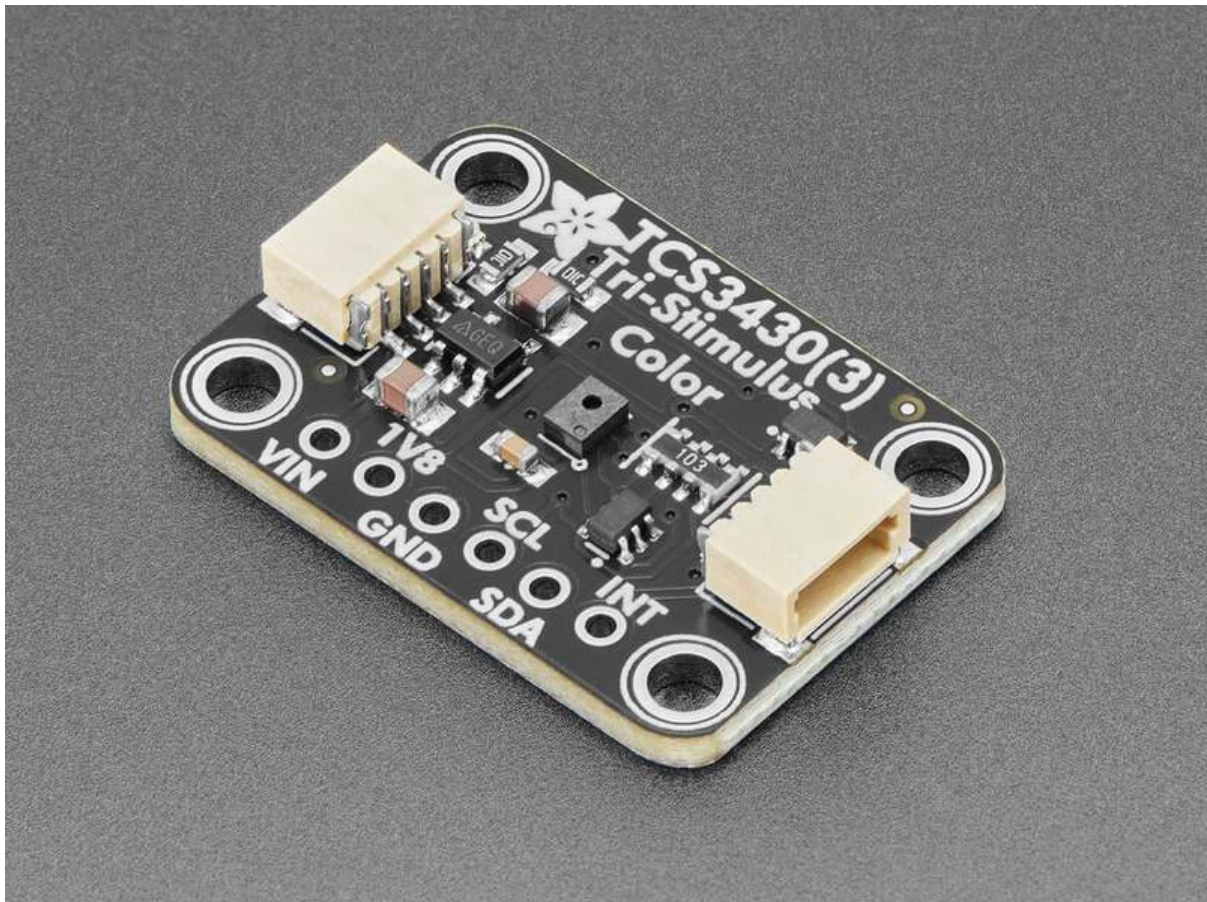




Adafruit TCS3430 / TCS34303 Ambient Tri-Stimulus Color Sensor

Created by Tim C



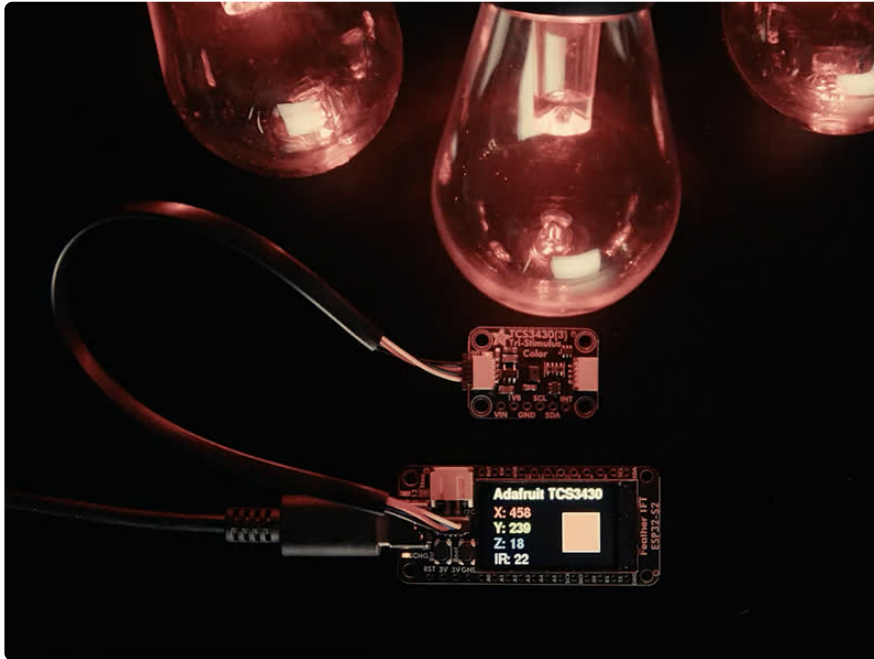
<https://learn.adafruit.com/adafruit-tcs3430-tcs34303-ambient-tri-stimulus-color-sensor>

