Interim Project Presentation On Forest Fire Alarm System

Department of ECE

National Institute of Technology Agartala

Acknowledgement

We would like to thank our supervisor, ***Dr. Mitra Barun Sarkar*** for being a great mentor and the best adviser we could ever have. His advice, encouragement and critics are source of innovative ideas, inspiration and causes behind the successful completion of this report. The confidence shown on us by him was the biggest source of inspiration for us. It has been a privilege working with him.

Group -11

Biprajit Deb (21UEC061)
Abhishek Debnath (21UEC007)
Santa Sarkar (21UEC016)
Ratnadeep Das (21UEC058)

Objective

To Design and Develop a Real time Forest Fire Alarm System capable of detecting fire at early stage to prevent large scale disasters.

Why is it needed?

Forest fires pose a significant threat to ecosystems, wildlife, and human civilization. Early detection is crucial for minimizing damage and ensuring quick response. This project aims to create an efficient system using IoT to detect and notify about potential fire hazards.

Forest Fire Incidents in recent years

Australian Bushfires (2019-2020) - *Nearly 3 billion animals killed*

Amazon Rainforest Fire (2019-2020) - *Over 17 million hectares area affected*

California Wildfires (2023) - *affects thousands hectares of area*

Key Takeaways from history

Forest fires have devastating environmental, social, and economic consequences.

The recurrence of such incidents underscores the need for real-time monitoring systems to enable early detection and swift response.

What are we Proposing?

A cost-effective, Arduino-based system using several sensors (temperature and humidity , flame , smoke) to monitor fire-prone areas continuously and alert the authorities via SMS.

Real-Time Fire Detection System

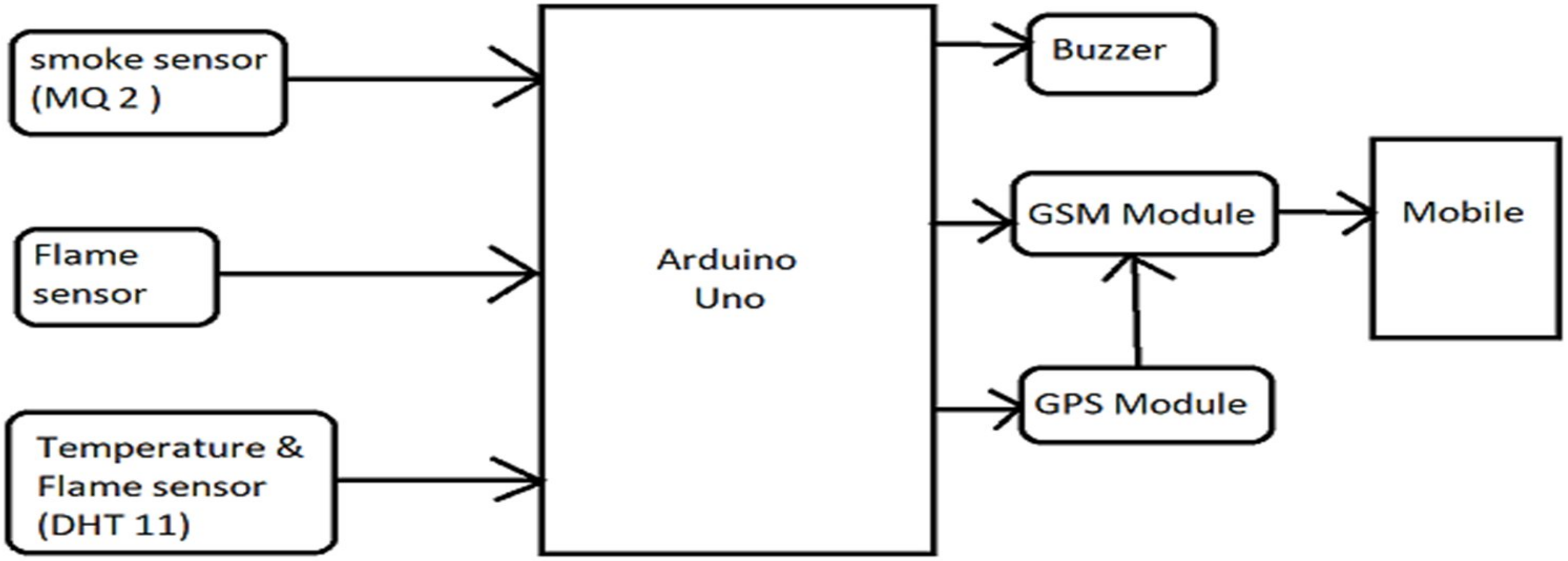
Salient Features:

Immediate Alerts: *SMS notification with location on G-map for instant emergency response*

Sustainability: *Low power consumption and eco-friendly design*

Future Integration: *Predictive analysis to identify fire-prone regions using historical data.*

System Architecture (Block Diagram)



Components Used

Hardware Requirements

- ARDUINO UNO
- TEMPERATURE & HUMIDITY SENSOR (DHT-11)
- SMOKE SENSOR (MQ-2)
- GSM/GPRS/GPS MODULE (SIM808)
- BUZZER

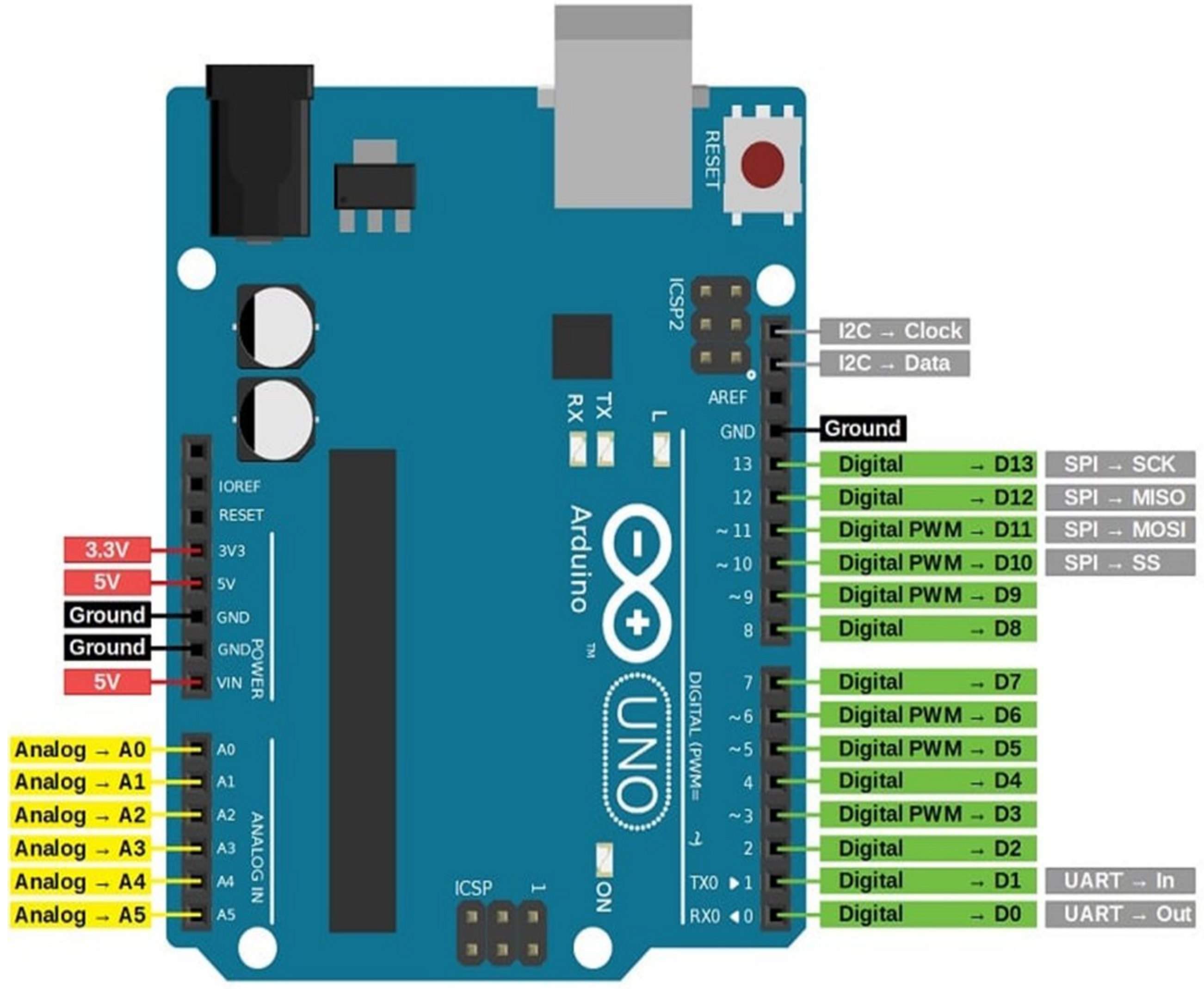
Software tools

PROTEUS 8 PRO
FOR SIMULATING THE CIRCUIT DESIGN.

