BIRZEIT UNIVERSITY

Faculty of Engineering & Technology Electrical & Computer Engineering Department

INSTRUMENTATION AND MEASUREMENT (ENEE4304)

Design Project

Prepared by:

Eman Asfour	1200206
Mohammad Emayya	1202133
Omar Bader	1201443

Instructor: Nasser Ismail

Section: 1

Date: 25/06/2024

1. Abstract

This project endeavors to deepen understanding and practical application of sensor theory by developing a system using an Arduino Mega. Initially, the project focuses on implementing a range of instrumentation and measurement functions, spanning from controlling relays to constructing a simple voltmeter and integrating an ultrasonic distance sensor. The overarching objective is to cultivate proficiency in Arduino programming while mastering the art of interfacing diverse sensors with the microcontroller board. In the preliminary phase, the system was meticulously designed, built, and optionally simulated using ORCAD & Tinker cad, providing invaluable hands-on experience. Through this process, a profound comprehension of Arduino's capabilities in relay control, voltage measurement, and ultrasonic distance detection was attained. Additionally, adeptness in LCD (16x2) interfacing was honed, enabling the real-time display of measurement results and system feedback. Further exploration delved into sensor technologies encompassing the DHT11 digital temperature sensor, thermistor temperature sensor, Wheatstone bridge, difference amplifier, and Light Dependent Resistor (LDR). A nuanced understanding of sensor principles and their pragmatic applications was cultivated by seamlessly integrating these sensors into the system architecture. This project served as a platform for theoretical exploration and facilitated practical learning in system design and implementation. Through meticulous experimentation and theoretical inquiry, invaluable insights were gleaned, fostering a solid foundation for future endeavors in the realm of instrumentation and measurement.

Table of Contents

List of Figures

List of tables

List of Equations

2. *Study and Comparison*

This study aims to evaluate and compare four widely used environmental monitoring sensors: ultrasonic sensors, LDR (Light Dependent Resistor) sensors, NTC thermistors, and DHT digital luminosity sensors. It explores the necessity of signal conditioning circuits and provides a detailed analysis of each sensor's characteristics, including accuracy, precision, and other relevant factors. By comparing these sensors, the research highlights their advantages and disadvantages, aiding in the selection of the most appropriate sensor for various environmental monitoring applications.

1) Identification of Sensors

i. DHT 11 Sensor

The DHT11 is a basic, low-cost digital sensor used to measure temperature and humidity. It consists of a capacitive humidity sensor and a thermistor, which together provide accurate and reliable data. The sensor outputs a digital signal on a single data pin, making it easy to interface with microcontrollers such as Arduino and Raspberry Pi.

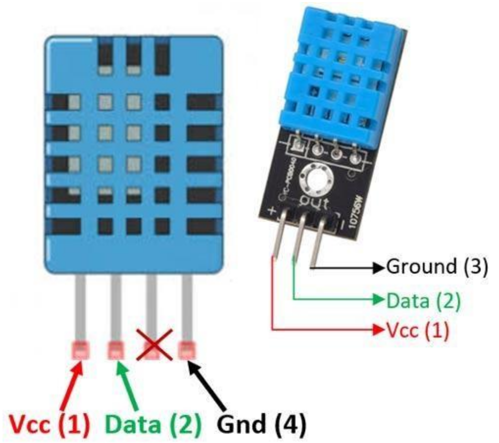

Figure 1: DHT Sensor [1]

ii. PTC Sensor 103

The PTC 103 thermistor is a reliable and economical sensor designed for temperature measurement. As the temperature increases, the resistance of the PTC 103 thermistor rises, allowing it to provide accurate and consistent data. This sensor generates an analog signal that can be easily integrated with microcontrollers like Arduino and Raspberry Pi through appropriate signal conditioning circuits, making it an ideal choice for various temperature-sensing applications.

Figure 2: PTC 103 Sensor [2]

iii. LDR Sensor

The LDR (Light Dependent Resistor) is an affordable, straightforward sensor used to gauge light intensity. It works by altering its resistance based on the amount of ambient light, with resistance decreasing as light levels increase, thus delivering dependable data on illumination. This sensor produces an analog signal, which can be seamlessly connected to microcontrollers like Arduino and Raspberry Pi using suitable signal conditioning circuits.

Figure 3: LDR Sensor

iv. Ultrasonic Sensor

An ultrasonic sensor measures distance by emitting high-frequency sound waves and detecting their echoes. This method offers precise distance measurements and is widely used in various applications, such as obstacle avoidance in robotics, liquid level detection, and proximity sensing in industrial automation systems. Its accuracy and reliability make it a popular choice for both consumer electronics and industrial equipment.

Figure 4: Ultrasonic sensor

2) Signal Conditioning Circuit:

i. PTC Sensor

The PTC sensor usually necessitates a straightforward voltage divider circuit, such as a Wheatstone bridge, to convert its resistance variations into a measurable voltage. This setup includes a series resistor and a voltage measurement component, enabling accurate detection of the resistance changes and, consequently, the temperature.

Table 1: Comparison between the DHT11 sensor and the PTC thermistor sensor

Properties	**DHT11 Sensor**	**PTC Thermistor**
Measurement	Temperature and Humidity	Temperature
Accuracy	Moderate	High
Precision	Moderate	High
Response Time	Slow	Fast
Signal Output	Digital	Analog
Interfacing	Easy (single digital pin)	Requires voltage divider
Cost	Low	Low
Temperature Range	Limited	Wide
Humidity Range	Wide	Not applicable
Signal Conditioning	On-chip	External voltage divider needed

ii. LDR Sensor

The LDR (Light Dependent Resistor) sensor typically requires a simple voltage divider circuit, such as a Wheatstone bridge, to convert its resistance changes into a measurable voltage. This configuration includes a series resistor and a voltage measurement component, which allows for precise detection of resistance variations in response to changing light levels. This setup enables accurate monitoring and measurement of light intensity through the LDR sensor.

3. Design and Calculations

This section aims to present the design and calculations for a Wheatstone bridge difference amplifier, tailored for two types of sensors: the thermistor temperature sensor and the LDR. Additionally, it will cover the integration of the ultrasonic sensor without the use of a Wheatstone bridge. Each part will detail the specific configuration, design considerations, and necessary calculations for effectively integrating these sensors into their respective circuits. First, the thermistor temperature sensor will be discussed, focusing on how its resistance changes with temperature and the corresponding calculations for the Wheatstone Bridge and difference amplifier to ensure accurate temperature readings. Next, the design for the LDR (Light Dependent Resistor) will be outlined. This part will explain how the LDR's resistance varies with light intensity and the calculations needed to convert these variations into measurable voltage changes through the Wheatstone bridge and difference amplifier. Finally, the ultrasonic sensor configuration will be presented. This section will cover the specific connection required to process the ultrasonic sensor's outputs, ensuring precise distance measurements without the use of a Wheatstone bridge. By breaking down the design and calculations for each sensor type, this section provides a comprehensive guide to integrating the thermistor and LDR sensors with a Wheatstone bridge difference amplifier, and the ultrasonic sensor directly into its appropriate circuit, facilitating accurate and reliable data acquisition.

I. The thermistor temperature

Data collection began by building a circuit with a DHT11 sensor and a PTC thermistor. The heat was applied to the sensors using a hair dryer, and the resistance was measured with an ohmmeter, with the results recorded in Table 2. The next step involved designing a Wheatstone bridge and various amplifier circuits, calibrated with a reference resistor of 8.68kΩ at a temperature of 27°C. The voltage bridge was measured at its maximum temperature, resulting in a value of 1.65. Based on this measurement, the resistances were designed starting from the minimum temperature where the voltage bridge registered 0. The variable 'b' in the linear equation was determined from this setup, and variable 'a' was identified at the maximum temperature where the voltage bridge reached 1.65. The signal conditioning, ranging from 0 to 5, guided the design of resistances based on the output voltage of the differential amplifier. Subsequently, the circuit was simulated using OrCAD, ensuring that the desired output voltage was achieved.

