SKU:SEN0537

Introduction

This ENS160 Air Quality Sensor (I2C), based on ScioSense's new ENS160 sensor chip, is specifically designed for indoor air quality monitoring and offers detection of multiple IAQ data (TVOC, eCO2, AQI). The combination of the innovative TrueVOC™ technology and the metal oxide (MOX) technology brings this sensor superior accuracy, fast response, anti-interference, etc.
With intelligent on-chip algorithms, the ENS160 can directly output rich and easy-to-understand environmental data. Preheat the sensor 3 minutes before use to obtain accurate data more quickly. What's more, the built-in automatic baseline correction algorithm ensures the long-term stability of the sensor.

Air Quality Index

AQI Reference

Level	Description	Suggestion	Recommended Stay Time
5	Extremely bad	In unavoidable circumstances, please strengthen ventilation	Not recommended to stay
4	Bad	Strengthen ventilation, find sources of pollution	Less than one month
3	Moderate	Strengthen ventilation, close to the water source	Less than 12 months
2	Good	Maintain adequate ventilation	Suitable for long-term living
1	Excellent	No suggestion	Suitable for long-term living

eCO2/CO2 Concentration Reference

eCO2/CO2	Level	Result/Suggestion
21500	Terrible	Indoor air pollution is serious/Ventilation is required
1000-1500	Bad	Indoor air is polluted/ Ventilation is recommended
800-1000	Moderate	It is OK to ventilate
600-800	Good	Keep normal
400-600	Eexcellent	No suggestion

TVOC Concentration Reference

TOVC (ppb)	Effects on Human Health
>6000	Headaches and nerve problem
750-6000	Restless and headache
50-750	Restless and uncomfortable
<50	No effect

Features

- Probe shape makes easy installation
- Multiple IAQ outputs (TVOC, eCO2, AQI), various environmental data
- Less than 3 minutes warm-up time, get data faster
- Built-in algorithm, more accurate output
- Automatic baseline correction, more stable for long-term use

Applications

- Air purification/home automation device
- Environmental pollution detection
- HVAC/Ventilation System
- Building Automation/Smart Thermostat
- Electrical Appliance/Range Hood

Specification

- Supply Voltage: 3.3V~5V
- Operating Current: 29mA
- Preheat Time: <3 minutes
- Communication Mode: I2C
- I2C Address: 0x53
- Operating Temperature: -10℃~80℃
- Operating Humidity: 5%RH~95%RH (Non-condensing)
- eCO2 Detection Range: 400ppm~65000ppm
- TVOC Detection Range: 0ppb~65000ppb
- Cable Length: 1m