ARDUINO IDE
FOR PROGRAMMING AND UPLOADING THE PROGRAM TO THE MICROCONTROLLER.

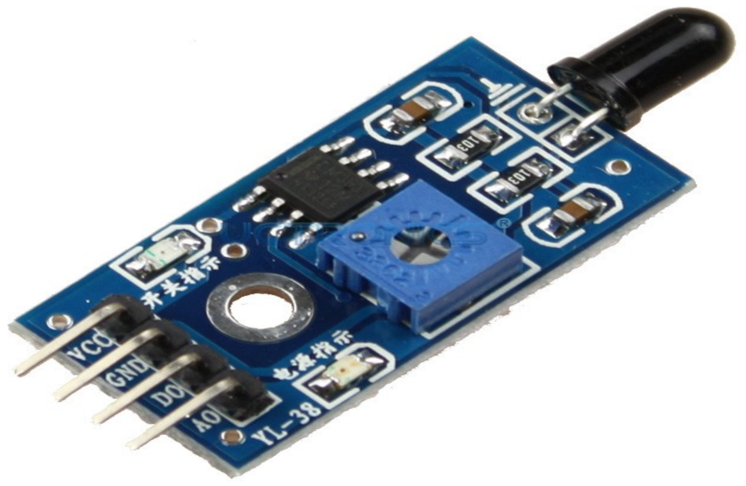
Arduino UNO

The Arduino UNO microcontroller board is based on the ATmega328P microcontroller and has 14 digital I/O pins, 6 analog input pins, a USB connection, a power jack, and an ICSP header.



Flame Sensor

A flame sensor detects the presence of fire or flame. Upon detecting a flame or fire, the sensor generates an electrical signal.

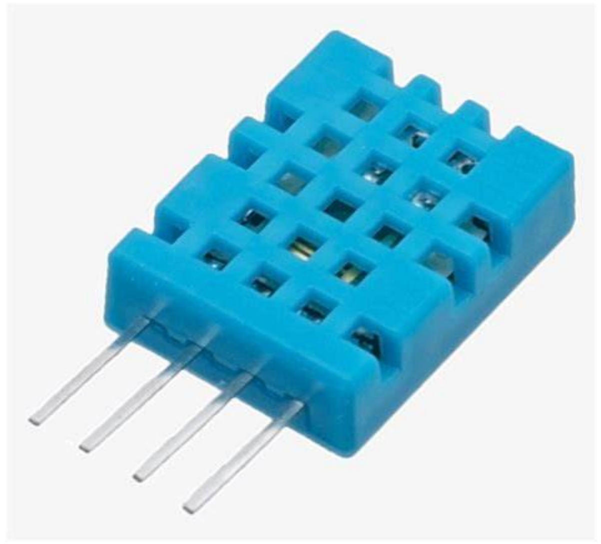


Smoke Sensor

The MQ-2 sensor is a versatile gas sensor that can detect various gases, including smoke, LPG, and methane.

DHT-11

The DHT11 sensor is used to measure temperature and humidity. It operates with a range of 0–50°C for temperature and 20–80% for humidity.



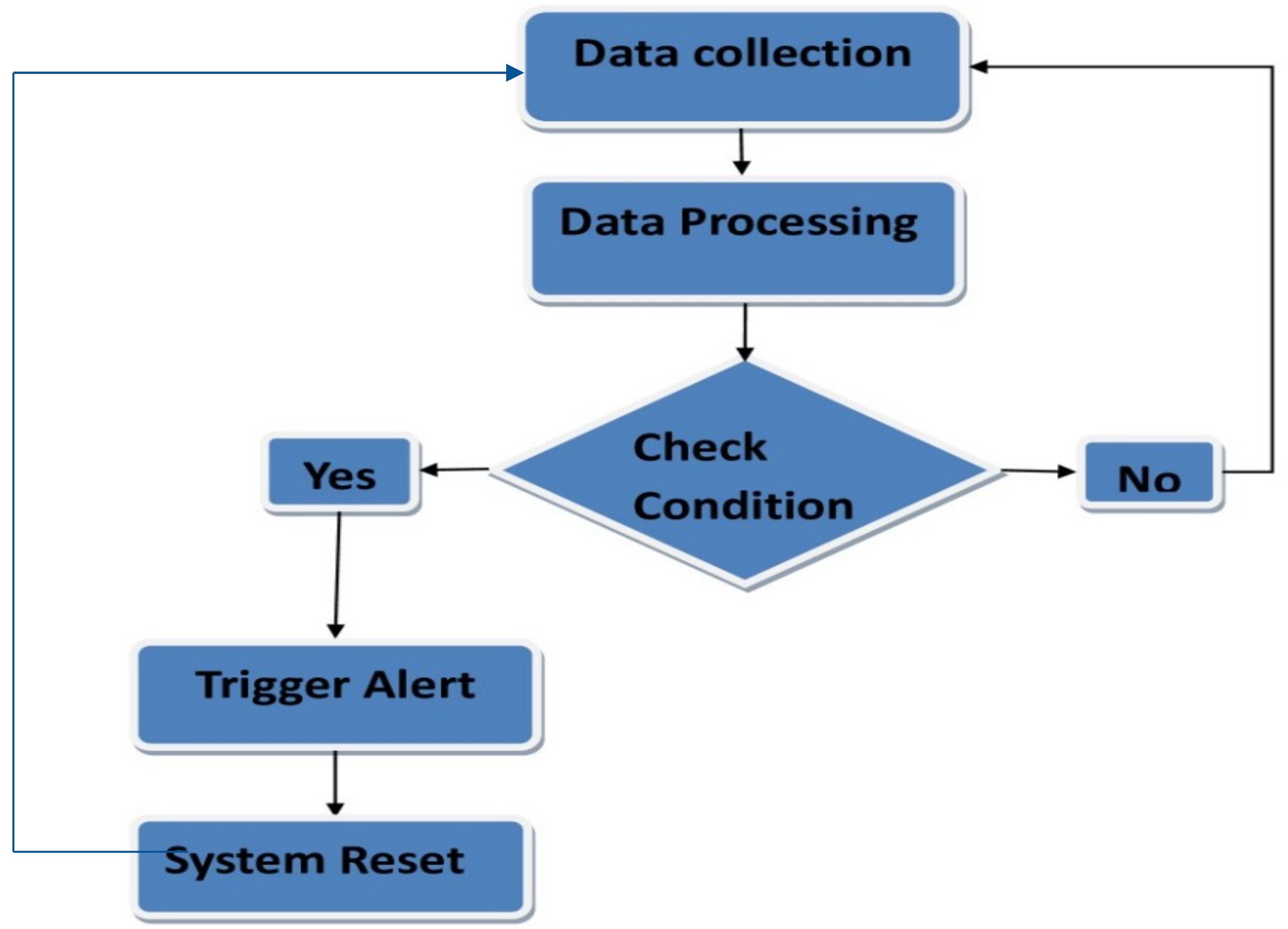
Buzzer

A buzzer is an electronic component that produces sound when an electric current is applied to it. It's used in devices to provide audible alerts.

GSM/GPRS/GPS Module

The SIM808 is a quad-band GSM/GPRS/GPS module designed for M2M applications. It supports voice calls, SMS, data transmission, and GPS positioning. It's easy to integrate with microcontrollers and other devices, making it ideal for IoT and remote projects.

Workflow of the System



1.Data Collection: Sensors (flame, temperature, smoke) continuously monitor environmental parameters.

