**Circuit Diagram:**

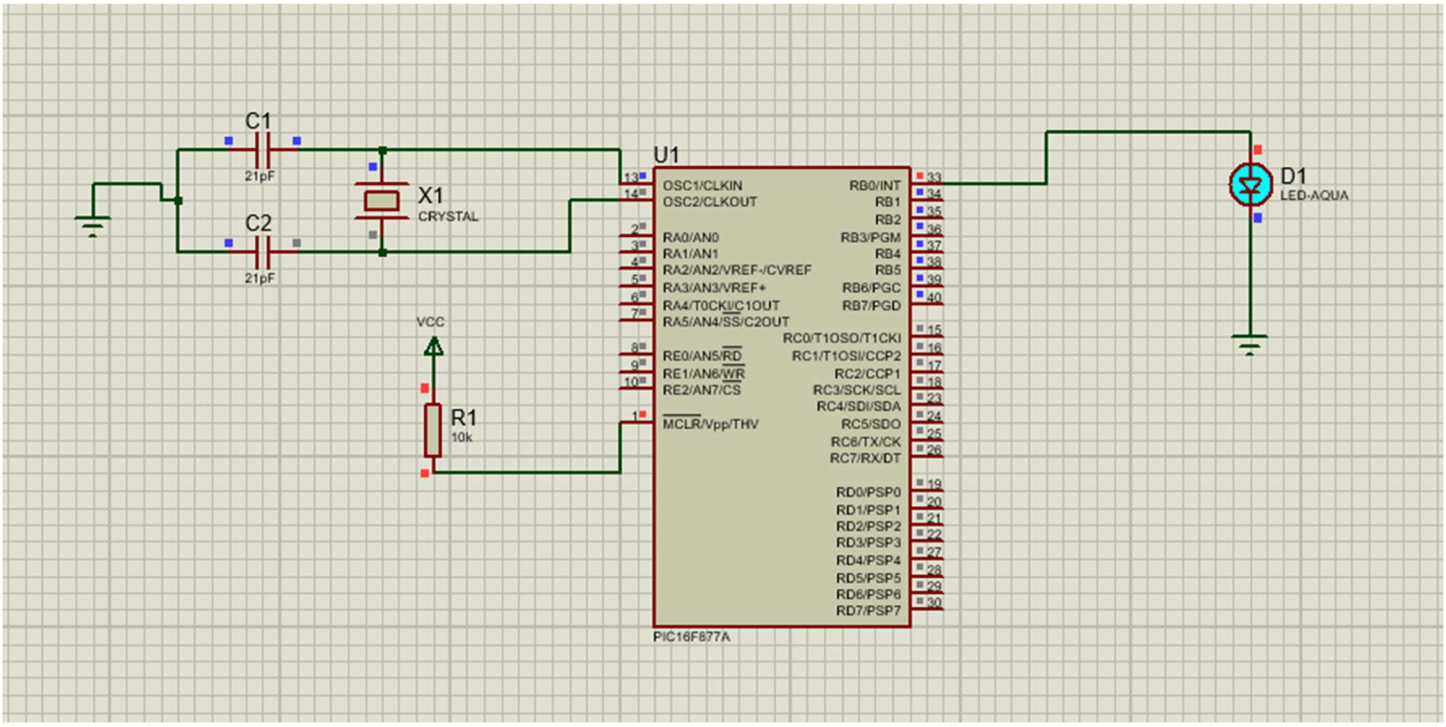


**Figure: Interfacing the LED blinking using the PIC16F877A**

**Source Code:**

```
void main() {
  TRISB = 0;    // PORTB as output
  PORTB = 0;    // Initialize PORTB

  while(1) {
    PORTB.F0 = 1;   // Turn ON
    Delay_ms(1000);
    PORTB.F0 = 0;   // Turn OFF
    Delay_ms(500);
  }
}
```

