Mini-Project
Air Quality Monitoring System

Name : A.Abishek Raja
RRN : 210191601004

Overview:

An Air Quality Monitoring System (AQMS) is a comprehensive network of sensors and instruments designed to measure air pollutants and meteorological conditions in real-time. It collects data from various monitoring stations, processes it to calculate air quality indices, and presents this information to the public through websites and apps. AQMS is essential for regulatory compliance, public health protection, and environmental management. It aids in research, emergency response, and raising awareness about air quality issues. Regular calibration and maintenance are crucial for ensuring data accuracy and reliability.

Introduction:

In today's world, where air quality is increasingly becoming a major concern due to its profound impact on both the environment and human health, our mini project presents an innovative solution: an Air Quality Monitoring System (AQMS). This system, equipped with a range of sensors and data processing capabilities, offers real-time monitoring of air pollutants and meteorological conditions, providing valuable insights for environmental management and public health protection. By focusing on the development and implementation of the AQMS, our project aims to enhance awareness of air quality issues and promote data-driven decision-making, contributing to the overall well-being and quality of life for communities and the environment.

Components:

The system comprises the following components,

Arduino Board:

The Arduino serves as the central controller, responsible for sensor data processing, IoT communication, and local display management.

Breadboard and Jumper Wires:

These fundamental components facilitate the electrical connections between sensors, controllers, and displays.

Power Source:

To ensure continuous system operation, a power source supplies the necessary voltage to the Arduino and sensors.

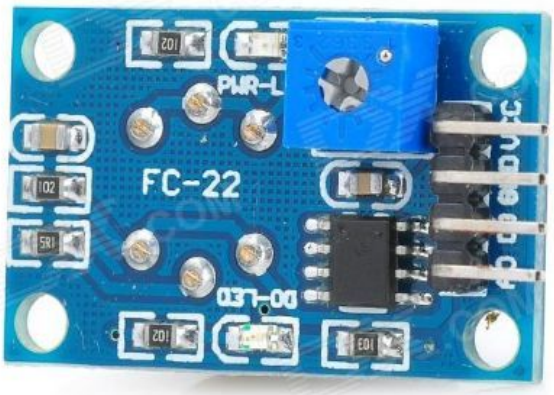
MQ-135 gas sensor:

The gas sensor module consists of a steel exoskeleton under which a sensing element is housed.This sensing element is subjected to current through connecting leads. This current is known as heating current through it, the gases coming close to the sensing element get ionized and absorbed by the sensing element. This changes the resistance of the sensing element which alters the value of the current going out of it.

Pin Configuration MQ-135 Gas Sensor:

From left to right first pins are as follows:

A0 - Analog output
D0 - Digital output
GND - Ground
Vcc - Supply (5V)



Specifications of MQ-135 gas sensor

- Wide detecting scope
- Fast response and High sensitivity
- Stable and long-life Simple drive circuit
- Used in air quality control equipment for buildings/offices, is suitable for detecting of NH3, NOx, alcohol, Benzene, smoke, CO2, etc.
- Size: 35mm x 22mm x 23mm (length x width x height)
- Working voltage: DC 5 V
- Signal output instruction.
- Dual signal output (analog output, and high/low digital output)
- 0 ~ 4.2V analog output voltage, the higher the concentration the higher the voltage.

HC 05 Bluetooth Module:

The HC-05 is a popular Bluetooth module commonly used for wireless communication with microcontrollers and other devices. It supports both serial communication (UART) and Bluetooth Classic profiles, making it suitable for a wide range of applications, including wireless data transfer and remote control.

