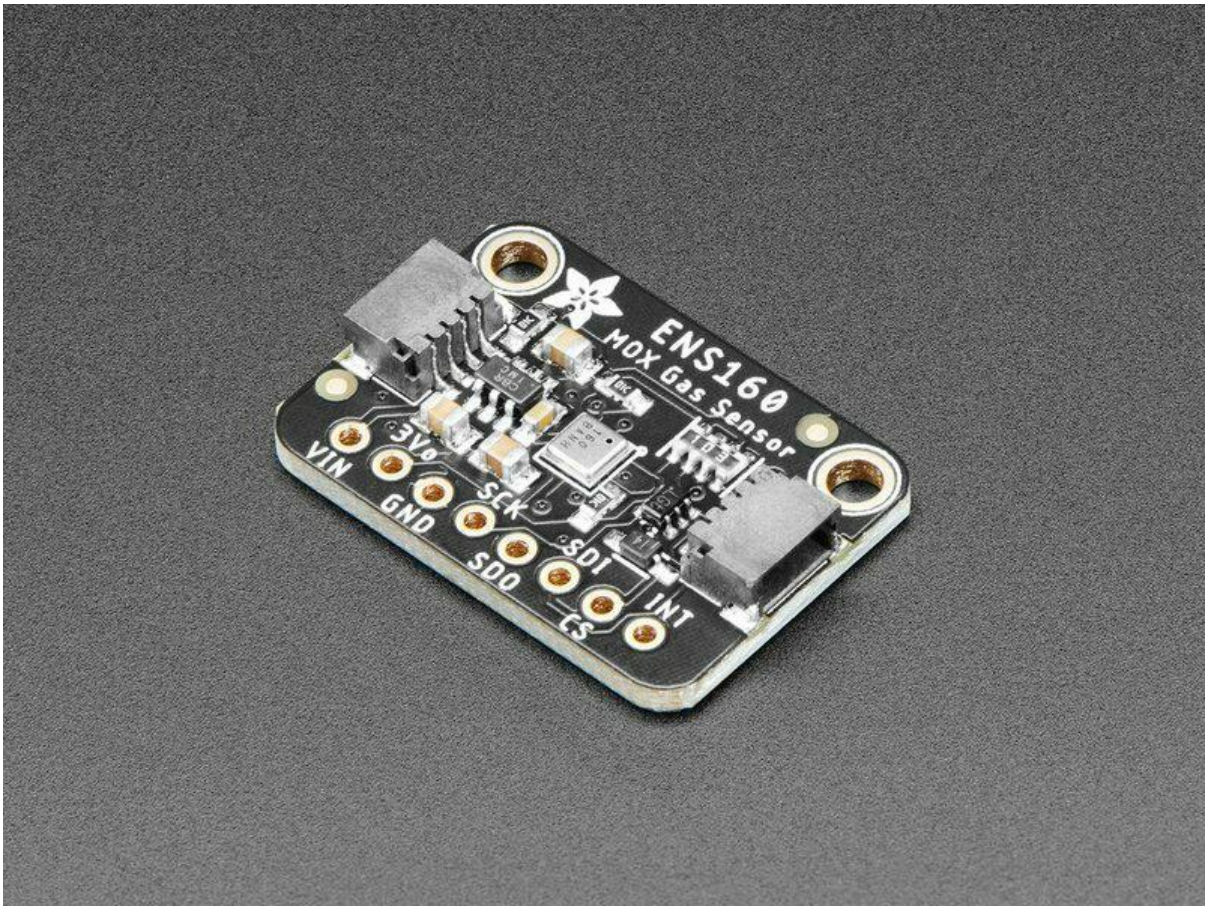
Adafruit ENS160 MOX Gas Sensor

Created by Liz Clark

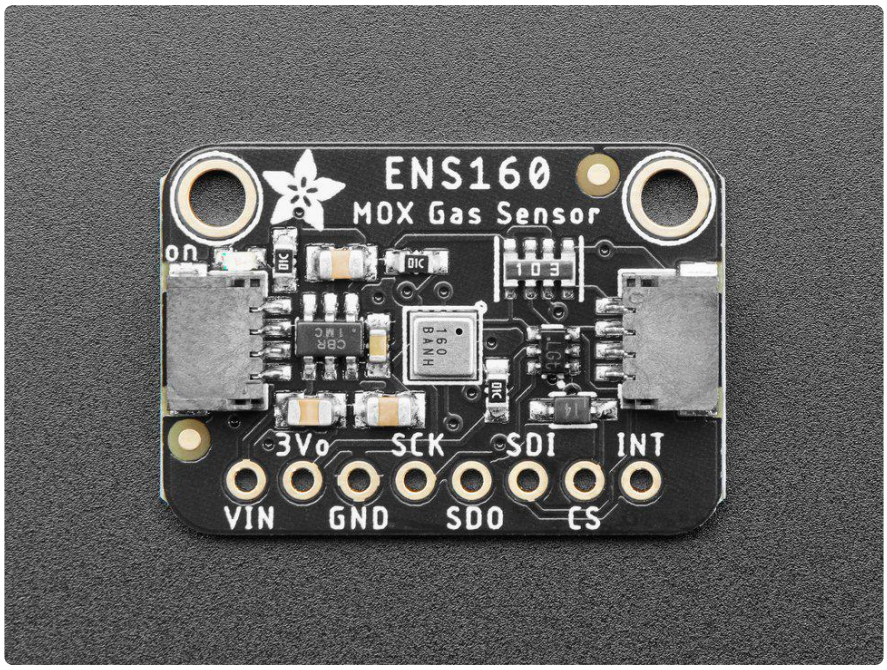


https://learn.adafruit.com/adafruit-ens160-mox-gas-sensor

Last updated on 2022-09-27 07:02:14 PM EDT

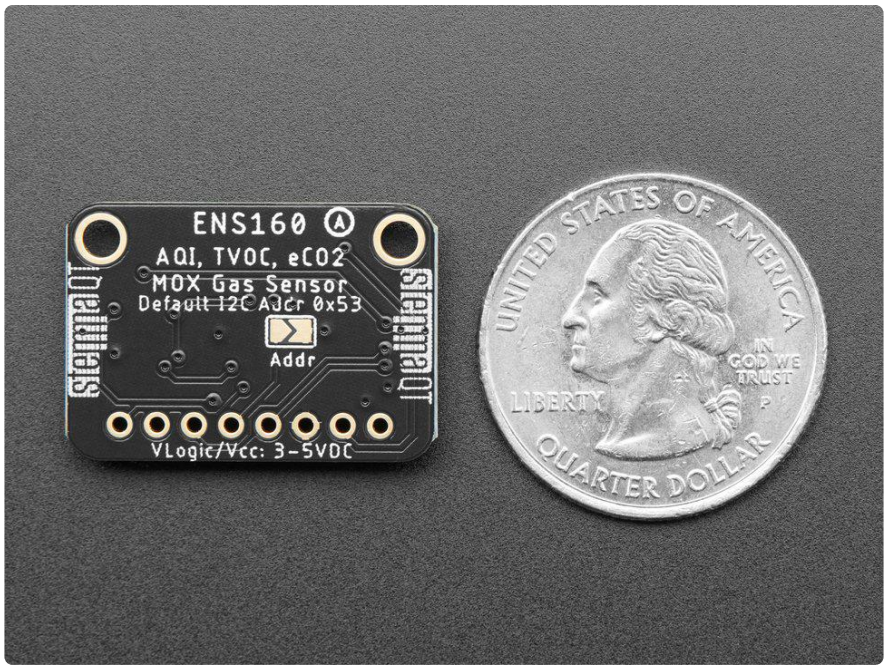
Table of Contents

Overview



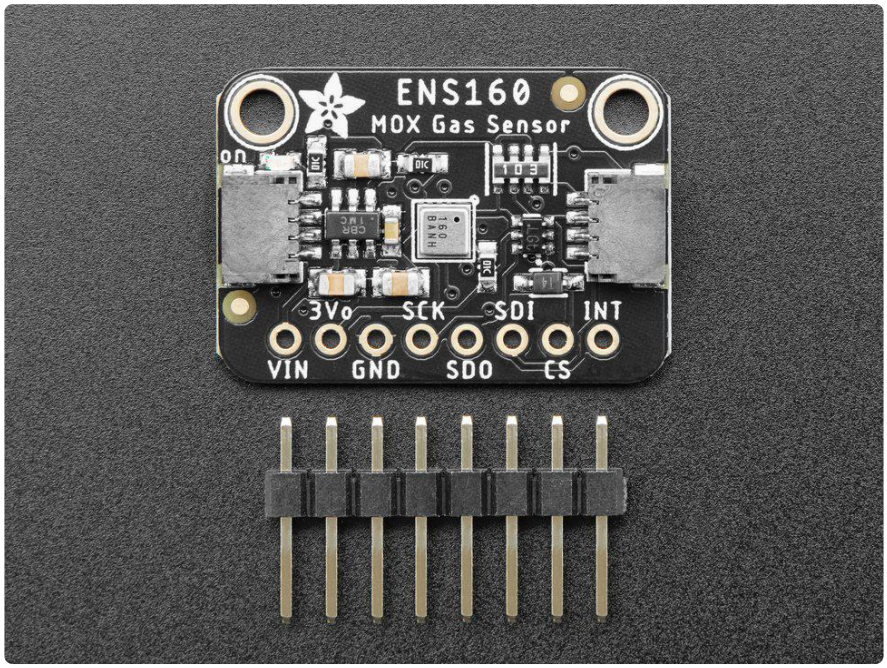
sniff *sniff* ... do you smell that? No need to stick your nose into a carton of milk anymore, you can build a digital nose with the ENS160 Gas Sensor, a fully integrated MOX gas sensor. This is a very fine air quality sensor from the sensor experts at ScioSense (https://adafru.it/11dX), with I2C interfacing so you don't have to manage the heater and analog reading of a MOX sensor. It combines multiple metal-oxide sensing and heating elements on one chip to provide more detailed air quality signals.

The ENS160 is the replacement for the popular, but now-discontinued CCS811 (https://adafru.it/11dY). It has similar functionality but does require all new driver code so be aware if you are updating from an original design with the CCS811 that some work is required to upgrade.

