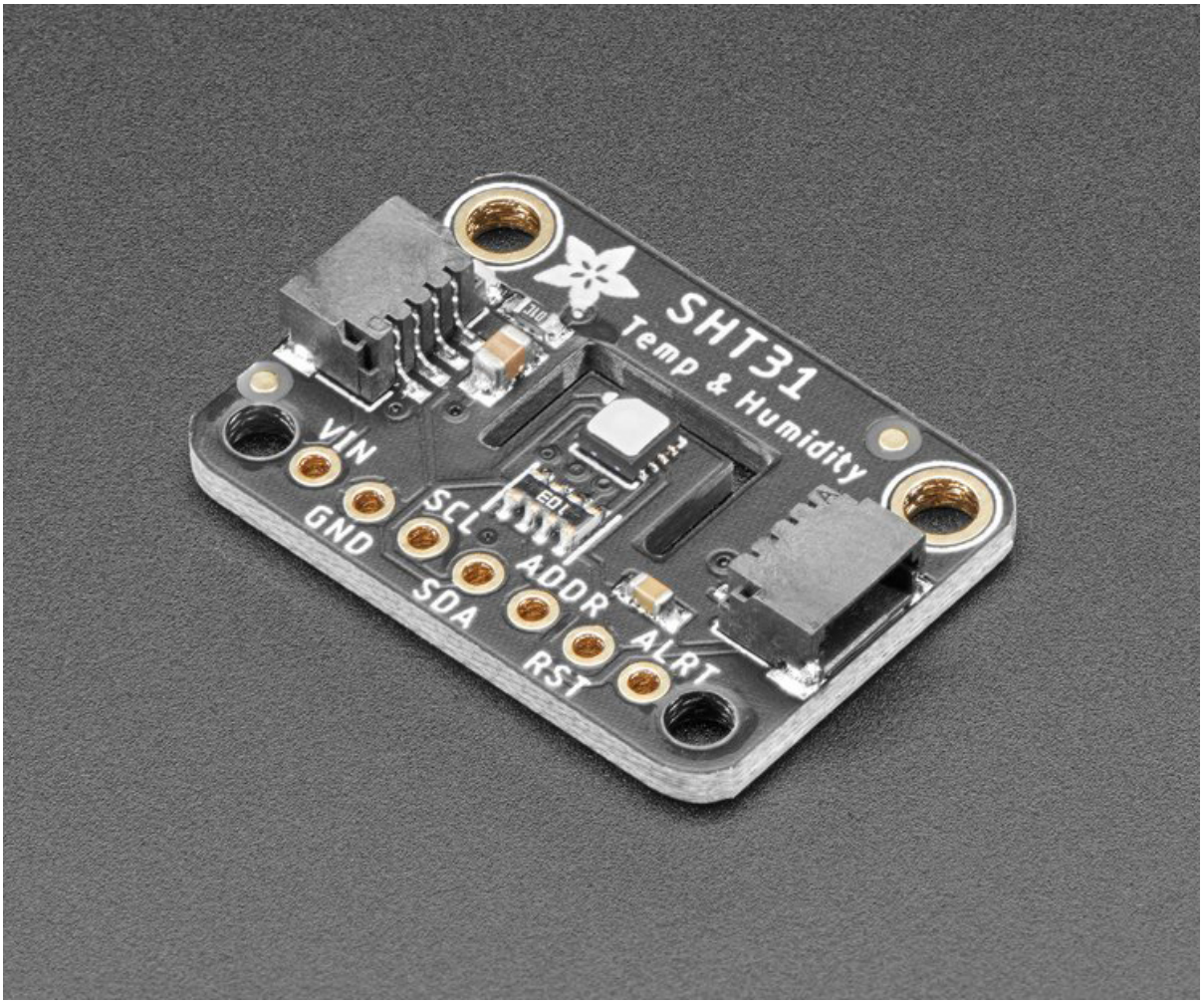
Adafruit SHT31-D Temperature & Humidity Sensor Breakout

Created by lady ada

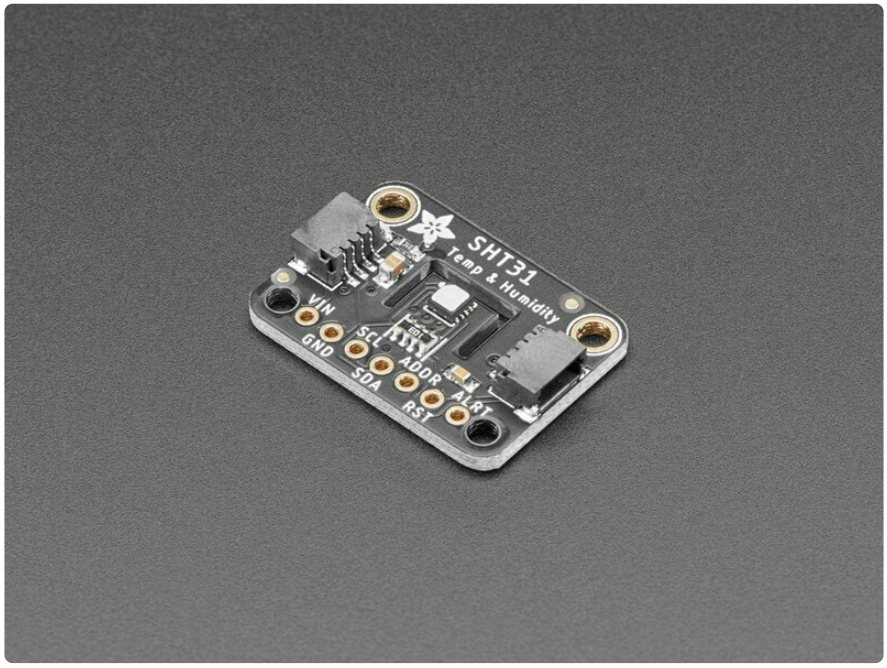


https://learn.adafruit.com/adafruit-sht31-d-temperature-and-humidity-sensor-breakout

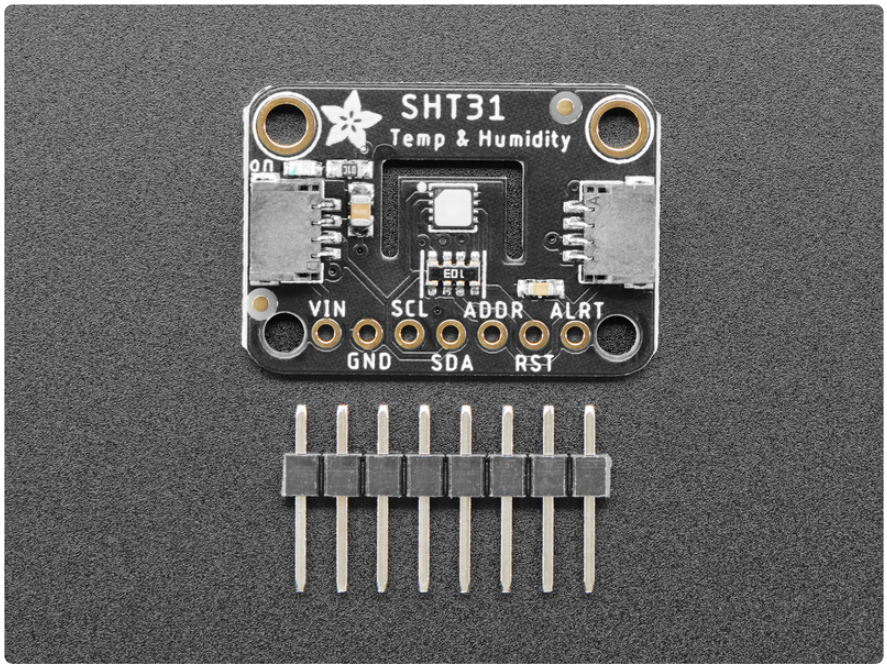
Last updated on 2026-08-20 03:29:28 PM UTC

