International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 10 | Oct - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Create a Digital Thermometer Using Arduino and a Temperature Sensor.

Ms. Sanchita Jamdhade
Department of E&TC JSPM's BSP
Wagholi Pune, Maharastra

Ms. Vaishnavi Jarad
Department of E&TC JSPM's BSP
Wagholi Pune, Maharastra

Mr. Pranav Joshi
Department of E&TC JSPM's BSP
Wagholi Pune, Maharastra

Mr. Yogesh Kale
Department of E&TC JSPM's BSP
Wagholi Pune, Maharastra

Ms.Gauri Eknath Dhaktode
Deparment of E&TC JSMP's BSP
Wagholi Pune, Maharastra

Abstract— Temperature monitoring is an important part of many electronic, industrial, environmental, and healthcare applications. Traditional thermometers may require manual observation, while digital temperature monitoring systems can provide electronic readings that are easy to observe and record. This paper presents a proposed Arduino-based digital temperature monitoring system using a temperature sensor.

The proposed system consists of an Arduino Uno microcontroller board, an LM35 temperature sensor, and a display interface. The LM35 sensor measures the surrounding temperature and produces an analog voltage proportional to the temperature. The Arduino reads this voltage through its analog input, converts the sensor reading into a temperature value, and displays the result in degrees Celsius through the Serial Monitor or an optional LCD display.

The system is designed to provide a simple, low-cost, and easy-to-understand method of temperature measurement. It demonstrates the application of embedded systems, sensor interfacing, analog-to-digital conversion, and microcontroller-based data processing. The proposed design can be used for educational purposes and can be extended for environmental monitoring and other temperature-sensing applications. Practical implementation and experimental testing are required to evaluate the accuracy, response time, and reliability of the system.

Keywords— Arduino, Digital Thermometer, LM35 Temperature Sensor, Embedded System, Temperature Monitoring, Microcontroller, Analog-to-Digital Conversion.

1. INTRODUCTION

Temperature is one of the most important physical parameters measured in electronic and industrial systems. Temperature monitoring is required in laboratories, homes, storage facilities, industrial equipment, and environmental monitoring applications. Accurate temperature information helps users observe changes in surrounding conditions and take suitable action when required.

Conventional thermometers are commonly used to measure temperature. However, some conventional methods require manual observation and do not directly provide electronic data for further processing. Digital temperature monitoring systems overcome these limitations by using electronic sensors and microcontrollers to measure and display temperature readings.

The development of embedded systems has made it possible to design compact and economical temperature monitoring devices. An embedded system combines hardware and software to perform a specific function. In a digital thermometer, the temperature sensor detects temperature, while the microcontroller processes the sensor signal and produces a readable output.

The proposed Arduino-based Digital Temperature Monitoring System uses an LM35 temperature sensor and an Arduino Uno board. The sensor produces an analog voltage corresponding to the measured temperature. The Arduino converts the analog signal into a digital value and calculates the temperature in degrees Celsius. The processed reading is displayed through the Serial Monitor or an optional LCD display.

The main aim of this system is to demonstrate a simple method of digital temperature measurement using commonly available electronic components. The design provides a basic platform for understanding sensor interfacing, embedded programming, and real-time data monitoring.

2. OBJECTIVES OF THE SYSTEM

The main objectives of the proposed system are:

1. To design a simple digital temperature monitoring system using Arduino.
2. To measure surrounding temperature using an LM35 temperature sensor.
3. To interface the temperature sensor with the Arduino analog input.
4. To convert sensor voltage readings into temperature values in degrees Celsius.
5. To display temperature readings through the Serial Monitor or an optional LCD display.
6. To develop an economical and easy-to-understand embedded system prototype.
7. To study the use of sensors and microcontrollers in temperature measurement.
8. To provide a basic platform for future improvements in automatic temperature monitoring.

3. PROPOSED SYSTEM

The proposed system consists of a temperature-sensing unit, a microcontroller unit, and an output display interface. The LM35 temperature sensor senses the surrounding temperature and generates an analog voltage proportional to the measured temperature.

The analog output of the sensor is connected to an analog input pin of the Arduino Uno. The Arduino reads the analog signal using its built-in analog-to-digital converter. The measured analog value is converted into voltage, and the corresponding temperature is calculated using the sensor's conversion relationship.