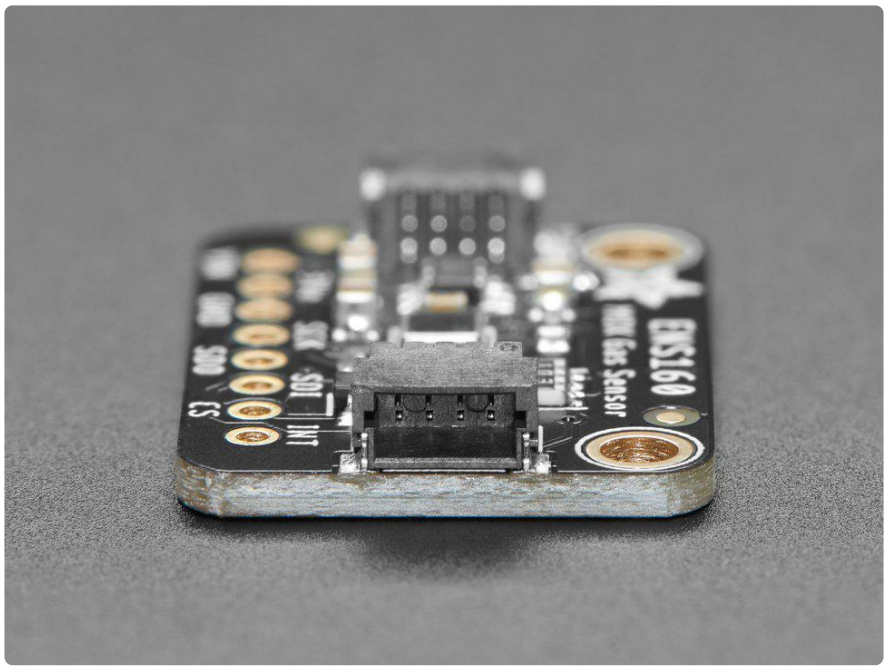


The ENS160 has a 'standard' hot-plate MOX sensor, as well as a small microcontroller that controls power to the plate, reads the analog voltage, and provides an I2C interface to read from. ScioSense provides an Arduino library with examples of reading the four raw resistance values and also the TVOC and eCO2 (https://adafru.it/PaX) and a Python/CircuitPython library (https://adafru.it/11dZ) that can be used with Linux computers like the Raspberry Pi or our CircuitPython boards.

Please note, this sensor, like all VOC/gas sensors, has variability, and to get precise measurements you will want to calibrate it against known sources! That said, for general environmental sensors, it will give you a good idea of trends and comparison.



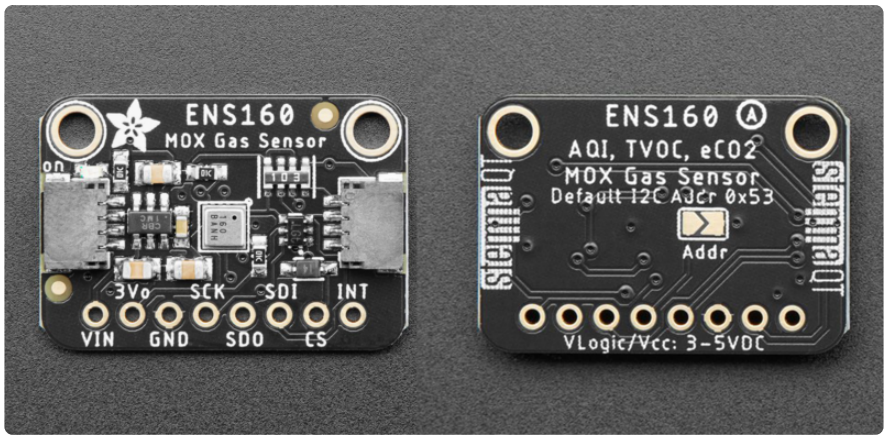
Another nice element to this sensor is the ability to set temperature and humidity compensation for better accuracy (https://adafru.it/11d-). An external humidity sensor is required and then the RH% is written over I2C to the sensor, so it can better calibrate the MOX sensor reading and reduce humidity/temperature-based variations.



Nice sensor right? So we made it easy for you to get right into your next project. The surface-mount sensor is soldered onto a custom-made PCB in the STEMMA QT form factor (https://adafru.it/LBQ), making them easy to interface with. The STEMMA QT connectors (https://adafru.it/JqB) on either side are compatible with the SparkFun Qwiic (https://adafru.it/Fpw) I2C connectors. This allows you to make solderless connections between your development board and the ENS160 or to chain it with a wide range of other sensors and accessories using a compatible cable (https://adafru.it/JnB). QT Cable is not included, but we have a variety in the shop (https://adafru.it/JnB)

We've of course broken out all the pins to standard headers and added a 3.3V voltage regulator and level shifting to allow you to use it with either 3.3V or 5V systems such as the Arduino Uno, or Feather M4.

Pinouts



The default I2C address is 0x53.

Power Pins

- VIN - this is the power pin. Since the gas sensor chip may use 3 VDC, we have included a voltage regulator on board that will take 3-5VDC and safely convert it down. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V microcontroller like Arduino, use 5V.
- 3Vo - this is the 3.3V output from the voltage regulator, you can grab up to 100mA from this if you like.
- GND - common ground for power and logic.

I2C Logic Pins

- SCL (labeled SCK) - I2C clock pin, connect to your microcontroller I2C clock line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
- SDA (labeled SDI) - I2C data pin, connect to your microcontroller I2C data line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
- STEMMA QT (https://adafru.it/Ft4) - These connectors allow you to connectors to dev boards with STEMMA QT connectors or to other things with various associated accessories (https://adafru.it/Ft6).

Address Pin

On the back of the board is one address jumper, labeled Addr. This jumper allows you to chain up to 2 of these boards on the same pair of I2C clock and data pins. To do so, you solder the jumper "closed" by connecting the two pads.