Table of Contents

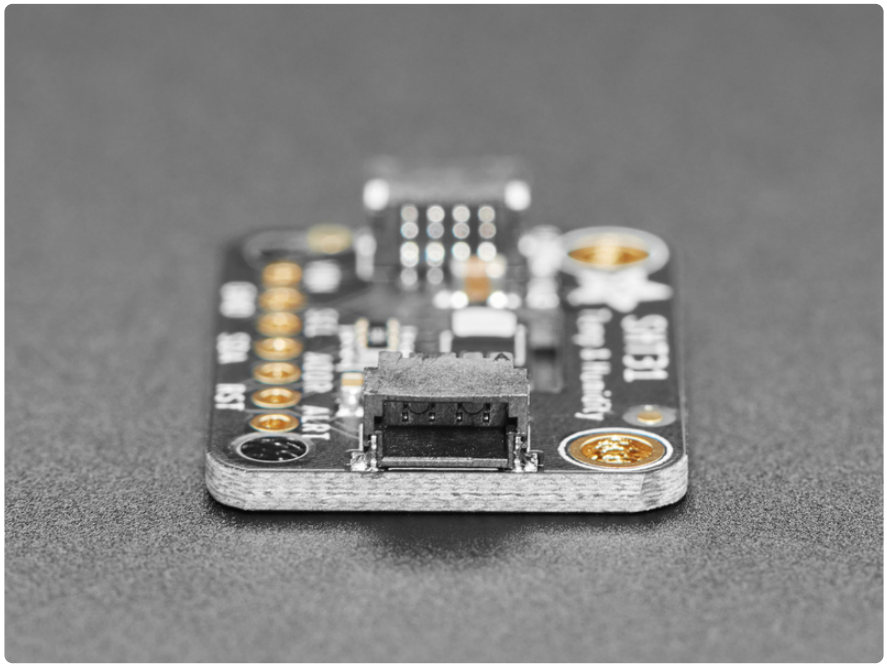
Overview



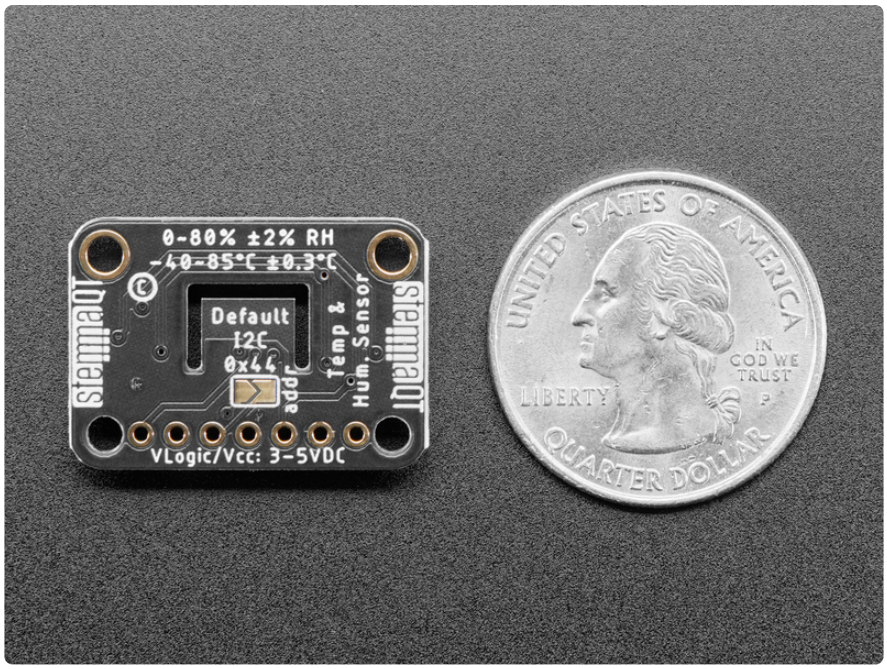
Sensirion Temperature/Humidity sensors are some of the finest & highest-accuracy devices you can get. And, finally we have some that have a true I2C interface for easy reading. The **SHT31-D** sensor has an excellent ±2% relative humidity and ±0.3°C accuracy for most uses.



Unlike earlier SHT sensors, this sensor has a true I2C interface, with two address options. It also is 3V or 5V compliant, so you can power and communicate with it using any microcontroller or microcomputer.

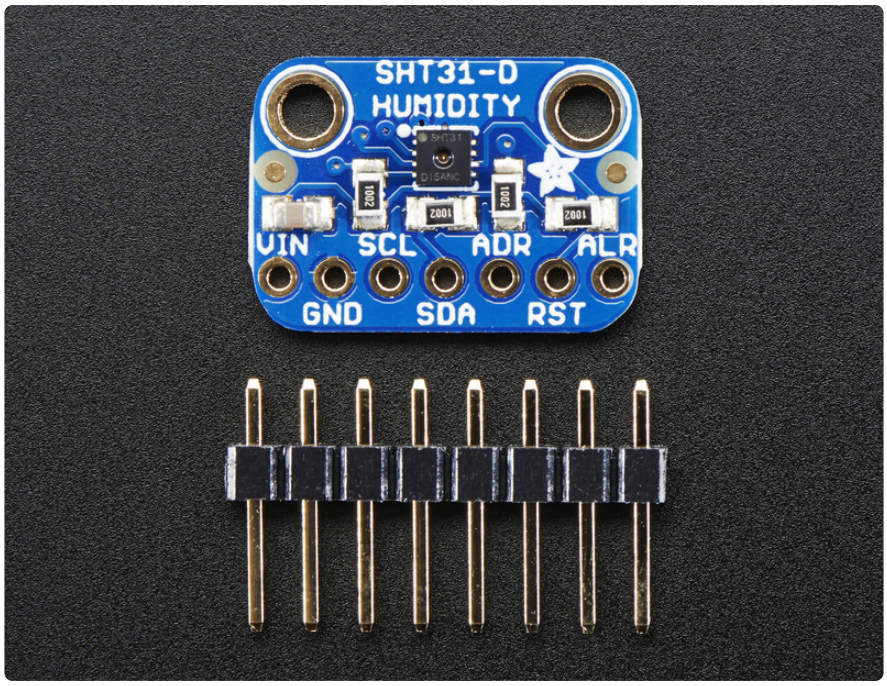


To get you going fast, we spun up a custom made PCB with the SHT31-D and some supporting circuitry such as pullup resistors and capacitors, in the **STEMMA QT** form factor (https://adafru.it/LBQ), making them easy to interface with. The STEMMA QT connectors (https://adafru.it/JqB) on either side are compatible with the SparkFun Qwiic (https://adafru.it/Fpw) I2C connectors. This allows you to make solderless connections between your development board and the SHT31-D or to chain them 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/17VE).



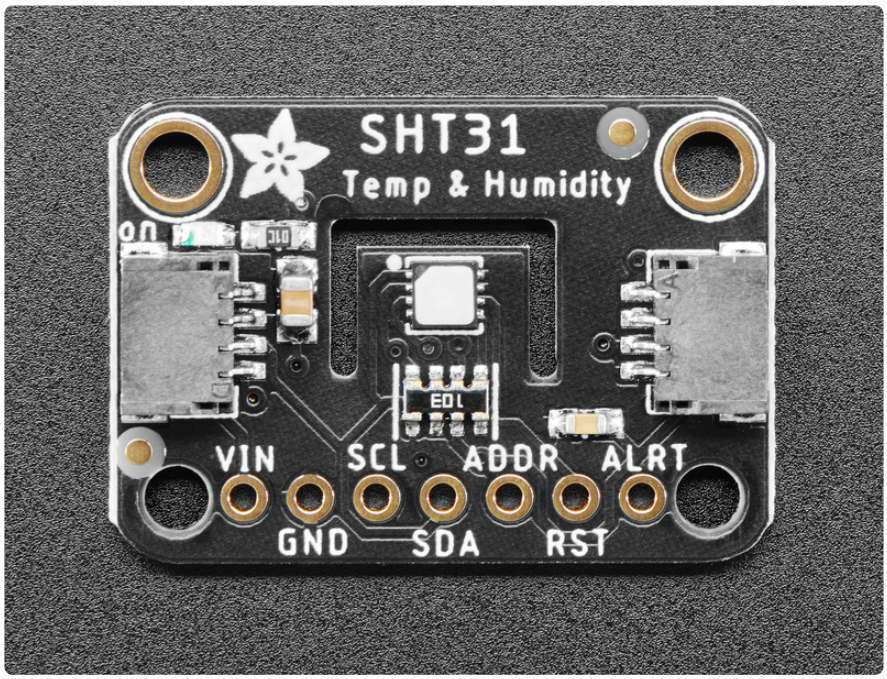
Each order comes with one fully assembled and tested PCB breakout and a small piece of header.

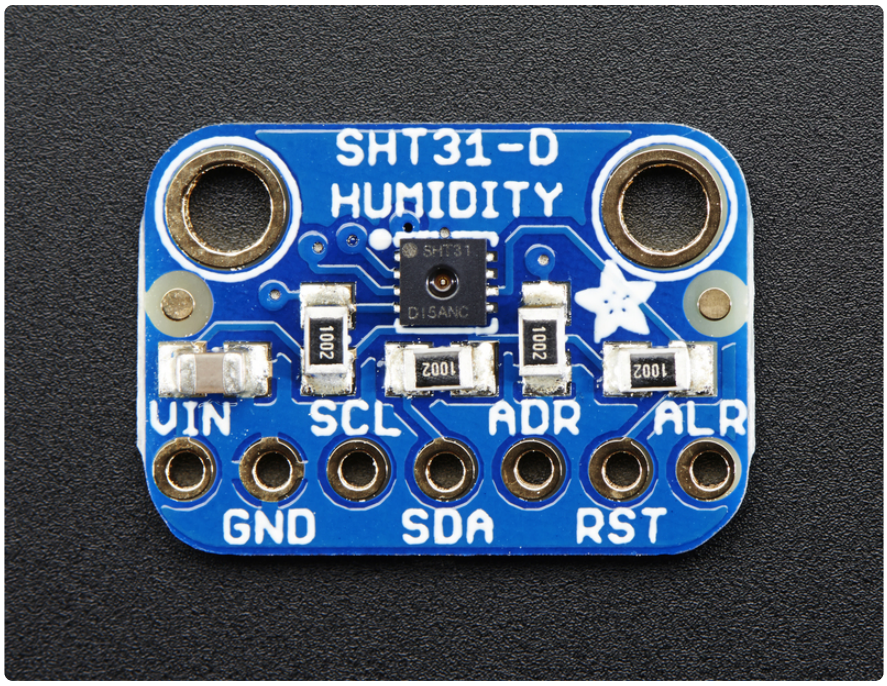
Text emphasized with a blue exclamation:	There are two versions of this board - the STEMMA QT version shown above, and the original header-only version shown below. Code works the same on both!



Pinouts

The SHT31-D is a I2C sensor. That means it uses the two I2C data/clock wires available on most microcontrollers, and can share those pins with other sensors as long as they don't have an address collision. For future reference, the default I2C address is **0x44** and you can also select address **0x45** by connecting the **ADDR** pin to a high voltage signal.





Power Pins:

- **Vin** - this is the power pin. The chip can use 2.5-5VDC for power. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V micro like Arduino, use 5V. For a 3.3V controller like a Raspbery Pi, connect to 3.3V
- **GND** - common ground for power and logic

I2C Logic pins:

- **SCL** - I2C clock pin, connect to your microcontrollers I2C clock line. This pin has a 10K pullup resistor to Vin
- **SDA** - I2C data pin, connect to your microcontrollers I2C data line. This pin has a 10K pullup resistor to Vin
- STEMMA QT **(https://adafru.it/Ft4)** - These connectors allow you to connect to development boards with **STEMMA QT** connectors, or to other things, with various associated accessories (https://adafru.it/Ft6).

Other Pins:

- **ADR** - This is the I2C address selection pin. This pin has a 10K pull down resistor to make the default I2C address **0x44**. You can tie this pin to Vin to make the address **0x45.**
- **RST** - Hardware reset pint. Has a 10K pullup on it to make the chip active by default. Connect to ground to do a hardware reset!

