Fire Alarm System with Data Logging Capabilities

Course Code: ESE 4126
Course Name: Sessional on ESE 4125

Lab Report

Submitted by:
Mahmud Abdul Aziz Rohan
Roll No: 2013021
Year: 4th Term: 1st
Department of Energy Science and Engineering
Khulna University of Engineering & Technology

Submitted to:
Dr. Md. Hasan Ali
Professor
&
Mostafizur Rahaman
Lecturer
Department of Energy Science and Engineering
Khulna University of Engineering & Technology
Khulna-9203, Bangladesh

Date of Submission: November 6, 2025

1 Introduction

Fire incidents represent a pervasive and perishing threat to both human life and property across residential, commercial, and industrial settings [1]. The primary factor to initiate a fast response and reduce damage requires immediate and dependable detection of fire events [2]. Traditional fire detection systems often lack the real-time monitoring and advanced diagnostic capabilities that modern embedded systems can provide [3]. The combination of inexpensive dependable microcontrollers with sensor systems enables the creation of cost-effective fire safety solutions which deliver both accessibility and operational efficiency [3].

This project aims to design an Intelligent Fire Alarm System utilizing the Arduino Uno microcontroller, a widely recognized and easily programmable platform based on the ATmega328P [3]. The system includes multiple fire detection sensors, such as smoke detectors (MQ-2/MQ-7) and temperature sensors (e.g., TMP36), to monitor the environment for signs of a potential fire.

A critical feature of this project is its data logging capability. The system's ability to both capture and save sensor information with event timestamps delivers superior benefits when compared to conventional alarm systems [4]. Data logging enables permanent system observation which supports environmental condition analysis before alarm activation and helps decrease false alarms by tracking sensor data patterns over time [3]. The collected data serves multiple purposes such as system diagnostics and threshold optimization and post-incident analysis.

The primary objective of this report is to detail the design, construction, and testing of the fire alarm system, demonstrating the effective integration of the Arduino Uno, sensor components, and a data logging mechanism to create a reliable, low-cost safety solution.

2 Objectives

Fire alarm systems are essential safety equipments. Making them capable of logging data will enable safety officers to analyze the fire hazard and risk in any particular environment for a specific period of time and will allow them to make predictions about them. Keeping this in context, the objectives of this experiment are