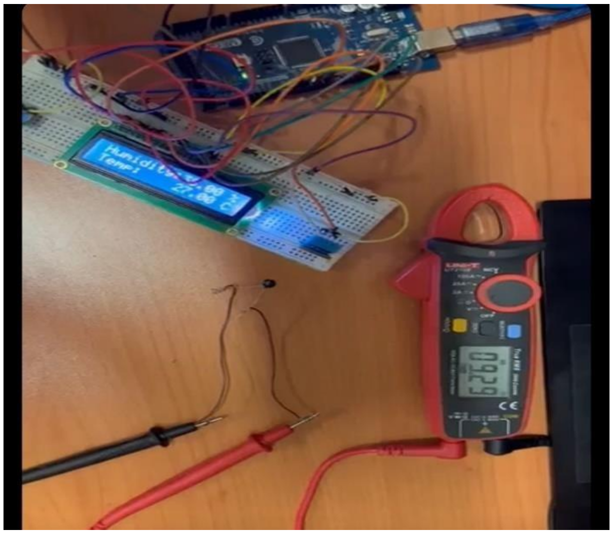

Figure 5: The part of the data

Table 2: Temperature Data

Temp	R(KΩ)
27	8.68
28	7.79
29	7.41
30	7.02
31	6.705
32	6.395
33	5.85
34	5.3
35	5.14
36	5.33
37	5.38
38	5.16
39	4.89

The data presented shows the relationship between temperature and resistance (R) for a sensor, with temperature values ranging from 27°C to 39°C and corresponding resistance values from 8.68 KΩ to 4.89 KΩ.

To ensure accuracy and precision in the measurements, each data point was carefully collected. The hair dryer was positioned 46 cm away from the sensors (DHT11 and PTC 103) at an angle of approximately 45 degrees. This setup was chosen to prevent the components from being directly affected by the heat and potentially getting damaged. The heat was applied for 5 minutes, which was deemed sufficient to observe consistent changes in resistance without causing any harm to the sensors. The distance, angle, and time were carefully considered to obtain precise and reliable data. An ohmmeter was used to read the changes in resistance accurately. Positioning the hair dryer at approximately 45 degrees helped to distribute the heat more evenly across both the PTC and DHT11 sensors, reducing the risk of overheating any specific component. This angle ensured that the heat was applied in a controlled manner, minimizing direct exposure and potential damage. This meticulous approach contributed to the high quality of the data collected, reflecting a decreasing trend in resistance as the temperature increased, consistent with the expected behavior of the sensor. The chosen time, distance, and angle were appropriate for achieving accurate and consistent measurements.

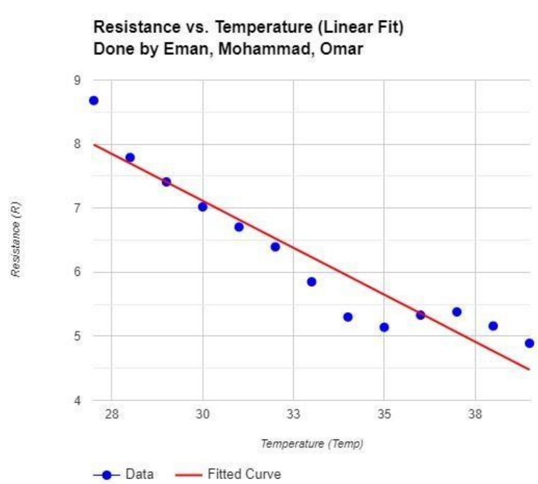

Figure 6: Resistance vs. Temperature (Linear Fit) [3]

To obtain the linear relationship between resistance (R) and temperature (Temp), an online Matplotlib compiler is used, and the Python code provided in Appendix 1 is executed to generate the linear curve from the data.

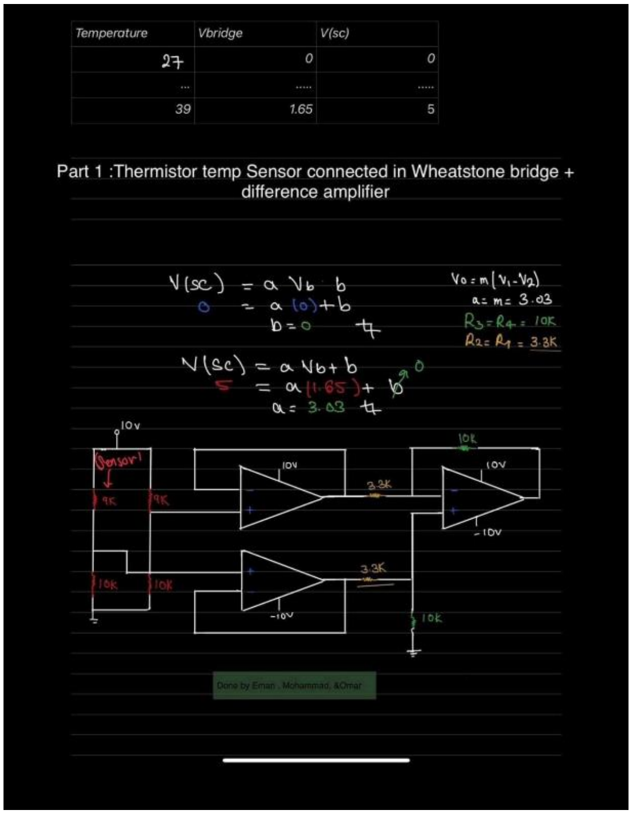

Figure 7: The Design and calculation

Equation 1: The Voltage of Signal Conditions

$$V_{SC} = a\,V_{bridge} + b\,0$$
$$= a\ 0 + b$$
$$b = 0$$

$$5 = a\,(1.65) + b$$

Equation 2: The Voltage of Difference amplifier

$$V_o = m\,(V_1 - V_2)$$
$$a = m = 3.03$$
$$R_4 = R_3 = mR = 10k$$
$$R_3 = R_4 = 3.3k$$
$$R_1 = R_2 = 10k$$

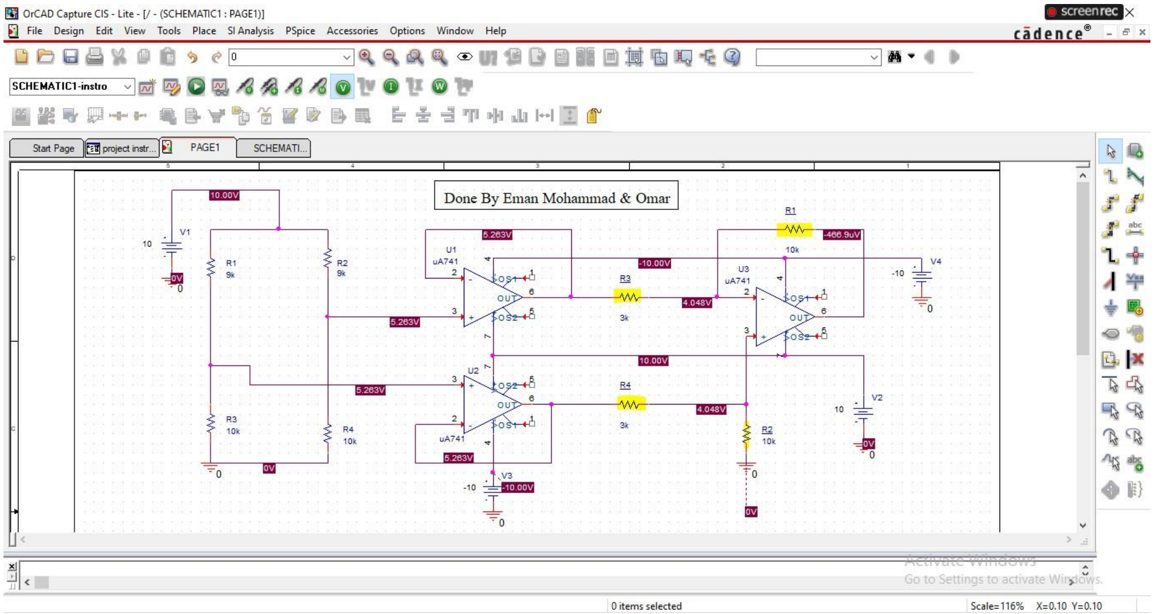

Figure 8: Simulation by ORCAD

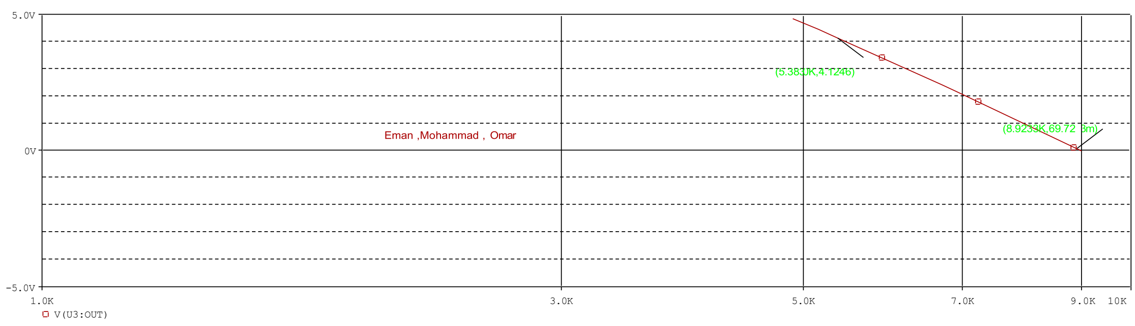

Figure 9: The Relationship between changes R (Sensor) with output voltage

In Figure 9, it can be seen that a linear relationship exists between the temperature changes and the output voltage. This indicates that when the temperature changes, a linear reaction is observed in the output voltage, as shown in Figure 6.

