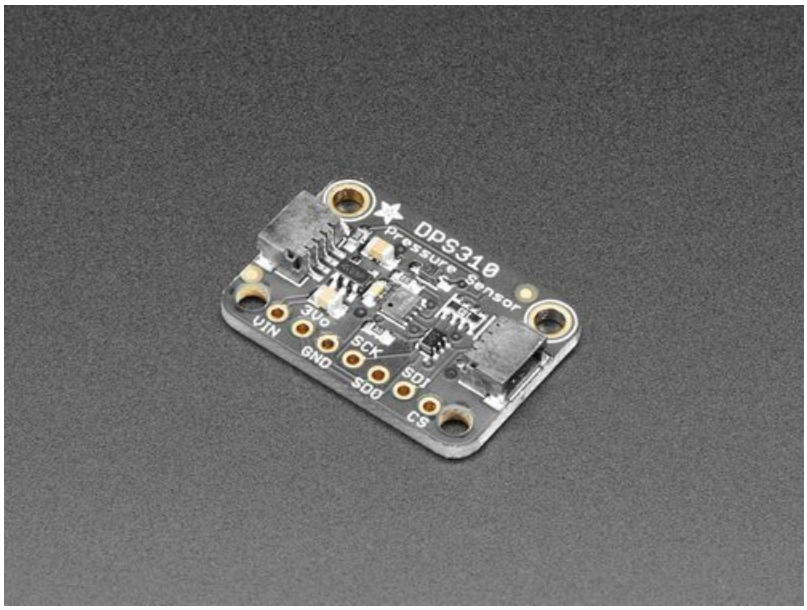
adafruit learning system

Adafruit DPS310 Precision Barometric Pressure and Altitude Sensor

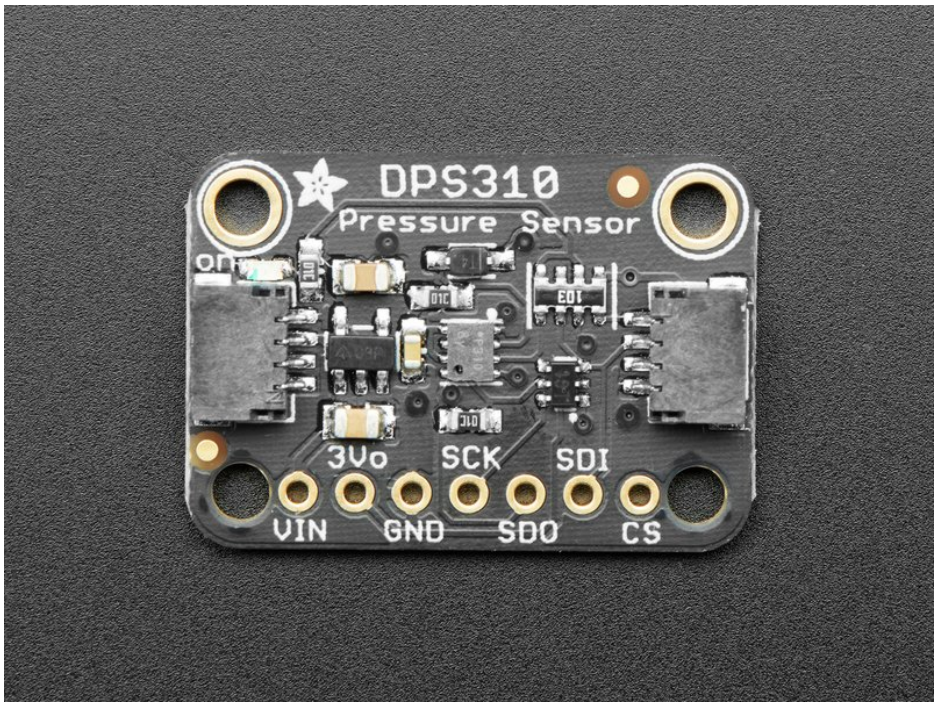
Created by Kattni Rembor



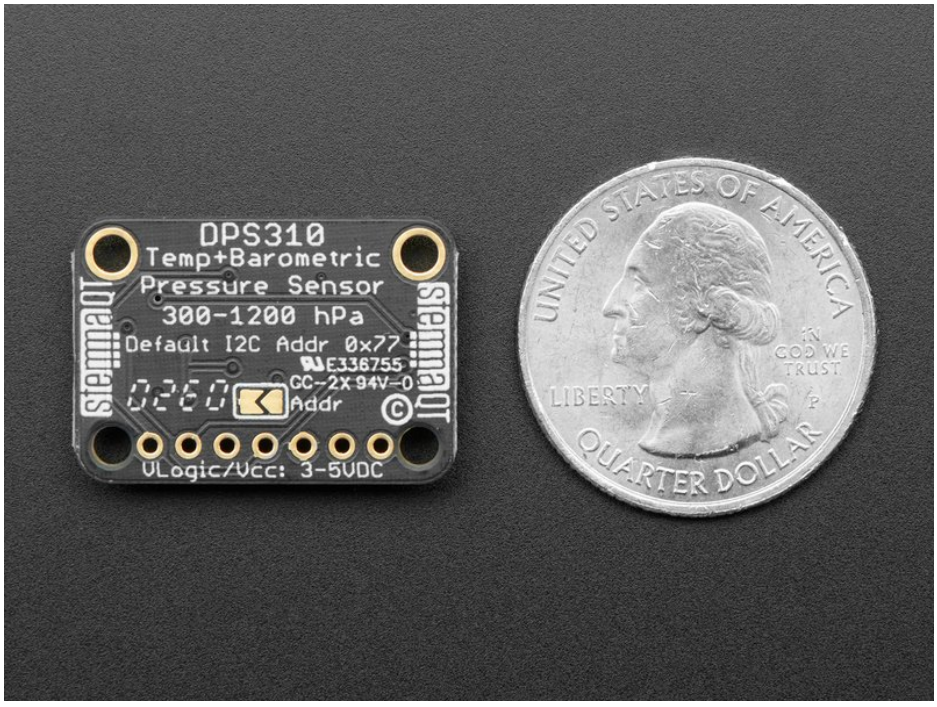
Last updated on 2021-05-06 06:53:11 PM EDT

