Multi Gas Air Detection in Atmosphere with Visual Feedback for Quality Status

Dr. G. Sundar[1], K. Ganga Durai[2], P. Karthi[3], T. Karthik[4], K. S. Siva Rajan[5]

[1]Professor and Head of the Department, Sri Shakthi Institute of Engineering & Technology, Coimbatore-641062

[2,3,4,5]Student, B.E. Electrical and Electronics Engineering, Sri Shakthi Institute of Engineering & Technology, Coimbatore-641062

Abstract- The rise in air pollution levels poses serious threats to human health, the environment, and industrial safety. Monitoring the presence of hazardous gases in real-time has become essential to ensure a safe living and working environment. The Multi Gas Air Detection System with Visual Feedback is designed to detect and display the concentration levels of multiple harmful gases in the surrounding air using an efficient and cost-effective setup. It interfaces with various MQ series gas sensors to monitor gas concentrations continuously. These sensors provide analog signals proportional to the gas levels, which are processed by the ESP32. For real-time visualization, the system uses an I2C OLED display (128x64 resolution) to show gas concentration values in parts per million (PPM). Additionally, the system provides visual feedback using color codes or warning icons on the display to indicate air quality status, ranging from safe to hazardous levels. Thresholds are programmed to alert users when dangerous concentrations are detected.

Keywords- Real-Time Detection, Air Quality Monitoring, IOT, Pollution Monitoring

I. INTRODUCTION

Air pollution has become one of the most critical environmental issues of modern times, posing serious threats to human health, ecosystems, and climate. Exposure to harmful gases such as carbon monoxide (CO), methane (CH_4), ammonia (NH_3), LPG, and other toxic substances can cause respiratory problems, poisoning, and even fatal accidents in industries and homes. Hence, continuous monitoring of air quality is essential to ensure a safe environment.

The Multi Gas Air Detection System with Visual Feedback is developed as a practical solution for detecting multiple harmful gases in real-time and providing immediate visual indications to the user. The system primarily uses MQ series gas sensors, which are cost-effective and capable of detecting various gases like CO, smoke, methane, and overall air quality. These sensors produce analog outputs based on gas concentrations. The core of the system is an ESP32 microcontroller, chosen for its high processing speed, integrated Wi-Fi, and multiple GPIOs. It reads sensor data, processes it, and drives the visual output. For displaying real-time gas concentration levels, an I2C OLED display (128x64) is used. This compact display shows numeric values (PPM) and visual indicators such as icons, bars, or color codes, making the system user-friendly and informative. The system is designed to be portable, energy-efficient, and scalable, making it suitable for a wide range of applications, including homes, industries, laboratories, parking areas, and public spaces. Threshold levels are set in the software to trigger alerts when dangerous gas levels are detected, ensuring quick preventive actions. In summary, this project aims to provide a simple, affordable, and effective air quality monitoring solution by combining ESP32, gas sensors, and visual feedback via OLED display, thus contributing to environmental safety and public health awareness

II. MATERIALS AND METHODS

The development of the Multi Gas Air Detection System with Visual Feedback involved several key hardware components and a systematic approach to integration. The primary controller used was the ESP8266 NodeMCU, chosen for its built-in Wi-Fi capability and analog input pin. The system utilized three gas sensors: the MQ2 sensor to detect smoke and LPG, the MQ7 sensor for carbon monoxide, and the MQ135 sensor for air pollutants like ammonia and benzene. Since the ESP8266 has only one analog input, a 74HC4051 analog multiplexer was used to switch between sensor signals, allowing all three sensors to be connected and read using a single pin. The gas concentration data was displayed in real-time on a 16x2 I2C LCD display, which simplified wiring and made data more readable. A 9V battery was used as the main power supply, with a 7805 voltage regulator ensuring a consistent 5V supply to all components. The entire circuit was assembled on a breadboard using jumper wires to enable easy prototyping and testing. The ESP8266 was programmed to sequentially read sensor values, process the data, and display the gas concentrations on the LCD screen at regular intervals, enabling real-time air quality monitoring.

