IOT281: Smoke Detection System

Arduino + TinkerCad (Simulation)

4 September 2025

Aphelele Mdebuka (601706)
Enrique De Sousa (603110)
Jacobus Petrus Geldenhuys (602755)
Migal Groenewald (602974)

Table of Content

Background **Knowledge**

A smoke detector is a very important safety tool. It gives people an early warning when there is smoke or gas in theair. This early warning can save lives because it gives time to escape or stop the fire before it becomes a threat.

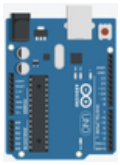
Our project uses the MQ-2 gas sensor with an Arduino Uno. The MQ-2 can pick up smoke and also gases like LPG, propane, methane and hydrogen. When there is a high concentration of smoke/gas in the air, the sensor changes its reading and sends that number to the Arduino. The Arduino then checks against set optimality standards.

If the gas level is too high, the Arduino will initiate the alarm system The alarm has three main components (amongst others):

- Red LED – turns on or flashes to show danger.
- Buzzer – makes a loud sound so people can hear it right away.
- LCD screen – shows the gas level and tells if the state is “Optimal” or “Not Optimal”
- Pushbutton - The button is used to reset the alarm. The alarm will not stop until someone checks and presses the button, so people don’t ignore a real problem.

Smoke detectors like this can be used in homes, bedrooms, offices and even factories. Many real smoke alarms are made as small devices on the wall or ceiling. By building one with Arduino, we learn step by step how sensors andalarms work together to keep people safe.

Components & **Justification**



- **Arduino Uno R3** — This is the brain of the project. It reads the values from the sensor and then decides what to do. We used it because it is simple, easy to program, and works well with many sensors and displays.

- **MQ-2 Gas Sensor** — This sensor can detect smoke and gases like LPG,propane, and methane. When the gas level is high, it gives an analog signal to the Arduino. We used it because it is cheap, reliable, and perfect for fire and gas safety projects.

- **16x2 LCD (I2C)** — The LCD shows the gas level and tells the user about details of the system visually. Without it, people would not know what the system is reading. We used it because it is clear, easy to read, and only needs two pins with I2C.

- **Piezo Buzzer** — The buzzer makes a loud sound when smoke or gas is detected. Sound is important because people may not always look at the screen. We used it because it gives an instant warning that something is wrong.

- **Red LED** — The LED lights up when gas levels are high. It is a visual warning that supports the buzzer. We used it because sometimes in a noisy place the buzzer may not be heard, but the LED can still be seen.