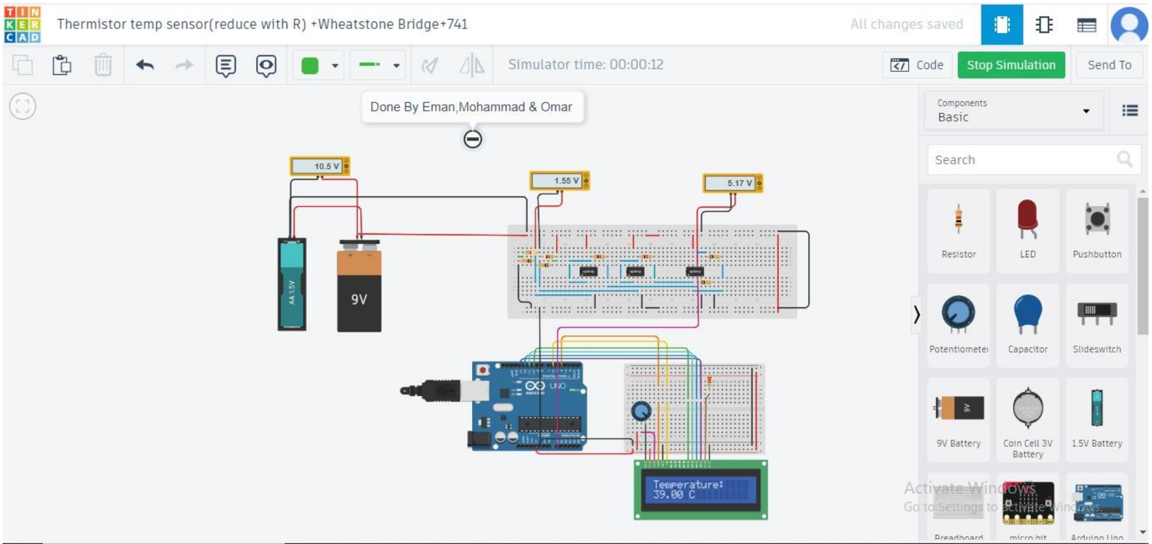

Figure 10: Simulation of the circuit by Tinker cad

Figure 10 shows that the output voltage is 5.17V, which is approximately what was designed when the sensor resistance (R) is at its minimum of 4.89kΩ. When the resistance is 8.9kΩ, the output voltage is 98mV, as depicted in Figure 11, which is approximately 0.

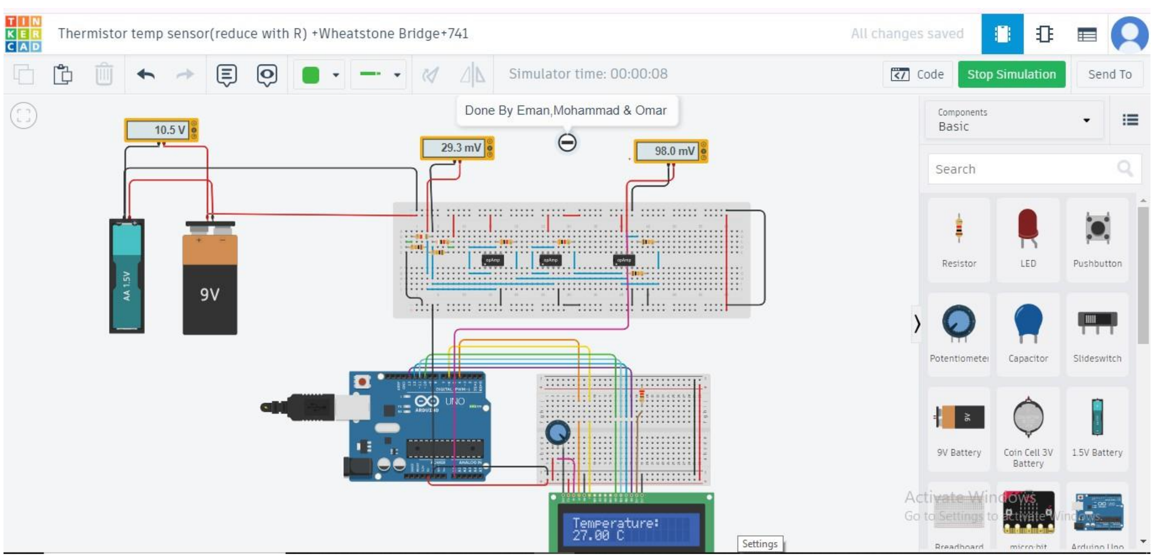

Figure 11: The Circuit when R (Sensor) =8.9k

II. The LDR is connected in a Wheatstone bridge + difference amplifier

In this step, the resistance is measured using the ohmmeter, and lux measurements are obtained using the Light Meter LM-3000 application [4]. The data presented in Table 3 is gathered by gradually increasing the light intensity using the same Light Meter LM-3000 device, establishing a linear relationship between these variables. Subsequently, in the Appendix 2 matplotlib online compiler, the data is plotted as a linear curve to visually depict the linear relationship, as shown in the figure.

Table 3: The data between Lux and Resistor

LUX$(\frac{lm^2}{m})$	R(KΩ)
3.4	93
20	18.2
30	4.7
60	6.48
90	3.68
120	5.53
150	2.6
200	2.23
250	1.9
300	1.77
350	1.61
400	1.48
450	1.37
500	1.306
550	1.24
600	1.18
620	1.16

The table3 presented shows a relationship between light intensity (lux) and resistance (R) for an LDR sensor. The measurements were taken at various lux levels, ranging from 3.4 to 620 lux, with corresponding resistance values from 93 KΩ to 1.16 KΩ. To ensure accuracy and reliability, each measurement was repeated three times. Additionally, different applications were used during the measurement process to obtain the best possible data. This thorough approach helped in minimizing errors and verifying the consistency of the results. The observed data demonstrates a decreasing trend in resistance as the lux levels increase, indicating the proper functioning of the LDR sensor.

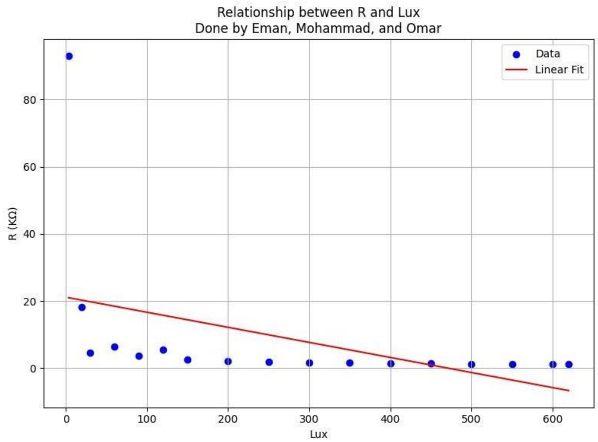

Figure 12: Linear curve fitting of Lux with R

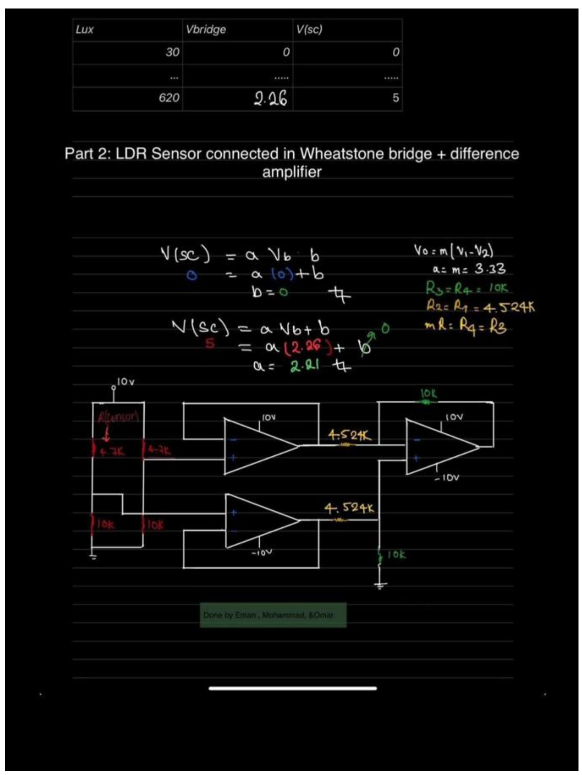

Figure 13: The design for part 2

The LDR was configured to be part of a Wheatstone bridge with a difference amplifier, using a reference resistance of 4.7k ohms at 30 lux, which is the maximum value within the design range of up to 30 lux (minimum Lux). A bridge voltage of 2.26V was measured by applying linear equation 4. Subsequently, maximum voltage signal conditioning was applied, and the constants 'a' and 'b' were determined by setting Vscto 0 and the bridge voltage V Bridge to 0. The gain of the amplifier, 'a', was calculated to be 2.21. Therefore, R4and R3were assumed to be 10k ohms, while R1and R2 were set to 4.524k ohms.