Last updated on 2026-03-18 02:29:23 PM UTC

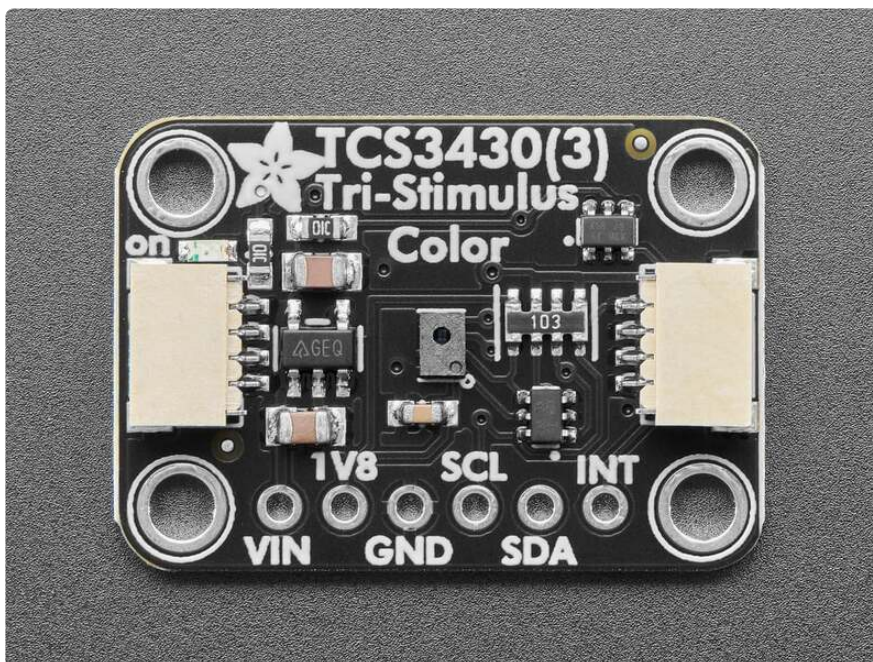
Table of Contents

Overview	3
Pinouts	5
<ul style="list-style-type: none">• Power Pins• I2C Data Pins• Interrupt Pin• LED and LED Jumper	
CircuitPython and Python	8
<ul style="list-style-type: none">• CircuitPython Microcontroller Wiring• Python Computer Wiring• Python Installation of TCS3430 Library• CircuitPython Usage• Python Usage• Example Code	
Python Docs	12
Arduino	13
<ul style="list-style-type: none">• Wiring• Library Installation• Example Code	
Arduino Docs	17
Downloads	17
<ul style="list-style-type: none">• Files• Schematic and Fab Print	

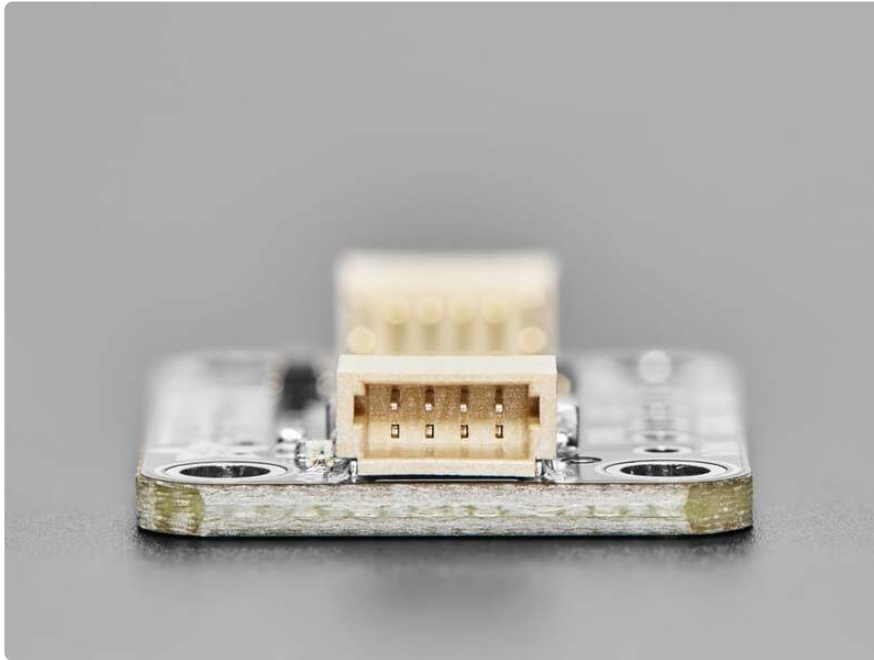
Overview



The Adafruit TCS3430 / TCS34303 (both names are used) Ambient Light Tri-Stimulus Color Sensor is a modern take on the common 'RGB' color sensor: this time with CIE XYZ plus IR diode sensors rather than RGB, so you can match it to your existing color space.