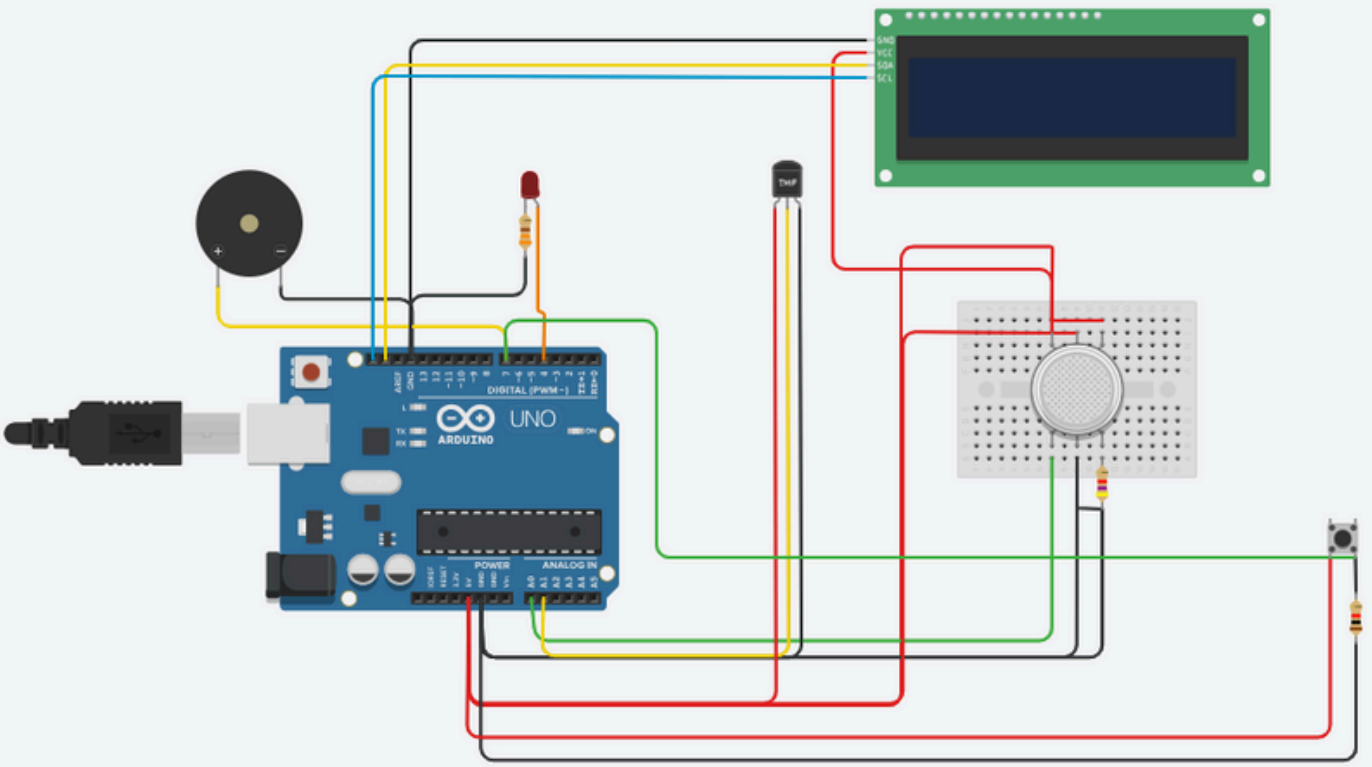
- **Pushbutton** (NO) — The button is used to reset the alarm after the danger is gone. We used it so that the alarm does not stop by itself. This way, the user must check and confirm safety before turning it off.

- **Resistors** — The resistors protect the LED and other parts from too much current. We used them because without resistors, the LED could burn out or the button could give wrong signals.