**Circuit Diagram:**

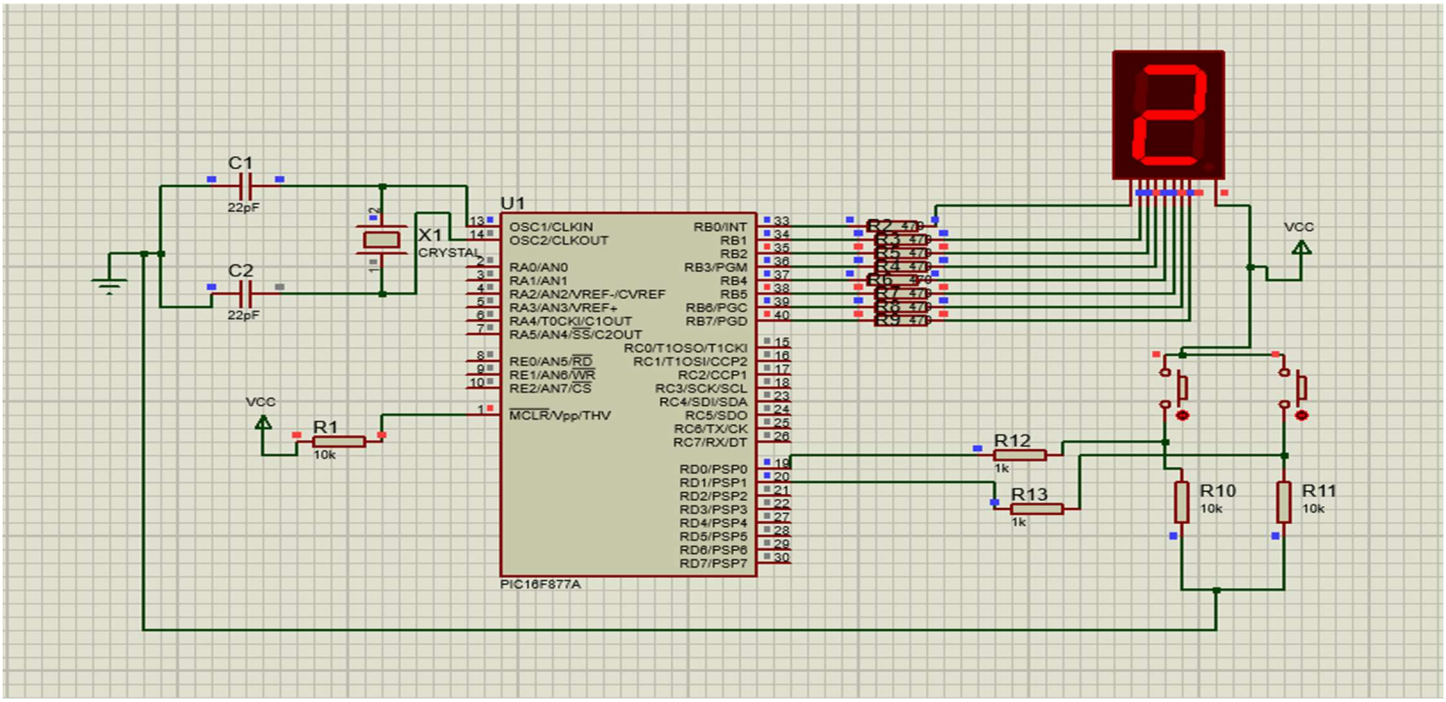


**Figure: Interfacing single digit 7 segment LED display with push bu on using the PIC16F877A**

**Source Code:**

```
char digits[] = {0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0x80,0x90};
void main() {
  int i = 0;
  TRISB = 0;   // 7-seg output
  TRISD = 0xFF; // buttons input
  while(1) {
    PORTB = digits[i];
    if(PORTD.F0 && i < 9) {   // Increment
      Delay_ms(150);
      if(PORTD.F0) i++;
    }
    if(PORTD.F1 && i > 0) {   // Decrement
      Delay_ms(150);
      if(PORTD.F1) i--;
    }
  }
}
```

