Hardware Project

EE25BTECH11007- Aniket and EE25BTECH11021-Achyuta

1 Objective

Design and implement a digital thermometer that measures temperature using a PT100 resistance temperature detector (RTD), processes the signal with an Arduino microcontroller, and displays the temperature on a 16×2 LCD.

2 Materials Required

- LCD
- Potentiometer
- Arduino Uno
- Cable
- Jumper Wires
- 100 Ω resistor
- Breadboard

3 Background

This project builds a digital thermometer using a PT100 RTD sensor, an Arduino Uno for data acquisition and processing, and a 16×2 LCD for display. A simple voltage divider converts the temperature dependent resistance into a measurable voltage. We calibrate the system with least squares polynomial fitting (NumPy/Matplotlib), then embed those coefficients in the Arduino sketch to compute temperature in real time.

4 Procedure

- Connections
  Assemble all circuit components as shown in the diagram on the next page. Use the electric kettle to raise the $PT-100's$ temperature, then take measurements following the subsequent steps
- Arduino Code

  - **Wiring:** +5V – R_{FIXED} (100 Ω) – PT100 – GND; tap goes to `A0`.
  - **`setup()`:**
    - `Serial.begin(9600)` starts serial output.
    - `lcd.begin(16,2)` initializes the 16×2 LCD.
  - **`loop()`:**
    1) Read 10-bit ADC on `A0`: `analogRead(sensorPin)` $\in [0, 1023]$.
    2) Convert count to node voltage:

$$V = \frac{V_{\text{REF}}}{1023} \times \text{ADC}, \qquad V_{\text{REF}} = 5.0 \text{ V}.$$

3) Get PT100 resistance from the divider:

$$R_{\mathrm{PT}} = R_{\mathrm{FIXED}} \cdot \frac{V}{V_{\mathrm{REF}} - V}.$$

4) Convert resistance to temperature (linear PT100):

$$T\,(^\circ\mathrm{C}) = \frac{R_{\mathrm{PT}} - 100\ \Omega}{0.385\ \Omega/^\circ\mathrm{C}}.$$

5) Clear and update the LCD.
6) Print ADC, V, R_{PT}, and T to the Serial Monitor.
7) `delay(1000)` and repeat.

5 Theory

For the PT100 we use the Callendar–Van Dusen relation on the 0–100 °C range.

we collect calibration data by measuring both quantities over a range of known temperatures. Let the measured data points be (T_i, V_i), $i = 1, 2, \ldots, n$.

$$V(T) = n_0 + n_1 T + n_2 T^2$$

$$\mathbf{C} = \mathbf{X}^\top \mathbf{n}$$

where

$$\mathbf{X} = \begin{pmatrix} 1 & T_1 & T_1^2 \\ 1 & T_2 & T_2^2 \\ \vdots & \vdots & \vdots \\ 1 & T_n & T_n^2 \end{pmatrix}, \qquad \mathbf{C} = \begin{pmatrix} V_1 \\ V_2 \\ \vdots \\ V_n \end{pmatrix}.$$

and $\mathbf{n} = \begin{pmatrix} n_0 \\ n_1 \\ n_2 \end{pmatrix}$ are the unknown coefficients. Using the *least squares*

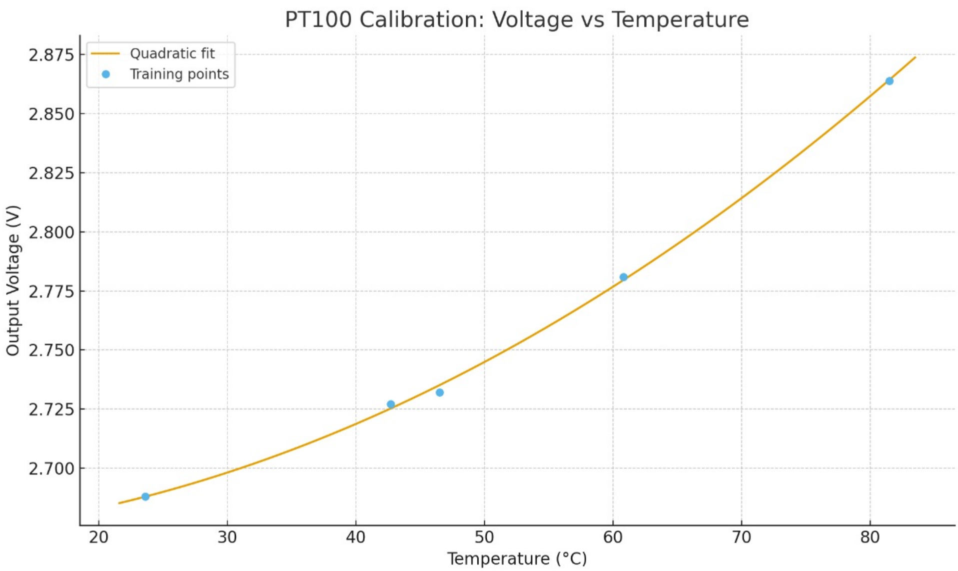
$$\mathbf{n} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{C}.$$

