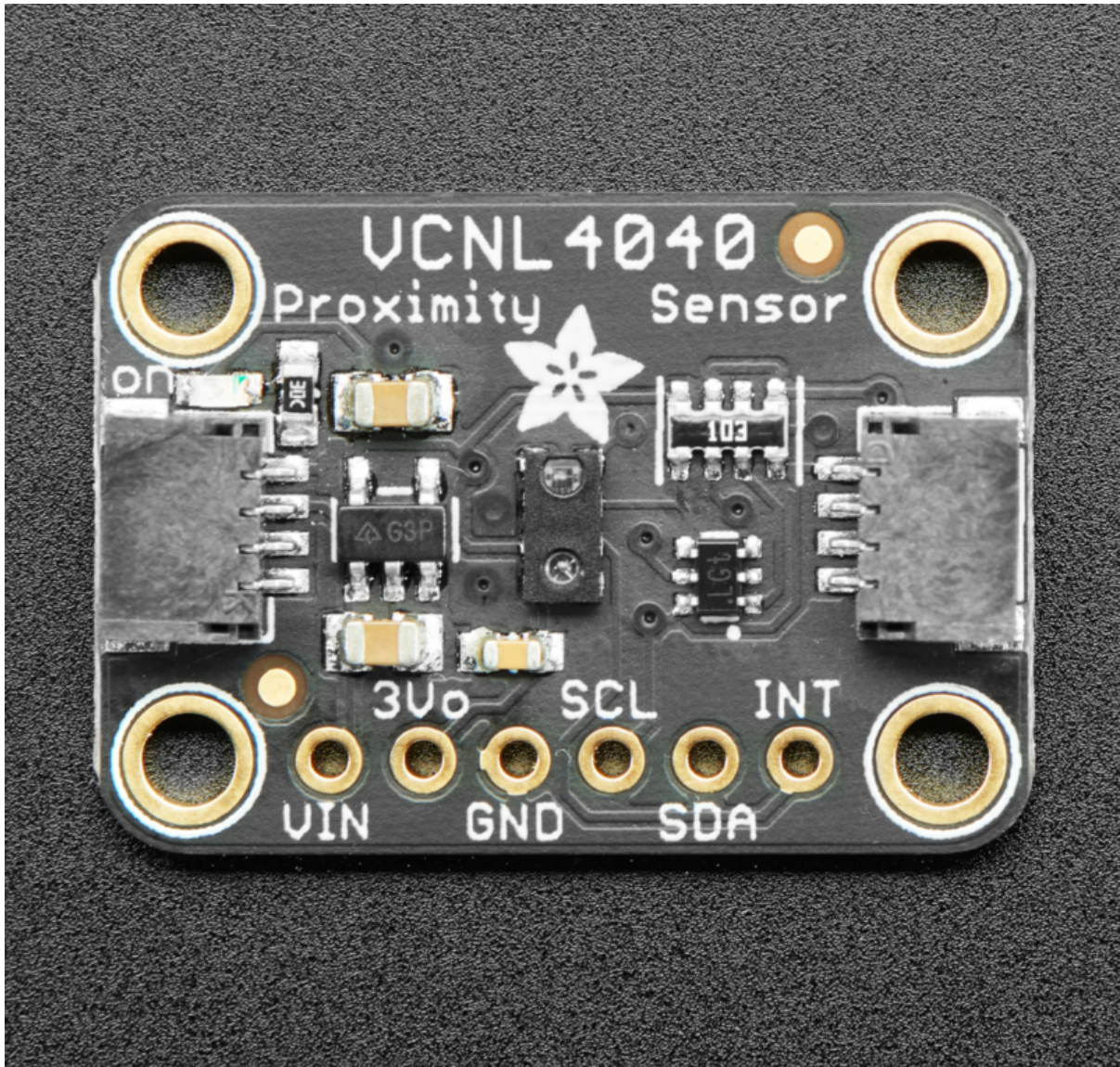




Adafruit VCNL4040 Proximity Sensor

Created by Bryan Siepert



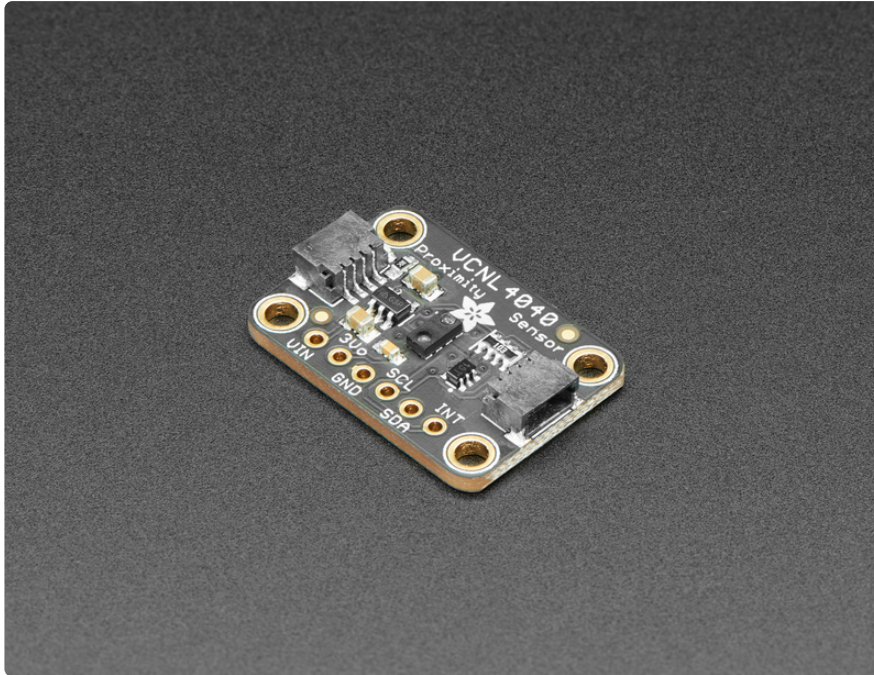
<https://learn.adafruit.com/adafruit-vcnl4040-proximity-sensor>