On the front of the board is one address pin, labeled SDO. Just like the jumper, this pin allows you to change the I2C address to connect multiple boards by connecting it to GND.

The default I2C address is 0x53. The other address option can be calculated by “subtracting” the Addr from the base of 0x53.

Addr sets the lowest bit with a value of 1. The final address is 0x53 - Addr which would be 0x52.

The table below shows all possible addresses, and whether the pin should be high (open) or low (closed).

ADDR	Addr
0x53	H
0x52	L

SPI Logic Pins

- SCK - This is the SPI clock pin, its an input to the gas sensor.
- SDO - This is the Serial Data Out pin, for data sent from the gas sensor to your processor.
- SDI - This is the Serial Data In pin, for data sent from your processor to the gas sensor.
- CS - This is the Chip Select pin, drop it low to start an SPI transaction. Its an input to the gas sensor.

Please note the libraries do not support the SPI interface.

Interrupt Pin

- INT - Interrupt signal output. Can be pulled low on sensor reading thresholds.

Power LED

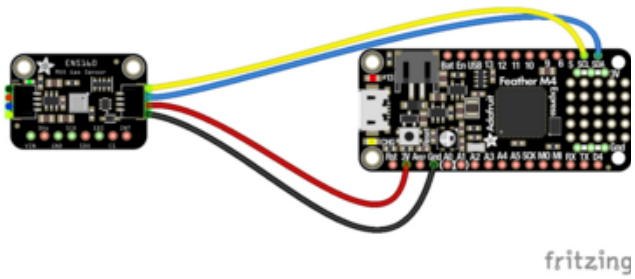
- Power LED - In the upper left corner, above the STEMMA connector, on the front of the board, is the power LED, labeled on. It is the green LED.

CircuitPython & Python

It's easy to use the ENS160 with Python or CircuitPython, and the Adafruit_CircuitPython_ENS160 (https://adafru.it/11dZ) module. This module allows you to easily write Python code that allows you to read the ENS160 gas sensor. You can use this gas sensor with any CircuitPython microcontroller board or with a computer that has GPIO and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library (https://adafru.it/BSN).

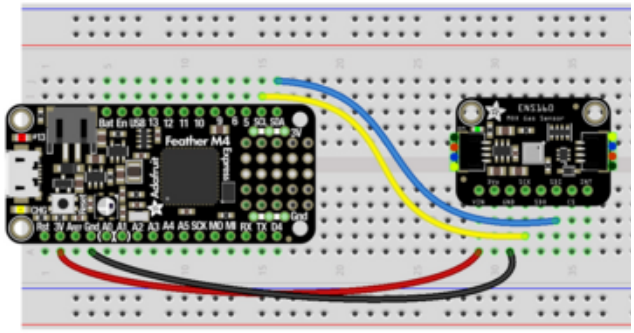
CircuitPython Microcontroller Wiring

First, wire up an ENS160 to your board exactly as shown below. Here's an example of wiring a Feather M4 to the ENS160 with I2C using one of the handy STEMMA QT (https://adafru.it/Ft4) connectors:



Board 3V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)

You can also use standard 0.100" pitch headers to wire it up on a breadboard:

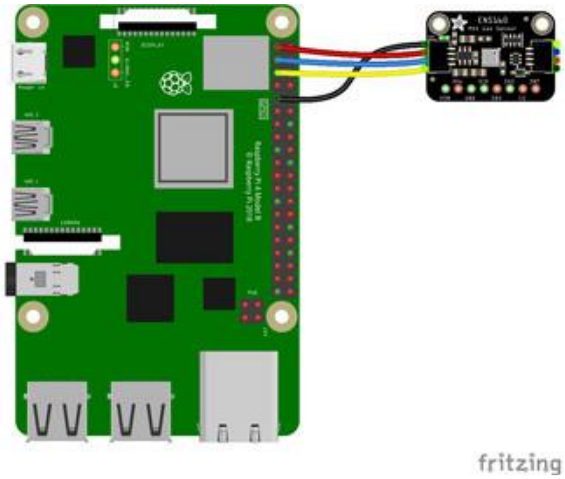


Board 3V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)

Python Computer Wiring

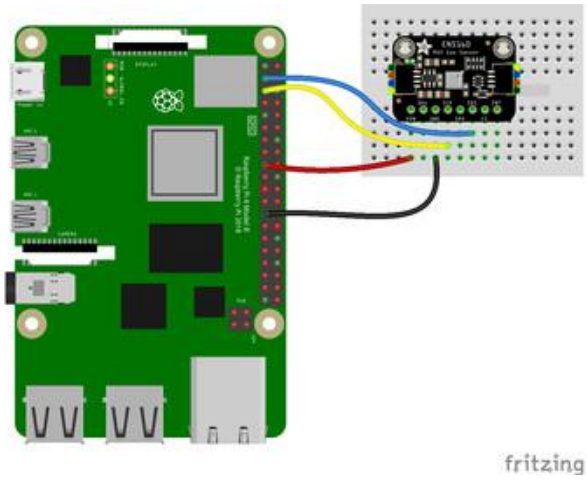
Since there's dozens of Linux computers/boards you can use, below shows wiring for Raspberry Pi. For other platforms, please visit the guide for CircuitPython on Linux to see whether your platform is supported (https://adafru.it/BSN).

