x20;
      PORTC = 0xff;
      msDelay(10);
      PORTD = 0x10;
      PORTC = 0x18;
      msDelay(10);
      PORTD = 0x08;
      PORTC = 0x18;
      msDelay(10);
      PORTD = 0x04;
      PORTC = 0xff;
      msDelay(10);
      PORTD = 0x02;
      PORTC = 0xff;
      msDelay(10);
      PORTD = 0x01;
      PORTC = 0x00;
      msDelay(10);
 } }
```

...ng an analog output ...ional to the temperature.

- It provides output voltage in centigrade(celcius). It doesnot

```
portb= 0x00;
porte = 0x00;
trisd =0xff;
portd=0;

while(1)
 {
    porte.f0 = 0;
    portb = segment[i/10];
    delay_ms(10);
    porte.f0 = 1;
    porte.f1 =0;
    portb = segment [i %, 10];
    delay_ms(10);
    porte.f1=1;
    if (portd.f0==1)   //increasing
      { i++;
        while (portd.f0==1);
      }

    if (portd.f1==1)
     {
       i--;
      while(portd.f1==1);
     }
   if (i<0 || i>99)
     i=0;
} }
```

Experiment no: 05

Name of the experiment: write a program to interface LED display with push button using pic microcontroller.

Theory:

Objectives:

1) To learn push button interfacing with pic microcontroller.
2) To learn how to control a led using a push button.
3) To design and understand the circuit diagram.

Theory:

The push-button is a basic input device in the embedded system. It is used to control the operation of any output device using the microcontroller or control unit. It basically breaks the electrical circuit and interrupts the flow of current.

The push-button is basic mechanical on-off buttons that act as control devices. It short circuits the line when it is pressed and opens when it is not pressed.

In circuit pull-up and pull-down resistor use to convert ~~init~~ infinite or zero resistance into the digital signal. On the basis of the pull-up and pull-down resistor, we can interface the switch in two way. The value of pull-up and pull-down resistor depends on the microcontroller.

Positive Logic: In this connection, we use a pull-down resistor connected to the ground. when we pressed the switch then logic asserts high and when we disconnect the switch logic assert low.

Negative Logic: In this connection, we use a pull-up resistor connected to Vcc. When we pressed the switch then logic asserts low and when we disconnect the switch logic assert high.

Apparatus Required: PIC16F877A, crystal, capacitor, resistor, LED, push button.

Circuit Diagram:

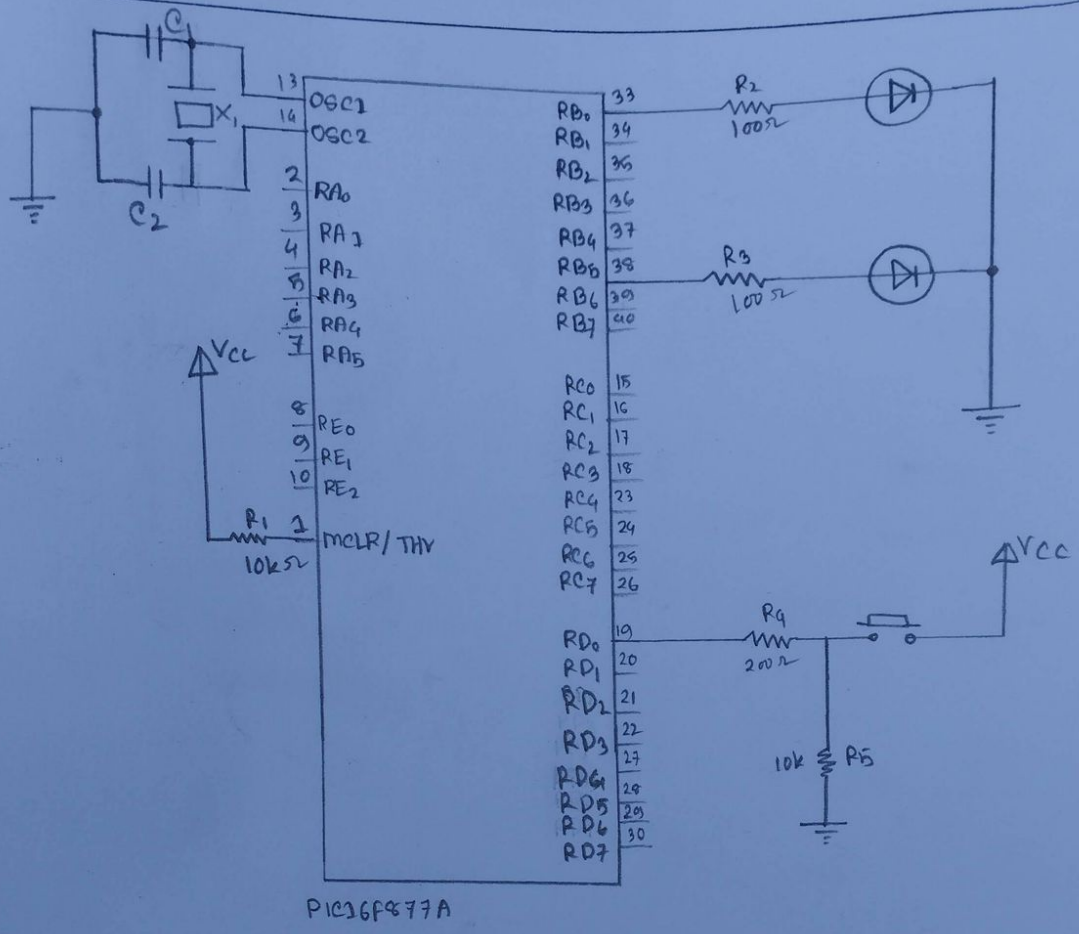


Fig: Button interfacing using pic microcontroller.

Mikro C program:

```
void main()
  {
   int count=0;
   TRISD=0xff;
   TRISB=0x00;
   PortB=0;
   while(1)
    {
       if(PortD.f0==1)
         {
           delay_ms(200);
           if(count==0)
```

```
    else
      count = 0;
  }
  if(count == 0)
    {
      PortB.f0 = 0;
      PortB.f5 = 0;
      delay_ms(200);
    }
  else
    {
      PortB.f0 = 1;
      PortB.f5 = 1;
      delay_ms(200);
    }
 }
}
```

Experiment no: 06

Name of the experiment: Dot matrix display interfacing with PIC16F877A microcontroller.

Objective:

Theory: A Dot-matrix display is a display device which contains light emitting diodes aligned in the form of matrix. Dot matrix displays are used in applications where symbol, graphic, characteristics, alphabets numbers are needed to be displayed in static as well as scrolling motion.

A typical dot matrix display is shown below:

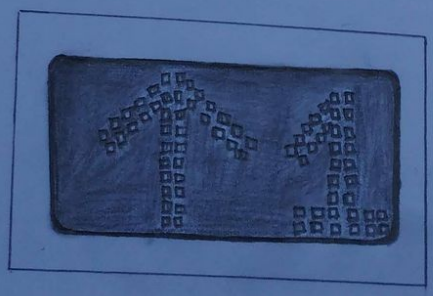


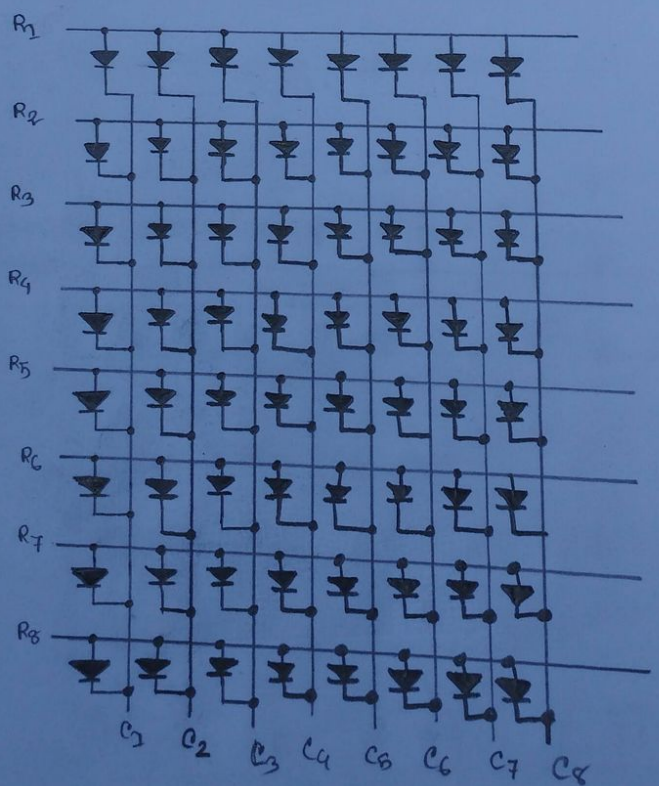
fig-a: A typical dot matrix display is using in lift

the temperature.

• It provides output voltage in centigrade(celsius)

Types of dot matrix: Dot matrix display is manufactured in various dimensions like 5×7, 8×8, 16×8, 32×8, 64×64 and 128×64 where the numbers represent LED's in rows and columns.