The calculated temperature is then displayed in degrees Celsius through the Arduino IDE Serial Monitor. An LCD display can also be connected when a standalone display is required.

The system continuously repeats the sensing and processing steps while it is powered and programmed to operate. The user can observe temperature changes through successive readings. The system is intended for basic temperature monitoring and educational demonstrations rather than certified medical or industrial measurement.

The proposed system follows a simple sequence:

Temperature sensing → Analog signal acquisition → Arduino processing → Temperature calculation → Display of temperature

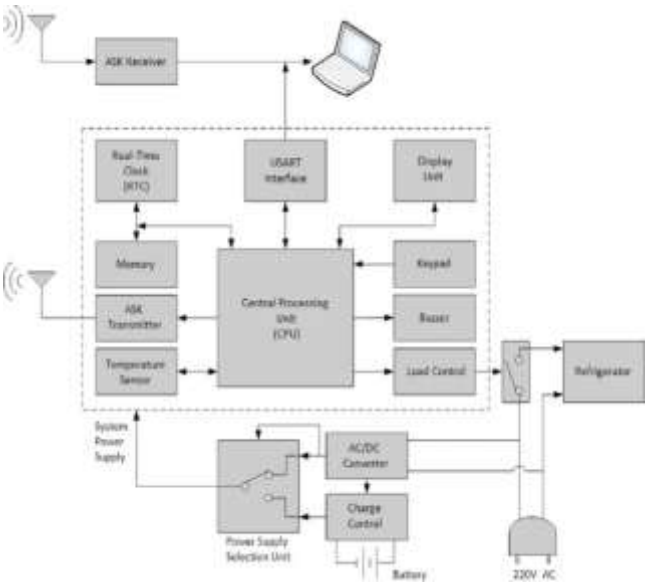


Fig.1 Block diagram of the proposed Arduino-based Digital Temperature Monitoring System.

4.HARDWARE AND SOFTWARE REQUIREMENTS

4.1 Hardware Components

Component	Function
ESP32 development board	Reads sensor data, processes movement information, and supports wireless communication.
Arduino Uno Development Board	Converts analog readings into digital values. Processes the sensor data using the uploaded program.
LM35 Temperature Sensor	The LM35 is an analog temperature sensor whose output voltage is proportional to temperature in degrees Celsius. Its nominal scale factor is 10 mV per degree Celsius.
USB power supply or suitable battery	Supplies power to the controller and sensor.
Laptop or smartphone	Displays monitoring information and supports configuration.
Breadboard and Jumper Wires	These components connect the Arduino and temperature sensor during prototype development. They allow the circuit to be assembled and tested without designing a permanent printed circuit board.

Hardware-Based System

1. ESP32 development board



Fg. 1.1 ESP32 development board

Main function: Processing and control unit.

The ESP32 acts as the brain of the system. It receives motion data from the IMU sensors, performs calculations, counts exercise repetitions, and transmits the processed information to a smartphone or computer.

Detailed working

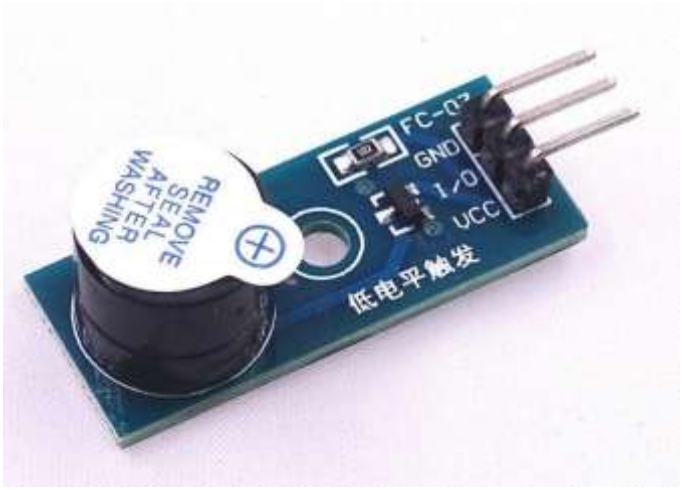
1. Sensor communication: The ESP32 communicates with the MPU6050 sensors through the I²C digital interface using the SDA and SCL signals.
2. Data acquisition: It reads acceleration and angular-velocity measurements at a predefined sampling rate.
3. Signal processing: It filters noisy readings and uses an orientation-estimation algorithm to estimate body-segment orientations.
4. Angle calculation: It calculates the relative orientation of the two body segments to estimate the selected joint angle.
5. Repetition counting: It detects movement cycles using calibrated thresholds and a state-based algorithm.
6. Wireless transmission: It sends timestamped measurements and exercise statistics through Bluetooth Low Energy (BLE) or Wi-Fi.
7. Fault handling: It can detect missing sensor readings or communication failures and flag invalid measurements.

