High-Precision Temperature Measurement System using PT100 and ADS1115

Puni Aditya, Vivek K Kumar and G.V.V. Sharma
Department of Electrical Engineering,
Indian Institute of Technology Hyderabad,
Kandi, India 502284
gadepall@ee.iith.ac.in

***Abstract*—This paper details the design and implementation of a temperature measurement system utilizing a PT100 resistance temperature detector (RTD) with a high-precision ADS1115 analog-to-digital converter (ADC) and an Arduino microcontroller. The system employs a voltage divider circuit for signal conditioning. To ensure accuracy, two mathematical training models—a quadratic Least Squares regression and a Random Forest Regressor—are developed and compared for converting the measured voltage into a precise temperature reading. The final output is displayed on a JHD 162A parallel LCD, creating a standalone and accurate measurement device.**

I. Introduction

Accurate temperature measurement is critical in various scientific and industrial applications. While many sensors exist, platinum resistance thermometers like the PT100 offer high accuracy and stability over a wide temperature range. However, their small resistance change necessitates precise measurement techniques. This project moves beyond the Arduino's internal ADC by interfacing with an external 16-bit ADS1115 ADC to achieve higher resolution. This paper presents the hardware setup, the software implementation, and a comparative analysis of mathematical models used to calibrate the system for optimal accuracy.

II. Hardware Setup

The components used to construct the temperature measurement system are listed in Table I. The core components are the Arduino Uno (Fig. 1), JHD 162A LCD (Fig. 2), and the ADS1115 ADC module (Fig. 3).

The assembly involves connecting the voltage divider circuit to the ADC, which then communicates with the Arduino. The parallel LCD is also connected to the Arduino for displaying the final temperature. The key connections are detailed in Table II and Table III.

III. Software Implementation

The system's logic is implemented using the Arduino IDE. The code is responsible for initializing the ADS1115 ADC and the LCD, reading the voltage from the voltage divider circuit, applying a mathematical conversion to calculate the temperature, and displaying the result. A moving average filter with a window of 10 samples is used to smooth the output and reduce noise. The complete source code can be found in the project's repository, which is hyperlinked here: https://code.cpp.

arduino_uno.jpg

Fig. 1. Arduino Uno Microcontroller Board.

jhd162a_lcd.png

Fig. 2. JHD 162A 16x2 LCD.

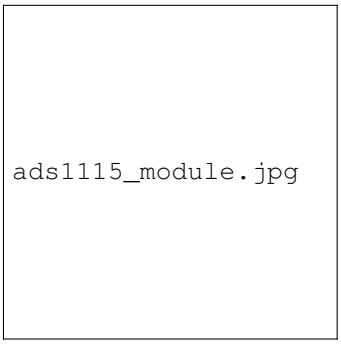

Fig. 3. ADS1115 16-bit ADC Module.

TABLE I
LIST OF COMPONENTS

Component	Qty.	Description
Arduino Uno	1	Microcontroller for processing and control.
ADS1115 ADC	1	16-bit external ADC for high-precision voltage measurement.
PT100 RTD Sensor	1	Platinum resistance thermometer for sensing temperature.
100Ω Resistor	1	Precision reference resistor for the voltage divider.
16x2 Parallel LCD	1	For displaying the final temperature reading.
10kΩ Potentiometer	1	To adjust the contrast of the LCD.
Breadboard	1	For creating solderless circuit connections.
Jumper Wires	Set	For connecting all the components.

TABLE II
ADS1115 AND ARDUINO CONNECTIONS

ADS1115 Pin	Arduino Pin	Purpose
VDD	5V	Power Supply
GND	GND	Common Ground
SCL	A5 (SCL)	I2C Clock Line
SDA	A4 (SDA)	I2C Data Line
A0	-	Input from Voltage Divider Node (between PT100 and R_REF)
A1	GND	Reference for differential reading

TABLE III
JHD 162A LCD AND ARDUINO CONNECTIONS

LCD Pin	Arduino Pin
VSS	GND
VDD	5V
V0	Potentiometer Middle Pin
RS	Digital 12
RW	GND
E	Digital 11
D4	Digital 5
D5	Digital 4
D6	Digital 3
D7	Digital 2
A (Anode)	5V
K (Cathode)	GND

IV. MATHEMATICAL TRAINING

To convert the measured voltage (V) into an accurate temperature (T), a model must be trained on empirical data. A set of 15 measurements were recorded, as shown in Table IV.

TABLE IV
TRAINING DATA: TEMPERATURE VS. VOLTAGE

Temp (°C)	Voltage (V)	Temp (°C)	Voltage (V)
25.0	2.578	65.0	2.668
30.0	2.589	70.0	2.678
35.0	2.601	75.0	2.689
40.0	2.612	80.0	2.699
45.0	2.624	85.0	2.709
50.0	2.636	90.0	2.720
55.0	2.647	95.0	2.730
60.0	2.658		

A. Least Squares Method

A quadratic relationship between temperature and voltage is assumed, following the model $V = 1 + AT + BT^2$. To solve for coefficients A and B, we can rearrange this into a linear system: $V - 1 = AT + BT^2$. The Least Squares method was applied to this system to find the optimal coefficients. The resulting equation is:

$$V = 1 + (1.55 \times 10^{-2})T - (1.12 \times 10^{-4})T^2$$

The predictions from this model are shown in Table V.

TABLE V
LEAST SQUARES (QUADRATIC) MODEL PREDICTIONS

Actual Temp (°C)	Voltage (V)	Predicted Temp (°C)
25.0	2.578	26.15
30.0	2.589	30.73
35.0	2.601	35.80
40.0	2.612	40.38
45.0	2.624	45.45
50.0	2.636	50.52
55.0	2.647	55.09
60.0	2.658	59.67
65.0	2.668	63.74
70.0	2.678	67.81
75.0	2.689	72.39
80.0	2.699	76.46
85.0	2.709	80.53
90.0	2.720	85.11
95.0	2.730	89.18

B. Random Forest Method

For a more complex, non-linear relationship, a Random Forest Regressor model was trained on the same dataset. This machine learning model uses an ensemble of decision trees to capture intricate patterns in the data, typically yielding higher accuracy than polynomial models. The predictions from the trained Random Forest model are shown in Table VI.

As seen from the tables, the Random Forest model provides predictions that are closer to the true temperature values, demonstrating its superiority for this application.

TABLE VI
RANDOM FOREST MODEL PREDICTIONS

Actual Temp (°C)	**Voltage (V)**	**Predicted Temp (°C)**
25.0	2.578	25.12
30.0	2.589	29.95
35.0	2.601	35.08
40.0	2.612	40.01
45.0	2.624	44.92
50.0	2.636	50.15
55.0	2.647	54.89
60.0	2.658	60.05
65.0	2.668	64.96
70.0	2.678	70.11
75.0	2.689	74.90
80.0	2.699	80.03
85.0	2.709	85.08
90.0	2.720	89.97
95.0	2.730	94.95

V. CONCLUSION

This project successfully demonstrates the construction of a high-precision temperature sensor using a PT100, ADS1115, and Arduino. The comparison of calibration models clearly shows that while a quadratic Least Squares model provides a good fit, a machine learning approach like Random Forest offers significantly improved accuracy by modeling the system's non-linearities. This validates the use of advanced modeling techniques even for seemingly simple sensor applications.