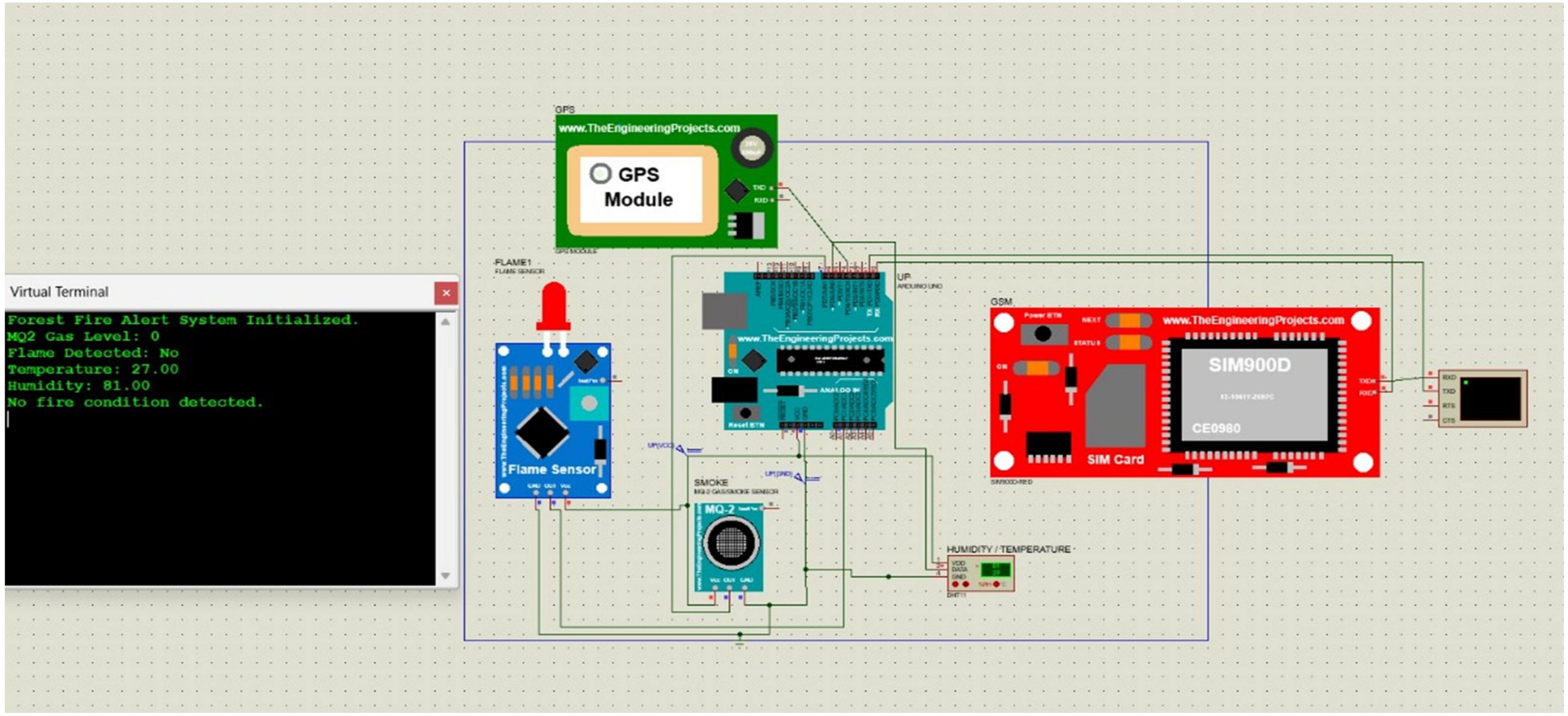
2. Data Processing: Arduino processes the sensor data to detect abnormalities indicating fire.

3. Condition Check: System evaluates if sensor values exceed predefined thresholds.

4. Trigger Alert: If fire is detected, a buzzer sounds, and an SMS alert is sent via GSM module.

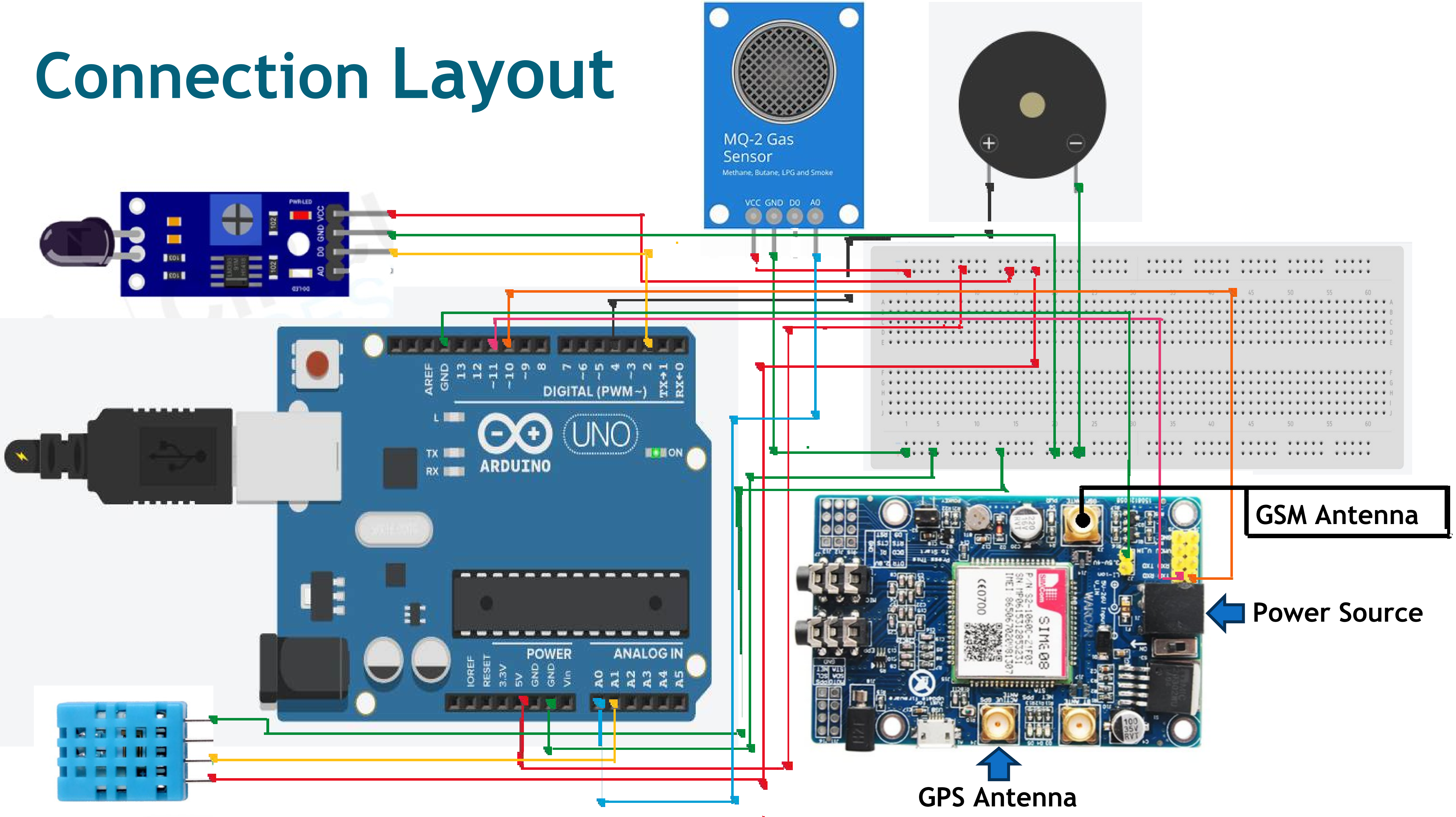
5. System Reset: Once the fire is controlled, the system resets for continuous monitoring.

Circuit Simulation (in Proteus)



```
Forest Fire Alert System Initialized.
MQ2 Gas Level: 0
Flame Detected: No
Temperature: 27.00
Humidity: 81.00
No fire condition detected.
```

Connection Layout



Pin Connections

Flame Sensor: DO pin to Arduino Digital pin 2, VCC to Arduino 5V, GND to Arduino GND.

