Hardware Assignment

Tejas U, Sree Vardhan M

AI25BTECH11038, AI25BTECH11020

1 AIM:

The aim is to design and implement a digital thermometer that measures temperature using a **PT-100 RTD**, signal through an Arduino microcontroller, and display the temperature on a 16X2 LCD which is calculated using Linear regression.

2 COMPONENTS:

PT-100 RTD , Arduino microcontroller , Jumper wires , Breadboard , Potentiometer and 1 standard resistor.

3 PROCEDURE:

CIRCUIT ASSEMBLY:

- Place the standard resistor and PT-100 on the breadboard to form a simple voltage divider.
- Connect the junction between them to Arduino's A0 pin.
- Connect +5 V and GND from Arduino to the two ends of the divider.
- The 16 * 2 LCD can be connected using the regular 4 - bit connections. (**Note:** since we have only one pot, we need to ignore contrast setting.)

  DATA COLLECTION (CALIBRATION):
- Print the voltage value to the LCD display.
- Place the PT-100 in different known temperature environments such as:
- Ice water (0 °C), Room temperature (25 °C), Warm water (50 °C), Hot water (75–100 °C).
- For each environment, record:
  a) The reference temperature (from thermometer).
  b) The measured voltage from the LCD display.

4 CIRCUIT DIAGRAM:

The connections involved:
Arduino to LCD display:
(RS, EN, D4, D5, D6, D7, VSS, VCC, VEE, A, K) = (Digital Pin 12, Digital Pin 11, Digital Pin 5, Digital Pin 4, Digital Pin 3, Digital Pin 2, GND, 5V, GND, 5V, GND).
Standard Resistor (R1):
a) Connect one end to +5V.
b) Connect the other end to one of the "same color" wires of the PT100 and to A0.
Potentiometer:
a) Connect one outer pin to the other "same color" wire of the PT100.
b) Connect the other outer pin to GND.
c) Connect the center pin to A0 (along with R1 and the PT100's third wire).
PT100 Connections:
a) Two wires of the same color (e.g., red) connect to one side of the PT100.
b) The third wire (e.g., white) connects to the other side of the PT100.
c) Connect one of the "same color" wires to +5V (through R1).
d) Connect the other "same color" wire to one end of the potentiometer.
e) Connect the "third wire" to the junction of R1 and the potentiometer, and also to A0.

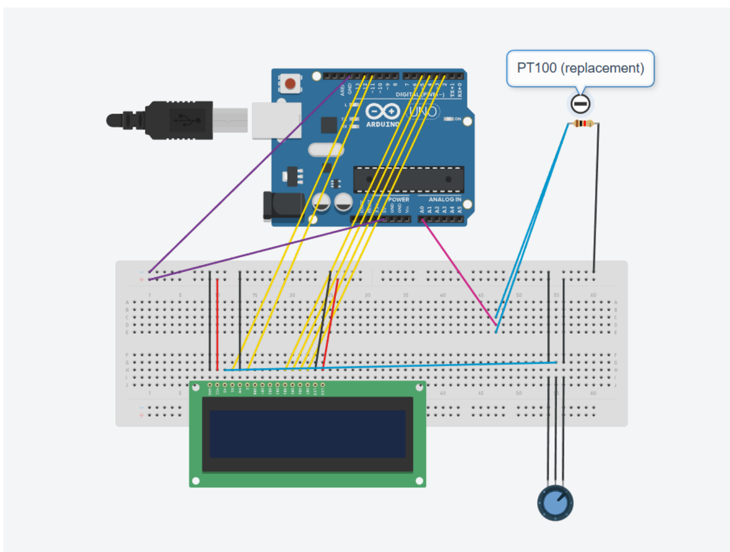

Figure 1: CIRCUIT DIAGRAM

5 USING LINEAR REGRESSION:

Linear Regression Model for Calibration The main formula used in calculating the temperature (using the PT100) is

$$T \approx a_0 + [a_1 \cdot V] + [a_2 \cdot V^2] \tag{1}$$

Now, based on the values of Voltages obtained and Temperatures of the surroundings, we are supposed to approximate the values of a_0, a_1 and a_2. The equation 1 can be expressed in the form of a matrix equation

$$A \cdot \mathbf{x} = b \tag{2}$$

where $A = \begin{bmatrix} 1 & T_1 & T_1^2 \\ 1 & T_2 & T_2^2 \\ \vdots & \vdots & \vdots \\ 1 & T_n & T_n^2 \end{bmatrix}$, $x = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix}$ and $b = \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_n \end{bmatrix}$

This equation can be solved using the **method of least squares** as mentioned in the question.
Since we need minimum error in finding the values of the constants in equation 1, The residual $||Ax - b||^2$ should be minimum which means, $(Ax - b) \cdot (Ax - b)^T$ should be minimum on x.
Differentiating the equation and setting the gradient to zero,

$$f(\mathbf{x}) = ||A\mathbf{x} - b||^2 = (A\mathbf{x} - b)^T \cdot (A\mathbf{x} - b). \tag{3}$$

$$d[f(\mathbf{x})]/dx = 0 \tag{4}$$

$$2A^T \cdot A \cdot \mathbf{x} - 2A^T \cdot b = 0 \tag{5}$$

$$A^T \cdot A \cdot \mathbf{x} = A^T \cdot b \tag{6}$$

From here,

$$\mathbf{x} = (A^T A)^{-1} \cdot A^T \cdot b \tag{7}$$

We can use this equation 7 to evaluate **x** in python for us.

