Digital Thermometer

1st U.V.K Sudeepthi
School of AI
Amrita Vishwa Vidyapeetham
Coimbatore, India

2nd P. Varshitha
School of AI
Amrita Vishwa Vidyapeetham
Coimbatore, India

3rd M.V Mukesh
School of AI
Amrita Vishwa Vidyapeetham
Coimbatore, India

***Abstract*—Temperature monitoring and measurement are important functions in many areas of application such as healthcare, environmental science, industrial automation control, and in daily home applications. Conventional analog thermometers are easy, but they can be imprecise, slow, and uninterfacable. To complement these shortcomings, this paper details the design and construction of a microcontroller digital thermometer based on the LM35 temperature sensor and Arduino Uno system. The LM35 sensor is chosen based on its accuracy, linear output, and simple interfacing capability, providing a linearly proportional voltage output with respect to Celsius temperature. System architecture includes the LM35 sensor being interfaced with the Arduino Uno's analog input. The analog voltage signal is digitized using the onboard Analog-to-Digital Converter (ADC), and the temperature in degrees Celsius is determined by application of a pre-determined, temperature-scaled formula. The resulting temperature data is then displayed on a 16x2 LCD screen for offline observation or sent as a serial communication to a web interface for real-time monitoring. This web interface option shows how the device can be extended into Internet of Things (IoT) applications, facilitating remote access and data logging.**

I. Introduction

Temperature measurement is an important function in a broad range of applications ranging from medical diagnostics, food safety, and environmental monitoring to industrial process control and home automation. The capability to measure and control temperature can improve safety, maximize efficiency, and facilitate health and well-being. Historically, mercury or alcohol thermometers have been used for such purposes; however, these analog instruments have a number of drawbacks. They tend to be fragile, are not capable of interfacing with digital systems, offer no data storage or logging feature, and pose environmental and health hazards as a result of the toxic chemicals involved. Consequently, the need for precise, reliable, and easily integratable temperature sensing systems has given rise to the use and development of digital thermometers.Recent developments in embedded systems and sensor technology have made it possible to develop low-cost, portable, and effective digital thermometers. Microcontrollers like the Arduino Uno have transformed prototyping by providing a general-purpose platform for integrating sensors, processing data in real time, and interfacing output. This work suggests a digital thermometer system based on the LM35 temperature sensor that gives a linear voltage output proportional to the temperature in Celsius. The LM35 is characterized by its high precision, low self-heating, and simple calibration, and thus it is well suited for incorporation into microcontroller-based systems.In the suggested arrangement, the LM35 is brought to the Arduino Uno's analog input, from where the analog signal is sensed and processed with the help of its built-in Analog-to-Digital Converter (ADC). Processed digital output is then available on a 16x2 LCD display to be read locally, or as an option to be sent via serial communication over the web interface for real-time reading and remote access. The ease of use of the system makes it suitable for incorporation in educational institutions to teach interfacing of sensors and embedded programming, whereas its precision and upgradability render it applicable to practical implementation within homes, clinics, and industry units.The goal of this project is to create a trustworthy, affordable digital thermometer that proves real-time temperature acquisition and output, with possible IoT-based add-ons. In addition to the fundamental implementation, this paper also discusses system calibration methods, measurement error sources, and potential enhancements like wireless communication, mobile app integration, and cloud-based logging. With accuracy, usability, and flexibility combined, the suggested digital thermometer system is a practical solution for a range of temperature monitoring applications in both theoretical and practical contexts.

II. Methodology

A. Component Selection

LM35 Temperature Sensor: The LM35 was used because it has a high accuracy and linear output with a voltage directly proportional to the temperature in Celsius. It is inexpensive, has a low self-heating rate, and is easy to interface with microcontrollers such as Arduino.

Arduino Uno Microcontroller: The Arduino Uno was chosen due to its common usage, low cost, simplicity of programming, and provision of analog input pins, which are required for sensing data from the LM35 sensor.

16x2 LCD Display: This was utilized for graphical output, allowing real-time temperature readings to be displayed. It is cheap and simple to integrate with the Arduino for basic output purposes.

Potentiometer: A potentiometer controlled the contrast of the LCD screen so that visibility could be varied based on different lighting conditions.

B. Circuit Design and Connections

Connecting LM35 to Arduino: The LM35 temperature sensor was connected to the analog input pin (A0) of the Arduino

Uno. The VCC and GND pins of the LM35 were connected to the corresponding pins on the Arduino for power.

LCD Display Wiring: The LCD was connected to the digital pins of the Arduino Uno. For ease of wiring, the LCD was configured to work in 4-bit mode, requiring only four data lines and minimizing pin usage on the Arduino.

Power Supply: The Arduino Uno was powered through the USB connection to a computer or an external power adapter, while the LM35 was powered by the 5V pin of the Arduino.

C. *Analog-to-Digital Conversion*

Temperature Calculation Formula: The LM35 gives an analog output, which the Arduino reads through its internal 10-bit ADC. The ADC value is between 0 and 1023, corresponding to a voltage range of 0 to 5V.

The Arduino code converts the analog ADC value to a temperature in Celsius using the following formula:

$$T(C) = \left(\frac{\text{Analog Reading} \times 5.0}{1024} \right) \div 0.01$$

This formula takes the analog reading, converts it into voltage, and then calculates the temperature in Celsius, as the LM35 sensor outputs 10mV per degree Celsius.

D. *Programming and Calibration*

Arduino Sketch Programming: An Arduino sketch (code) was developed to read the analog sensor values, convert them to a temperature reading, and output the temperature on the 16x2 LCD. The code utilizes simple functions to set up the LCD and continuously update the temperature shown on the LCD.