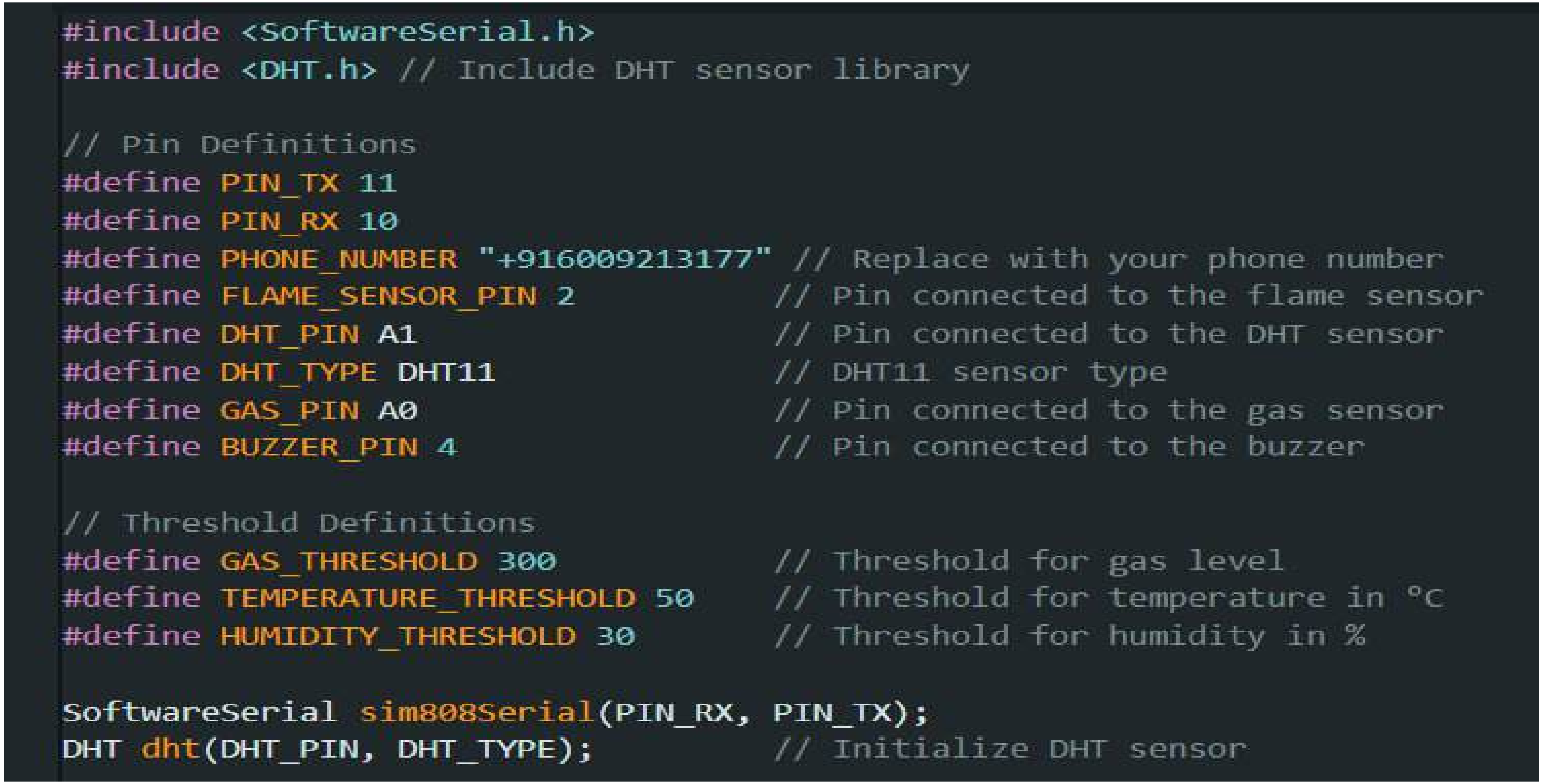
MQ2: The AO pin to Arduino Analog pin A0, VCC to Arduino 5V, GND to Arduino GND.

DHT-11: AO pin to Arduino Analog pin A1, VCC to Arduino 5V, GND to Arduino GND.

Buzzer: Anode(longer pin) to Arduino Digital pin 4 and Cathode(shorter pin) to Arduino GND.

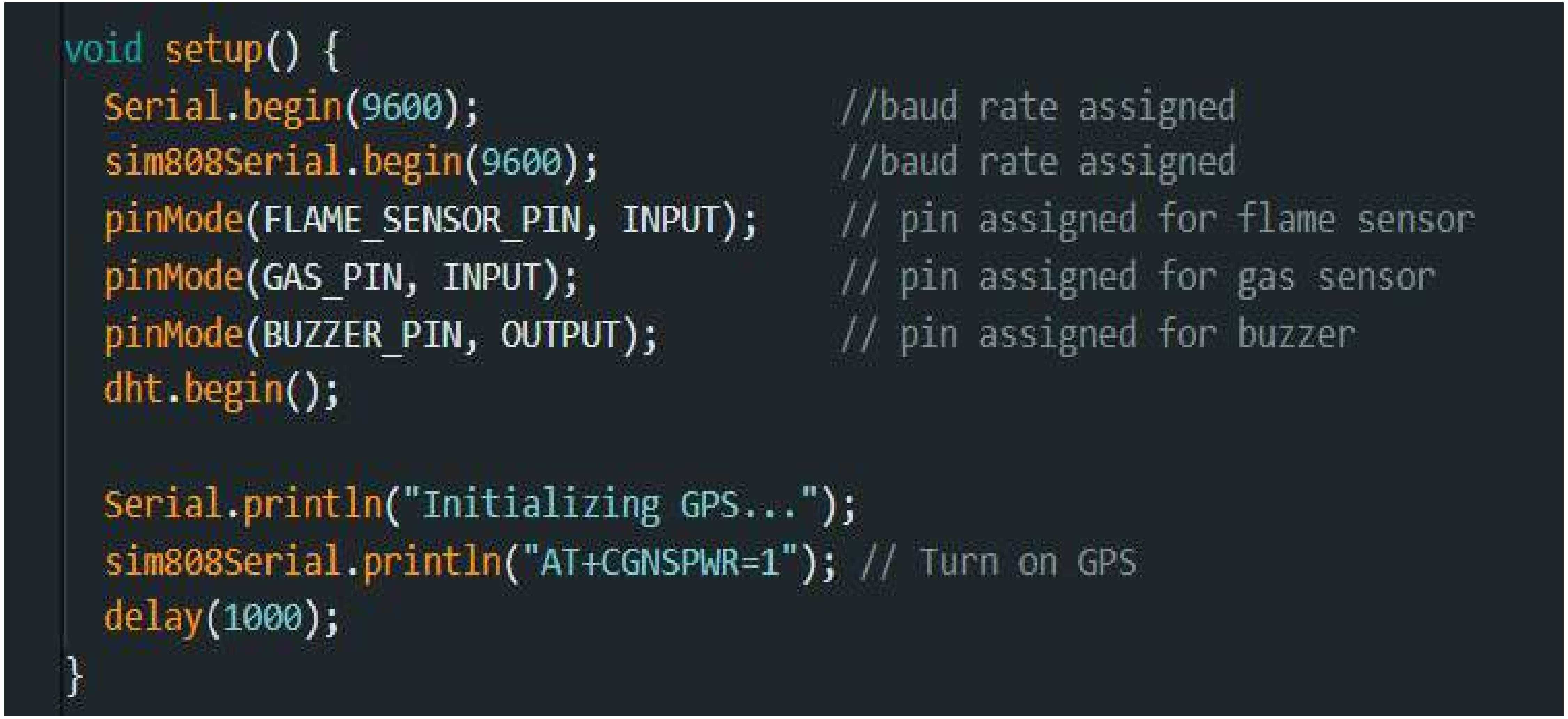
SIM808: TXD and RXD pin to Arduino Digital pin 10 and 11 respectively and the GND to Arduino GND.