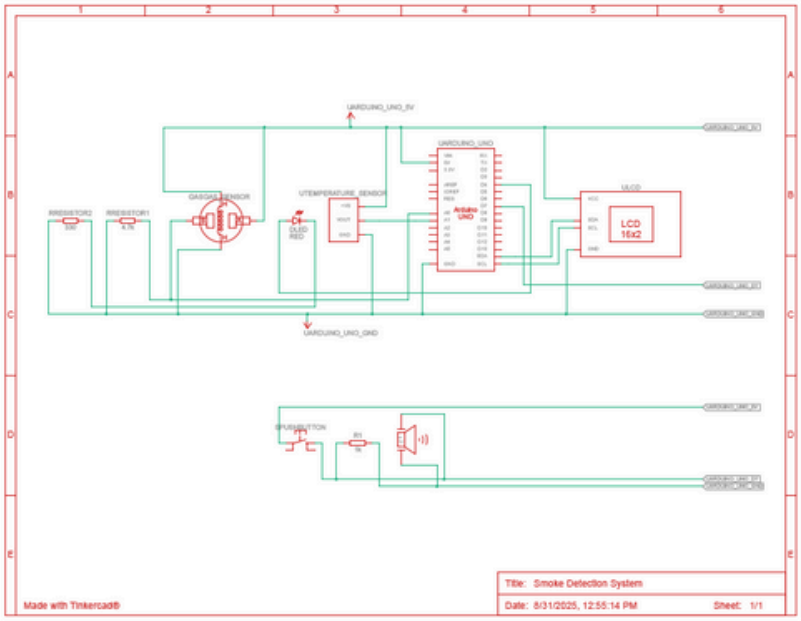
Observed **Wiring**

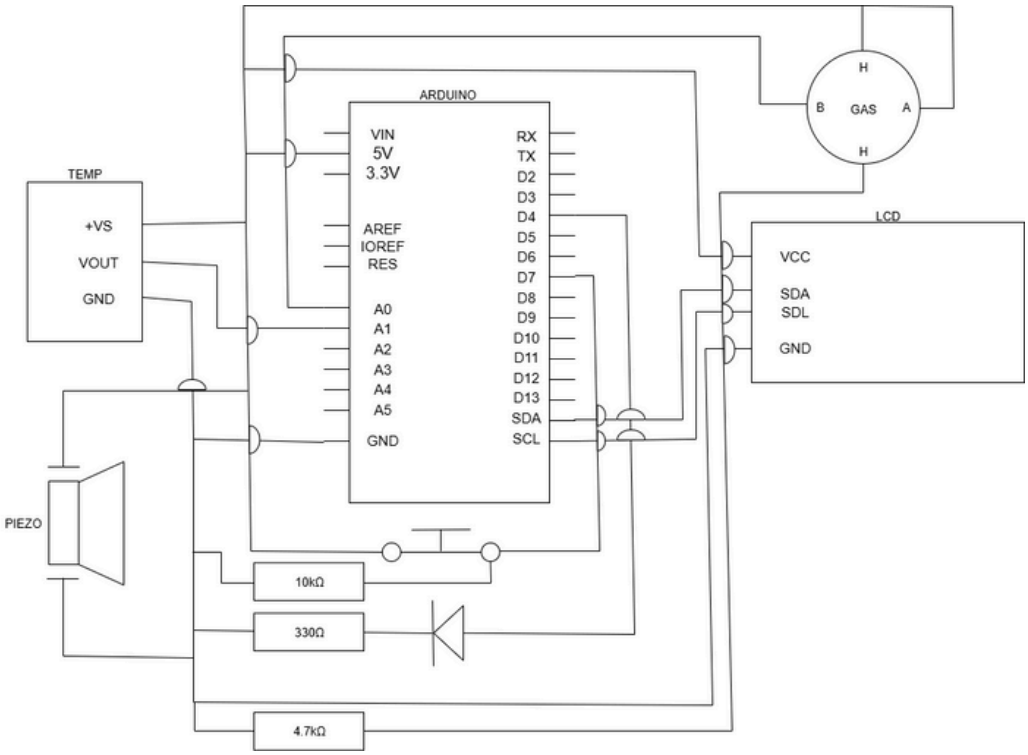
- MQ-2: VCC -> 5V, GND -> GND, AOUT -> A0
- LCD (I2C): VCC -> 5V, GND -> GND, SDA -> A4, SCL -> A5
- Buzzer: + -> D11, - -> GND
- LED: Anode -> D13 (220Ω), Cathode -> GND
- Pushbutton: One side -> D2, other -> GND (use INPUT_PULLUP in code)
- All GNDs connected to Arduino GND; VCCs to 5V

Smoke Detection **System** (TinkerCAD circuit)



Smoke Detection **System** (Circuit Schematic)





Smoke Detection **System** (C++ Code)

main.cpp

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

// LCD setup
LiquidCrystal_I2C lcd(0x27, 16, 2);

// Pin assignments
const int smokeSensorPin = A0;   // MQ-2 gas sensor
const int tempSensorPin  = A1;   // TMP36 temperature sensor
const int buzzerPin      = 7;    // Piezo buzzer
const int ledPin         = 4;    // Red LED
const int buttonPin      = 6;    // Pushbutton

// Variables
int smokeLevel = 0;
int tempC = 0;
bool alarmActive = false;

// Thresholds
const int tempThreshold  = 30;   // Temperature threshold
const int smokeThreshold = 400;  // Gas threshold

Windsurf: Refactor | Explain | Generate Function Comment | ×
void setup() {
  pinMode(smokeSensorPin, INPUT);
  pinMode(tempSensorPin, INPUT);
  pinMode(buzzerPin, OUTPUT);
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT_PULLUP); // pressed = LOW

  Serial.begin(9600);
```

main.cpp

```cpp
void setup() {
  // LCD init
  lcd.init();
  lcd.backlight();
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("System Initializing");
  delay(2000);
  lcd.clear();
}

Windsurf: Refactor | Explain | Generate Function Comment | ×
void loop() {
  // Read sensors
  smokeLevel = analogRead(smokeSensorPin);

  // TMP36 conversion
  tempC = map(((analogRead(tempSensorPin) - 20) * 3.04), 0, 1023, -20, 120);

  // Debug
  Serial.print("Smoke: ");
  Serial.print(smokeLevel);
  Serial.print(" | Temp: ");
  Serial.println(tempC);