Equation 3: The Voltage of Signal Conditions

$$V_{SC} = a\,V_{bridge} + b$$

Equation 4: The Voltage when the Voltage Bridge equals zero

$$0 = a\,0 + b$$
$$b = 0$$

$$5 = a\,(2.26) + b$$

Equation 5: The Voltage of Difference amplifier

$$V_o = m\,(V_1 - V_2)$$

m represents the differential gain of the amplifier

$$a = m = 2.21$$
$$R_4 = R_3 = mR = 10k$$
$$R_3 = R_4 = 10k$$
$$R_1 = R_2 = 4.524k$$

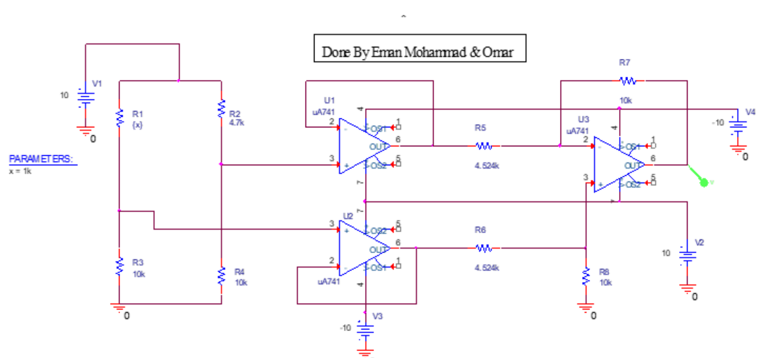

Figure 14: The Simulation circuit using parameter in ORCAD

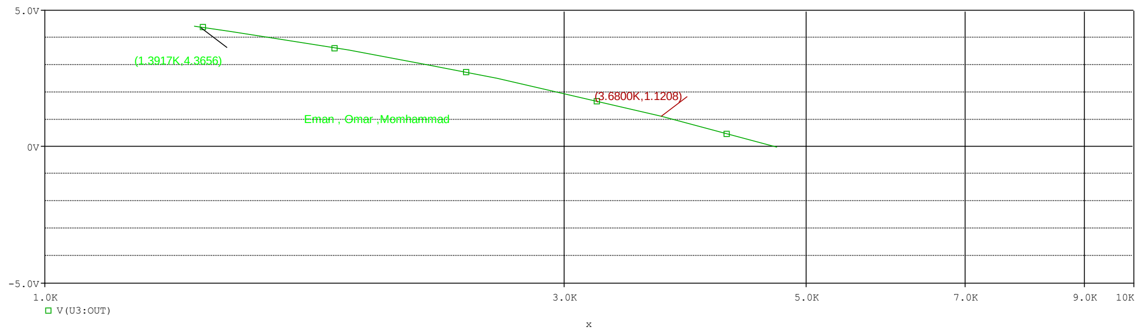

Figure 15: The Linear relationship between change R and output voltage

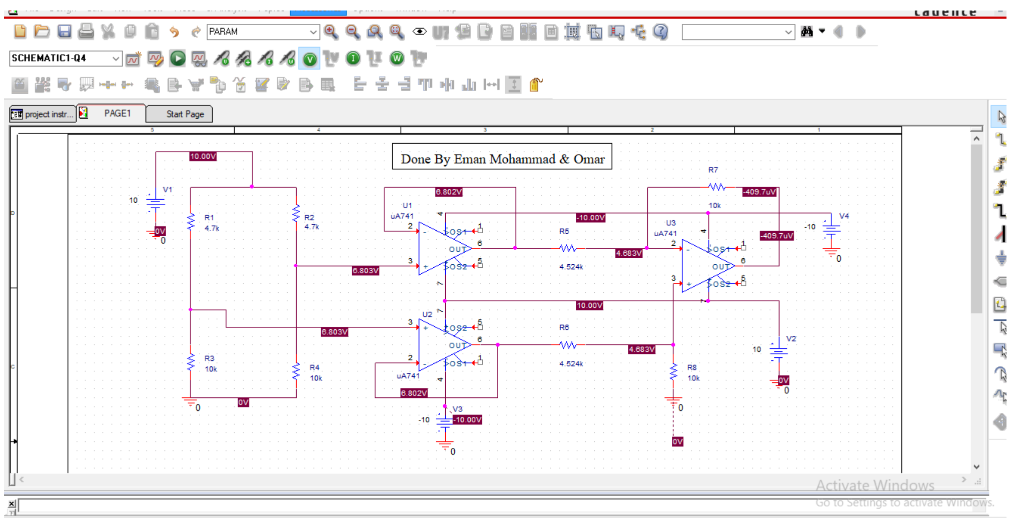

Figure 16: The LDR connected the Wheatstone bridge with a different amplifier with the max R

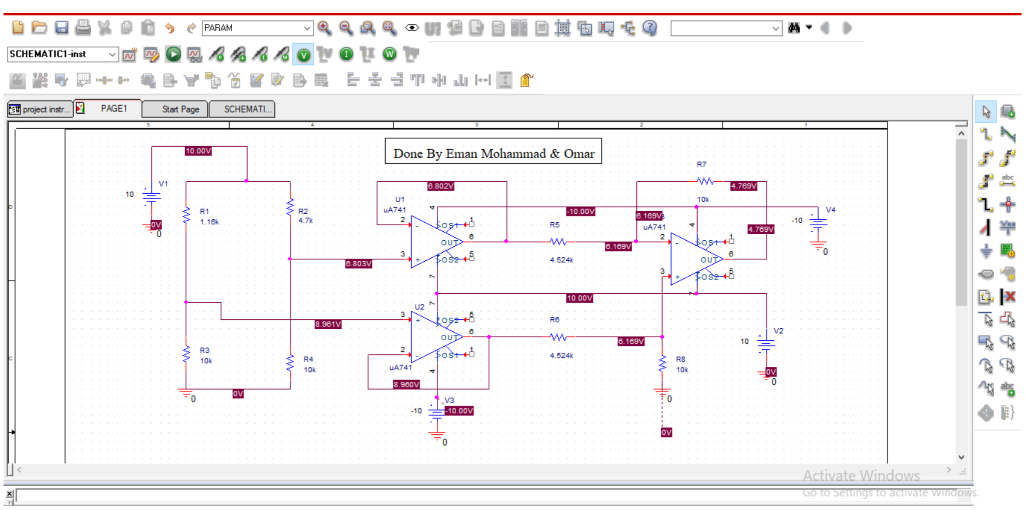

Figure 17: The LDR connected the Wheatstone bridge with a different amplifier with the min R

