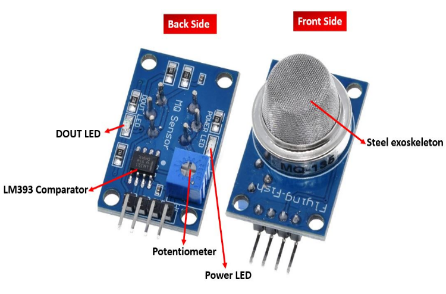
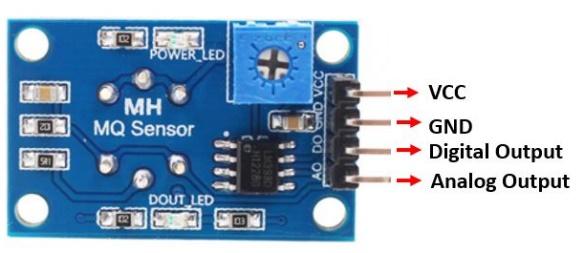
# 1-Interfacing MQ-135 Gas Sensor with Arduino:

MQ-135 sensor belongs to the MQ series that are used to detect different gasses present in the air. The MQ-135 sensor is used to detect gasses such as NH3,NOx, alcohol, Benzene, smoke,CO2 ,etc. steel exoskeleton houses a sensing device within the gas sensor module. MQ-135 gas sensor module features both analog output fetched from its AO pin and digital output fetched from its DO pin(not used ). The analog output voltage lies between 0-VCC where the output voltage increases relatively with the concentration of gas vapors coming in contact with the sensor. Under standard conditions, this output voltage from the sensor is directly proportional to the concentration of CO2 gas in PPM. This output voltage is converted to a digital value (0-1023) via the analog to digital converter in Arduino. This value is equal to the gas concentration in PPM.

The backside of the module consists of two LEDs: DOUT LED (lights when DO is LOW) and the power LED (lights when the module is powered on), an LM393 comparator, and a potentiometer.

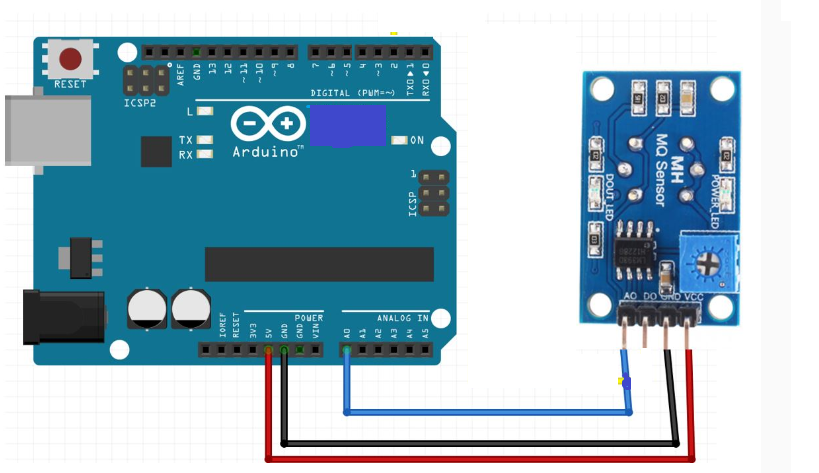
| **MQ-135 Module** | **Arduino** |
| --- | --- |
| VCC | 3.3V |
| GND | GND |
| AO | A0 |
| DO | Not used |

The MQ-135 sensor module consists of four pins namely VCC, GND, DO, and AO. The table below gives a brief description of them.

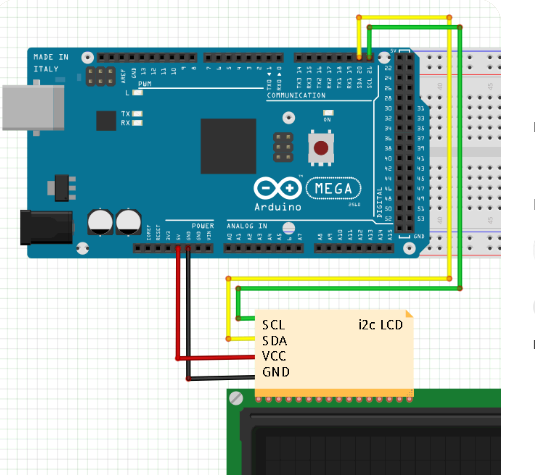
The table below shows the connections you need to make between the MQ3 sensor module and Arduino using both the analog output and the digital output pins of the sensor.

Connect MQ-135 sensor’s VCC pin with 5V terminal of Arduino UNO. This will power up the sensor.Additionally, we will connect the analog pin AO with A0 and DO will not be connected. Follow the connection diagram below, to connect your devices accordingly.