**Circuit Diagram:**

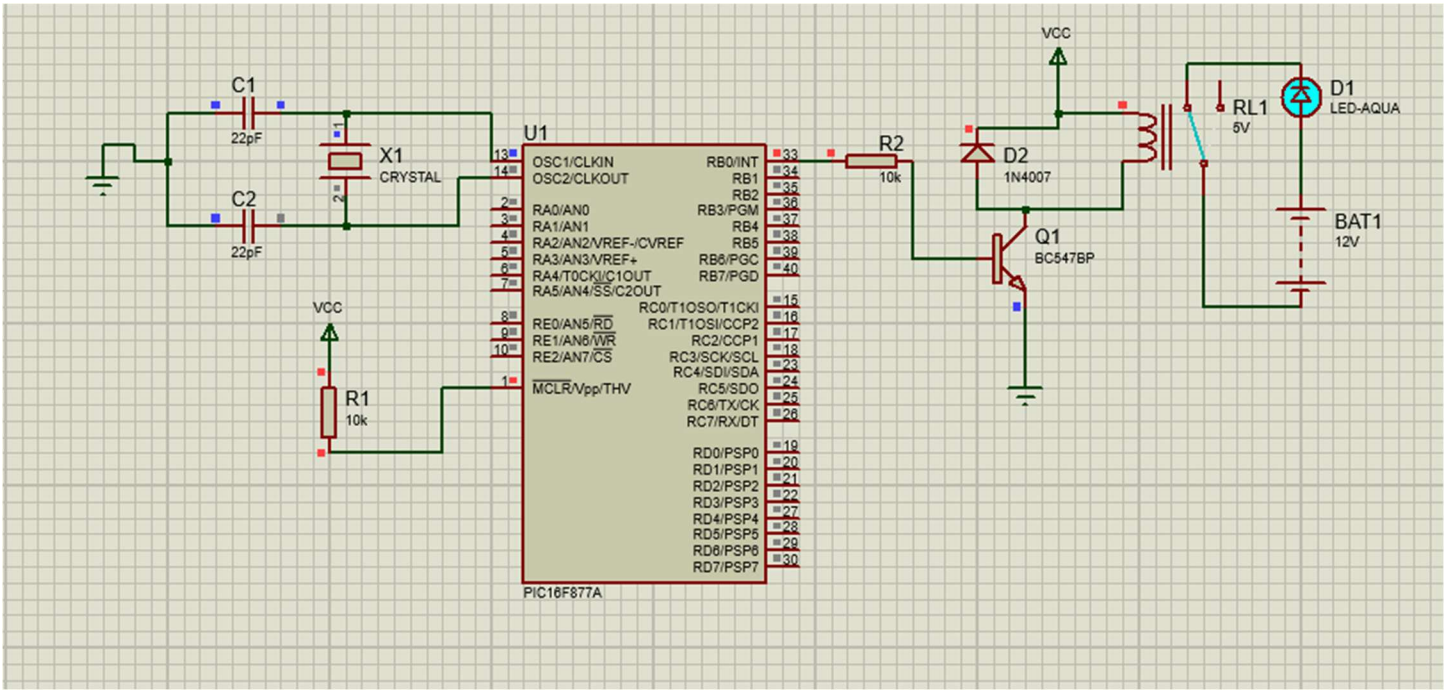


**Figure: Interfacing the relay using the PIC16F877A**

**Source Code:**

```
void main() {
  TRISB.F0 = 0;   // Set RB0 as output
  PORTB.F0 = 0;   // Initialize relay OFF

  while(1) {
    PORTB.F0 = 1;   // Turn relay ON (load ON)
    Delay_ms(2000);

    PORTB.F0 = 0;   // Turn relay OFF (load OFF)
    Delay_ms(2000);
  }
}
```

**Circuit Diagram:**

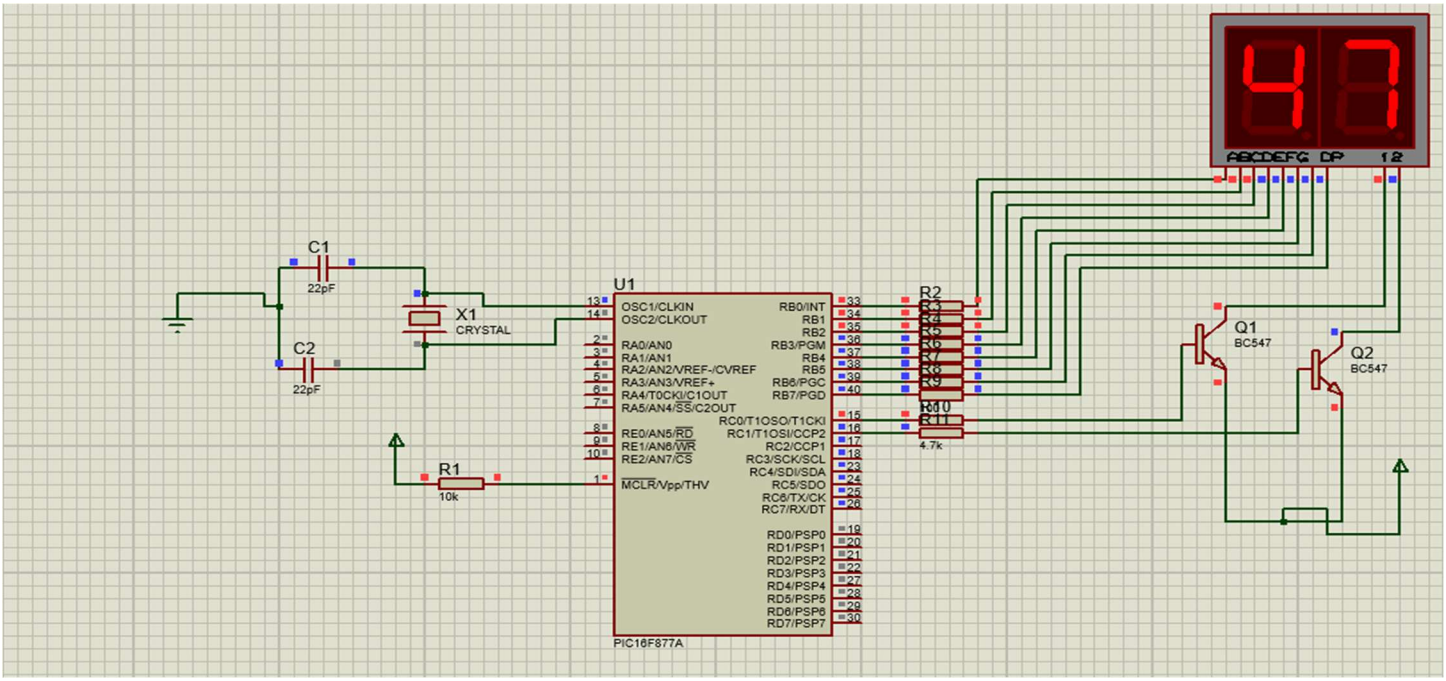


**Figure: Interfacing the two digit number using 7 segment multiplexing using the PIC16F877A**

**Source Code:**

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

void main() {
  int i, tens, ones, j;
  TRISB = 0;  // 7-seg data
  TRISC = 0;  // digit select
  PORTB = 0;
  PORTC = 0;

  while(1) {
    for(i = 0; i < 100; i++) {
      tens = i / 10;
      ones = i % 10;

      for(j = 0; j < 10; j++) {
        PORTC.F0 = 0;       // Tens digit ON
        PORTB = digits[tens];
        Delay_ms(5);
        PORTC.F0 = 1;       // OFF