- **ALR** - Alert/Interrupt output. You can set up the sensor to alert you when an event has occurred. Check the datasheet for how you can set up the alerts

Changing the I2C Address

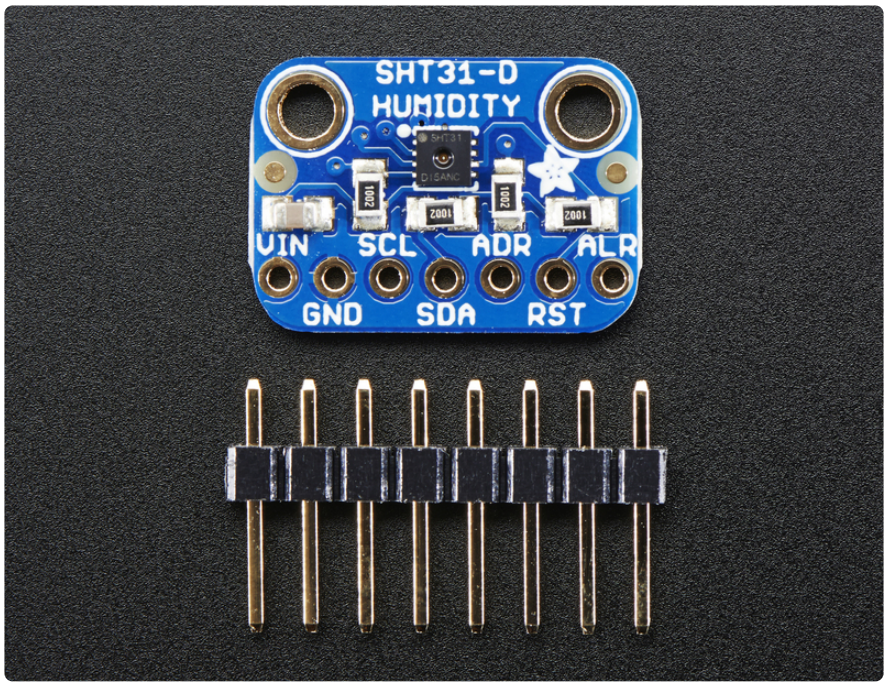
The default I2C address is 0x44. You can change a sensor to use 0x45 by tying the **ADR** pin to Vin.

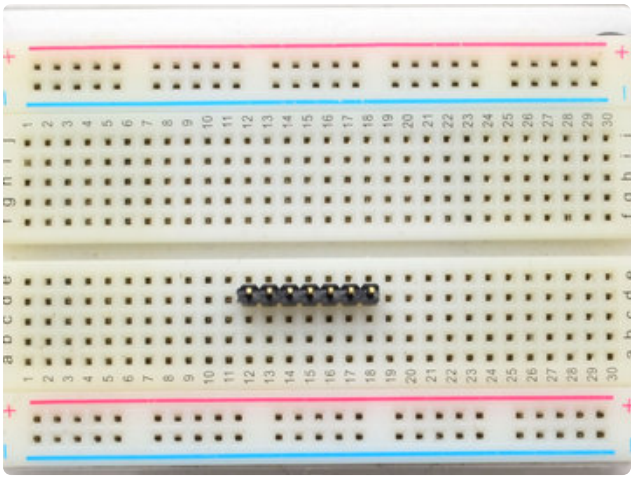
Thus two sensors can be used on one I2C bus with those addresses. If you need more sensors, look at the guide below.

Working with I2C Devices
By Carter Nelson
 Overview

https://learn.adafruit.com/working-with-i2c-devices/overview

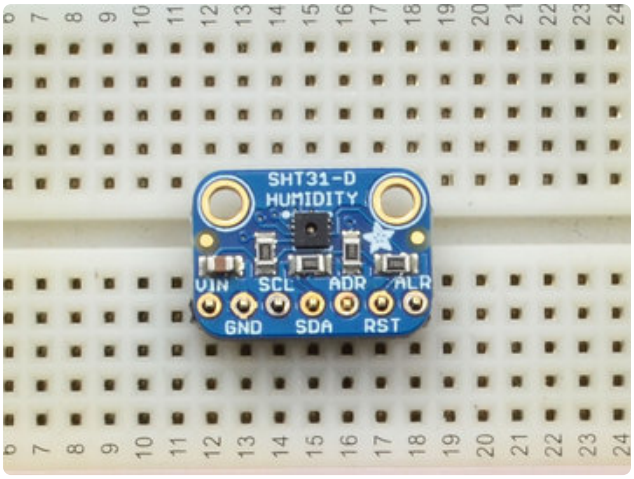
Assembly





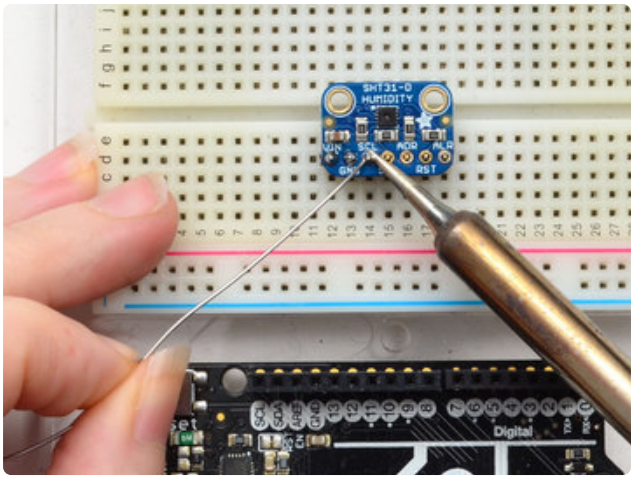
Prepare the header strip:

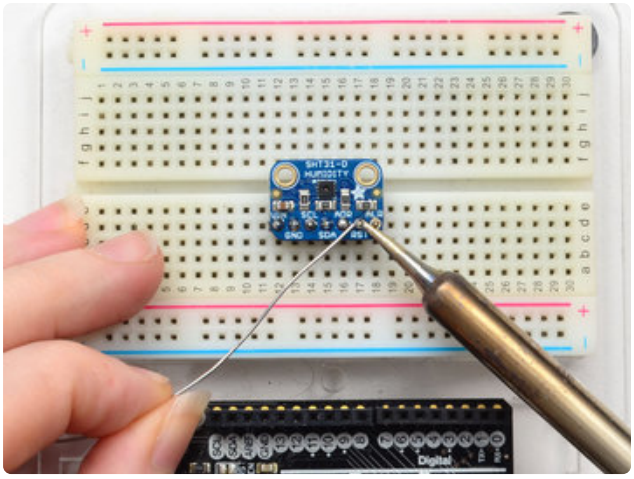
Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - **long pins down**



Add the breakout board:

Place the breakout board over the pins so that the short pins poke through the breakout pads

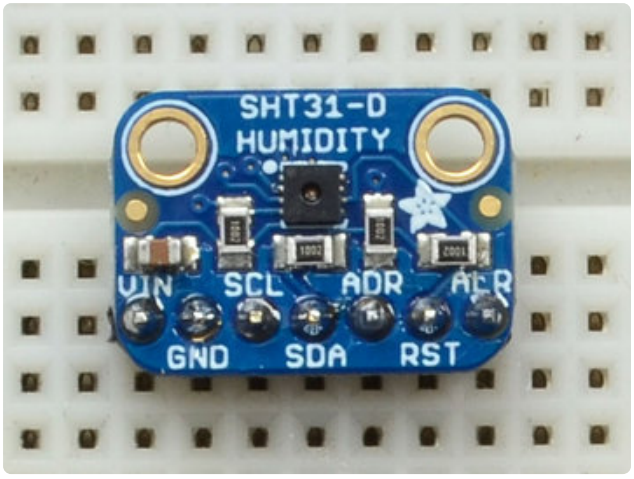




And Solder!

Be sure to solder all pins for reliable electrical contact.

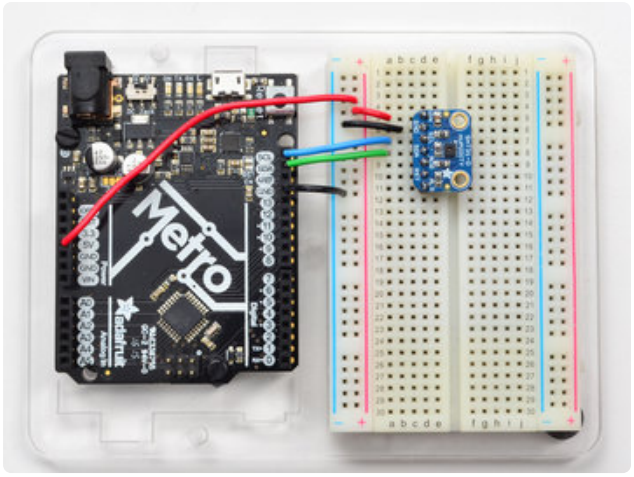
(For tips on soldering, be sure to check out our Guide to Excellent Soldering (https://adafru.it/aTk)).

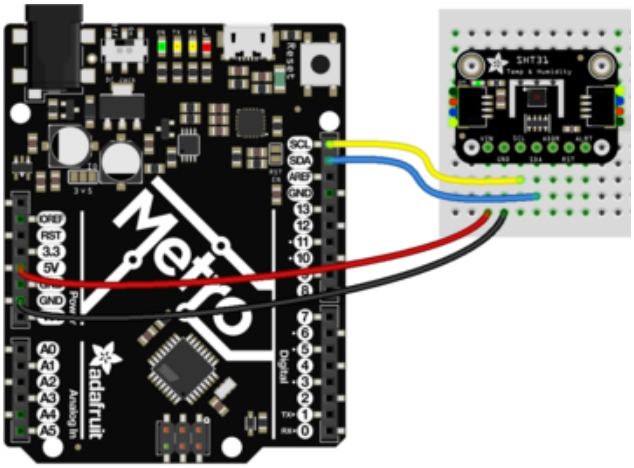


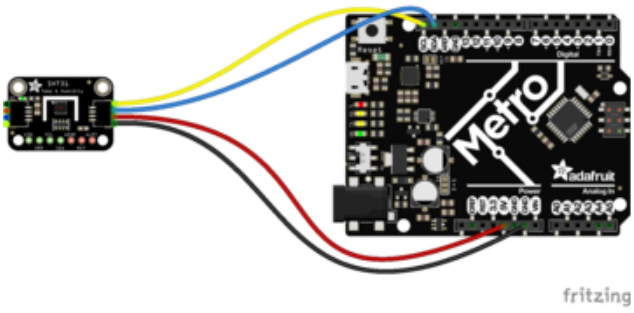
You're done! Check your solder joints visually and continue onto the next steps

Arduino Code

You can easily wire this breakout to any microcontroller, we'll be using an Arduino. For another kind of microcontroller, just make sure it has I2C, then port the code - its pretty simple stuff!