Last updated on 2024-06-03 02:50:19 PM EDT

Table of Contents

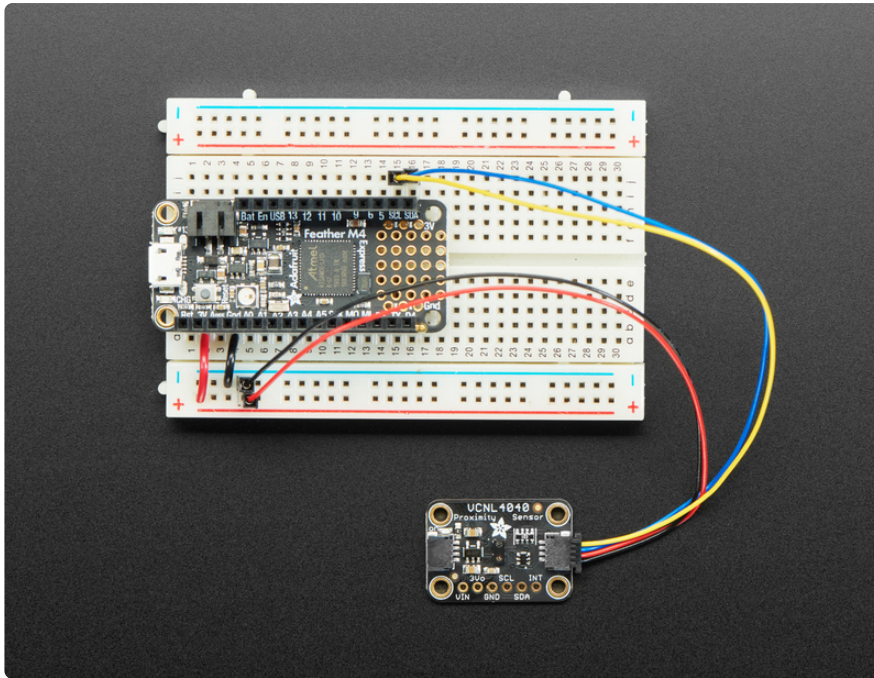
Overview	3
Pinouts	5
<ul style="list-style-type: none">• Power Pins• I2C Logic Pins• Other Pins	
Arduino	6
<ul style="list-style-type: none">• Installation• Load Example• Example Code	
Arduino Docs	9
Python & CircuitPython	9
<ul style="list-style-type: none">• CircuitPython Microcontroller Wiring• Python Computer Wiring• CircuitPython Installation of VCNL4040 Library• Python Installation of VCNL4040 Library• CircuitPython & Python Usage• Full Example Code	
Python Docs	13
WipperSnapper	13
<ul style="list-style-type: none">• What is WipperSnapper• Wiring• Usage	
Downloads	20
<ul style="list-style-type: none">• Files• Schematic• Fab Print	

Overview

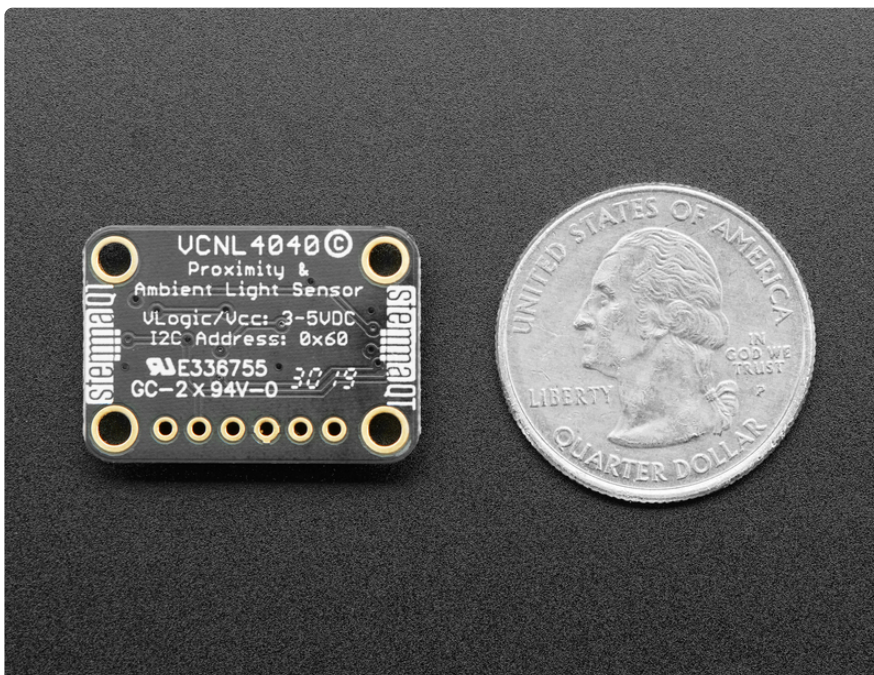


The VCNL4040 is a handy two-in-one sensor, with a proximity sensor that works from 0 to 200mm (about 7.5 inches) and light sensor with range of 0.0125 to 6553 lux.

We've all been there. That thing is close but how close? When you need to measure a small distance with reasonable accuracy, such as the rough height of particularly calm bumble bee, the VCNL4040 Proximity Sensor from Vishay can do that for you. If perchance you also needed to measure the amount of light at the same time, perhaps to let the bee to know if it's time for bed, you're in luck! The VCNL4040 can do that too (bumble bee not included, we tried putting it in the anti-static bag but it started buzzing in a threatening manner).



"OK, finally I can get started on my bee measuring and light sensing project, but how do I use it?" you say. To make life easier so you can focus on your important work, we've taken the VCNL4040 and put it onto a breakout PCB along with support circuitry to let you use this little wonder with 3.3V (Feather) 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 \(https://adafruit.com/product/444\)](https://adafruit.com/product/444) compatible [STEMMA QT \(https://adafruit.com/product/444\)](https://adafruit.com/product/444) 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 to easily interface with the VCNL4040 and make approximate approximations of proximity in no time!