Guide Contents

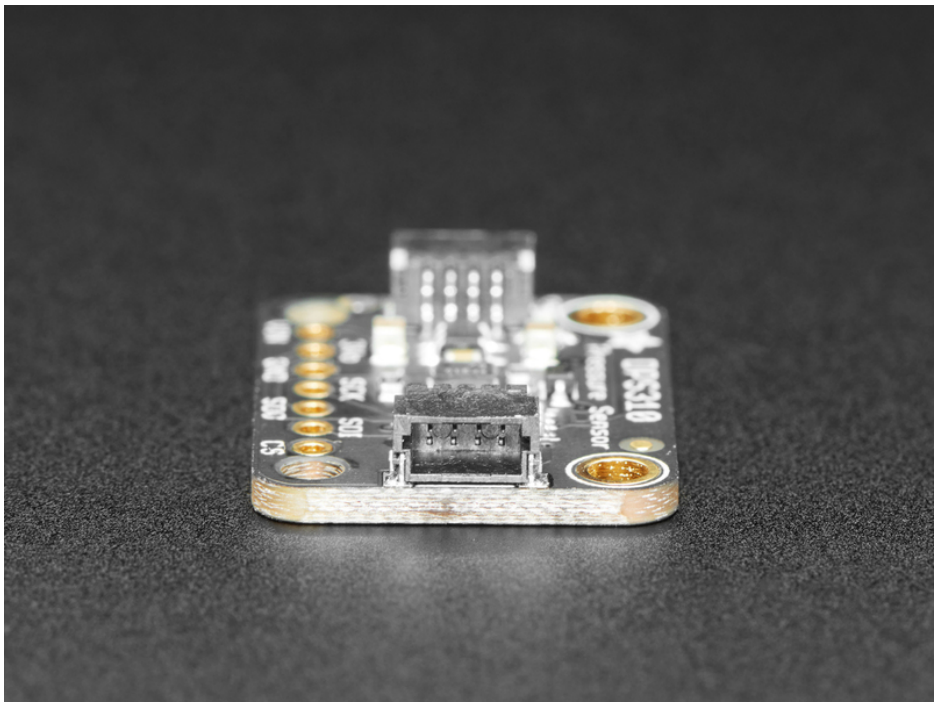
Overview



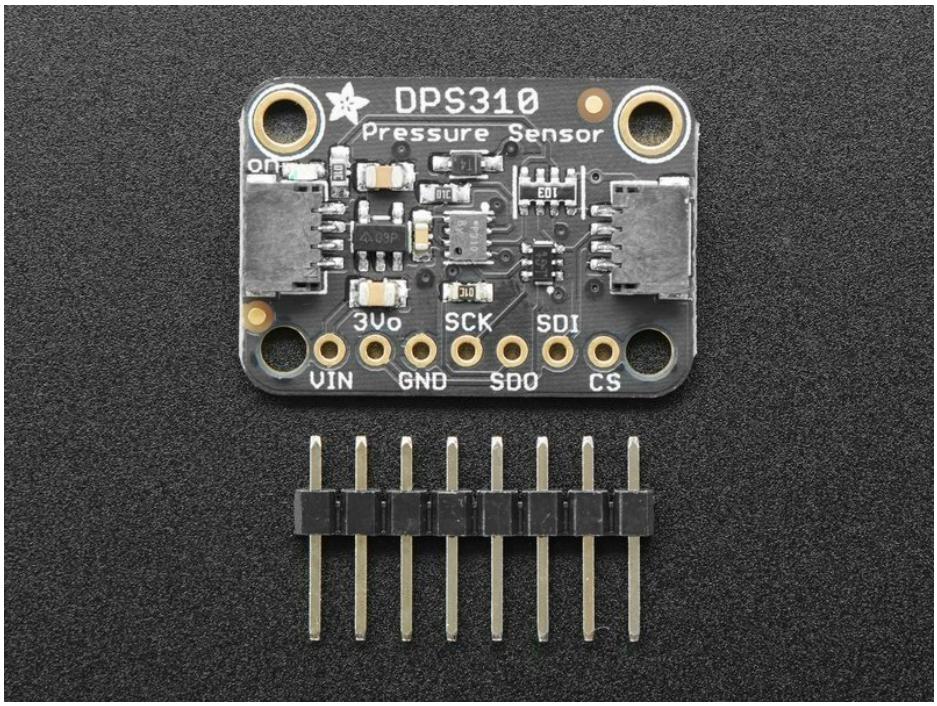
How high are you right now? If you had a precision altitude sensor, you would know for sure! The DPS310 sensor from Infineon a high precision barometric sensor, perfect for measuring altitude changes with a up to ±0.002 hPa (or ±0.02 m) precision high precision mode and ± 1 hPa absolute accuracy. That means you can know your absolute altitude with 1 meter accuracy when you set the sea-level pressure, and measure changes in altitude with up to 2 cm precision. This makes it a great