Arduino Program



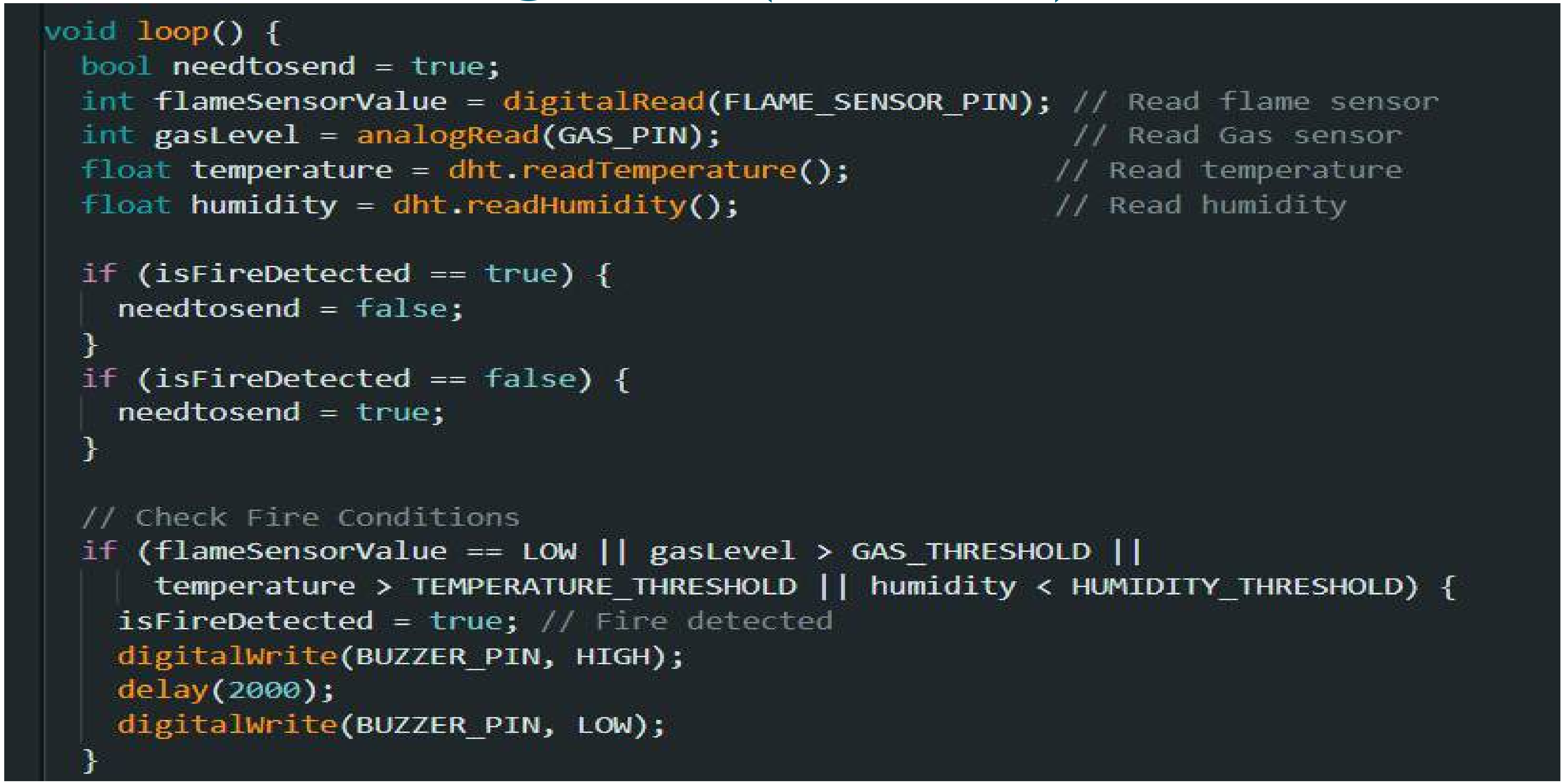
```
#include <SoftwareSerial.h>
#include <DHT.h> // Include DHT sensor library

// Pin Definitions
#define PIN_TX 11
#define PIN_RX 10
#define PHONE_NUMBER "+916009213177" // Replace with your phone number
#define FLAME_SENSOR_PIN 2          // Pin connected to the flame sensor
#define DHT_PIN A1                  // Pin connected to the DHT sensor
#define DHT_TYPE DHT11              // DHT11 sensor type
#define GAS_PIN A0                  // Pin connected to the gas sensor
#define BUZZER_PIN 4                // Pin connected to the buzzer

// Threshold Definitions
#define GAS_THRESHOLD 300           // Threshold for gas level
#define TEMPERATURE_THRESHOLD 50    // Threshold for temperature in °C
#define HUMIDITY_THRESHOLD 30       // Threshold for humidity in %

SoftwareSerial sim808Serial(PIN_RX, PIN_TX);
DHT dht(DHT_PIN, DHT_TYPE);         // Initialize DHT sensor
```

Arduino Program (contd.)



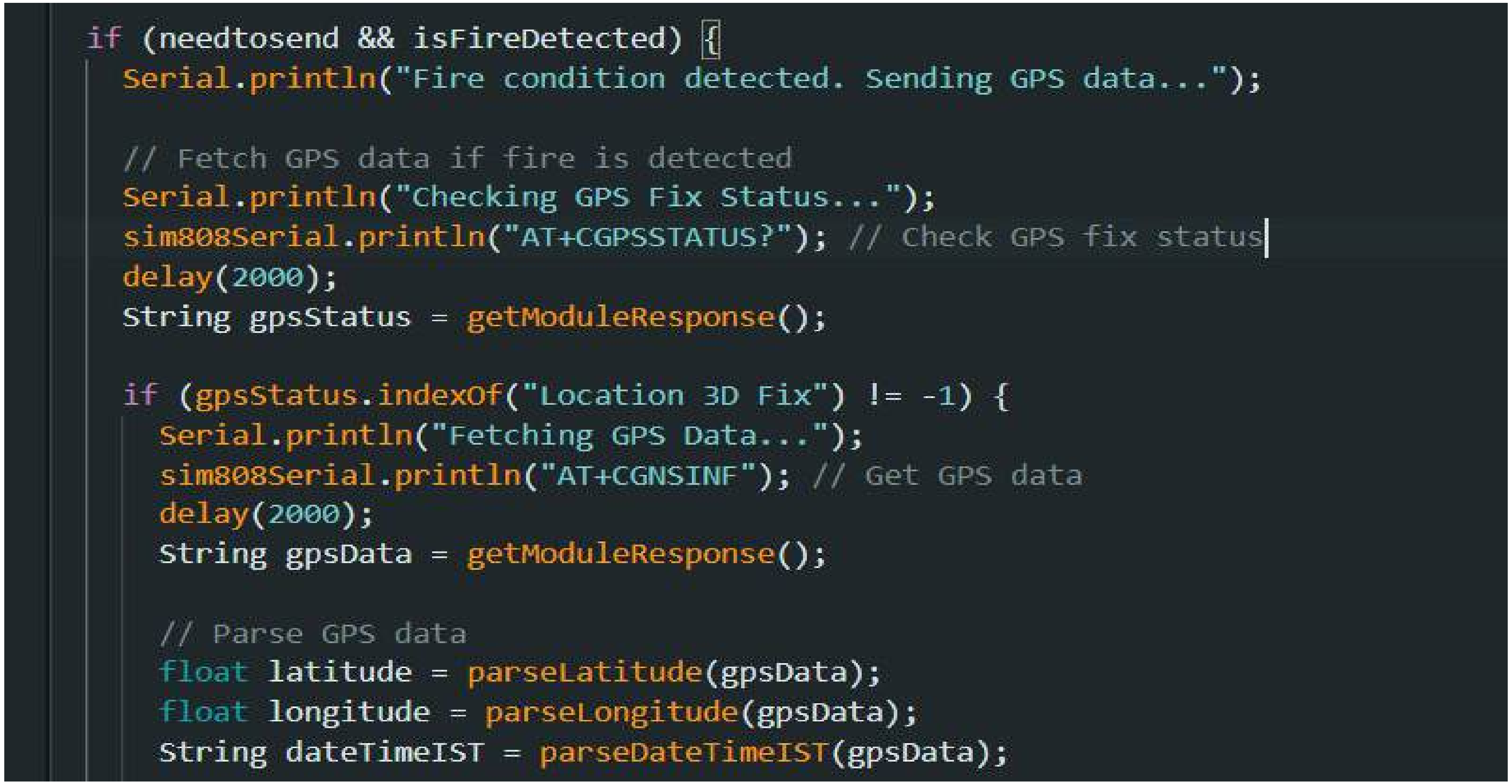
```
void setup() {
  Serial.begin(9600);                    //baud rate assigned
  sim808Serial.begin(9600);              //baud rate assigned
  pinMode(FLAME_SENSOR_PIN, INPUT);      // pin assigned for flame sensor
  pinMode(GAS_PIN, INPUT);               // pin assigned for gas sensor
  pinMode(BUZZER_PIN, OUTPUT);           // pin assigned for buzzer
  dht.begin();

  Serial.println("Initializing GPS...");
  sim808Serial.println("AT+CGNSPWR=1"); // Turn on GPS
  delay(1000);
}
```

Arduino Program (contd.)