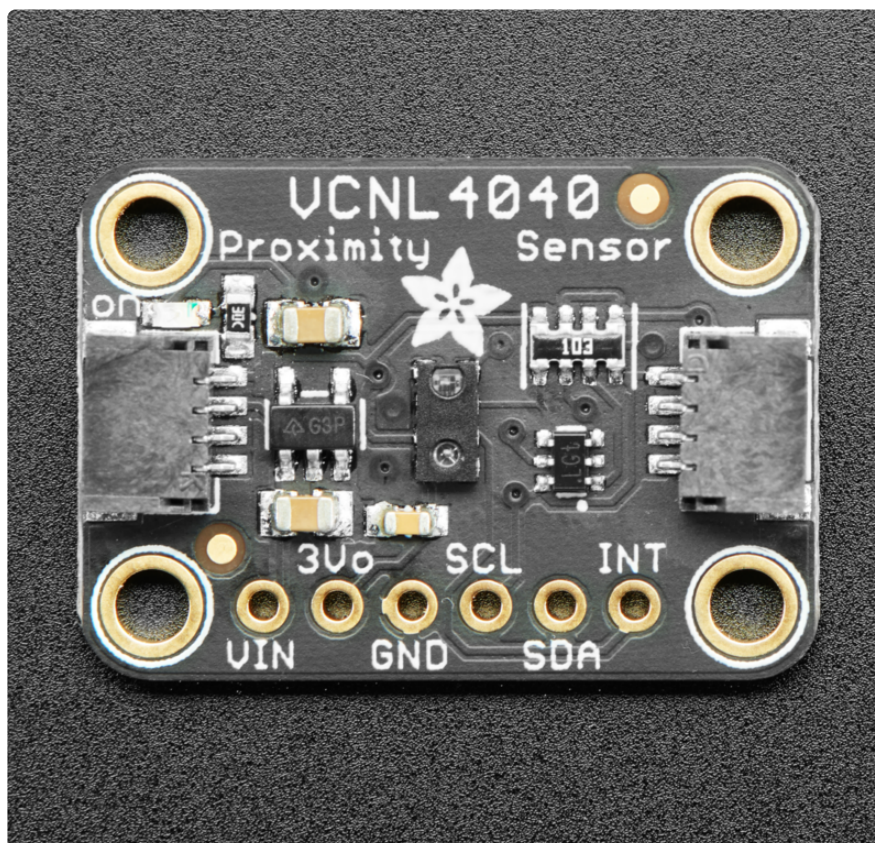
To give you some ability to tune your measurements for your situation, the VCNL4040 lets you adjust the integration time (how long light is being measured) to set the sensitivity of your measurements depending on the lighting conditions. With an integration time of 80ms, since the sensor is only collecting measurements for a short period of time, each measured bit of light represents a larger amount of light. For this reason the maximum range in lux for the 80ms integration time is 6553.5 lux.

If you use a longer integration time and get the same amount of light measurements "events", you know that there is comparatively less light because we're waiting around longer for light to show up and getting the same number, meaning the rate of light events is lower. For this reason at the maximum integration time of 640ms the VCNL4040 will only measure up to 819.2 lux.

It is also worth keeping in mind that as you increase the integration time you are also increasing the sensitivity and resolution within the measurement range

In addition to adjusting the sensitivity of light measurements, you can change the current and duty cycle of the IR LED that powers the proximity detection to adjust how sensitive proximity measurements are

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 micro 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

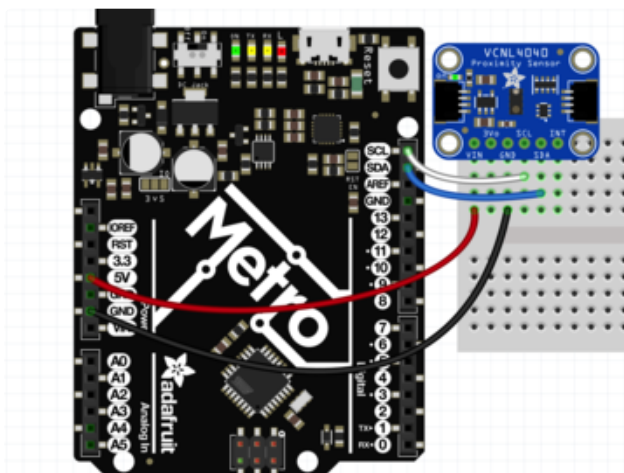
I2C Logic Pins

- **SCL** - this is the I2C clock pin, connect to your microcontroller's I2C clock line.
- **SDA** - this is the I2C data pin, connect to your microcontroller's I2C data line
- **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) (<https://adafru.it/Ft6>)

Other Pins

- **INT** - This is the interrupt pin. You can setup the VCNL4040 to pull this low when certain conditions are met such as proximity or lux level thresholds being crossed.

Arduino

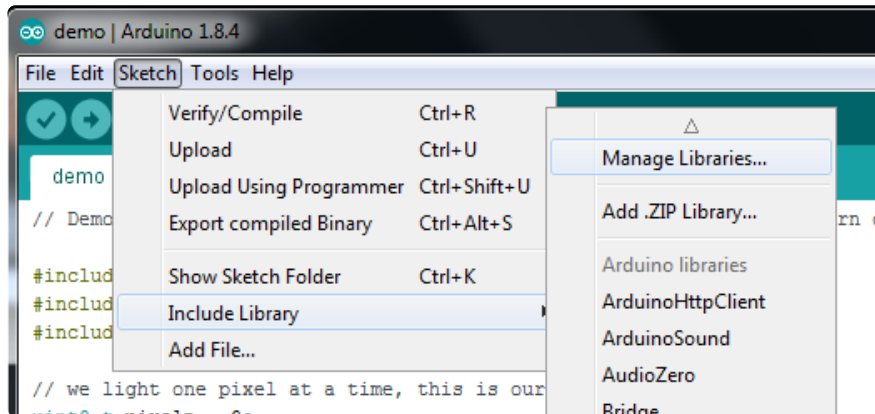


- Connect **Metro or Arduino 5V** to board **VIN**
- Connect **Metro or Arduino GND** to board **GND**
- Connect **Metro or Arduino SCL** to board **SCL**
- Connect **Metro or Arduino SDA** to board **SDA**

