1. A report of the states of gas detector and over temperature detector be sent to the PC when keys 2 and 3 are pressed respectively.

#include "mbed.h"

// Sensor Inputs

AnalogIn gasSensor(A0); // Gas sensor (0–100%)

AnalogIn tempSensor(A1); // LM35 temperature sensor (°C)

// Button Inputs

DigitalIn key2(D3, PullDown); // Key 2: gas check

DigitalIn key3(D2, PullDown); // Key 3: temperature check

// Serial Communication

UnbufferedSerial pc(USBTX, USBRX, 115200);

// Message buffer

char msgBuffer[128];

// Function to send data to PC

void sendToPC(const char\* msg) {

pc.write(msg, strlen(msg));

}

// Read gas level as percentage

int getGasLevelPercent() {

return (int)(gasSensor.read() \* 100.0f);

}

// Read temperature in °C

int getTemperatureCelsius() {

return (int)(tempSensor.read() \* 3.3f \* 100.0f); // 10mV/°C (LM35)

}

// MAIN

int main() {

sendToPC("System ready. Press Key2 for Gas or Key3 for Temperature.\r\n");

while (true) {

// Report gas level when Key2 is pressed

if (key2.read()) {

int gasLevel = getGasLevelPercent();

snprintf(msgBuffer, sizeof(msgBuffer), "[Key2] Gas Level: %d%%\r\n", gasLevel);

sendToPC(msgBuffer);

while (key2.read()); // Wait for release

ThisThread::sleep\_for(200ms); // Debounce

}

// Report temperature when Key3 is pressed

if (key3.read()) {

int tempC = getTemperatureCelsius();

snprintf(msgBuffer, sizeof(msgBuffer), "[Key3] Temperature: %d°C\r\n", tempC);

sendToPC(msgBuffer);

while (key3.read()); // Wait for release

ThisThread::sleep\_for(200ms); // Debounce

}

ThisThread::sleep\_for(100ms); // Polling delay

}

}

1. Send the current alarm state to the PC at regular intervals

#include "mbed.h"

// Sensors

AnalogIn gasSensor(A0);

AnalogIn tempSensor(A1);

// Serial

UnbufferedSerial pc(USBTX, USBRX, 115200);

// EventQueue and Ticker

EventQueue queue;

Thread queueThread;

Ticker alarmTicker;

// Buffer

char msgBuffer[128];

// Safe serial print

void sendToPC(const char\* msg) {

    pc.write(msg, strlen(msg));

}

// Get readings

int getGasLevelPercent() {

    return (int)(gasSensor.read() \* 100.0f);

}

int getTemperatureCelsius() {

    return (int)(tempSensor.read() \* 3.3f \* 100.0f);

}

// Determine alarm state

const char\* getAlarmState(int temp, int gas) {

    return (temp > 40 || gas > 60) ? "TRIGGERED" : "NORMAL";

}

// Alarm task (runs safely in thread context)

void sendAlarmStatus() {

    int temp = getTemperatureCelsius();

    int gas = getGasLevelPercent();

    const char\* state = getAlarmState(temp, gas);

    snprintf(msgBuffer, sizeof(msgBuffer),

             "[Alarm] Temperature: %d°C, Gas: %d%% → Alarm: %s\r\n",

             temp, gas, state);

    sendToPC(msgBuffer);

}

// MAIN

int main() {

    queueThread.start(callback(&queue, &EventQueue::dispatch\_forever));

    sendToPC("System running. Alarm status will update every 10 seconds.\r\n");

    // Use a lambda to call queue safely from ticker

    alarmTicker.attach([] {

        queue.call(sendAlarmStatus);

    }, 10s);

    while (true) {

        ThisThread::sleep\_for(500ms);  // Idle loop

    }

}

Output:

Alarm] Temperature: 26°C, Gas: 36% → Alarm: NORMAL

[Alarm] Temperature: 38°C, Gas: 100% → Alarm: TRIGGERED

[Alarm] Temperature: 34°C, Gas: 90% → Alarm: TRIGGERED

[Alarm] Temperature: 28°C, Gas: 62% → Alarm: TRIGGERED

[Alarm] Temperature: 33°C, Gas: 85% → Alarm: TRIGGERED

[Alarm] Temperature: 27°C, Gas: 57% → Alarm: NORMAL

[Alarm] Temperature: 34°C, Gas: 85% → Alarm: TRIGGERED

[Alarm] Temperature: 34°C, Gas: 75% → Alarm: TRIGGERED

[Alarm] Temperature: 31°C, Gas: 66% → Alarm: TRIGGERED

[Alarm] Temperature: 30°C, Gas: 61% → Alarm: TRIGGERED

[Alarm] Temperature: 28°C, Gas: 56% → Alarm: NORMAL

[Alarm] Temperature: 27°C, Gas: 53% → Alarm: NORMAL

1. Continuously send data to the PC regarding the alarm state, temperature readings, and gas detection status.

#include "mbed.h"

// Sensors

AnalogIn gasSensor(A0);             // Gas sensor input

AnalogIn tempSensor(A1);            // LM35 temperature sensor input

// Serial connection

UnbufferedSerial pc(USBTX, USBRX, 115200);

// Event queue and ticker

EventQueue queue;

Thread queueThread;

Ticker dataTicker;

// Buffer for messages

char msgBuffer[256];

// Send message to serial monitor

void sendToPC(const char\* msg) {

    pc.write(msg, strlen(msg));

}

// Get gas level as %

int getGasLevelPercent() {

    return (int)(gasSensor.read() \* 100.0f);

}

// Get temperature in °C

int getTemperatureCelsius() {

    return (int)(tempSensor.read() \* 3.3f \* 100.0f);  // 10mV = 1°C

}

// Determine alarm state

const char\* getAlarmState(int temp, int gas) {

    return (temp > 40 || gas > 60) ? "TRIGGERED" : "NORMAL";

}

// Task to send all readings

void sendAllData() {

    int temp = getTemperatureCelsius();

    int gas = getGasLevelPercent();

    const char\* alarm = getAlarmState(temp, gas);

    snprintf(msgBuffer, sizeof(msgBuffer),

        "\r\n=== Continuous System Data ===\r\n"

        "Temperature: %d°C\r\n"

        "Gas Level: %d%%\r\n"

        "Alarm State: %s\r\n",

        temp, gas, alarm

    );

    sendToPC(msgBuffer);

}

// MAIN

int main() {

    queueThread.start(callback(&queue, &EventQueue::dispatch\_forever));

    sendToPC("System started. Sending continuous monitoring data...\r\n");

    // Schedule data transmission every 5 seconds

    dataTicker.attach([] {

        queue.call(sendAllData);

    }, 5s);

    while (true) {

        ThisThread::sleep\_for(1s);  // Idle

    }

}

Output :