Pinout

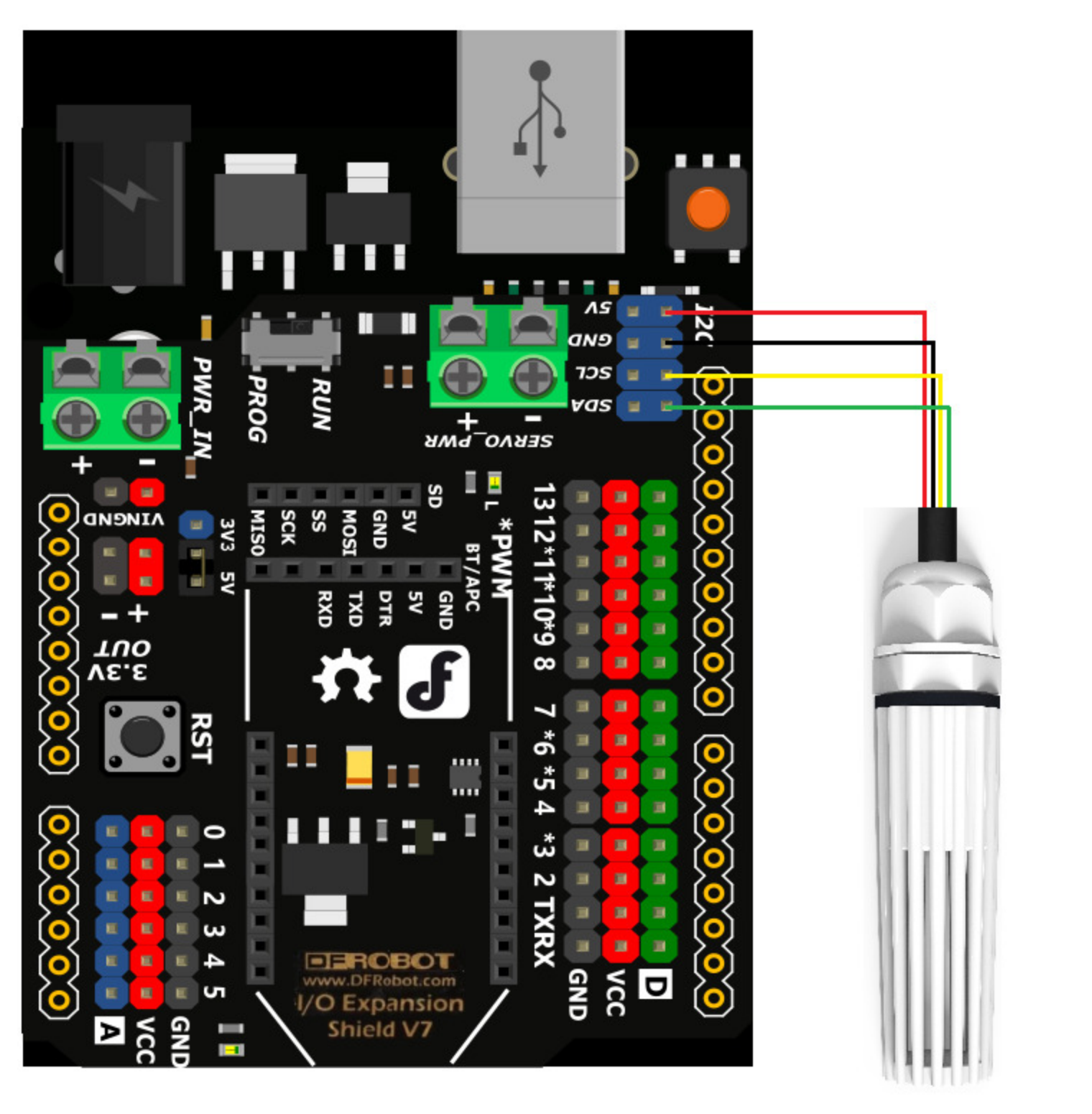
Num	Silkscreen	Description
1	Red Wire	Power +
2	Black Wire	Power -
3	Yellow Wire	I2C clock line
4	Green Wire	I2C data line

Tutorial for Arduino

Requirements

- **Hardware**
  - DFRduino UNO R3 (or similar) x 1
  - ENS160 Air Quality Sensor (I2C) x 1
  - Wires
- **Software**
  - Arduino IDE
  - Download and install the **ENS160 Library & Examples** (About how to install the library?)

Connection Diagram



- For other motherboards, connect to the corresponding SCL and SDA interface

Sample Code - Polling for Data

- Burn the program to DFRuino UNO through Arduino IDE, open the serial monitor to see the printed sensor status, AQI level, and TVOC and eCO2 concentration in sequence.

Sensor Operating Status

Status	Description
0	Operate normally
1	Preheat for 3 minutes when powered on (until no more inital startup status for the sensor)
2	Initial startup, the first 1 hours when powered on (the sensor will no longer be in this status after 24 hours of continuous operation, if there is a power failure during this period, the sensor will still enter the initial startup status after power-on again)

