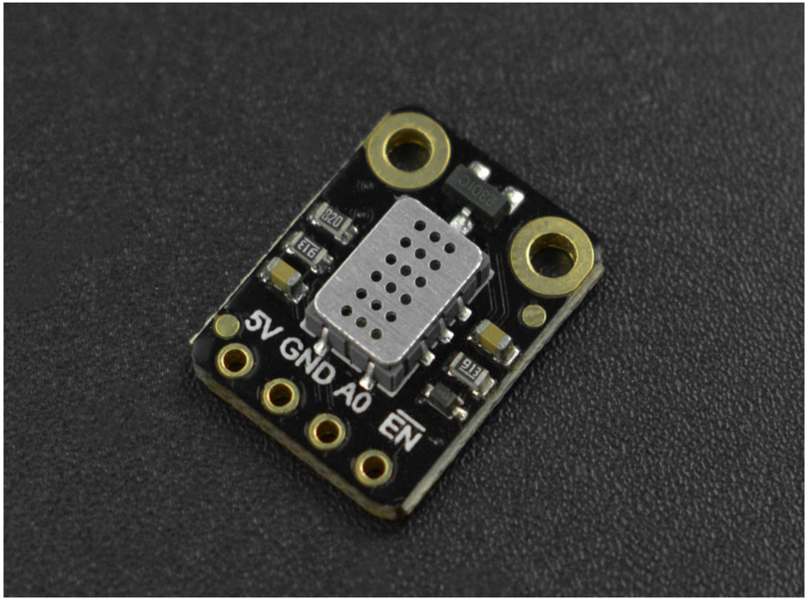
SKU:SEN0441 (https://www.dfrobot.com/product-2435.html)

(https://www.dfrobot.com/product-2435.html)



Introduction

This is a hydrogen and nitrogen oxide gas concentration sensor launched by DFRobot. The sensor uses MEMS technology to support gas concentration detection of H2, NO2 and NO. The matching sample code integrates the concentration conversion formula of various gas to facilitate the testing and use of sensors. The product supports 5V power supply, analog voltage output, and has power supply enable/disable pins for low power consumption.

Features

- Support detection of a variety of harmful gas
- Integrate the calculation formulas of various gas concentration
- Low power consumption
- I2C digital output
- Compatible with the 3.3~5.5V master controller

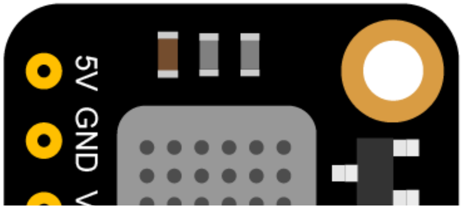
Specification

- Detection of Physical Quantities: Gas concentration or gas leakage of H_2, NO_2, NO
- Operating Voltage: 4.9 ~ 5.1V DC
- Power Dissipation: 0.45W
- Output Signal: analog quantity

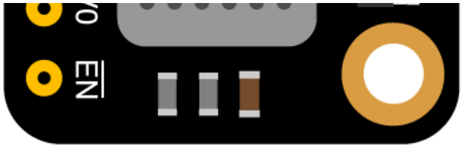
- Measuring Range:
    - 1 – 1000ppm(Hydrogen H_2)
    - 0.05 – 10ppm(Nitrogen dioxide NO_2)

- Working Temperature: -30 ~ 85℃
- Working Humidity: 5 ~ 95%RH (No condensation)
- Storage Temperature: -40~85℃
- Lifespan: >2 years （in the air)
- Circuit Board Size: 12mm×16mm
- Mounting Hole Size: internal diameter 2mm/external diameter 4mm
- Weight： 7g

Board Overview



5V
GND
MiCS-271
()



No.	Name	Function
1	5V	5V+
2	GND	-
3	A0	Analog Quantity Output
4	EN	Power Supply Enable/Disable Pins(enable at a low level, power off at a high level)

