

MQ-07 is a high-sensitivity sensor commonly used in electronics projects to detect **Carbon Monoxide (CO)**. The **MQ-07** is a metal oxide semiconductor (MOS) type sensor. It is specifically calibrated to detect **CO** concentrations in the air, typically ranging from 10 to 10,000 ppm.

MQ-07 requires a "high/low" heating cycle to function accurately. It usually switches between a high voltage (5V) to clean the sensor and a lower voltage.

When **carbon monoxide** is present, the conductivity of the sensor increases. The higher the gas concentration, the lower the electrical resistance.

MQ-07 has 4 pins:

- **VCC:** Connects to 5V.
- **GND:** Connects to Ground.
- **Digital Out (DO):** Provides a HIGH/LOW signal based on a threshold (set by the onboard potentiometer).
- **Analog Out (AO):** Provides a variable voltage (0-5V) representing the gas concentration. **I used this for calibration.**

NB:

Problem That I faced with my Arduino and its Solution:

If you face the problem where your device can't recognize the specific driver for the **FT232R USB UART** chip that is required for your specific "**Arduino Uno**" Then,

Please do this:

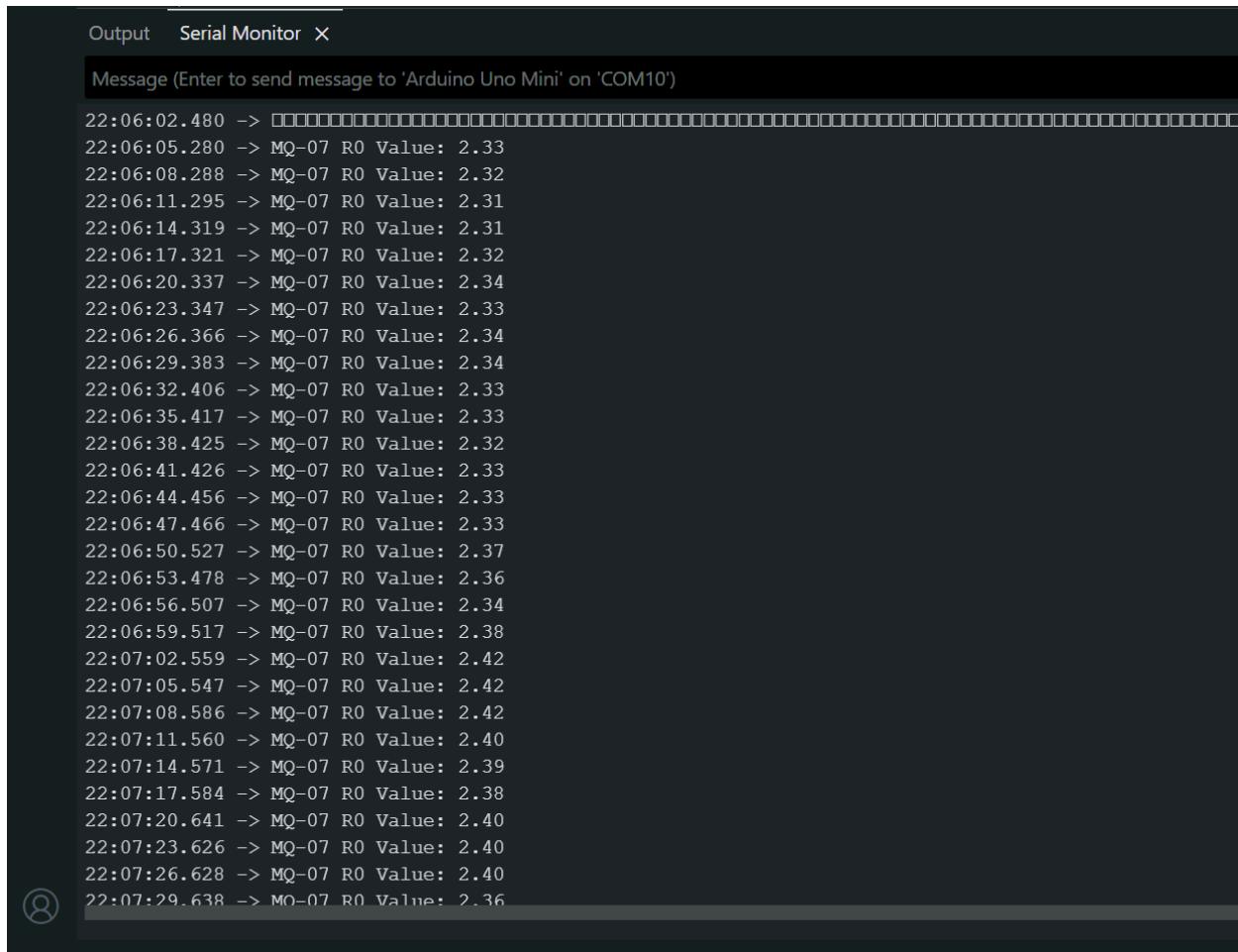
Manually install the **FTDI VCP (Virtual COM port)** drivers from this website :
<https://ftdichip.com/drivers/vcp-drivers/>

After installing it, now you will be able to see the **Port** connection in your Arduino IDE which was not seen before.

