#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

#include <DHT.h>

#include <TinyGPS++.h>

#include <Adafruit\_INA219.h>

// Pin Definitions

#define LM35\_PIN PA1    // Analog pin connected to LM35

#define DHTPIN PA0      // Digital pin connected to DHT22

#define DHTTYPE DHT22   // Define DHT22 sensor type

#define MQ2\_DIGITAL\_PIN PA6 // Digital pin connected to MQ-2 Smoke Sensor DO (Digital Output)

#define FLAME\_SENSOR\_PIN PA4 // Digital pin connected to Flame Sensor

#define BUZZER\_PIN PA5      // Digital pin connected to Buzzer

// Initialize objects

DHT dht(DHTPIN, DHTTYPE);

LiquidCrystal\_I2C lcd(0x27, 20, 4); // LCD with I2C address 0x27 (20x4)

Adafruit\_INA219 ina219;

#define SIM800 Serial1      // SIM800 on Serial1 (PA9 = TX, PA10 = RX)

//#define GPS\_SERIAL Serial2  // GPS on Serial2 (PA2 = TX, PA3 = RX)

//#define FLAME\_SENSOR\_PIN PA4

HardwareSerial GPS\_SERIAL(PA3, PA2); // RX, TX for GPS (connect GPS TX to PA3)

//LiquidCrystal\_I2C lcd(0x27, 20, 4); // LCD at I2C address 0x27

TinyGPSPlus gps;

float shuntVoltage = 0;

float busVoltage = 0;

float current\_mA = 0;

float LoadVoltage = 0;

void sendATCommand(String command);

void sendSMS(String message);

void setup() {

  SIM800.begin(9600);   // For SIM800

  GPS\_SERIAL.begin(9600); // For GPS

  Serial.begin(9600);   // Start Serial Communication

  dht.begin();          // Initialize DHT sensor

ina219.begin();

  lcd.init();           // Initialize LCD

  lcd.backlight();      // Turn on the backlight

  sendATCommand("AT");

  delay(2000);

  sendATCommand("AT+CMGF=1");  // SMS text mode

  delay(2000);

  sendATCommand("AT+CSCS=\"GSM\"");  // GSM charset

  delay(2000);

  Serial.println("Ready to detect");

  // Entry Display

  lcd.setCursor(4, 1);

  lcd.print("FDAS SYSTEM");

  delay(3000);          // Display for 3 seconds

  lcd.clear();

  pinMode(MQ2\_DIGITAL\_PIN, INPUT); // Set the MQ-2 digital output pin as input

  pinMode(FLAME\_SENSOR\_PIN, INPUT); // Set the flame sensor digital output pin as input

  pinMode(BUZZER\_PIN, OUTPUT); // Set the buzzer pin as output

  randomSeed(analogRead(0)); // Initialize randomness

  bool smokeAlertSent = false;

  bool flameAlertSent = false;

}

void loop() {

 // Continuously read GPS data

  while (GPS\_SERIAL.available()) {

      gps.encode(GPS\_SERIAL.read());

  }

  current\_mA = ina219.getCurrent\_mA();

  shuntVoltage = ina219.getShuntVoltage\_mV();     // Voltage across shunt resistor

  busVoltage = ina219.getBusVoltage\_V();          // Bus voltage

  loadVoltage = busVoltage + (shuntVoltage / 1000.0);  // Calculate total voltage

  // Read LM35 analog value

  int adcLM35 = analogRead(LM35\_PIN);

  // Convert ADC value to voltage for LM35

    float voltageLM35 = ((adcLM35 \* 3.3) / 4095.0) \* 2.0001;

  // Convert voltage to temperature (LM35: 10mV/°C)

  float lm35Temp = voltageLM35 \* 100.0;

  // Read DHT22 temperature

  float dhtTemp = dht.readTemperature();  // Celsius

  // Check if DHT22 readings are valid

  if (isnan(dhtTemp)) {

    Serial.println("Failed to read from DHT sensor!");

    lcd.clear();

    lcd.setCursor(0, 0);

    lcd.print("DHT Read Error!");

    delay(2000);

    return;

  }

  // Read MQ-2 digital output value (HIGH or LOW)

  int smokeDetected = digitalRead(MQ2\_DIGITAL\_PIN);

  // Read Flame Sensor digital output value (HIGH or LOW)

  int flameState = digitalRead(FLAME\_SENSOR\_PIN);

  // Debugging Output

  Serial.print("LM35 Temp: "); Serial.print(lm35Temp, 2);

  Serial.print(" °C | DHT22 Temp: "); Serial.print(dhtTemp, 2);

  Serial.print(" °C | Smoke Detected: ");

  if (smokeDetected == HIGH) {

    Serial.println("Yes");

  } else {

    Serial.println("No");

  }

  // Display on LCD

  lcd.clear();

  lcd.setCursor(0, 0);

  lcd.print("LM35 Temp: ");

  lcd.print(lm35Temp, 2);

  lcd.print((char)223);  // Degree symbol

  lcd.print("C");

  lcd.setCursor(0, 1);

  lcd.print("DHT22 Temp: ");

  lcd.print(dhtTemp, 2);

  lcd.print((char)223);  // Degree symbol

  lcd.print("C");

  // Check if smoke is detected

  lcd.setCursor(0, 2);

  if (smokeDetected == HIGH) {

    lcd.print("NO Smoke");

  } else {

    lcd.print("Smoke Detected!");

  }

  lcd.setCursor(0, 3);

  lcd.print("I:");

  lcd.print(current\_mA, 2);  // 2 decimal places to save space

  lcd.print("mA ");

  lcd.print("V:");

  lcd.print(LoadVoltage, 2); // 2 decimal places

  lcd.print("V");

  // Check for flame detection and trigger buzzer

  if (flameState == LOW) {  // Flame detected

  digitalWrite(BUZZER\_PIN, HIGH);

    delay(3000);

    digitalWrite(BUZZER\_PIN, LOW);

        String smsMessage = "FLAME DETECTED\n";

        if (gps.location.isValid() && gps.location.age() < 10000) {

            smsMessage += "Lat: ";

            smsMessage += String(gps.location.lat(), 6);

            smsMessage += "\nLng: ";

            smsMessage += String(gps.location.lng(), 6);

        } else {

            smsMessage += "GPS ERROR";

        }

        sendSMS(smsMessage);

        delay(10000);  // Wait 10 seconds to avoid flooding

    }

  delay(500);

delay(1000);

}

void sendATCommand(String command) {

    SIM800.println(command);

    delay(1000);

    Serial.println("Command: " + command);

    String response = "";

    while (SIM800.available()) {

        response += (char)SIM800.read();

    }

    Serial.println("Modem Response: " + response);

}

void sendSMS(String message) {

    Serial.println("Sending SMS...");

    SIM800.println("AT+CMGS=\"+9186609\*\*\*\*\*\""); // Replace with your number

    delay(2000);

    SIM800.print(message);

    delay(500);

    SIM800.write(26); // CTRL+Z

    delay(5000);

    Serial.println("SMS Sent: " + message);

}