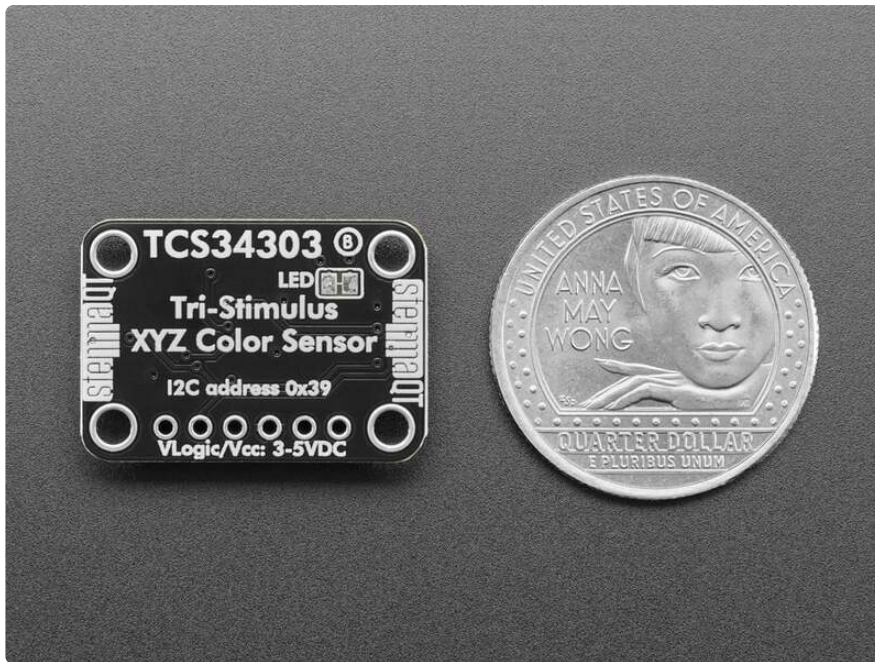


It acts a bit differently than the color sensors folks are used to, where there are sensors at wavelengths 437nm (X1), 574nm (X2), 537nm (Y) and 434nm (Z) and then the user is expected to calibrate it against a known color space analyzer for their application. For folks that want something that just spits out CIE and Lux data, the [OPT4048 is more complete](http://adafru.it/6335) (<http://adafru.it/6335>). However, there may be some use cases for people who want the raw count data only!

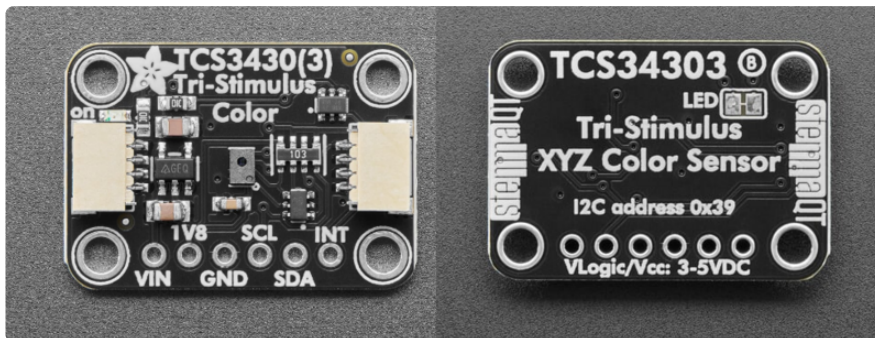


To make the sensor easy to use, we've taken the TCS34303 and put it onto a breakout PCB along with support circuitry to let you use this little wonder with 3.3V (Feather/Raspberry Pi) or 5V (Arduino/ Metro328) logic levels. Additionally since it speaks I2C, you can easily connect it up with two wires (plus power and ground!). We've even included [SparkFun qwiic](https://adafru.it/Fpw) (<https://adafru.it/Fpw>) compatible [STEMMA QT](https://adafru.it/Ft4) (<https://adafru.it/Ft4>) connectors for the I2C bus so **you don't even need to solder!**

Just wire up to your favorite micro and you can use our CircuitPython/Python or [Arduino drivers](https://adafru.it/1aBr) (<https://adafru.it/1aBr>) to easily interface with the TCS34303 and get XYZ+IR color readings in a minute! **QT Cable is not included**, [but we have a variety in the shop](http://adafru.it/4399) (<http://adafru.it/4399>).

