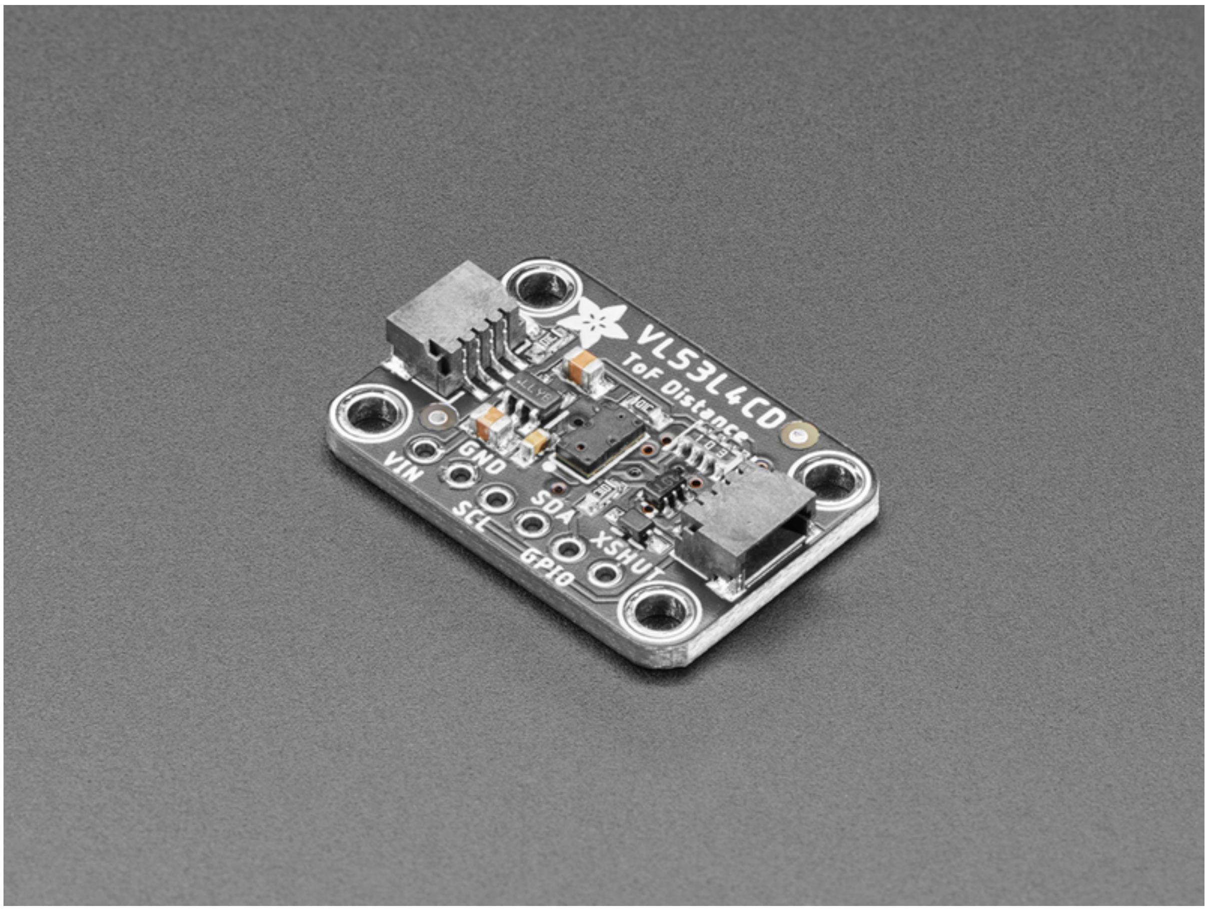
Adafruit VL53L4CD Time of Flight Distance Sensor