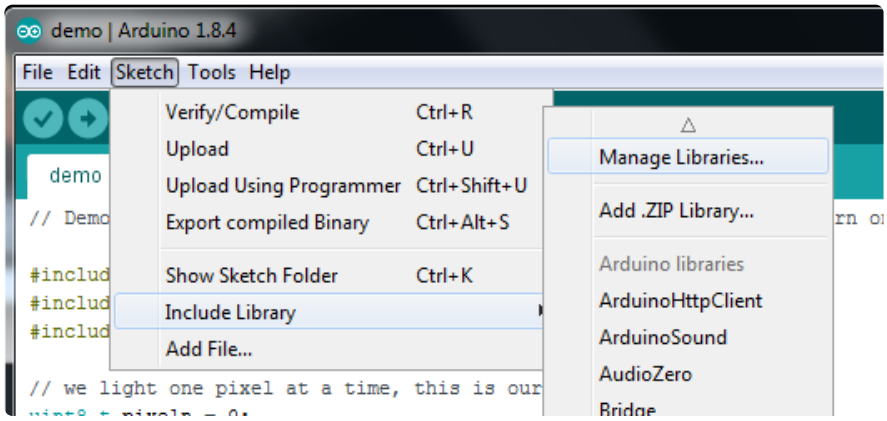
Connect **Vin** to the power supply, 3-5V is fine. **(red wire on STEMMA QT version)** Use the same voltage that the microcontroller logic is based off of. For most Arduinos, that is 5V
Connect **GND** to common power/data ground **(black wire on STEMMA QT version)**
Connect the **SCL** pin to the I2C clock **SCL** pin on your Arduino. **(yellow wire on STEMMA QT version)** On an UNO & '328 based Arduino, this is also known as **A5**, on a Mega it is also known as **digital 21** and on a Leonardo/Micro, **digital 3**
Connect the **SDA** pin to the I2C data **SDA** pin on your Arduino. **(blue wire on STEMMA QT version)** On an UNO & '328 based Arduino, this is also known as **A4**, on a Mega it is also known as **digital 20** and on a Leonardo/Micro, **digital 2**

The SHT31-D has a default I2C address of **0x44** which you can change to **0x45** by connecting the **ADR** pin to the **VIN** pin

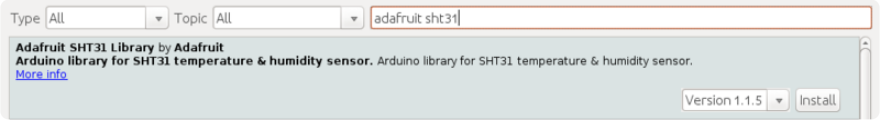
Download Adafruit_SHT31

To begin reading sensor data, you will need to download the **Adafruit SHT31** library from the Arduino library manager.

Open up the Arduino library manager:



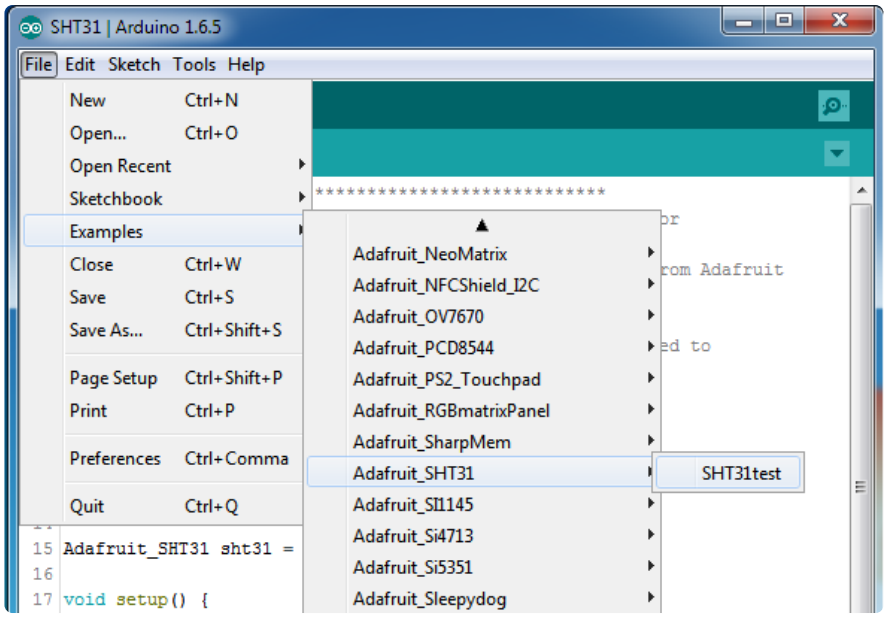
Search for the **Adafruit SHT31** library and install it



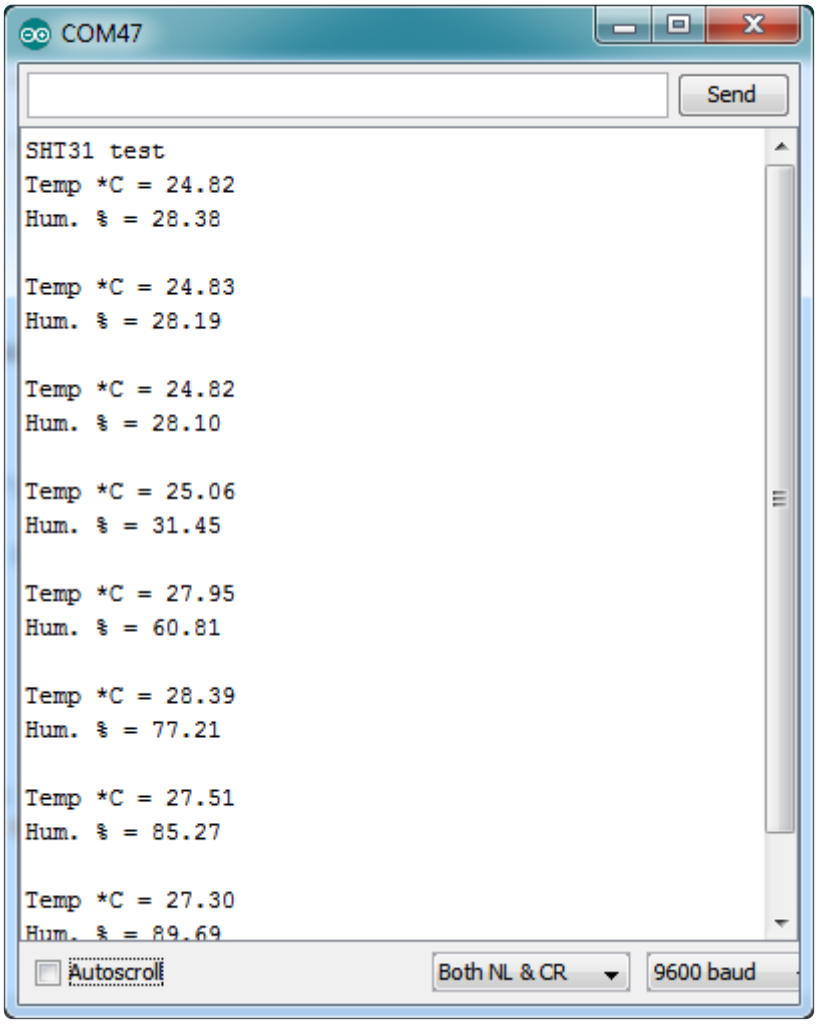
We also have a great tutorial on Arduino library installation at: http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use (https://adafru.it/aYM)

Load Demo

Open up **File->Examples->Adafruit_SHT31->SHT31test** and upload to your Arduino wired up to the sensor



Thats it! Now open up the serial terminal window at 9600 speed to begin the test.



You can try breathing on the sensor to increase the humidity. The sensor reacts very fast!

Library Reference

The library we have is simple and easy to use

You can create the **Adafruit_SHT31** object with:

```
Adafruit_SHT31 sht31 = Adafruit_SHT31();
```

There are no pins to set since you must use the I2C bus!

Then initialize the sensor with:

```
sht31.begin(0x44)
```

This function returns **True** if the sensor was found and responded correctly and **False** if it was not found

The **0x44** is the i2c address you have the sensor set up for. By default its 0x44, you can also adjust the sensor for **0x45** and then pass that value in

Once initialized, you can query the temperature in °C with

```
sht31.readTemperature()
```

Which will return floating point (decimal + fractional) temperature. You can convert to Fahrenheit by multiplying by 1.8 and adding 32 as you have learned in grade school!

Reading the humidity is equally simple. Call

```
sht31.readHumidity()
```

to read the humidity also as a floating point value between 0 and 100 (this reads % humidity)

We also have a few helper functions. Want to soft-reset the sensor? Use

```
sht31.reset()
```

There's also a heater built into the sensor, used to heat/evaporate any condensation. You can turn it on or off with

```
sht31.heater(true)
sht31.heater(false)
```

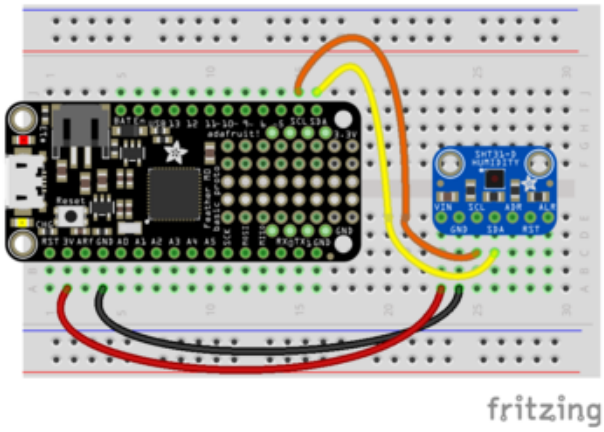
Python & CircuitPython

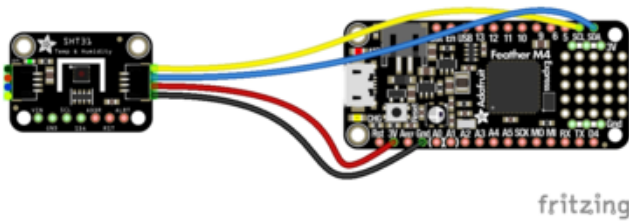
It's easy to use the SHT31-D sensor with Python and CircuitPython, and the Adafruit CircuitPython SHT31D (https://adafru.it/C1W) module. This module allows you to easily write Python code that reads the humidity and temperature from the sensor.