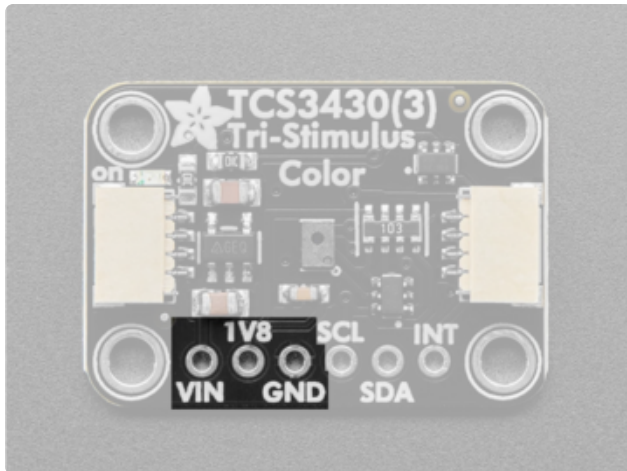


Pinouts



The fixed I2C address is **0x39**.

Power Pins

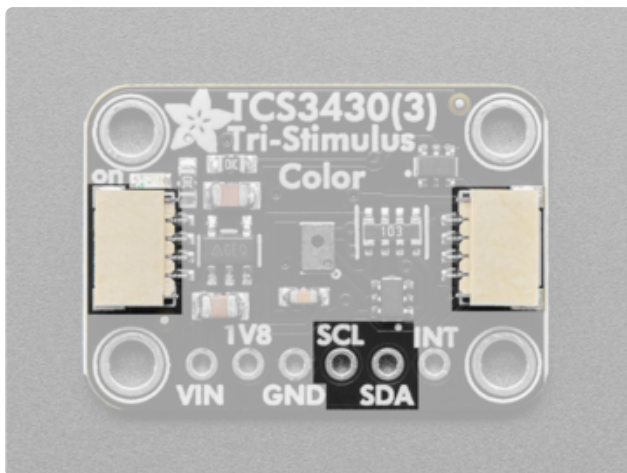


Vin - this is the power pin. Since the sensor chip uses 1.8VDC, we have included a voltage regulator on board that will take 3-5VDC and safely convert it to 1.8VDC. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V micro like Arduino, use 5V.

1V8 - this is the 1.8V output from the voltage regulator, you can grab up to 100mA from this if you like

GND - common ground for power and logic

I2C Data Pins

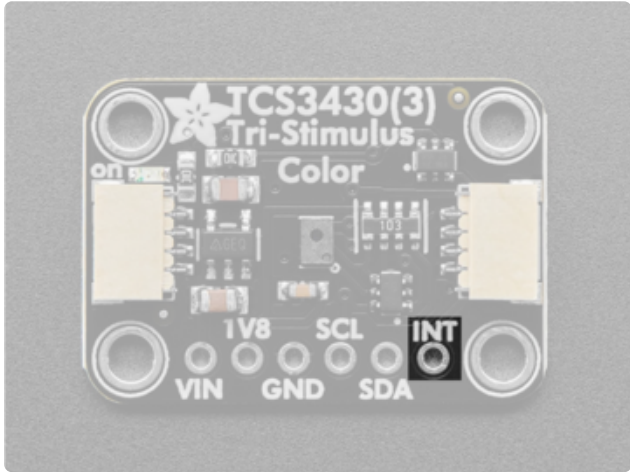


SCL - the I2C clock pin, connect to your microcontroller's I2C clock line.

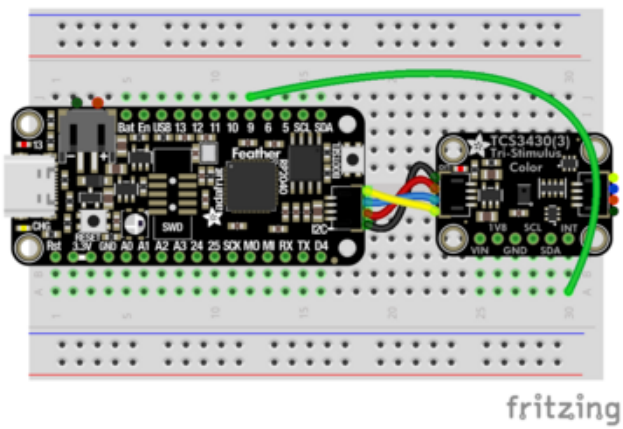
SDA - the I2C data pin, connect to your microcontroller's I2C data line.

[STEMMA QT \(https://adafru.it/Ft4\)](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 \(https://adafru.it/Ft6\)](https://adafru.it/Ft6)