  // Reset system if button pressed
  if (digitalRead(buttonPin) == LOW) {
    alarmActive = false;
    noTone(buzzerPin);
    digitalWrite(ledPin, LOW);
    lcd.clear();
```

Smoke Detection **System** (C++ Code)

main.cpp

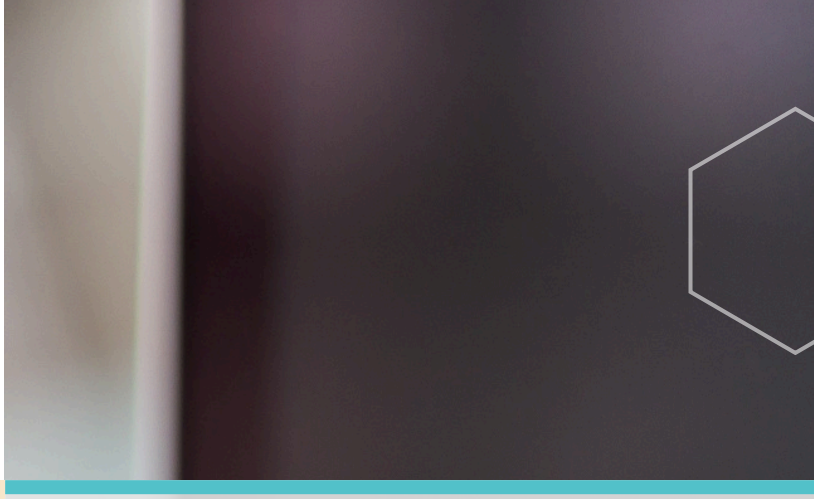
```
42  void loop() {
56    if (digitalRead(buttonPin) == LOW) {
61      lcd.setCursor(0, 0);
62      lcd.print("System Reset");
63      delay(1000);
64      lcd.clear();
65    }
66
67    // Temp
68
69    if (tempC > tempThreshold) {
70      digitalWrite(ledPin, HIGH); // LED ON if above 30
71      lcd.setCursor(0, 0);
72      lcd.print("Temp ALERT!     ");
73      lcd.setCursor(0, 1);
74      lcd.print("Temp: ");
75      lcd.print(tempC);
76      lcd.print("C   ");
77    } else {
78      digitalWrite(ledPin, LOW);  // LED OFF if ≤30
79      lcd.setCursor(0, 0);
80      lcd.print("Temperature OK ");
81      lcd.setCursor(0, 1);
82      lcd.print("Temp: ");
83      lcd.print(tempC);
84      lcd.print("C   ");
85    }
```

main.cpp