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).

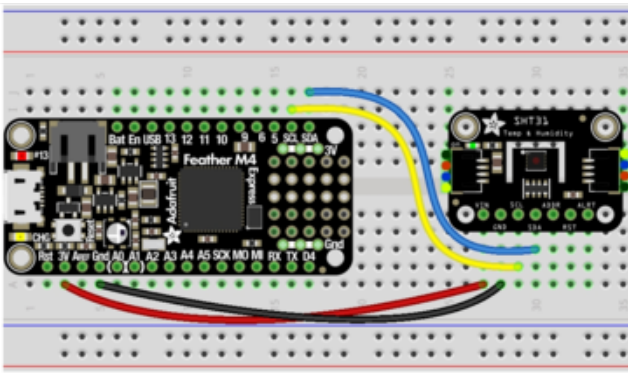
CircuitPython Microcontroller Wiring

First wire up a SHT31-D to your board exactly as shown on the previous pages for Arduino using an I2C connection. Here's an example of wiring a Feather M0 to the sensor with I2C:





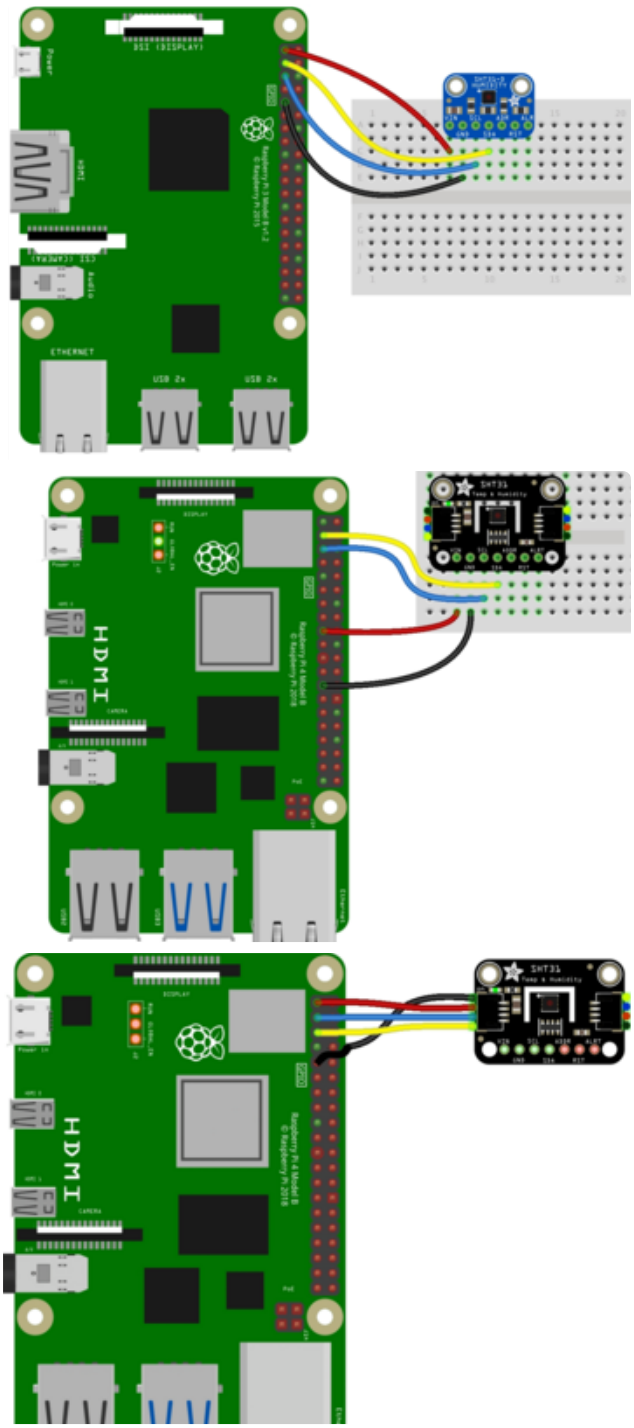
Board 3V to **sensor VIN (red wire on STEMMA QT version)**
Board GND to **sensor GND (black wire on STEMMA QT version)**
Board SCL to **sensor SCL (yellow wire on STEMMA QT version)**
Board SDA to **sensor SDA (blue wire on STEMMA QT version)**



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) (https://adafru.it/BSN).

Here's the Raspberry Pi wired with I2C:

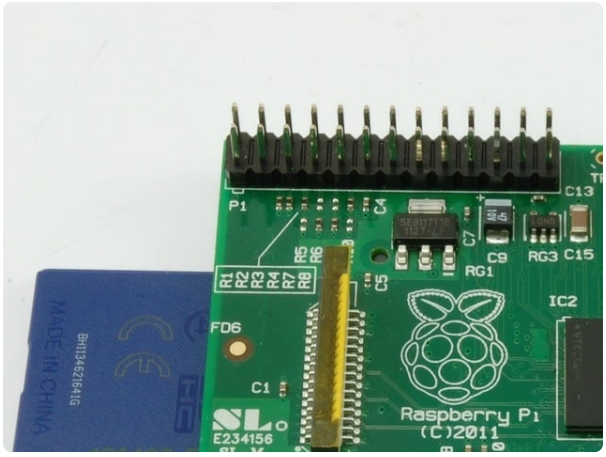


Pi 3V3 to **sensor VIN (red wire on STEMMA QT version)**
Pi GND to **sensor GND (black wire on STEMMA QT version)**
Pi SCL to **sensor SCL (yellow wire on STEMMA QT version)**
Pi SDA to **sensor SDA (blue wire on STEMMA QT version)**

Enabling I2C on Raspberry Pi

The I2C pins may not be enabled by default on thge Raspberry Pi. No worries, there are simple instructions to do this using `raspi-config`.

Click the button below for a link to the page describing how to do this.



Adafruit's Raspberry Pi Lesson 4. GPIO Setup
By Simon Monk
 Configuring I2C

https://learn.adafruit.com/adafruits-raspberry-pi-lesson-4-gpio-setup/configuring-i2c

Enabling I2C on Other Single Board Computers

Please refer to the manufacturer's documentation on enabling I2C, as different boards seem to have different instructions on how to do so.

CircuitPython Installation of SHT31D Library

Next you'll need to install the Adafruit CircuitPython SHT31D (https://adafru.it/C1W) 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/zdx). Our introduction guide has a great page on how to install the library bundle (https://adafru.it/ABU) for both express and non-express boards.

Remember for non-express boards like the, you'll need to manually install the necessary libraries from the bundle:

- **adafruit_sht31d.mpy**
- **adafruit_bus_device**

Before continuing make sure your board's lib folder or root filesystem has the **adafruit_sht31d.mpy,** and **adafruit_bus_device** 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 SHT31D 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-sht31d`

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 and Python Usage

To demonstrate the usage of the sensor we'll initialize it and read the humidity and temperature from the board's Python REPL.

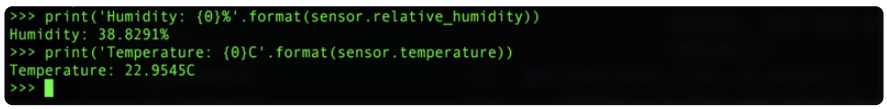
Run the following code to import the necessary modules and initialize the I2C connection with the sensor:

```
import board
import busio
import adafruit_sht31d
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_sht31d.SHT31D(i2c)
```

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

- **relative_humidity** - The relative humidity measured by the sensor, this is a value from 0-100%.
- **temperature** - The temperature measured by the sensor, a value in degrees Celsius.

```
print('Humidity: {0}%'.format(sensor.relative_humidity))
print('Temperature: {0}C'.format(sensor.temperature))
```



That's all there is to using the SHT31D with Python and CircuitPython!

Below is a complete example that measures the sensor readings and prints them every two seconds. Save this as **code.py** on your board and open the REPL to see the output.

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

import time

import board

import adafruit_sht31d

# Create sensor object, communicating over the board's default I2C bus
i2c = board.I2C()  # uses board.SCL and board.SDA
# i2c = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a microcontroller
sensor = adafruit_sht31d.SHT31D(i2c)