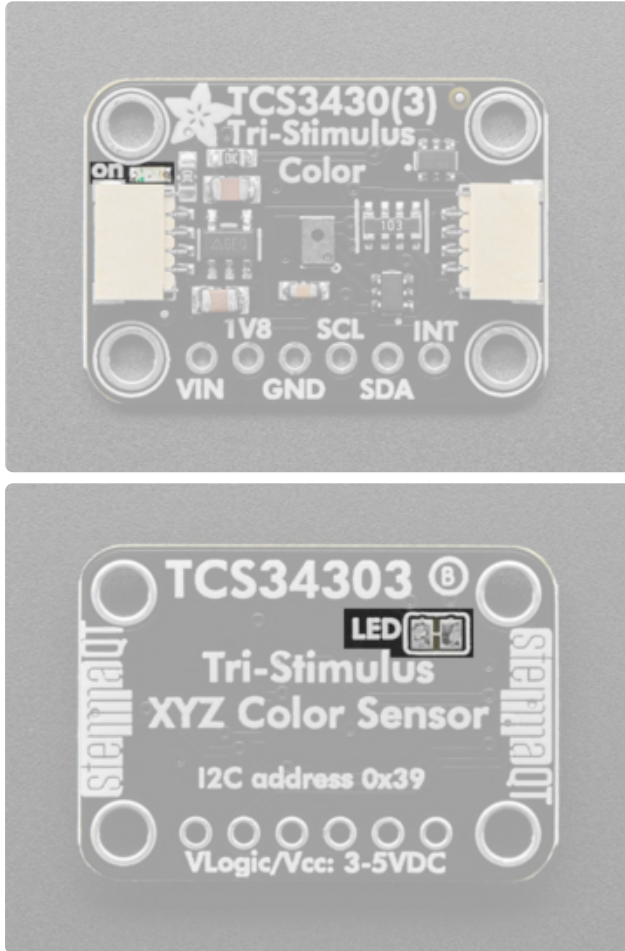
Interrupt Pin



INT - the interrupt pin. It is an open-drain interrupt. To use, connect the pin to a GPIO pin on your microcontroller.



LED and LED Jumper



Above the left STEMMA QT connector there is a small red **ON** power indicator LED. When the breakout is powered this LED will turn on.

On the back of the TCS3403 breakout there is a jumper with a small trace between two pads that can be cut to disable the **ON** power indicator LED on the board.

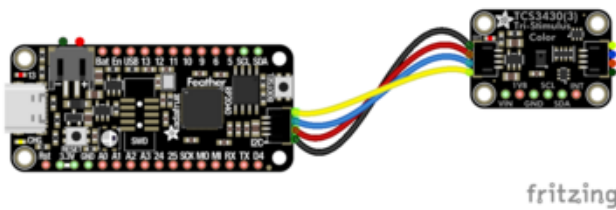
CircuitPython and Python

It's easy to use the **TCS3430** with Python or CircuitPython, and the [Adafruit_CircuitPython_TCS3430 \(https://adafru.it/1aBs\)](https://adafru.it/1aBs) module. This module allows you to easily write Python code to read data from the sensor.

You can use this driver with any CircuitPython microcontroller board or with a computer that has GPIO and Python [thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library \(https://adafru.it/BSN\)](https://adafru.it/BSN).

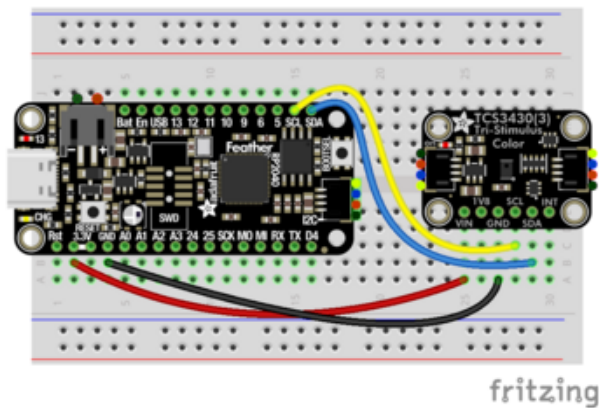
CircuitPython Microcontroller Wiring

First wire up a TCS3430 to your board exactly as shown below. Here's an example of wiring a Feather RP2040 to the sensor with I2C using one of the handy [STEMMA QT \(https://adafru.it/Ft4\)](https://adafru.it/Ft4) connectors:



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

The following is the TCS3430 wired to a Feather RP2040 using a solderless breadboard:

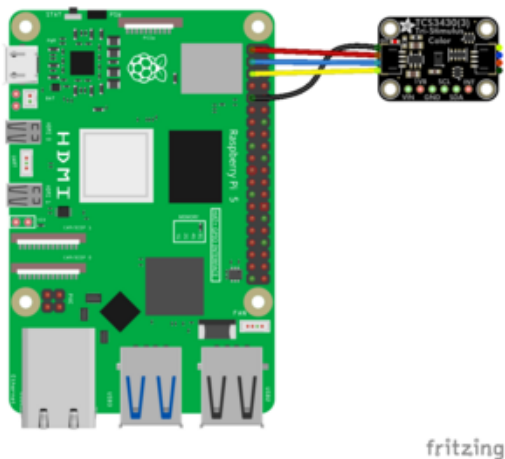


- Board 3V to sensor VIN (red wire)
- Board GND to sensor GND (black wire)
- Board SCL to sensor SCL (yellow wire)
- Board SDA to sensor SDA (blue wire)

Python Computer Wiring

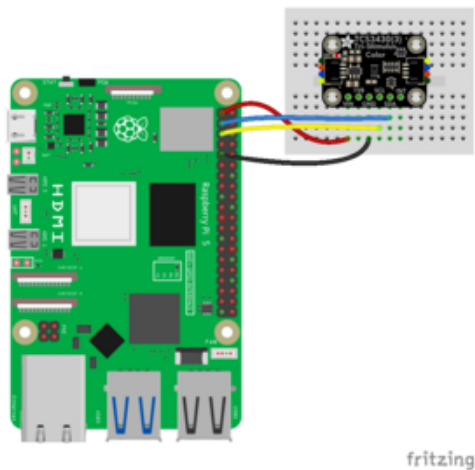
Since there are dozens of Linux computers/boards you can use, we will show wiring for Raspberry Pi. For other platforms, [please visit the guide for CircuitPython on Linux to see whether your platform is supported \(https://adafru.it/BSN\)](https://adafru.it/BSN).