        PORTC.F1 = 0;       // Ones digit ON
        PORTB = digits[ones];
        Delay_ms(5);
        PORTC.F1 = 1;       // OFF
      }
    }
  }
}
```

**Circuit Diagram:**

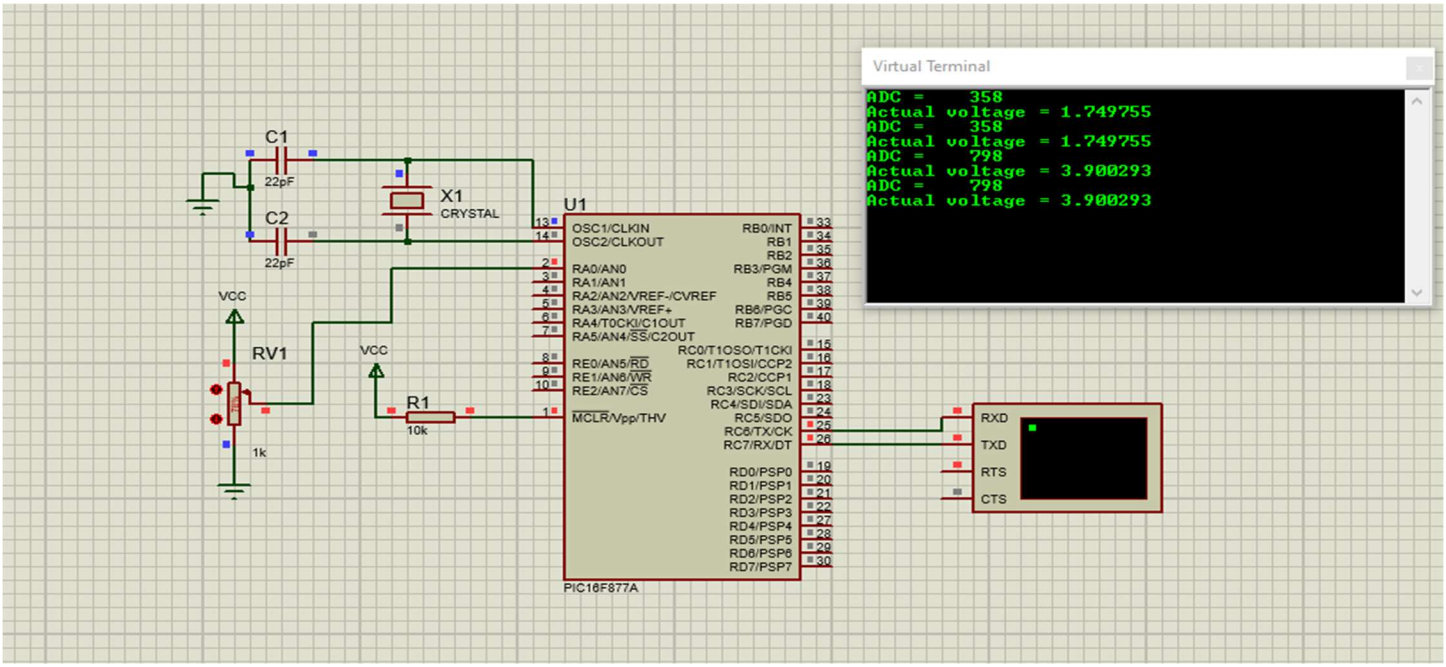


**Figure: Implement Analog to Digital Conversion (ADC) feature of PIC microcontroller using the PIC16F877A**

**Source Code:**

```
int valADC;
char x[7];  // buffer must be at least 7 chars
float voltage;

void main()
{
  UART1_Init(9600);
  ADC_Init();