Example

When a patient performs a knee-flexion exercise, the ESP32 processes measurements from sensors attached to the thigh and lower leg. It estimates the relative angle, detects completed repetitions, and updates the monitoring application.

Why it is needed: Without the ESP32, the sensor measurements would not be processed and coordinated by the proposed embedded system.

2.Buzzer or vibration motor



Fg.1.2Buzzer or Vibration motor

This component gives feedback without requiring the patient to continuously watch the display.

Working in the system -

1. The ESP32 checks the repetition count, movement status, or configured alert conditions.
2. When a condition is met, it sends a control signal to the output circuit.
3. A buzzer produces a sound, or a vibration motor provides tactile feedback.
4. The output stops after the programmed duration.

Possible feedback examples include:

- A short beep when the prescribed repetition target is reached.
- A vibration when a sensor becomes disconnected.
- A warning signal when a configured monitoring threshold is crossed.

A suitable driver transistor or MOSFET may be necessary for a motor or other load that draws more current than a GPIO can supply. Medical alerts must be designed and validated carefully; an ordinary prototype buzzer is not a certified emergency alarm.

3.Breadboard, jumper wires and connectors.



Fg.1.3 Breadboard, Jumper wires and Connectors

These components connect the hardware during development.

Working in the system

- The breadboard allows circuits to be assembled without soldering.
- Jumper wires carry power, ground and communication signals.
- Connectors make sensor replacement and testing easier.
- The developer can test the circuit before designing a permanent PCB.

For an academic prototype, use short, secure connections and common ground where required. Check every module's voltage compatibility before connecting it to the ESP32.

4.2 Software Requirements

Arduino IDE: Used to write, compile, and upload the embedded program.

Embedded C/C++: Used to write the program that reads the sensor, calculates temperature, and displays the output.

Serial Monitor: Used to observe temperature values transmitted by the Arduino through the USB serial connection.

Optional LCD Library: Used when a compatible LCD is included in the system.

The software repeatedly reads the analog input, calculates the corresponding temperature, and displays the updated value at a selected interval.

5. WORKING METHODOLOGY

The working methodology of the proposed system is divided into the following steps.

Step 1: Sensor Connection

The LM35 temperature sensor is connected to the Arduino according to its datasheet. The supply, ground, and analog output connections are checked before powering the circuit.

Step 2: Program Development

A program is written using the Arduino IDE. The program configures the analog input and serial communication and defines the calculations required to convert sensor readings into temperature values.

Step 3: Temperature Sensing

When the system is powered, the LM35 senses the surrounding temperature. Its output voltage changes according to the temperature being measured.

Step 4: Data Acquisition

The Arduino reads the analog output voltage using its analog input and built-in analog-to-digital converter. The resulting digital reading represents the sensor voltage within the limits of the selected voltage reference.

Step 5: Temperature Calculation

The program converts the digital reading into voltage and calculates the temperature using the LM35's nominal conversion relationship.
For a typical Arduino Uno using a nominal 5 V analog reference and a 10-bit ADC:

Sensor voltage = (ADC reading × 5.0) / 1023

Temperature in °C = Sensor voltage × 100

These equations assume the nominal reference voltage and the LM35's nominal 10 mV/°C scale factor. Actual accuracy depends on the hardware and calibration.

Step 6: Display of Readings

The calculated temperature is sent to the Serial Monitor in degrees Celsius. If an LCD is installed, the program can display the temperature on the LCD instead.

Step 7: Continuous Monitoring

The program repeats the measurement and display process at regular intervals. This allows the user to observe changes in temperature over time.

Step 8: Verification and Testing

The readings can be compared with a suitable reference thermometer under controlled conditions. Multiple measurements can be recorded to evaluate measurement differences, stability, and response time.

