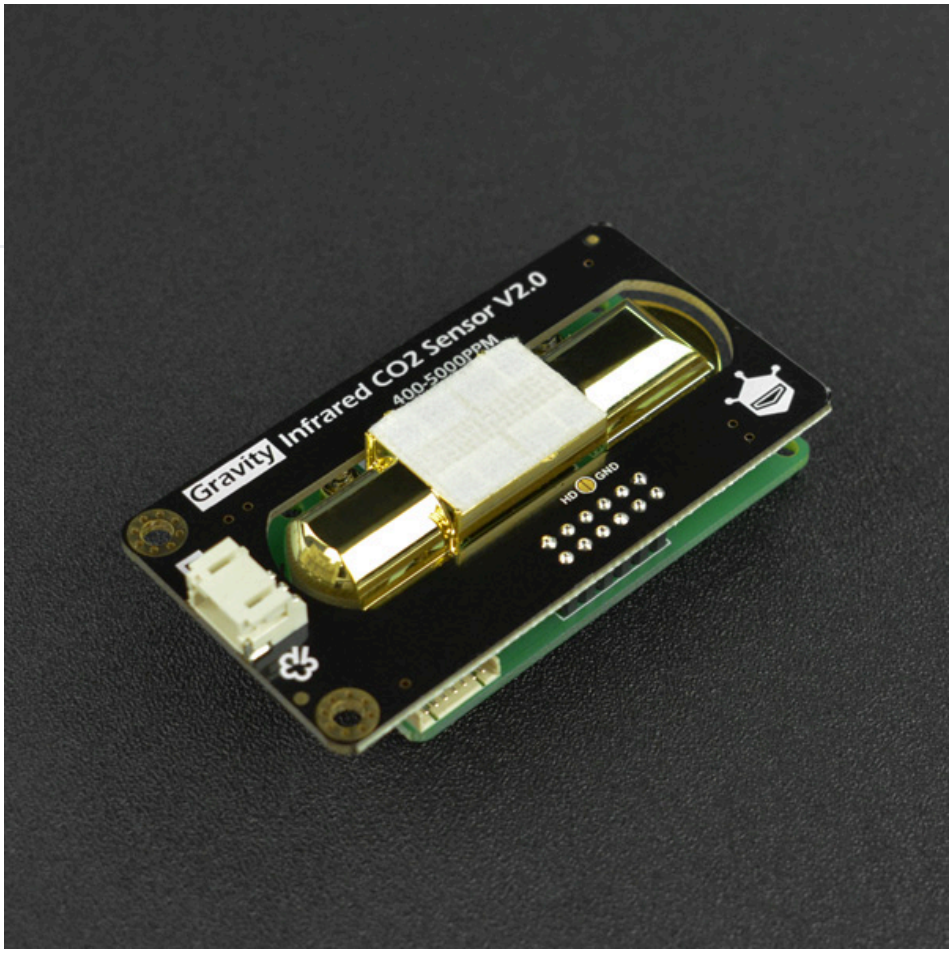
(https://www.dfrobot.com/product-1549.html)



Introduction

DFRobot released its latest high-precision infrared arduino CO2 sensor. The effectively measuring range is from 400 to 5000ppm. This sensor is based on non-dispersive infrared (NDIR) technology and has good selectivity and oxygen-free dependency. Besides, its service life could up to 5 years! It integrates temperature compensation and support DAC output. The current CO2 concentration can be read with only one digital port. Most importantly, the product is easy to use and is compatible with all types of microcontrollers and sensors.

In addition, this product is a high-performance sensor that combines the technology of mature infrared absorption gas detection with precision optical circuit design, as well as sophisticated circuit design. It has characteristics such as high sensitivity, high resolution, low power consumption, fast response, anti-water vapor interference, no poisoning, high stability and long life.

This sensor is directly compatible with the DFRobot Arduino IO expansion board thanks to its external DFRobot Gravity interface. This simplifies the use of the sensor as it is plug-and-play and no additional wiring required. This product could be widely used in HVAC, indoor air quality monitoring, industrial process and security

protection monitoring, agriculture and animal husbandry production process monitoring, etc.