  while(1)
  {
    valADC = ADC_Read(0);     // read ADC from RA0
    voltage = (valADC * 5.0) / 1023.0;
    IntToStr(valADC, x);      // convert to string
    UART1_Write_Text("ADC = ");
    UART1_Write_Text(x);
    UART1_Write(13);          // new line
    Delay_ms(1000);           // 1 sec delay
    FloatToStr(voltage, x);
    UART1_Write_Text("Actual voltage = ");
    UART1_Write_Text(x);
    UART1_Write(13);          // new line
    Delay_ms(1000);
  }
}
```

**Circuit Diagram:**

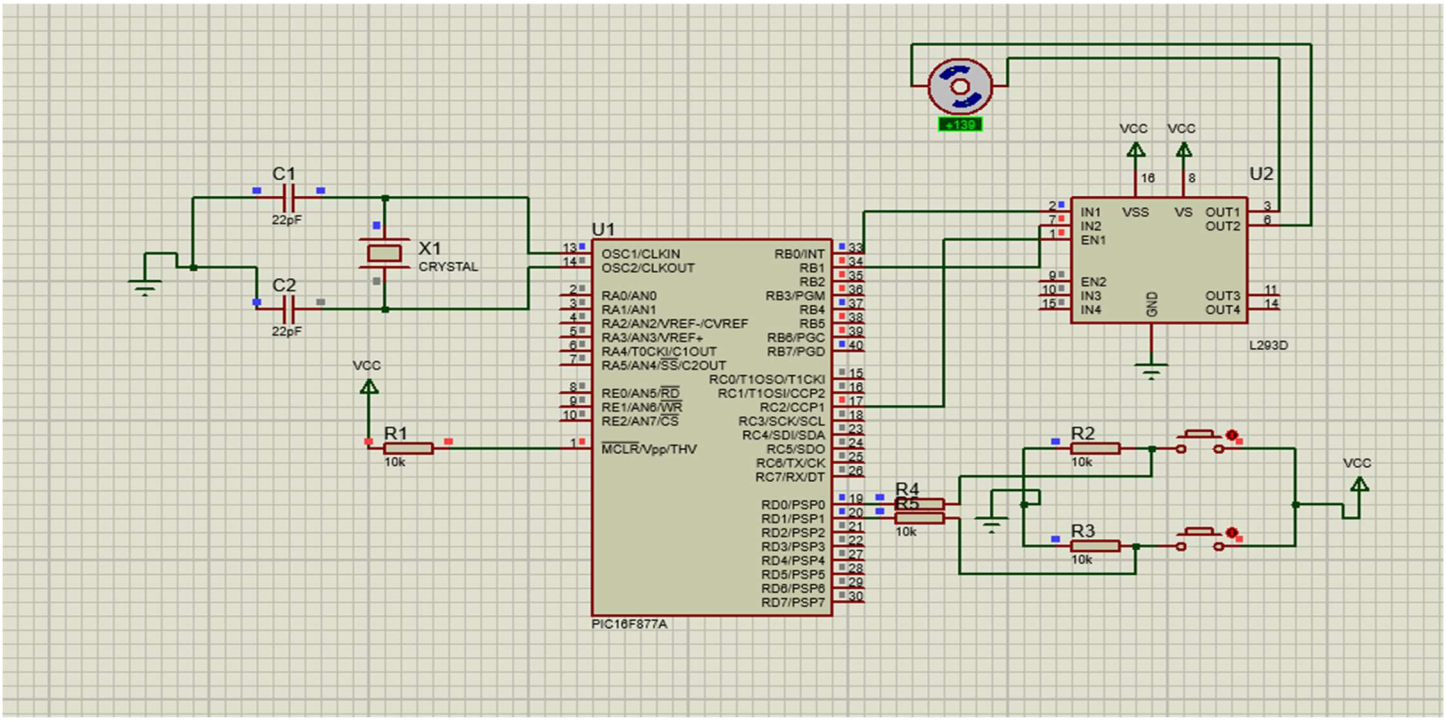


**Figure: Interfacing PWM and speed control of DC motor using the PIC16F877A**

**Source Code:**

```
void main() {
  short duty = 0;
  TRISD = 0xFF;   // Buttons
  TRISB = 0x00;   // Motor

  PORTB.F0 = 0;   // Anti-clockwise
  PORTB.F1 = 1;

  PWM1_Init(1000);
  PWM1_Start();
  PWM1_Set_Duty(duty);

