جامعة الإسكندرية
ALEXANDRIA UNIVERSITY

EEP112 - Electrical and Electronic Measurements

Final Project Report

Arduino-based Ohmmeter and Voltmeter Instrument

Ahmed Saeed Muhammed	21010091
Yahya Emad El-Deen Muhammed	21011557
Muhammed Ibrahim Risk	21011047
Yahya Saeed Hamed	21010705
Mariam Ahmed Fathy	19016627

Electronics and Communication Engineering Department – ECEE

May 9, 2023(Spring)

1 Introduction

This project involves building an Arduino-based voltmeter and ohmmeter. The project requires writing code using the Arduino IDE to measure voltages and resistances using an LCD display anda switch to select different ranges. The Arduino board is programmed to take multiple readings and calculate the average voltage value, and then use this value to calculate the unknown resistance inthe circuit. The ohmmeter function includes several ranges that can be selected by the user, and the Arduino board prompts the user to select the correct range or add a resistor if necessary.

2 Objective

Making a digital measuring instrument that can measure both voltage and resistance, functioning as a voltmeter and a multirange ohmmeter, with as much accuracy as possible, and a small as possible range of error, and displaying the measured value on a lcd display.

3 Components

- **LCD** (16×2)
  This 16 × 2 LCD packs 32 characters into an outline smaller than that of most two-line displays. A LED backlight enables optimal viewing in all lighting conditions. This unit uses the HD44780 interface found on most parallel character displays.
- **Arduino uno**
  Arduino Uno is an open-source microcontroller board based on the ATmega328P microcontroller and can be programmed using the Arduino Integrated Development Environment (IDE).
- **Resistors** (1K, 2K, 10K, 2.2K, 22K, 270K)
- **DIP Switch**
- **Wires**
- **Breadboard**

4 Procedure

4.1 Hardware

Connect the circuit as shown below:

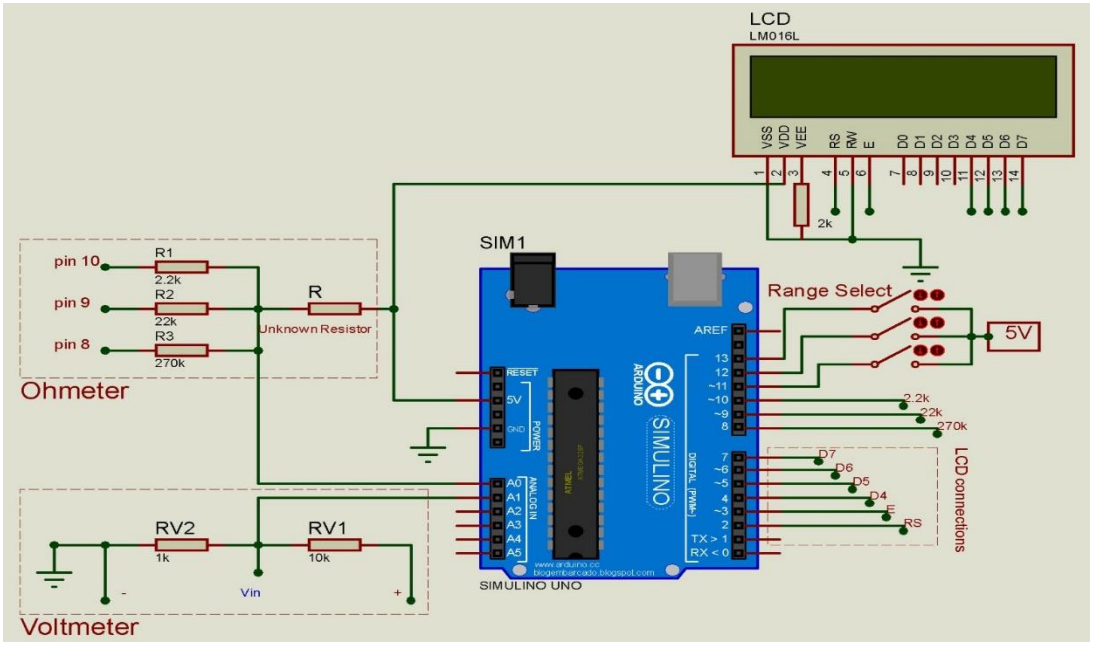

Figure 1: Ohmmeter and Voltmeter Instrument Hardware connections

1. Connect the LCD pins as follows:
    - Pin 1 to GND
    - Pin 2 to 5V
    - Pin 3 to a 2 KOhm resistor for adjusting contrast.
    - Pin 4 to Arduino digital pin 2
    - Pin 5 to GND
    - Pin 6 to Arduino digital pin 3
    - Pin 11 to Arduino digital pin 4
    - Pin 12 to Arduino digital pin 5
    - Pin 13 to Arduino digital pin 6
    - Pin 14 to Arduino digital pin 7
2. For the voltmeter connections:
    - Connect two reference resistors (1 Kohm and 10 Kohm) in parallel.
    - Connect the common pin to the Arduino's analog A1 pin.
    - Connect one cable for positive and one for negative to measure the desired unknown voltage.

3. For the ohmmeter connections:
    - Use three reference resistances (2.2 Kohm, 22 kohm, and 270 Kohm).
    - Connect one of the resistances to the common pin.
    - Connect the common pin to the Arduino’s analog A0 pin.
4. Use a range select switch to choose the reference resistor for the ohmmeter:
    - Connect the three switches to Arduino digital pins 11, 12, and 13 in order.
    - Connect all switches to 5V from the other end.

4.2 Software

1. Initiate the LCD display using the Arduino's IDE.

2. Initiate and identify the values to be measured.

```
#include   <Wire.h>
#include <LiquidCrystal.h>

LiquidCrystal lcd(2,3,4,5,6,7);    //rs,e,d4,d5,d6,d7

float Vm = 0.0; // unknown voltage
float R2=0; // unknown resistor
```

3. Write a voltmeter function that calculates the desired voltage using the average of multiple readings.

```
////////////////////////////////      Voltameter
   ////////////////////////////////
void calculate_voltage()
{
  float r1= 10000.0;
  float r2= 1000.0;
  float v_ref=5.00;
  int sensorValue =0.0;
  float vin=0.0;
  float r_ratio= (r2/(r1+r2));



for(int i=0;i<30;i++)
  {
    sensorValue = sensorValue +analogRead(A1);
    delay(3);
  }

  sensorValue= sensorValue/30;
```

```
  vin=((sensorValue*v_ref)/1023.0);
  Vm=vin/r_ratio;
}
```

4. Initiate and identify the Ohmmeter's variables.

```
//////////////////////////////// Ohmmeter
    ////////////////////////////////////

int analogPin= 0;
int V_measured= 0;
int Vin= 5;
float Vout= 0;

float buffer= 0;

int ch2K = 10;
int ch20K = 9;
int ch200K = 8;


int Scale2k=13;
int Scale20k=12;
int Scale200k=11;
```

5. Set up functions for each ohmmeter's range. These functions take the voltage reading, calculate the unknown resistance, and print it out if it's within range. If needed, the function will ask you to change the range.
   - **2K Scale**

```
void fn2kscale ()
{
if (digitalRead(Scale2k))
  {

  pinMode(ch2K,OUTPUT);
  pinMode(ch20K,INPUT);
  pinMode(ch200K,INPUT);
  digitalWrite(ch2K,LOW);

  float R1= 2.15; // Set this values to the value of the used resistor in
   K ohms
  V_measured= analogRead(analogPin);
```

```
13
14        buffer= V_measured * Vin;
15        Vout= (buffer)/1024.0; //in volts
16        buffer= (Vin/Vout) -1;
17        R2= R1 * buffer*1000; //*1000 because we express it in ohms
18
19        if (R2 > 2200)
20        {
21          lcd.setCursor(0,1);
22          lcd.print(" Increase/insert");
23          delay(1000);
24           lcd.clear();
25        }
26
27        if (R2 < 2200)
28        {
29          lcd.setCursor(0,1);
30          lcd.print("R1: ");
31          lcd.print(R2);
32          lcd.setCursor(13,1);
33          lcd.print("Ohm");
34          delay(1000);
35        }
36    }
37 }
```

- **20K Scale**

```
1  void fn20kscale()
2  {
3  if (digitalRead(Scale20k))
4    {
5    pinMode(ch2K,INPUT);
6    pinMode(ch20K,OUTPUT);
7    pinMode(ch200K,INPUT);
8    digitalWrite(ch20K,LOW);
9
10   float R1= 21.3; // Set this values to the value of the used resistor in
       K ohms
11   V_measured= analogRead(analogPin); //in 8bits
12
13       buffer= V_measured * Vin;
14       Vout= (buffer)/1024.0;  //in volts
```

```
      buffer= (Vin/Vout) -1;
      R2= R1 * buffer;

      if (R2 > 22)
      {
        lcd.setCursor(0,1);
        lcd.print(" Increase/insert");
        delay(1000);
        lcd.clear();
      }

      if (R2 <22)
      {

        lcd.setCursor(0,1);
        lcd.print("R2: ");
        lcd.print(R2);
        delay(1000);
        lcd.setCursor(12,1);
        lcd.print("Kohm");
        delay(1000);
      }

    }
}
```