$$\mathbf{n} = \begin{pmatrix} -8757.7551517 \\ 6030.94728525 \\ -1028.18535177 \end{pmatrix}.$$

6 Training Model

TABLE I: PT100 measurements

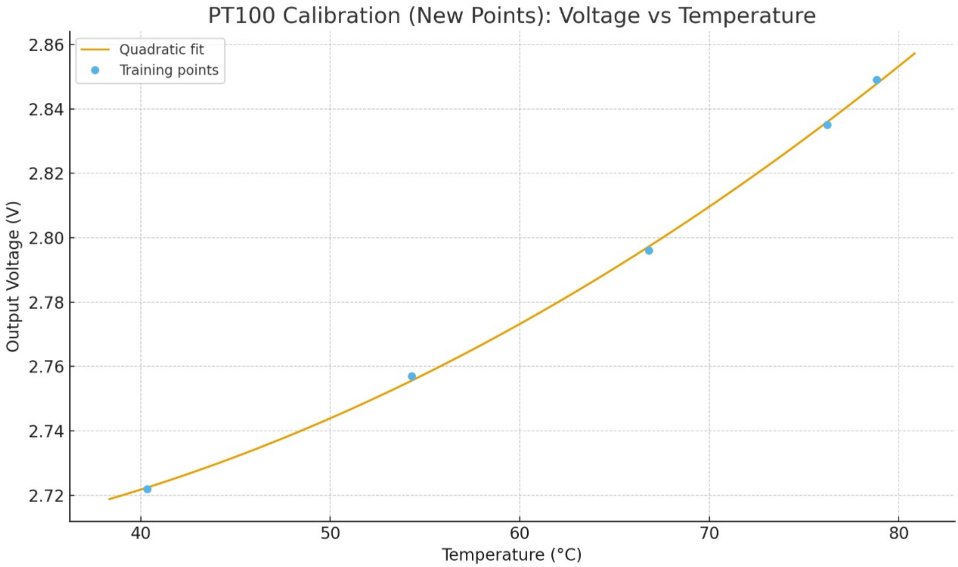
Temperature T (°C)	Voltage V (V)
23.6	2.688
42.7	2.727
46.5	2.732
60.8	2.781
81.5	2.864



7 Validation Dataset

TABLE II: Validation Dataset

Temperature (°C)	Voltage (V)
54.78	2.757
66.48	2.796
76.01	2.835
79.09	2.849
40.13	2.722



8 Result and Error Analysis

TABLE III: Result and Error Analysis

Actual Temperature	Calculated Temperature	Voltage	Error
41.4	40.36	2.722	0.74
54.6	54.3	2.757	0.30
66.1	66.8	2.796	0.70
74.4	76.01	2.835	1.61
79.1	78.83	2.849	0.27

Average Error : 0.724

9 Conclusion

We built a PT100–Arduino thermometer using a simple 100 Ω voltage divider, sampled with the Uno's 10-bit ADC, and converted to °C via a least-squares calibration $T(V) = a_0 + a_1 V + a_2 V^2$. Across the calibrated span (room to ~ 80 °C), the system delivered stable real-time readings with a typical error of ± 1–2 °C, confirmed against reference baths and a physics cross-check (Callendar–Van Dusen).