Real-Time Environmental Monitoring and Air Quality Sensing

TEAM MEMBERS:

MAADHINI K

JOVITHA S

AIM:

To design and implement a system that monitors environmental parameters in real-time, such as air quality (gas concentration), and provides visual feedback on an OLED display while alerting users via a buzzer when gas levels exceed safe thresholds.

Components Required:

1. **Raspberry Pi Pico** – Microcontroller for data acquisition and control.
2. **Gas Sensor (MQ-2 / MQ-135)** – To detect harmful gases like LPG, smoke, and CO_2.
3. **OLED Display (SSD1306, 128x64 pixels)** – To display real-time gas concentration and status messages.
4. **Buzzer** – To sound an alarm when gas levels are unsafe.
5. **Jumper Wires and Breadboard** – For connections and circuit prototyping.
6. **Power Supply (3.3V or USB connection)** – To power the system.

Component Specifications:

raspberry pi pico

ARM Cortex M0+ processor, 264KB RAM, 2MB Flash, 3.3V operating voltage

Gas Sensor(MQ-2/MQ-135)

Detects LPG, smoke, CO_2

analog output, supply voltage 5V or 3.3V

OLED Display (SSD1306)

128x64 pixels, I2C interface, operating voltage 3.3V or 5V, monochrome display

Burzzer

DC buzzer, operating voltage 3.3V or 5V, triggered by GPIO pin

Pin configuration:

MQ135	raspberry pi pico
VCC	3.3V
GND	GND
AOUT	GP26

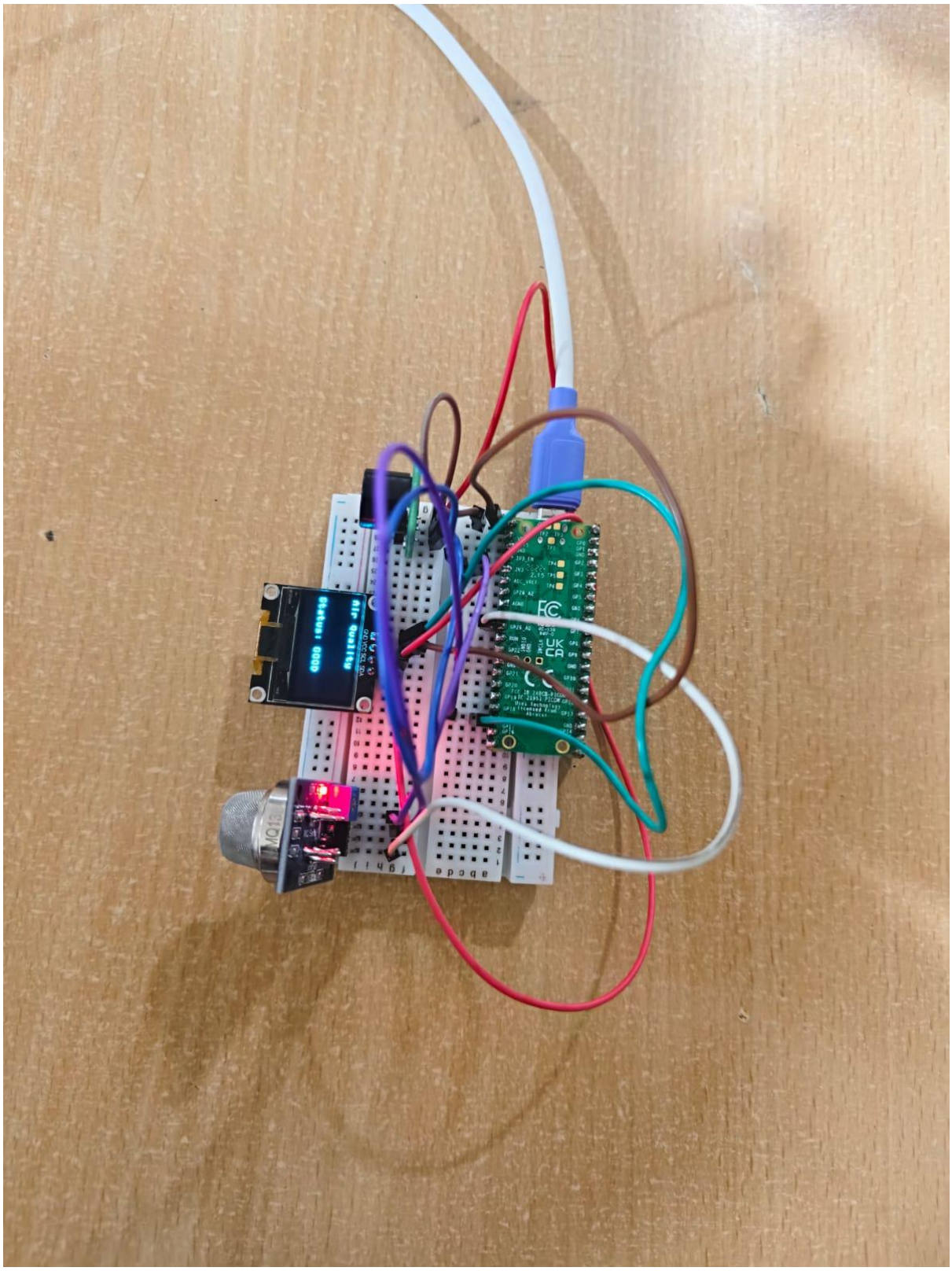
OLED	raspberry pi pico
VCC	3.3V
GND	GND
SDA	GP4
SCL	GP4