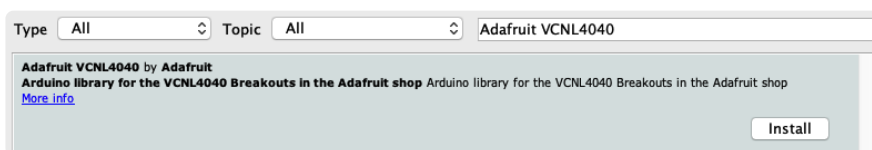
The final results should resemble the illustration above, showing an [Adafruit Metro](http://adafru.it/2488) (<http://adafru.it/2488>) development board.

Installation

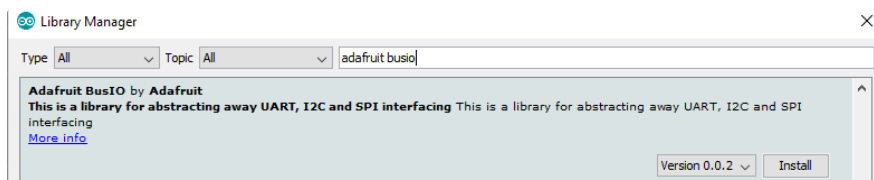
You can install the **Adafruit VCNL4040 Library** for Arduino using the Library Manager in the Arduino IDE:



Click the **Manage Libraries ...** menu item, search for **Adafruit VCNL4040**, and select the **Adafruit VCNL4040** library:



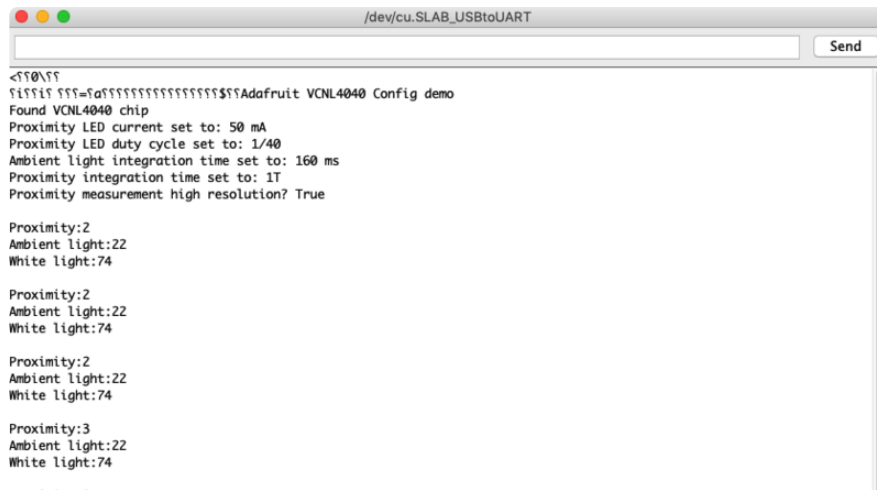
Then follow the same process for the **Adafruit BusIO** library.



Load Example

Open up **File -> Examples -> Adafruit VCNL4040 -> vcnl4040_test** and upload to your Arduino wired up to the sensor.

Upload the sketch to your board and open up the Serial Monitor (**Tools->Serial Monitor**) at 115200 baud. You should see the the values for the current configuration settings printed on startup, followed by readings for ambient light, proximity levels, and raw white light readings similar to this:



You can place your hand close to the sensor to see the proximity value change, and to see the ambient light levels change you can either cover the sensor or shine a light at it.

