

Index

SL No	Experiment Name	Page No
1	Write a program for Blinking LED using PIC Microcontroller.	
2	Write a program to count 0 to 9 in 7 segment display using PIC Microcontroller.	
3	Write a program to display ADC value in the virtual terminal using PIC Microcontroller.	
4	Write a program to display 2 digits number using 7 segment multiplexing technique.	
5	Write a program to interface LED with push button using PIC Microcontroller.	
6	Write a program for dot matrix display interfacing with PIC Microcontroller.	
7	Write a program to LM35 temperature sensor data read and display corresponding sensor value through LCD display.	
8	Write a program to control a high voltage load using mechanical relay.	
9	Write a program for DC motor speed control using PWM and PIC Microcontroller.	
10	Write a program to control servo motor using PIC Microcontroller.	
11	Write a program for interfacing stepper motor using PIC Microcontroller.	

Circuit diagram:

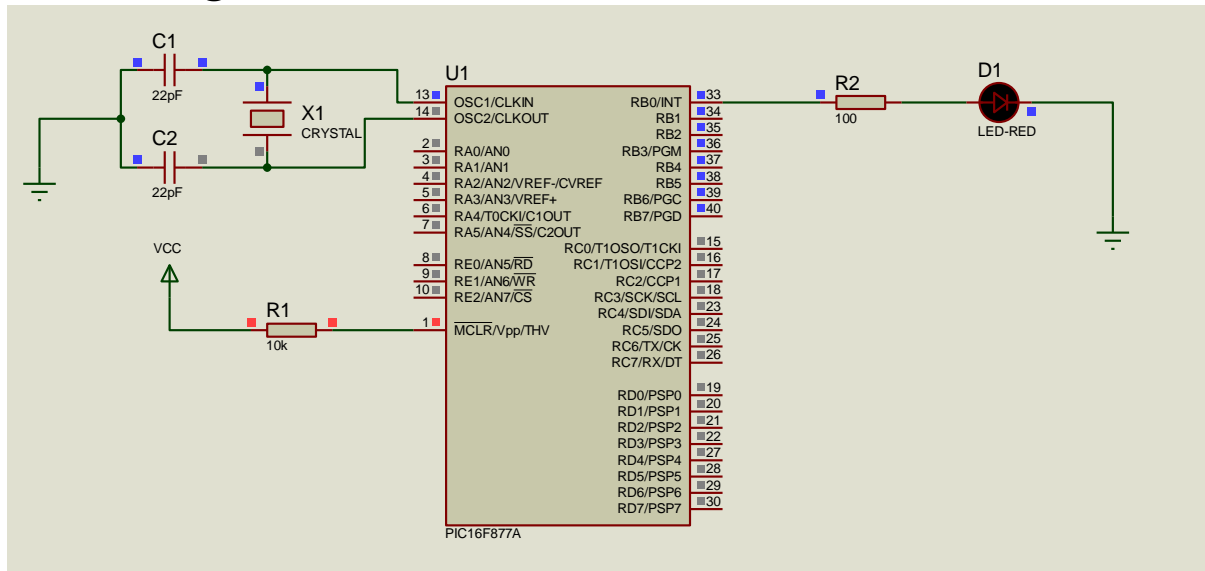


Figure 01: Circuit diagram for led blinking using for lop

Code:

```
//Using for lop
void main() {
    int i;
    Trisb=0x00;
    portb=0x00;

    for (i=0;i<50;i++)
    {
        portb.f0=0xff;
        delay_ms(2000);
        portb.f0=0;
        delay_ms(1000);
    }
}
```

Circuit diagram:

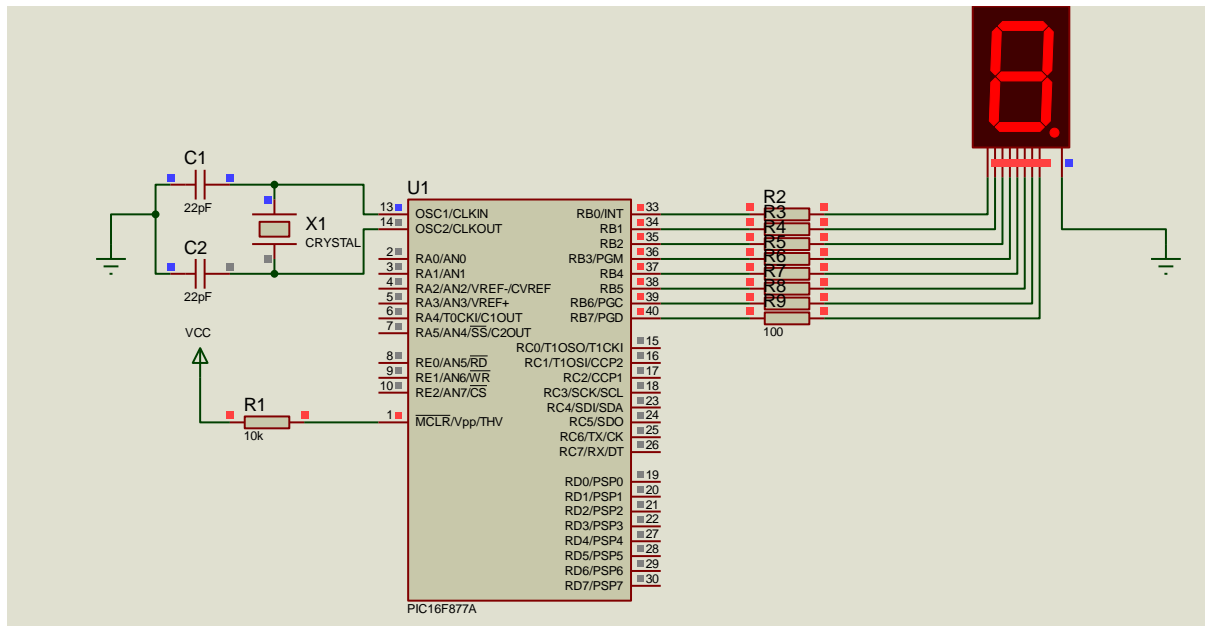


Figure 01: Circuit diagram for counting 0-9 using 7 segment common cathod display

Code:

```
//Counting 0-9
```

```
char arraycc[]={0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F};
```

```
void main()
{
    int i=0;
    trisb=0x00;
    portb=0x00;
    for(i=0;i<10;i++)
    {
        portb=arraycc[i];
        delay_ms(500);
    }
}
```

Circuit diagram:

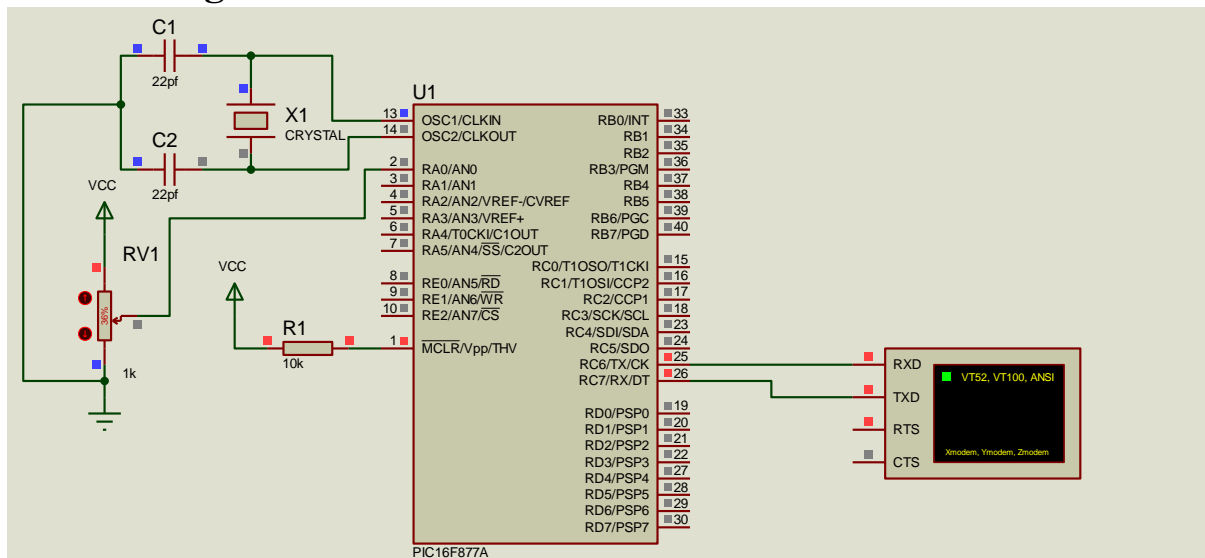


Figure 01: Circuit diagram to display Analog to Digital (ADC) value using virtual terminal