Construction of Matrix display: In Dot matrix display, multiple LED's are wired together in rows and columns. The matrix pattern is made either in row anode and column cathode or row cathode and column anode pattern. In row anode column cathode pattern, the entire row is anode while all columns serve as cathode.



Fig+b: Construction of Dot matrix display

the temperature.

• It provides output

In Dot Matrix Display, each LED can be control individually by controlling the current through each pair of column and row.

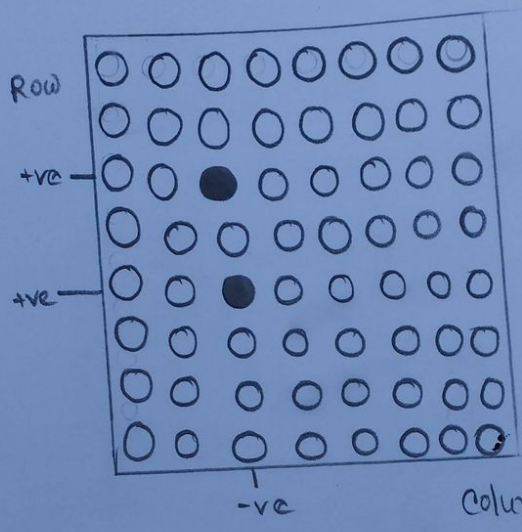


fig-c: working mechanism of dot matrix display

How to generate code for Dot Matrix Display :

	1	2	3	4	5	6	7	8	Bin	HEX
1	0	1	1	0	0	1	1	0	0b00000000	0x00
2	0	1	1	0	0	1	1	0	0b11111111	0xff
3	0	1	1	0	0	1	1	0	0b11111111	0xff
4	0	1	1	1	1	1	1	0	0b00011000	0x18
5	0	1	1	1	1	1	1	0	0b00011000	0x18
6	0	1	1	0	0	1	1	0	0b11111111	0xff
7	0	1	1	0	0	1	1	0	0b1111 1111	0xff
8	0	1	1	0	0	1	1	0	0b0000 0000	0x00

...rtional to the temperature.

- It provides output voltage in ...

Circuit Diagram:

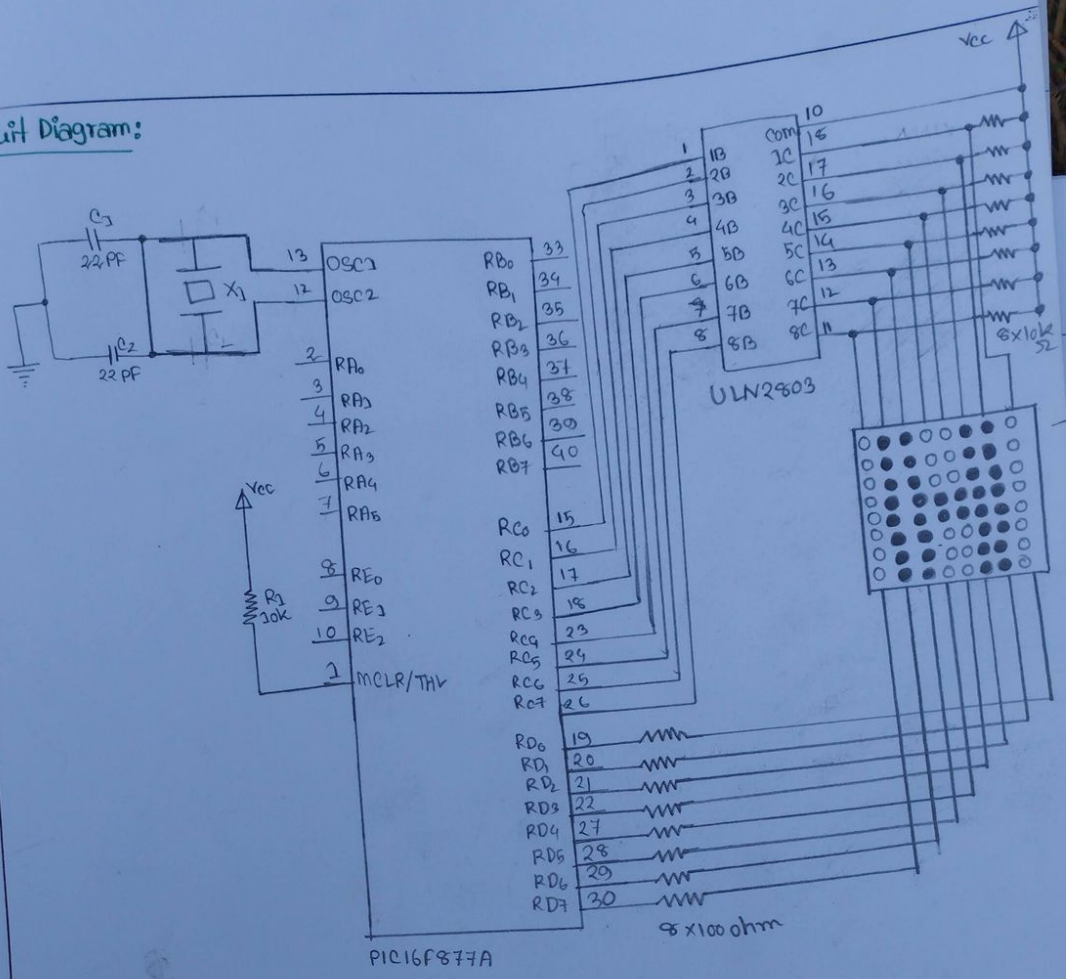


figure: Proteus simulation of dot matrix LED.

Appartus required: pic16f877A, crystal, capacitor, resistor, 8×8 dot matrix display, Darlington transistor array.

Mikro C code:

```
void MsDelay(unsigned char Time)
  {
     unsigned char y,z;
     for(y=0; y<Time; y++)
        { for(z=0; z<20; z++);
  }
```

... to the temperature.

• It provides output ...

Experiment no: 07

Experiment name: LM35 Temperature Sensor Data Read and Display Using LCD Module.

Objectives:

1) To learn about using LCD module.
2) To learn and understand the circuit diagram of LCD module.
3) To learn interfacing LCD with PIC Microcontroller.

Theory:

To interface LCD with a pic microcontroller, we use general-purpose input-output pins (GPIO pins). Because we send control and data signals to LCD through these I/O pins. ~~Therefore, w~~

16X2 LCD Pinout:

It consists of 14 pins. There are 8 data pins from $D_0 - D_7$ and three control pins such as RS, RW and E. LED+ and LED- pins are used to control the backlight LED.

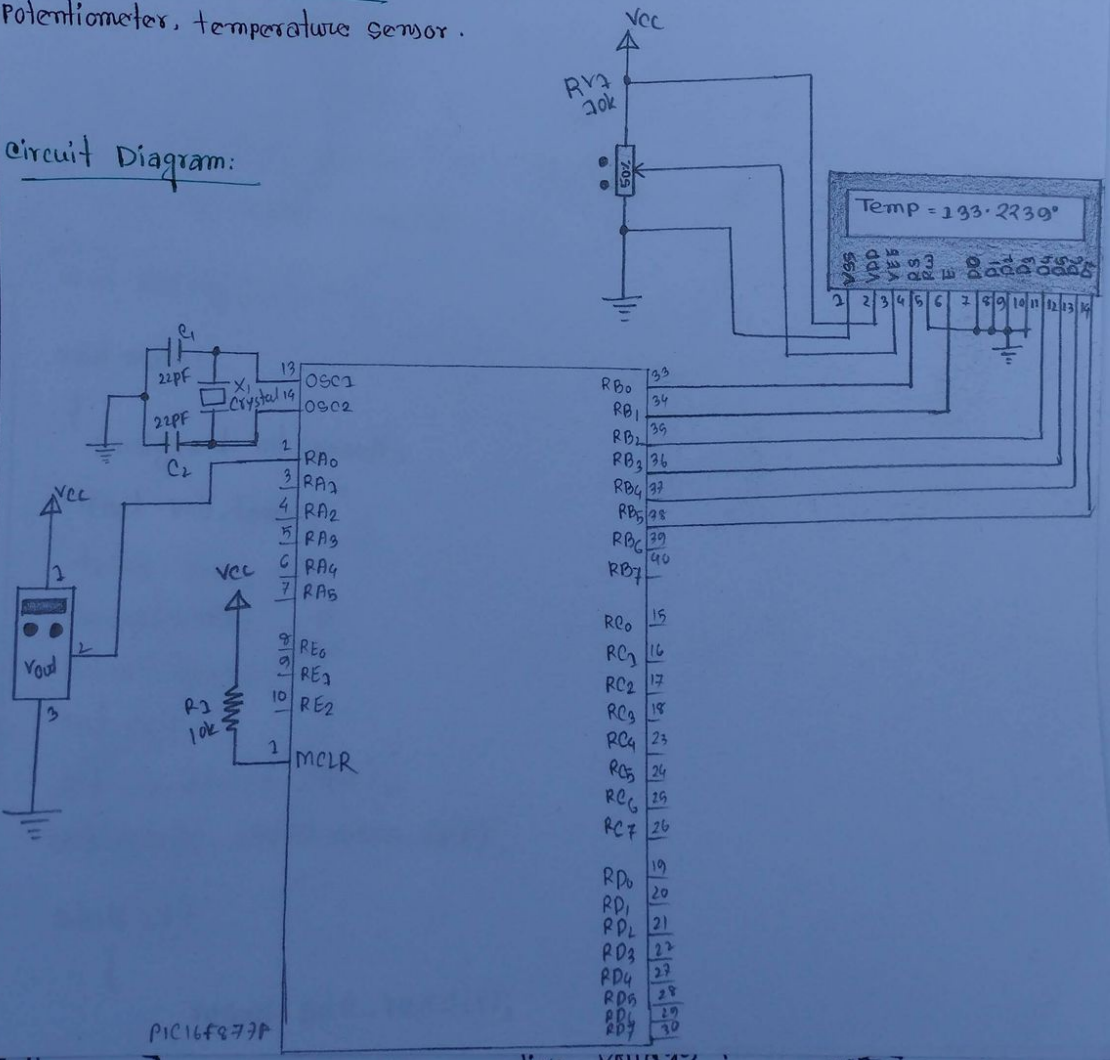
LM35 Temperature sensor:

- LM35 is a temperature measuring device having an analog output voltage proportional to the temperature.
- It provides output voltage in centigrade(celcius). It doesnot require any external calibration circuitry.
- A ... by the negative

- The sensitivity of LM35 is 10 mv/degree celcius. As temperature increases, output voltage also increases.
- It is a 3-terminal sensor used to measure surrounding temperature ranging from -55°C to 150°C.
- LM35 gives temperature output which is more precise than thermistor output.

~~Aim~~ Apparatus Required: PIC16F877A, capacitor, resistor, crystal, LCD display, Potentiometer, temperature sensor.

Circuit Diagram:



may stop working by the negative voltage

Mikroc Code:

```
//LCD module connections
sbit LCD_RS at RB0_bit;
sbit LCD_EN at RB1_bit;
sbit LCD_D4 at RB2_bit;
sbit LCD_D5 at RB3_bit;
sbit LCD_D6 at RB4_bit;
sbit LCD_D7 at RB5_bit;

sbit LCD_RS_Direction at TRISB0_bit;
sbit LCD_EN_Direction at TRISB1_bit;
sbit LCD_D4_Direction at TRISB2_bit;
sbit LCD_D5_Direction at TRISB3_bit;
sbit LCD_D6_Direction at TRISB4_bit;
sbit LCD_D7_Direction at TRISB5_bit;
//End LCD module connections

char display[16]="";

void main()
{
  unsigned int result;
  float volt,temp;
  trisb=0x00;
  trisa=0xff;
  adcon1=0x80;
  lcd_init();
  lcd_cmd(_lcd_clear);
  lcd_cmd(_LCD_CURSOR_OFF);

  while(1)
    {
        result=adc_read(0);
```

may stop working by the negative voltage produced

```
volt = result * 4.88;
temp = volt/10;
lcd_out(1,1,"Temp = ");
floattostr(temp, display);
lcd_out_cp(display);
lcd_chr(1,16,223); // print at pos(row=1, col=13) "o" = 223 = 0x df
lcd_out_cp("C"); // celcius
}
}
```

Experiment no: 08

Name of the experiment: write a program to control a high voltage load using mechanical relay.

Objectives:

1) To learn how to interface ac load through relay with pic microcontroller.

2) To design and operate the circuit of relay interfacing.

Theory:

A relay is an electromagnetic switch which is used to switch high voltage / current using low power circuits. Relay isolates low power circuits from high power circuits. It is activated by energizing a coil wounded on a soft iron core. A relay should not be directed directly connected to a micro controller. Because-

- A microcontroller is not able to supply current required for the working of relay. Maximum current that a pic Microcontroller hande is 25 mA while a relay needs about 50-100 mA current.

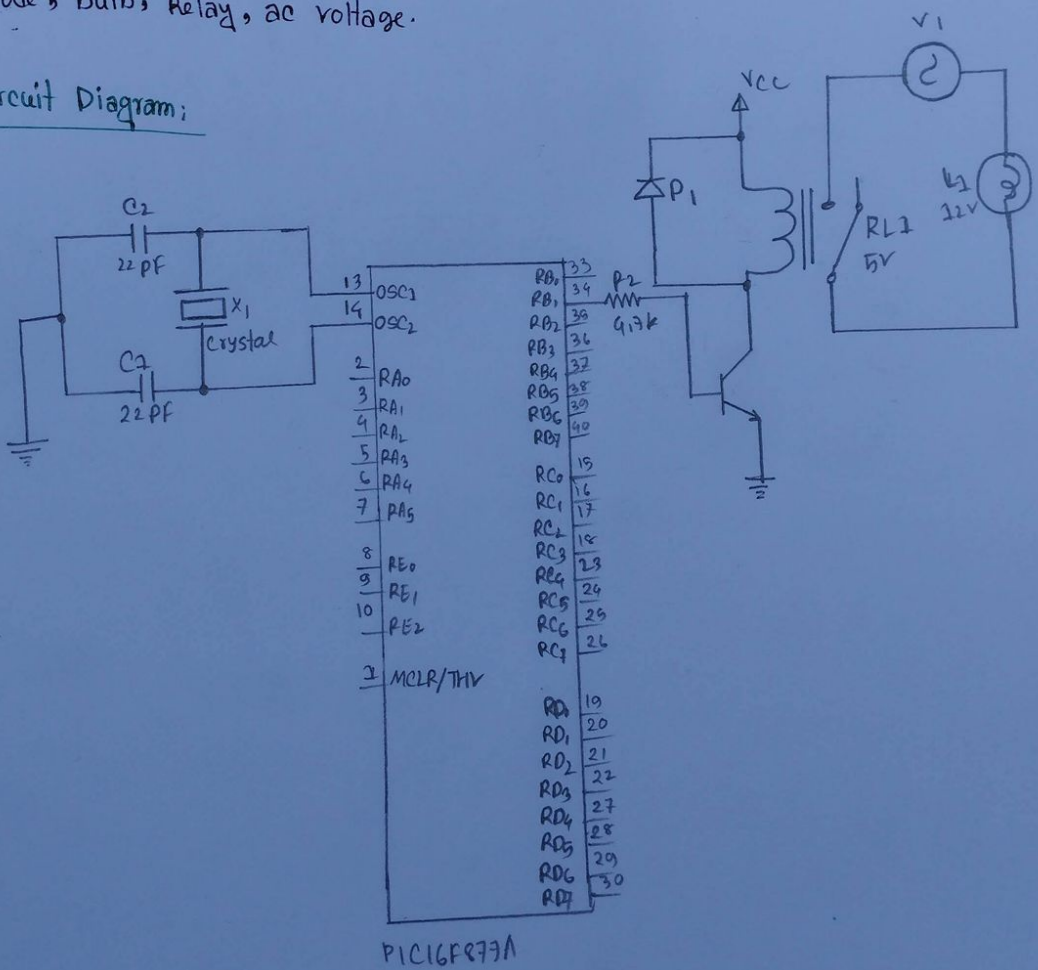
- A relay is activated by energizing its coil. Microcontroller may stop working by the negative voltages produced in the relay [illegible] stays low is called "off time".

due to its back emf.

A relay can be easily interfaced with microcontroller using a transistor, which carries the current required for operation of the relay.

Apparatus Required: PIC16F877A, crystal, capacitor, resistor, transistor (BC547BP), diode, bulb, Relay, ac voltage.

Circuit Diagram:



time"

Mikro C Code:

```
void main()
  {
     TRISB = 0x00;
     PortB = 0x00;
     while(1)
     {
        PortB.f1 = 1;
        delay_ms(2000);
        PortB.f1 = 0;
        delay_ms(2000);
      }
}
```

... the signal ... the

Experiment no: 09

Name of the experiment: DC motor speed control using PWM and microcontroller.

Objectives:

1. To learn how to control the average power delivered to a load and using PWM and how to control the speed of the DC motor.
2. To understand the circuit diagram.

Theory:

PWM stands for Pulse width Modulation. A modulation technique that generates variable width pulses to represent the amplitude of an analog input signal.

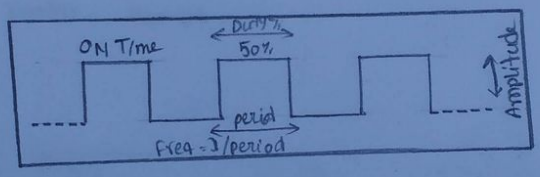
PWM is a type of signal which can be produced from a digital IC such as microcontroller or 555 timer. The signal thus produced will have a train of pulses and these pulses will be in form of a square wave. That is, at any given instance of time the wave will either be high or will be low. The duration at which the signal stays high is called "on time" and the duration at which the signal stays low is called "off time".

like inverters or converters. To control a servo motor we need

Duty cycle of the PWM: A PWM signal stays on for a particular time and then stays off for the rest of the period. The percentage of time in which the PWM signal remains High is called as duty cycle. If the signal is always ON it is in 100% duty cycle and it it is always off then 0% duty cycle.

Formula:

$$\text{Duty cycle} = \frac{\text{Turn ON time}}{\text{Turn ON time} + \text{Turn OFF time}}$$



By controlling the duty cycle from 0% to 100% we can control the "on time" of PWM signal and thus the width of signal.

Frequency of PWM: The frequency of PWM signal determines how fast a PWM completes one period.

Frequency = 1 / Time period

Time period = ON time + off time