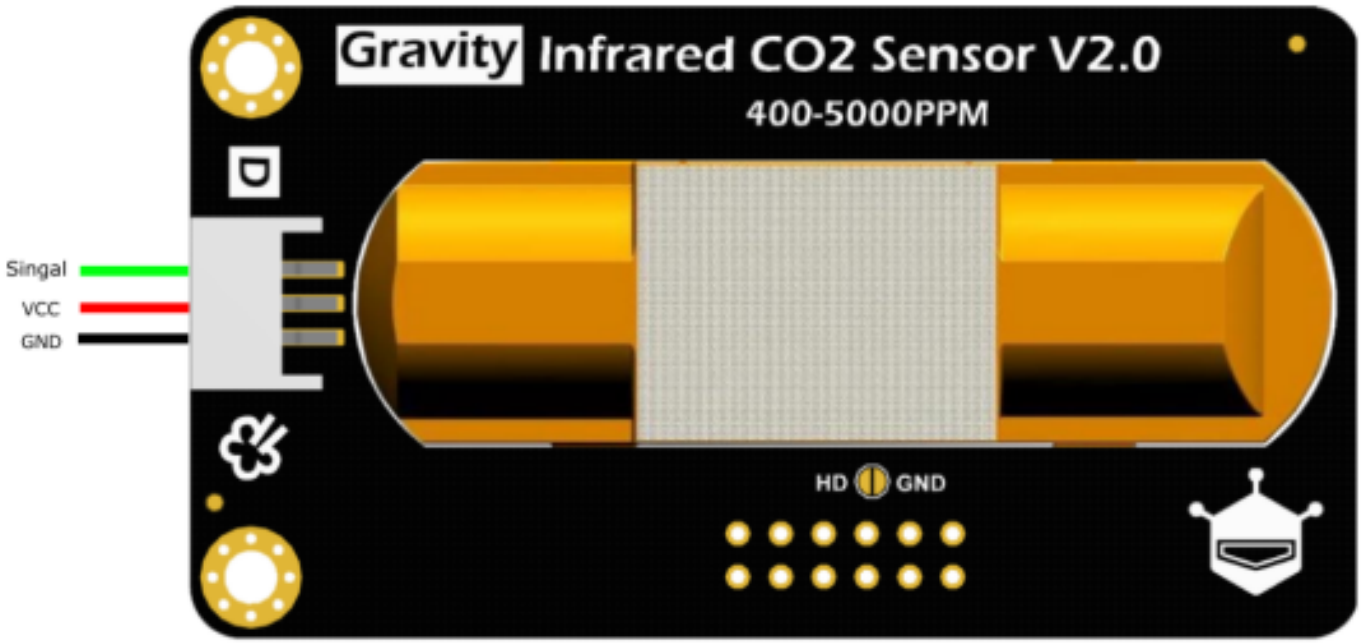
Feature

- Gas cell with gold plating, waterproof and anti-corrosion
- High sensitivity and low power consumption
- Excellent stability
- Temperature compensation and excellent linear output
- High cycle life
- Anti-water vapor interference and no poisoning
- No poisoning

Specification

- Gas Detection: Carbon Dioxide (CO2)
- Supply Voltage: DC (5.0±0.1)V
- Average Current: <40mA (@5V power supply)
- Peak Current: <125 (@5V power supply)
- Output Signal: PWM
- Preheating Time: 1min
- Response Time: T90 < 120s
- Operating Temperature: -10℃ - 50℃
- Operating Humidity: 0 - 95% RH (no condensation)
- Measuring Range: 400 - 5000 ppm
- Resolution: 1ppm
- Accuracy: ±(50ppm+5% reading)
- Dimension: 69mm×37mm/2.72×1.46"

Board Overview



Num	Label	Description
1	Signal	Signal Output (0.4-2V)
2	VCC	+
3	GND	-

Zero Calibration

- **Method 1: Manual zero calibration**
  Short circuit the HD and GND of the sensor to calibrate it. It always needs to last for over 7 seconds at a low level. Make sure that the sensor runs stably for over 20 minutes at a concentration of 400ppm before the calibration.
- **Method 2: Automatic zero calibration**
  The automatic calibration function means that the sensor will intelligently

determine the zero point according to the ambient CO2 concentration and automatically calibrate it after a period of continuous operation. The calibration starts from power-on and is performed once every 24 hours. The zero point for

automatic calibration is 400 ppm. This calibration is suitable for office and home enviornment.

Tutorial

This tutorial is designed for you to learn how to use the infrared CO2 sensor to measure the current CO2 concentration in the air in 5 minutes.

Compatibility

MCU	Work Well	Work Wrong	Untested	Remarks
Arduino Uno	√			
FireBeetle-ESP8266	√			
FireBeetle-ESP32	√			
FireBeetle-M0	√			
Micro:bit	√			

NOTE: ALL MCU NEED 5V POWER SUPPLY.