Here's the Raspberry Pi wired with I2C using the STEMMA 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)

Here's the Raspberry Pi wired with I2C 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 TCS3430 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 \(https://adafru.it/BSN\)](https://adafru.it/BSN)!

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

- `pip3 install adafruit-circuitpython-tcs3430`

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!

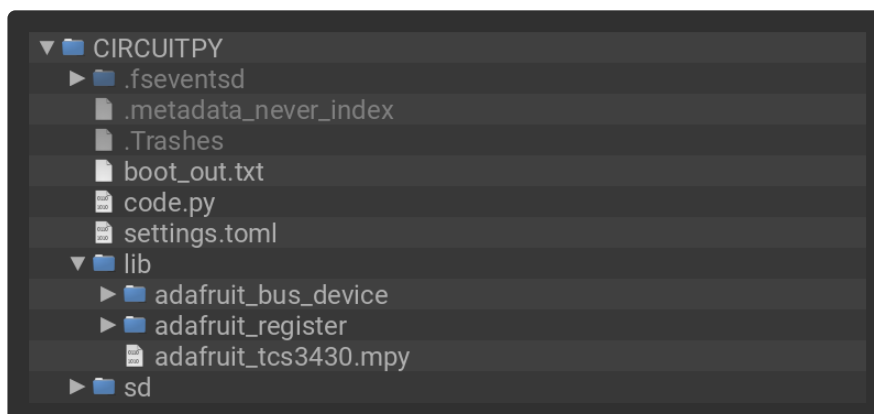
CircuitPython Usage

To use with CircuitPython, you need to first install the **Adafruit_CircuitPython_TCS3430** 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 **code.py** file to your **CIRCUITPY** drive.

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

- **adafruit_bus_device/**
- **adafruit_register/**
- **adafruit_tcs3430.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

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

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

```

# SPDX-FileCopyrightText: Copyright (c) 2026 Tim Cocks for Adafruit Industries
# SPDX-License-Identifier: MIT

"""Basic test for TCS3430 XYZ Tristimulus Color Sensor."""

import time

import board

from adafruit_tcs3430 import TCS3430, ALSGain, InterruptPersistence

i2c = board.I2C()
tcs = TCS3430(i2c)

print("TCS3430 Basic Test")
print("TCS3430 found!")

# --- Tweak these settings for your environment ---
tcs.als_gain = ALSGain.GAIN_64X # 1X, 4X, 16X, 64X, or 128X
tcs.integration_time = 100.0 # 2.78ms to 711ms

while True:
    x, y, z, ir1 = tcs.channels
    print(f"X: {x} Y: {y} Z: {z} IR1: {ir1}")

    time.sleep(1.0)

```

First, the sensor gets initialized over I2C. In the main loop the light channel data is read from the sensor and the values are printed to the serial console once per second.

```

code.py output:
TCS3430 Basic Test
TCS3430 found!
X: 327 Y: 1132 Z: 116 IR1: 1366
X: 328 Y: 1133 Z: 116 IR1: 1367
X: 186 Y: 559 Z: 93 IR1: 841
X: 5 Y: 3 Z: 2 IR1: 27
X: 175 Y: 818 Z: 48 IR1: 874
X: 174 Y: 813 Z: 48 IR1: 875
X: 5760 Y: 3687 Z: 1228 IR1: 1416
X: 173 Y: 812 Z: 48 IR1: 873
X: 173 Y: 811 Z: 48 IR1: 872
X: 174 Y: 812 Z: 48 IR1: 874

```

Python Docs

[Python Docs \(https://adafru.it/1aBq\)](https://adafru.it/1aBq)

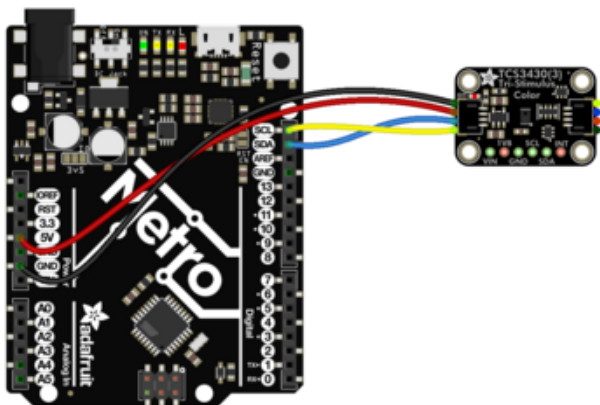
Arduino

Using the TCS3430 breakout with Arduino involves wiring up the breakout to your Arduino-compatible microcontroller, installing the [Adafruit_TCS3430](https://adafru.it/1aBr) (<https://adafru.it/1aBr>) 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 sensor **VIN**.

