**CO2 Sensor SKU:SEN0159**

From Robot Wiki

**Contents**

- 1 Introduction
- 2 Specification
- 3 Connecting Diagram
- 4 Tutorial
- 5 Sample code
  - 5.1 Sample code1
  - 5.2 Sample code2

**Introduction**

"Greenhouse Effect" is melting the iceberg every minute,. By knowing the exact concentration of CO2, we can do something to reduce the CO2 and to protect our earth. For that reason, a HQ CO2 sensor is designed by DFRobot engineer. This is the first CO2 sensor in OSHW market. The output voltage of the module falls as the concentration of the CO2 increases. The potentiometer onboard is designed to set the threshold of voltage. As long as the CO2 concentration is high enough (voltage is lower than threshold), a digital signal (ON/OFF) will be released.

- It has MG-811 sensor module onboard which is highly sensitive to CO2 and less sensitive to alcohol and CO, Low humidity&temperature dependency. All components have industrial quality which means stability and reproducibility.
- Onboard heating circuit brings the best temperature for sensor to function. 5V power input will be boosted to 6V for heating.
- This sensor has an onboard conditioning circuit for amplifying output signal.

**Notice:** 7~12V is necessary to power the microcontroller when using the CO2 Sensor

**Specification**

- Operating voltage: 5V
- Interface: Analog
- One digital output
- High quality connector
- Immersion gold surface
- Onboard heating circuit
- Size: 32x42mm
Connecting Diagram

![Connecting Diagram](image)

**Tutorial**

How to use this module?
It is very easy.
You need to set potentiometer onboard to the threshold value. Just make the red led turn off. With the CO2 concentration is enough high to make the sensor output voltage higher than threshold value, the led will be turned on. If you connect a buzzer to the module(right side), you will hear the alarm.

**Sample code**

**Sample code1**

```cpp
void setup(){
  Serial.begin(9600);
}

void loop(){
  Serial.print("Sample value:");
  Serial.println(analogRead(0));
  delay(100);
}
```
Sample code2

/***********************Demo for MG-811 Gas Sensor Module V1.1******************************/

Author: Tiequan Shao: tiequan.shao@sandboxelectronics.com
Peng Wei: peng.wei@sandboxelectronics.com

License: Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)

Note: This piece of source code is supposed to be used as a demonstration ONLY. More sophisticated calibration is required for industrial field application.

Sandbox Electronics 2012-05-31

*******************************************************************************/

/**************************Hardware Related Macros*******************************/

#define MG_PIN (0) //define which analog input channel you are going to use
#define BOOL_PIN (2)
#define DC_GAIN (8.5) //define the DC gain of amplifier

/**************************Software Related Macros*******************************

#define READ_SAMPLE_INTERVAL (50) //define how many samples you are going to take in normal operation
#define READ_SAMPLE_TIMES (5) //define the time interval(in milisecond) between each samples in normal operation

/**************************Application Related Macros******************************

//These two values differ from sensor to sensor. user should determine this value.
#define ZERO_POINT_VOLTAGE (0.324) //define the output of the sensor in volts when the concentration of CO2 is 400PPM
#define REACTION_VOLTGAE (0.020) //define the voltage drop of the sensor when move the sensor from air into 1000ppm CO2

/**************************Globals***********************************************


//two points are taken from the curve.
//with these two points, a line is formed which is "approximately equivalent" to the original curve.

(lg400, 0.324), point2: (lg4000, 0.280)

//slope = ( reaction voltage ) / (log400

-log1000)

void setup()
```c
44 {
45   Serial.begin(9600); //UART setup, baudrate = 9600bps
46   pinMode(BOOL_PIN, INPUT); //set pin to input
47   digitalWrite(BOOL_PIN, HIGH); //turn on pullup resistors
48
49   Serial.print("MG-811 Demostration\n");
50 }

52 void loop()
53 {
54   int percentage;
55   float volts;
56
57   volts = MGRead(MG_PIN);
58   Serial.print("SEN0159:");
59   Serial.print(volts);
60   Serial.print("V");
61
62   percentage = MGGetPercentage(volts,CO2Curve);
63   Serial.print("CO2:");
64   if (percentage == -1) {
65     Serial.print("<400");
66   } else {
67     Serial.print(percentage);
68   }
69   Serial.print("ppm");
70   Serial.print(" Time point:");
71   Serial.print(millis());
72   Serial.print("\n");
73
74   if (digitalRead(BOOL_PIN)){
75     Serial.print("=====BOOL is HIGH=====");
76   } else {
77     Serial.print("=====BOOL is LOW=====");
78   }
79
80   Serial.print("\n");
81   delay(200);
82 }
83
84 }/*****************************  MGRead *********************************************
85 Input:   mg_pin - analog channel
86 Output:  output of SEN-000007
87 Remarks: This function reads the output of SEN-000007
88*******************************************************************************/
89 float MGRead(int mg_pin)
```
int i;
float v=0;

for (i=0;i<READ_SAMPLE_TIMES;i++) {
    v += analogRead(mg_pin);
    delay(READ_SAMPLE_INTERVAL);
}

v = (v/READ_SAMPLE_TIMES) *5/1024 ;
return v;

/*****************************  MQGetPercentage **********************************
Input:   volts   - SEN-000007 output measured in volts
         pcurve  - pointer to the curve of the target gas
Output:  ppm of the target gas
Remarks: By using the slope and a point of the line. The x(logarithmic value of ppm)
of the line could be derived if y(MG-811 output) is provided. As it is a
logarithmic coordinate, power of 10 is used to convert the result to non-logarithmic
value.
return pow(10, ((volts/DC_GAIN)-pcurve[1])/pcurve[2]+pcurve[0]);

}