**EXPERIMENT NO: 8 PART B**

Objectives: Design IOT system based on ESP32 interfacing with MQ-2 (FC-22) to measure Carbon-dioxide in the air

Hardware & Software required

Arudio IDE software with ESP 32 software

MQ-2 Gas sensor (CO2 sensor)

ESP 32 Board

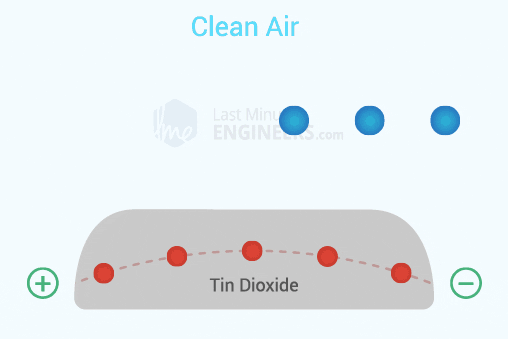
USB cable

Jumper wires

Theory: **How Does a Gas Sensor Work?**

When a SnO2 semiconductor layer is heated to a high temperature, oxygen is adsorbed on the surface. When the air is clean, electrons from the conduction band of tin dioxide are attracted to oxygen molecules. This creates an electron depletion layer just beneath the surface of the SnO2 particles, forming a potential barrier. As a result, the SnO2 film becomes highly resistive and prevents electric current flow.

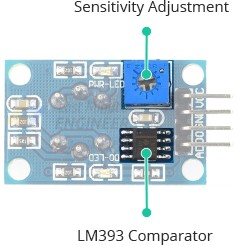
In the presence of reducing gasses, however, the surface density of adsorbed oxygen decreases as it reacts with the reducing gasses, lowering the potential barrier. As a result, electrons are released into the tin dioxide, allowing current to freely flow through the sensor.



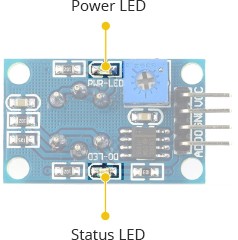
## MQ2 Gas Sensor Module Hardware Overview

The MQ2 gas sensor is simple to use and has two different outputs. It not only provides a binary indication of the presence of combustible gasses, but also an analog representation of their concentration in air.

The sensor’s analog output voltage (at the A0 pin) varies in proportion to the concentration of smoke/gas. The higher the concentration, the higher the output voltage; the lower the concentration, the lower the output voltage. The animation below shows the relationship between gas concentration and output voltage. This analog signal is digitized by an LM393 High Precision Comparator and made available at the Digital Output (D0) pin.



The module includes a potentiometer for adjusting the sensitivity of the digital output (D0). You can use it to set a threshold so that when the gas concentration exceeds the threshold value, the module outputs LOW otherwise HIGH. Rotating the knob clockwise increases sensitivity and counterclockwise decreases it.

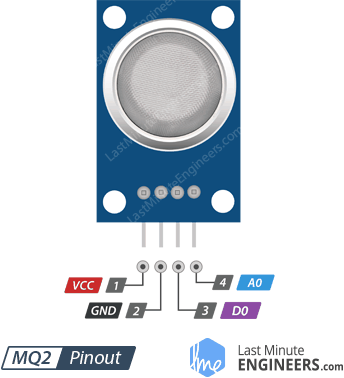


In addition, the module has two LEDs. The Power LED illuminates when the module is turned on, and the Status LED illuminates when the gas concentration exceeds the threshold value.

Here are the specifications:

|  |  |
| --- | --- |
| Operating voltage | 5V |
| Load resistance | 20 KΩ |
| Heater resistance | 33Ω ± 5% |
| Heating consumption | <800mw |
| Sensing Resistance | 10 KΩ – 60 KΩ |
| Concentration Range | 200 – 10000ppm |
| Preheat Time | Over 24 hour |

## MQ2 Gas Sensor Module Pinout



VCC supplies power to the module. Connect it to the 5V output of your Arduino.