START

↓

Initialize Arduino and serial communication

↓

Read analog value from the LM35 sensor

↓

Convert analog reading into voltage

↓

Calculate temperature in degrees Celsius

↓

Display temperature reading

↓

Wait for the selected sampling interval

↓

Read the sensor again

↓

Continue monitoring

Fig. 2. Flowchart of the proposed temperature monitoring process.

6. SYSTEM DESIGN AND DATA MONITORING

The proposed system is designed around three main stages: sensing, processing, and displaying data.

Sensing stage: The LM35 measures the surrounding temperature and generates a corresponding analog voltage.

Processing stage: The Arduino reads the analog signal, converts the reading into voltage, and calculates the temperature using the programmed conversion formula.

Display stage: The calculated temperature is sent to the Serial Monitor or an optional LCD display for observation.

The system can display readings in degrees Celsius at regular intervals. These readings can be recorded manually during testing to study temperature variations. If additional software and storage facilities are introduced, the system can also be extended to maintain a time-based temperature log.

For a basic prototype, the Arduino Uno, LM35 sensor, USB cable, and computer are sufficient to demonstrate the sensing and display process. An LCD can be added if independent operation without a computer is required.

Correct wiring, an appropriate voltage reference, suitable sensor placement, and a consistent measurement procedure are important for obtaining reliable readings.

7. EXPECTED RESULTS AND PERFORMANCE EVALUATION

The proposed system is expected to measure surrounding temperature using the LM35 sensor and display the calculated values through the Arduino Serial Monitor or an optional LCD.

When the surrounding temperature changes, the sensor output voltage changes accordingly. The Arduino processes the new reading and updates the displayed temperature. The system therefore demonstrates the basic operation of an embedded digital thermometer.

The actual performance must be evaluated through practical implementation and testing. Numerical accuracy should not be claimed until measurements have been collected.

The following parameters can be used for evaluation:

1. Temperature Measurement Difference

Compare the system reading with a suitable reference thermometer under the same conditions.

Measurement difference = System reading − Reference reading.

2. Reading Stability

Observe the displayed temperature while the sensor is kept in a stable environment. Record fluctuations over a selected period.

3. Response Time

Measure the time taken for the displayed reading to respond to a controlled temperature change.

4. Repeatability

Repeat measurements under similar conditions and compare the recorded values.

5. Display and Communication

Verify that the Serial Monitor or LCD displays the calculated readings correctly and updates at the selected interval.

These tests can help determine the suitability of the prototype for basic educational and temperature-monitoring applications.

8. ADVANTAGES AND APPLICATIONS

Advantages

Simple circuit design and easy implementation.

Low-cost components commonly used in embedded-system projects.

Digital display of measured temperature.

Easy programming and modification using Arduino IDE.

Useful for learning sensor interfacing and analog-to-digital conversion.

Can be extended with additional display or data-recording features.

Suitable for laboratory demonstrations and educational projects.

Applications

- ✧ Basic room-temperature monitoring.
- ✧ Temperature observation in educational laboratories.
- ✧ Environmental temperature monitoring.
- ✧ Temperature monitoring in small storage spaces, within the sensor's specified operating range.
- ✧ Demonstration of sensor interfacing in embedded-system laboratories.
- ✧ Development of introductory temperature-monitoring prototypes.
- ✧ Educational demonstrations of microcontroller-based measurement systems.

The prototype should not be treated as a certified medical thermometer or as a replacement for equipment requiring validated measurement accuracy.

9. LIMITATIONS AND SAFETY CONSIDERATIONS

The performance of the proposed system depends on the sensor characteristics, Arduino analog reference voltage, circuit connections, sensor placement, and calibration.

The LM35 provides an analog output, so measurement resolution and accuracy are affected by the analog-to-digital converter and reference voltage. A nominal 5 V reference is an approximation unless the actual reference is measured or otherwise characterized.

The sensor may require time to respond to temperature changes. Its readings can also be affected by placement, airflow, nearby heat sources, and thermal contact with surrounding objects.

Loose connections, incorrect pin identification, electrical noise, or an incorrect conversion formula can produce unreliable readings. The sensor must be connected according to the datasheet, and the circuit must operate within the specified voltage and temperature limits.