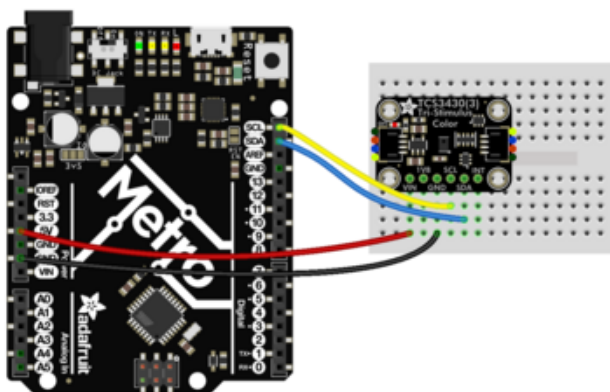
Here is an Adafruit Metro wired up to the sensor using the STEMMA QT connector:



fritzing

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

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

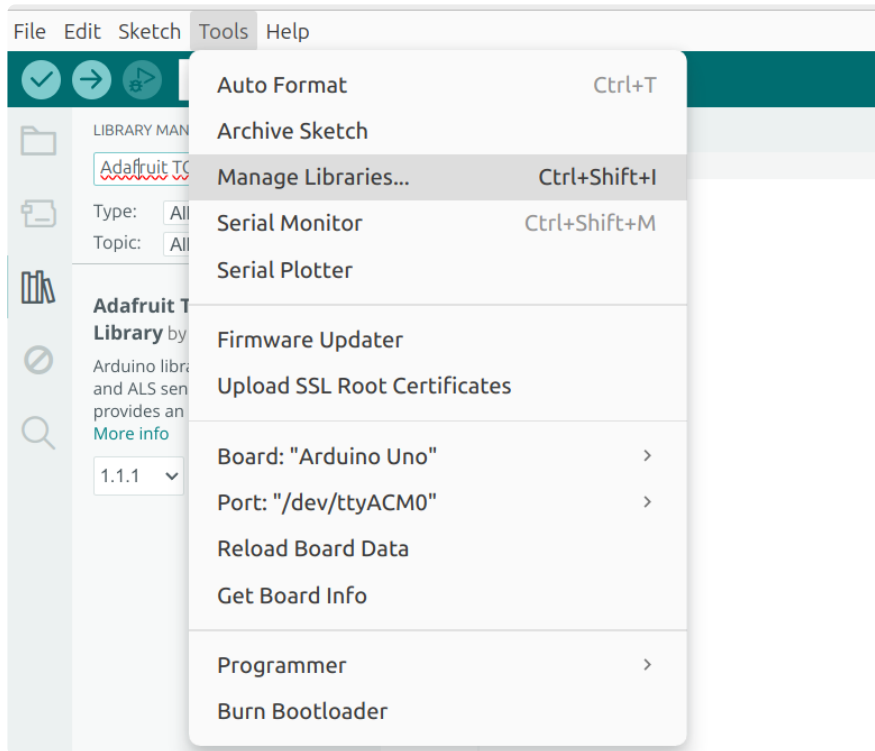


fritzing

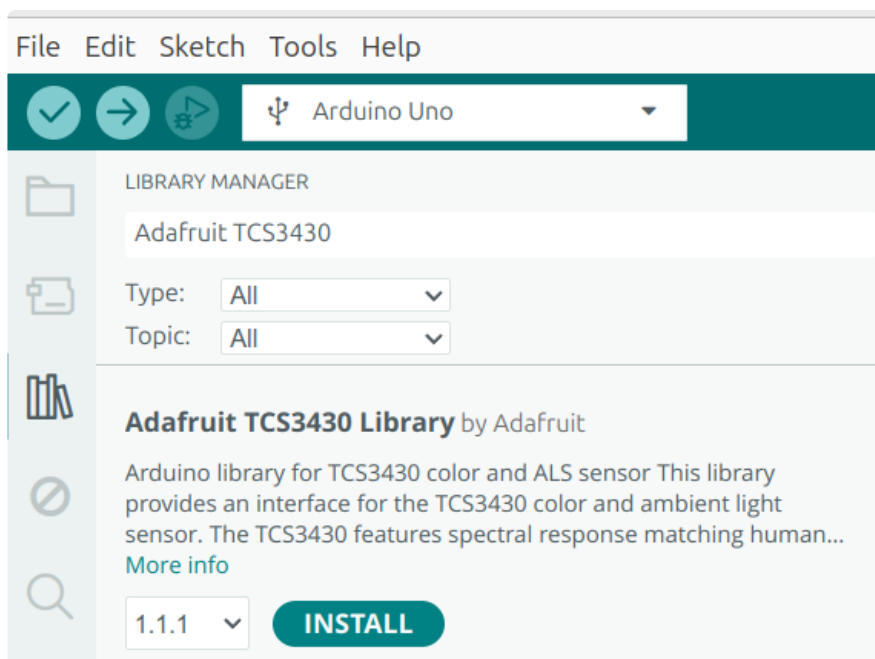
- Board 5V to breakout VIN (red wire)
- Board GND to breakout GND (black wire)
- Board SCL to breakout SCL (yellow wire)
- Board SDA to breakout SDA (blue wire)