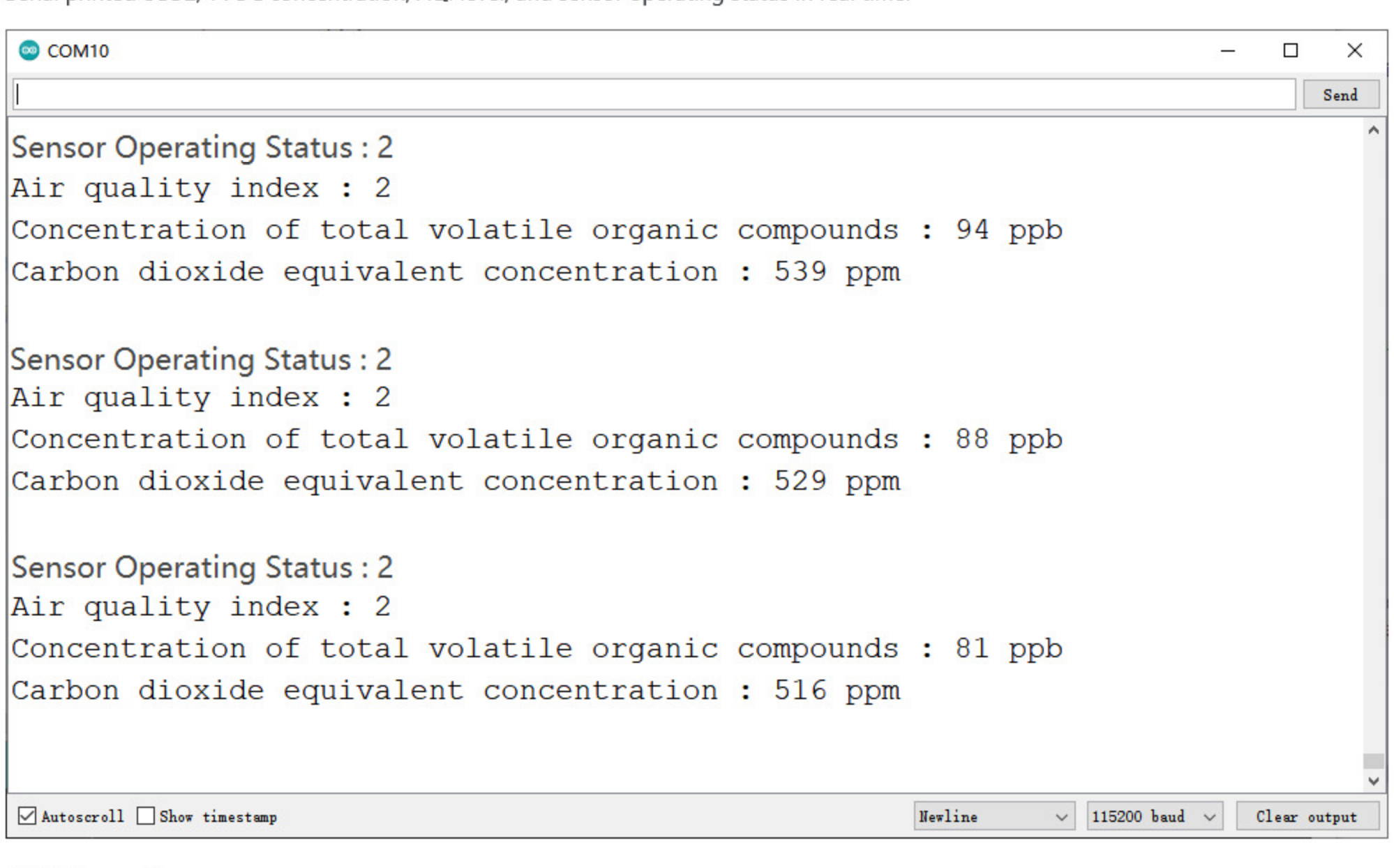
For more details, please refer to Chapter 10 in the Chip Manual.

Note: The ambient temperature and humidity will affect the accuracy of the data, please fill in the current ambient temperature and humidity in the `setTempAndHum(/*temperature=*/temp, /*humidity=*/hum);` function.

```
/*!
 * @file  getMeasureData.ino
 * @brief  Get the sensor data by polling (use 3.3V main controller for Fermion version)
 * @details  Configure the sensor power mode and measurement parameters (for compensating the calibrated temperature and relative humidity in gas measurement)
 * @copyright  Copyright (c) 2010 DFRobot Co.Ltd (http://www.dfrobot.com)
 * @license  The MIT License (MIT)
 * @author  [qsjhyy](yihuan.huang@dfrobot.com)
 * @version  V1.0
 * @date  2021-10-26
 * @url  https://github.com/DFRobot/DFRobot_ENS160
 */
#include <DFRobot_ENS160.h>

DFRobot_ENS160_I2C ENS160(&Wire, /*iicAddr*/ 0x53);

void setup(void)
{
  Serial.begin(115200);

  // Initialize the sensor
  while( NO_ERR != ENS160.begin() ){
    Serial.println("Communication with device failed, please check connection");
    delay(3000);
  }

  /**
   * Set power mode
   * mode Configurable power mode:
   *   ENS160_SLEEP_MODE: DEEP SLEEP mode (low power standby)
   *   ENS160_IDLE_MODE: IDLE mode (No data measurement by the sensor)
   *   ENS160_STANDARD_MODE: STANDARD Gas Sensing Modes
   */
  ENS160.setPWRMode(ENS160_STANDARD_MODE);

  /**
   * Set ambient temperature and humidity for calibration and compensation of the measured data.
   * temperature the current ambient temperature, float type, unit: C
   * humidity    the current ambient humidity, float type, unit: %rH
   */
  ENS160.setTempAndHum(/*temperature=*/25.0, /*humidity=*/50.0);

}

void loop()
{
  /**
   * Get the sensor operating status
   * Return value: 0-Normal operation,
   *         1-Warm-Up phase, During first 3 minutes after power-on.
   *         2-Initial Start-Up phase, During first full hour of operation after initial power-on.Only once in the sensor’s lifetime.
   *           Note that the status will only be stored in the non-volatile memory after an initial 24h of continuous
   *           operation. If unpowered before conclusion of said period, the ENS160 will resume "Initial Start-up" mode
   *           after re-powering.
   */
  uint8_t Status = ENS160.getENS160Status();
  Serial.print("Sensor operating status : ");
  Serial.println(Status);
  /**
   * Get the air quality index
   * Return value: 1-Excellent, 2-Good, 3-Moderate, 4-Poor, 5-Unhealthy
   */
  uint8_t AQI = ENS160.getAQI();
  Serial.print("Air quality index : ");
  Serial.println(AQI);

  /**
   * Get TVOC data
   * Return value range: 0-65000, unit: ppb
   */
  uint16_t TVOC = ENS160.getTVOC();
  Serial.print("Concentration of total volatile organic compounds : ");
  Serial.print(TVOC);
  Serial.println(" ppb");

  /**
   * Get eCO2 data
   * Return value range: 400-65000, unit: ppm
   */
  uint16_t ECO2 = ENS160.getECO2();
  Serial.print("Carbon dioxide equivalent concentration : ");
  Serial.print(ECO2);
  Serial.println(" ppm");

  Serial.println();
  delay(1000);
}
```