The basic system does not automatically provide cloud monitoring, remote alerts, or long-term data storage. These features require additional hardware or software.

The proposed system is intended for educational and general-purpose temperature monitoring. It should not be used for medical diagnosis, safety-critical temperature control, or other applications requiring certified accuracy without suitable calibration, validation, and protective measures.

10. FUTURE SCOPE

The proposed Arduino-based digital temperature monitoring system can be improved in the future by adding advanced features to make it more accurate, reliable, and user-friendly. An LCD display can be integrated to show the measured temperature clearly. Wireless communication technologies such as Wi-Fi or Bluetooth can be added to transmit temperature readings to a smartphone or computer for remote monitoring.

The system can also be connected to an IoT platform to store temperature data and monitor temperature variations over time. A buzzer can be included to generate an alert when the temperature exceeds a predefined limit. A vibration motor may also be used to provide a vibration alert when required. In addition, advanced temperature sensors and calibration techniques can improve measurement accuracy. These improvements can make the proposed system useful for home

monitoring, laboratories, educational projects, and other temperature-monitoring applications.

11. CONCLUSION

The proposed Arduino-Based Digital Temperature Monitoring System demonstrates a simple method of measuring and displaying temperature using an LM35 sensor and an Arduino Uno microcontroller. The sensor converts temperature into an analog voltage, while the Arduino reads the signal, calculates the temperature in degrees Celsius, and displays the result through the Serial Monitor or an optional LCD.

The design provides a practical introduction to embedded systems, analog sensor interfacing, analog-to-digital conversion, and microcontroller programming. Its simple architecture and commonly available components make it suitable for educational demonstrations and basic temperature-monitoring applications.

The system can be extended with wireless communication, IoT connectivity, data logging, and automatic alerts. However, its measurement accuracy and practical usefulness must be established through suitable testing and calibration. The proposed system provides a foundation for understanding and developing more advanced temperature-monitoring applications.

ACKNOWLEDGEMENT

The authors would like to express their sincere gratitude to their project guide and the Department of Electronics and Telecommunication Engineering at JSPM's Bhivrabai Sawant Polytechnic, Wagholi, Pune, for their guidance and support in preparing this research paper. The authors also acknowledge the technical literature and research publications that helped them understand wearable sensing, embedded systems, and rehabilitation monitoring. This acknowledgement should be retained only if it accurately reflects the support received.

REFERENCES

1) Arduino, "Arduino UNO Rev3," Arduino Official Documentation. Available: https://docs.arduino.cc/hardware/uno-rev3/
2) Arduino, "Arduino Language Reference," Arduino Official Documentation. Available: https://docs.arduino.cc/language-reference/
3) Texas Instruments, "LM35 Precision Centigrade Temperature Sensors," LM35 product documentation and datasheet. Available: https://www.ti.com/product/LM35
4) Arduino, "Arduino IDE," Arduino Official Documentation. Available: https://docs.arduino.cc/software/ide/
5) M. A. Mazidi, S. Naimi, and S. Naimi, *The AVR Microcontroller and Embedded Systems: Using Assembly and C*. Pearson.
6) A. S. Sedra and K. C. Smith, *Microelectronic Circuits*. Oxford University Press.
7) Relevant research articles on temperature sensing, microcontroller-based monitoring, and embedded measurement systems should be selected and verified from the original publications before submission.

BIOGRAPHIES (OPTIONAL)



Dhaktode Gauri Eknath is a Lecturer in the Department of E&TC, JSPM's BSP, Wagholi, Pune

Email:**gauridhaktode123@gmail.com**



Sanchita Prakash Jamdhade is a Student in Department of E&TC, JSPM's BSP, Wagholi, Pune.

E-mail: **sanchitaprakashjamdhade@gmail.com**



Vaishnavi Chandrakant Jarad is a Student in Department of E&TC, JSPM's BSP, Wagholi, Pune.

E-mail: **vaishnavijarad29@gmail.com**



Pranav Vijay Joshi is a Student in Department of E&TC, JSPM's BSP, Wagholi, Pune.

E-mail: **pranav7261@gmail.com**



Yogesh Ashok Kale is a Student in Department of E&TC, JSPM's BSP, Wagholi, Pune.

E-mail: **yogeshkale1193@gmail.com**