- **200K Scale**

```
void fn200kscale(){
  if (digitalRead(Scale200k))
  {

  pinMode(ch2K,INPUT);
  pinMode(ch20K,INPUT);
  pinMode(ch200K,OUTPUT);

  digitalWrite(ch200K,LOW);

  float R1= 275; // Set this values to the value of the used resistor in K
     ohms
  V_measured= analogRead(analogPin); //in 8bits

      buffer= V_measured * Vin;
```

```
      Vout= (buffer)/1024.0; //in volts
      buffer= (Vin/Vout) -1;
      R2= R1 * buffer;

      if (R2 > 300)
      {
        lcd.setCursor(0,1);
        lcd.print("Insert Res.");
        delay(1000);
        lcd.clear();
      }

      if (R2 < 300)
      {
        lcd.setCursor(0,1);
        lcd.print("R3: ");
        lcd.print(R2);
        delay(1000);
        lcd.setCursor(12,1);
        lcd.print("Kohm");
        delay(1000);
      }

  }

}
```

6. Write the ohmmeter's function that calls the appropriate range function according to the chosen range or tells you to select a scale or put a resistor to measure if no reading is present.

```
void resistance_calc()
{
  ////////////////////-2k-//////////////////////
  if(digitalRead(13))
    fn2kscale();
//////////////////////////////////////////////

////////////////////-20k-//////////////////////
  if(digitalRead(12))
    fn20kscale();
//////////////////////////////////////////////

////////////////////-200k-//////////////////////
```

```
  if(digitalRead(11))
    fn200kscale();
////////////////////////////////////////
  if ( (digitalRead(Scale2k)==LOW) && (digitalRead(Scale20k)==LOW) && (
   digitalRead(Scale200k)==LOW) )
  {
    lcd.setCursor(0,1);
    lcd.print("ScaleAndResistor");
    delay(1000);
  }
}
```

7. Write the setup function that reads the chosen to range from the switch connected and starts the LCD and the board, then the loop function that calls the voltmeter function prints the measured value and then calls the ohmmeter's function.

```
void setup()
{
  lcd.begin(16, 2);
  Serial.begin(9600);
  pinMode(V_measured,INPUT);
  pinMode(analogPin,INPUT);


  //Detect the range mode (0-1kK; 10k-100k; 100k-1M)
  pinMode(Scale2k,INPUT);
  pinMode(Scale20k,INPUT);
  pinMode(Scale200k,INPUT);



//We set this pins as input for now.
  pinMode(ch2K,INPUT);
  pinMode(ch20K,INPUT);
  pinMode(ch200K,INPUT);


}

void loop()
{
calculate_voltage();
```

```
  lcd.setCursor(0,0);
  lcd.print("V= ");
  lcd.print(Vm);
  lcd.setCursor(9,0);
  lcd.print("Volts");
  delay(10);
  resistance_calc();
}
```

After writing the code, upload it to the Arduino board using the IDE. Once uploaded, the Arduino will begin measuring voltage and resistance according to the code's instructions.

5 Discussions, Results, and Challenges:

- **Voltmeter Design**

The Voltmeter design is very simple and unique, as the main concept that has been focused on was the Voltage Divider Rule.

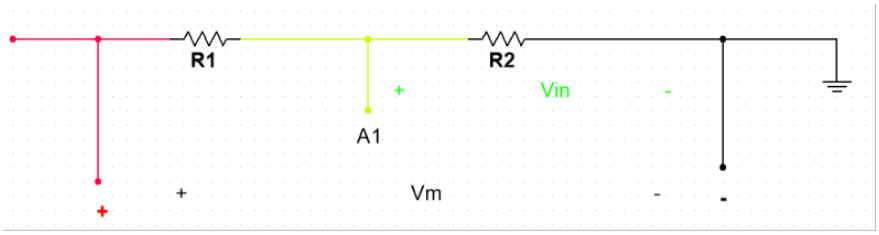

Figure 2: Voltmeter Design Schematic

As the positive and negative terminals are connected to the voltage source that should be measured, there a voltage drop takes place on R1. Then, A1, which is the analog pin of the Arduino reads the voltage drop across the second resistor. After that, the previous code converts the data which has been recorded by the analog pin and converted it into digital data in a quantity that represents the volt across the second resistor. Finally, the known value is the R1, R2, and the voltage drop across R2 -Vin-. Then, here the Voltage divider Rule takes Place. As known from the Rule:

$$V_{in} = V_m * \frac{R_2}{R_1 + R_2} \quad (1)$$

Then, it is easy to calculate Vm.