Example Code

The following code is part of the standard library and illustrates the basic function of sensing light levels and measuring proximity:

```
#include <Adafruit_VCNL4040.h>

Adafruit_VCNL4040 vcnl4040 = Adafruit_VCNL4040();

void setup() {
  Serial.begin(115200);
  // Wait until serial port is opened
  while (!Serial) { delay(1); }

  Serial.println("Adafruit VCNL4040 Config demo");

  if (!vcnl4040.begin()) {
    Serial.println("Couldn't find VCNL4040 chip");
    while (1);
  }
  Serial.println("Found VCNL4040 chip");

  //vcnl4040.setProximityLEDCurrent(VCNL4040_LED_CURRENT_200MA);
  Serial.print("Proximity LED current set to: ");
  switch(vcnl4040.getProximityLEDCurrent()) {
    case VCNL4040_LED_CURRENT_50MA: Serial.println("50 mA"); break;
    case VCNL4040_LED_CURRENT_75MA: Serial.println("75 mA"); break;
    case VCNL4040_LED_CURRENT_100MA: Serial.println("100 mA"); break;
    case VCNL4040_LED_CURRENT_120MA: Serial.println("120 mA"); break;
    case VCNL4040_LED_CURRENT_140MA: Serial.println("140 mA"); break;
    case VCNL4040_LED_CURRENT_160MA: Serial.println("160 mA"); break;
    case VCNL4040_LED_CURRENT_180MA: Serial.println("180 mA"); break;
    case VCNL4040_LED_CURRENT_200MA: Serial.println("200 mA"); break;
  }

  //vcnl4040.setProximityLEDDutyCycle(VCNL4040_LED_DUTY_1_40);
  Serial.print("Proximity LED duty cycle set to: ");
  switch(vcnl4040.getProximityLEDDutyCycle()) {
    case VCNL4040_LED_DUTY_1_40: Serial.println("1/40"); break;
```

```

    case VCNL4040_LED_DUTY_1_80: Serial.println("1/80"); break;
    case VCNL4040_LED_DUTY_1_160: Serial.println("1/160"); break;
    case VCNL4040_LED_DUTY_1_320: Serial.println("1/320"); break;
}

//vcnl4040.setAmbientIntegrationTime(VCNL4040_AMBIENT_INTEGRATION_TIME_80MS);
Serial.print("Ambient light integration time set to: ");
switch(vcnl4040.getAmbientIntegrationTime()) {
    case VCNL4040_AMBIENT_INTEGRATION_TIME_80MS: Serial.println("80 ms"); break;
    case VCNL4040_AMBIENT_INTEGRATION_TIME_160MS: Serial.println("160 ms"); break;
    case VCNL4040_AMBIENT_INTEGRATION_TIME_320MS: Serial.println("320 ms"); break;
    case VCNL4040_AMBIENT_INTEGRATION_TIME_640MS: Serial.println("640 ms"); break;
}

//vcnl4040.setProximityIntegrationTime(VCNL4040_PROXIMITY_INTEGRATION_TIME_8T);
Serial.print("Proximity integration time set to: ");
switch(vcnl4040.getProximityIntegrationTime()) {
    case VCNL4040_PROXIMITY_INTEGRATION_TIME_1T: Serial.println("1T"); break;
    case VCNL4040_PROXIMITY_INTEGRATION_TIME_1_5T: Serial.println("1.5T"); break;
    case VCNL4040_PROXIMITY_INTEGRATION_TIME_2T: Serial.println("2T"); break;
    case VCNL4040_PROXIMITY_INTEGRATION_TIME_2_5T: Serial.println("2.5T"); break;
    case VCNL4040_PROXIMITY_INTEGRATION_TIME_3T: Serial.println("3T"); break;
    case VCNL4040_PROXIMITY_INTEGRATION_TIME_3_5T: Serial.println("3.5T"); break;
    case VCNL4040_PROXIMITY_INTEGRATION_TIME_4T: Serial.println("4T"); break;
    case VCNL4040_PROXIMITY_INTEGRATION_TIME_8T: Serial.println("8T"); break;
}

//vcnl4040.setProximityHighResolution(false);
Serial.print("Proximity measurement high resolution? ");
Serial.println(vcnl4040.getProximityHighResolution() ? "True" : "False");

Serial.println("");
}

void loop() {

    Serial.print("Proximity:"); Serial.println(vcnl4040.getProximity());
    Serial.print("Ambient light:"); Serial.println(vcnl4040.getLux());
    Serial.print("Raw white light:"); Serial.println(vcnl4040.getWhiteLight());
    Serial.println("");

    delay(500);
}

```

Arduino Docs

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

Python & CircuitPython

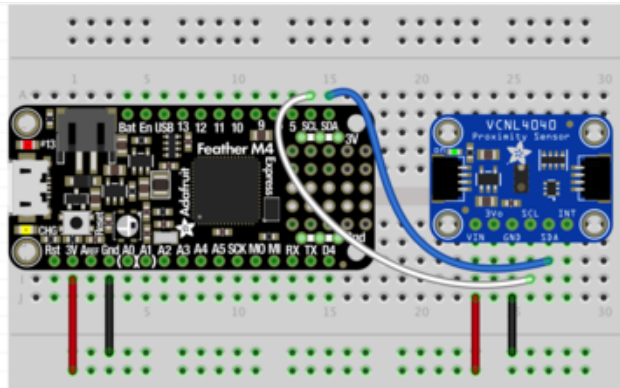
It's easy to use the VCNL4040 sensor with CircuitPython and the [Adafruit CircuitPython VCNL4040 \(https://adafru.it/Ftd\)](https://adafru.it/Ftd) module. This module allows you to easily write Python code that reads the ambient light levels, white light levels, and proximity data from the breakout.

You can use this sensor with any CircuitPython microcontroller board or with a Linux

single board 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 VCNL4040 to your board exactly as follows. Here is an example of the VCNL4040 wired to a [Feather](http://adafru.it/3857) (<http://adafru.it/3857>) using I2C:



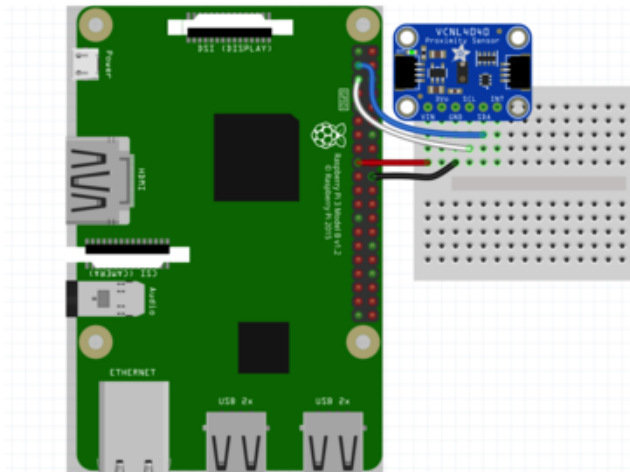
Board 3V to sensor VIN
Board GND to sensor GND
Board SCL to sensor SCL
Board SDA to sensor SDA

Note: This breakout includes pullup resistors on the I2C lines, no external pullups are required.

Python Computer Wiring

Since there's dozens of Linux computers/boards you can use we will show wiring for [Raspberry Pi \(http://adafru.it/3055\)](http://adafru.it/3055). 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
Pi GND to sensor GND
Pi SCL to sensor SCL
Pi SDA to sensor SDA

CircuitPython Installation of VCNL4040 Library

You'll need to install the [Adafruit CircuitPython VCNL4040 \(https://adafru.it/Ftd\)](https://adafru.it/Ftd) library on your CircuitPython board.

First make sure you are running the [latest version of Adafruit CircuitPython \(https://adafru.it/Amd\)](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 \(https://adafru.it/uap\)](https://adafru.it/uap). Our CircuitPython starter guide has [a great page on how to install the library bundle \(https://adafru.it/ABU\)](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_vcnl4040.mpy`
- `adafruit_bus_device`
- `adafruit_register`

Before continuing make sure your board's lib folder or root filesystem has the `adafruit_vcnl4040.mpy`, `adafruit_bus_device`, and `adafruit_register` files and folders copied over.