Expected Results

Serial printed eCO2, TVOC concentration, AQI level, and sensor operating status in real time.



API Function

```
  /**
   * @fn setPWRMode
   * @brief Set power supply mode
   * @param mode Configurable power mode:
   * @n       ENS160_SLEEP_MODE: DEEP SLEEP mode (low power standby)
   * @n       ENS160_IDLE_MODE: IDLE mode (low-power)
   * @n       ENS160_STANDARD_MODE: STANDARD Gas Sensing Modes
   * @return None
   */
  void setPWRMode(uint8_t mode);

  /**
   * @fn setINTMode
   * @brief Interrupt config (INT)
   * @param mode Interrupt mode to be set, perform OR operation on the following to get mode::
   * @n       Interrupt occur when new data appear in DATA_XXX register (new measured data can be obtained): eINTModeDIS, Disable interrupt; eINTModeEN, Enable interrupt
   * @n       Interrupt pin output driving mode: eINTPinOD, Open drain; eINTPinPP, Push pull
   * @n       Interrupt pin active level: eINTPinActiveLow, Active low; eINTPinActiveHigh, Active high
   * @return None
   */
  void setINTMode(uint8_t mode);

  /**
   * @fn setTempAndHum
   * @brief Users write ambient temperature and relative humidity into ENS160 for calibration and compensation of the measured gas data.
   * @param ambientTemp Compensate the current ambient temperature, float type, unit: C
   * @param relativeHumidity Compensate the current ambient humidity, float type, unit: %rH
   * @return None
   * @return None
   */
  void setTempAndHum(float ambientTemp, float relativeHumidity);

  /**
   * @fn getENS160Status
   * @brief This API is used to get the sensor operating status
   * @return Operating status:
   * @n        eNormalOperation: Normal operation;
   * @n        eWarmUpPhase: Warm-Up phase;
   * @n        eInitialStartUpPhase: Initial Start-Up phase;
   * @n        eInvalidOutput: Invalid output
   */
  uint8_t getENS160Status(void);

  /**
   * @fn getAQI
   * @brief Get the air quality index calculated on the basis of UBA
   * @return Return value range: 1-5 (Corresponding to five levels of Excellent, Good, Moderate, Poor, and Unhealthy)
   */
  uint8_t getAQI(void);

  /**
   * @fn getTVOC
   * @brief Get TVOC concentration
   * @return Return value range: 0-65000, unit: ppb
   */
  uint16_t getTVOC(void);

  /**
   * @fn getECO2
   * @brief Get CO2 equivalent concentration calculated according to the detected data of VOCs and hydrogen (eCO2 – Equivalent CO2)
   * @return Return value range: 400-65000, unit: ppm
   * @note Five levels: Excellent(400 - 600), Good(600 - 800), Moderate(800 - 1000),
   * @n                 Poor(1000 - 1500), Unhealthy(> 1500)
   */
  uint16_t getECO2(void);
```