Here's the Raspberry Pi wired to the gas sensor using I2C and a STEMMA QT (https://adafru.it/Ft4) connector:



Pi 3V to sensor VIN (red wire)
Pi GND to sensor GND (black wire)
Pi SCL to sensor SCL (yellow wire)
Pi SDA to sensor SDA (blue wire)

Finally here is an example of how to wire up a Raspberry Pi to the gas sensor using a solderless breadboard:



Pi 3V to sensor VIN (red wire)
Pi GND to sensor GND (black wire)
Pi SCL to sensor SCL (yellow wire)
Pi SDA to sensor SDA (blue wire)

Python Installation of ENS160 Library

You'll need to install the Adafruit_Blinka library that provides the CircuitPython support in Python. This may also require enabling I2C on your platform and verifying you are running Python 3. Since each platform is a little different, and Linux changes often, please visit the CircuitPython on Linux guide to get your computer ready (https://adafru.it/BSN)!

Once that's done, from your command line run the following command:

- `pip3 install adafruit-circuitpython-ens160`

If your default Python is version 3, you may need to run `pip` instead. Make sure you aren't trying to use CircuitPython on Python 2.x, it isn't supported!

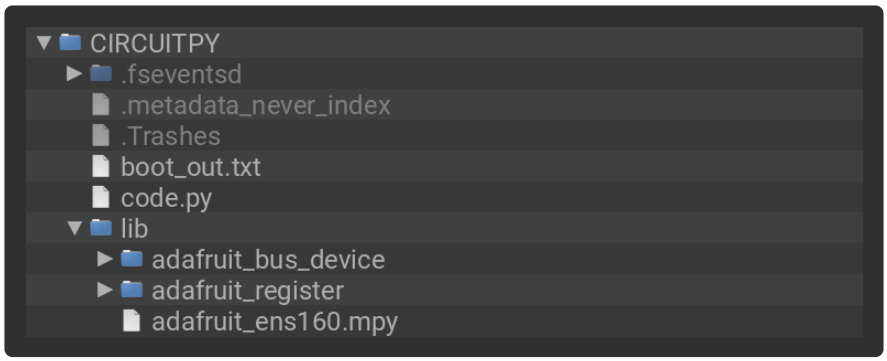
CircuitPython Usage

To use with CircuitPython, you need to first install the ENS160 library, and its dependencies, into the lib folder on your CIRCUITPY drive. Then you need to update code.py with the example script.

Thankfully, we can do this in one go. In the example below, click the Download Project Bundle button below to download the necessary libraries and the code.py file in a zip file. Extract the contents of the zip file, and copy the entire lib folder and the c ode.py file to your CIRCUITPY drive.

Your CIRCUITPY/lib folder should contain the following folders and file:

- adafruit_bus_device/
- adafruit_register/
- adafruit_ens160.mpy



Python Usage

Once you have the library `pip3` installed on your computer, copy or download the following example to your computer, and run the following, replacing code.py with whatever you named the file:

```
python3 code.py
```

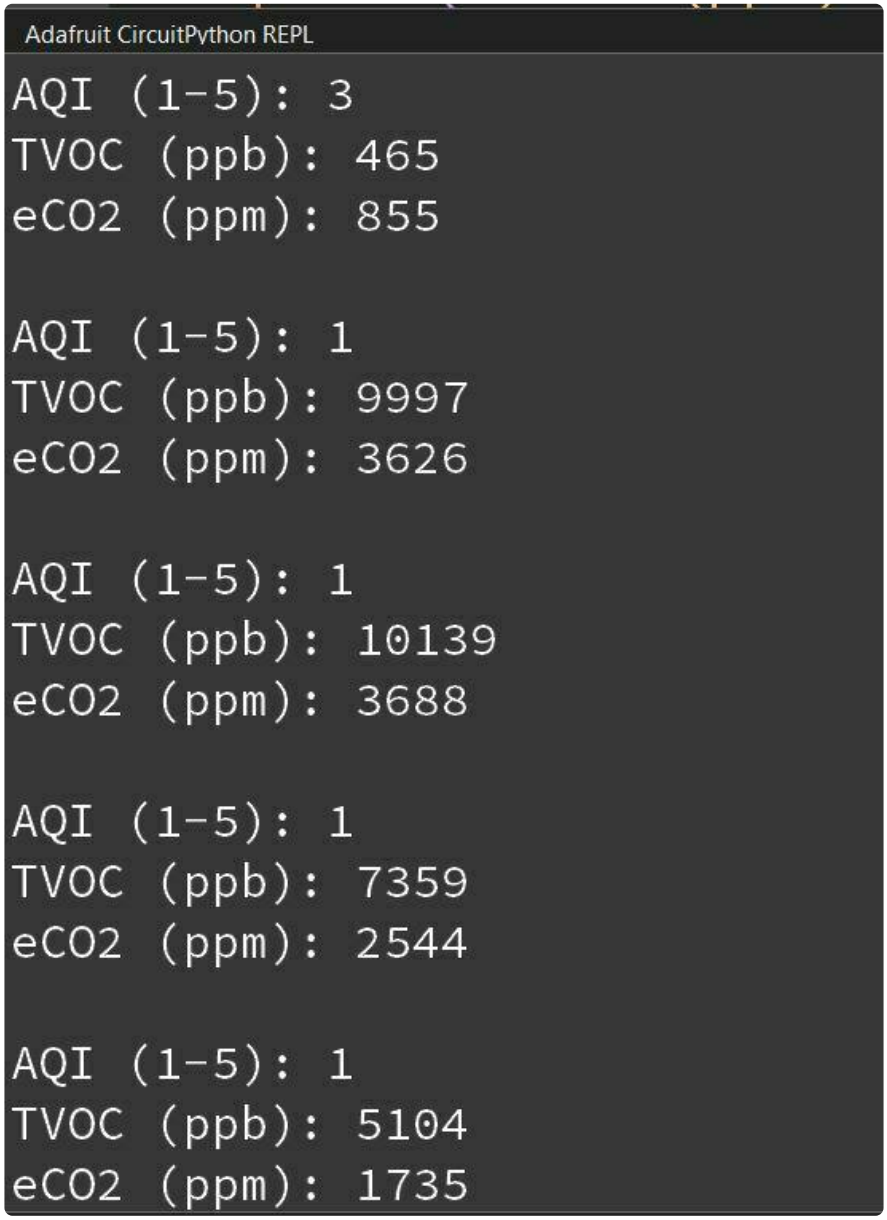
Example Code

```
# SPDX-FileCopyrightText: Copyright (c) 2022 ladyada for Adafruit Industries
#
# SPDX-License-Identifier: Unlicense

import time
import board
import adafruit_ens160

i2c = board.I2C()  # uses board.SCL and board.SDA

ens = adafruit_ens160.ENS160(i2c)

# Set the temperature compensation variable to the ambient temp
# for best sensor calibration
ens.temperature_compensation = 25
# Same for ambient relative humidity
ens.humidity_compensation = 50


while True:
    print("AQI (1-5):", ens.AQI)
    print("TVOC (ppb):", ens.TVOC)
    print("eCO2 (ppm):", ens.eCO2)
    print()
```

```
    # new data shows up every second or so
    time.sleep(1)
```