i to design a fire alarm system using Arduino UNO in Tinkercad

ii to incorporate data logging methods with the fire alarm system

iii to extract the logged data from the system

3 Fire Alarm System with Data Logging Capabilities

3.1 Arduino UNO R3

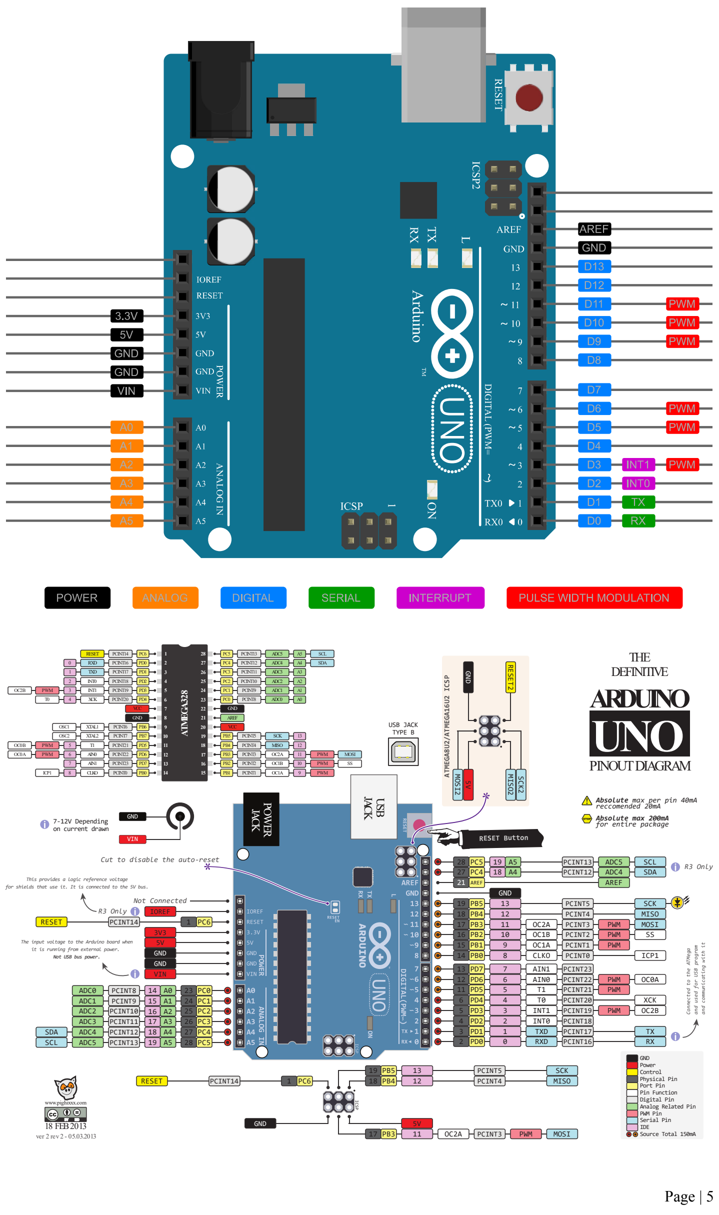


Fig. 1: Arduino UNO R3

3.2 System Description

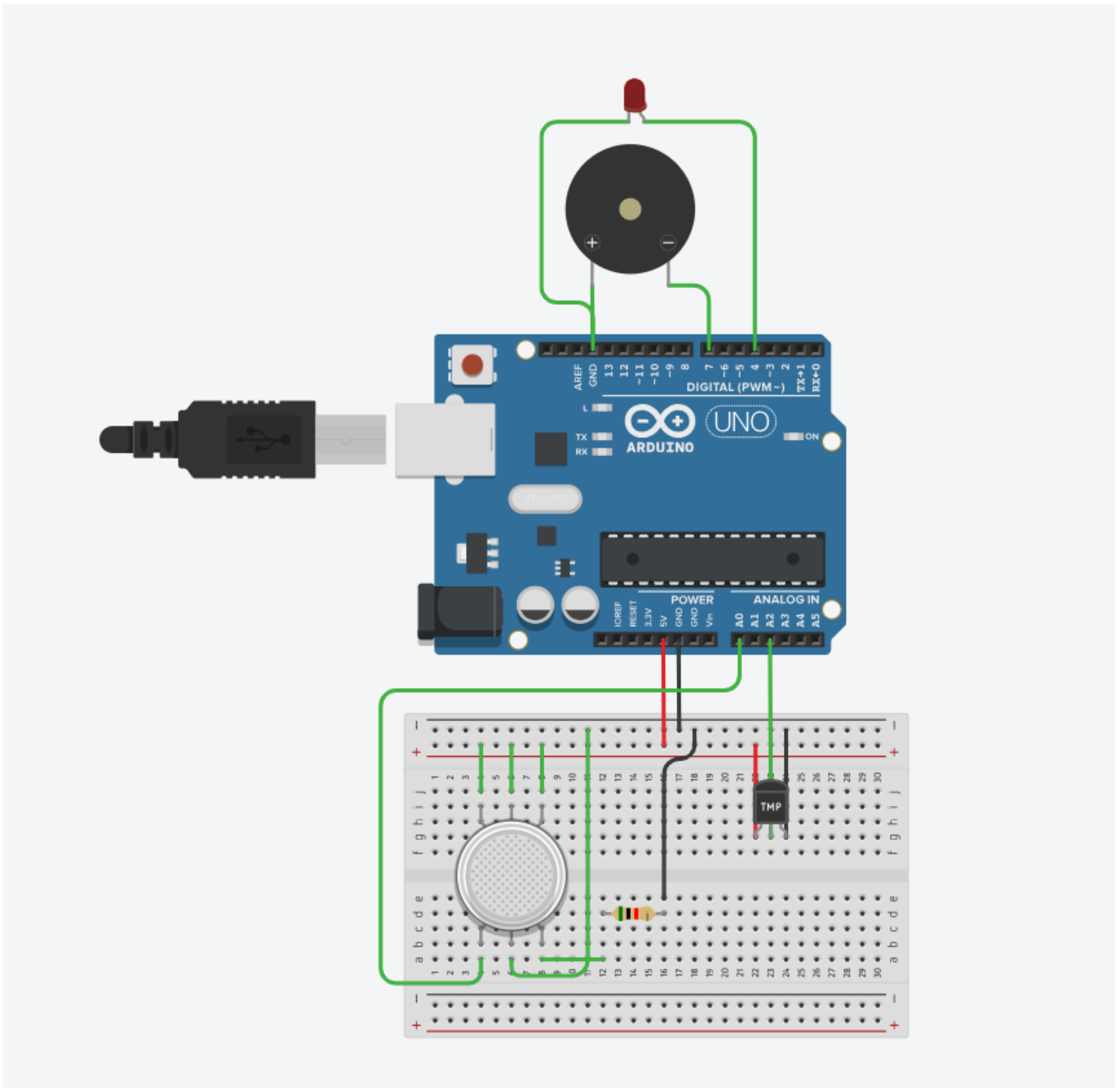


Fig. 2: Diagram of the setup in Tinkercad

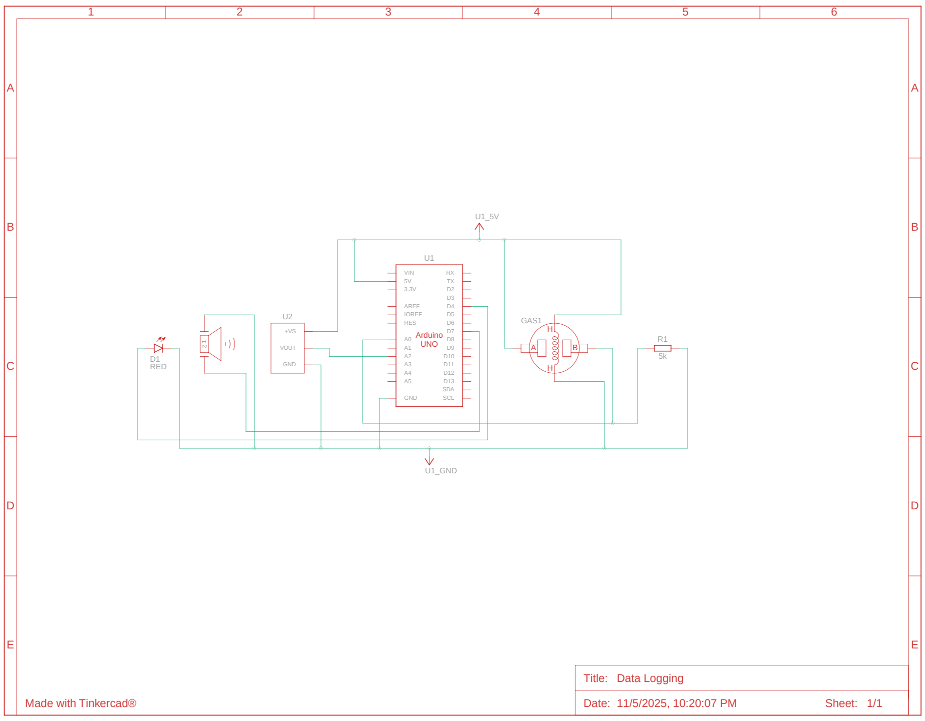


Fig. 3: Schematic diagram of the circuit used in the project

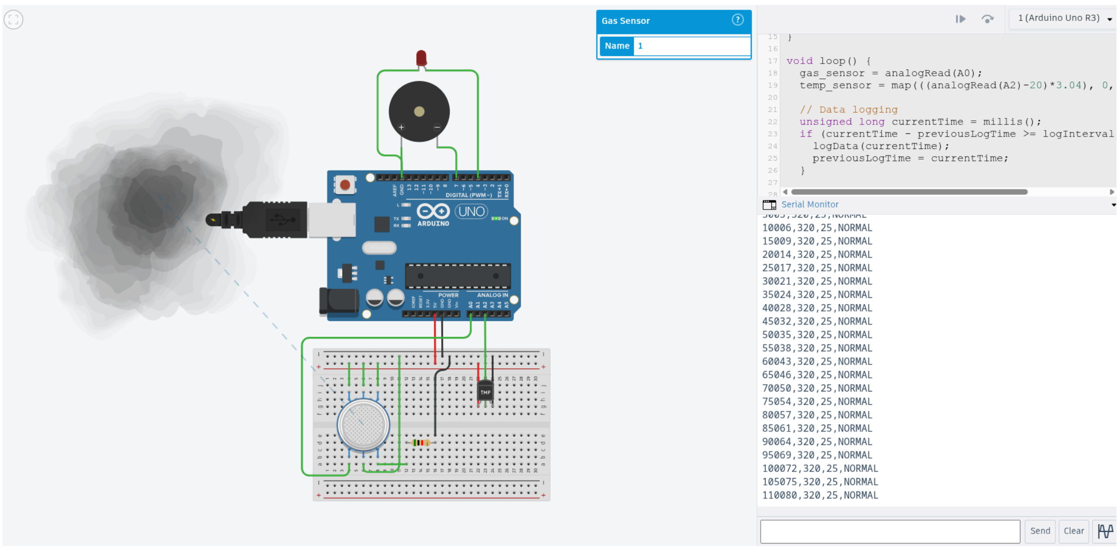
3.3 Code

```
int gas_sensor = 0;
int temp_sensor = 0;
unsigned long previousLogTime = 0;
const unsigned long logInterval = 5000;  // Log every 5 seconds

void setup() {
  pinMode(A0, INPUT);
  pinMode(A2, INPUT);
  Serial.begin(9600);