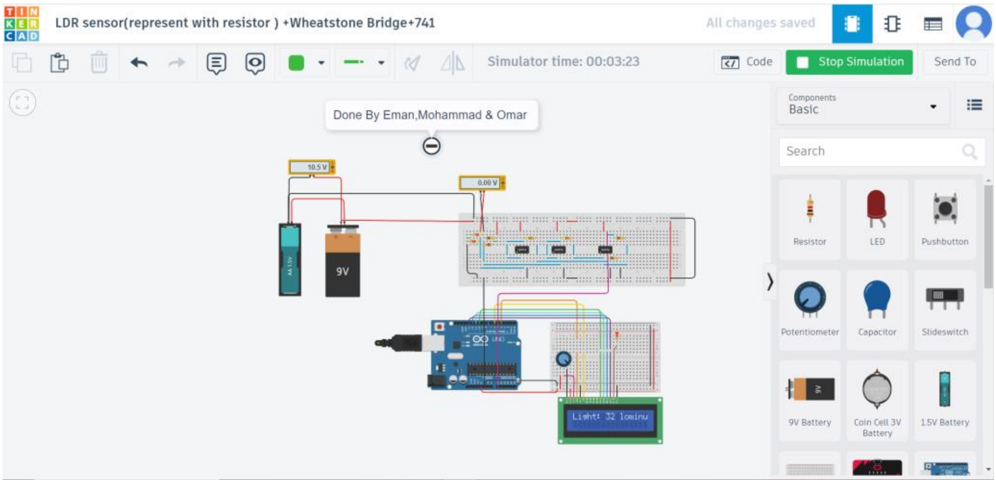

Figure 18: The LDR with Wheatstone bridge and difference amplifier on R =4.7k

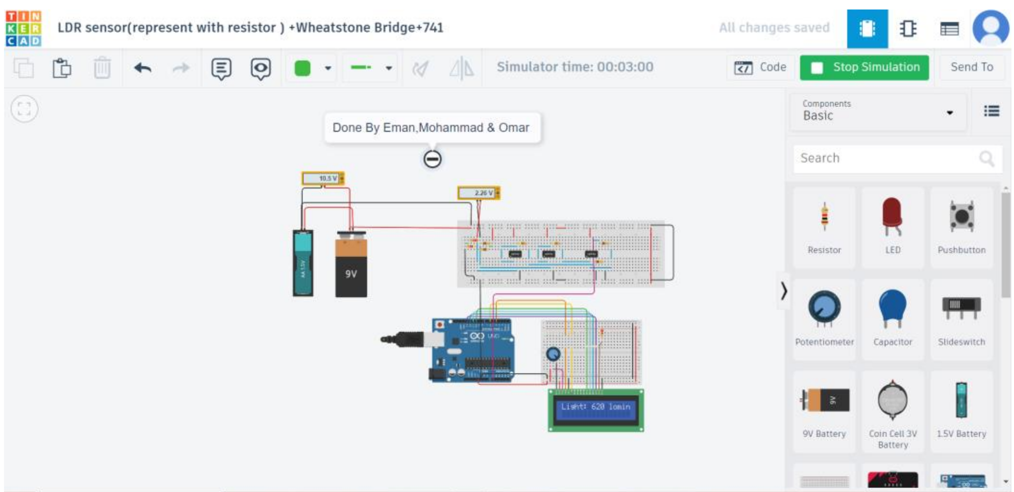

Figure 19: The LDR with Wheatstone bridge and difference amplifier on R =1.16k

As shown in Figure 12, a linear relationship was observed between lux and resistance. This indicates that changes in lux and resistance resulted in a linear correlation. Similarly, Figure 15 demonstrates that the output voltage maintained a linear relationship with changes in the LDR sensor or the corresponding resistor. Consequently, the desired outcome of achieving a linear relationship between resistance, lux, and output voltage was successfully obtained.

III. The LEDs are to be added for detection of (4 water levels) in a 1m height water tank

Four LEDs are to be incorporated to detect four water levels within a 1m height water tank. Sensors will be strategically positioned at different heights within the tank to measure the water levels accurately. The measured data from these sensors will be used to activate corresponding LEDs, visually indicating the current water level status. This system will allow for real-time monitoring of water levels without the need for direct observation, ensuring efficient management and prevention of overflow or underflow situations.