Sensor Connection: MQ2, MQ7, and MQ135 sensors are connected to the 74HC4051 multiplexer, which routes one sensor signal at a time to the ESP8266's analog pin.
Microcontroller Setup: The ESP8266 NodeMCU is programmed to read the analog signals from the multiplexer, convert them to gas concentration values, and update the display accordingly.
Display Output: The processed values are sent to a 16x2 I2C LCD, which shows the gas name and concentration in real time.
Power Supply: A 9V battery supplies power, and a 7805 regulator ensures that the components receive a stable 5V supply.
Real-Time Monitoring: The system continuously monitors air quality and updates sensor values every few seconds on the LCD.

III. SOFTWARE FLOW

A. Coding

```
 #include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);
#define S0 14
#define S1 12
#define S2 13
#define E_PIN -1
#define ANALOG_PIN A0

const char* sensorNames[] = {"MQ135", "MQ7",
"MQ2"};
#define NUM_SENSORS 3

void setup() {
  Serial.begin(115200);
  lcd.init();
  lcd.backlight();
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  if (E_PIN != -1) {
    pinMode(E_PIN, OUTPUT);
    digitalWrite(E_PIN, LOW);
  }
  lcd.clear();
}

void loop() {
  for (int channel = 0; channel < NUM_SENSORS;
channel++) {
    selectMuxChannel(channel);
    delay(100);

    int sensorValue = analogRead(ANALOG_PIN);
    String airStatus =
getAirQualityStatus(sensorValue);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(sensorNames[channel]);
    lcd.print(": ");
    lcd.print(sensorValue);
    lcd.setCursor(0, 1);
    lcd.print("Status: ");
    lcd.print(airStatus);
    Serial.print(sensorNames[channel]);
    Serial.print(" = ");
    Serial.print(sensorValue);
    Serial.print(", Status: ");
    Serial.println(airStatus);
    delay(3000);
  }
}

void selectMuxChannel(int channel) {
  digitalWrite(S0, bitRead(channel, 0));
  digitalWrite(S1, bitRead(channel, 1));
  digitalWrite(S2, bitRead(channel, 2));
}

String getAirQualityStatus(int val) {
  if (val < 200) return "Good";
  else if (val < 400) return "Moderate";
  else return "Bad";
}
```

B. Explanation

1. Library Inclusions and LCD Initialization.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);
```

-The Wire.h library manages I2C communication.
-The LiquidCrystal_I2C.h library controls the I2C LCD display.
-The LCD is initialized at I2C address 0x27 with 16 columns and 2 rows.

2. Multiplexer Control Pins and Sensor Names

```
#define S0 14
#define S1 12
#define S2 13
#define E_PIN -1
#define ANALOG_PIN A0

const char* sensorNames[] = {"MQ135", "MQ7", "MQ2"};
#define NUM_SENSORS 3
```

-S0, S1, and S2 are GPIO pins controlling the 3-bit address input of the 74HC4051 multiplexer, selecting which sensor output is fed to the microcontroller's analog pin (ANALOG_PIN).

-E_PIN is set to -1 meaning the enable pin is not used or disabled.
-sensorNames stores the names of the three gas sensors for display.
-NUM_SENSORS is the total number of sensors connected.

3. Setup Function

```
void setup() {
  Serial.begin(115200);
  lcd.init();
  lcd.backlight();
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  if (E_PIN != -1) {
    pinMode(E_PIN, OUTPUT);
    digitalWrite(E_PIN, LOW);
  }
  lcd.clear();
}
```

-Initializes serial communication at 115200 baud rate for debugging.
-Initializes the LCD and turns on backlight.
-Sets multiplexer control pins as outputs.
-If enable pin was used, sets it as output and disables it (LOW).
-Clears the LCD screen.

4. Main Loop

```
void loop() {
  for (int channel = 0; channel < NUM_SENSORS; channel++) {
    selectMuxChannel(channel);
    delay(100);

    int sensorValue = analogRead(ANALOG_PIN);
    String airStatus = getAirQualityStatus(sensorValue);
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print(sensorNames[channel]);
    lcd.print(": ");
    lcd.print(sensorValue);
    lcd.setCursor(0, 1);
    lcd.print("Status: ");
    lcd.print(airStatus);
    Serial.print(sensorNames[channel]);
    Serial.print(" = ");
    Serial.print(sensorValue);
    Serial.print(", Status: ");
    Serial.println(airStatus);
    delay(3000);
  }
}
```

-Loops through each sensor (0 to 2).
-Calls selectMuxChannel(channel) to set the multiplexer to route the correct sensor signal to the analog input.
-Waits 100 ms for signal stabilization.
-Reads analog value from the sensor via the multiplexer.
-Determines air quality status by calling getAirQualityStatus() with the analog value.
-Clears and updates the LCD with sensor name, reading, and status.

-Prints sensor data with status to the serial port for monitoring.
-Waits 3 seconds before switching to the next sensor.

5. Multiplexer Channel Selection