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

Python Installation of VCNL4040 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:

- `sudo pip3 install adafruit-circuitpython-vcnl4040`

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 light levels and proximity 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 busio
import adafruit_vcnl4040

i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_vcnl4040.VCNL4040(i2c)
```

```
>>> import time
>>> import board
>>> import busio
>>> import adafruit_vcnl4040
>>> i2c = busio.I2C(board.SCL, board.SDA)
>>> sensor = adafruit_vcnl4040.VCNL4040(i2c)
>>>
```

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

- **lux** - The ambient light data in lux
- **proximity** - The proximity data measurement

For example to print ambient light level and proximity info:

```
>>> sensor.lux
384.3
>>> sensor.proximity
45
```

For more details, check out the [library documentation \(https://adafru.it/Fte\)](https://adafru.it/Fte).

That's all there is to using the VCNL4040 sensor with CircuitPython!

Full Example Code

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

import time
import board
import adafruit_vcnl4040

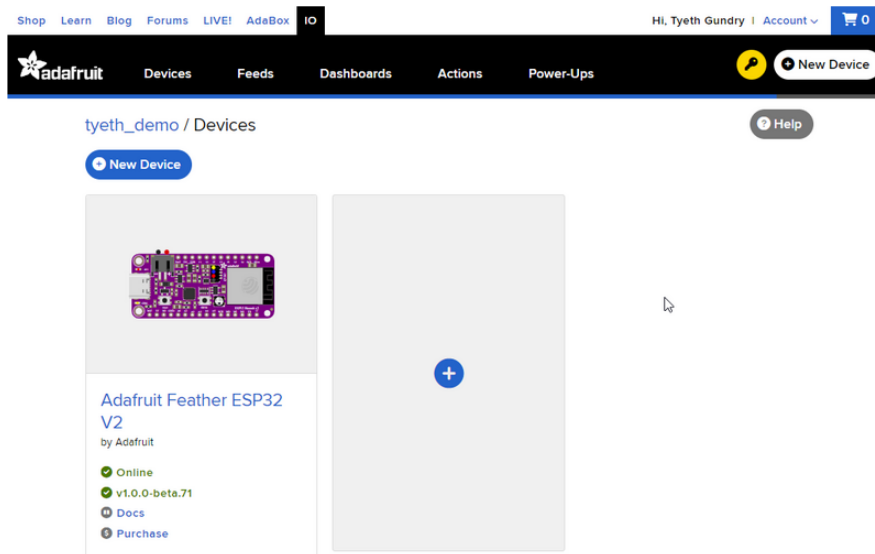
i2c = board.I2C() # uses board.SCL and board.SDA
# i2c = board.STEMMA_I2C() # For using the built-in STEMMMA QT connector on a
microcontroller
sensor = adafruit_vcnl4040.VCNL4040(i2c)

while True:
    print("Proximity:", sensor.proximity)
    print("Light: %d lux" % sensor.lux)
    time.sleep(1.0)
```

Python Docs

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

WipperSnapper



The screenshot shows the Adafruit website's WipperSnapper interface. At the top, there's a navigation bar with links for Shop, Learn, Blog, Forums, LIVE!, AdaBox, and IO. The user is logged in as 'Hi, Tyeth Gundry' with an account menu and a shopping cart icon. Below the navigation bar, there's a main header with 'tyeth_demo / Devices' and a 'Help' button. A 'New Device' button is also present. The main content area shows a device card for the 'Adafruit Feather ESP32 V2' by Adafruit. The card includes a small image of the device, a status indicator 'Online', the version 'v1.0.0-beta.71', and links for 'Docs' and 'Purchase'. To the right of the device card is a large grey area with a blue plus sign, indicating a placeholder for another device.

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 \(https://adafru.it/fsU\)](https://adafru.it/fsU), a web platform designed ([by Adafruit! \(https://adafru.it/Bo5\)](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.

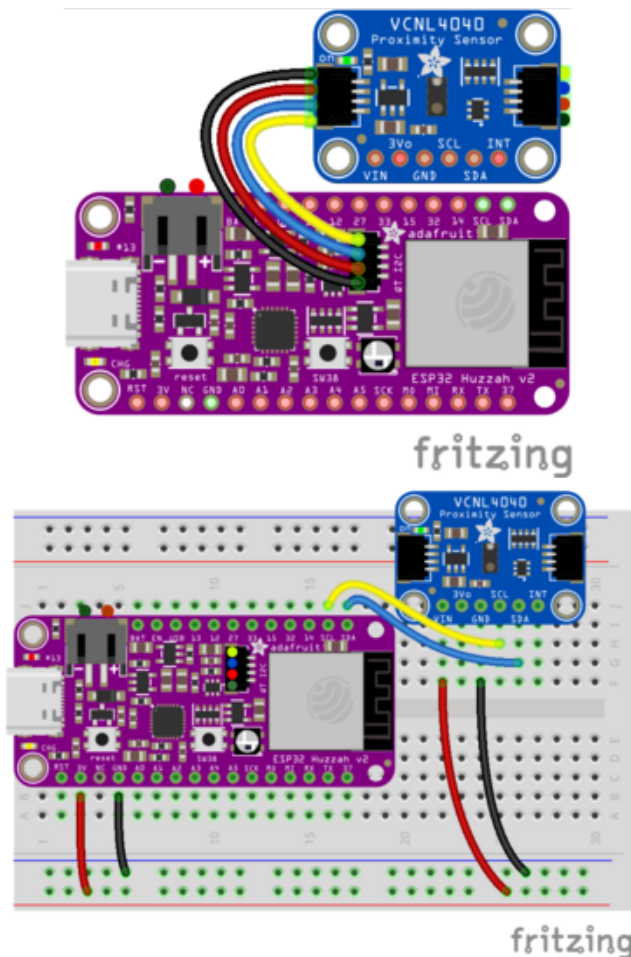
If you've never used WipperSnapper, click below to read through the quick start guide before continuing.