Library Installation

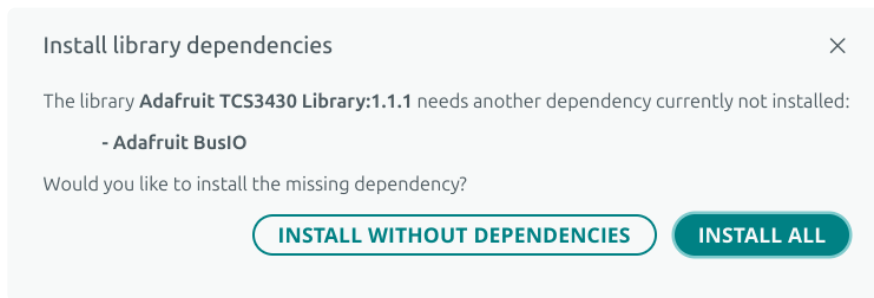
You can install the **Adafruit_TCS3430** library for Arduino using the Library Manager in the Arduino IDE.



Click the **Manage Libraries ...** menu item, search for **Adafruit TCS3430**, and select the **Adafruit TCS3430** library:



If asked about dependencies, click "Install all".



If the "Dependencies" window does not come up, then you already have the dependencies installed.



If the dependencies are already installed, you must make sure you update them through the Arduino Library Manager before loading the example!

Example Code

```
/*!
 * @file basicctest.ino
 *
 * Basic test sketch for TCS3430 XYZ Tristimulus Color Sensor.
 * Polls the ALS interrupt flag to know when new data is ready
 * instead of using a fixed delay.
 *
 * Limor 'ladyada' Fried with assistance from Claude Code
 * MIT License
 */

#include "Adafruit_TCS3430.h"

Adafruit_TCS3430 tcs = Adafruit_TCS3430();

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

  Serial.println(F("TCS3430 Basic Test"));

  if (!tcs.begin()) {
    Serial.println(F("Failed to find TCS3430 chip"));
    while (1) {
      delay(10);
    }
  }
}
```

```

Serial.println(F("TCS3430 found!"));

// --- Tweak these settings for your environment ---
tcs.setALSGain(TCS3430_GAIN_64X); // 1X, 4X, 16X, 64X, or 128X
tcs.setIntegrationTime(100.0); // 2.78ms to 711ms

// Enable ALS interrupt so we can poll AINT for data ready
tcs.enableALSInt(true);
tcs.setInterruptPersistence(TCS3430_PERS_EVERY);
tcs.clearALSInterrupt();
}

void loop() {
  // Wait for new data
  if (tcs.isALSInterrupt()) {
    uint16_t x, y, z, ir1;
    if (!tcs.getChannels(&x, &y, &z, &ir1)) {
      Serial.println(F("Failed to read channels"));
    } else {
      Serial.print(F("X: "));
      Serial.print(x);
      Serial.print(F(" Y: "));
      Serial.print(y);
      Serial.print(F(" Z: "));
      Serial.print(z);
      Serial.print(F(" IR1: "));
      Serial.println(ir1);
    }

    tcs.clearALSInterrupt();
  }

  // Do something else!
  delay(10);
}

```

Upload the sketch to your board and open up the Serial Monitor (**Tools -> Serial Monitor**) at 115200 baud. You'll see the TCS3430 recognized over I2C. Then, the reading values of different light channels will be printed out to the Serial Monitor.

Output Serial Monitor ×

Message (Enter to send message to 'Arduino Uno' on '/dev/ttyUSB0')

```
TCS3430 Basic Test
TCS3430 found!
X: 1 Y: 2 Z: 2 IR1: 23
X: 2 Y: 2 Z: 2 IR1: 23
X: 2 Y: 2 Z: 2 IR1: 23
X: 1 Y: 2 Z: 2 IR1: 23
X: 2 Y: 2 Z: 2 IR1: 24
X: 6 Y: 18 Z: 3 IR1: 37
X: 119 Y: 226 Z: 73 IR1: 166
X: 218 Y: 270 Z: 160 IR1: 248
X: 217 Y: 276 Z: 140 IR1: 230
X: 218 Y: 275 Z: 143 IR1: 231
X: 218 Y: 273 Z: 148 IR1: 235
X: 218 Y: 277 Z: 143 IR1: 232
```

Arduino Docs

[Arduino Docs \(https://adafru.it/1aAW\)](https://adafru.it/1aAW)

Downloads

Files

- [TCS3430 Datasheet \(https://adafru.it/1aBt\)](https://adafru.it/1aBt)
- [TCS3430 ColorMatrix Calibration Document \(https://adafru.it/1aBu\)](https://adafru.it/1aBu)
- [EagleCAD PCB files on GitHub \(https://adafru.it/1aBv\)](https://adafru.it/1aBv)
- [Fritzing object in the Adafruit Fritzing Library \(https://adafru.it/1aBw\)](https://adafru.it/1aBw)

Schematic and Fab Print