Tutorial for Arduino

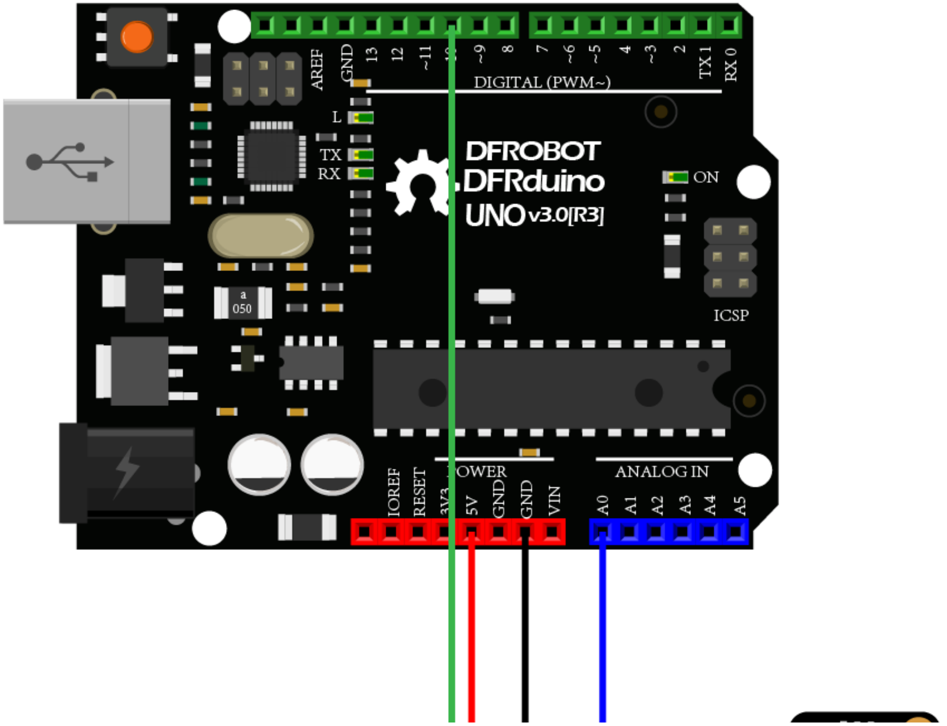
Download the program to the UNO, open Serial Port Monitor to view the gas concentration, raw data, and other parameters.

Requirements

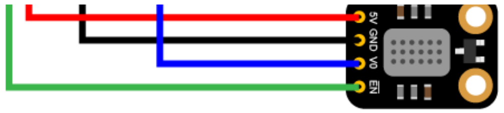
Requirements

- **Hardware**
    - DFRduino UNO R3 (https://www.dfrobot.com/product-838.html) (or similar) x 1
    - SEN0441 Fermion: MEMS Gas Sensor - MiCS-2714 () x1
    - M-M/F-M/F-F Jumper wires
- **Software**
    - Arduino IDE (https://www.arduino.cc/en/Main/Software)
    - Download and install the **DFRobot MICS Library** (https://github.com/DFRobot/DFRobot_MICS/archive/refs/heads/master.zip) (About how to install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))

Connection Diagram


AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX 1
RX 0
DIGITAL (PWM~)
L
TX
RX
DFROBOT
DFRduino
UNO v3.0[R3]
ON
ICSP
a
050
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5

()



Sample 1 - Read Gas Concentration(PPM) Data Calculated by the Sensor

Download the sample program to the Arduino UNO, and open Serial Port Monitor for viewing the NO2 gas concentration (PPM) data.