If running CircuitPython: Once everything is saved to the CIRCUITPY drive, connect to the serial console (https://adafru.it/Bec) to see the data printed out!

If running Python: The console output will appear wherever you are running Python.



In the example, the gas sensor is instantiated on I2C. Then, in the loop, `AQI`, `TVOC` and `eCO2` readings are printed to the REPL every second.

Python Docs

Python Docs (https://adafru.it/11cM)

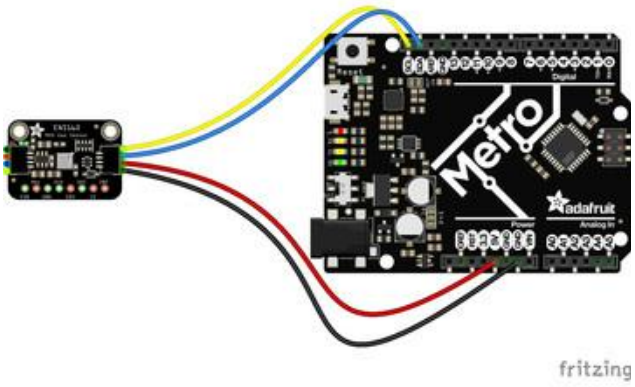
Arduino

Using the ENS160 gas sensor with Arduino involves wiring up the sensor to your Arduino-compatible microcontroller, installing the ScioSense ENS160 (https://adafru.it/11cN) library and running the provided example code.

Wiring

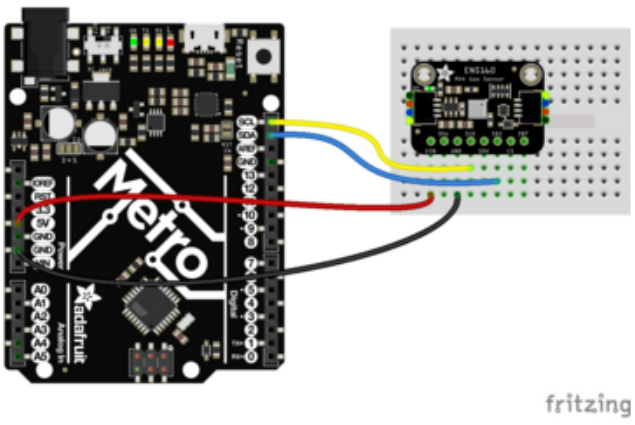
Wire as shown for a 5V board like an Uno. If you are using a 3V board, like an Adafruit Feather, wire the board's 3V pin to the ENS160 VIN.

Here is an Adafruit Metro wired up to the ENS160 using the STEMMA QT connector:



Board 5V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)

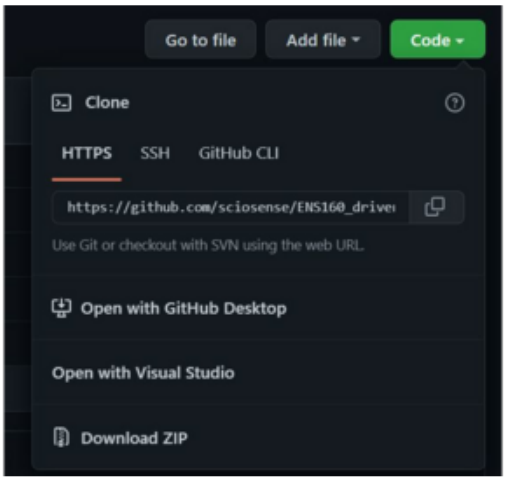
Here is an Adafruit Metro wired up using a solderless breadboard:



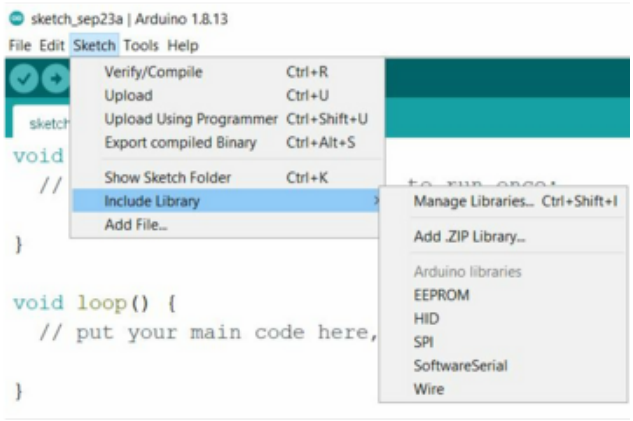
Board 5V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)

Library Installation

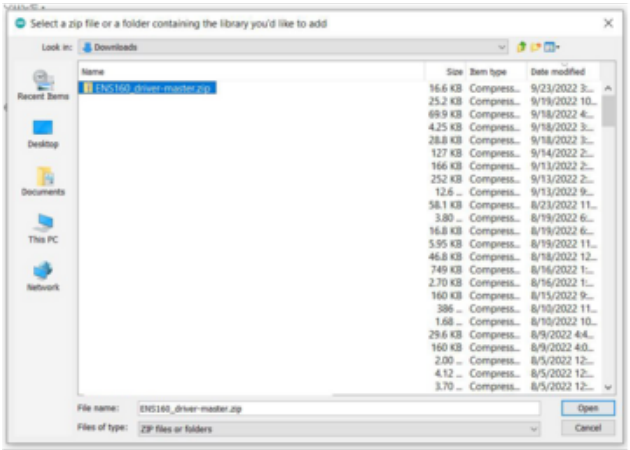
You can install the ScioSense ENS160_driver library for Arduino by downloading it from GitHub (https://adafru.it/11cN) and adding it to the Arduino IDE.



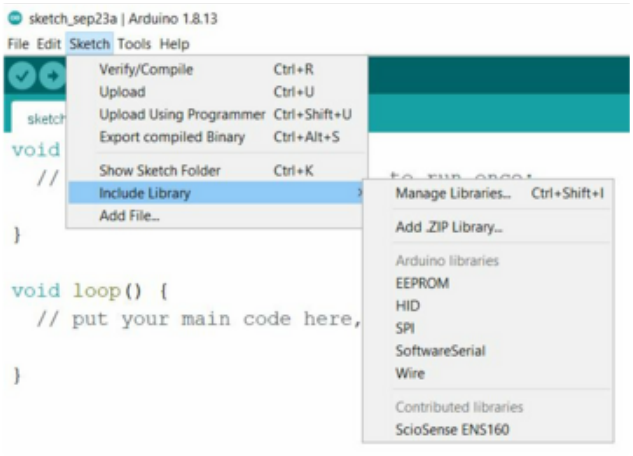
In the GitHub repository, click on the green Code button. In the dropdown menu, click on Download ZIP. This downloads a zipped file folder containing the library.



In the Arduino IDE, click on Sketch - Include Library - Add .ZIP Library...



A file directory will open. Navigate to your downloaded .ZIP file folder and click Open.

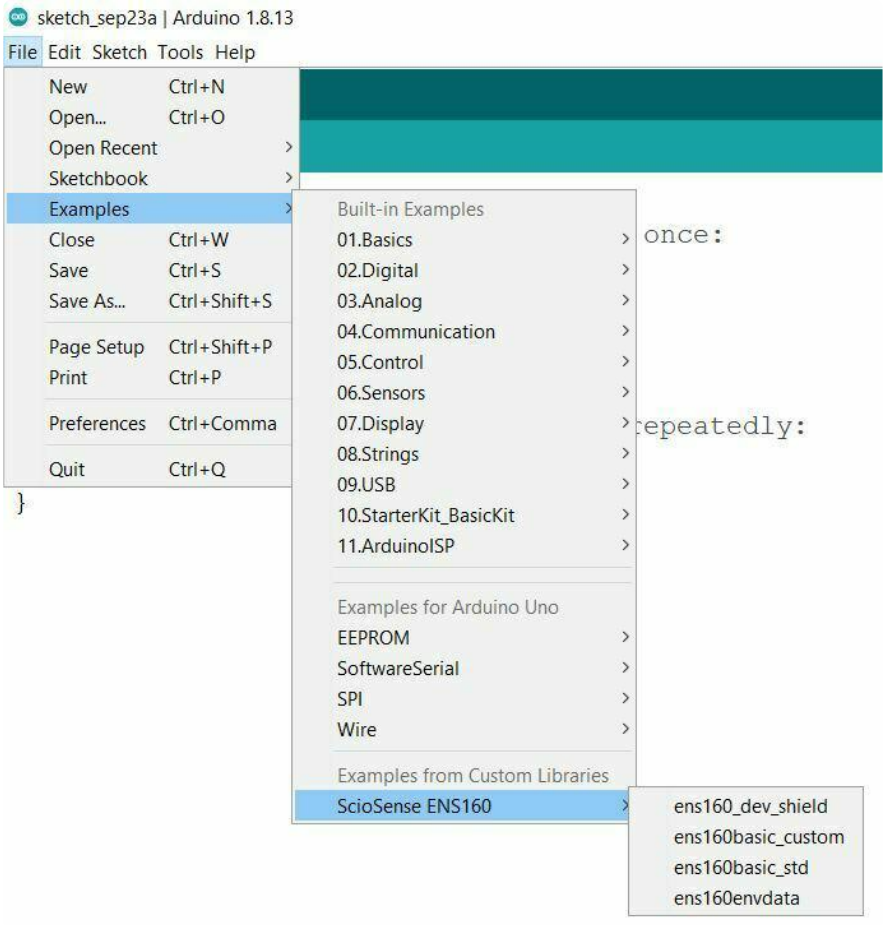


To confirm that the library has been installed, in the Arduino IDE click on Sketch - Include Library. You should see the ScioSense ENS160 library listed under Contributed libraries.

