MEASURE 0 TO 100VDC USING MICRO CONTROLLER

MICROCONTROLLER :- PIC16F877A

IDE & COMPILER:-

MPlab IDE with XC8 compiler.

CALCULATION:-

- Voltage Divider Calculation:-

The voltage divider will consist of two resistors, R1 and R2, connected in series. The input voltage (Vin) is applied across the series combination of R1 and R2, and the output voltage (Vout) is taken from the junction of R1 and R2to ground.

The formula for the output voltage Vout is given by: Vout = Vin+R2/R1+R2

To scale 0-100V to 0-5V:-

5V = 100V * R2/R1+R2

By solving the above equation you get **R1 = 190kohm** and **R2= 10kohm.**

Algorithm

1. Initialization

1.1 Configure the microcontroller.

1.2 Initialize the CLCD.

1.3 Initialize the ADC.

2. Main Loop

2.1 Display static text "Lithium Power" on the first line of the CLCD.

2.2 Repeat the following steps indefinitely:

2.2.1 Read the ADC value.

2.2.2 Convert the ADC value to voltage.

2.2.3 Scale the voltage to the range 0-100V.

2.2.4 Ensure the voltage is within 0-100V.

2.2.5 Format the voltage as a string.

2.2.6 Display the voltage on the second line of the CLCD.

2.2.7 Delay for 500 ms before repeating.

CODE

Main.c

```
#include <xc.h>
#include <stdio.h>
#include "clcd.h"

#pragma config WDTE = OFF       // Watchdog Timer Enable bit (WDT disabled)

#define _XTAL_FREQ 20000000

void init_config(void) {
    init_clcd();
    // Initialize ADC
    ADCON0 = 0x41; // ADC enabled, select AN0 channel
    ADCON1 = 0x8E; // Right justify result, VDD as reference, AN0 as analog input
}

unsigned int read_adc(void) {
    __delay_us(20); // Acquisition time
    GO_nDONE = 1; // Start conversion
    while (GO_nDONE); // Wait for conversion to finish
    return ((unsigned int)(ADRESH << 8) + ADRESL); // Return result
}

void main(void) {
    char buffer[16];
    unsigned int adc_value;
    float voltage;

    init_config();

    // Display static text on the first line
    clcd_print("Lithium Power", LINE1(2));

    while (1) {
        adc_value = read_adc(); // Read ADC value
        voltage = ((float)adc_value * 5.0f) / 1023.0f; // Scale to 0-5V

        // Assuming a voltage divider that scales 0-100V to 0-5V
        voltage = voltage * 20.0f; // Scale 0-5V to 0-100V

        // Ensure the voltage is displayed in the range 0-100V
        if (voltage > 100.0f) {
            voltage = 100.0f;
        } else if (voltage < 0.0f) {
            voltage = 0.0f;
        }
```

```
        sprintf(buffer, "Voltage: %.2fV", voltage);
        clcd_print(buffer, LINE2(0));

        __delay_ms(500); // Delay for display update
    }
    return;
}
```

Clcd.c

```
#include <xc.h>
#include "clcd.h"

static void clcd_write(unsigned char byte, unsigned char mode)
{
    CLCD_RS = (unsigned char)mode;

    CLCD_DATA_PORT = byte & 0xF0;
    CLCD_EN = HI;
    __delay_us(100);
    CLCD_EN = LOW;

    CLCD_DATA_PORT = (unsigned char)((byte & 0x0F) << 4);
    CLCD_EN = HI;
    __delay_us(100);
    CLCD_EN = LOW;

    __delay_us(4100);
}

static void init_display_controller(void)
{
    /* Startup Time for the CLCD controller */
    __delay_ms(30);

    /* The CLCD Startup Sequence */
    clcd_write(EIGHT_BIT_MODE, INST_MODE);
    __delay_us(4100);
    clcd_write(EIGHT_BIT_MODE, INST_MODE);
    __delay_us(100);
    clcd_write(EIGHT_BIT_MODE, INST_MODE);
    __delay_us(1);

    clcd_write(FOUR_BIT_MODE, INST_MODE);
    __delay_us(100);
    clcd_write(TWO_LINES_5x8_4_BIT_MODE, INST_MODE);
    __delay_us(100);
    clcd_write(CLEAR_DISP_SCREEN, INST_MODE);
    __delay_us(500);
    clcd_write(DISP_ON_AND_CURSOR_OFF, INST_MODE);
    __delay_us(100);
```