sensor for use in drones or other altitude-sensitive robots. This sensor would also do well in any environmental sensing kit, you can use it to predict weather system changes



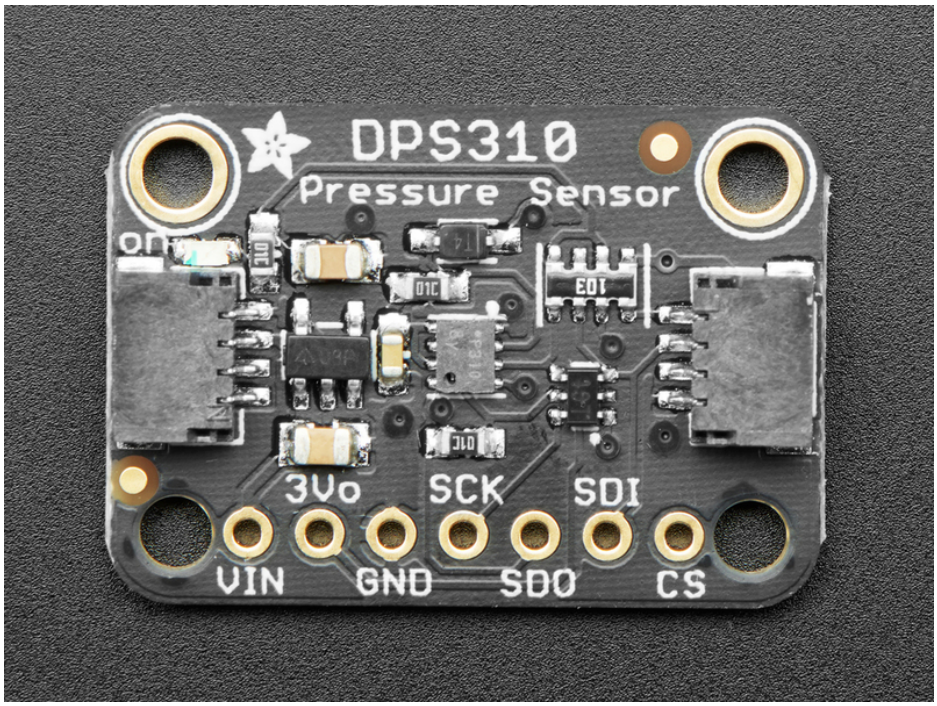
You can use this sensor with either I2C or SPI, so it's easy to integrate into projects. It also has a temperature sensor built in, with ± 0.5°C accuracy. For the lowest noise readings, set it up to take multiple measurements and perform a low-pass filter, that capability is built in! You can use it from 300 to 1200 hPa and in ambient temperature ranges from -40 to 85 °C.