Connection Diagram

Steps

- Connect the module with Arduino according to the connection diagram above.
- Download and install the **DFRobot MICS Library** (https://github.com/DFRobot/DFRobot_MICS/archive/refs/heads/master.zip) (About how to install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))
- Open Arduino IDE
  - Upload the following sample code to the Arduino UNO
  - Or open the **getGasPPM.ino** code in the library file sample, revise the configuration in

the code (comment part of the I2C link code, uncomment the codes for breakout version), and change the gas type to NO2, then burn the modified code to the Arduino UNO

- Open the serial port monitor of the Arduino IDE, adjust the baud rate to 115200, and observe the print results.

```
/*!
  * @file  getGasPPM.ino
  * @brief Reading Gas concentration, A concentration of one part per million (PPM).
  * @n When using IIC device, select I2C address, set the dialing switch A0, A1 (Address_0
  * @n When using the Breakout version, connect the adcPin and PowerPin
  * @copyright   Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
  * @licence     The MIT License (MIT)
  * @author      ZhixinLiu(zhixin.liu@dfrobot.com)
  * @version     V1.1
  * @date        2021-04-19
  * @get         from https://www.dfrobot.com
  * @url         https://github.com/dfrobot/DFRobot_MICS
  */
#include "DFRobot_MICS.h"

#define CALIBRATION_TIME   3                      // Default calibration time is three mir

// When using the Breakout version, use the following program to construct an object from
/**!
```

```
/**!
  adcPin is A0~A5
  powerPin is General IO
*/
#define ADC_PIN   A0

#define POWER_PIN 10
DFRobot_MICS_ADC mics(/*adcPin*/ADC_PIN, /*powerPin*/POWER_PIN);

void setup()
{
  Serial.begin(115200);
  while(!Serial);
  while(!mics.begin()){
    Serial.println("NO Deivces !");
    delay(1000);
  } Serial.println("Device connected successfully !");

  /**!
    Gets the power mode of the sensor
    The sensor is in sleep mode when power is on,so it needs to wake up the sensor.
    The data obtained in sleep mode is wrong
   */
  uint8_t mode = mics.getPowerState();
  if(mode == SLEEP_MODE){
    mics.wakeUpMode();
    Serial.println("wake up sensor success!");
  }else{
    Serial.println("The sensor is wake up mode");
  }


  /**!
     Do not touch the sensor probe when preheating the sensor.
     Place the sensor in clean air.
     The default calibration time is 3 minutes.
  */
  while(!mics.warmUpTime(CALIBRATION_TIME)){
    Serial.println("Please wait until the warm-up time is over!");
    delay(1000);
  }
}

void loop()
{
  /**!
    Gas type:
    MICS-4514 You can get all gas concentration
    MICS-5524 You can get the concentration of CH4, C2H5OH, H2, NH3, CO
    MICS-2714 You can get the concentration of NO2
    Methane           (CH4)    (1000 - 25000)PPM
    Ethanol           (C2H5OH) (10   - 500)PPM
    Hydrogen          (H2)     (1    - 1000)PPM
    Ammonia           (NH3)    (1    - 500)PPM
    Carbon Monoxide   (CO)     (1    - 1000)PPM

    Nitrogen Dioxide (NO2)     (0.1  - 10)PPM
  */
  float gasdata = mics.getGasData(NO2);
  Serial.print(gasdata,1);
  Serial.println(" PPM");
  delay(1000);
  //mics.sleepMode();
}
```

