Alexandria University

Faculty of Engineering

Electrical Power & Machines Dep.

Measurements project report

Multirange ohmmeter

Measurements (EEP 112)

Prepared by:

Rowan Mohamed 21010548

Tasneem Alaa 21010396

Abdelrahman Gaber 21010730

Youssef Yasser 21011633

Amr Ossama 21010895

Amr Tolba 21010906

For Dr. Thanaa Elsayed Sharaf-Eldin

1-Introduction

Our project is an auto-ranging digital ohmmeter, designed to measure resistance of any range without requiring manual range selection. This device is useful because it eliminates the need for manual range selection, making measurements faster and more precise. Traditional AVO-meters require the user to select the range, which can be time-consuming and result in inaccurate measurements. Our ohmmeter solves this problem by automatically selecting the appropriate range. We chose to use an Arduino for this project due to its versatility and popularity as a microcontroller that allows for easy measurement of voltage and resistance. The wide online community of Arduino users provided resources and support for our project. Additionally, the Arduino's compatibility with various sensors and other components made it an ideal choice for our auto-ranging ohmmeter. By leveraging the Arduino's capabilities, we were able to write and upload code to control the device and display measurement results on an LCD screen. Our goal was to create an efficient, accurate, and user-friendly device that provides fast and reliable resistance measurements. The auto-ranging digital ohmmeter addresses the limitations of traditional AVO-meters and provides a more streamlined and effective solution for resistance measurement.

2-Objective

The main objective of our project was to develop an auto-ranging digital ohmmeter capable of measuring resistance of any range automatically. We aimed to create a device that would eliminate the need for manual range selection, thus making 4 resistance measurements faster and more accurate.

Additionally, we hoped to achieve a device that would be user-friendly and easy to operate. The significance of achieving this objective lies in the improved accuracy and efficiency of resistance measurements. Traditional AVO-meters require manual range selection, which can lead to inaccurate measurements due to human error. By automating the range selection process, our auto-ranging ohmmeter eliminates this source of error and delivers more reliable results. This is particularly important in fields such as electronics, where precise resistance measurements are critical for ensuring proper circuit operation. By achieving our objective of an auto-ranging digital ohmmeter, we provide a more effective solution for resistance measurement, which can lead to improved performance and reduced costs in various applications.

3-Components

1. Arduino Uno: This is a microcontroller board based on the ATmega328P. The Arduino board is responsible for controlling the operation of the ohmmeter, processing measurement data, and displaying the results on the LCD screen.

2. CM4051BE Multiplexer: This is an analog multiplexer used in the circuit to select the appropriate reference resistance for the measurement range selected. The CM4051BE multiplexer allows us to select one of the eight reference resistances used in the circuit. 5

3. Eight Reference Resistances: We used a total of eight reference resistors of varying values to create the range of resistances that can be measured by the ohmmeter. The values of these resistors are carefully calibrated to ensure accurate measurements.

4. Potentiometer: This is a variable resistor used to adjust the contrast of the LCD screen. The potentiometer allows the user to adjust the brightness of the display to suit their preferences.

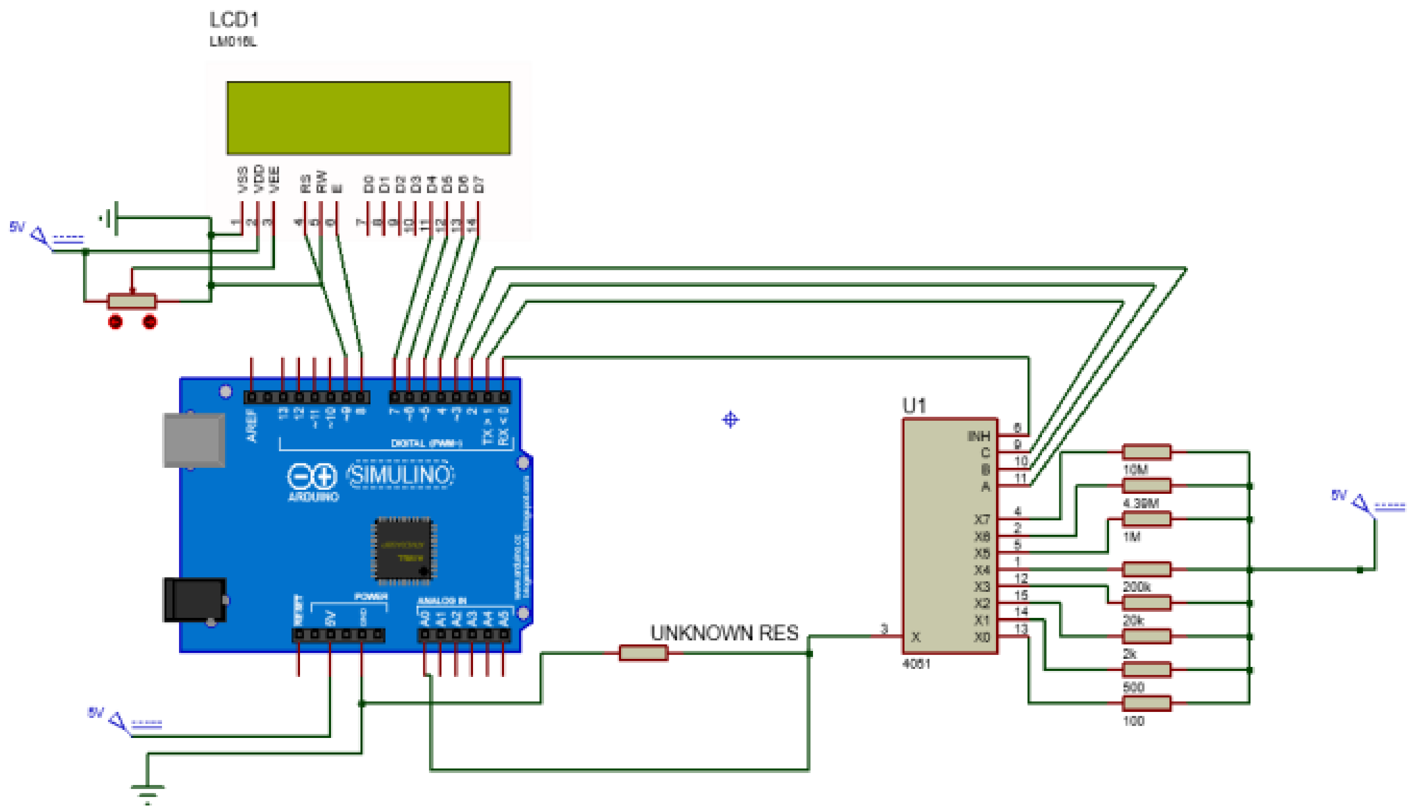
5. LCD Screen: The LCD screen is used to display the measurement results. The screen is connected to the Arduino board and displays the measured resistance in ohms.

