FOSSEE Winter Internship 2025

NAME: DANDUGULA MADHU

Branch: MSc Geo Informatics (1st year)-Post Graduation

Graduation: Btech-CSe_IoT(2021-2025)

Task 1: Sensor–Actuator Integration (Arduino)

Goal:
Build a simple real-time system that **monitors an analog sensor** and **triggers an actuator** based on a threshold.

1. Introduction

A real-time temperature monitoring system is essential in many applications such as home automation, industrial safety, and environmental monitoring. This project uses an Arduino Uno, an analog temperature sensor (LM35/TMP36), two LEDs, and a buzzer to detect temperature levels and trigger visual and audio alerts based on a predefined threshold.

The system continuously reads analog temperature values and responds instantly, making it suitable for real-time embedded control.

---

2. Objective

- To design and implement a real-time temperature monitoring system.
- To trigger a **hot indicator (LED + buzzer)** when temperature crosses a threshold.
- To trigger a **cool indicator (LED)** when temperature is below or equal to the threshold.
- To understand interfacing analog sensors and actuators with microcontrollers.

---

3. Components Used

Component	**Quantity**	**Description**
Arduino Uno	1	Microcontroller board for processing
LM35/TMP36 Temperature Sensor	1	Provides analog temperature output
LED (Red)	1	Indicates temperature above threshold
LED (Green/Blue)	1	Indicates normal temperature
Buzzer	1	Audible warning on high temperature
220Ω Resistors	2	Current limiting for LEDs
Connecting Wires	—	Circuit connections
USB Cable	1	Power and programming

---

4. Circuit Diagram (Description)

Sensor (LM35/TMP36) Connections

- Left pin → **5V**
- Middle pin → **A0**
- Right pin → **GND**

LED Connections

Hot LED (Red):

- Arduino Pin 9 → 220Ω Resistor → LED Anode (+)
- LED Cathode (−) → GND

Cool LED (Green/Blue):

- Arduino Pin 8 → 220Ω Resistor → LED Anode (+)
- LED Cathode (−) → GND

Buzzer

- Pin 10 → Buzzer (+)
- Buzzer (−) → GND

---

5. Working Principle

1. The LM35/TMP36 produces an analog voltage proportional to temperature.
2. Arduino reads this analog value on **A0** and converts it into temperature (°C).
3. The system compares the measured temperature with a predefined threshold value.
4. If temperature **exceeds** the threshold:
   - Hot LED turns ON
   - Buzzer turns ON
   - Cool LED remains OFF
5. If temperature is **≤ threshold**:
   - Cool LED turns ON
   - Hot LED and buzzer remain OFF

The loop runs continuously, enabling **real-time response**.

---

6. Arduino Code

```
const int sensorPin = A0;

const int coolLed = 8;

const int hotLed = 9;
const int buzzer = 10;

float threshold = 35.0; // Temperature threshold in °C

void setup() {
  pinMode(coolLed, OUTPUT);
  pinMode(hotLed, OUTPUT);
  pinMode(buzzer, OUTPUT);
  Serial.begin(9600);
}

void loop() {
  int sensorValue = analogRead(sensorPin);

  float voltage = sensorValue * (5.0 / 1023.0);
  float temperature = voltage * 100.0;  // LM35/TMP36
calculation

  Serial.print("Temperature: ");
  Serial.println(temperature);

  if (temperature > threshold) {
    digitalWrite(hotLed, HIGH);
    digitalWrite(buzzer, HIGH);
    digitalWrite(coolLed, LOW);
  }
  else {
    digitalWrite(coolLed, HIGH);
    digitalWrite(hotLed, LOW);
    digitalWrite(buzzer, LOW);
  }

  delay(200);
}
```

7. Applications

- Industrial temperature monitoring
- Server room temperature alarms
- Smart home systems
- Greenhouse temperature control
- Safety alert systems

---

8. Advantages

- Low-cost and easy to build
- Real-time temperature detection
- Audible and visual alerts
- Low power consumption
- Scalable for multiple sensors

---

9. Conclusion

This project successfully demonstrates a real-time temperature monitoring system using Arduino. By utilizing a simple analog sensor and basic actuators, the system provides immediate feedback when temperature crosses a critical limit. It serves as a practical example of embedded system design, sensor interfacing, and actuator control.