GND is the ground pin.

D0 indicates the presence of combustible gasses. D0 becomes LOW when the gas concentration exceeds the threshold value (as set by the potentiometer), and HIGH otherwise.

A0 produces an analog output voltage proportional to gas concentration, so a higher concentration results in a higher voltage and a lower concentration results in a lower voltage.

## Calibrating the MQ2 Gas Sensor

Because the MQ2 is a heater-driven sensor, the calibration of the sensor may drift if it is left in storage for an extended period of time.

When first used after a long period of storage (a month or more), the sensor must be fully warmed up for 24-48 hours to ensure maximum accuracy.

If the sensor has recently been used, it will only take 5-10 minutes to fully warm up. During the warm-up period, the sensor typically reads high and gradually decreases until it stabilizes.

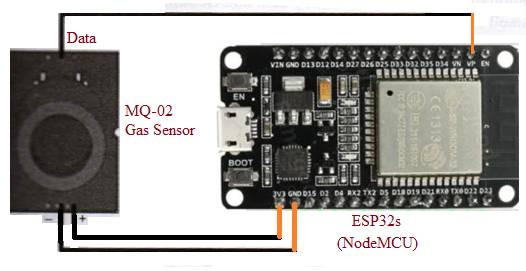
Procedure & Circuit diagram

In our first experiment, we will read the analog output to determine the concentration of the gas and see if it is within acceptable limits.

### Wiring

Let us connect the MQ2 gas sensor to the ESP32. Begin by connecting the VCC pin to the ESP32 5V pin and the GND pin to the ESP32 Ground pin. Finally, connect the module’s A0 output pin to Analog pin on the ESP32

The following image shows the wiring.



### Finding the threshold value. To determine whether the gas concentration is within acceptable limits, you need to record the values your sensor outputs when exposed to various amounts of smoke/gas.

Sketch

// starting of the sketch

#define MQ2Pin 36

float sensorValue; //variable to store sensor value

void setup()

{

Serial.begin(9600); // sets the serial port to 9600

Serial.println("MQ2 warming up!");

delay(20000); // allow the MQ2 to warm up

}

void loop()

{

sensorValue = analogRead(MQ2Pin); // read analog input pin 0

Serial.print("Sensor Value: ");

Serial.println(sensorValue);

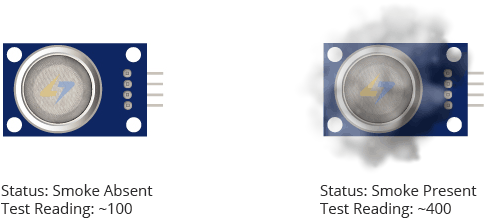
//delay(2000); //wait 2s for next reading

}

// end of the sketch

//When you run the sketch, you should see readings similar to the ones below:

//• In the absence of smoke/gas (around 100)

//• In the presence of smoke/gas (around 400)

This test may require some trial and error. Once you have the readings, you can use them as a threshold to trigger an action.

The sketch below determines whether the gas concentration is within acceptable limits.

/\* Change the threshold value with your own reading \*/

//start of the sketch for co2 detection sensor

#define Threshold 400

#define MQ2pin 36

float sensorValue; //variable to store sensor value

void setup()

{

Serial.begin(9600); // sets the serial port to 9600

Serial.println("MQ2 warming up!");

delay(20000); // allow the MQ2 to warm up

}

void loop()

{

sensorValue = analogRead(MQ2pin); // read analog input pin 0

Serial.print("Sensor Value: ");

Serial.print(sensorValue);

if(sensorValue> Threshold)

{

Serial.print(" | Smoke detected!");

}

Serial.println("");

delay(2000); // wait 2s for next reading

}

//end of the sketch co2 detection sensor If everything is fine, you should see something similar on the serial monitor.

Results and discussion