BUZZER	raspberry pi pico
+	GP15
-	GND

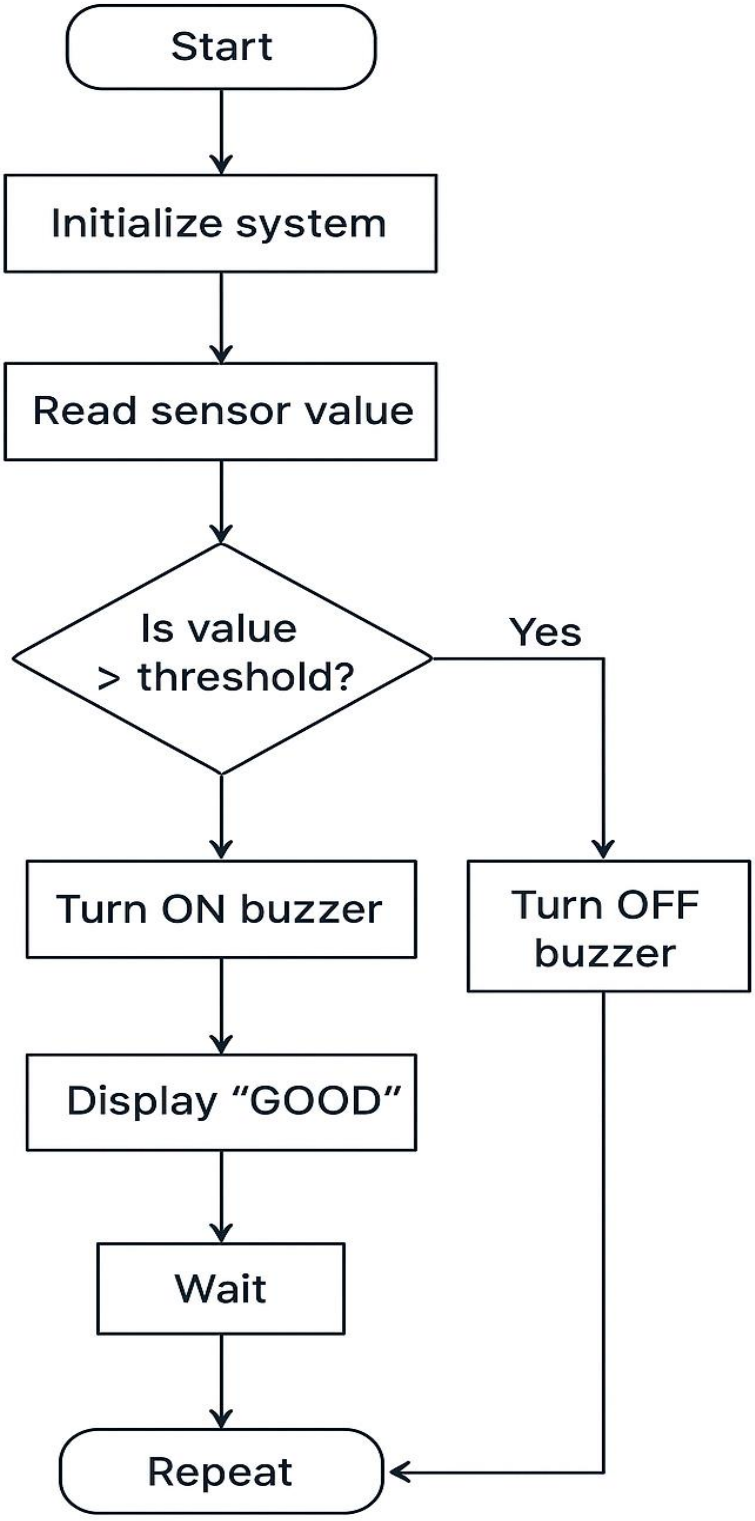
Procedure

1. **Connect the components** – gas sensor to ADC pin, buzzer to GPIO, OLED (SSD1306) via I2C on the Raspberry Pi Pico, and power the circuit using 3.3V.
2. **Initialize the system** – set up the ADC to read sensor data, configure I2C for the OLED, and set the buzzer pin as output.
3. **Read sensor data** – continuously read the gas sensor's analog signal and process it to determine the air quality level.
4. **Display on OLED** – show the gas concentration and status messages like "Normal" or "Danger" on the display.
5. **Activate buzzer** – turn the buzzer ON when the gas level exceeds the safe threshold, otherwise keep it OFF.
6. **Loop the monitoring** – keep reading, displaying, and alerting in real-time to ensure continuous air quality monitoring.

Circuit diagram:



Flowchart


Start
Initialize system
Read sensor value
Is value > threshold?
Yes
Turn ON buzzer
Turn OFF buzzer
Display “GOOD”
Wait
Repeat

:

Program:

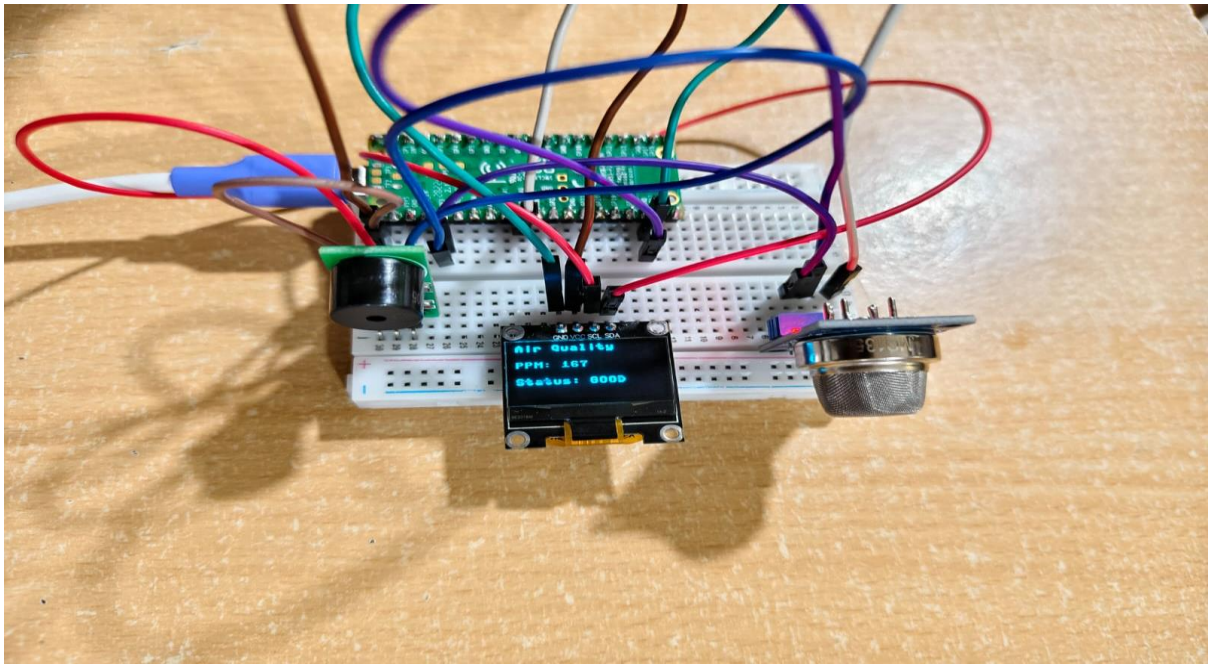
```
from machine import Pin, ADC, I2C
import ssd1306
import time
# MQ135 on ADC0 (GP26)
mq135 = ADC(Pin(26))
# Buzzer on GP15
buzzer = Pin(15, Pin.OUT)
# I2C for OLED (use your wiring!)
# If SDA=GP4, SCL=GP5 -> use this:
i2c = I2C(0, scl=Pin(5), sda=Pin(4), freq=400000)\
# If SDA=GP0, SCL=GP1 -> use this instead:
# i2c = I2C(0, scl=Pin(1), sda=Pin(0), freq=400000)
# Initialize OLED (0x3C = 60)
oled = ssd1306.SSD1306_I2C(128, 64, i2c, addr=0x3C)
# Threshold for air quality (adjust after calibration)
THRESHOLD = 100
def get_ppm(value):
    """
    Convert raw ADC (0–65535) to a simulated PPM scale (0–1000).
    NOTE: For real accuracy, you need MQ135 calibration curves.
    """
    return int((value / 65535) * 1000)
while True:
    raw_value = mq135.read_u16()
    ppm = get_ppm(raw_value)
    # Clear display
    oled.fill(0)
    oled.text("Air Quality", 0, 0)
```