loopcount = 0
while True:
    print(f"\nTemperature: {sensor.temperature:0.1f} C")
    print(f"Humidity: {sensor.relative_humidity:0.1f} %")
    loopcount += 1
    time.sleep(2)
    # every 10 passes turn on the heater for 1 second
    if loopcount == 10:
        loopcount = 0
        sensor.heater = True
        print("Sensor Heater status =", sensor.heater)
        time.sleep(1)
        sensor.heater = False
        print("Sensor Heater status =", sensor.heater)
```

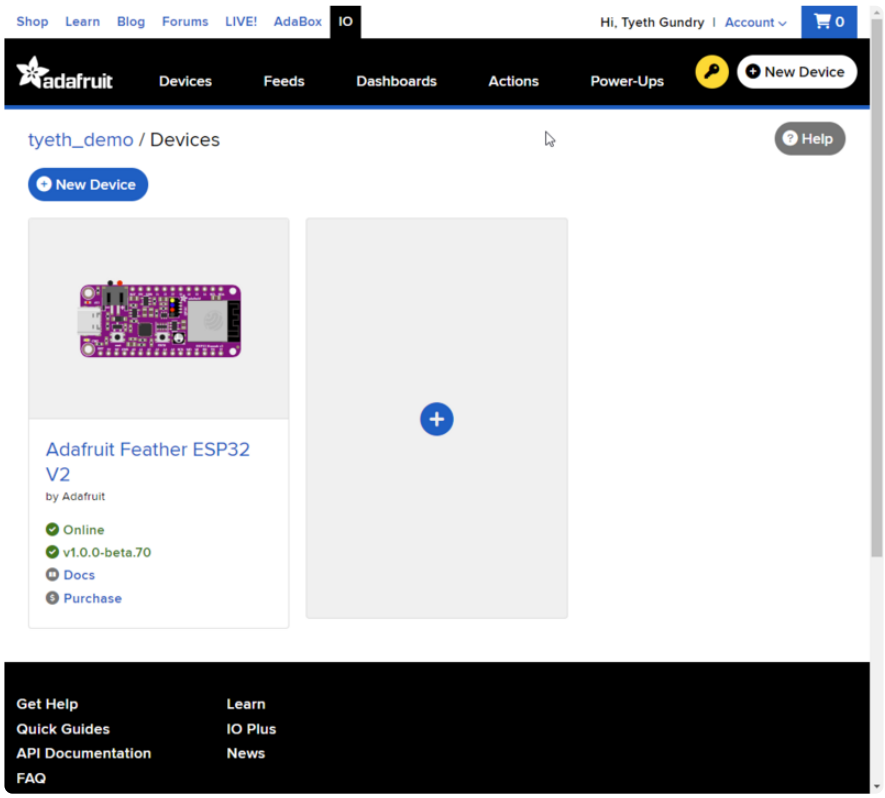
https://github.com/adafruit/Adafruit_CircuitPython_SHT31D/blob/main/examples/sht31d_simpletest.py

Text emphasized with a blue exclamation:	A user noted that for a Raspberry Pi Pico, which has multiple I2C interfaces, might better use i2c = busio.I2C(scl=board.GP1, sda=board.GP0). If having Pico issues, try this.

Python Docs

Python Docs (https://adafru.it/C3I)

WipperSnapper



What is WipperSnapper

WipperSnapper is a firmware designed to turn any WiFi-capable board into an Internet-of-Things device without programming a single line of code. WipperSnapper connects to Adafruit IO (https://adafru.it/fsU), a web platform designed (by Adafruit! (https://adafru.it/Bo5)) to display, respond, and interact with your project's data.

Simply load the WipperSnapper firmware onto your board, add credentials, and plug it into power. Your board will automatically register itself with your Adafruit IO account.

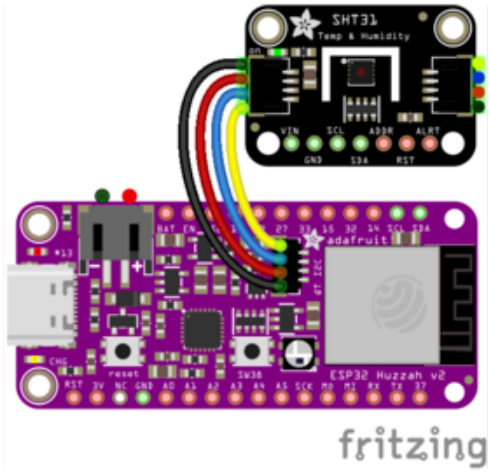
From there, you can add components to your board such as buttons, switches, potentiometers, sensors, and more! Components are dynamically added to hardware, so you can immediately start interacting, logging, and streaming the data your projects produce without writing code.

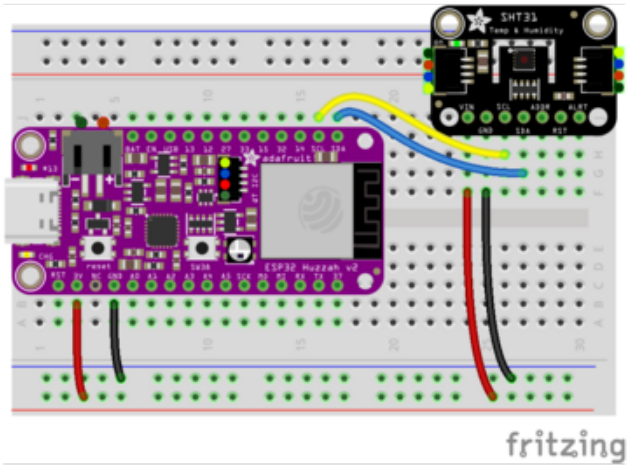
If you've never used WipperSnapper, click below to read through the quick start guide before continuing.

https://adafru.it/Vfd

Wiring