  pinMode(7, OUTPUT);
  pinMode(4, OUTPUT);

  // Print CSV headers
  Serial.println("Timestamp,Gas_Reading,Temperature,Status");
}

void loop() {
  gas_sensor = analogRead(A0);
  temp_sensor = map(((analogRead(A2)-20)*3.04), 0, 1023, -40, 125);

  // Data logging
  unsigned long currentTime = millis();
  if (currentTime - previousLogTime >= logInterval) {
    logData(currentTime);
    previousLogTime = currentTime;
  }

  // Alarm system
  if (gas_sensor >= 400 || temp_sensor >= 30) {
    tone(7, 200005, 100);
    digitalWrite(4, HIGH);
  } else {
    digitalWrite(7, LOW);
    digitalWrite(4, LOW);
  }

  delay(1000);
}

void logData(unsigned long timestamp) {
  Serial.print(timestamp);
  Serial.print(",");
  Serial.print(gas_sensor);
  Serial.print(",");
  Serial.print(temp_sensor);
  Serial.print(",");

  if (gas_sensor >= 400 || temp_sensor >= 30) {
    Serial.println("ALERT");
  } else {
    Serial.println("NORMAL");
  }
}
```

Listing 1: sketch.ino

The `setup()` function is used for initilizing the input and output ports. The `loop()` function runs with a delay of 1 s declared in the last line of the function as `delay(1000)`. The conditions for the alart signals are either the gas sensor readings should be above 400 or the temperature reading should be above 30 degree celsius or both. The `logData(unsigned long timestamp)` function is used for logging the data to the serial monitor. From this monitor the logged output can be copied and used as a CSV file for further analysis. The time lag between each data logging is set at 5000 ms is the begining of the code as a global variable `const unsigned long logInterval`.