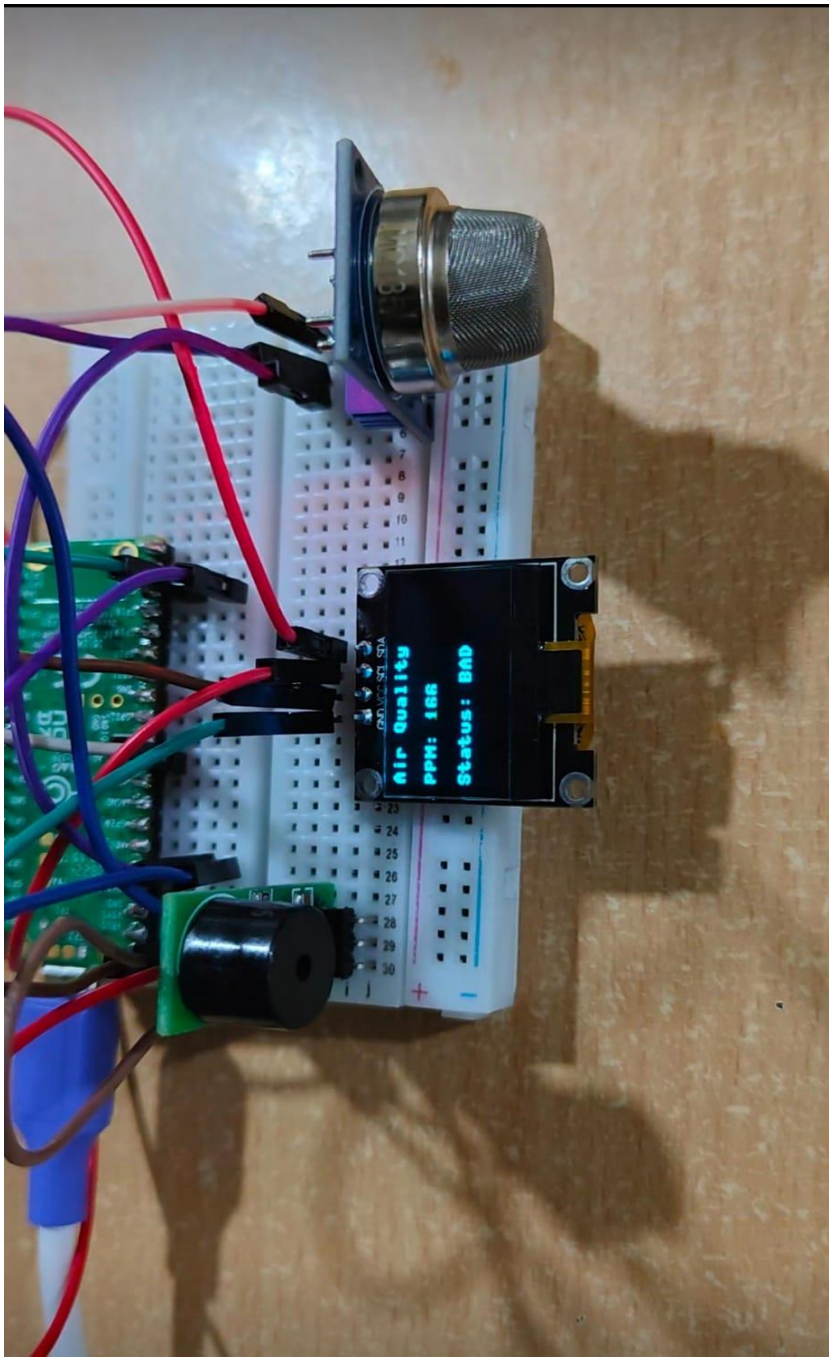
```
    oled.text("PPM: {}".format(ppm), 0, 20)


    if ppm > THRESHOLD:
        oled.text("Status: BAD", 0, 40)
        buzzer.value(1)  # Turn buzzer ON
    else:
        oled.text("Status: GOOD", 0, 40)
        buzzer.value(0)  # Turn buzzer OFF
    oled.show()
    print("Raw:", raw_value, " PPM:", ppm)
    time.sleep(1)
```

EXECUTION:

The execution of the Real-Time Environmental Monitoring and Air Quality Sensing system begins with connecting the MQ135 gas sensor, OLED display (SSD1306), and buzzer to the Raspberry Pi Pico. The gas sensor is connected to an ADC pin to read analog signals, the OLED is interfaced using I2C communication, and the buzzer is set to respond to GPIO signals. Once powered, the system initializes by configuring the ADC, setting up the OLED, and preparing the buzzer. The system then continuously reads the gas concentration from the sensor and converts the raw data into a PPM value. This information is displayed on the OLED screen, showing the current air quality status. Whenever the gas concentration crosses the defined threshold, the buzzer is activated to alert users about unsafe conditions, while a normal reading keeps the buzzer off and shows “GOOD” on the screen. The process loops every second, ensuring real-time monitoring and prompt alerts. The execution confirms that the system operates smoothly, providing clear visual feedback and audible warnings, making it an effective tool for ensuring air safety in homes, offices, or industrial areas.


GND VCC SCL SDA
Air Quality
PPM: 167
Status: GOOD



RESULT:

The Real-Time Environmental Monitoring and Air Quality Sensing system was successfully implemented using the Raspberry Pi Pico, gas sensor, OLED display, and buzzer. The system continuously monitored the air for harmful gases and displayed the current gas concentration on the OLED screen. When the gas level exceeded the predefined safe threshold, the buzzer was activated, alerting users to potential danger. The system operated reliably, providing real-time updates and warnings, which helps in ensuring safety and preventing hazardous situations. This setup can be effectively used in homes, offices, and industrial environments to monitor air quality and respond promptly to unsafe conditions.