Tutorial for Raspberry Pi

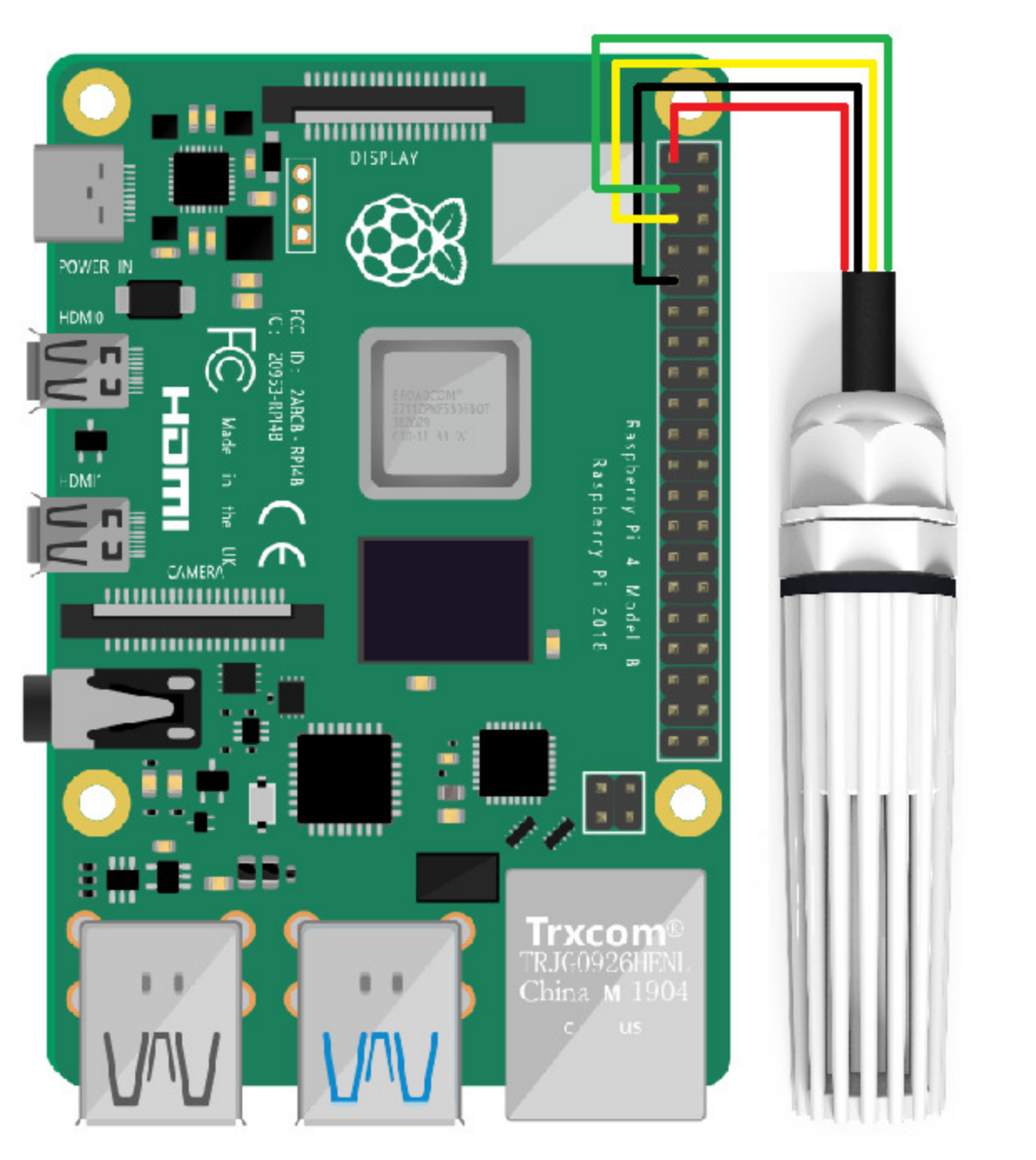
Note: Please let the sensor run for 1 hour first to ensure the accuracy of the data when using it for the first time, and then run (warm up) for 3 minutes later before use. And the ambient temperature and humidity are required for compensating measured data.

Requirements

- **Hardware**
  - Raspberry Pi 4 Model B - 4GB (or similar) x 1
  - ENS160 Air Quality Sensor (I2C) x 1
  - Wires
- **Software**
  - ENS160 Air Quality Sensor Python Library
  - Download RASPBIAN Official OS

Connection Diagram

Connect the module to the Raspberry Pi according to the wiring diagram.