To make life easier, so you can focus on your important work, we've taken the sensor and put it onto a breakout PCB along with support circuitry to let you use it with 3.3V (Feather/Raspberry Pi) or 5V (Arduino/ Metro328) logic levels. Additionally, since it speaks I2C, you can easily connect it up with two wires (plus power and ground!). We've even included SparkFun qwiic (https://adafru.it/Fpw) compatible STEMMA QT **(https://adafru.it/Ft4)** connectors for the I2C bus so **you don't even need to solder!** Just wire up to your favorite microcontroller and you can use our CircuitPython/Python or Arduino drivers (https://adafru.it/IFR) to easily interface with the DPS310.


DPS310
Pressure Sensor
3Vo
SCK
SDI
VIN
GND
SDO
CS

Pinouts



Power Pins

- **Vin** - this is the power pin. Since the sensor chip uses 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:

- **SCK** - This is *also* the I2C **clock pin SCL**, connect to your microcontroller's I2C clock line. This pin is level shifted so you can use 3-5V logic, and there's a **10K pullup** on this pin.
- **SDI** - This is *also* the I2C **data pin SDA**, connect to your microcontroller's I2C data line. This pin is level shifted so you can use 3-5V logic, and there's a **10K pullup** on this pin.
- **SDO** - This is *also* the I2C address pin **ADR**. Pulling this pin **low to GND** or bridging the solder jumper on the back will change the I2C address from **0x77** to **0x76**
- STEMMA QT **(https://adafru.it/Ft4) -** These connectors allow you to connect to dev boards with **STEMMA QT** connectors or to other things with various associated accessories (https://adafru.it/Ft6)

SPI Logic pins:

All pins going into the breakout have level shifting circuitry to make them 3-5V logic level safe. Use whatever logic level is on **Vin!**

- **SCK** - The **S**PI **C**lock pin, it's an input to the chip
- **SDO** - The **S**erial **D**ata **O**ut / **M**icrocontroller **I**n **S**ensor **O**ut, for data sent from the DPS310 to your processor.
- **SDI** - The **S**erial **D**ata **I**n / **M**icrocontroller **O**ut **S**ensor **I**n pin, for data sent from your processor to the

DPS310

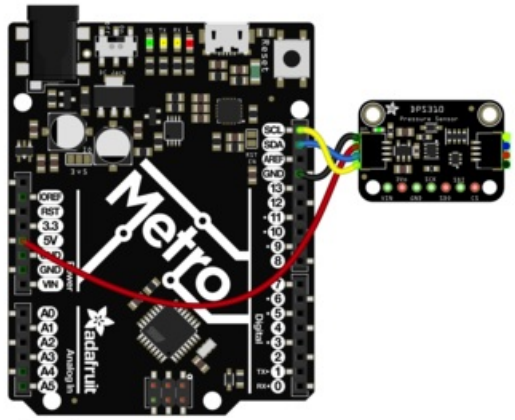
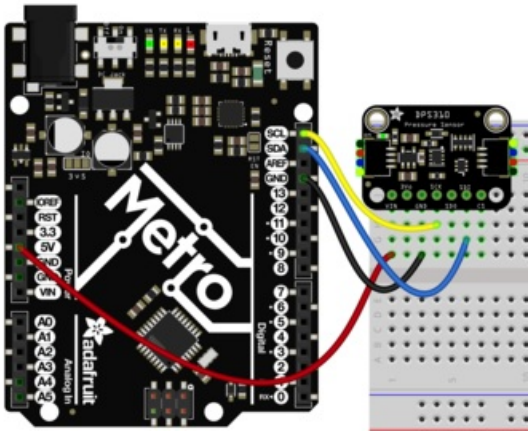
- **CS** - The **C**hip **S**elect pin, drop it low to start an SPI transaction. Its an input to the chip

If you want to connect multiple DPS310's to one microcontroller, have them share the **SDI**, **SDO** and **SCK** pins. Then assign each one a unique **CS** pin.

Arduino
I2C Wiring

Use this wiring if you want to connect via I2C interface

By default, the I2C address is **0x77**. If you add a jumper from D**DO** to **GND** the address will change to **0x76**

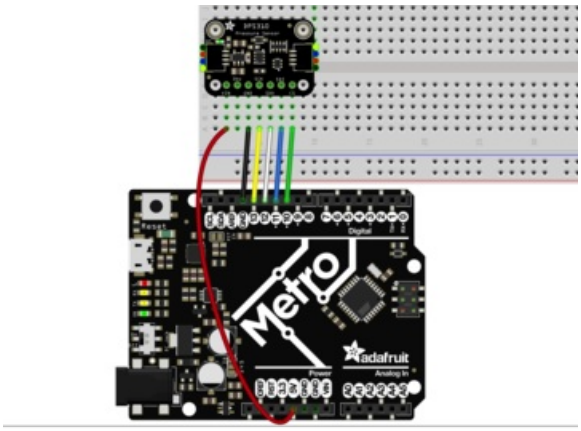
- 
- 

- Connect **board VIN (red wire)** to **Arduino 5V** if you are running a **5V** board Arduino (Uno, etc.). If your board is **3V,** connect to that instead.
- Connect **board GND (black wire)** to **Arduino GND**
- Connect **board SCL (yellow wire)** to **Arduino SCL**
- Connect **board SDA (blue wire)** to **Arduino SDA**

The final results should resemble the illustration above, showing an Adafruit Metro development board.