Sample Code

```
/*!
 * @file  CO2SensorPWMInterface.ino
 * @brief  This example The sensors detect CO2
 * @details  Infrared CO2 Sensor range : 400-4980ppm
 * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @license  The MIT License (MIT)
 * @author  [qsjhyy](yihuan.huang@dfrobot.com)
 * @version  V2.0
 * @date  2023-01-15
 */

#if defined(ESP32) || defined(ESP8266)
  // D7 pin is used as interrupt pin by default, other non-conflicting pins ca
  #define SENSOR_DATA_PIN   (D7)   // Sensor PWM interface
  #define INTERRUPT_NUMBER   digitalPinToInterrupt(SENSOR_DATA_PIN)   // inter
#elif defined(ARDUINO_SAM_ZERO)
  // Pin 5 is used as interrupt pin by default, other non-conflicting pins car
  #define SENSOR_DATA_PIN   (5)   // Sensor PWM interface
  #define INTERRUPT_NUMBER   digitalPinToInterrupt(SENSOR_DATA_PIN)   // inter
#else
  /* The Correspondence Table of AVR Series Arduino Interrupt Pins And Termina
   * ----------------------------------------------------------------------
   * |                                        |  DigitalPin  | 2  | 3  |
   * |    Uno, Nano, Mini, other 328-based    |--------------------------
   * |                                        | Interrupt No | 0  | 1  |
   * |----------------------------------------------------------------------
   * |                                        |    Pin       | 2  | 3  | 21 |
   * |              Mega2560                  |--------------------------
   * |                                        | Interrupt No | 0  | 1  | 2  |
   * |----------------------------------------------------------------------
   * |                                        |    Pin       | 3  | 2  | 0  |
   * |    Leonardo, other 32u4-based          |--------------------------
   * |                                        | Interrupt No | 0  | 1  | 2  |
   * |----------------------------------------------------------------------
```