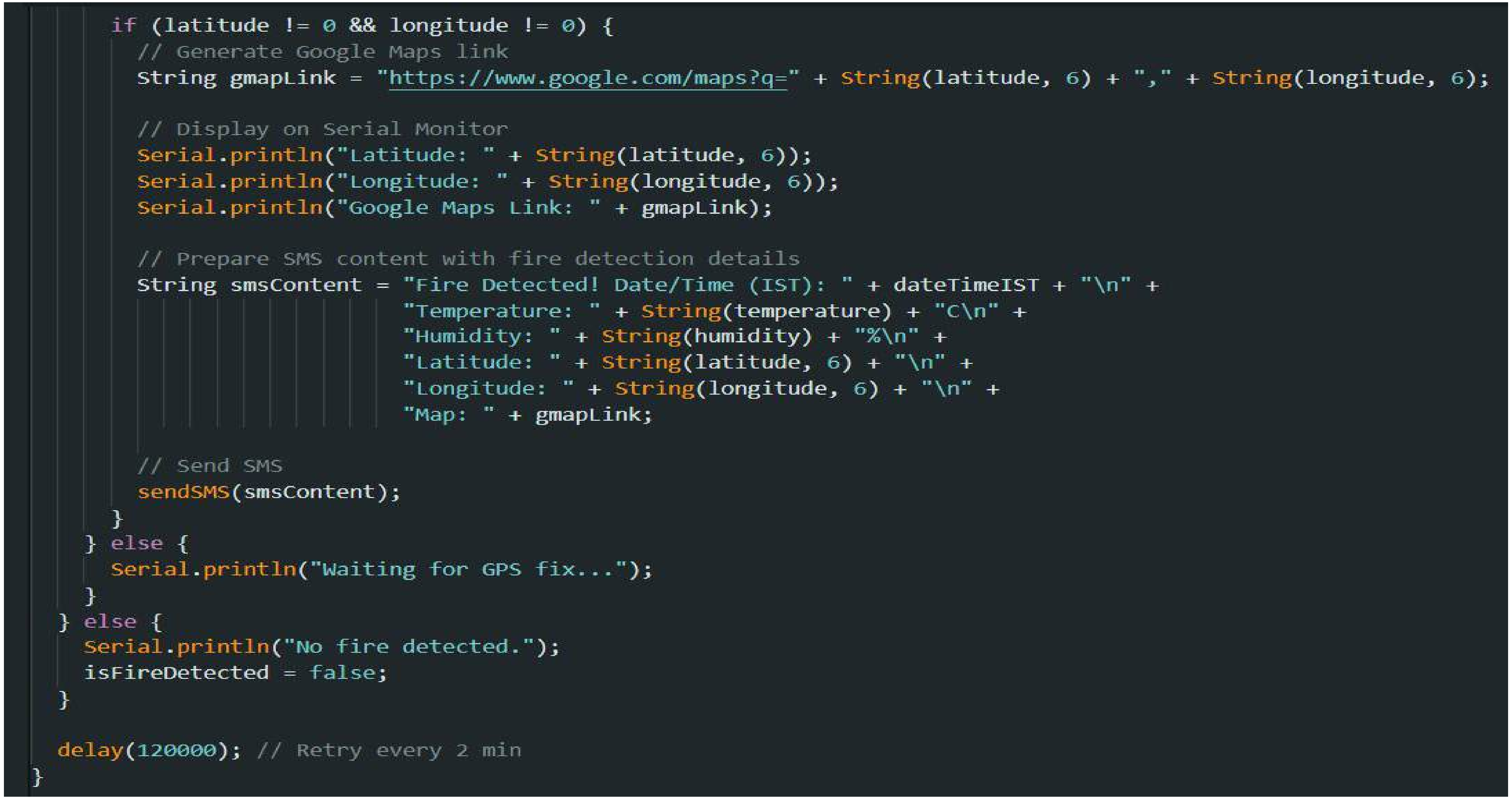
```
void loop() {
  bool needtosend = true;
  int flameSensorValue = digitalRead(FLAME_SENSOR_PIN); // Read flame sensor
  int gasLevel = analogRead(GAS_PIN);                   // Read Gas sensor
  float temperature = dht.readTemperature();           // Read temperature
  float humidity = dht.readHumidity();                 // Read humidity

  if (isFireDetected == true) {
    needtosend = false;
  }
  if (isFireDetected == false) {
    needtosend = true;
  }

  // Check Fire Conditions
  if (flameSensorValue == LOW || gasLevel > GAS_THRESHOLD ||
      temperature > TEMPERATURE_THRESHOLD || humidity < HUMIDITY_THRESHOLD) {
    isFireDetected = true; // Fire detected
    digitalWrite(BUZZER_PIN, HIGH);
    delay(2000);
    digitalWrite(BUZZER_PIN, LOW);
  }
```

Arduino Program (contd.)



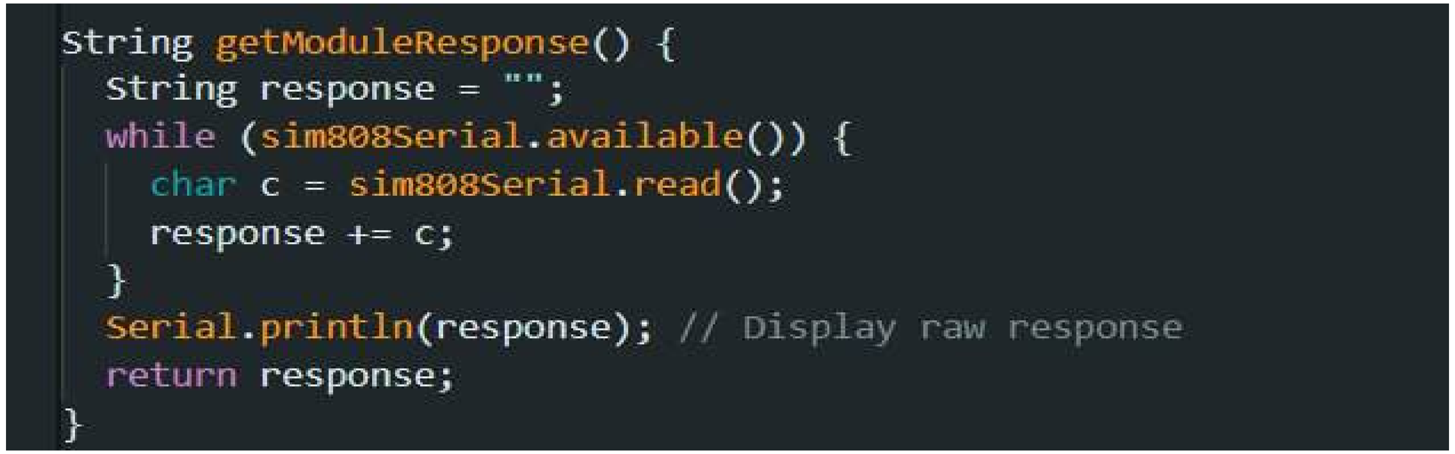
```
if (needtosend && isFireDetected) {
  Serial.println("Fire condition detected. Sending GPS data...");

  // Fetch GPS data if fire is detected
  Serial.println("Checking GPS Fix Status...");
  sim808Serial.println("AT+CGPSSTATUS?"); // Check GPS fix status
  delay(2000);
  String gpsStatus = getModuleResponse();

  if (gpsStatus.indexOf("Location 3D Fix") != -1) {
    Serial.println("Fetching GPS Data...");
    sim808Serial.println("AT+CGNSINF"); // Get GPS data
    delay(2000);
    String gpsData = getModuleResponse();

    // Parse GPS data
    float latitude = parseLatitude(gpsData);
    float longitude = parseLongitude(gpsData);
    String dateTimeIST = parseDateTimeIST(gpsData);
```

Arduino Program (contd.)