Results

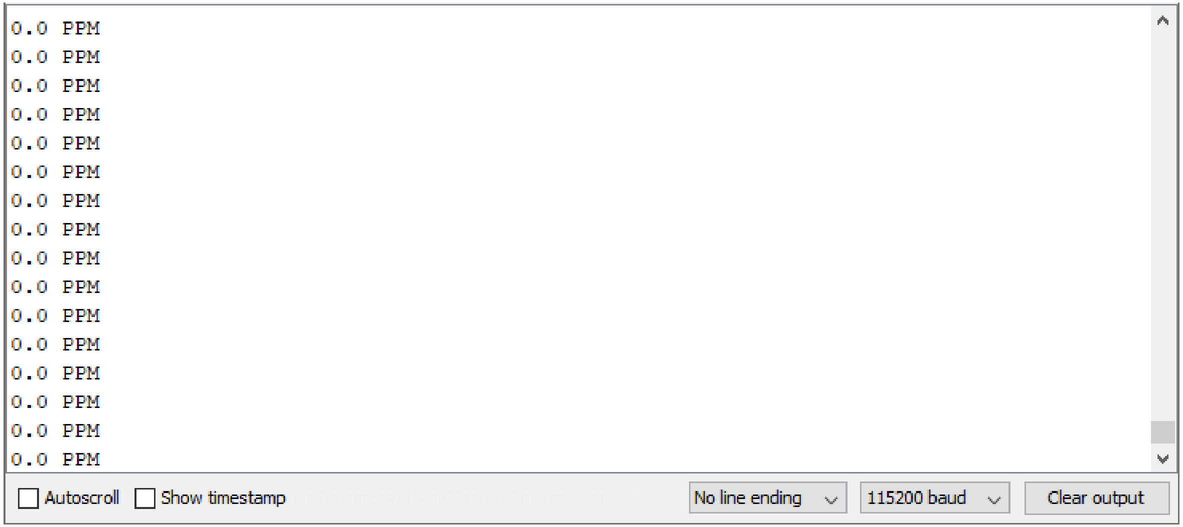
Open the serial port monitor and warm up for approximately 3 minutes to obtain the NO2 gas concentration data.

Note:

1. If you need to detect other gases, modify the detected gas settings in the sample code

yourself.

2. The sensor takes 3 minutes to warm up.



```
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
0.0 PPM
```

Autoscroll: unchecked, Show timestamp: unchecked, No line ending, 115200 baud

()

Sample Code 2 - Detection of Gas Leakage

The MEMS chip MiCS-2714 reacts to the NO2, H2 and NO gas, but the linearity of some gas concentration feedback value is poor. Therefore, the concentration value does not have a reference value, but can be used as the determination basis of the gas leakage. If you need a leak detection of the relevant gas, you can download the sample program to the Arduino UNO, open Serial Port monitor to see if the set gas is leaking.

Steps

- Connect the module with Arduino according to the connection diagram above.
- Download and install the **DFRobot MICS Library** (https://github.com/DFRobot/DFRobot_MICS/archive/refs/heads/master.zip) (About how to install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))
- Open Arduino IDE
  - Upload the following sample code to the Arduino UNO
  - Or open the **getGasExist.ino**code in the library file sample, revise the configuration in the code (comment part of the I2C link code, uncomment the codes for breakout version), and change the gas type to NO2, then burn the modified code to the Arduino UNO
- Open the serial port monitor of the Arduino IDE, adjust the baud rate to 115200, and observe the print results.