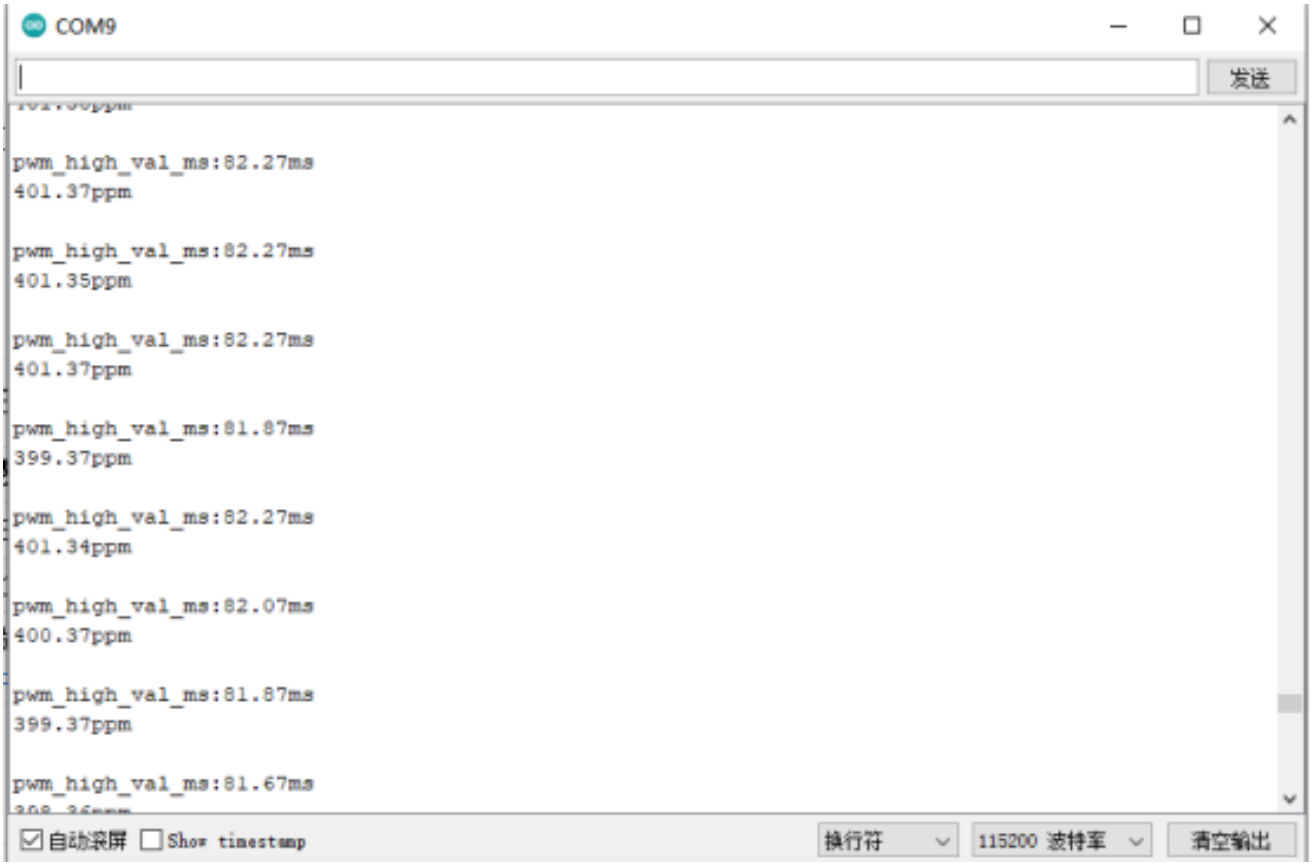
```
   * ---------------------------------------------------------------------
   *                     The Correspondence Table of micro:bit Interrupt Pins
   * ---------------------------------------------------------------------
   * |             micro:bit                      | DigitalPin |P0-P20 can be
   * |  (When using as an external interrupt,     |-------------------------
   * |no need to set it to input mode with pinMode)|Interrupt No|Interrupt num
   * |---------------------------------------------------------------------
   */
  // Open the external interrupt 0, connect INT1/2 to the digital pin of the m
  // UNO(2), Mega2560(2), Leonardo(3), microbit(P0).
  #define SENSOR_DATA_PIN   (2)   // Sensor PWM interface
  #define INTERRUPT_NUMBER   (0)   // interrupt number
#endif

// Used in interrupt, calculate pulse width variable
volatile unsigned long pwmHighStartTicks=0, pwmHighEndTicks=0;
volatile unsigned long pwmHighVal=0, pwmLowVal=0;
// interrupt flag
volatile uint8_t flag=0;

void interruptChange()
{
  if (digitalRead(SENSOR_DATA_PIN)) {
    pwmHighStartTicks = micros();    // store the current micros() value
    if(2 == flag){
      flag = 4;
      if(pwmHighStartTicks > pwmHighEndTicks) {
        pwmLowVal = pwmHighStartTicks - pwmHighEndTicks;
      }
    }else{
      flag = 1;
    }
  } else {
    pwmHighEndTicks = micros();    // store the current micros() value
    if(1 == flag){
      flag = 2;
      if(pwmHighEndTicks > pwmHighStartTicks){
        pwmHighVal = pwmHighEndTicks - pwmHighStartTicks;
      }
    }
  }
}
```

```
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  Serial.println("beginning...");
  pinMode(SENSOR_DATA_PIN, INPUT);
  attachInterrupt(INTERRUPT_NUMBER, interruptChange, CHANGE);
}

void loop() {
  if(flag == 4){
    flag = 1;
    float pwmHighVal_ms = (pwmHighVal * 1000.0) / (pwmLowVal + pwmHighVal);

    if (pwmHighVal_ms < 0.01){
      Serial.println("Fault");
    }
    else if (pwmHighVal_ms < 80.00){
      Serial.println("preheating");
    }
    else if (pwmHighVal_ms < 998.00){
      float concentration = (pwmHighVal_ms - 2) * 5;
      // Print pwmHighVal_ms
      Serial.print("pwmHighVal_ms:");
      Serial.print(pwmHighVal_ms);
      Serial.println("ms");
      //Print CO2 concentration
      Serial.print(concentration);
      Serial.println("ppm");
    }else{
      Serial.println("Beyond the maximum range : 398~4980ppm");
    }
    Serial.println();
  }
}
```

Result

Please wait at least 3 minutes (preheat process) until the data is stable. At this time,

the sensor can display the CO2 concentration.



FAQ

Q. Can I use the CO2 sensor with 3.3V microcontroller?

A. Yes, but you need to power the CO2 sensor with a 5V power supply separately, and change the conversion formula in the code **float voltage = sensorValue*(3300/1024.0);**

Q. Why is the reading of the CO2 sensor unstable?

A. The high sensitivity of the CO2 sensor may lead to relatively large fluctuations in the reading. The deviation of the reading can be reduced by taking the average of several values.

If you have any questions about using this product, please check the **FAQ list**

If you have any questions about using this product, please check the **FAQ list** (https://www.dfrobot.com/forum/topic/315493) for that product for a corresponding solution.

And for any questions, advice or cool ideas to share, please visit the **DFRobot Forum** (https://www.dfrobot.com/forum/).

More Documents

FAQ

For any questions, advice or cool ideas to share, please visit the **DFRobot Forum** (https://www.dfrobot.com/forum/).