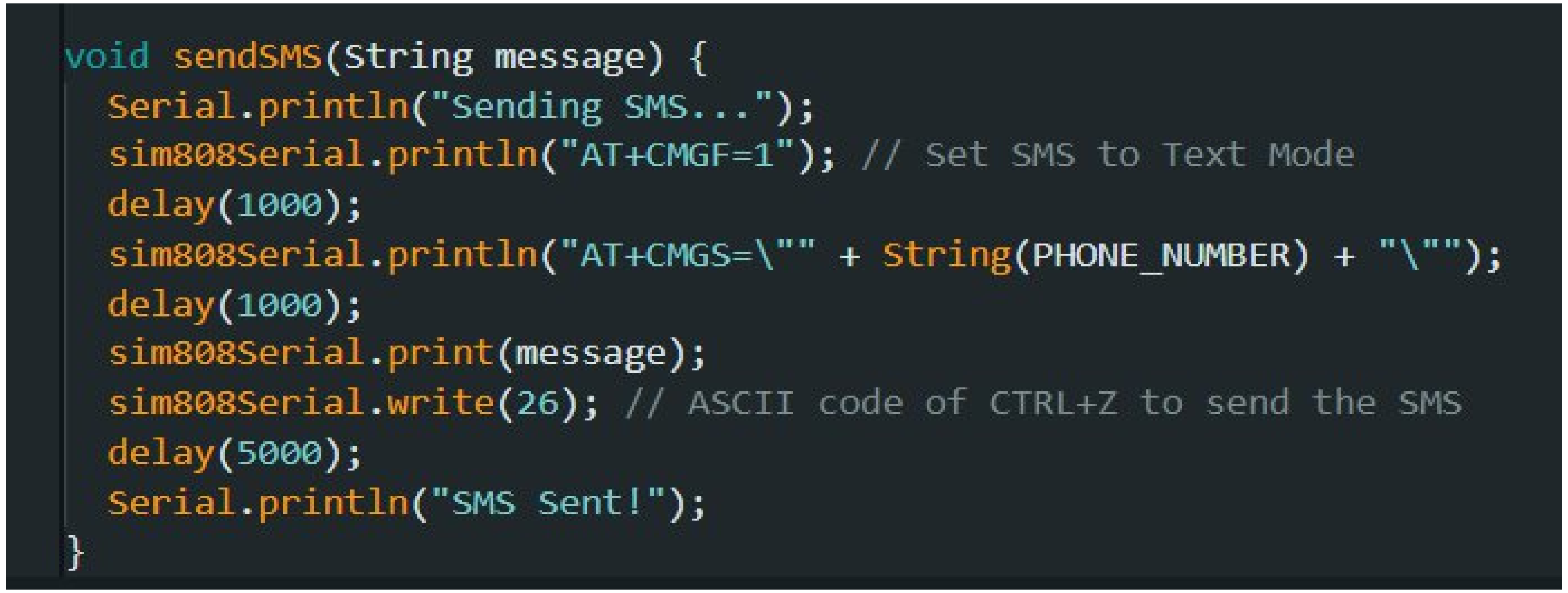


```
      if (latitude != 0 && longitude != 0) {
        // Generate Google Maps link
        String gmapLink = "https://www.google.com/maps?q=" + String(latitude, 6) + "," + String(longitude, 6);

        // Display on Serial Monitor
        Serial.println("Latitude: " + String(latitude, 6));
        Serial.println("Longitude: " + String(longitude, 6));
        Serial.println("Google Maps Link: " + gmapLink);

        // Prepare SMS content with fire detection details
        String smsContent = "Fire Detected! Date/Time (IST): " + dateTimeIST + "\n" +
                            "Temperature: " + String(temperature) + "C\n" +
                            "Humidity: " + String(humidity) + "%\n" +
                            "Latitude: " + String(latitude, 6) + "\n" +
                            "Longitude: " + String(longitude, 6) + "\n" +
                            "Map: " + gmapLink;

        // Send SMS
        sendSMS(smsContent);
      }
    } else {
      Serial.println("Waiting for GPS fix...");
    }
  } else {
    Serial.println("No fire detected.");
    isFireDetected = false;
  }

  delay(120000); // Retry every 2 min
}
```

Arduino Program (contd.)

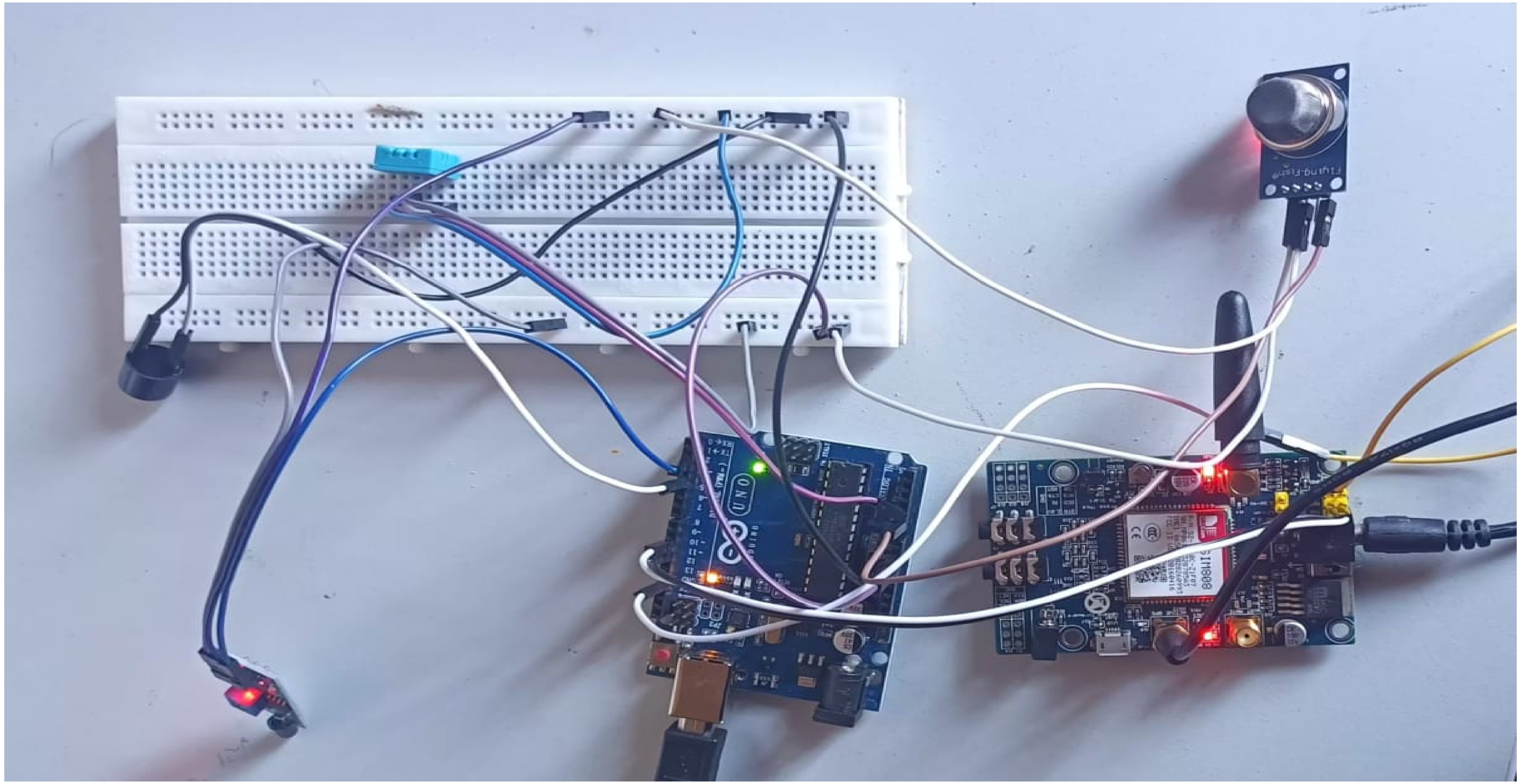


```
String getModuleResponse() {
  String response = "";
  while (sim808Serial.available()) {
    char c = sim808Serial.read();
    response += c;
  }
  Serial.println(response); // Display raw response
  return response;
}
```