SPI Wiring

Since this is a SPI-capable sensor, we can use hardware or 'software' SPI. To make wiring identical on all microcontrollers, we'll begin with 'software' SPI. The following pins should be used:

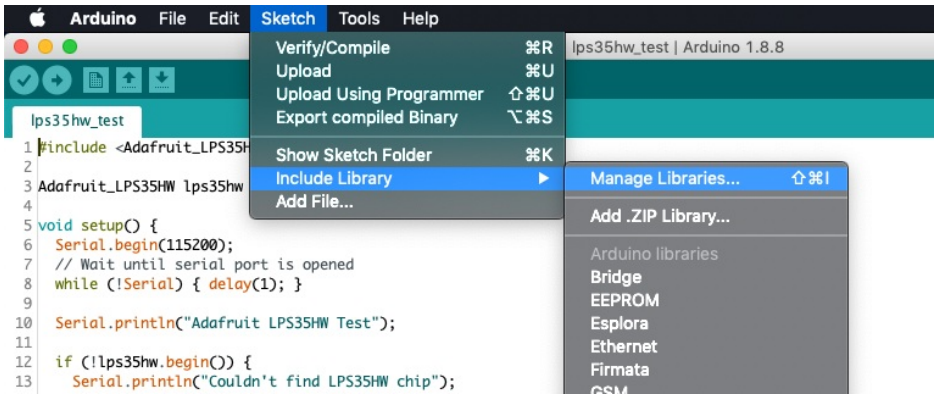


- Connect **Vin** to the power supply, 3V or 5V is fine. Use the same voltage that the microcontroller logic is based off of
- Connect **GND** to common power/data ground
- Connect the **SCK** pin to **Digital #13** but any pin can be used later
- Connect the **SDO** pin to **Digital #12** but any pin can be used later
- Connect the **SDI** pin to **Digital #11** but any pin can be used later
- Connect the **CS** pin **Digital #10** but any pin can be used later

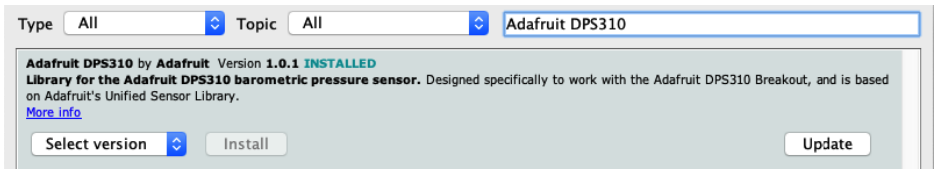
Later on, once we get it working, we can adjust the library to use hardware SPI if you desire, or change the pins to others.

Library Installation

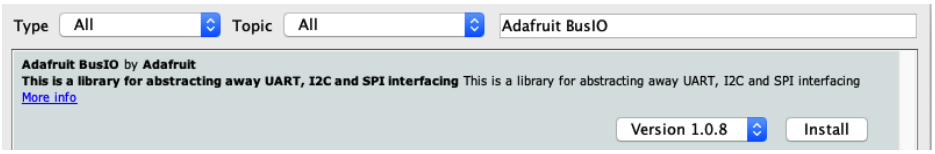
You can install the **Adafruit DPS310 Library** for Arduino using the Library Manager in the Arduino IDE.



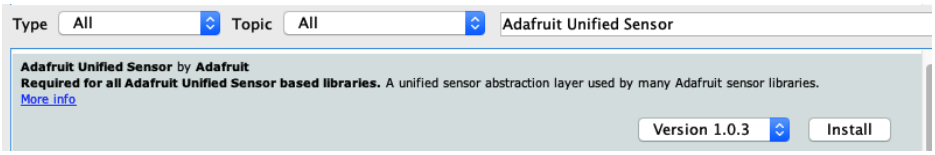
Click the **Manage Libraries ...** menu item, search for **Adafruit DPS310** and select the **Adafruit DPS310** library:



Then follow the same process for the **Adafruit BusIO** library.



Finally follow the same process for the **Adafruit Unified Sensor** library:



Load Example

Open up **File -> Examples -> Adafruit DPS310 -> dps310_simpletest** and upload to your Arduino wired up to the sensor.