  while(1) {
    if(RD0_bit && duty < 250) { duty += 10; PWM1_Set_Duty(duty); Delay_ms(100); }
    if(RD1_bit && duty > 0)  { duty -= 10; PWM1_Set_Duty(duty); Delay_ms(100); }
  }
}
```

**Circuit Diagram:**

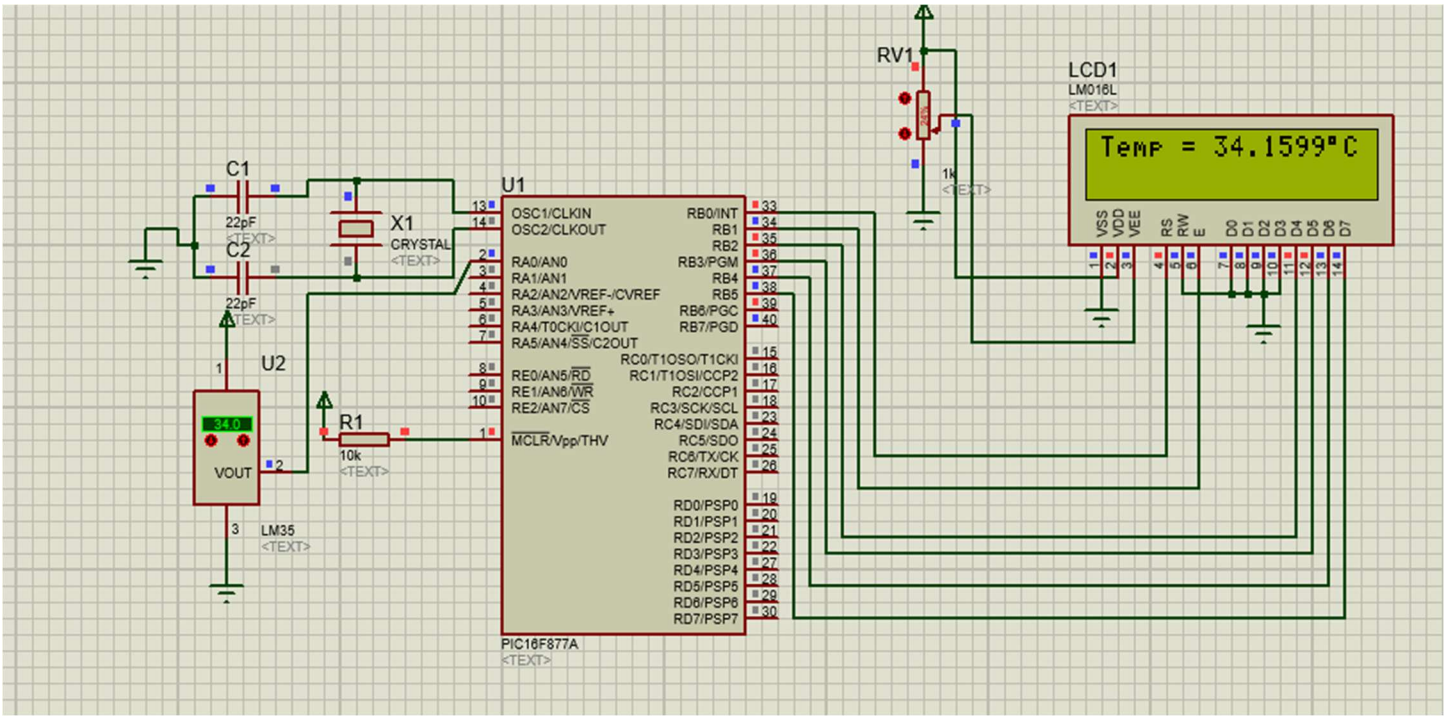


**Figure: Interfacing the LM35 temperature sensor with LCD display using the PIC16F877A**

**Source 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;  // LCD on PORTB ? output
  TRISA = 0xFF;  // RA0 ? input for LM35
  ADCON1 = 0x80;  // Configure RA0 as analog, Vref+ = Vdd, Vref- = GND
```

```
  Lcd_Init();
  Lcd_Cmd(_LCD_CLEAR);
  Lcd_Cmd(_LCD_CURSOR_OFF);

  while(1) {
    result = ADC_Read(0);       // Read ADC from channel 0 (AN0)
    volt = result * 4.88;       // mV (5000/1023 ~ 4.88)
    temp = volt / 10.0;         // °C (since 10mV per °C)

    Lcd_Out(1,1,"Temp = ");
    FloatToStr(temp, display);
    Lcd_Out(1,8,display);       // Print temperature value
    Lcd_Chr(1,15,223);          // Degree symbol
    Lcd_Out_CP("C");
    Delay_ms(500);              // Update every 0.5s
  }
}
```