```
42  void loop() {
77    } else {
83      lcd.print(tempC);
84      lcd.print("C   ");
85    }
86
87    // Smoke
88
89    if (smokeLevel >= smokeThreshold) {
90      alarmActive = true;
91      // Tone frequency scales with smoke level
92      int freq = map(smokeLevel, smokeThreshold, 1023, 500, 4000);
93      tone(buzzerPin, freq);
94      lcd.setCursor(0, 0);
95      lcd.print("!! SMOKE ALERT !!");
96      lcd.setCursor(0, 1);
97      lcd.print("Level: ");
98      lcd.print(smokeLevel);
99      lcd.print("   ");
100   } else if (!alarmActive) {
101     noTone(buzzerPin); // Stop sound if safe and no active alarm
102   }
103
104   delay(5000); // refresh
105 }
106
```

Working Principle

- The MQ-2 sensor is always actively detecting the environment for gas presence. When there is smoke/gas, the sensor's gas concentration reading increases.
- The sensor sends this value to the Arduino Uno. The Arduino then compares it with a number we set in the code, called the threshold (deciding if it is safe or not).
- If the value is lower than the threshold, the Arduino says "everything is fine". The LCD displays "Temperature OK ", and the LED and buzzer stay off.
- But if the value is higher than the threshold, the Arduino "knows" there is danger. The LCD then shows "!! SMOKE ALERT !!" or "Temp ALERT!" The LED turns on, and the buzzer makes a loud sound.
- Once the air is clean again and the gas level goes below the threshold, the alarm will still stay on until the pushbutton is pressed. This is to make sure people don't ignore the danger. After pressing the button, the system resets and goes back to detecting for gas/smoke presence.



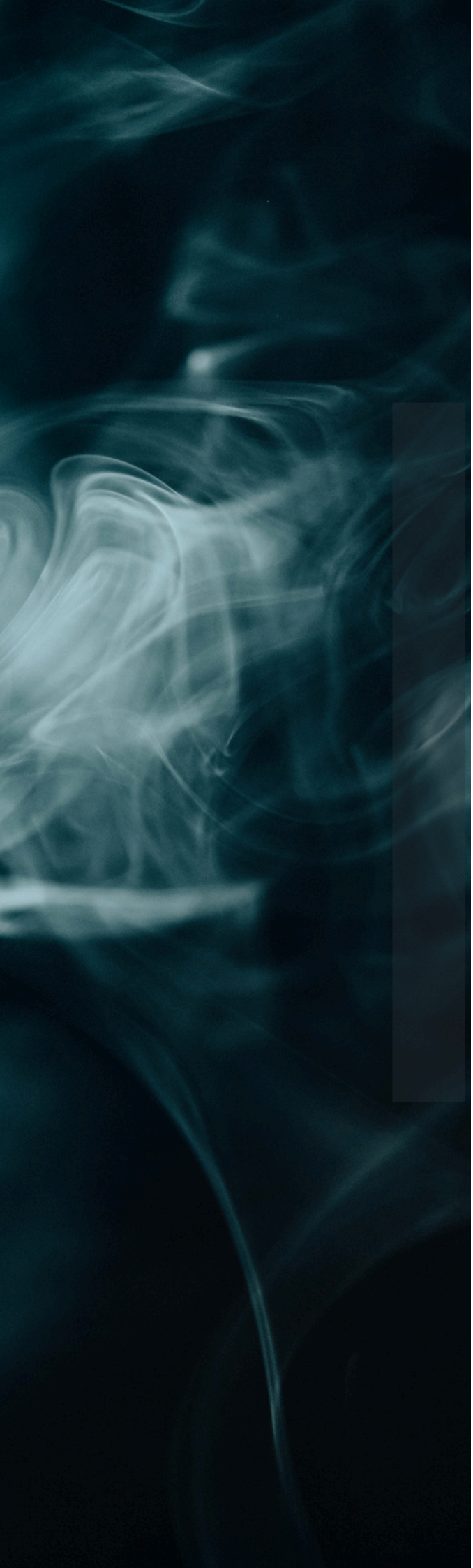
Testing and Calibration

Testing

- To test the system, we first power it on and check that the LCD shows the gas level. Then we put a small amount of smoke near the MQ-2 sensor (simulating a fire). The reading on the LCD should go up fast. When it goes higher than the threshold, the buzzerstarts to sound, the LED turns on, and the LCD shows “SMOKE ALERT!!!”.
- When the smoke goes away, the level on the LCD goes back down. After pressing the pushbutton, the buzzer stops, and the LED turns off. The LCD then shows “Optimal” again. This proves the system is working the way we want.

Calibration

- The MQ-2 sensor does not always give the same value in every place. A kitchen, factory, or classroom can all give different results. For this reason, we use calibration.
- In the Arduino code there is a threshold number (465 by default). If the alarm is too sensitive and goes off too often, we can make this number higher. If the alarm is not sensitive enough, we make the number lower. The alarm tone can also be set as required. It is best to test in the actual place where the detector will be used. Test smaller amounts of smoke and different temperatures and take note of the values. From this you can pick a safe number that works best.



Conclusion

The smoke detection system we built is a small but useful project. It shows how an Arduino and a simple sensor can be used to warn people about smoke or gas in the air. The MQ-2 sensor checks the air, the Arduino makes the decision, and the LED, buzzer and LCD give clear warnings. The pushbutton makes sure the user must confirm safety before stopping the alarm.

This project is low cost and easy to build, but still does an important job.It can be used in homes, offices, or small workshops to give an earlywarning and potentially save lives. The system can also be improved further by adding a phone alert or connecting it to the internet.

TinkerCAD Circuit **Link**

https://www.tinkercad.com/things/a4ITWrLXmSS-smoke-detection-system/editel?returnTo=https%3A%2F%2Fwww.tinkercad.com%2Fdashboard&sharecode=T2JdxRw8njGXv_aXSyKqNvdCFWO1vkU8nSLLlAlNTAM