Depending on whether you are using I2C or SPI, change the pin names and comment or uncomment the following lines.

```
if (! dps.begin_I2C()) {             // Can pass in I2C address here
  //if (! dps.begin_SPI(DPS310_CS)) {  // If you want to use SPI
```

Once you upload the code and open the Serial Monitor (**Tools->Serial Monitor**) at **115200** baud, you will see the current configuration printed, followed by the pressure, and temperature measurements. You should see something similar to this:

```
DPS310
DPS OK!
Temperature = 20.65 *C
Pressure = 1021.17 hPa

Temperature = 20.65 *C
Pressure = 1021.17 hPa
```

Carefully pressing on the small port on the top of the sensor will change the pressure and temperature readings. Give it a try and see how it works!

Example Code

```
// This example shows how to read temperature/pressure

#include <Adafruit_DPS310.h>

Adafruit_DPS310 dps;

// Can also use SPI!
#define DPS310_CS 10

void setup() {
  Serial.begin(115200);
  while (!Serial) delay(10);

  Serial.println("DPS310");
  if (! dps.begin_I2C()) {             // Can pass in I2C address here
  //if (! dps.begin_SPI(DPS310_CS)) {  // If you want to use SPI
    Serial.println("Failed to find DPS");
    while (1) yield();
  }
  Serial.println("DPS OK!");

  dps.configurePressure(DPS310_64HZ, DPS310_64SAMPLES);
  dps.configureTemperature(DPS310_64HZ, DPS310_64SAMPLES);
}

void loop() {
  sensors_event_t temp_event, pressure_event;

  while (!dps.temperatureAvailable() || !dps.pressureAvailable()) {
    return; // wait until there's something to read
  }

  dps.getEvents(&temp_event, &pressure_event);
  Serial.print(F("Temperature = "));
  Serial.print(temp_event.temperature);
  Serial.println(" *C");

  Serial.print(F("Pressure = "));
  Serial.print(pressure_event.pressure);
  Serial.println(" hPa");

  Serial.println();
}
```

Arduino Docs

Arduino Docs (https://adafru.it/ICC)

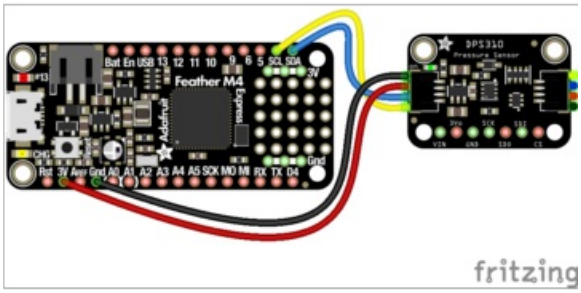
Python & CircuitPython

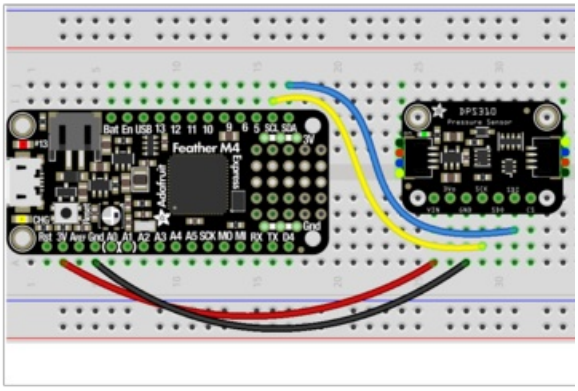
It's easy to use the DPS310 sensor with CircuitPython or Python and the Adafruit CircuitPython DPS310 (https://adafru.it/IFS) library. This library will allow you to easily write Python code that reads the barometric pressure, altitude and more from the sensor.

You can use this 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 a DSM310 breakout to your board exactly as shown below. Here's an example of wiring a Feather M4 to the sensor with I2C:

- 

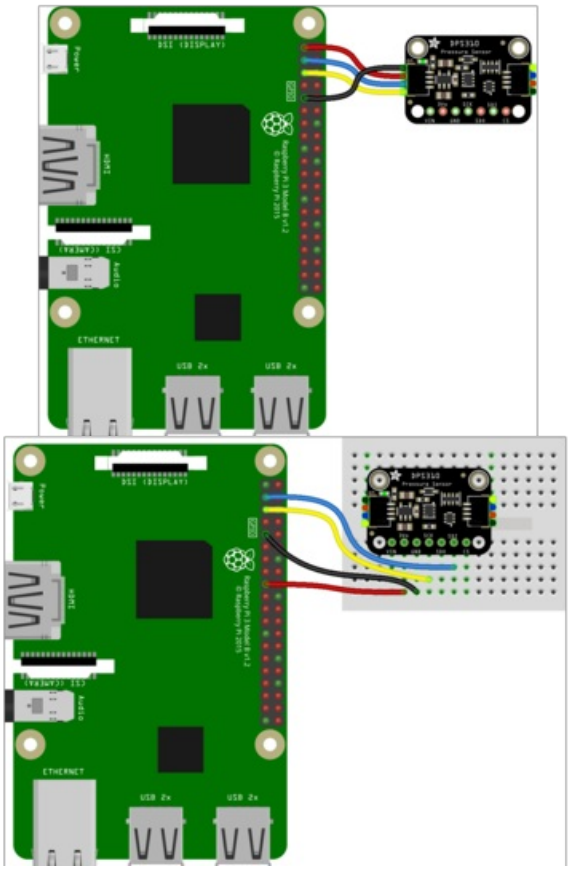
- 

- **Board 3V** to **sensor VIN (red wire)**
- **Board GND** to **sensor GND (black wire)**
- **Board SCL** to **sensor SCK/SCL (yellow wire)**
- **Board SDA** to **sensor SDI/SDA (blue wire)**

Python Computer Wiring

Since there's *dozens* of Linux computers/boards you can use, we will show 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 sensor using I2C:



- **Pi 3V** to **sensor VCC (red wire)**
- **Pi GND** to **sensor GND (black wire)**
- **Pi SCL** to **sensor SCK/SCL (yellow wire)**
- **Pi SDA** to **sensor SDI/SDA (blue wire)**

CircuitPython Installation of DPS310 Library

You'll need to install the Adafruit CircuitPython DPS310 (https://adafru.it/IFS) library on your CircuitPython board.

First make sure you are running the latest version of Adafruit CircuitPython (https://adafru.it/Amd) for your board.

Next you'll need to install the necessary libraries to use the hardware--carefully follow the steps to find and install these libraries from Adafruit's CircuitPython library bundle (https://adafru.it/ENC). Our CircuitPython starter guide has a great page on how to install the library bundle (https://adafru.it/ABU).

For non-express boards like the Trinket M0 or Gemma M0, you'll need to manually install the necessary libraries from the bundle:

- **adafruit_dps310.mpy**
- **adafruit_bus_device**
- **adafruit_register**

Before continuing make sure your board's **lib** folder has the **adafruit_dps310.mpy, adafruit_bus_device**, and **adafruit_register** files and folders copied over.

Next connect to the board's serial REPL (https://adafru.it/Awz)so you are at the CircuitPython `>>>` prompt

Python Installation of DPS310 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:

- `sudo pip3 install adafruit-circuitpython-dps310`

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

CircuitPython & Python Usage

To demonstrate the usage of the sensor we'll initialize it and read the barometric pressure measurements from the board's Python REPL.

Run the following code to import the necessary modules and initialize the I2C connection with the sensor:

```
import time
import board
import adafruit_dps310

i2c = board.I2C()

dps310 = adafruit_dps310.DPS310(i2c)
```

```
>>> import time
>>> import board
>>> import adafruit_dps310
>>> i2c = board.I2C()
>>> dps310 = adafruit_dps310.DPS310(i2c)
```

Now you're ready to read values from the sensor using these properties:

- **pressure -** The barometric pressure in hPa
- **temperature** - The temperature in degrees C

For example to print the pressure and temperature values:

```
print("Temperature = %.2f *C"%dps310.temperature)
print("Pressure = %.2f hPa"%dps310.pressure)
```

```
>>> print("Temperature = %.2f *C"%dps310.temperature)
Temperature = 18.58 *C
>>> print("Pressure = %.2f hPa"%dps310.pressure)
Pressure = 1021.20 hPa
```

For more details, check out the library documentation (https://adafru.it/IDK).

That's all there is to using the DPS310 Precision Barometric Pressure Sensor with CircuitPython!

Example Code

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

import time
import board
import adafruit_dps310

i2c = board.I2C()  # uses board.SCL and board.SDA
dps310 = adafruit_dps310.DPS310(i2c)