**Circuit Diagram:**

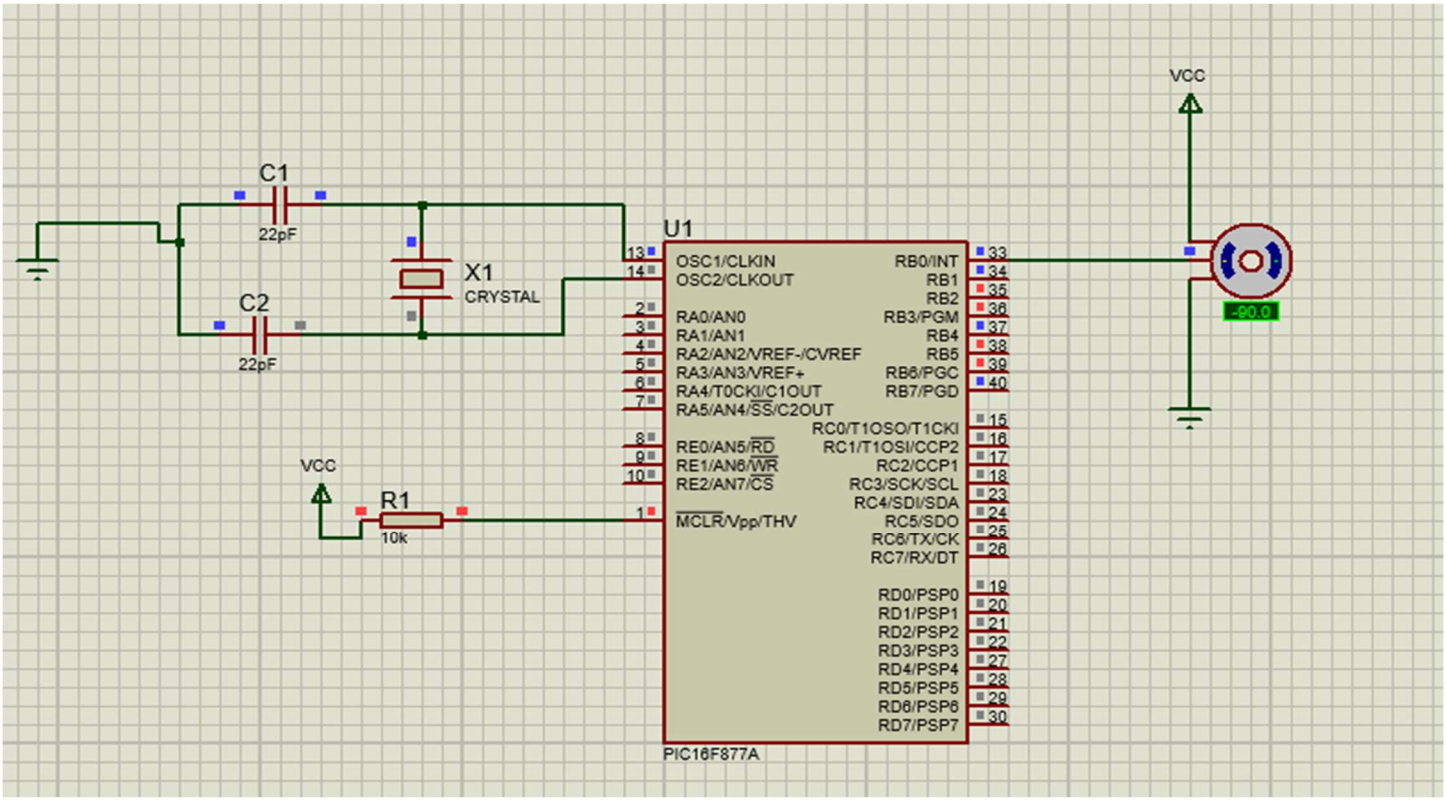


**Figure: Interfacing the servo motor using the PIC16F877A**

**Source Code:**

```
void servoRotate(int pulse) {
  unsigned int i;
  for(i=0;i<50;i++) {
    PORTB.F0 = 1;
    Delay_us(pulse);
    PORTB.F0 = 0;
    Delay_us(20000 - pulse);
  }
}

void main() {
  TRISB = 0;   // Output
  while(1) {
    servoRotate(800);   // 0°
    Delay_ms(2000);
    servoRotate(1500);  // 90°
    Delay_ms(2000);
    servoRotate(2200);  // 180°
  }
}
```