== Continuous System Data ===

Temperature: 25°C

Gas Level: 38%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 23°C

Gas Level: 38%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 23°C

Gas Level: 38%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 23°C

Gas Level: 37%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 23°C

Gas Level: 37%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 22°C

Gas Level: 37%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 22°C

Gas Level: 37%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 22°C

Gas Level: 37%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 34°C

Gas Level: 91%

Alarm State: TRIGGERED

=== Continuous System Data ===

Temperature: 32°C

Gas Level: 79%

Alarm State: TRIGGERED

=== Continuous System Data ===

Temperature: 29°C

Gas Level: 64%

Alarm State: TRIGGERED

=== Continuous System Data ===

Temperature: 29°C

Gas Level: 68%

Alarm State: TRIGGERED

=== Continuous System Data ===

Temperature: 26°C

Gas Level: 50%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 31°C

Gas Level: 77%

Alarm State: TRIGGERED

=== Continuous System Data ===

Temperature: 31°C

Gas Level: 77%

Alarm State: TRIGGERED

=== Continuous System Data ===

Temperature: 30°C

Gas Level: 71%

Alarm State: TRIGGERED

=== Continuous System Data ===

Temperature: 29°C

Gas Level: 66%

Alarm State: TRIGGERED

=== Continuous System Data ===

Temperature: 28°C

Gas Level: 61%

Alarm State: TRIGGERED

=== Continuous System Data ===

Temperature: 27°C

Gas Level: 57%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 26°C

Gas Level: 55%

Alarm State: NORMAL

=== Continuous System Data ===

Temperature: 26°C

Gas Level: 53%

Alarm State: NORMAL

1. Trigger a warning on the PC interface if temperature or gas levels exceed safe limits.

#include "mbed.h"

// Sensors

AnalogIn gasSensor(A0);             // Gas sensor

AnalogIn tempSensor(A1);            // LM35 temperature sensor

// Serial communication

UnbufferedSerial pc(USBTX, USBRX, 115200);

// Buffer

char msgBuffer[256];

// Send data to PC

void sendToPC(const char\* msg) {

    pc.write(msg, strlen(msg));

}

// Get gas level (%)

int getGasLevelPercent() {

    return (int)(gasSensor.read() \* 100.0f);

}

// Get temperature (°C)

int getTemperatureCelsius() {

    return (int)(tempSensor.read() \* 3.3f \* 100.0f);  // 10mV = 1°C

}

int main() {

    sendToPC("Continuous monitoring started...\r\n");

    while (true) {

        int temp = getTemperatureCelsius();

        int gas = getGasLevelPercent();

        // Print the current readings

        snprintf(msgBuffer, sizeof(msgBuffer),

                 "\r\n[Data] Temperature: %d°C | Gas Level: %d%%\r\n",

                 temp, gas);

        sendToPC(msgBuffer);

        // Trigger warning if unsafe

        if (temp > 40 || gas > 60) {

            sendToPC("!!! WARNING: Unsafe temperature or gas level detected !!!\r\n");

        }

        ThisThread::sleep\_for(1s);  // Delay between readings

    }

}

Output

[Data] Temperature: 24°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 35%

[Data] Temperature: 21°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 35%

[Data] Temperature: 21°C | Gas Level: 35%

[Data] Temperature: 22°C | Gas Level: 36%

[Data] Temperature: 22°C | Gas Level: 36%

[Data] Temperature: 35°C | Gas Level: 100%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 37°C | Gas Level: 100%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 37°C | Gas Level: 100%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 36°C | Gas Level: 100%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 37°C | Gas Level: 100%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 37°C | Gas Level: 100%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 36°C | Gas Level: 100%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 35°C | Gas Level: 100%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 33°C | Gas Level: 88%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 28°C | Gas Level: 62%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 26°C | Gas Level: 53%

[Data] Temperature: 29°C | Gas Level: 64%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 30°C | Gas Level: 74%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 31°C | Gas Level: 80%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 32°C | Gas Level: 85%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 32°C | Gas Level: 88%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 33°C | Gas Level: 90%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 34°C | Gas Level: 92%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 34°C | Gas Level: 93%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 33°C | Gas Level: 93%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 33°C | Gas Level: 93%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 34°C | Gas Level: 92%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 34°C | Gas Level: 92%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 34°C | Gas Level: 91%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 33°C | Gas Level: 91%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 33°C | Gas Level: 91%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 33°C | Gas Level: 90%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 33°C | Gas Level: 89%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 33°C | Gas Level: 88%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 32°C | Gas Level: 87%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 33°C | Gas Level: 86%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 32°C | Gas Level: 85%

!!! WARNING: Unsafe temperature or gas level detected !!!

[Data] Temperature: 32°C | Gas Level: 84%

!!! WARNING: Unsafe temperature or gas level detected !!!