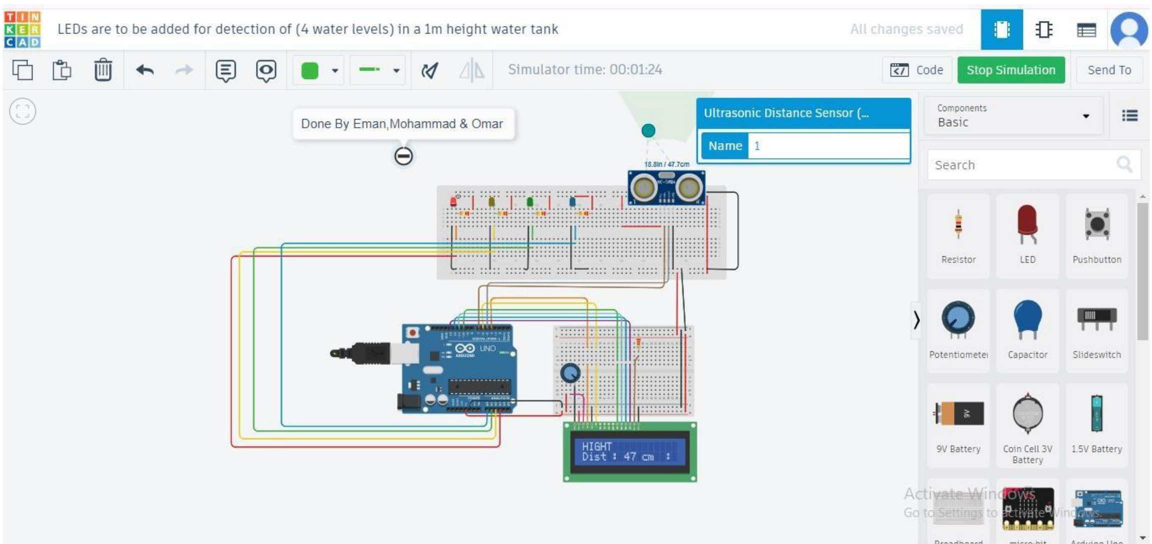

Figure 20: The Height when it is 50 cm

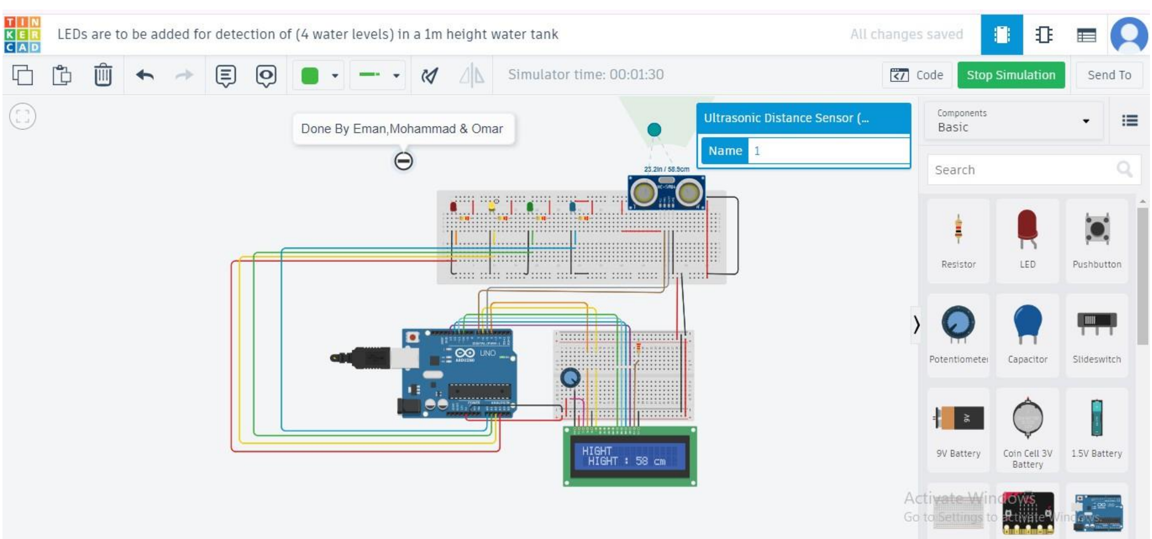

Figure 21: The Height when it is 70 cm

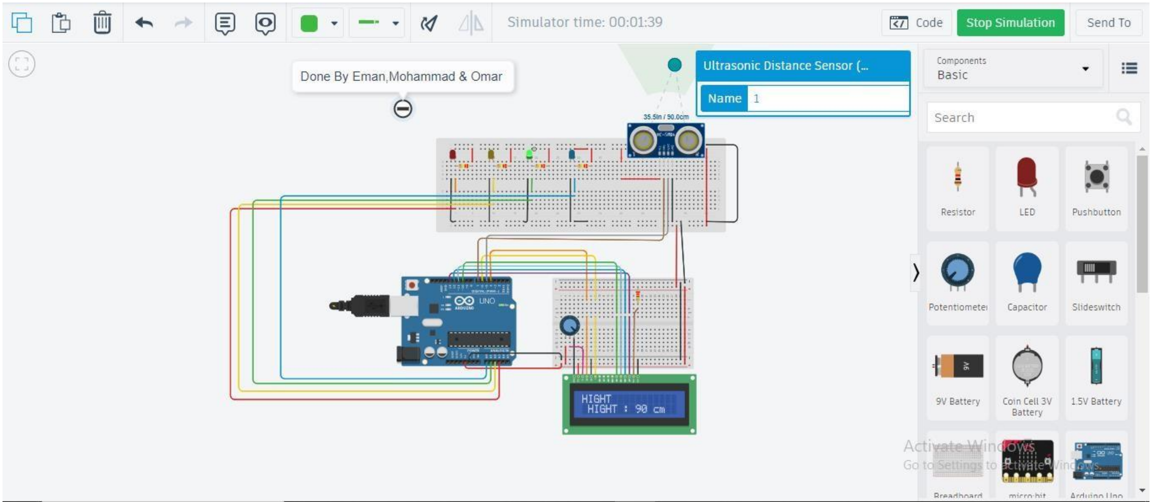

Figure 22: The Height when it is 90 cm

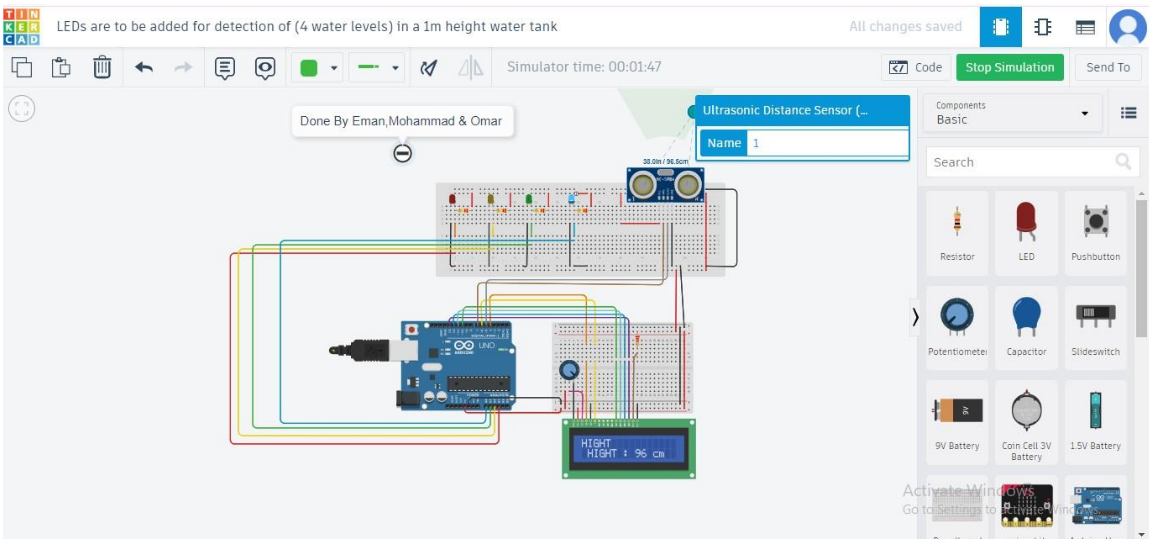

Figure 23: The Height when it is 100 cm

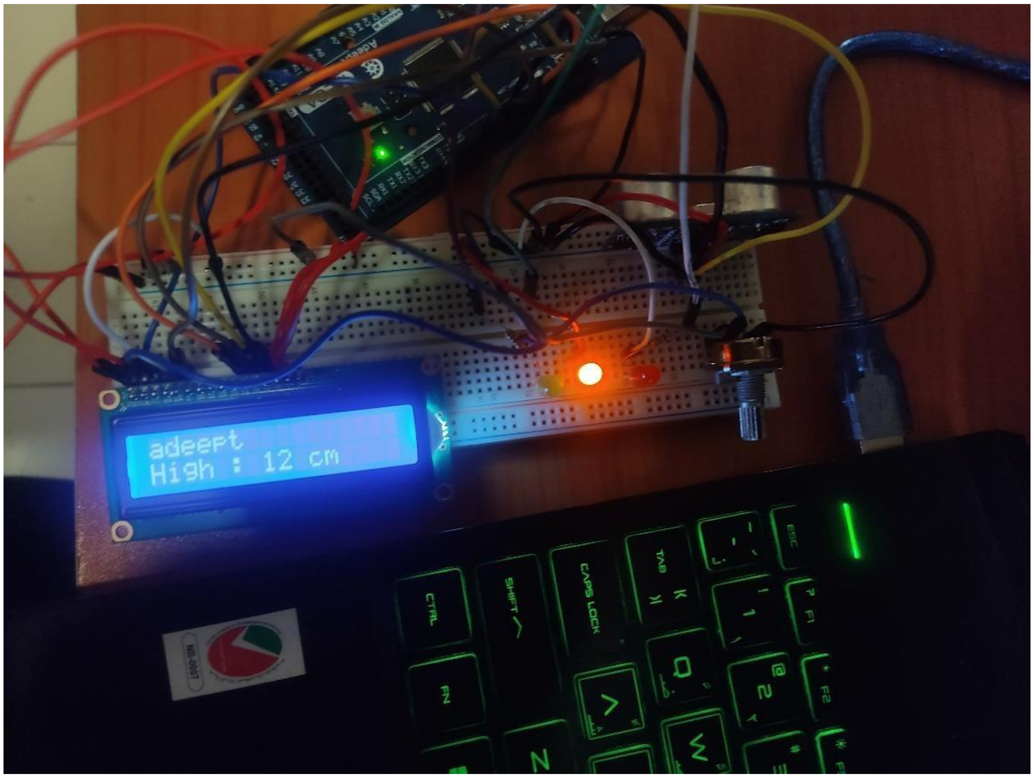

Figure 24 : The connection circuit

The connection and code for this part have been adjusted to be configured on Tinkercad, with minor modifications made.

4. Dissection

I. The thermistor temperature

Through meticulous repetition and careful measurement, the recorded data demonstrates a smooth and accurate trend. Each value is precise, and presented to two decimal places, ensuring reliability. Upon performing linear curve fitting, the data aligns perfectly into a linear relationship, affirming the robustness of the design. In practical implementation, the signal conditioning and bridge circuit perform effectively. However, upon running the code, discrepancies were noted as the LCD displayed varying temperature values, unlike the consistent readings observed in Tinkercad. Despite troubleshooting efforts confirming no errors, the functionality remains intact.