First, wire up an SHT31-D or SHT30 (**SHT3x**) to your board exactly as follows. Here is an example of the SHT31-D wired to an Adafruit ESP32 Feather V2 (http://adafru.it/5400) using I2C with a STEMMA QT cable (no soldering required) (http://adafru.it/4210)



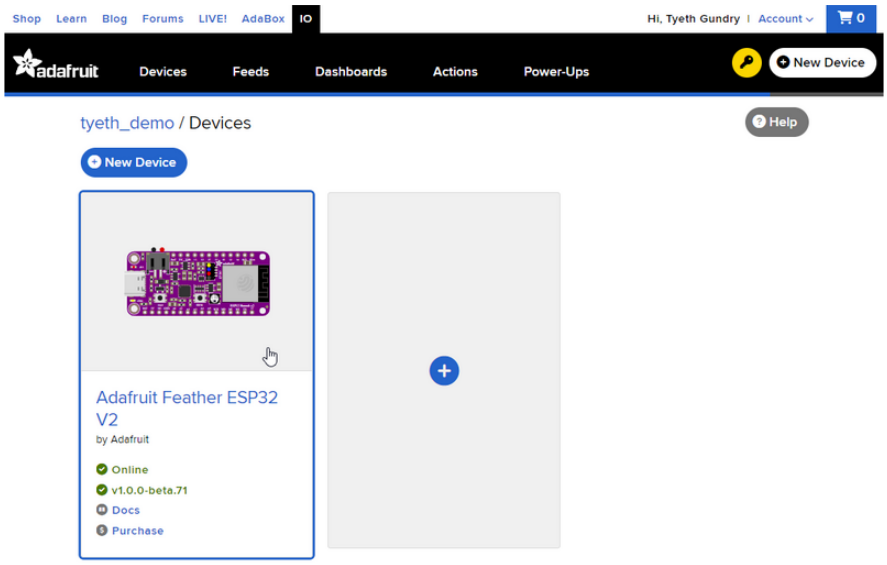


Board 3V to **sensor VIN (red wire on STEMMA QT)**
Board GND to **sensor GND (black wire on STEMMA QT)**
Board SCL to **sensor SCL (yellow wire on STEMMA QT)**
Board SDA to **sensor SDA (blue wire on STEMMA QT)**

Usage

Connect your board to Adafruit IO Wippersnapper and navigate to the WipperSnapper board list **(https://adafru.it/TAu)**.

On this page, **select the WipperSnapper board you're using** to be brought to the board's interface page.



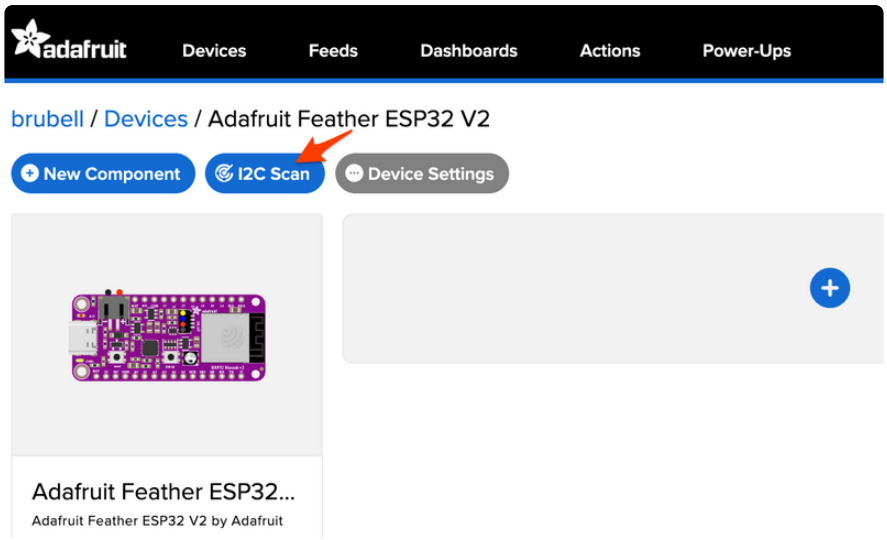
If you do not see your board listed here - you need [to connect your board to Adafruit IO (https://adafru.it/Vfd)](https://adafru.it/Vfd) first.

On the device page, quickly **check that you're running the latest version of the WipperSnapper firmware**.

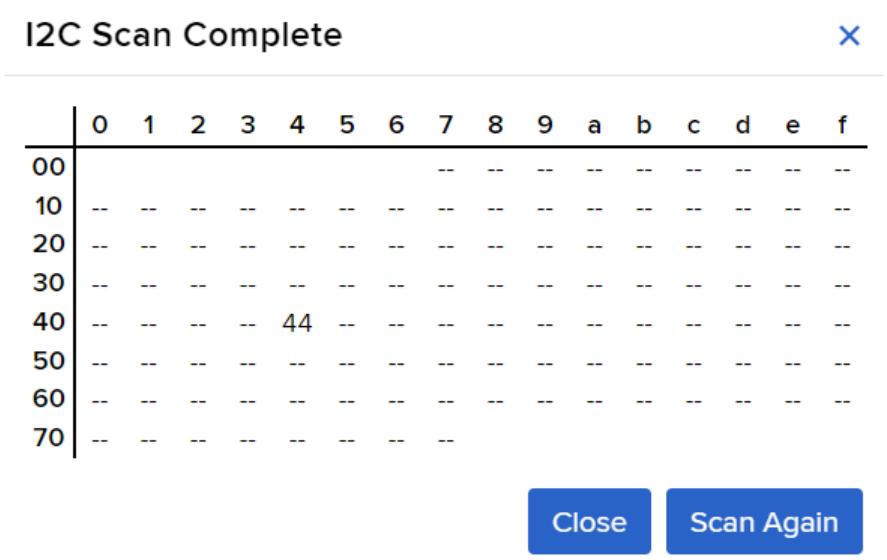
The device tile on the left indicates the version number of the firmware running on the connected board.

If the firmware version is green with a checkmark - continue with this guide.
If the firmware version is red with an exclamation mark "!" - [update to the latest WipperSnapper firmware (https://adafru.it/Vfd)](https://adafru.it/Vfd) on your board before continuing.

Next, make sure the sensor is plugged into your board and click the **I2C Scan** button.



You should see the SHT3x's default I2C address of `0x44` pop-up in the I2C scan list.



I don't see the sensor's I2C address listed!

First, double-check the connection and/or wiring between the sensor and the board.

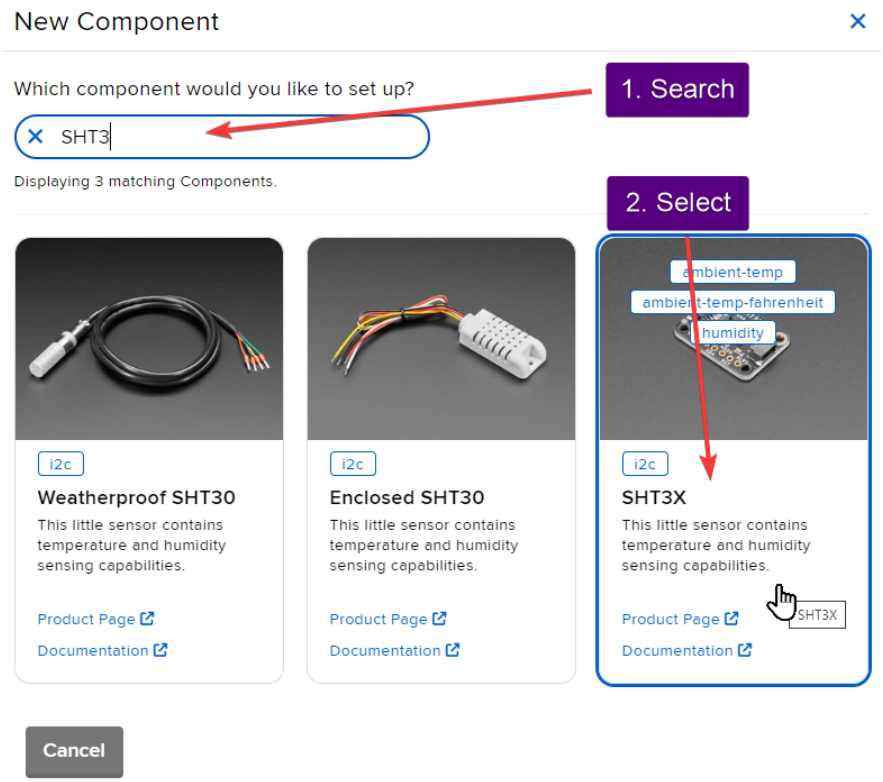
Then, reset the board and let it re-connect to Adafruit IO WipperSnapper.

With the sensor detected in an I2C scan, you're ready to add the sensor to your board.

Click the New Component button or the + button to bring up the component picker.

Adafruit IO supports a large amount of components. To quickly find your sensor, type `SHT3` into the search bar, then select the matching **SHT30/SHT3x** component.