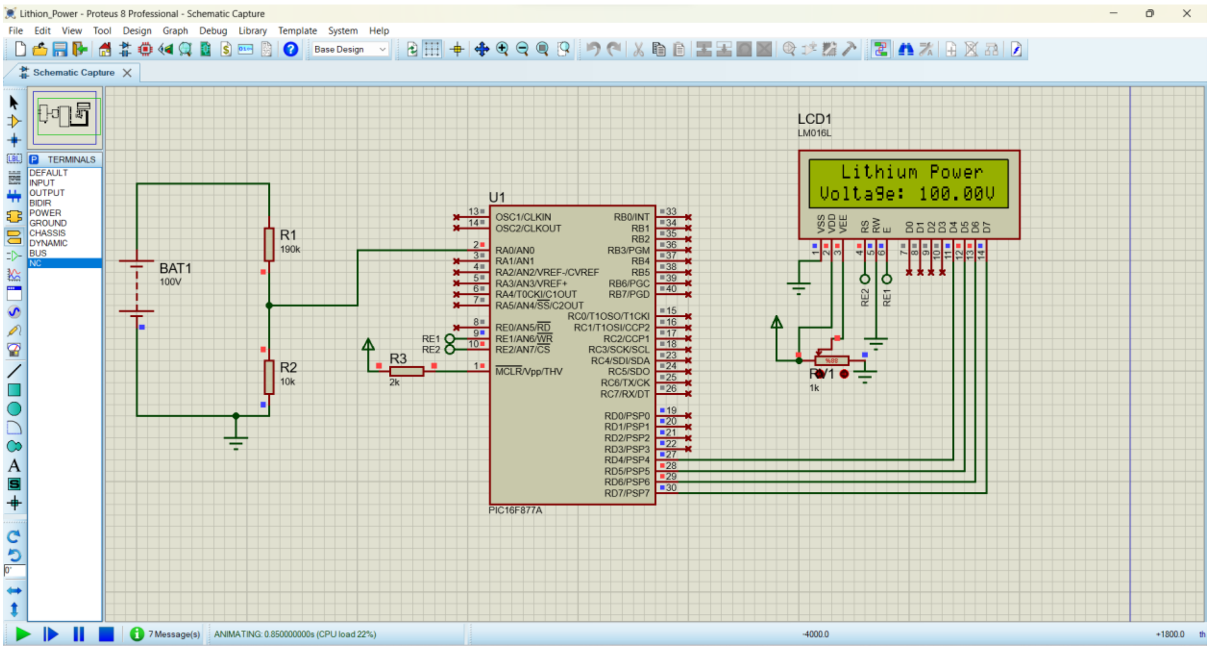
```
}

void init_clcd(void)
{
   /* Setting the CLCD Data Port as Output */
   CLCD_DATA_PORT_DDR = 0x00;

   /* Setting the RS and EN lines as Output */
   CLCD_RS_DDR = 0;
   CLCD_EN_DDR = 0;

   init_display_controller();
}

void clcd_putch(const char data, unsigned char addr)
{
   clcd_write(addr, INST_MODE);
   clcd_write(data, DATA_MODE);
}

void clcd_print(const char *str, unsigned char addr)
{
   clcd_write(addr, INST_MODE);

   while (*str != '\0')
   {
      clcd_write(*str, DATA_MODE);
      str++;
   }
}
```