Working Steps:

The MQ-07 is designed to work on a **cycle** (High voltage for 60s, then Low voltage for 90s) to get the most accurate results.

- 1) At first I let the gas sensor's VCC port connected to the 5V pin of Arduino Uno for about 5-10 minutes so that the gas sensor gets heated up and show accurate results.
- 2) After that, I connected the A0 (Analogue Pin) of MQ-07 to Arduino's A0 pin and GND pin to Arduino's GND pin
- 3) After that I uploaded the code for the Calibration Phase. Here's the code link of the phase: [CODE](#)
- 4) Then I opened the Serial Monitor to see its readings. Here's what I saw:



The screenshot shows the Arduino Serial Monitor window. The title bar says "Output" and "Serial Monitor". The main area is a text box with a black background and white text. It displays a series of timestamped messages from an Arduino Uno Mini connected via COM10. Each message consists of a timestamp followed by "MQ-07 R0 Value: <value>". The values fluctuate between 2.33 and 2.42. The timestamp starts at 22:06:02.480 and continues sequentially. A small circular icon with a question mark is visible in the bottom-left corner of the monitor window.

```
22:06:02.480 -> MQ-07 R0 Value: 2.33
22:06:05.280 -> MQ-07 R0 Value: 2.32
22:06:08.288 -> MQ-07 R0 Value: 2.31
22:06:11.295 -> MQ-07 R0 Value: 2.31
22:06:14.319 -> MQ-07 R0 Value: 2.31
22:06:17.321 -> MQ-07 R0 Value: 2.32
22:06:20.337 -> MQ-07 R0 Value: 2.34
22:06:23.347 -> MQ-07 R0 Value: 2.33
22:06:26.366 -> MQ-07 R0 Value: 2.34
22:06:29.383 -> MQ-07 R0 Value: 2.34
22:06:32.406 -> MQ-07 R0 Value: 2.33
22:06:35.417 -> MQ-07 R0 Value: 2.33
22:06:38.425 -> MQ-07 R0 Value: 2.32
22:06:41.426 -> MQ-07 R0 Value: 2.33
22:06:44.456 -> MQ-07 R0 Value: 2.33
22:06:47.466 -> MQ-07 R0 Value: 2.33
22:06:50.527 -> MQ-07 R0 Value: 2.37
22:06:53.478 -> MQ-07 R0 Value: 2.36
22:06:56.507 -> MQ-07 R0 Value: 2.34
22:06:59.517 -> MQ-07 R0 Value: 2.38
22:07:02.559 -> MQ-07 R0 Value: 2.42
22:07:05.547 -> MQ-07 R0 Value: 2.42
22:07:08.586 -> MQ-07 R0 Value: 2.42
22:07:11.560 -> MQ-07 R0 Value: 2.40
22:07:14.571 -> MQ-07 R0 Value: 2.39
22:07:17.584 -> MQ-07 R0 Value: 2.38
22:07:20.641 -> MQ-07 R0 Value: 2.40
22:07:23.626 -> MQ-07 R0 Value: 2.40
22:07:26.628 -> MQ-07 R0 Value: 2.40
22:07:29.638 -> MQ-07 R0 Value: 2.36
```

I set the baud to 9600. You can see MQ-07's R0 value printing every few seconds. At first, the number will change quickly. Then I performed a 15-minute stabilization period until the number stays almost the same.

Then I implemented a power-law regression formula ($PPM = 100 \times (Rs/R0)^{-1.53}$) to convert analog voltage into PPM values.

As a result, I found the value: 2.65 for the R0 value.

- 5) Then I opened another sketch file where I uploaded the "**Detection**" code.
- 6) After uploading the code I reopened the serial monitor to see the reading.

Problem Faced:

But unfortunately I found that my **Raw value (13-14)** is extremely low. For an Arduino, the raw analog range is 0 to 1023. A value of 14 means the sensor is sending almost 0 volts.

As the Raw value was so low, my **Ratio (27.20)** is very high, which is why I see that **PPM (0.64,0.65,0.67,.....)**.

Solution:

Later, I still could see that "Clean Air" Raw value stays at 14,13,12.... So I tried to recalculate the Rs using my current data:

$$V_{out} = 14 \times (5.0 / 1023.0) = \text{approximately } 0.068V$$

$$Rs = (5.0 - 0.068) / 0.068 = \text{approximately } 72.5 \text{ ohm}$$

- 7) Then I replaced the value of R0 with Rs so that there will be accurate readings.
- 8) Then I uploaded the code and reopened the Serial Monitor. This is what I saw:

As a result, the **ratio** is hovering around **1.0**, and the **PPM** is sitting at roughly **100**.