**Circuit Diagram:**

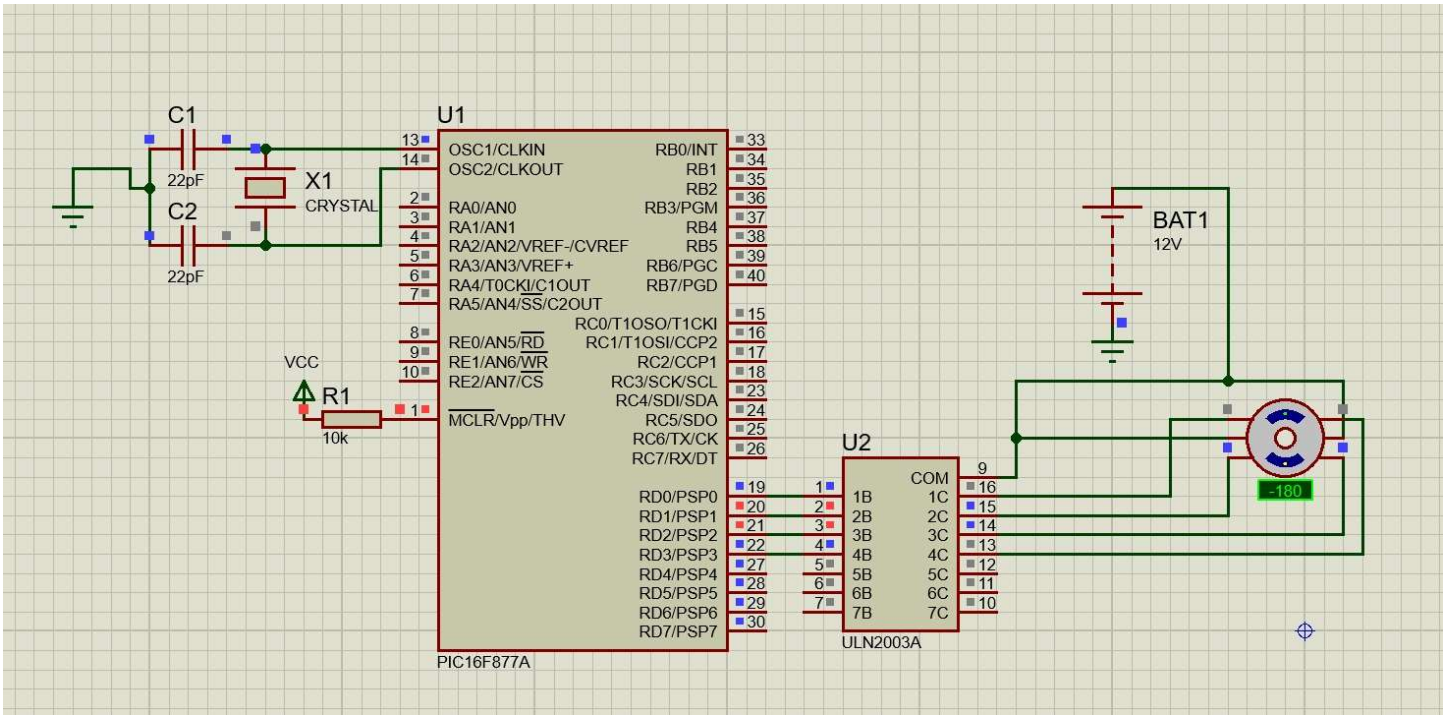


**Figure: Interfacing the servo motor using the PIC16F877A**

**Source Code:**

```
void main()
{
  TRISD = 0b00000000; // PORT D as output port
  PORTD = 0b11111111;

  do
  {
    PORTD = 0b00000011;
    Delay_ms(500); // delay of 0.5s

    PORTD = 0b00000110;
    Delay_ms(500);

    PORTD = 0b00001100;
    Delay_ms(500);

    PORTD = 0b00001001;
    Delay_ms(500);
    // energizing two phases at a time
  } while(1); // loop executed infinite times
}
```

**Circuit Diagram:**

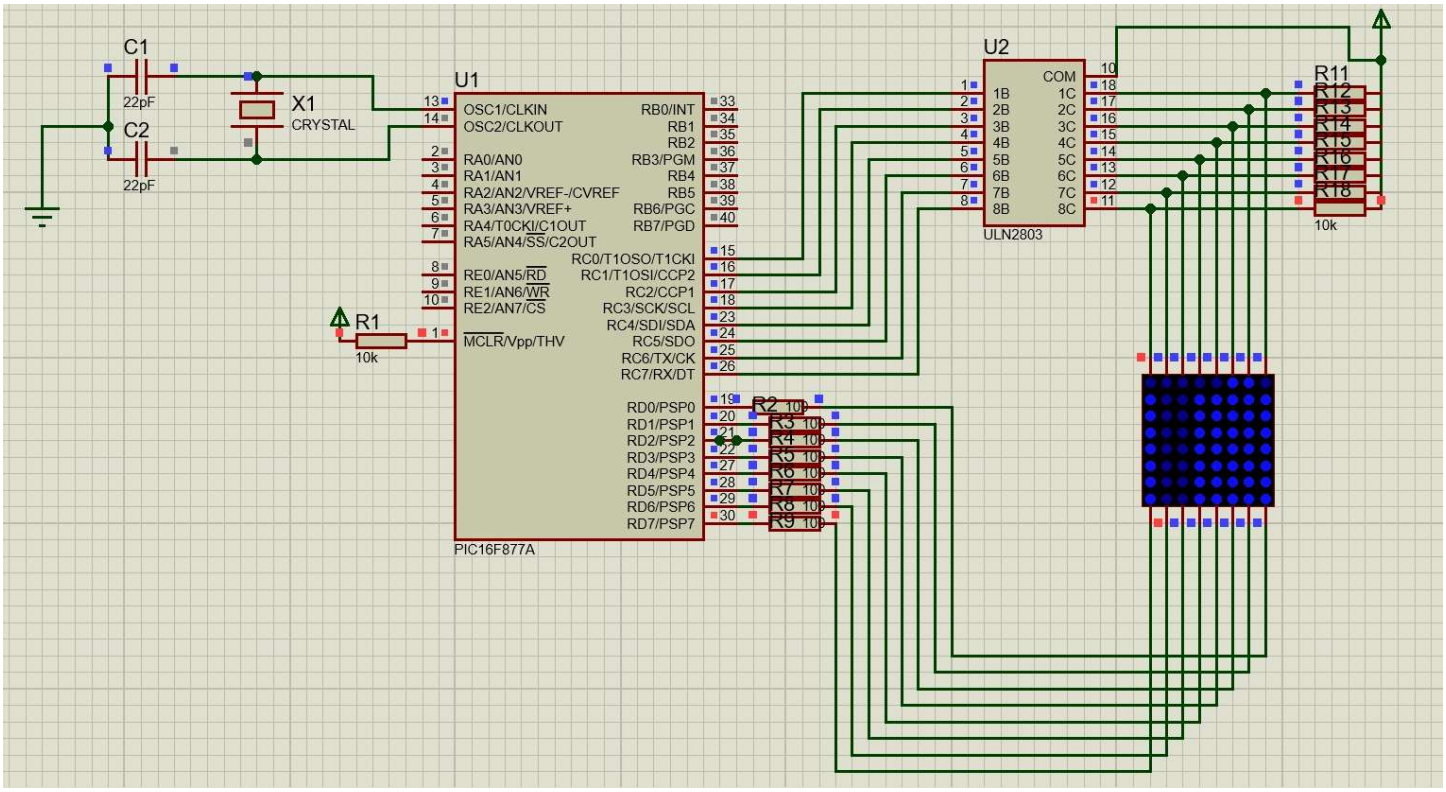


**Figure: Interfacing the dot matrix display using the PIC16F877A**

**Source Code:**

```
void MSDelay(unsigned char Time)
{
  unsigned char y, z;
  for(y = 0; y < Time; y++)
    for(z = 0; z < 250; z++);
}
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);
  }
}
```