Normally the PWM signals generated by microcontroller will be around 500 Hz which is used in high speed switching devices like inverters or converters. To control a servo motor we need

to produce pwm signals with 50Hz frequency.

Features of L293D:

1) width supply-voltage range 4.5V to 36V and separate input-logic supply
2) Internal ESD protection and High-Noise Immunity inputs.
3) Output current 1A per channel (600mA for L293D)
4) Peak output current 2A per channel (1.2A for L293D)
5) Output clamp Diodes for Inductive Transient Suppression (L293D)

Apparatus required: pic microcontroller, crystal, resistor, capacitor, L293D IC, DC motor, switch.

Circuit Diagram:

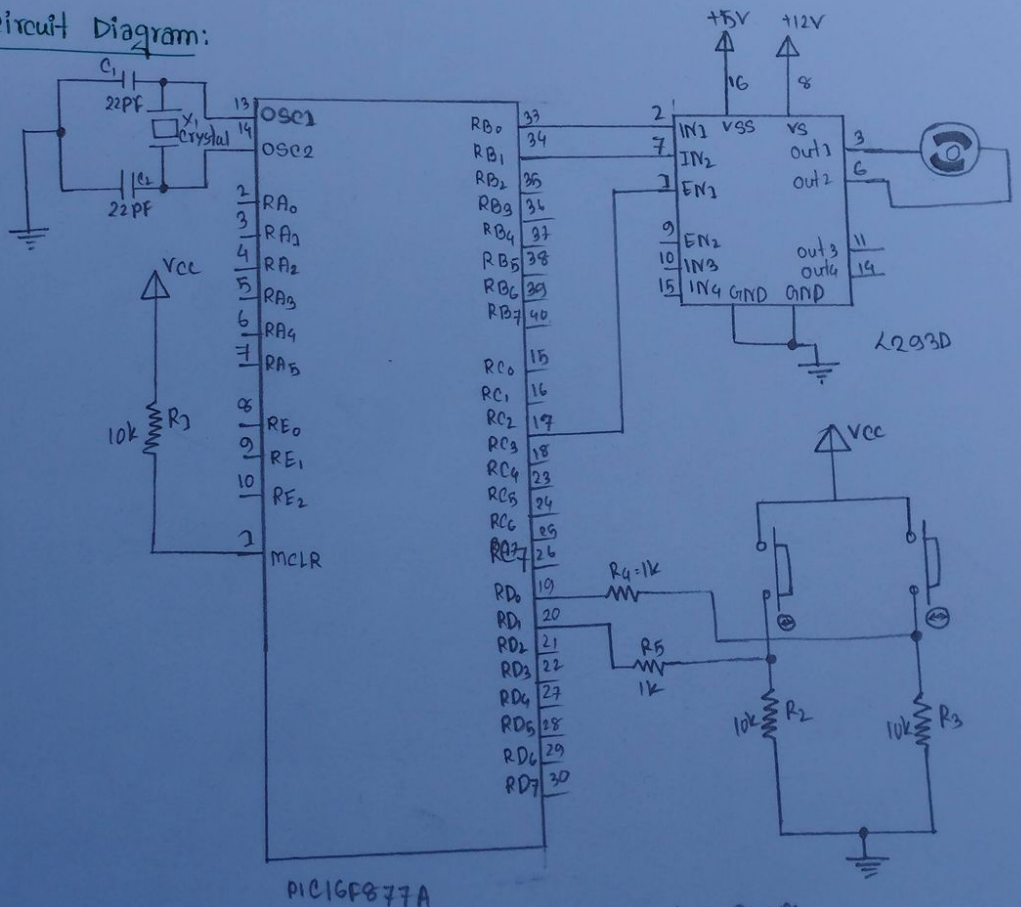


figure: DC motor speed control using pwm

Experiment no: 10

Name of the experiment: Write a program to control servo motor using pic microcontroller.

Objectives:

1. To design and understand the circuit diagram of servo motor.
2. To learn how servo motor works and learn the interfacing of a servo motor with PIC16F877A microcontroller.
3. Programming to control servo motor and hardware connections of servo motor with PIC16F877A microcontroller.

Theory:

A servo motor is a special kind of motor that operations upon the instructions. It provides angular precision, which means unlike other electrical motors that keep on rotating until power is applied to them and stops when the power is switched off, the servo motor rotates only to a certain degree. or until it is required to and then the motor stops and waits for the next instructions to carry out further action. servo motor controlled with the help of servomechanism.

Here we are using a servo whose angular rotation is limited to 0–180°. The input to its control line determines the positions demanded for the output shaft.

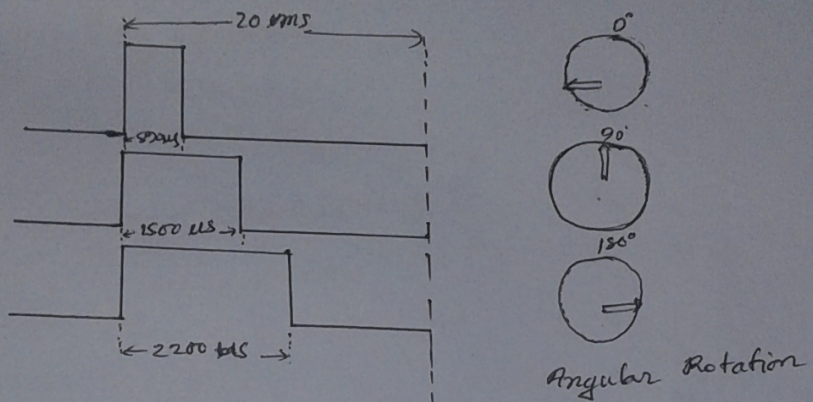
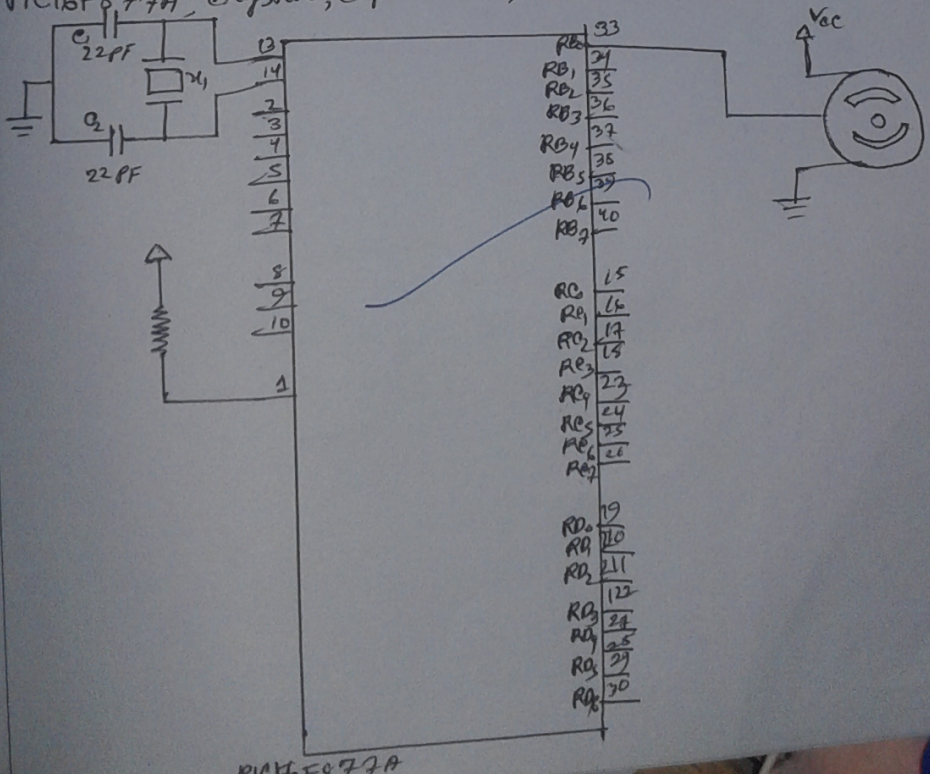


Fig: a Angular rotation of servo motor

Apparatus required :

PIC16F877A, crystal, capacitor, resistor, servo motor.



Mikro C code;