Clcd.h

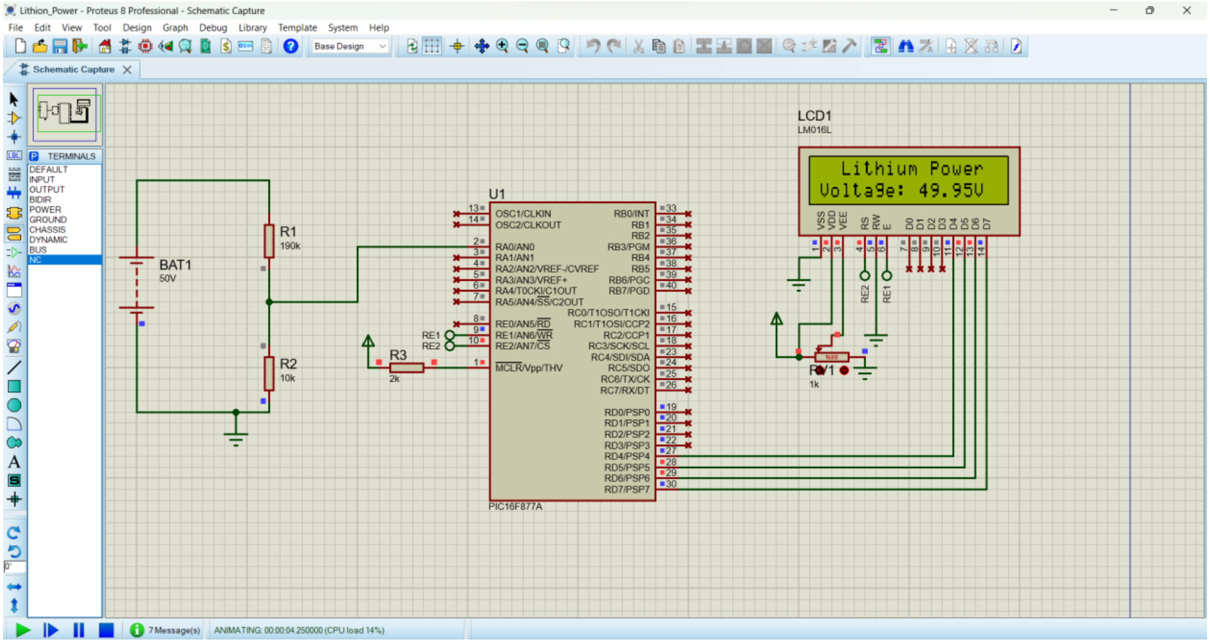
```
/*
 * File:   clcd.h
 */

#ifndef CLCD_H
#define CLCD_H

#define _XTAL_FREQ              20000000

#define CLCD_DATA_PORT_DDR        TRISD
#define CLCD_RS_DDR               TRISE2
#define CLCD_EN_DDR               TRISE1

#define CLCD_DATA_PORT            PORTD
#define CLCD_RS                   RE2
#define CLCD_EN                   RE1

#define INST_MODE                 0
#define DATA_MODE                 1
```

```
#define HI                    1
#define LOW                   0

#define LINE1(x)              (0x80 + x)
#define LINE2(x)              (0xC0 + x)

#define EIGHT_BIT_MODE            0x33
#define FOUR_BIT_MODE             0x02
#define TWO_LINES_5x8_4_BIT_MODE   0x28
#define CLEAR_DISP_SCREEN          0x01
#define DISP_ON_AND_CURSOR_OFF     0x0C

void init_clcd(void);
void clcd_putch(const char data, unsigned char addr);
void clcd_print(const char *str, unsigned char addr);

#endif   /* CLCD_H */
```