**Quickstart: Adafruit IO
WipperSnapper**

<https://adafru.it/Vfd>

Wiring

First, wire up an VCNL4040 to your board exactly as follows. Here is an example of the VCNL4040 wired to an [Adafruit ESP32 Feather V2 \(http://adafru.it/5400\)](http://adafru.it/5400) using I2C [with a STEMMA QT cable \(no soldering required\) \(http://adafru.it/4210\)](http://adafru.it/4210)



Board 3V to sensor VIN (red wire on STEMMA QT)

Board GND to sensor GND (black wire on STEMMA QT)

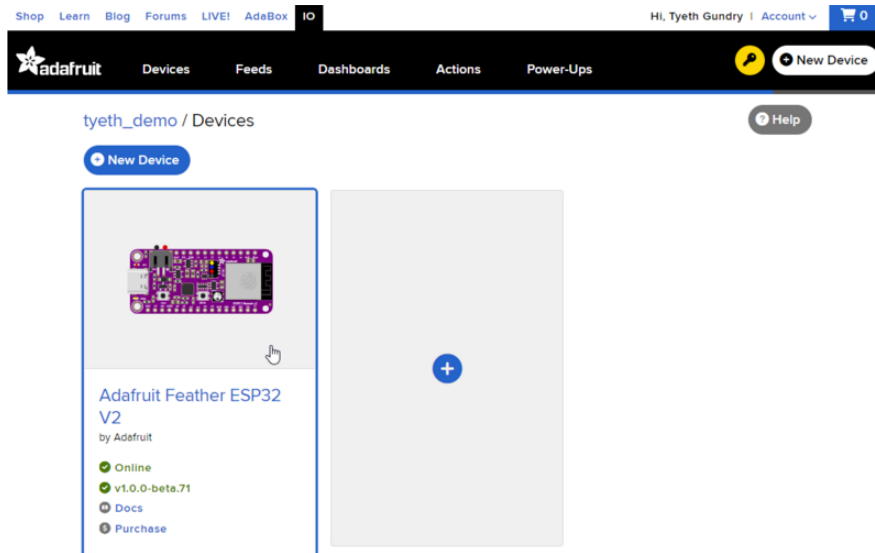
Board SCL to sensor SCL (yellow wire on STEMMA QT)

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 \(https://adafru.it/TAu\)](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.

Adafruit Feather ESP32 V2

by Adafruit

✓ Online

✓ v1.0.0-beta.70 

📖 Docs

💰 Purchase

Adafruit Feather ESP32 V2

by Adafruit

✓ Online

! v1.0.0-beta.68  Update 

📖 Docs

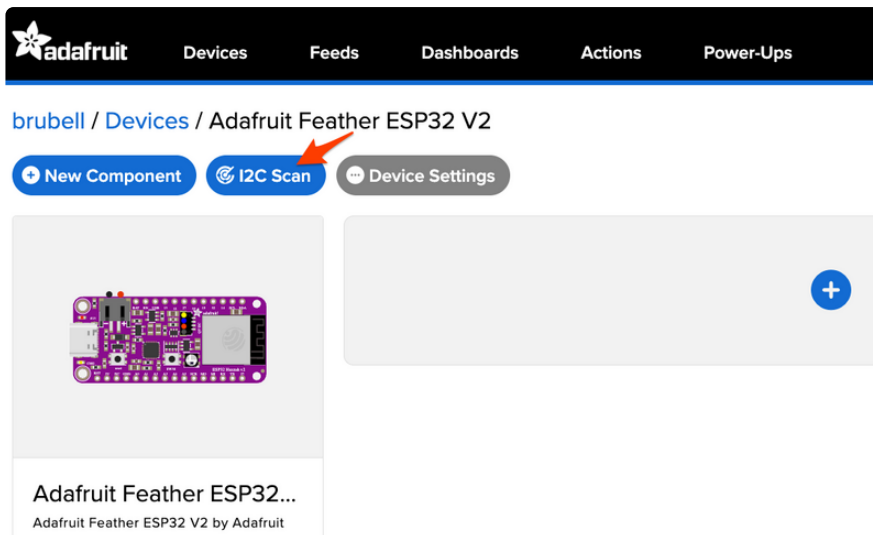
💰 Purchase

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 VCNL4040's default I2C address of `0x60` pop-up in the I2C scan list.

I2C Scan Complete ✕

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
00								--	--	--	--	--	--	--	--	--
10	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
20	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
30	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
40	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
50	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
60	60	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
70	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

Close Scan Again

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

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

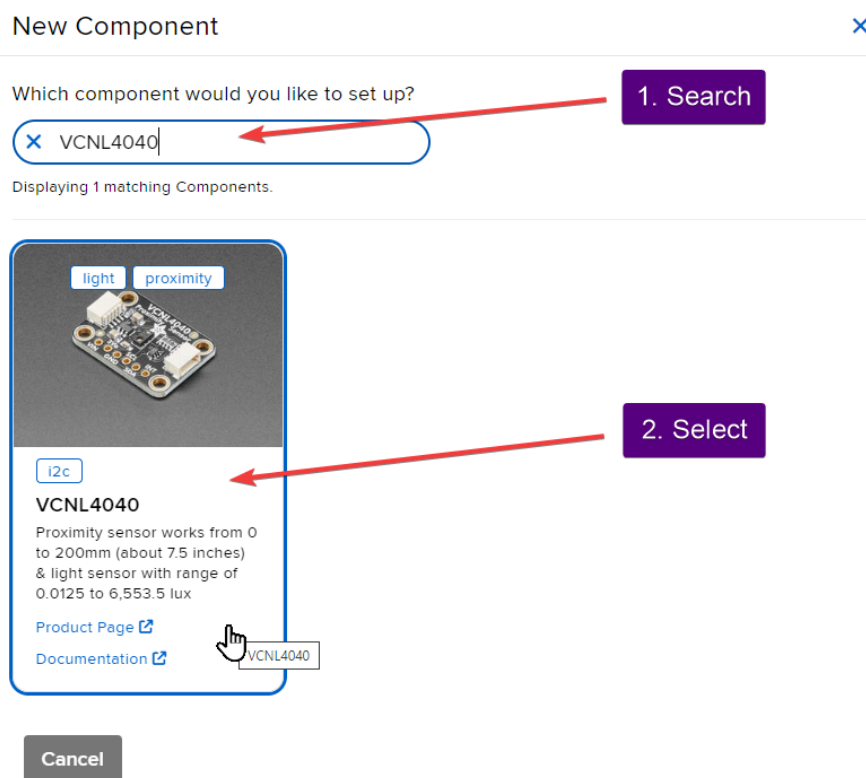
Then, reset the board and let it re-connect to Adafruit IO WipperSnapper.