Pin Configuration HC 05 Bluetooth Module:

The HC-05 Bluetooth module typically has six pins. Here's a standard pin configuration for the HC-05 module:

- VCC (Voltage Supply): Connect to a 5V power supply.
- GND (Ground): Connect to the ground (0V) of your power supply.
- TX (Transmit): Connect to the RX (Receive) pin of the microcontroller or device you're communicating with.
- RX (Receive): Connect to the TX (Transmit) pin of the microcontroller or device you're communicating with.
- STATE: This pin is an optional status indicator pin, which can be used to check the module's status (e.g., if it's in pairing mode). It's not always used in every application and may be left unconnected.
- EN (Enable/Key): This pin is sometimes used to enable or disable the module, typically via a microcontroller. If you connect this pin to GND, it usually puts the module into AT mode, allowing you to configure it. If you connect it to VCC, it's in data mode for normal communication.

Working:

Sensor Data Collection:

- The MQ-135 gas sensor is used to measure the concentration of various air pollutants, including carbon dioxide (CO2), ammonia (NH3), and volatile organic compounds (VOCs).

- The sensor is connected to the Arduino board, which serves as the microcontroller for data collection and processing.

Data Acquisition:

- The Arduino board collects data from the MQ-135 sensor by reading analog voltage values from the sensor's analog output pin.
- The sensor's analog output voltage varies in response to changes in gas concentrations.

Data Processing and Analysis:

- The Arduino runs a program (sketch) that processes the analog voltage readings and converts them into pollutant concentration values.
- The program may apply calibration factors to improve data accuracy.
- The processed data can be used to calculate air quality indices (AQI) or other relevant metrics for specific pollutants, if required.

Real-time Display (Arduino Serial Monitor):

- The Arduino is connected to a computer via USB, and the data is output to the computer's serial monitor.
- Real-time air quality data, such as pollutant concentrations, can be viewed through the Arduino IDE's serial monitor.

Bluetooth Data Transmission (HC-05 Module):

- The system incorporates an HC-05 Bluetooth module to enable wireless communication and data transmission to external devices.
- The HC-05 module is connected to the Arduino board, facilitating the transfer of air quality data to remote devices, such as smartphones or computers.

Bluetooth Module Operation:

- The HC-05 module is configured to establish a Bluetooth connection with a compatible device (e.g., a smartphone or computer) through a serial Bluetooth terminal application.
- The Arduino board communicates with the HC-05 module via its hardware or software serial interface, enabling data transmission.

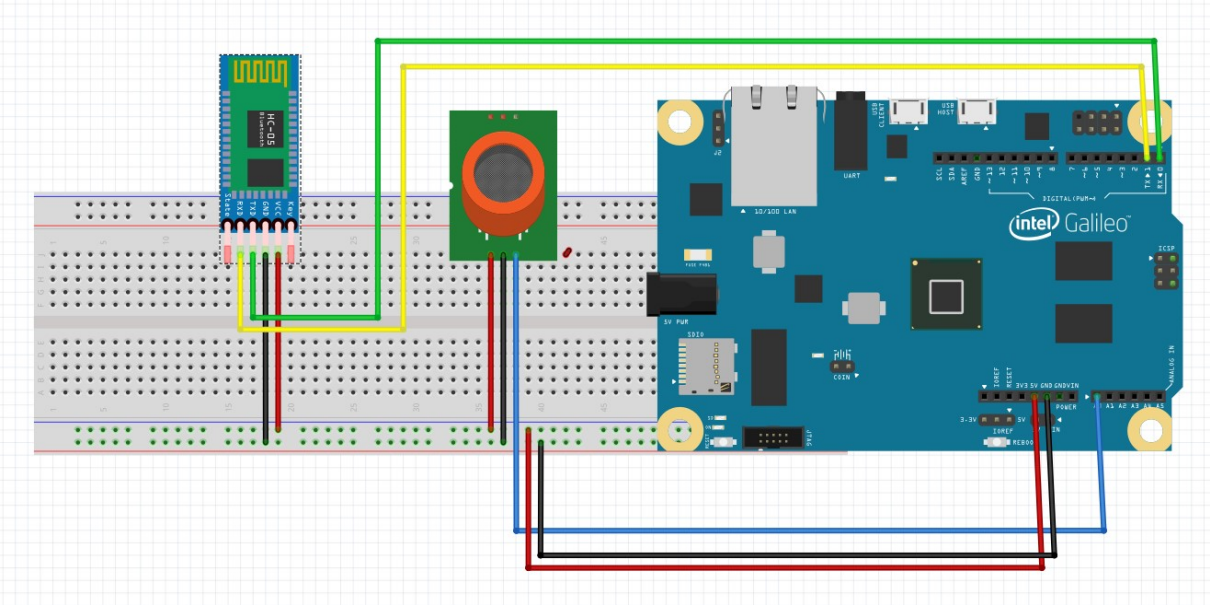
Real-time Data Transfer:

- As the Arduino processes the air quality data, it forwards the results to the HC-05 module.
- The HC-05 module wirelessly transmits the air quality data to the paired external device in real-time.
- Users can conveniently view and analyze the air quality information on the connected device's Bluetooth terminal application.

User Interaction:

Users have the flexibility to access air quality data on their preferred devices and perform further analysis or take appropriate actions based on the received information.

Circuit Diagram:



Hardware Connection:

Connect the MQ-135 gas sensor to the Arduino:

- Connect the VCC (or +) pin of the MQ-135 sensor to the 5V output on the Arduino.
- Connect the GND (or -) pin of the MQ-135 sensor to one of the GND pins on the Arduino.
- Connect the AOUT (analog output) pin of the MQ-135 sensor to analog pin A0 on the Arduino.

Connect the HC-05 Bluetooth module to the Arduino:

- Connect the TX (transmit) pin of the HC-05 to digital pin 2 (RX) on the Arduino.
- Connect the RX (receive) pin of the HC-05 to digital pin 3 (TX) on the Arduino.
- Connect the VCC (or +) pin of the HC-05 to the 5V output on the Arduino.
- Connect the GND (or -) pin of the HC-05 to one of the GND pins on the Arduino.

Here is a simplified schematic of the connections:

MQ-135 VCC ------- 5V (Arduino)
MQ-135 GND ------- GND (Arduino)
MQ-135 AOUT ------ A0 (Arduino)

HC-05 TX ---------> D2 (Arduino RX)

HC-05 RX ---------> D3 (Arduino TX)
HC-05 VCC ------- 5V (Arduino)
HC-05 GND ------- GND (Arduino)

Arduino Code :

```
#include <SoftwareSerial.h>

SoftwareSerial BTSerial(2, 3);  // RX, TX pins for HC-05
int sensorPin = A0;  // Analog input for the gas sensor

void setup() {
  Serial.begin(9600);
  BTSerial.begin(9600);  // Default HC-05 baud rate
  pinMode(sensorPin, INPUT);
}

void loop() {
  int sensorValue = analogRead(sensorPin);  // Read sensor value
  String airQuality;

  if (sensorValue >= 0 && sensorValue <= 100) {
    airQuality = "Excellent";
  }
else if (sensorValue > 100 && sensorValue <= 200) {
    airQuality = "Good";
  }
else if (sensorValue > 200 && sensorValue <= 300) {
    airQuality = "Moderate";
  }
else if (sensorValue > 300 && sensorValue <= 400) {
    airQuality = "Unhealthy for Sensitive Groups";
  }
else {
    airQuality = "Very Unhealthy";
  }

  // Send the air quality to Bluetooth terminal
  BTSerial.print("Air Quality: ");
  BTSerial.print(airQuality);
  BTSerial.print(" (Sensor Value: ");
  BTSerial.print(sensorValue);
  BTSerial.println(")");
```

```
  // Display the air quality on the serial monitor
  Serial.print("Air Quality: ");
  Serial.print(airQuality);
  Serial.print(" (Sensor Value: ");
  Serial.print(sensorValue);
  Serial.println(")");

  delay(1000);  // Adjust the delay as needed
}
```

Troubleshooting:

When issues arise, users can follow these troubleshooting steps:

Hardware Connections:

Double-check that all hardware components are correctly connected, and there are no loose wires.