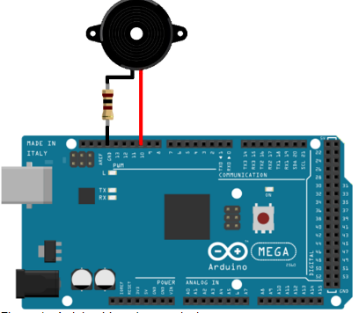
# 2- Interfacing I2C-LCD with Arduino:

On this I2C module you can control an LCD with ease using 2 wires connected to your Arduino board via input SDA and the SCL see the illustration below to find the correct pin where you can connect your I2C module. By default the module is configured with the address 0x27. To control the I2C module you need to include the LiquidCrystal\_i2C library.



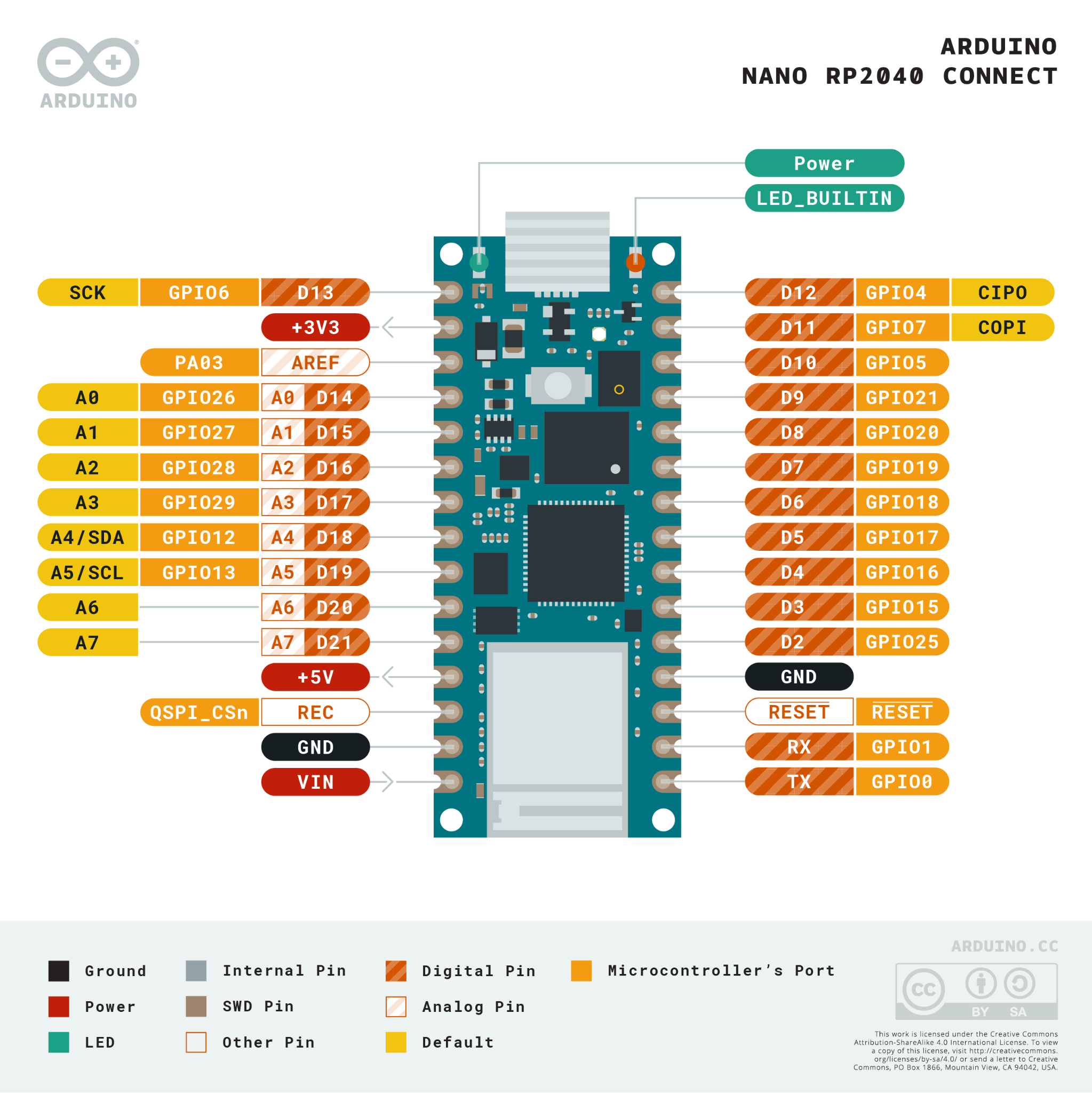
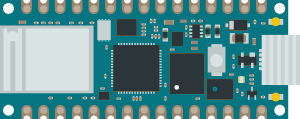
# 3- Interfacing buzzer with Arduino:

A buzzer is an audio signaling device which is mostly used as an alarming device or a user input feedback in devices. The long leg of the buzzer is its positive leg which is connected to the desired digital pin of the arduino whereas the short one is the negative which is connected to the ground

4- Interfacing with MQTT:

This was done with Arduino Nano RP2040 Connect.