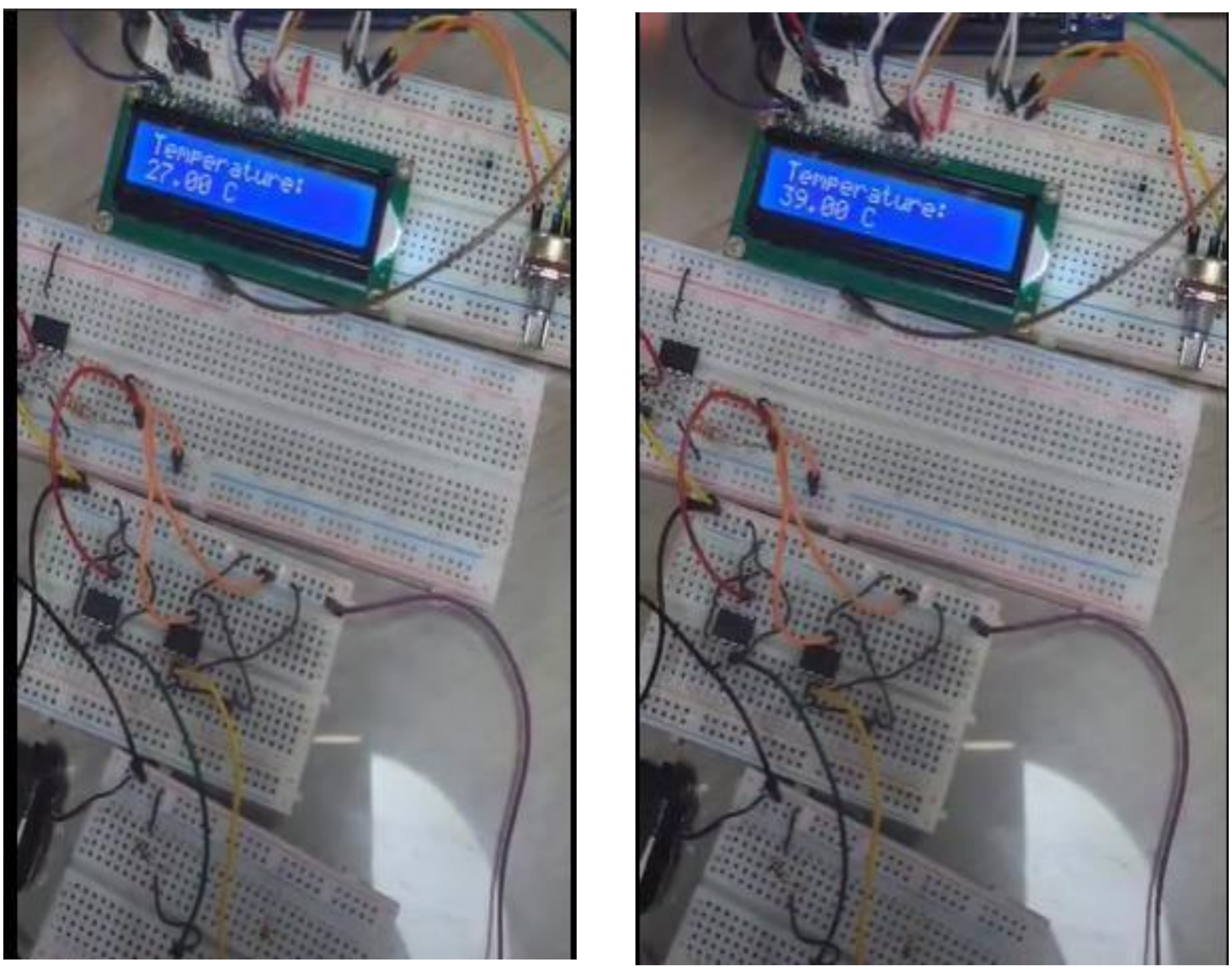

Figure 25: The connection circuit of part 1

II. The LDR is connected in a Wheatstone bridge + difference amplifier

The lux values in the dataset are presented as whole numbers, while the resistance values are meticulously recorded in two decimal places. This precision ensures the accuracy and reliability of the data, adhering closely to expected trends in sensor behavior. The coherence of the data points is evident as they follow a logical pattern, displaying a clear relationship between lux and resistance. Upon conducting linear curve fitting, the data conforms seamlessly to a linear model, validating the robustness of the experimental design and measurement methodology. During practical implementation, the signal conditioning and bridge circuit perform exceptionally well, demonstrating effective integration and functionality within the setup. However, upon executing the code, discrepancies emerged when the LCD displayed fluctuating temperature values, contrary to the consistent readings observed during the simulation in Tinkercad. Extensive troubleshooting efforts were undertaken to identify any potential errors, yet no issues were found that could explain the discrepancies. Despite these challenges, the overall functionality of the system remains intact, showcasing the resilience of the design under real-world conditions.

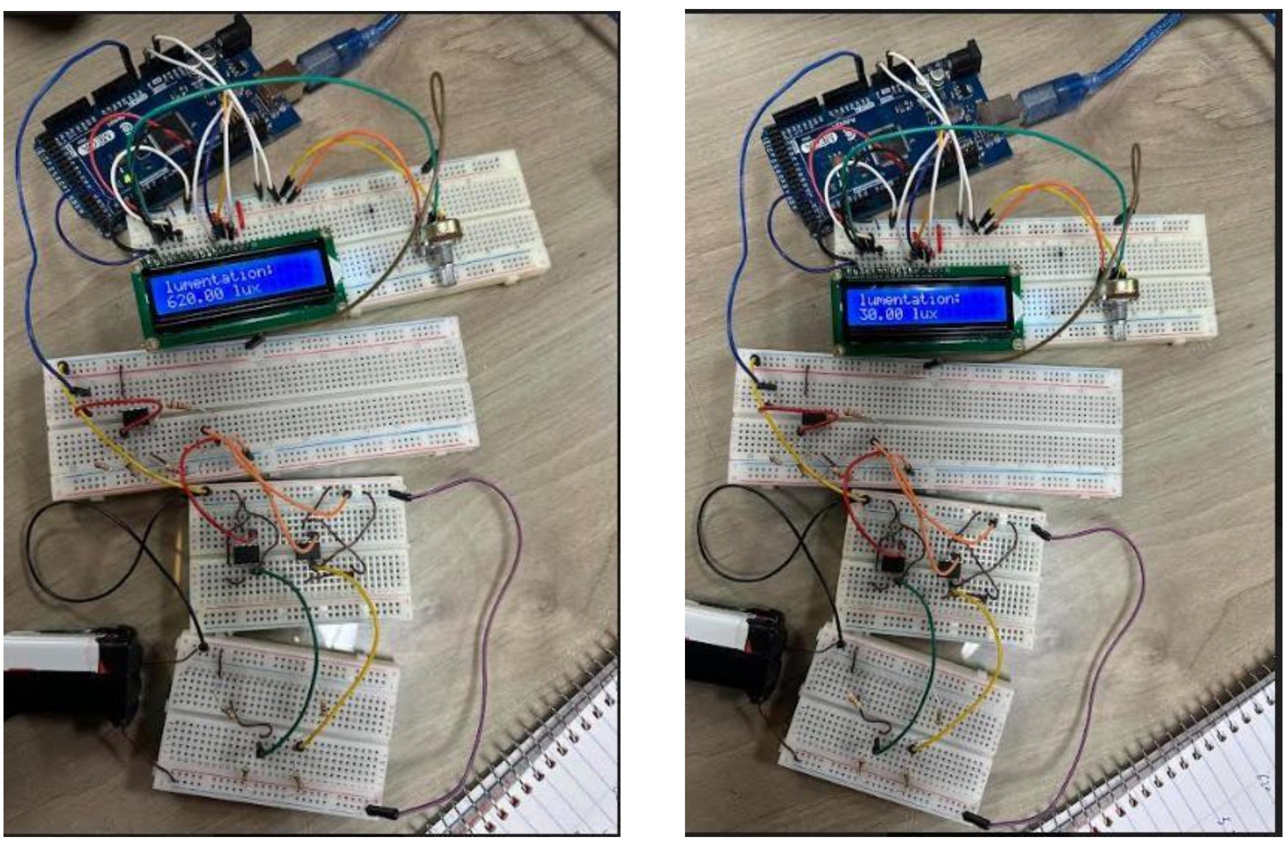

Figure 26: The circuit Connection of part 2

III. The LEDs are to be added for detection of (4 water levels) in a 1m height water tank

The connection setup for the ultrasonic sensor was meticulously executed, achieving the desired outcome of accurately sensing and displaying the height on the LCD screen, as outlined in the appendix code. The implementation proceeded smoothly, with the sensor reliably measuring height and promptly relaying the data to the LCD. This successful integration underscores the effectiveness of the chosen components and the robustness of the system design.

5. *Conclusion*

Throughout this project with the Arduino Mega, we explored sensor theory and practical applications comprehensively. From fundamental exercises in relay control, voltage measurement, and ultrasonic distance sensing, we progressed to integrating advanced sensors like the DHT11, thermistors, Wheatstone bridge, difference amplifiers, and LDRs. Each component's design and implementation were meticulously tested and optionally simulated using ORCAD & Tinker cad, ensuring theoretical understanding and practical proficiency in Arduino programming and sensor interfacing. Despite encountering challenges such as discrepancies in LCD temperature readings compared to Tinker cad simulations, rigorous troubleshooting validated the system's functionality and underscored the importance of systematic problem-solving in engineering projects. Moreover, the project emphasized collaborative teamwork, highlighting the significance of cooperation in achieving project objectives. Overall, this project has equipped us with valuable skills in instrumentation, measurement, and sensor technology, laying a solid foundation for future endeavors in engineering and innovation.

6. *Appendix*

1. Linear curve fitting of Resistance vs. Temperature

```
import matplotlib.pyplot as plt

# Data for temperature (Temp) and resistance (R)
temp = [27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39]
resistance = [8.68, 7.79, 7.41, 7.02, 6.705, 6.395, 5.85, 5.3, 5.14, 5.33, 5.38, 5.16, 4.89]

# Create a linear fit using least squares regression
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X=[[temp[i]] for i in range(len(temp))], y=resistance)

# Get the coefficients for the linear equation (y = mx + b)
slope = model.coef_[0]
intercept = model.intercept_

# Generate data points for the fitted line
fitted_temp = range(min(temp), max(temp) + 1)  # Ensure x-axis covers all data points
fitted_resistance = [slope * t + intercept for t in fitted_temp]

# Create the plot
plt.figure(figsize=(8, 6))  # Set plot size

# Plot the original data points
plt.scatter(temp, resistance, color='blue', label='Data')

# Plot the fitted line
plt.plot(fitted_temp, fitted_resistance, color='red', label='Fitted Curve')

# Add labels and title
plt.xlabel('Temperature (Temp)')
plt.ylabel('Resistance (R)')
plt.title('Resistance vs. Temperature (Linear Fit)')

# Add legend
plt.legend()

# Rotate x-axis labels for better readability
plt.xticks(rotation=45)

# Show the plot
plt.grid(True)
plt.tight_layout()
plt.show()
```