Code Verification:

Verify that the Arduino code is uploaded correctly and that there are no syntax errors.

Sensor Functionality:

Ensure the sensors are positioned accurately, and they are functioning as expected.

Serial Output:

Monitor the serial output for any error messages or debug information to diagnose problems.

Customization:

Selection of Gas Sensors:

Depending on the pollutants you want to measure, you can choose different gas sensors. For example, if you are interested in CO2 levels, you can select a CO2 sensor. Each sensor may require different calibration procedures and data processing.

Calibration:

Accurate calibration of the sensors is crucial for obtaining reliable air quality data. Follow the calibration guidelines provided by the sensor manufacturer or calibrate the sensors based on your specific application.

Threshold Values:

Adjust the threshold values for air quality categorization based on your local air quality standards or project requirements. You can define custom AQI ranges that better match your criteria.

Data Logging:

Implement data logging to record air quality data over time. You can save the data to an SD card, cloud storage, or any other suitable method for later analysis.

Alerts and Notifications:

Integrate alerts and notifications when air quality levels exceed predefined thresholds. This can be done using sound alerts, email notifications, or mobile app notifications.

Power Supply:

Consider the power source for your project. If it needs to be portable or run on battery power, select an appropriate power supply solution.

Enclosure:

Design or select a suitable enclosure for your project to protect it from environmental factors and ensure it fits the intended application.

Data Analysis and Visualization:

If you plan to perform in-depth data analysis or create visualizations, you can use software tools and platforms such as Python, MATLAB, or data visualization libraries to gain insights from the collected data.

Integration with IoT:

If your project is part of an Internet of Things (IoT) system, you can integrate it with other IoT devices or platforms to create a comprehensive environmental monitoring solution.

Limitation:

Sensor Accuracy and Calibration:

Gas sensors may have limitations in accuracy, precision, and sensitivity. Calibration is essential but can be challenging, and sensors may require periodic recalibration.

Environmental Variability:

Air quality measurements can be affected by environmental factors like temperature, humidity, and weather conditions, which can introduce variability into the data.

Sensor Lifespan:

Gas sensors have a limited lifespan and may require replacement or maintenance after a certain number of operating hours or exposure to specific gases.

Interference:

Cross-gas interference can impact sensor accuracy when other gases are present in the environment, affecting the measurements.

Data Interpretation and Actionability:

Interpreting air quality data and converting it into actionable information can be challenging, requiring expertise in data analysis and understanding of air quality standards and regulations.

Future Consideration:

Advanced Sensors:

Keep an eye on advancements in sensor technology. New sensor technologies may offer improved accuracy, sensitivity, and the ability to detect additional pollutants.

IoT Integration:

Consider integrating the project into an Internet of Things (IoT) ecosystem. This can enable remote monitoring, data sharing, and real-time alerts, as well as integration with smart home or smart city systems.

Machine Learning and AI:

Explore the potential of machine learning and artificial intelligence for data analysis. These technologies can help predict air quality trends, identify pollution sources, and provide more accurate forecasts.

Miniaturization:

Advancements in sensor miniaturization and low-power consumption can make portable, battery-operated air quality monitors more practical.

Data Accessibility:

Ensure that the project provides accessible and user-friendly data interfaces for both experts and the general public. Open data platforms and visualization tools can enhance data dissemination.

Conclusion:

In summary, the air quality monitoring project plays a pivotal role in gauging and enhancing air quality. It acknowledges limitations related to sensors, calibration, and environmental variables. Future considerations encompass advanced sensors, IoT integration, AI utilization, miniaturization, and accessible data interfaces. By remaining adaptable and addressing evolving air quality concerns, the project is poised to contribute to a healthier environment and public well-being.

References:

https://blog.asksensors.com/air-quality-sensor-mq135-cloud-mqtt/

https://www.airnow.gov/aqi/aqibasics/#:~:text=AQI%20values%20at%20or%20below,is%20divided%20into%20six%20categories.