After confirming, close out and relaunch the Arduino IDE.

Make sure to relaunch the Arduino IDE after installing the library.

Example Code



To launch the example code in the Arduino IDE, click on File - Examples - ScioSense ENS160 and select ens160basic_std. The example code can also be found on GitHub in the library repository (https://adafru.it/11e0).

I2C Address

In the Arduino library, `ENS160_I2CADDR_0` is I2C address `0x52` and `ENS160_I2CADDR_1` is I2C address `0x53`.

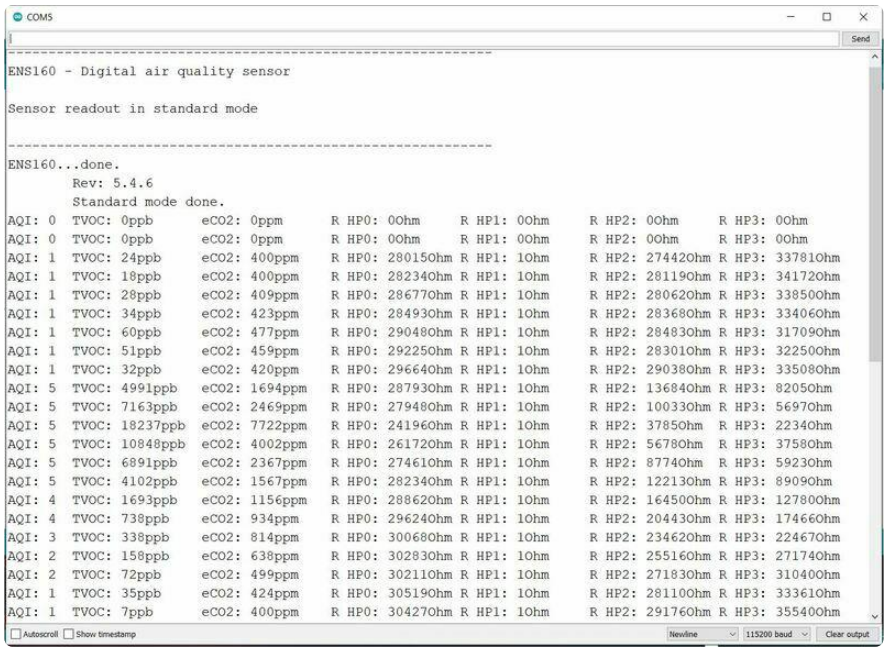
The default I2C address for the breakout board is `0x53`. To run the example code for address `0x53`, edit the top of the example code by commenting out `ENS160_I2CADDR_0` and uncommenting `ENS160_I2CADDR_1`:

```
#include "ScioSense_ENS160.h"  // ENS160 library
//ScioSense_ENS160      ens160(ENS160_I2CADDR_0);
ScioSense_ENS160      ens160(ENS160_I2CADDR_1);
```

To use the sensor with address `0x52`, leave the example code as-is with `ENS160_I2CADDR_0` uncommented.

```
#include "ScioSense_ENS160.h"  // ENS160 library
ScioSense_ENS160      ens160(ENS160_I2CADDR_0);
//ScioSense_ENS160      ens160(ENS160_I2CADDR_1);
```

```
/*------------------------------------------------------------------------
  MAIN LOOP FUNCTION
  Cylce every 1000ms and perform measurement
 ------------------------------------------------------------------------*/

void loop() {

  if (ens160.available()) {
    ens160.measure(0);

    Serial.print("AQI: ");Serial.print(ens160.getAQI());Serial.print("\t");
    Serial.print("TVOC: ");Serial.print(ens160.getTVOC());Serial.print("ppb\t");
    Serial.print("eCO2: ");Serial.print(ens160.geteCO2());Serial.print("ppm\t");
    Serial.print("R HP0: ");Serial.print(ens160.getHP0());Serial.print("Ohm\t");
    Serial.print("R HP1: ");Serial.print(ens160.getHP1());Serial.print("Ohm\t");
    Serial.print("R HP2: ");Serial.print(ens160.getHP2());Serial.print("Ohm\t");
    Serial.print("R HP3: ");Serial.print(ens160.getHP3());Serial.println("Ohm");
  }
  delay(1000);
}
```



Upload the sketch to your board and open up the Serial Monitor (Tools -> Serial Monitor) at 115200 baud. You should see that the sketch has found your connected I2C ENS160 gas sensor. Then, sensor readings for AQI, TVOC, eCO2 and the raw resistance values of the four hot plates in the sensor are printed to the Serial Monitor every second.

Arduino Docs

Arduino Docs (https://adafru.it/11cN)

Downloads

Files

- ENS160 Datasheet (https://adafru.it/11e1)
- EagleCAD PCB files on GitHub (https://adafru.it/11e2)
- Fritzing object in the Adafruit Fritzing Library (https://adafru.it/11e3)

Schematic and Fab Print