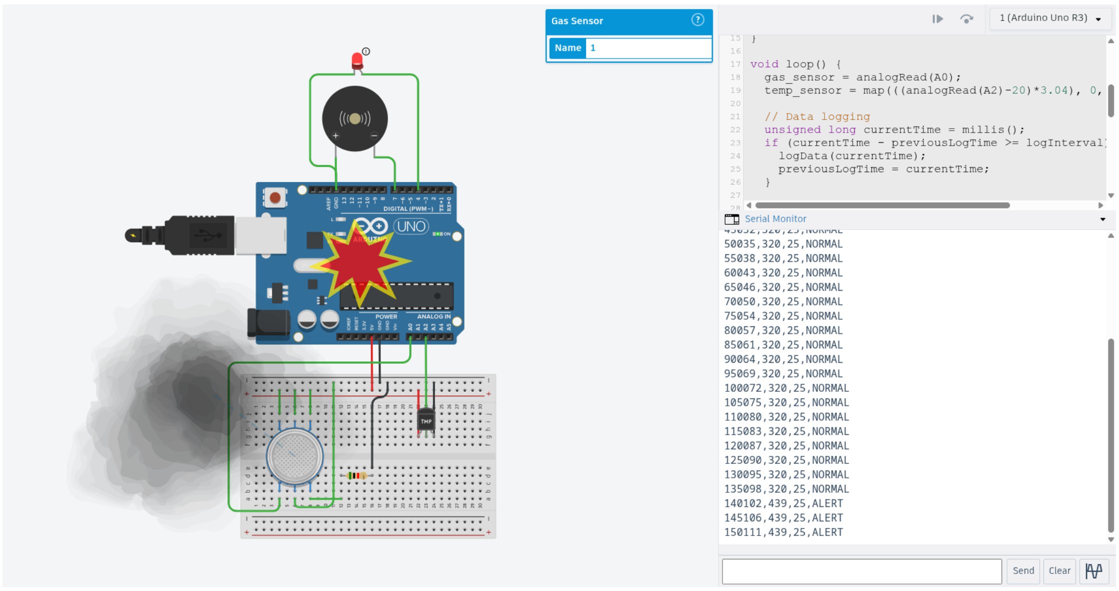
4 Result



```
}

void loop() {
  gas_sensor = analogRead(A0);
  temp_sensor = map(((analogRead(A2)-20)*3.04), 0,

  // Data logging
  unsigned long currentTime = millis();
  if (currentTime - previousLogTime >= logInterval
    logData(currentTime);
    previousLogTime = currentTime;
  }
```

```
10006,320,25,NORMAL
15009,320,25,NORMAL
20014,320,25,NORMAL
25017,320,25,NORMAL
30021,320,25,NORMAL
35024,320,25,NORMAL
40028,320,25,NORMAL
45032,320,25,NORMAL
50035,320,25,NORMAL
55038,320,25,NORMAL
60043,320,25,NORMAL
65046,320,25,NORMAL
70050,320,25,NORMAL
75054,320,25,NORMAL
80057,320,25,NORMAL
85061,320,25,NORMAL
90064,320,25,NORMAL
95069,320,25,NORMAL
100072,320,25,NORMAL
105075,320,25,NORMAL
110080,320,25,NORMAL
```

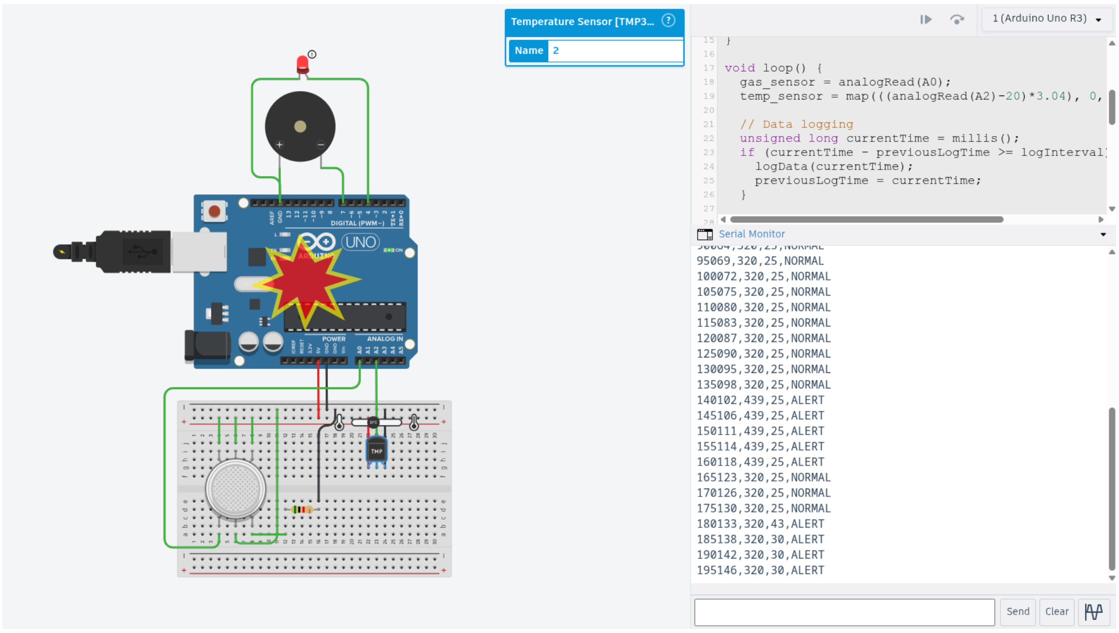
Fig. 4: Fire Alarm System under normal conditions



```
}

void loop() {
  gas_sensor = analogRead(A0);
  temp_sensor = map(((analogRead(A2)-20)*3.04), 0,

  // Data logging
  unsigned long currentTime = millis();
  if (currentTime - previousLogTime >= logInterval)
    logData(currentTime);
    previousLogTime = currentTime;
  }
```

```
50035,320,25,NORMAL
55038,320,25,NORMAL
60043,320,25,NORMAL
65046,320,25,NORMAL
70050,320,25,NORMAL
75054,320,25,NORMAL
80057,320,25,NORMAL
85061,320,25,NORMAL
90064,320,25,NORMAL
95069,320,25,NORMAL
100072,320,25,NORMAL
105075,320,25,NORMAL
110080,320,25,NORMAL
115083,320,25,NORMAL
120087,320,25,NORMAL
125090,320,25,NORMAL
130095,320,25,NORMAL
135098,320,25,NORMAL
140102,439,25,ALERT
145106,439,25,ALERT
150111,439,25,ALERT
```