Calibration: Calibration of the system was done by comparing the reading from the digital thermometer against that of a known accurate instrument (e.g., mercury thermometer). When differences were encountered, the conversion formula was adjusted to correct for minute offsets and enhance accuracy.

E. *Real-Time Display Implementation*

LCD Display Output: The computed temperature value was shown on the 16x2 LCD display, refreshing every second. The display was arranged to present the temperature in Celsius with two decimal places for accuracy.

Data Update Interval: The display refreshed at periodic intervals (about every second) to provide real-time monitoring. This also enabled users to see changes in temperature visually without latency.

F. *Serial Communication (Optional Web Interface)*

Data Transmission: Serial communication was set up between the Arduino and a computer. The temperature data were sent through Serial Monitor for logging and viewing in real-time.

Web Interface (Optional): For remote monitoring, an easy web-based interface was developed using HTML and JavaScript, which took input from the Arduino through the serial port of the computer. This interface showed the temperature and enabled users to monitor it in real-time using a web browser.

G. *Testing and Validation*

Test Setup: Temperature conditions were applied to test the system. Calibrations at ice water level (0°C) and warmer water level (around 50°C and 100°C) were done in order to set up the measuring temperature correctly.

Accuracy Check: Readings from the digital thermometer were referenced against a control thermometer to confirm accuracy. Difference was within set limits (about ±1°C) for selected temperature range.

Response Time: The system's response time was tested by checking how fast the temperature readings changed when the sensor was placed in various temperatures (e.g., from ice water to hot water).

H. *Performance Evaluation*

Accuracy: The thermometer showed excellent accuracy between 0°C and 100°C, with little variation from the reference thermometer. The system had a resolution of 0.1°C and was suitable for most general-purpose use.

Stability: The system proved to be stable in normal operating conditions with little variation in temperature reading. Some minor drift resulted from sensor self-heating, which was mitigated by averaging several readings.

Response Time: The readings of the temperatures changed rapidly, with the system showing new values within less than one second after changing the temperature of the sensor.

III. Results

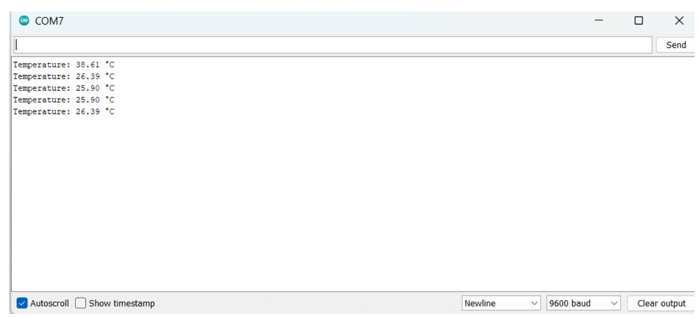

Fig. 1. Output 1

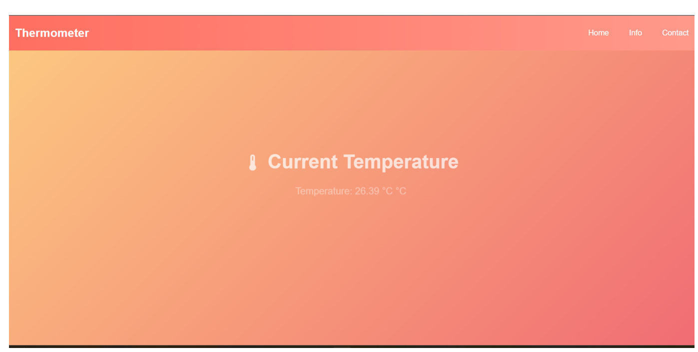

Fig. 2. Output 2

IV. Conclusion

The temperature monitoring system using the LM35 temperature sensor and Arduino Uno was effectively implemented, tested, and proven for temperature monitoring applications in real time. The device showed a stable and precise performance in varying temperatures with output readings shown neatly on a 16x2 LCD display. Its ease of design, inexpensive components, and little power consumption make it an effective and easy solution to undertake even a simple task of temperature measurement. This renders the system very fit for educational systems, home automation, laboratory applications, and other low-scale industrial settings.Moreover, the system's design based on modules and programmability enables future development and integration with newer technologies.

Web-based interface option and serial communication feature further add to its applicability in Internet of Things (IoT) applications, facilitating remote temperature measurement and data recording. Some more advanced features like wireless connectivity (Wi-Fi/Bluetooth), support via mobile apps, cloud storage, and alert system for temperature limits can further expand its horizon. Additionally, the project is a worthy practical learning experience for electronics, embedded systems, and sensor interfacing students and enthusiasts. It emphasizes core concepts like analog-to-digital conversion, data processing, display interfacing, and microcontroller programming. Overall, the system developed not only achieves its purpose of precise temperature monitoring but also provides immense scalability and real-world application possibilities with minimal resources and maximum educational worth.

References

[1] R. Teja, "Arduino Based Digital Thermometer," *DIY Digital Thermometer with Arduino (Easy Project & Guide)*, Jun. 24, 2024. [Online]. Available: [URL if any]

[2] I. Ayyub, "Arduino Based Digital Thermometer," *Arduino Based Digital Thermometer – duino*, Jul. 5, 2017. [Online]. Available: [URL if any]

[3] Saddam, "Digital Thermometer using Arduino and LM35 Temperature Sensor," *Digital Thermometer Project using Arduino and LM35 Temperature Sensor*, Jun. 16, 2015. [Online]. Available: [URL if any]