Code:

```
int valAdc;
char x[4];
void main() {
    UART1_Init(9600);
    ADC_Init();
    while(1) {
        valAdc = ADC_Read(0);
        IntToStr(valAdc, x);
        UART1_Write_Text("Analog Value= ");
        UART1_Write_Text(x);
        UART1_Write(13);
        strcpy(x, " ");
        delay_ms(1000);
    }
}
```

Circuit diagram:

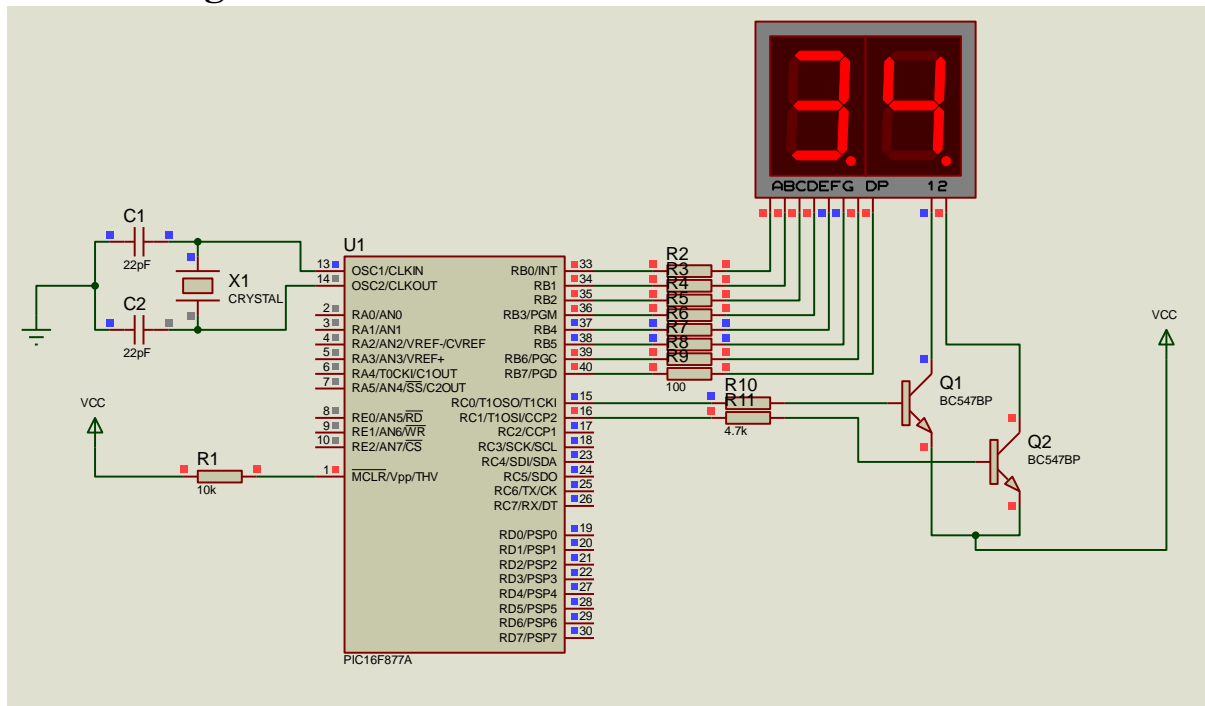


Figure 01: Circuit diagram to cout 0-99 using common cathode 2 digits 7 segment display

Code:

```
char arraCC[] = { 0xBF, 0x86, 0xDB, 0xCF, 0xE6, 0xED, 0xFD, 0x87, 0xFF, 0xEF };
```

```
void main()
```

```
{
```

```
    int mod = 0, res = 0, i = 0, k = 0;
```

```
    // Set port directions for controlling the display
```

```
    TRISB = 0x00;
```

```
    TRISC = 0x00;
```

```
    TRISD = 0x00;
```

```
    // Initialize port values
```

```
    portb = 0x00;
```

```
    portc = 0x00;
```

```
    portd = 0x00;
```

```
    while(1)
```

```
    {
```

```

// Loop through numbers from 0 to 99
for(i = 0; i <= 99; i++)
{
    res = i / 10; // Calculate tens digit
    mod = i % 10; // Calculate units digit

    // Display each digit for a brief delay
    for(k = 0; k < 10; k++)
    {
        portc.f0 = 0x00;    // Activate power for left digit
        portb = arraCC[res]; // Set segment data for tens digit
        delay_ms(10);       // Delay

        portc.f0 = 0xff;    // Deactivate power for left digit

        portc.f1 = 0x00;    // Activate power for right digit
        portb = arraCC[mod]; // Set segment data for units digit
        delay_ms(10);       // Delay

        portc.f1 = 0xff;    // Deactivate power for right digit
    }
}
}
}