while True:
    print("Temperature = %.2f *C" % dps310.temperature)
    print("Pressure = %.2f hPa" % dps310.pressure)
    print("")
    time.sleep(1.0)
```

Python Docs

Python Docs (https://adafru.it/IDK)

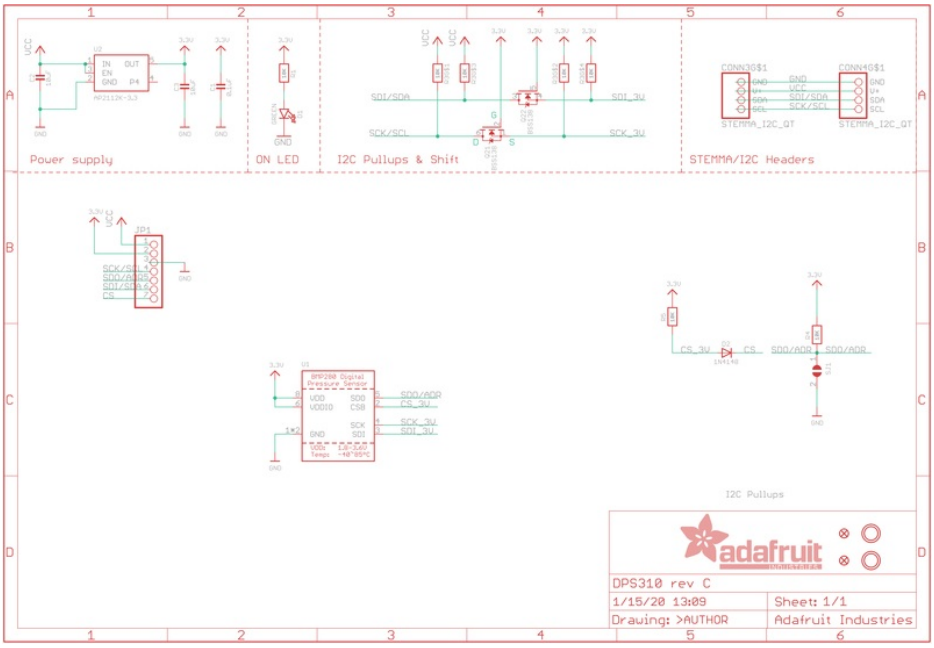
Downloads

Files

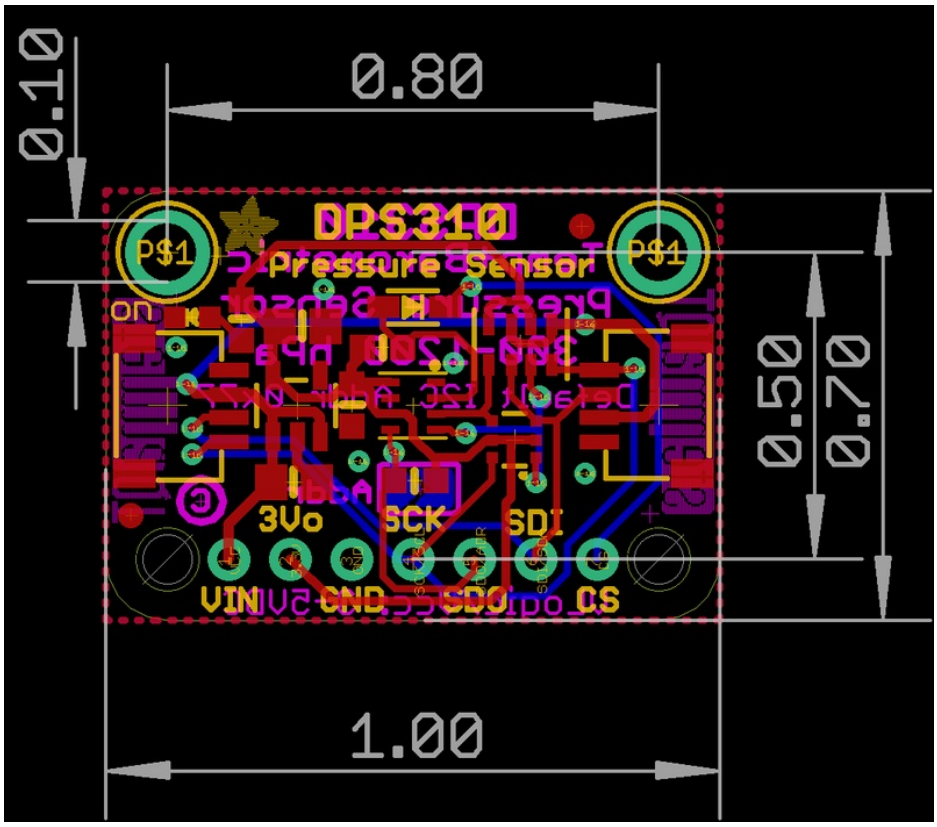
- DPS310 Datasheet (https://adafru.it/IFT)
- EagleCAD files on GitHub (https://adafru.it/IFU)
- Fritzing object in Adafruit Fritzing Library (https://adafru.it/IFV)

Schematic

Note: The schematic includes the BMP280 because they are the exact same pinouts and dimensions.



Fab Print


0.10
0.80
DPS310
Pressure Sensor
P$1
P$1
0.50
0.70
3Vo
SCK
SDI
VIN
GND
SDO
CS
1.00