To get started the following steps were essential to add the board to our IDE:

* open the board manager from **Tools > Board – Board Manager**.
* Search for rp2040 and install the **Arduino Mbed OS Nano Boards** package – accept any pop-ups.
* **Close** the board manager and return to the IDE.

Extra libraries were also installed from the Library Manager:

* Click **Tools -> Manage Libraries** to open the Library Manager.
* Search for **WifiNINA, PubSubClient,**  and install the latest version of the library.

The first section of the code involved programming the publisher. it consisted of the following steps:

1- declaring the password and the user name of the WIFI connected to:

char ssid[] = ".........."; // your network SSID (name)

char pass[] = "........." ;

2-Creating a partially initialized client instance.

WiFiClient wifiClient;

PubSubClient mqttClient(wifiClient);

3-Declaring the MQTT broker and port

const char broker[] = "test.mosquitto.org";

int port = 1883;

4-Attempting to connect to the WIFI

if (WiFi.status() == WL\_NO\_MODULE)

{Serial.println("Communication with WiFi module failed!");

while (true);

}

while (WiFi.begin(ssid, pass) == WL\_CONNECTED)

{

Serial.print("Connected to Network named: ");

Serial.println(ssid); // print the network name (SSID);

delay(10000);

}

5-Attempting to connect to the MQTT server

Serial.print("Attempting to connect to the MQTT broker: ");

Serial.println(broker);

mqttClient.setServer(broker, port);

while(1){

if (!mqttClient.connect("ard","chehab","123456789")) {

Serial.println("MQTT connection failed! Error code = ");

Serial.println(mqttClient.state());

}

else break;

}

Serial.println("You're connected to the MQTT broker!");

Serial.println();

6-publishing the message

Serial.println("publishing string ");

boolean rc = mqttClient.publish("test","test Message");

mqttClient.loop(); //call loop

APPENDIX: Complete Arduino Mega 2560 pins assignments

| **Device** | **Device Pins** | **Arduino Pins** |
| --- | --- | --- |
| **MQ-135 Module** | A0 | A0 |
| **I2C LCD** | SCL | SCL A5 |
| SDA | SDA A4 |
| **Buzzer** | Positive leg | D2 |
|  |  |  |

# References:

1. <https://microcontrollerslab.com/interfacing-mq-135-gas-sensor-arduino/>
2. <https://create.arduino.cc/projecthub/Arnov_Sharma_makes/lcd-i2c-tutorial-664e5a>
3. <https://www.14core.com/wiring-i2c-module-on-16x2-lcd-with-sclsda/>
4. <https://maker.pro/arduino/projects/buzzer>
5. <https://www.youtube.com/watch?v=bQ54De84Ww4>
6. <https://circuitdigest.com/microcontroller-projects/sending-arduino-data-to-webpage>
7. <https://docs.espressif.com/projects/esp-at/en/release-v2.2.0.0_esp8266/AT_Command_Set/Wi-Fi_AT_Commands.html>
8. <https://espressif-docs.readthedocs-hosted.com/projects/esp-at/en/release-v2.1.0.0_esp32s2/AT_Command_Examples/TCP-IP_AT_Examples.html>
9. <https://www.youtube.com/watch?v=c1st5cVRRzo>
10. <https://www.emqx.com/en/blog/esp8266-connects-to-the-public-mqtt-broker>
11. <https://www.emqx.com/en/blog/esp8266-connects-to-the-public-mqtt-broker#run-and-test>
12. <https://sonyarouje.com/2016/03/15/mqtt-communication-with-arduino-using-esp8266-esp-01/>
13. https://docs.arduino.cc/hardware/nano-rp2040-connect
14. [Get Started with Arduino Nano RP2040 Connect - OKdo](https://www.okdo.com/getting-started/get-started-with-arduino-nano-rp2040-connect/#h-4-web-server-example-toc)
15. //https://docs.arduino.cc/tutorials/uno-wifi-rev2/uno-wifi-r2-mqtt-device-to-device
16. https://pubsubclient.knolleary.net/api