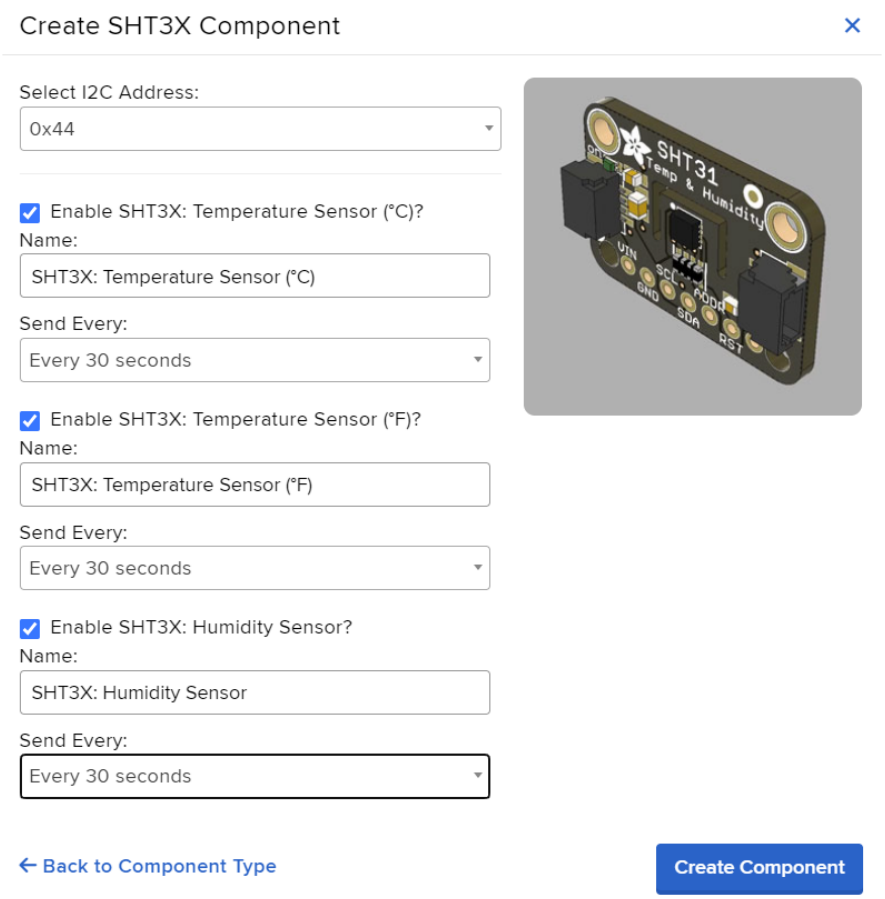
For the SHT31-D select the **SHT3x** (which supports all of the Sensirion SHT3x series, i.e. SHT30/31/35) from the component picker.



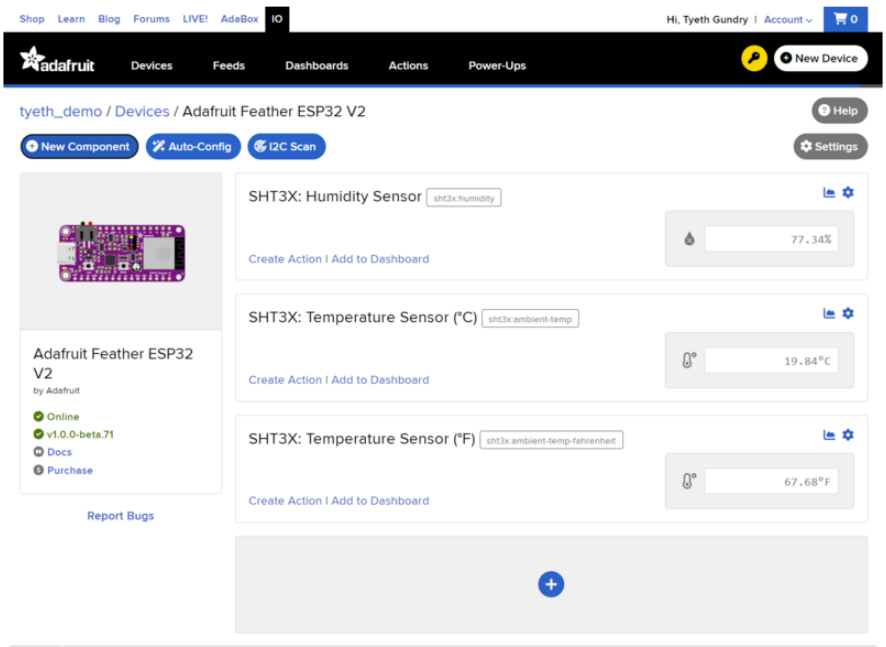
On the component configuration page, the SHT3x's sensor address should be listed along with the sensor's settings.

The **Send Every** option is specific to each sensor's measurements. This option will tell the Feather how often it should read from the SHT3x sensor and send the data to Adafruit IO. Measurements can range from every 30 seconds to every 24 hours.

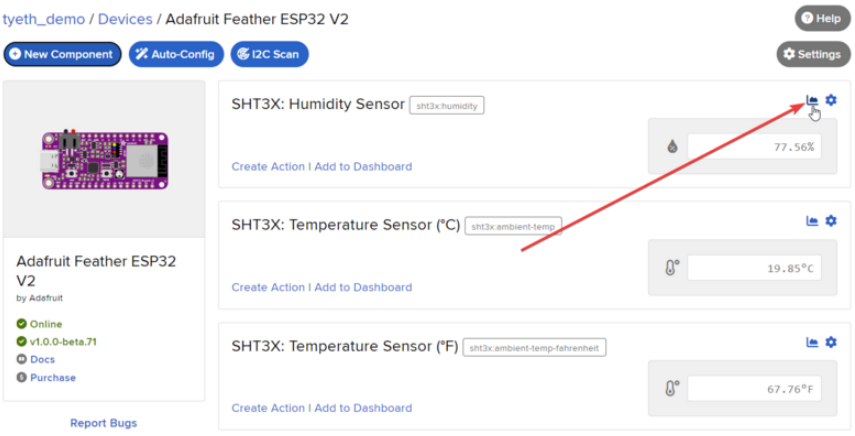
For this example, set the **Send Every** interval to every 30 seconds.



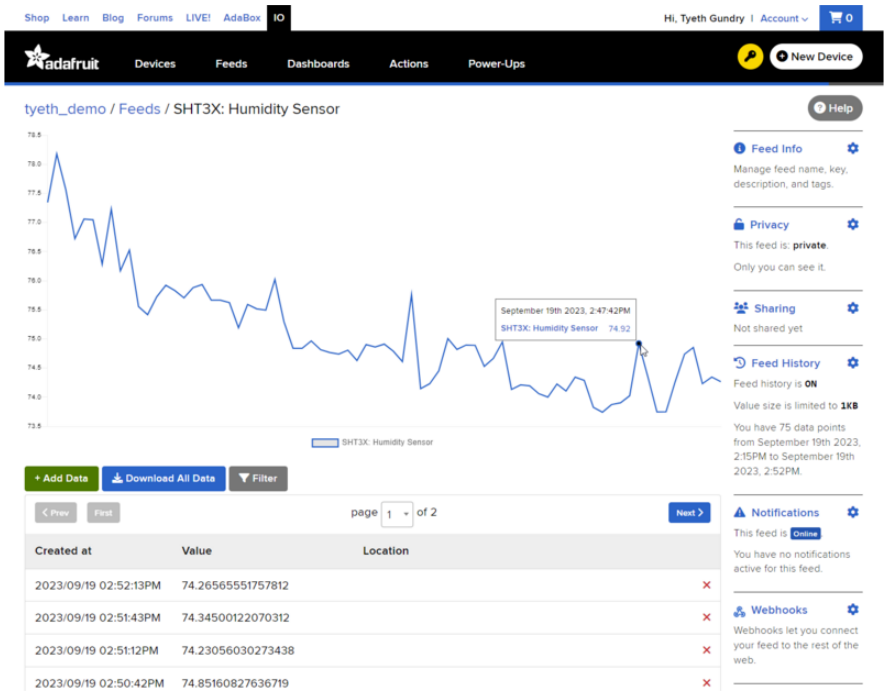
Your device interface should now show the sensor components you created. After the interval you configured elapses, WipperSnapper will automatically read values from the sensor(s) and send them to Adafruit IO.



To view the data that has been logged from the sensor, click on the graph next to the sensor name.



Here you can see the feed history and edit things about the feed such as the name, privacy, webhooks associated with the feed and more. If you want to learn more about how feeds work, [check out this page](https://adafru.it/10aZ).



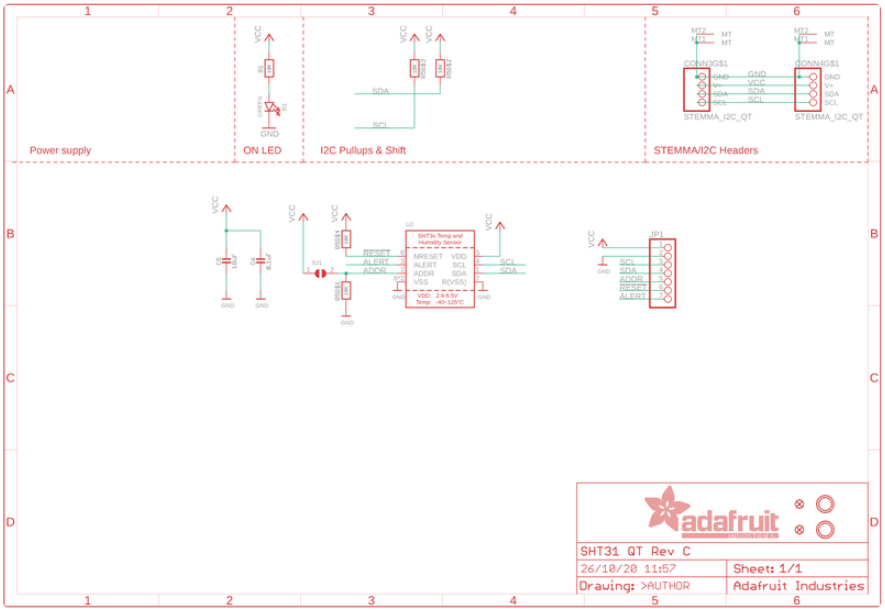
Downloads

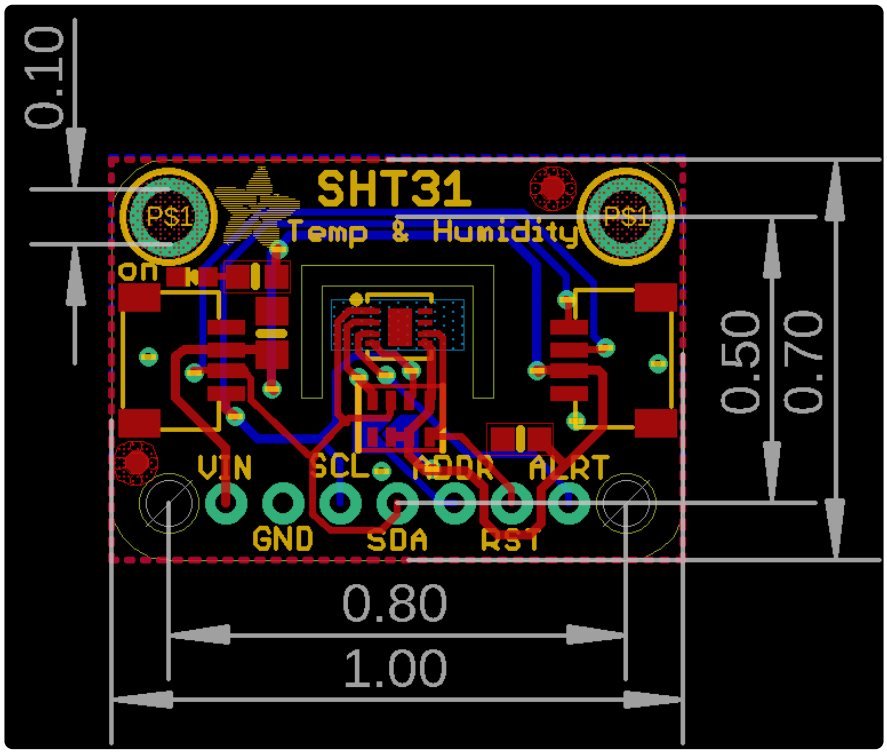
Datasheets & Files

- [SHT31-DIS datasheet](https://adafru.it/k6a)

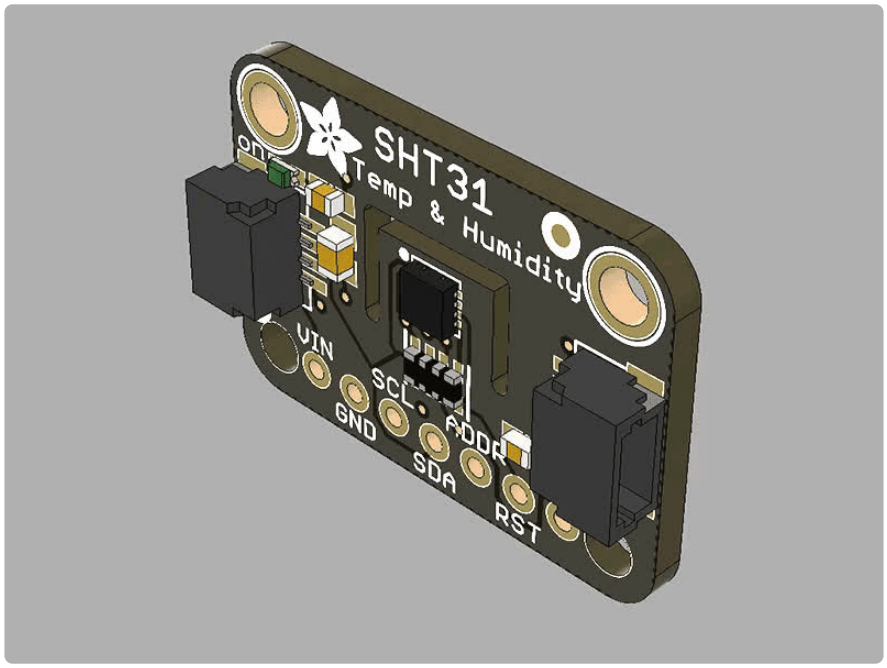
- EagleCAD PCB Files on GitHub (https://adafru.it/owF)
- 3D models on GitHub (https://adafru.it/18cP)
- Fritzing object available in the Adafruit Fritzing Library (https://adafru.it/REo)

Schematic and Fab Print - STEMMA QT Version



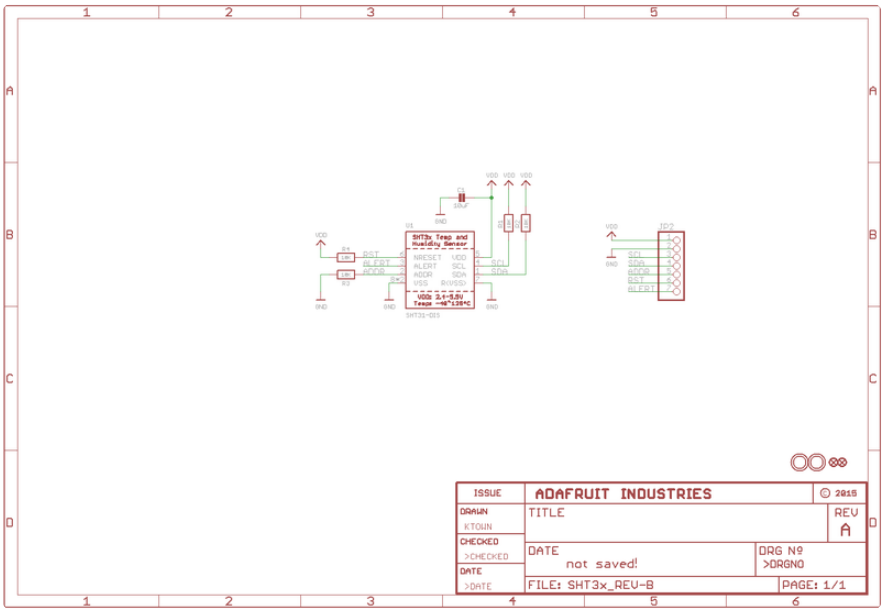


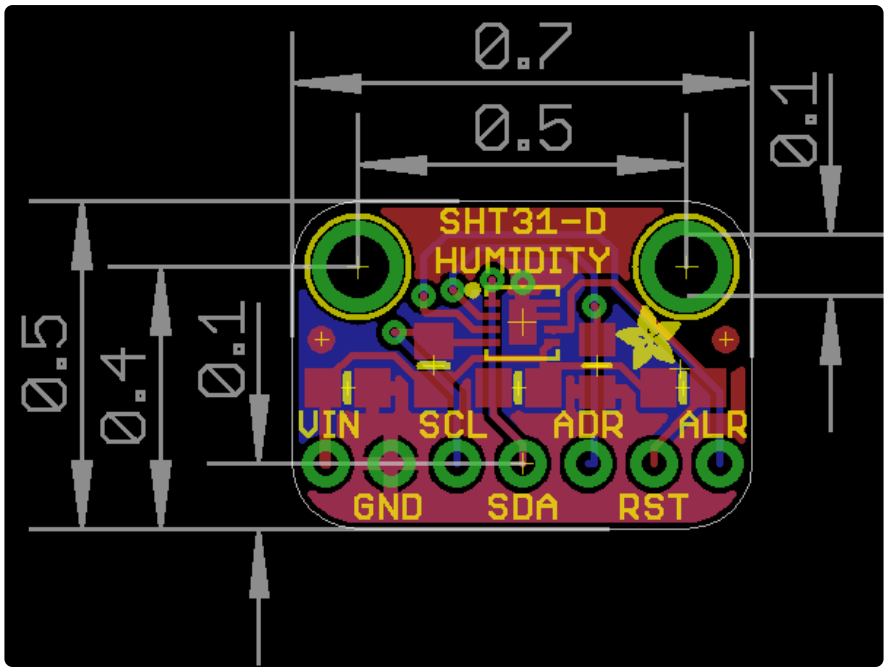
3D Model



Schematic and Fab Print - Original version

Click to enlarge




0.7
0.5
0.1
0.5
0.4
0.1
SHT31-D
HUMIDITY
VIN
SCL
ADR
ALR
GND
SDA
RST