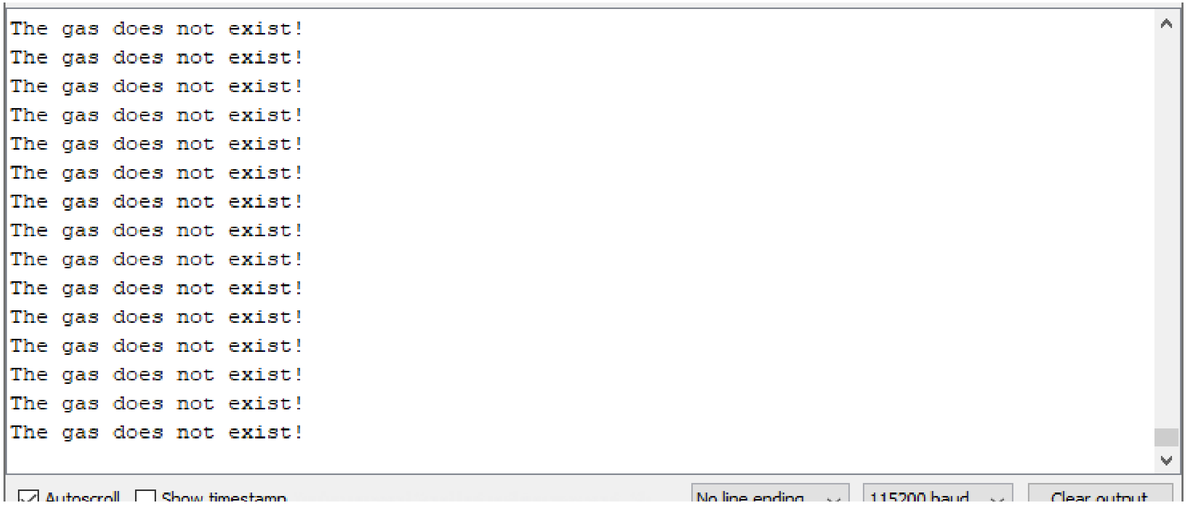
```
/*!
  * @file  getGasExist.ino
  * @brief Reading Gas concentration, A concentration of one part per million (PPM).
  * @n When using IIC device, select I2C address, set the dialing switch A0, A1 (Address_
  * @n When using the Breakout version, connect the adcPin and PowerPin
  * @copyright   Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
  * @licence     The MIT License (MIT)
  * @author      ZhixinLiu(zhixin.liu@dfrobot.com)
  * @version     V1.1
  * @date        2021-04-19
  * @get         from https://www.dfrobot.com
  * @url         https://github.com/dfrobot/DFRobot_MICS
  */

#include "DFRobot_MICS.h"

#define CALIBRATION_TIME   3                      // Default calibration time is three mi

// When using the Breakout version, use the following program to construct an object from
// When using the Breakout version, use the following program to construct an object from
/**!
  adcPin is A0~A5
  powerPin is General IO
*/

#define ADC_PIN   A0
#define POWER_PIN 10
DFRobot_MICS_ADC mics(/*adcPin*/ADC_PIN, /*powerPin*/POWER_PIN);

void setup()
{
  Serial.begin(115200);
  while(!Serial);
  while(!mics.begin()){
    Serial.println("NO Deivces !");
    delay(1000);
  } Serial.println("Device connected successfully !");

  /**!
    Gets the power mode of the sensor
    The sensor is in sleep mode when power is on,so it needs to wake up the sensor.
    The data obtained in sleep mode is wrong
   */
  uint8_t mode = mics.getPowerState();
  if(mode == SLEEP_MODE){
    mics.wakeUpMode();
    Serial.println("wake up sensor success!");
  }else{
    Serial.println("The sensor is wake up mode");
  }

  /**!
     Do not touch the sensor probe when preheating the sensor.
     Place the sensor in clean air.
     The default calibration time is 3 minutes.
  */
  while(!mics.warmUpTime(CALIBRATION_TIME)){
    Serial.println("Please wait until the warm-up time is over!");
    delay(1000);
  }
}

void loop()
{
  /**!
    Type of detection gas
    MICS-4514 You can get all gas state
    MICS-5524 You can get the state of CO, CH4, C2H5OH, C3H8, C4H10, H2, H2S, NH3
    MICS-2714 You can get the state of NO2, H2, NO
    MICS-2714 You can get the state of NO2, H2 ,NO
      CO       = 0x01  (Carbon Monoxide)
      CH4      = 0x02  (Methane)
      C2H5OH   = 0x03  (Ethanol)
      C3H8     = 0x04  (Propane)
      C4H10    = 0x05  (Iso Butane)
      H2       = 0x06  (Hydrogen)
      H2S      = 0x07  (Hydrothion)
      NH3      = 0x08  (Ammonia)
      NO       = 0x09  (Nitric Oxide)
      NO2      = 0x0A  (Nitrogen Dioxide)
  */
  int8_t gasFlag = mics.getGasExist(H2);
  if(gasFlag == EXIST){
    Serial.println("The gas exists!");
  }else{
    Serial.println("The gas does not exist!");
  }
  delay(1000);
  //mics.sleepMode();
}
```

Results

Open the serial port monitor and warm up for approximately 3 minutes to obtain the NO2 gas concentration data.