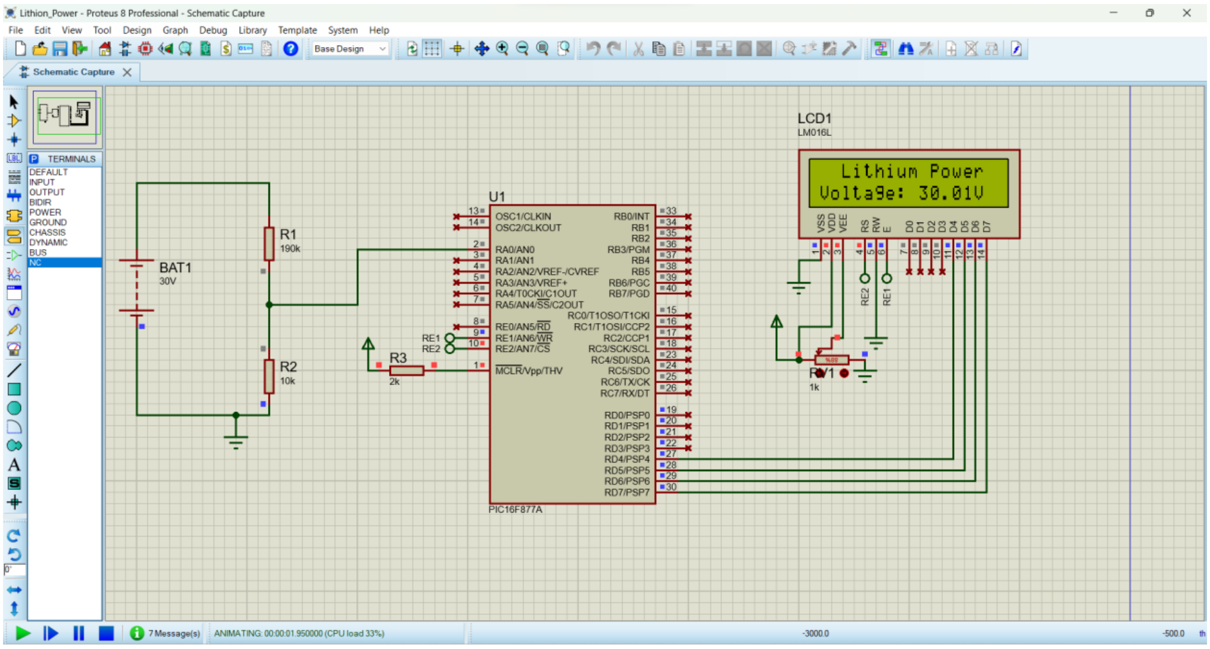
FOR 100V INPUT



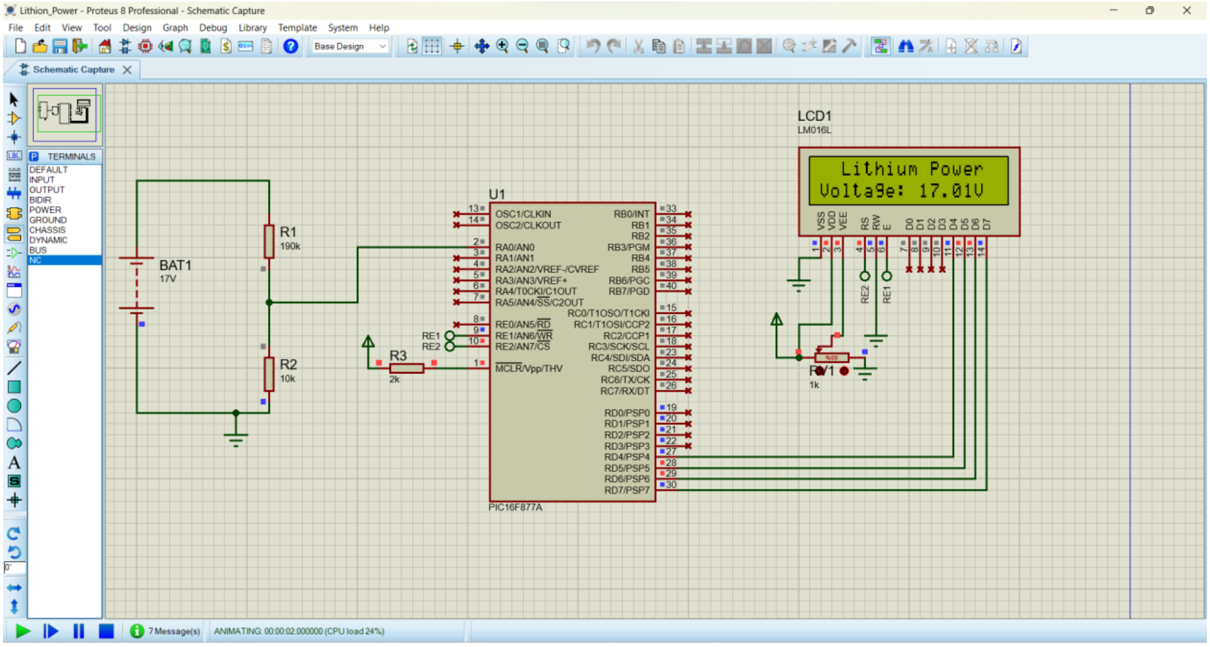
FOR 50V INPUT



FOR 30V INPUT



FOR 17V INPUT



NOTE:-

1. **Choose an Appropriate DC to DC Converter:**
   - Select a step-down (buck) converter that can handle your input voltage and is capable of being configured to output 5.5V. For example, the LM2596 adjustable voltage regulator is a popular choice.
2. **Configure the Output Voltage:**
   - Adjust the feedback resistors of the converter to set the output voltage to 5.5V. This can usually be done by referring to the datasheet of the specific DC to DC converter you are using.

Example Setup

1. **Select the Converter**: For example, using the LM2596 adjustable voltage regulator.
2. **Configure the Feedback Resistors**: The output voltage Vout of a buck converter can be set using the formula:

   Vout = Vref(1+R1/R2)

   where Vref is typically 1.23V for the LM2596. Choose appropriate resistor values R1R1R1 and R2R2R2 to achieve 5.5V output.
3. **Connect the Circuit**:
   - **Input**: Connect the input of the DC to DC converter to your power source.
   - **Output**: Connect the output of the converter to the microcontroller's power input.