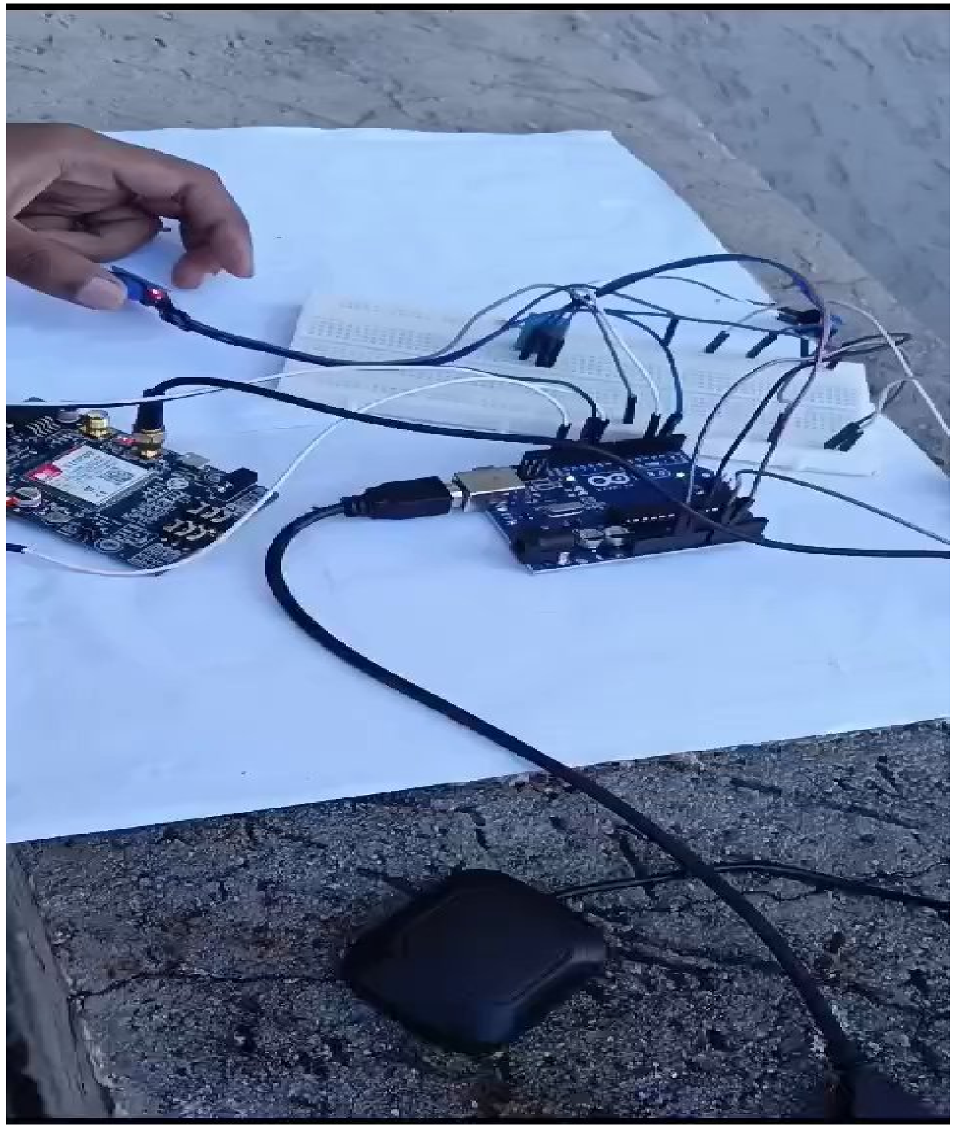


```
void sendSMS(String message) {
  Serial.println("Sending SMS...");
  sim808Serial.println("AT+CMGF=1"); // Set SMS to Text Mode
  delay(1000);
  sim808Serial.println("AT+CMGS=\"" + String(PHONE_NUMBER) + "\"");
  delay(1000);
  sim808Serial.print(message);
  sim808Serial.write(26); // ASCII code of CTRL+Z to send the SMS
  delay(5000);
  Serial.println("SMS Sent!");
}
```

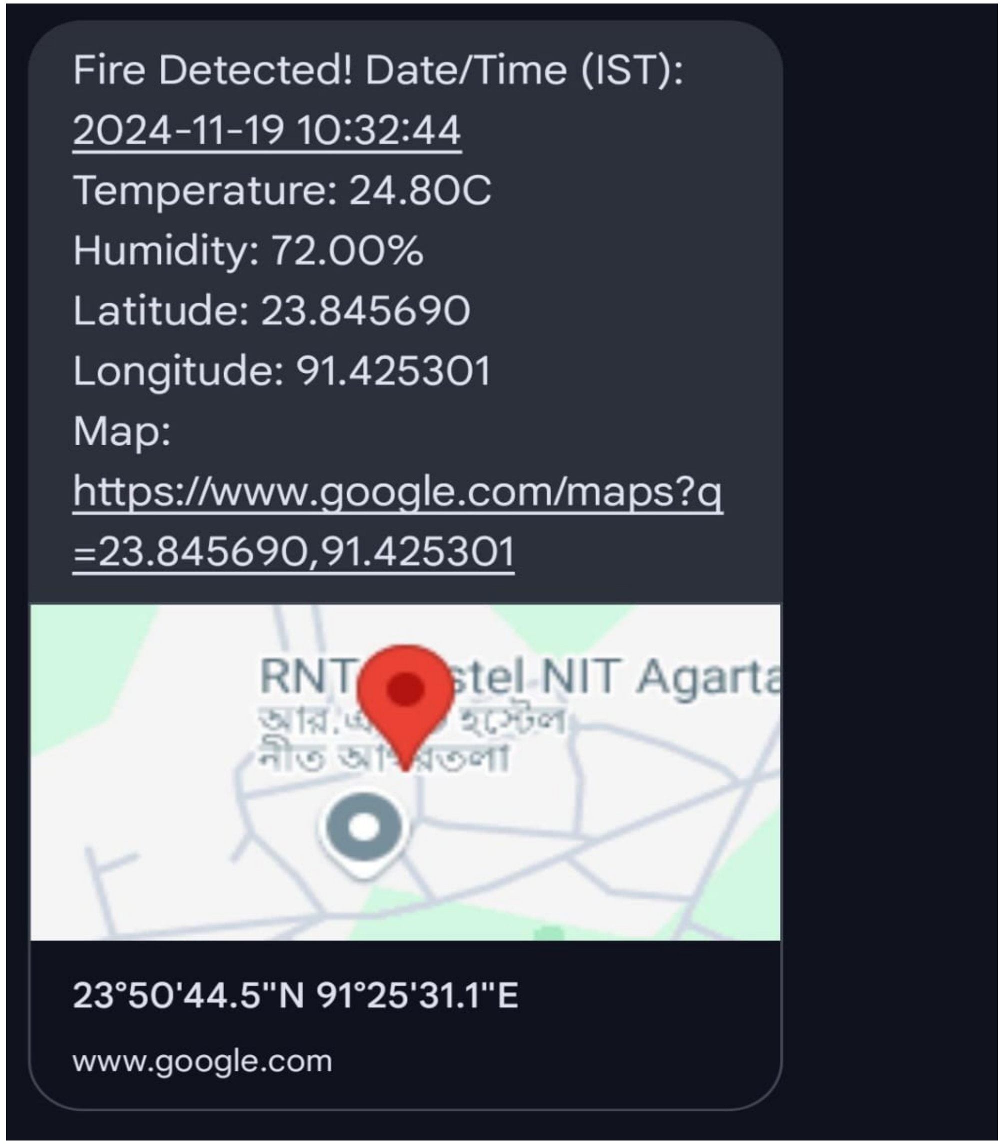
Practical Hardware Demonstration



Practical Working (Video)



SMS Alert Received on Mobile



Fire Detected! Date/Time (IST): 2024-11-19 10:32:44
Temperature: 24.80C
Humidity: 72.00%
Latitude: 23.845690
Longitude: 91.425301
Map:
https://www.google.com/maps?q=23.845690,91.425301

23°50'44.5"N 91°25'31.1"E

www.google.com

Challenges

Sensor Calibration: Achieving accurate threshold values for sensors in varying environmental conditions.

Power Supply: SIM808 GSM/GPS/GPRS module requires enough power to send the location.

Integration Complexity: Ensuring seamless communication between GSM, GPS, and the microcontroller.

Limitations

01

Limited Coverage: Effective only for small-scale monitoring; requires scaling for larger forest areas.

02

Dependency on GSM Network: Alerts rely on the availability and strength of mobile networks.

03

Energy Dependence: Continuous operation requires a reliable power source or solar integration for remote areas.

04

False Alarms: May trigger alarms due to non-fire heat or smoke sources without advanced filtering mechanisms.

Future Directions

Prediction Capabilities: *Planning to integrate AI models and ML algorithms to predict fire risks based on historical data and real-time weather conditions.*

LoRa Communication: *Add LoRa communication to send data or SMS over long distances with low power . Making it useful in remote forest areas.*

Solar Powered: *Implement solar-powered systems for remote operation.*

Scalability: *Deploy a network of sensors for large-area coverage.*

Conclusion

Successfully developed a **Forest Fire Alarm System** with real-time detection and alert features using sensors and GSM/GPRS/GPS module.

Aims to minimize fire damage by providing **early warnings** and enhancing disaster response.

Thus, contributed towards the **sustainable** environmental protection and disaster management.

And the future vision includes **prediction capabilities** and **IoT** integration for greater scalability and efficiency.

THANK YOU