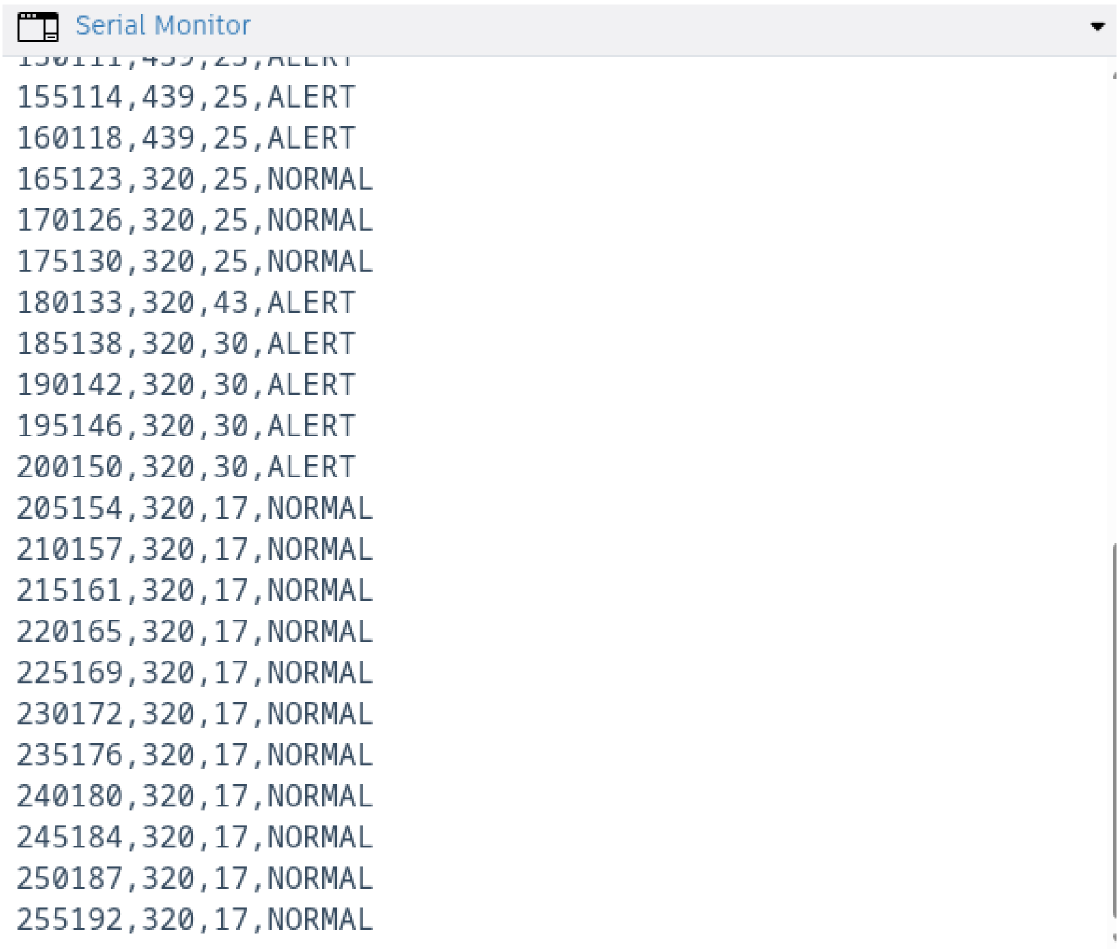
Fig. 5: Alarm engaged because of excess amount of gas



```
}

void loop() {
  gas_sensor = analogRead(A0);
  temp_sensor = map(((analogRead(A2)-20)*3.04), 0,

  // Data logging
  unsigned long currentTime = millis();
  if (currentTime - previousLogTime >= logInterval
    logData(currentTime);
    previousLogTime = currentTime;
  }
```

```
95069,320,25,NORMAL
100072,320,25,NORMAL
105075,320,25,NORMAL
110080,320,25,NORMAL
115083,320,25,NORMAL
120087,320,25,NORMAL
125090,320,25,NORMAL
130095,320,25,NORMAL
135098,320,25,NORMAL
140102,439,25,ALERT
145106,439,25,ALERT
150111,439,25,ALERT
155114,439,25,ALERT
160118,439,25,ALERT
165123,320,25,NORMAL
170126,320,25,NORMAL
175130,320,25,NORMAL
180133,320,43,ALERT
185138,320,30,ALERT
190142,320,30,ALERT
195146,320,30,ALERT
```