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 **VCNL4040** into the search bar, then select the **VCNL4040** component.



On the component configuration page, the VCNL4040'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 VCNL4040 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.

Create VCNL4040 Component



Select I2C Address:

0x60

Enable VCNL4040: Light Sensor?

Name:

VCNL4040: Light Sensor

Send Data:

Every 30 seconds

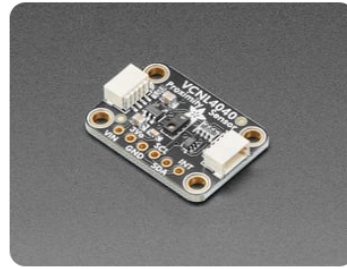
Enable VCNL4040: Proximity Sensor?

Name:

VCNL4040: Proximity Sensor

Send Data:

Every 30 seconds



[← Back to Component Type](#)

[Create Component](#)



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.

The screenshot shows the Adafruit IO dashboard for a device named 'tyeth_demo / Devices / Adafruit Feather ESP32 V2'. The dashboard includes navigation links for Shop, Learn, Blog, Forums, LIVE!, AdaBox, and IO. The user is logged in as 'Hi, Tyeth Gundry | Account'. The dashboard features several buttons: 'New Component', 'Auto-Config', 'I2C Scan', 'Help', and 'Settings'. Two sensor components are displayed:

- VCNL4040: Light Sensor** (vcnl4040.light) with a value of 284.00.
- VCNL4040: Proximity Sensor** (vcnl4040.proximity) with a value of 11.00.

Each sensor component has a 'Create Action | Add to Dashboard' link. A 'Report Bugs' link is also visible at the bottom.

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

tyeth_demo / Devices / Adafruit Feather ESP32 V2

New Component Auto-Config I2C Scan Help Settings

Adafruit Feather ESP32 V2 by Adafruit

- Online
- v1.0.0-beta.76
- Docs
- Purchase

Report Bugs

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 \(https://adafru.it/10aZ\)](https://adafru.it/10aZ).

tyeth_demo / Feeds / VCNL4040: Light Sensor

Feed Info: Manage feed name, key, description, and tags.

Privacy: This feed is **private**. Only you can see it.

Sharing: Not shared yet

Feed History: Feed history is **ON**. Value size is limited to **1KB**. You have 43 data points from March 1st 2024, 11:57AM to March 1st 2024, 12:18PM.

Notifications: This feed is **Online**. You have no notifications active for this feed.

Webhooks: Webhooks let you connect your feed to the rest of the web.

Disable Feed: Disabling a feed will remove it from your feed count and prevent you from adding new data to it.

License: No Default License

Created at	Value	Location
2024/03/01 12:18:20PM	4010	
2024/03/01 12:17:50PM	4018	
2024/03/01 12:17:20PM	4022	
2024/03/01 12:16:50PM	4189	
2024/03/01 12:16:20PM	4289	
2024/03/01 12:15:50PM	4358	
2024/03/01 12:15:19PM	4317	
2024/03/01 12:14:49PM	4471	
2024/03/01 12:14:19PM	4547	

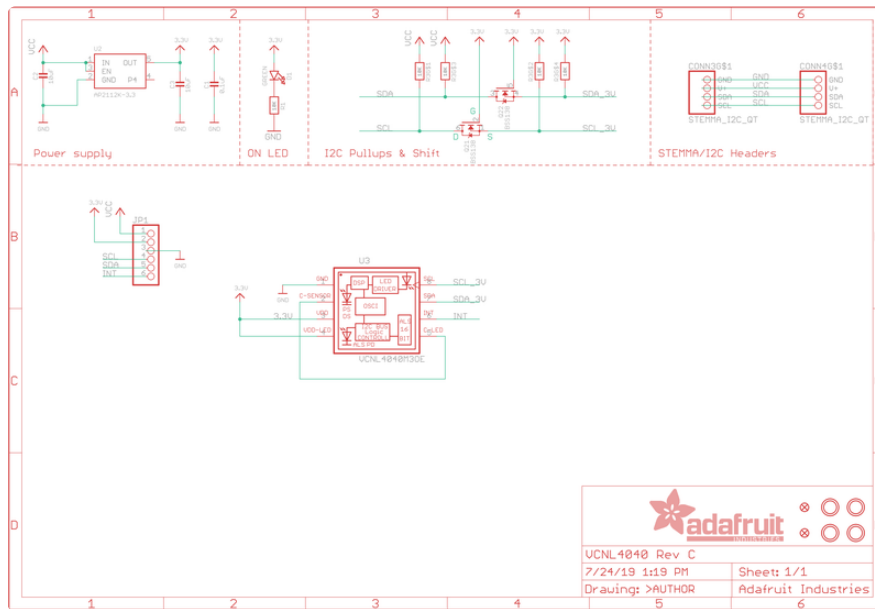
Downloads

Files

- [VCNL4040 Datasheet \(https://adafru.it/Ftf\)](https://adafru.it/Ftf)

- EagleCAD files on GitHub (<https://adafru.it/Ftg>)
- Fritzing object from Adafruit Fritzing Library (<https://adafru.it/Fth>)

Schematic



Fab Print