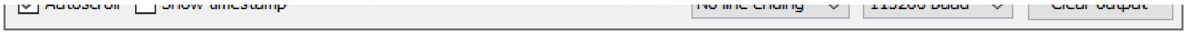
Note:

1. If you need to detect other gases, modify the detected gas settings in the sample code yourself.
2. The sensor takes 3 minutes to warm up.



```
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
The gas does not exist!
```

()



Sample Code 3 - Obtain the Sensor Analog Value

If you want to obtain the original sensor data, calculate the gas concentration or add your own temperature compensation functions, you can download the sample program to the Arduino UNO, open serial port monitor to view the original voltage output of the MEMS chip MiCS-2714.

Connection Diagram

Steps

- Connect the module with Arduino according to the connection diagram above.
- Download and install the **DFRobot MICS Library** (https://github.com/DFRobot/DFRobot_MICS/archive/refs/heads/master.zip) (About how to install the library? (https://www.arduino.cc/en/Guide/Libraries#.UxU8mdzF9H0))
- Open Arduino IDE
  - Upload the following sample code to the Arduino UNO
  - Or open the **getGasExist.ino** code in the library file sample, revise the configuration in the code (comment part of the I2C link code, uncomment the codes for breakout

version), and change the gas type to NO2, then burn the modified code to the Arduino UNO

- Open the serial port monitor of the Arduino IDE, adjust the baud rate to 115200, and observe the print results.

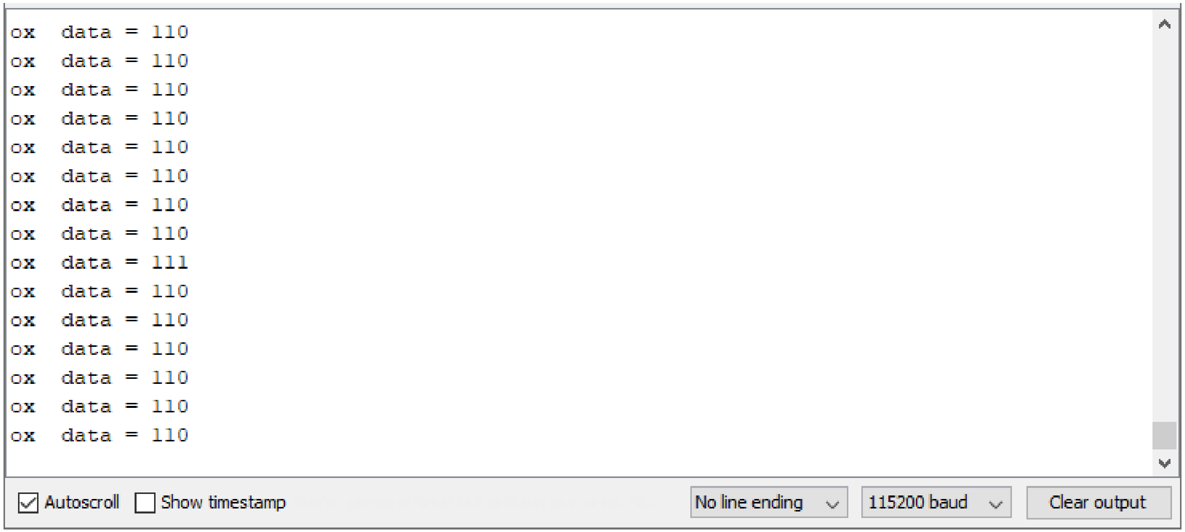
```
/*!
  * @file  getADCData.ino
  * @brief Reading MICS sensor ADC original value
  * @n When using IIC device, select I2C address, set the dialing switch A0, A1 (Address_0
  * @n When using the Breakout version, connect the adcPin and PowerPin
  * @copyright   Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
  * @licence     The MIT License (MIT)
  * @author      ZhixinLiu(zhixin.liu@dfrobot.com)
  * @version     V1.1
  * @date        2021-04-19
  * @get         from https://www.dfrobot.com
  * @url         https://github.com/dfrobot/DFRobot_MICS
  */

#include "DFRobot_MICS.h"

// When using the Breakout version, use the following program to construct an object from
/**!
  adcPin is A0~A5
  adcPin is A0~A5
  powerPin is General IO
*/
#define ADC_PIN   A0
#define POWER_PIN 10

DFRobot_MICS_ADC mics(/*adcPin*/ADC_PIN, /*powerPin*/POWER_PIN);


void setup()
{
  Serial.begin(115200);
  while(!Serial);
  while(!mics.begin()){
    Serial.println("NO Deivces !");
    delay(1000);
  } Serial.println("Device connected successfully !");


  /**!
    Gets the power mode of the sensor
    The sensor is in sleep mode when power is on,so it needs to wake up the sensor.
    The data obtained in sleep mode is wrong
   */
  uint8_t mode = mics.getPowerState();
  if(mode == SLEEP_MODE){
    mics.wakeUpMode();
    Serial.println("wake up sensor success!");
  }else{
    Serial.println("The sensor is wake up mode");
  }
}


void loop()
{
  int16_t ox_data  = 0;
  int16_t red_data = 0;
  /**!
    MICS-5524 Only OX_MODE ADC data can be obtained
    MICS-2714 Only RED_MODE ADC data can be obtained
    MICS-4514 Gravity can obtain AllMode ADC data
  */
  ox_data  = mics.getADCData(OX_MODE);
  //red_data = mics.getADCData(RED_MODE);
  Serial.print("ox  data = ");
  Serial.println(ox_data);
  //Serial.print("red data = ");
  //Serial.println(red_data);
  delay(1000);
}
```