- **Using Voltage Divider Rule:**

But why did the voltage divider rule in the first place? Why would anyone not plug the input voltage into the analog pin and know the applied Voltage?

This process at first look does not have any problem, and anyone can do it, only if the applied voltage is less than or equal to 5 V. As the reference voltage of the Arduino is equal to 5 V, the board cannot handle any volt more than that and that could cause damage to the board. So, the 2 resistors in

series can drop the voltage to a certain level that the board can handle.

- **Choosing the Range of the 2 Resistors:**

Despite the use of the voltage divider resistors, there is a maximum input voltage that could be measured, and this maximum voltage has been chosen as 55 V, as this range gives the user a comfortablezone to measure daily, quick, practical things like a car battery. As a result, the values of the 2 resistors had been calculated as follows:

$$\frac{V_{in}}{V_m} = \frac{R_2}{R_1 + R_2} \tag{2}$$

$$\frac{5}{55} = \frac{R_2}{R_1 + R_2} \tag{3}$$

$$\frac{1}{11} = \frac{R_2}{R_1 + R_2} \tag{4}$$

$$R_1 = 10 * R_2 \tag{5}$$

Based on that result, R1=10 KΩ, R2= 1 KΩ, and these resistors provide the user the measured voltage up to 55 V.

• Ohmmeter Design

As same as the voltmeter, the main idea here is to take advantage of the Voltage divider Rule.When the unknown resistor is connected, there is an unknown voltage drop across it, and it is not measurable in that case; however, the voltage drops across the second resistor, which is known, can be measured by the Arduino A0 analog pin, and in that case, the known values are R1, V across R1 -Vout-, Vin -5V-.

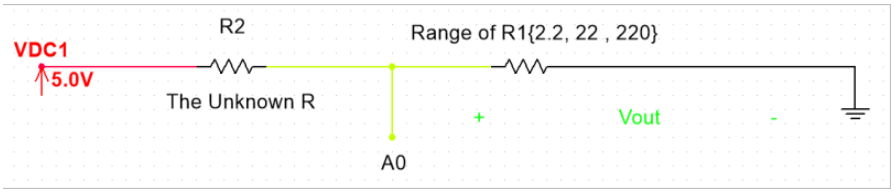

Figure 3: Ohmmeter Design Schematic

As the 2 resistors are in a series connection, then the same current is flowing across them in the same direction and quantity. Then:

$$I_1 = I_2 = I_t \tag{6}$$

$$\frac{V_{Runknown}}{R_{unknown/2}} = \frac{V_{out}}{R_1} \tag{7}$$

So,

$$R_2 = R_1 * \frac{V_{in} - V_{out}}{V_{out}} \tag{8}$$

- **The Various R1 Connection:**

Before this design has been made, there was a previous one that used only 1 resistor in R1 as a known resistor and the known voltage drop across it; however, if an unknown resistor had been measured, and this resistor is very larger so very smaller than R1, the error was fatal. So, the idea was to simulate the normal multimeter when used to measure a resistor value, and this idea was toset various R1 to each range of measured resistors.

- **The Ranges and the Values of R1:**

- R1= 2200 Ω → $R_{unknown}$ 0 → 2200 Ω

— $R1 = 22K\ \Omega \rightarrow R_{unknown}$ 2200 → 22.5K Ω

— $R1 = 220K\ \Omega \rightarrow R_{unknown}$ 22.5K → 230K Ω

6 Conclusion

The Arduino-based voltmeter and ohmmeter project provides a cost-effective and practical solution for measuring voltage and resistance in circuits. The project is easy to implement and offers a user- friendly interface with an LCD display and switch to select different ranges. The project demonstrateshow Arduino technology can be used in practical applications and provides a solid foundation for further exploration into the world of electronics and microcontrollers. Overall, the project offers an exciting opportunity to learn and experiment with Arduino and electrical circuits.