6 EXPLANATION OF CODES USED:

The python code used for calculating the matrix x

```
import numpy as np

temperatures = []
voltages = []

print("Enter 25 temperatures (T):")
for i in range(25):
    temperatures.append(float(input()))

print("Enter 25 corresponding voltages (V):")
for i in range(25):
    voltages.append(float(input()))

A = np.zeros((25, 3))
for i in range(25):
    A[i][0] = 1
    A[i][1] = voltages[i]
    A[i][2] = voltages[i]**2

b = np.array(temperatures)

p = A.T

q = p @ A

r = np.linalg.inv(q)

x = r @ p @ b

print("\nConstants:")
print("(for T = a0 + a1 * V + a2 * V^2)")
print(f"a0 = {x[0]}")
print(f"a1 = {x[1]}")
print(f"a2 = {x[2]}")
```

Figure 2: Least Square Method

The fairly simple code above defines a matrix A, x and b according to the notation above and prints the final approx. values of the constants.

It is just the creation of the matrices A and b which are later used in the evaluation of x.

Before having the values of the constants

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  lcd.begin(16, 2);
  lcd.print("Voltage Reader only");
  delay(1000);
}

void loop() {
  int sensorValue = analogRead(A0);
  float voltage = sensorValue * (5.0 / 1023);

  lcd.clear();
  lcd.print("V:");
  lcd.print(voltage, 2);
  lcd.print(" V");

  delay(500);
}
```

Figure 3: Initial code in the Audrino IDE

It's a simple code in C, which displays the voltage read by the "**A0** analog input".
The multiplication factor of **5/1023.0** is used as the voltage reading from analog input is multiplied by 5/1023.0 to convert the raw **ADC** (Analog-to-Digital Converter) value to a real-world voltage level in volts.
As the The Arduino Uno ADC converts input voltages between 0V and 5V into digital values from 0 to 1023 (a 10-bit range).

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  lcd.begin(16, 2);
  lcd.print("Voltage and Temperature");
  delay(1000);
}

void loop() {
  int sensorValue = analogRead(A0);
  float voltage = sensorValue * (5.0 / 1023);
  float a0 = ;
  float a1 = ;
  float a2 = ;

  temperature = a0 + (a1 * (voltage)) + (a2 * (voltage) * (voltage));

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("V:");
  lcd.print(voltage, 2);
  lcd.print(" V");
  lcd.setCursor(0, 1);
  lcd.print("T:");
  lcd.print(temperature, 2);
  lcd.print(" celcius");
  delay(800);
}
```

Figure 4: After gaining the values

After gaining the constants

It is just the previous code but with the addition of temperature variable which is calculated from the equation 1.

7 ERROR ANALYSIS:

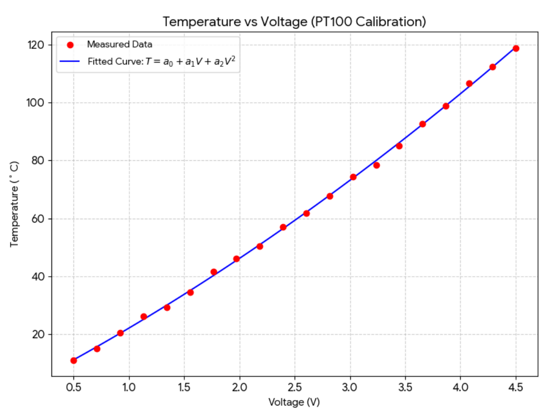

Figure 5: Plot from matplotlib

The Mean Absolute Error (MAE): $\approx 1.07C$.
(Note : It is the average of all the errors obtained from the datapoints that we have.)

Also, another useful method of checking accuracy is coefficient**oefficient of determination** which is defined as

$$R^2 = 1 - \frac{\text{Sum of squared errors}}{\text{Total sum of squares}} \quad (8)$$

For our PT100 Thermometer model $R^2 \approx 0.975$.

8 CONCLUSION:

Our Digital Thermometer made using a PT100 and a voltage divider is accurate and uses linear regression with the method of least squares to calculate the temperature as a function of voltage.
The final equation obtained by us is:

$$T = (-7635.8621618788) + (5168.0892433551 \cdot V) - (863.67569212067 \cdot V^2) \quad (9)$$