Results

Open the serial port monitor,warm up for about three minutes,and it's enable to you to obtain the original data.

Notes:

- **Two detection units(ox and red) are integrated in MiCS-4514 to calculate the concentration data of different gas. For specific curves, please refer to the sensor data curve in the wiki.**
- **The sensor takes 3 minutes to warm up.**



```
ox  data = 110
ox  data = 110
ox  data = 110
ox  data = 110
ox  data = 110
ox  data = 110
ox  data = 110
ox  data = 110
ox  data = 111
ox  data = 110
ox  data = 110
ox  data = 110
ox  data = 110
ox  data = 110
ox  data = 110
```

☑ Autoscroll ☐ Show timestamp No line ending 115200 baud Clear output

()

Precaution for Use

- Do not expose the sensor to high concentrations of organic solvents, silicone vapor or cigarette smoke to avoid poisoning the sensitive layer.

- For stable performance, preheat the module for about 3 minutes before testing.
- Sensors shall be placed in a filtered enclosure to protect them from water and dust.

Sensor Data Curve

The MiCS-2714 contains a NOx detection unit for detecting the gas concentration data for the NO2, NO, and H2. The reference curve is shown in the figure below.

FAQ

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

More Documents

- MiCS-5524 Datasheet (https://img.dfrobot.com.cn/wiki/5b973267c87e6f19943ab3ad/100cf79c7dd39dd1903fb5043a05d524.pdf)

- Schematics Diagram (https://img.dfrobot.com.cn/wiki/5b973267c87e6f19943ab3ad/3009c1f5b11f521513db3903d33b1209.pdf)

- Dimension and Component Layout (https://img.dfrobot.com.cn/wiki/5b973267c87e6f19943ab3ad/a1a4d9e8c56d96319695a90095ad1c98.pdf)

DFshopping_car1.png Get **Fermion: MEMS Gas Sensor - MiCS-2714 (Breakout)** (https://www.dfrobot.com/product-2435.html) from DFRobot Store or **DFRobot Distributor**. (https://www.dfrobot.com/index.php?route=information/distributorslogo)

Turn to the Top