```
Void servoRotate0()
  {
    unsigned int i;
    for(i=0; i<50; i++)
      {
        PORTB.F0 = 1;
        Delay_us(600);
        PORTB.F0 = 0;
        Delay_us(19200);
      }
      Void servoRotate180()
        {
          unsigned int i;
          for(i=0; i<50; i++)
            {
              PORTB.f.i = 1;
              Delay_us(2200);
              PORTB.F0 = 0;
              Delay_us(17800);
            }
        }
Void main()
  {
    TRISB = 0;
    do
      {
        servoRotate0();
        Delay_ms(2000);
        Servo.Rotate90();
        Delay_ms(2000);
        servoRotate180();
      } while(1);
  }
```

Experiment no: 11

Name of the experiment: Write a program for interfacing stepper motor with PIC Microcontroller.

Objectives:

1. To describe and explain the operation of a stepper motor
2. To describe motor with PIC16F877A and understand its circuit diagram.

Theory: A stepper motor is a brushless, synchronous DC electric motor, which device the full rotation into a number of equal steps. It finds applications in field of microcontroller such as robotics. Unipolar motor is the most popular stepper motor among electronics hobbyist because of its ease of operation and availability.

Stepper motor can be easily interfaced with PIC microcontroller by using ready made ICs such as L293D or ULN2003.

There are three different stepping motor for unipolar stepper motor.

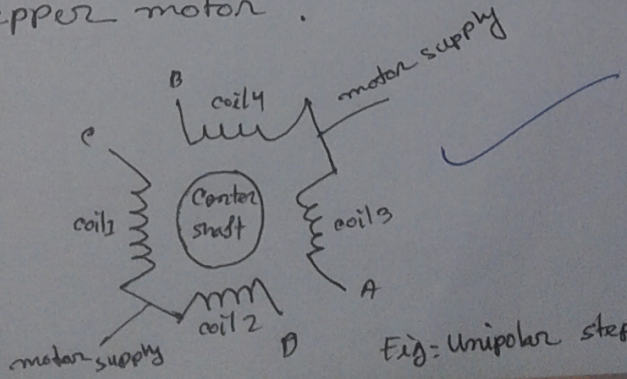


Fig: Unipolar stepper motor

Wave Drive: In this mode only one stator electromagnet is energised at a time. It has the same number of steps as the full step device but the torque is significantly less. It is namely used. It can be used where power consumption is more important than torque

Wave drive stepping sequence				
Step	A	B	C	D
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

Full Drive: In this mode two state electromagnets are energised at a time. It is the usual method used for driving and the motor will turn at its full torque in this mode of driving.

Full Drive Stepping Sequence				
Step	A	B	C	D
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1
4	1	0	0	1

Step Drive: In this stepping motor the phases arragaised. The mode is commonly used to increase angular. mechanism.

Half drive stepping sequence				
Step	A	B	C	D
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

Driving Bipolar Meta:

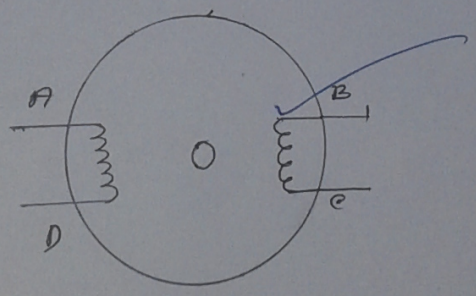
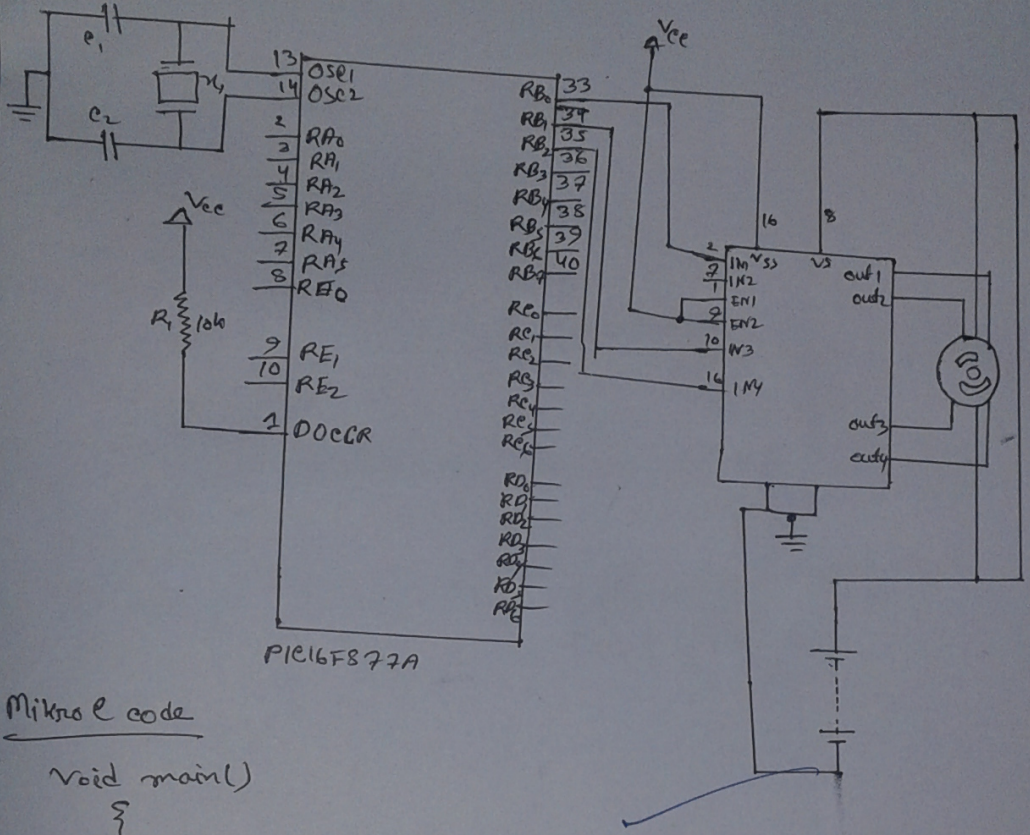


Fig: Bipolar stepper motor winding.

Apparatus required :

PIC16F877A, crystal, capacitor, registor, stepper motor, Battery, L293D.

Circuit Diagram :



Mikro C code

```
Void main()
  {
   CMCON = 0x07;
   ADCON1 = 0x06;
   TRISB = 0;
   PORTB = 0x0F;
   do
   {portB = 0b00000001;
    Delay_ms (500);
    PORTB = 0b00000011;
    Delay_ms(500);
    PORTB = 0b00000110;
    Delay_ms(500);
    PORTB = 0b00001100;
    Delay_ms(500)
```

```
PORTB = 0b00001100;
delay_ms (500);
PORTB = 0b00001000;
Delay_ms (500);
PORTB = 0b00001001;
Delay_ms (500);
} while (1);
}
```

Result and Discussion:

In this experiment we observe how to interface a stepper motor using PIC Microcontroller. We set portb as output. Here we connected a stepper motor when was start the simulator our servo stepper that first. Respectively, From this experiment we say observe that motor do this doing.