6. Wires: These are used to connect the various components of the circuit together.

7. Power Supply: A power supply is used to provide power to the Arduino board and the other components in the circuit.

8.Bread board: to connect everything together

Here is a schematic diagram of the circuit:



Here is the code of the Arduino:

```
#include <LiquidCrystal.h>
#define NUM_REF_RESISTORS 8
#define NUM_SELECT_PINS   3
#define MAX_ANALOG_VALUE 973.068
#define SWITCH_RESISTANCE 30
// FOR CALIBRATION
float rRef[NUM_REF_RESISTORS] = {80, 195, 850, 1200, 18000, 92000, 900000, 9000000};
//x0 x1 x2 x3 x4 x5 x6 x7 FOR CALIBRATION

const byte rSelPins[NUM_SELECT_PINS] = {3, 2, 1};//A  B  C
const byte enableMux = 0; // 1 = no connection, 0 = one of eight signals connected
int screenWidth, screenHeight;

LiquidCrystal lcd(9, 8, 7, 6, 5, 4);

void setup()
{
  pinMode(enableMux, OUTPUT);
  digitalWrite(enableMux, HIGH);

  for (int i = 0; i < NUM_SELECT_PINS; i++)
  {
    pinMode(rSelPins[i], OUTPUT);
    digitalWrite(rSelPins[i], HIGH);
  }
 // Initialize the LCD display
  lcd.begin(16, 2);

}
// This function scales the resistor value

char ScaleToMetricUnits(float *prVal, char fStr[])
{
  char unit;
  if (*prVal < 1000)
  {
    unit = ' ';
  }
  else if (*prVal >= 1000 && *prVal < 1000000)
  {
    *prVal /= 1000;
    unit = 'K';
  }
  else if (*prVal >= 1000000 && *prVal < 1000000000)
  {
    *prVal /= 1000000;
    unit = 'M';
  }
  else
  {
    *prVal /= 1000000000;
    unit = 'G';
  }
  for (int k=2, s=10; k >= 0; k--, s*=10)
  {
    if ((int)(*prVal) / s == 0)
    {
      dtostrf(*prVal, 4, k, fStr);
      break;
    }
   }
   return unit;
 }

 void DisplayResultsOnLCDScreen(char unit, char fStr[])
 {
   lcd.clear();
   lcd.setCursor(0, 0);
   lcd.print("ArduinOhmmeter");
   lcd.setCursor(0, 1);
   if (unit != 0)
   {
     lcd.print(fStr);
     lcd.print(" ");
     lcd.print(unit);
   }
   else
  {
    lcd.print("- - -");
  }
}
void loop()
{
  int cOut;
  float delta, deltaBest1 = MAX_ANALOG_VALUE, deltaBest2 = MAX_ANALOG_VALUE;
  float rBest1 = -1, rBest2 = -1, rR, rX;
  char unit = 0, fStr[16];
  for (byte count = 0; count < NUM_REF_RESISTORS; count++)
  {
```

```
digitalWrite(rSelPins[0], count & 1); // C
digitalWrite(rSelPins[1], count & 2); // B
digitalWrite(rSelPins[2], count & 4); // A

digitalWrite(enableMux, LOW);
delay(count + 200);
cOut = analogRead(A0);
digitalWrite(enableMux, HIGH);
delay(NUM_REF_RESISTORS - count);
//valid digitized values
if (cOut < MAX_ANALOG_VALUE)
{
  rR = rRef[count] + SWITCH_RESISTANCE;
  rX = (rR * cOut) / (MAX_ANALOG_VALUE - cOut);
  delta = (MAX_ANALOG_VALUE / 2.0 - cOut);
  if (fabs(delta) < fabs(deltaBest1))
  {
      rBest2 = rBest1;
      deltaBest1 = delta;
      rBest1 = rX;
    }
    else if (fabs(deltaBest2) > fabs(delta))
    {
      deltaBest2 = delta;
      rBest2 = rX;
    }
  }
}

if (rBest1 >= 0 && rBest2 >= 0)
{
  if (deltaBest1 * deltaBest2 < 0)
  {
    rX = rBest1 - deltaBest1 * (rBest2 - rBest1) / (deltaBest2 - deltaBest1);
   }
   else
   {
     rX = rBest1;
   }
   unit = ScaleToMetricUnits(&rX, fStr);
 }
 DisplayResultsOnLCDScreen(unit, fStr);
 delay(250);
}
```