```
void selectMuxChannel(int channel) {
  digitalWrite(S0, bitRead(channel, 0));
  digitalWrite(S1, bitRead(channel, 1));
  digitalWrite(S2, bitRead(channel, 2));
}
```

-Controls multiplexer pins S0, S1, and S2 to select the sensor channel.
-Uses bitwise operations to set each multiplexer control bit according to the channel number (0-7).

6. Air Quality Status Determination

```
String getAirQualityStatus(int val) {
  if (val < 200) return "Good";
  else if (val < 400) return "Moderate";
  else return "Bad";
}
```

-Converts the raw analog sensor value to a qualitative air quality status.

-Thresholds:

-Less than 200: Good air quality
-200 to 399: Moderate air quality
-400 and above: Bad air quality (higher gas concentration)

IV. RESULTS AND DISCUSSION

It was successfully tested in both indoor and controlled environments. The ESP8266 NodeMCU microcontroller accurately received analog signals from the gas sensors and displayed the processed data on the I2C 16x2 LCD screen in real time. Each sensor—MQ2, MQ7, and MQ135—responded correctly to its target gases. The MQ2 sensor detected the presence of smoke and LPG effectively when exposed to cigarette smoke and lighter gas. The MQ7 sensor showed strong sensitivity to carbon monoxide, responding quickly when held near a vehicle's exhaust. Similarly, the MQ135 sensor provided varying readings when exposed to common air pollutants, such as perfumes and room sprays, indicating changes in air quality.

The inclusion of the 74HC4051 multiplexer played a key role in optimizing the use of the single analog pin of the ESP8266, allowing multiple gas sensors to be connected without hardware conflict. The displayed readings were easy to interpret and updated every few seconds, providing users with real-time air quality status. Minor fluctuations were observed due to environmental changes, which is normal for analog gas sensors.

V. HELPFUL HINTS

A. Figures and Tables

S.No.	Component Name	Type	Qty.
1.	ESP8266 NodeMCU	Wi-Fi Enabled Microcontroller	1
2.	MQ2 Gas Sensor	Smoke, LPG, Methane Detection	1
3.	MQ7 Gas Sensor	Carbon Monoxide (CO) Sensor	1
4.	MQ135 Gas Sensor	Air Quality/ CO2/NH3 Sensor	1
5.	75HC4051 Multiplexer	8-channel Analog Multiplexer	1
6.	16x2 LCD Display(I2C)	I2C Interface 16x2 LCD	1
7.	9V Battery	Power Supply	1
8.	Voltage Regulator	7805(5V Output)	1
9.	BreadBoard	Standard Prototyping Board	1
10.	Jumper Wires	Various types	Few

Table 1: Components and Specifications

Block Diagram

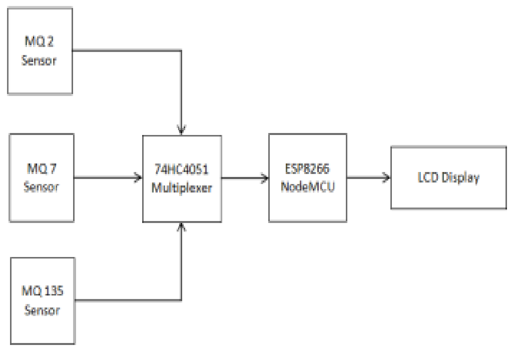

Fig. 1: Block diagram of a gas detection system using MQ2, MQ7, and MQ135 sensors connected via a 74HC4051 multiplexer to an ESP8266 NodeMCU, which processes the data and displays it on an LCD screen

Circuit Diagram

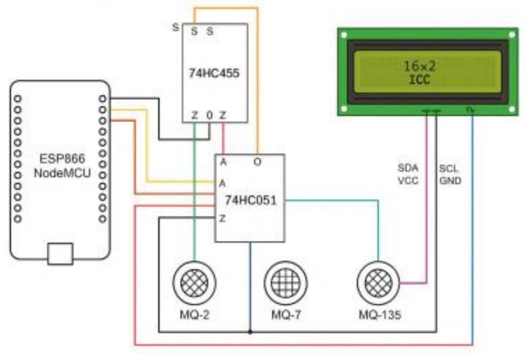

Fig. 2: Circuit diagram of a gas monitoring system using MQ-2, MQ-7, and MQ-135 265 gas sensors interfaced through 74HC455 and 74HC051 multiplexers to an ESP8266 266 NodeMCU, with output displayed on a 16×2 I2C LCD.

VII. CONCLUSION