2. Linear curve fitting of Resistance vs. LUX

```
import matplotlib.pyplot as plt
import numpy as np
from scipy.optimize import curve_fit

# Data provided
lux = np.array([3.4, 20, 30, 60, 90, 120, 150, 200, 250, 300, 350, 400, 450, 500, 550, 600,
620])
r = np.array([93, 18.2, 4.7, 6.48, 3.68, 5.53, 2.6, 2.23, 1.9, 1.77, 1.61, 1.48, 1.37, 1.306, 1.24,
1.18, 1.16])
team = ['Eman', 'Mohammad', 'Omar']  # Assuming these are the contributors

# Linear curve fitting function
def linear_func(x, a, b):
    return a * x + b

# Perform linear curve fitting
popt, pcov = curve_fit(linear_func, lux, r)

# Generate points for the fitted line
lux_fit = np.linspace(min(lux), max(lux), 100)
r_fit = linear_func(lux_fit, *popt)

# Plotting
plt.figure(figsize=(8, 6))

# Scatter plot of data points
plt.scatter(lux, r, color='blue', label='Data')

# Plot fitted line
plt.plot(lux_fit, r_fit, color='red', label='Linear Fit')

# Customize plot labels and title
plt.xlabel('Lux')
plt.ylabel('R (KΩ)')
plt.title('Relationship between R and Lux\nDone by {}, {}, and {}'.format(*team))
plt.legend()

plt.grid(True)
plt.tight_layout()
plt.show()
```

3. Resistance vs. Temperature

```
#include <LiquidCrystal.h> // Include the LCD library
// Done By Eman , Mohammad , & Omar
// Initialize the LCD module
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

int sensorPin = A0; // Select the analog input pin for the temperature sensor
float referenceVoltage = 5.0; // Reference voltage for the analog input (max voltage)
float referenceTemperature = 27.0; // Reference temperature for the sensor calibration

void setup() {
  // Initialize the serial communication
  Serial.begin(9600);

  // Initialize the LCD module with the number of columns and rows
  lcd.begin(16, 2);
}

void loop() {
  // Read the temperature sensor on A0
  int sensorValue = analogRead(sensorPin);

  // Convert the analog reading to voltage
  float voltage = sensorValue * referenceVoltage / 1023.0;

  // calculate the temperature in Celsius
  float temperature;

  if (voltage < 0.5) {
    temperature = 27.0;
  } else if (voltage >= 5.0) {
    temperature = 39.0;
  } else {
    // Map the temperature based on the voltage
    temperature = ((voltage - 1) * (39 - 27) / (5 - 1)) + 27;
  }

  // Display the temperature on the serial monitor
  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.println(" C");

  // Display the temperature on the LCD
  lcd.clear(); // Clear the LCD display
  lcd.setCursor(0, 0); // Set the cursor to the first column and first row
  lcd.print("Temperature:"); // Print the text
  lcd.setCursor(0, 1); // Set the cursor to the first column and second row
```

```
  // Display the temperature on the LCD
  lcd.clear(); // Clear the LCD display
  lcd.setCursor(0, 0); // Set the cursor to the first column and first row
  lcd.print("Temperature:"); // Print the text
  lcd.setCursor(0, 1); // Set the cursor to the first column and second row
  lcd.print(temperature); // Print the temperature value
  lcd.print(" C"); // Print the temperature unit

  // Delay for a short period
  delay(1000);
}
```

4. Resistance vs. lux

```
#include <LiquidCrystal.h>// Done By Eman , Omar &Mohammad

// Define the pins for LDR and LCD
const int ldrPin = A0;
const int rs = 4;
const int en = 6;
const int d4 = 10;
const int d5 = 11;
const int d6 = 12;
const int d7 = 13;
const int lcdColumns = 16;
const int lcdRows = 2;

// Define the minimum and maximum luminus values
const int minLuminus = 30;
const int maxLuminus = 620;

// Initialize the LCD object
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {
  // Initialize the LCD
  lcd.begin(lcdColumns, lcdRows);
  lcd.print("Light: ");
}

void loop() {
  // Read the value from the LDR
  int ldrValue = analogRead(ldrPin);

  // Convert the LDR value to voltage (0 to 5V)
  float voltage = (ldrValue / 1023.0) * 5.0;
```

```
  // Convert the LDR value to voltage (0 to 5V)
  float voltage = (ldrValue / 1023.0) * 5.0;

  // Calculate the luminus value based on the voltage
  int luminus = (voltage / 5.0) * (maxLuminus - minLuminus) + minLuminus;

  // Print the luminus value on the LCD
  lcd.setCursor(7, 0);
  lcd.print(luminus);
  lcd.print(" lominus");

  delay(100);
}
```

5. **The LEDs are to be added for detection of (4 water levels) in a 1m height water tank**

```
#include <LiquidCrystal.h>// Done By Eman , Omar &Mohammad

// Define the pins for LDR and LCD
const int ldrPin = A0;
const int rs = 4;
const int en = 6;
const int d4 = 10;
const int d5 = 11;#include <LiquidCrystal.h>

const int pingPin = 5; // pin connected to Echo Pin in the ultrasonic distance sensor
const int trigPin = 7; // pin connected to trig Pin in the ultrasonic distance sensor
const int led1 = A0; // LED connected to pin A0
const int led2 = A1; // LED connected to pin A1
const int led3 = A2; // LED connected to pin A2

LiquidCrystal lcd(4, 6, 10, 11, 12, 13);

void setup() {
 pinMode(pingPin, INPUT); // Set Echo pin as input
 pinMode(trigPin, OUTPUT); // Set Trig pin as output

 pinMode(led1, OUTPUT); // Set LED 1 pin as output
 pinMode(led2, OUTPUT); // Set LED 2 pin as output
 pinMode(led3, OUTPUT); // Set LED 3 pin as output

 lcd.begin(16, 2); // Set up the LCD
 lcd.clear(); // Clear the LCD screen
 delay(1000); // Delay for 1 second
}
```

```
void loop() {
  int cm = ping(pingPin);

  lcd.setCursor(0, 0); // Set cursor to column 0, line 0
  lcd.print("adeept "); // Print message

  lcd.setCursor(0, 1); // Set cursor to column 0, line 1
  if (cm > 50) {  // Consider high distance above 50cm (adjust threshold
based on your needs)
    lcd.print(" HIGH");  // Print "HIGH" for distances above threshold
  } else {
    lcd.print("High : ");
    lcd.print(cm); // Print distance
    lcd.print(" cm "); // Print unit
  }

  // Control LEDs based on distance
  digitalWrite(led1, LOW); // Turn off all LEDs initially
  digitalWrite(led2, LOW);
  digitalWrite(led3, LOW);

  if (cm <= 10) {
    digitalWrite(led1, HIGH); // Turn on LED 1 for distance <= 10cm
  } else if (cm <= 50) {
    digitalWrite(led2, HIGH); // Turn on LED 2 for distance <= 50cm
  } else {
    digitalWrite(led3, HIGH); // Turn on LED 3 for distance > 50cm
  }

  delay(500); // Delay for half a second
}

int ping(int pingPin) {
  long duration, cm;

  pinMode(trigPin, OUTPUT);
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(5);
  digitalWrite(trigPin, LOW);
  pinMode(pingPin, INPUT);
  duration = pulseIn(pingPin, HIGH);
  cm = microsecondsToCentimeters(duration);
  return cm;
}

long microsecondsToCentimeters(long microseconds) {
  return microseconds / 29 / 2;
}
```

```
long microsecondsToCentimeters(long
microseconds) {
  return microseconds / 29 / 2;
}

const int d6 = 12;
const int d7 = 13;
const int lcdColumns = 16;
const int lcdRows = 2;

// Define the minimum and maximum luminus
values
const int minLuminus = 30;
const int maxLuminus = 620;

// Initialize the LCD object
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

void setup() {
    // Initialize the LCD
    lcd.begin(lcdColumns, lcdRows);
    lcd.print("Light: ");
}

void loop() {
    // Read the value from the LDR
    int ldrValue = analogRead(ldrPin);

    // Convert the LDR value to voltage (0 to 5V)
    float voltage = (ldrValue / 1023.0) * 5.0;

    // Calculate the luminus value based on the
voltage
    int luminus = (voltage / 5.0) * (maxLuminus -
minLuminus) + minLuminus;

    // Print the luminus value on the LCD
    lcd.setCursor(7, 0);
    lcd.print(luminus);
    lcd.print(" lominus");

    delay(100);
}
```

6. References

[1] [Online]. Available: https://components101.com/sensors/dht11-temperature-sensor. [Accessed 24 5 2024].

[2] [Online]. Available: https://www.filotronix.com/variable-resistor/267-ntc-103-thermistor.html. [Accessed 25 5 2024].

[3] [Online]. Available: https://www.tutorialspoint.com/execute_matplotlib_online.php. [Accessed 26 5 2024].

[4] [Online]. Available: https://apps.apple.com/us/app/light-meter-lm-3000/id1554264761. [Accessed 20 5 2024].