Driver Installing

1.Enable Raspberry Pi I2C. Skip this step if it is already enabled. Open terminal and input the following commands and press "Enter":

`pi@raspberrypi:~ $ sudo raspi-config`

Then use the UP/Down keys to select "Interfacing Options", press Enter, select "P5 I2C" and press Enter to comfirm "Yes". Restart the Raspeberry Pi board.

2.To install Python dependency library and git, the Raspberry Pi needs to be connected to the Internet. Skip this step if already installed. In the terminal, input the following commands and press Enter:

`pi@raspberrypi:~ $ sudo apt-get update pi@raspberrypi:~ $ sudo apt-get install build-essential python-dev python-smbus git`

3.Download the ENS160 driver library. In the terminal, type the following commands and press Enter:

`pi@raspberrypi:~ $ cd Desktop/ pi@raspberrypi:~/Desktop $ git clone https://github.com/DFRobot/DFRobot_ENS160`

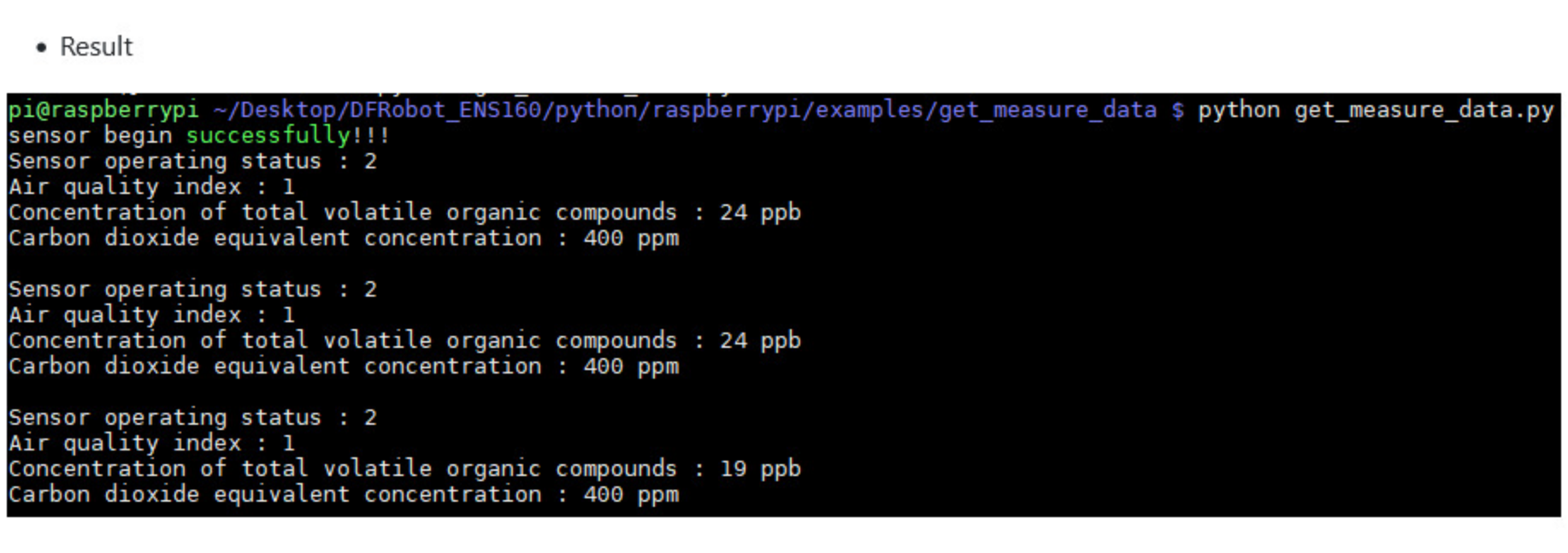
Sample Code - Polling for Data

- In the terminal, input the following command and press Enter to run the sample code:

`pi@raspberrypi:~/Desktop $ cd DFRobot_ENS160/python/raspberrypi/examples/get_measure_data`

`pi@raspberrypi:~/Desktop/DFRobot_ENS160/python/raspberrypi/examples/get_measure_data $python get_measure_data.py`

- Result



FAQ

For any questions, advice or cool ideas to share, please visit the **DFRobot Forum**.

More Documents

- Schematics
- ENS160 Datasheet

Get **ENS160 Air Quality Sensor (I2C)** from DFRobot Store or **DFRobot Distributor**.

Turn to the Top