The Multi Gas Air Detection System with Visual Feedback was designed and implemented successfully using ESP8266 NodeMCU and gas sensors (MQ2, MQ7, and MQ135). The system effectively detected multiple hazardous gases such as LPG, Carbon Monoxide, Smoke, and Air Pollutants, and provided real-time concentration readings via a 16x2 I2C LCD display. This project proved to be an efficient, low-cost, and reliable solution for continuous air quality monitoring. The use of the ESP8266 microcontroller allowed seamless integration of sensor data processing and visual feedback while offering flexibility for future enhancements like wireless data transfer and IoT integration. During testing, each sensor responded accurately to its target gas, and the system's performance demonstrated practical usability in environments such as kitchens, industrial areas, laboratories, and enclosed public spaces where toxic gases can pose serious risks. The visual feedback system makes it easy for even non-technical users to understand air conditions, enhancing safety and awareness.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to all those who supported and guided us throughout the successful completion of our project, "Multi Gas Air Detection System with Visual Feedback." This project would not have been possible without the encouragement, support, and assistance of many individuals.

REFERENCES

[1] IEEE International Geoscience and Remote Sensing Symposium, pp.III- 1370, III- 1373, 2008.

[2] Felstead TJ. The use of a road side remote sensing device to encourage voluntary vehicle emissions related maintenance. SEIG Conference; London'07. 2007. p. 1–18.

[3] N.Kularatna and B. H. Sudantha," An environmental air pollution monitoring system based on the IEEE 1451 standard for low cost requirements," IEEE Sensors J., vol.8, pp. 415422, Apr. 2008.

[4] Young Jin Jung, Yang Koo Lee, Dong Gyu Lee, Keun Ho Ryu, Silvia Nittel" Air pollution monitoring system based on geosensor network,"

[5] O.A.Postolache, J. M. D. Pereira, and P. M. B. S. Girao," Smart sensors network for air quality monitoring applications," IEEE Trans.Instrum. Meas., vol. 58, no. 9, pp. 3253-3262, Sep.2009.

[6] J. Sobota, R. Pisl, P. Balda and M. Schlegel, "Raspberry Pi and Arduino boards in control education", IFAC Proceedings Volumes, vol. 46, no. 17, pp. 7-12, 2013.

[7] L.Welling and L. Thomson, PHP and MySQL Web Development, United States of America: Pearson Education, 2008

[8] P. Traeg, "IoT Projects: Raspberry Pi vs Arduino", Universal Mind, 2015.

[9] Arvind RV, Raj RR, Raj RR, Prakash NK. Industrial automation using Wireless Sensor Networks. Indian Journal of Science and Technology. 2016.

[10] Jung YJ, Lee YK, Lee DG, Lee Y, Nittel S, Beard K, et. al. Design of sensor data processing steps in an air pollution monitoring system. Sensors. 2016.

[11] Peiyuan Sun, Research and Development of Intelligent Home Interior Environment Monitoring System [D], Jilin University of Architecture, 2017

[12] DHT11 temperature and humidity data analysis in Chinese, [online] Available: http://www.51hei.com.

[13] Arduino, "Arduino uno." Last visited on 06/09/2014

[14] Saifudaullah Md, Bahrudin Bin and Kassim. Rosni Abu, "Development of Fire Alarm System using Raspberry Pi and Arduino Uno." in, Malaysia:Faculty of Electrical Engineering University Teknologi MARA Selangor, 2013.

[15] Wolfram Donat, "Learn Raspberry Pi Programming with Python", c2014.

[16] Liu Peng, Fu Danni, Jiang Shengqian and Wang Mingjie, "A Movable Indoor Air Quality Monitoring System", 2nd International Conference on Cybernetics Robotics and Control, pp. 126-129, July 2017.