```

Circuit Diagram:

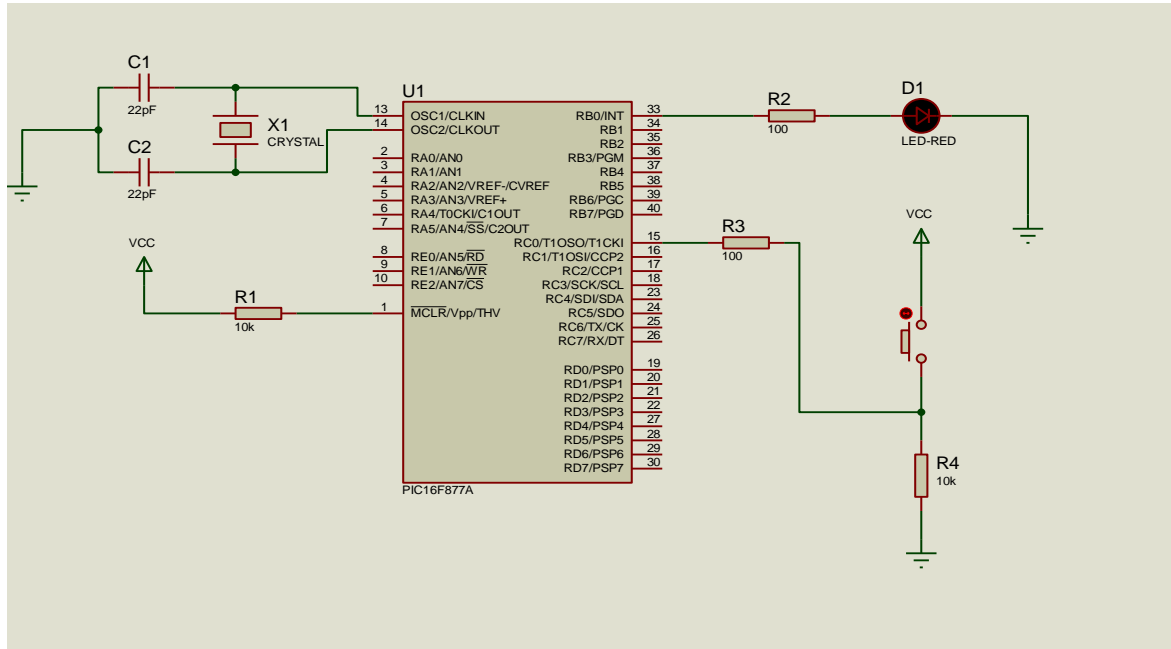


Figure 01: Circuit diagram for blinking led using push button

Code:

```
void main() {
    int i, bt_zero = 0;
    Trisb = 0x00;
    trisc = 0x01; // Set port C bit 0 as input
    portb = 0x00;
    portc = 0x00;
    while (1) {
        if (portc.f0 == 1) {
            delay_ms(150);
            if (portc.f0 == 1) {
                bt_zero++;
                if (bt_zero == 10) {
                    bt_zero = 0;
                }
            }
        }
        if (portc.f0 == 1) {
            for (i = 0; i < 50; i++) {
                portb.f0 = 1;
                delay_ms(1000);
                portb.f0 = 0;
                delay_ms(1000); // LED on for 1 second, off for 1 second
            }
        }
    }
}
```

Circuit Diagram:

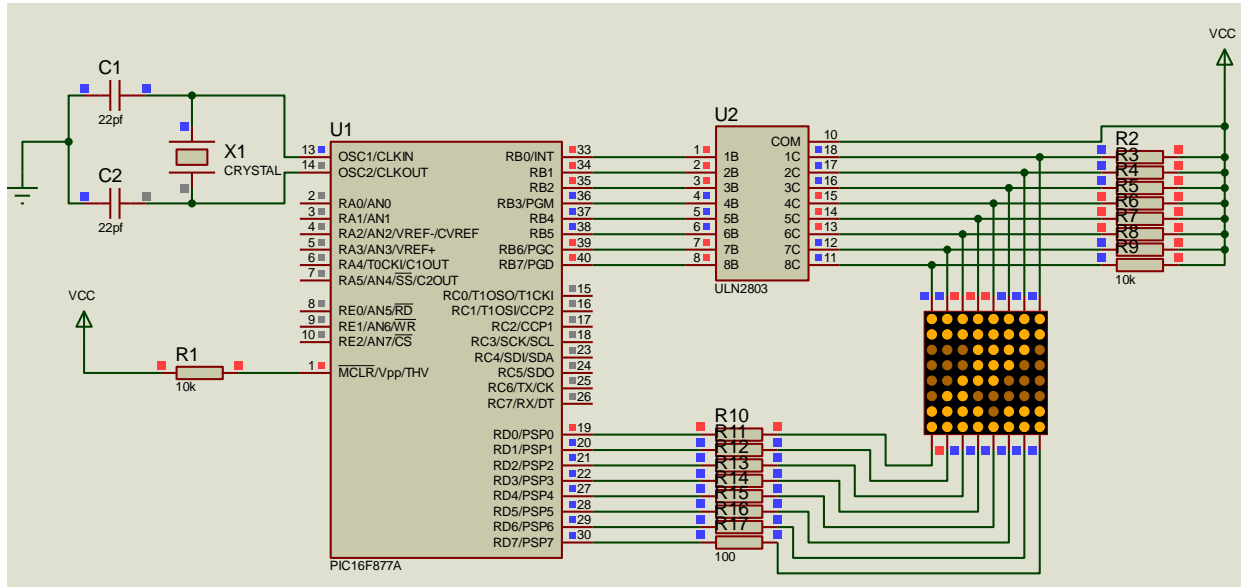


Figure 01: Circuit diagram for dot matrix display to visualize character

Code:

```

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

void main() {
    Trisb=0x00;
    Trisd=0x00;
    while(1){
        PortD=0x80;
        PortB=(0xC3);
        MSDelay(10);

        PortD=0x40;
        PortB=(0xE3);
        MSDelay(10);

        PortD=0x20;
        PortB=(0xF3);
        MSDelay(10);

        PortD=0x10;
        PortB=(0xF9);
        MSDelay(10);

        PortD=0x08;
        PortB=(0xFB);
        MSDelay(10);

        PortD=0x04;
        PortB=(0xCF);
        MSDelay(10);

        PortD=0x02;
        PortB=(0xC7);
        MSDelay(10);

        PortD=0x01;
        PortB=(0xC3);
        MSDelay(10);

    }
}
    
```

Code:

```

void main() {
    int bt=0;
    Trisb=0x00; //as output
    Trisc=0xff; // as input
    portb=0x00;
    while(1)
    {
        //button
        if(portc.f0==1)
        {
            delay_ms(150);
            if(portc.f0==1)
            {
                bt++;

                if(bt==10)
                {
                    bt=0;
                }
            }
        }
    }
    if(portc.f0==1)
    {
        portb.f0=1;//turn on relay
        delay_ms(10000); // relay on for 10
second
        portb.f0=0;//turn off relay
        delay_ms(10000); // off for 10 second
    }
}

```

Circuit Diagram:

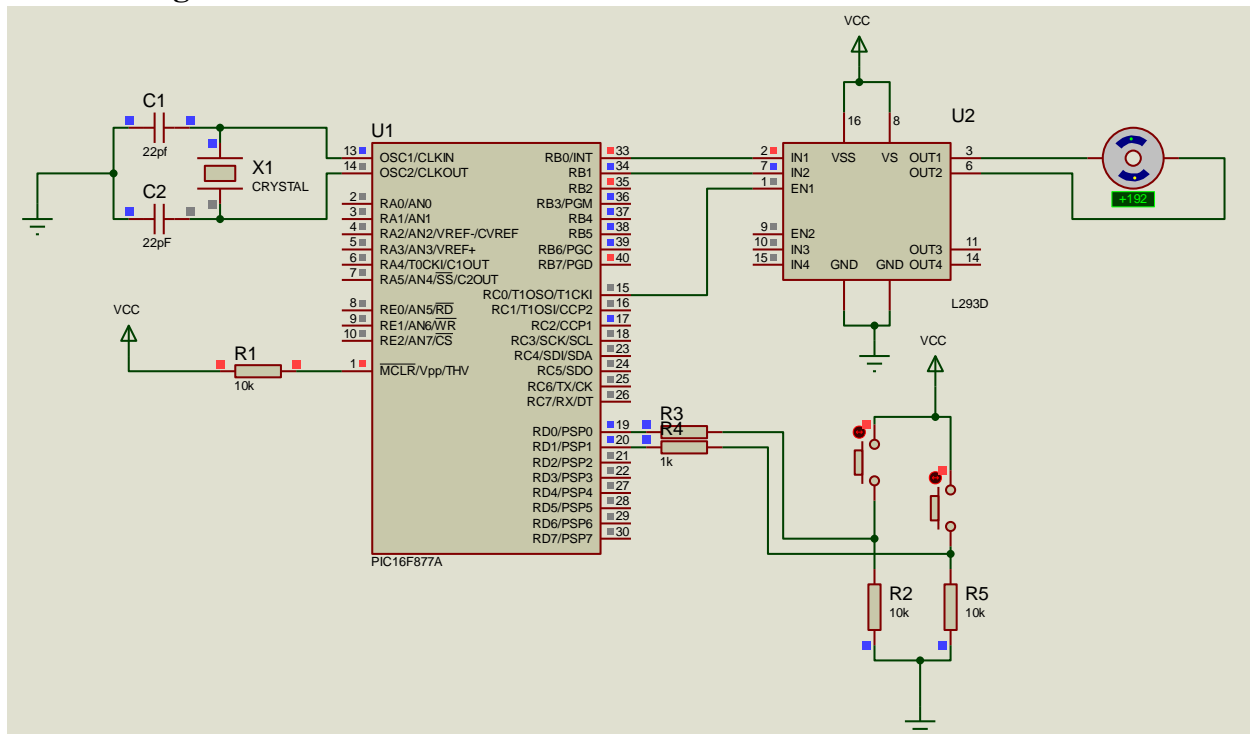


Figure 01: Circuit diagram for Speed control of DC motor using push button and PWM

Code:

```
void main() {
    short duty=0; // initial value for duty for motor
    TrisB=0x00; // as output
    TrisD=0xff; // as input
    portB.f0=0xff; //initialize the port
    portB.f1=0x00;

    PWM1_Init(1000);
    PWM1_Start();
    PWM1_Set_Duty(duty); // set current duty for pwm

    while(1)
    {
        if(portd.f0==1 && duty<250)
        {
            Delay_ms(100);
            if(portd.f0 && duty<250)
            {
```

```
    duty=duty+10;
    PWM1_Set_Duty(duty);
}
}

if(portd.fl==1 && duty>0)
{
    Delay_ms(100);
    if(portd.f0 && duty>0)
    {
        duty=duty-10;
        PWM1_Set_Duty(duty);
    }
}

delay_ms(10);
}

}
```

Circuit Diagram:

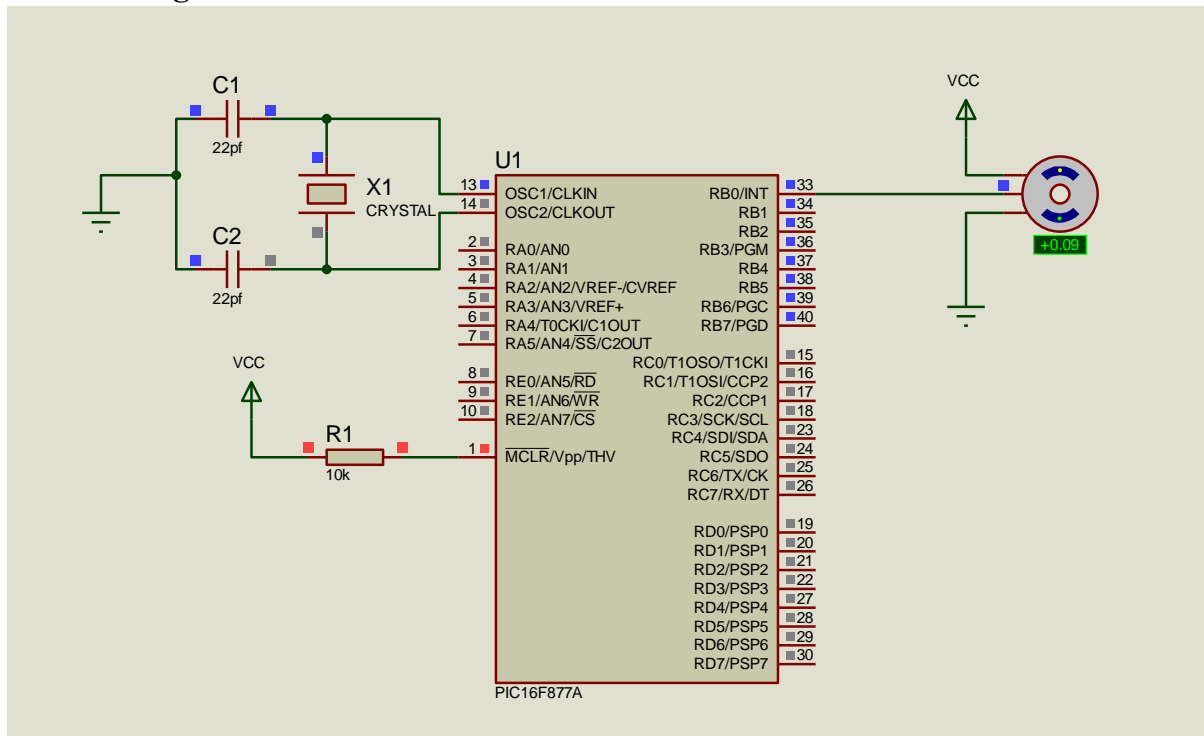


Figure 01: Circuit diagram for controlling servo motor (0-180)

Code:

```

void rotateLeft90();
void rotate0();
void rotateRight90();
int i;
void main(){
    Trisb=0x00;
    portb=0x00;
    while(1){
        rotateLeft90();
        delay_ms(2000);
        rotate0();
        delay_ms(2000);
        rotateRight90();
        delay_ms(2000);
    }
}
void rotateLeft90()

```

```
{  
  for(i=0;i<50;i++)  
  {  
    portb.f0=1;  
    delay_us(800);  
    portb.f0=0;  
    delay_us(19200);  
  }  
}
```

```
void rotate0()  
{  
  for(i=0;i<50;i++)  
  {  
    portb.f0=1;  
    delay_us(1500);  
    portb.f0=0;  
    delay_us(18500);  
  }  
}
```

```
void rotateRight90()  
{  
  for(i=0;i<50;i++)  
  {  
    portb.f0=1;  
    delay_us(2200);  
    portb.f0=0;  
    delay_us(17800);  
  }  
}
```

Circuit Diagram:

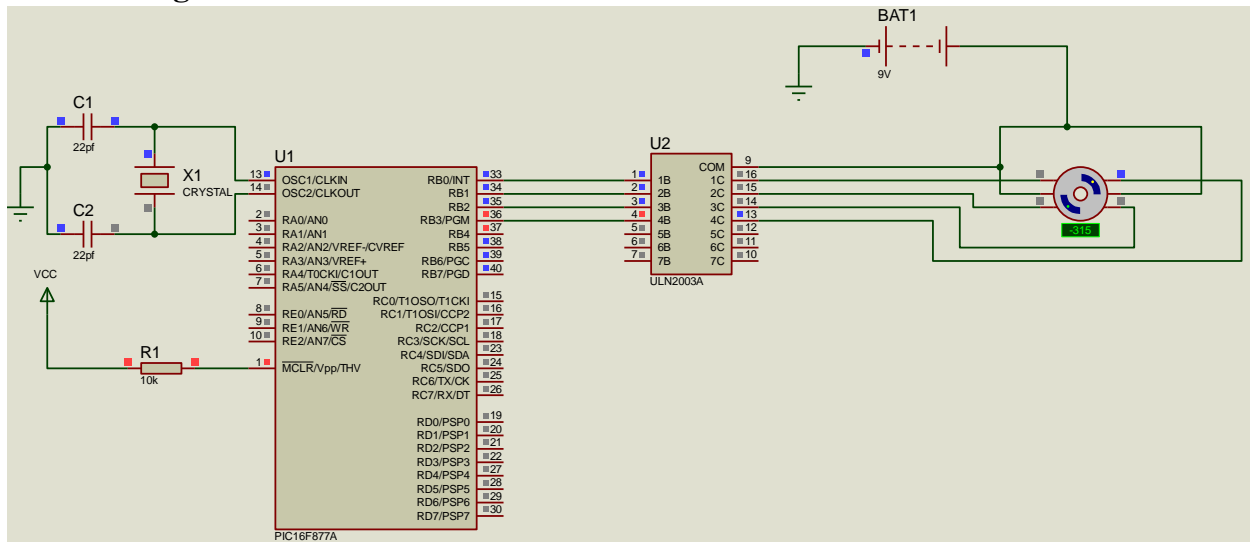


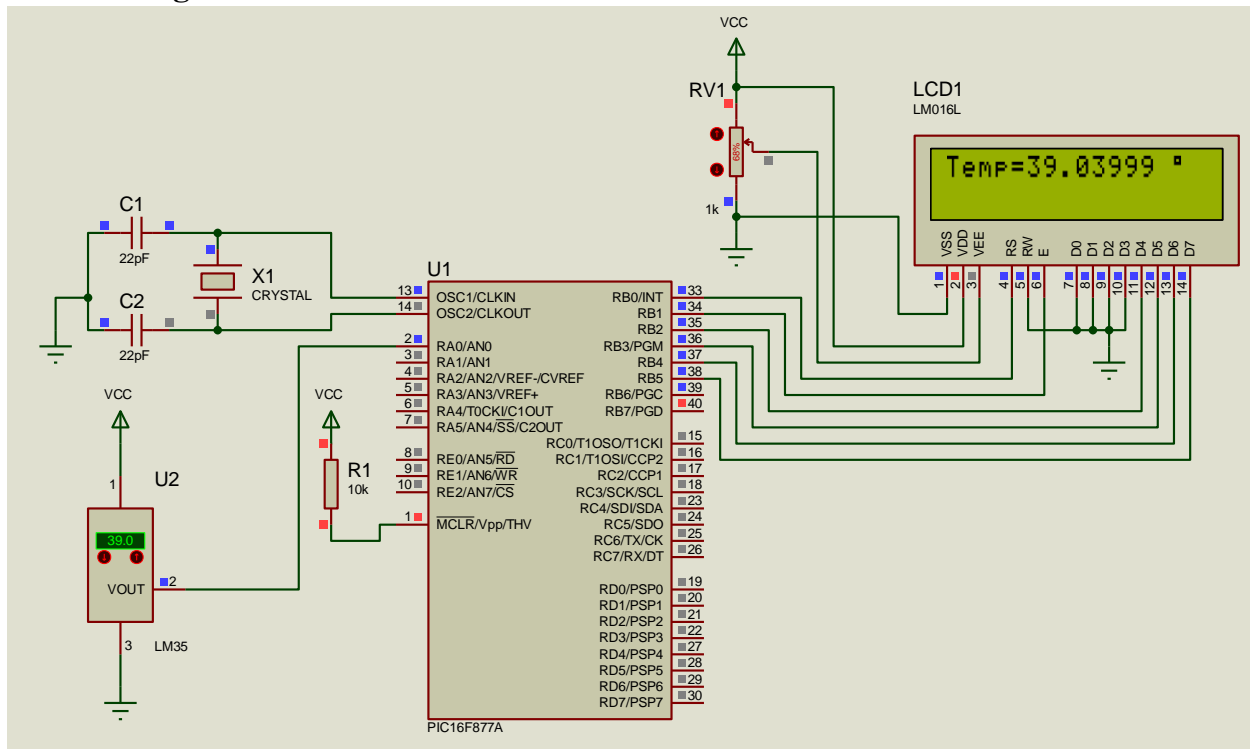
Figure 01: Circuit diagram for stepper motor control using pic microcontroller

Code:

```
void main() {
    Trisb=0b00000000;//Port b as output;
    portb=0b11111111;

    do{
        Portb=0b00000011;
        delay_ms(500);
        portb=0b00000110;
        delay_ms(500);
        portb=0b00001100;
        delay_ms(500);
        portb=0b00011000;
        delay_ms(500);
    }
    while(1);//loop excuted infinite
}
```

Circuit Diagram:



Code:

```
//LCD Module Connection
```

```
// Lcd pinout settings
```

```
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;
```

```
// Pin direction
```

```
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;
```

```
//LCD Module Connection
```

```
char display[16]="";
```

```
void main() {
```

```
    unsigned int result;
```

```
float volt,temp;
trisa=0x00;
trisa=0xff;
adcon1=0x80;
lcd_init();
lcd_cmd(_lcd_clear);
lcd_cmd(_LCD_CURSOR_OFF);
while(1)
{
    result=adc_Read(0);
    volt=result*4.88;
    temp=volt/10;
    lcd_out(1,1,"Temp=");
    FloatToStr(temp,display);
    lcd_out(1,6,display);
    lcd_chr(1,15,223);
    lcd_chr(1,16,"C");
}
}
```