Fig. 6: Alarm engaged because of high temperature



```
155114,439,25,ALERT
160118,439,25,ALERT
165123,320,25,NORMAL
170126,320,25,NORMAL
175130,320,25,NORMAL
180133,320,43,ALERT
185138,320,30,ALERT
190142,320,30,ALERT
195146,320,30,ALERT
200150,320,30,ALERT
205154,320,17,NORMAL
210157,320,17,NORMAL
215161,320,17,NORMAL
220165,320,17,NORMAL
225169,320,17,NORMAL
230172,320,17,NORMAL
235176,320,17,NORMAL
240180,320,17,NORMAL
245184,320,17,NORMAL
250187,320,17,NORMAL
255192,320,17,NORMAL
```

Fig. 7: Logged data in the serial monitor in CSV format

5 Discussion

The results show that the fire alarm system is functional in the simulated conditions. Figure 4 is showing the normal conditions when the alarm is not yet engaged. In Figure 5 and Figure 6, the alarm system is engaged because of excess gas and high temperature. This confirms the expected behaviour of the system under actual conditions. The data from the alarm system can be collected from the serial monitor as shown in Figure 7. This data can be anayzed using various tools for future predictions of abnormal conditions, or detecting anomaly in the system.

6 Conclusion

A fire alarm system was successfully designed in Tinkercad using Arduino UNO R3. Data logging was also incorporated using the serial print strategy. The data can be extracted from the serial monitor by copying from the monitor and storing it in a file. Further modifications can be made by using SD card module to store the data in a file.

References

[1] S. Suwarjono et al., “Design of a home fire detection system using arduino and sms gateway,” *Knowledge*, vol. 1, pp. 61–74, 1 2021. DOI: 10.3390/knowledge1010007.

[2] R. F. Siregar, A. Affandi, R. Rohana, A. R. Nasution, and I. Tanjung, “Iot smart control system: Smoke and fire detection using sim900a module,” *Journal of Electrical Technology UMY*, vol. 7, pp. 48–56, 2 2024. DOI: 10.18196/jet.v7i2.19908.

[3] B. Sarwar, I. S. Bajwa, N. Jamil, S. Ramzan, and N. Sarwar, “An intelligent fire warning application using iot and an adaptive neuro-fuzzy inference system,” *Sensors*, vol. 19, p. 3150, 14 2019. DOI: 10.3390/s19143150.

[4] D. L. Wilson, K. N. Williams, and A. Pillarisetti, “An integrated sensor data logging, survey, and analytics platform for field research and its application in hapin, a multi-center household energy intervention trial,” *Sustainability*, vol. 12, p. 1805, 5 2020. DOI: 10.3390/su12051805.