Created by Liz Clark

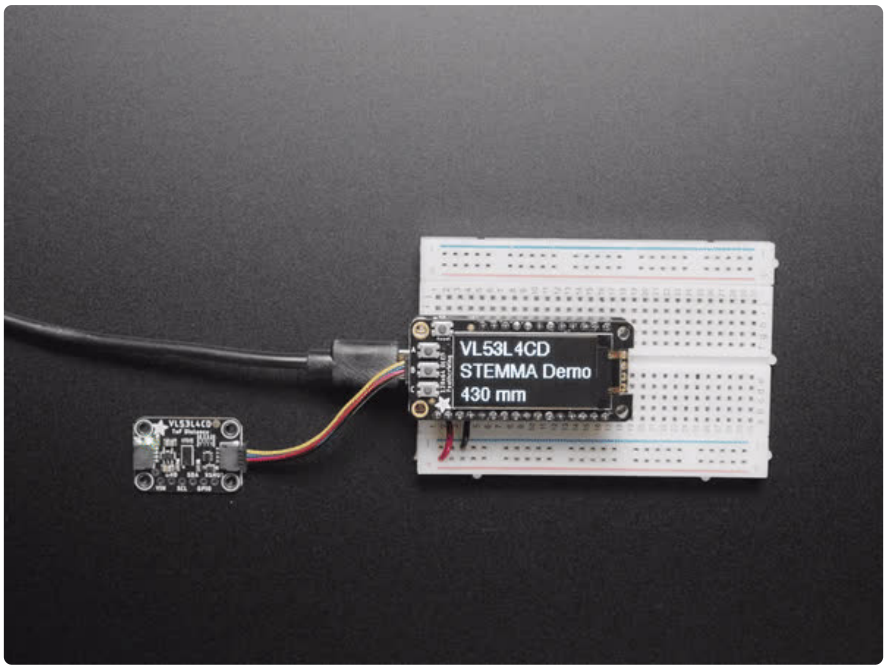


https://learn.adafruit.com/adafruit-vl53l4cd-time-of-flight-distance-sensor

Last updated on 2024-06-03 03:34:51 PM EDT

Table of Contents

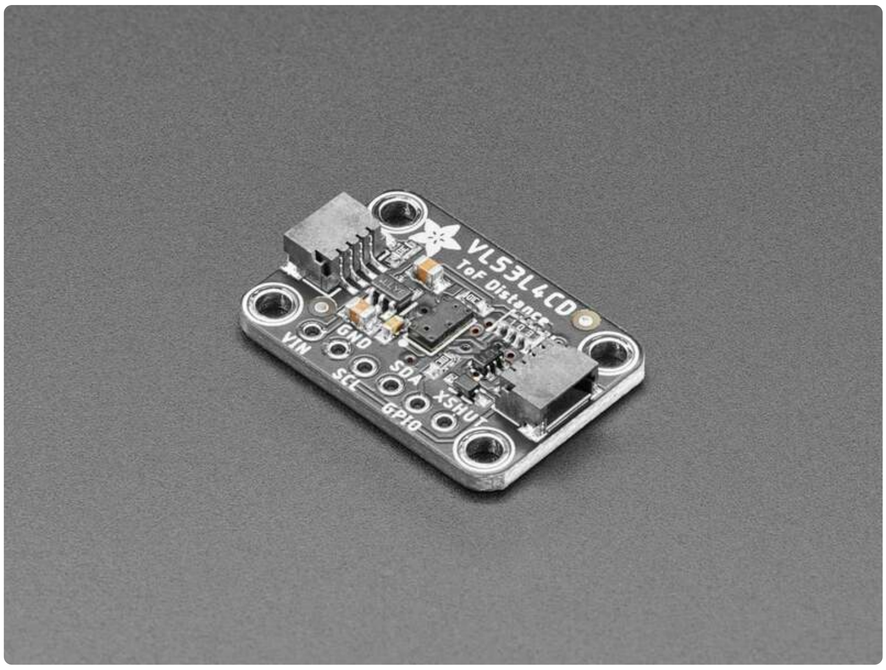
Overview



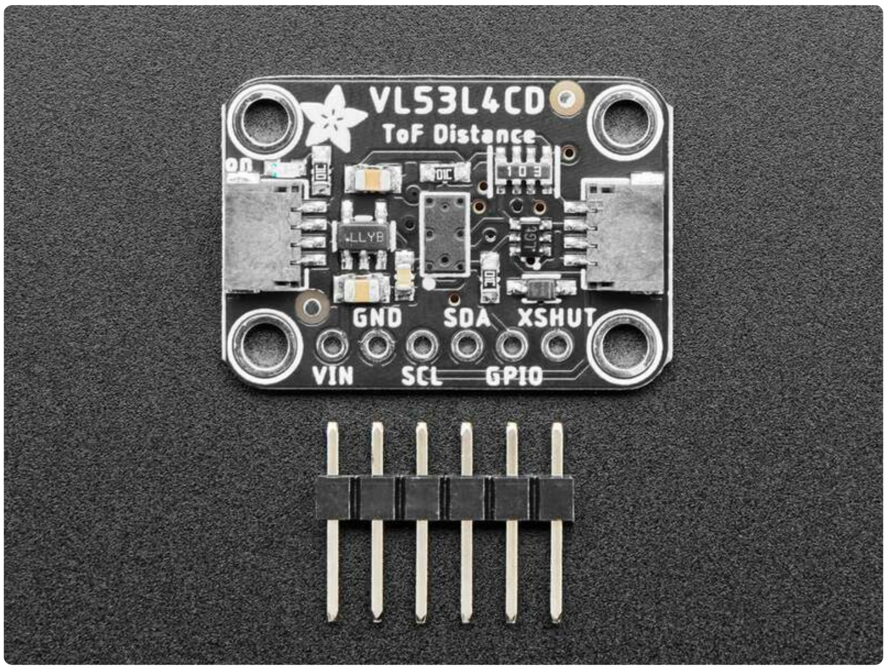
The **Adafruit VL53L4CD Time of Flight Sensor** is another great Time of Flight distance sensor from ST in the VL5 series of chips, this one is great for shorter distances. The sensor contains a very tiny invisible laser source and a matching sensor. The VL53L4CD can detect the "time of flight", or how long the light has taken to bounce back to the sensor. Since it uses a very narrow light source, it is good for determining distance of only the surface directly in front of it.

Unlike sonars that bounce ultrasonic waves, the 'cone' of sensing is very narrow. Unlike IR distance sensors which try to measure the amount of light bounced, the VL53 is much more precise and doesn't have linearity problems or 'double imaging' where you can't tell if an object is very far or very close.

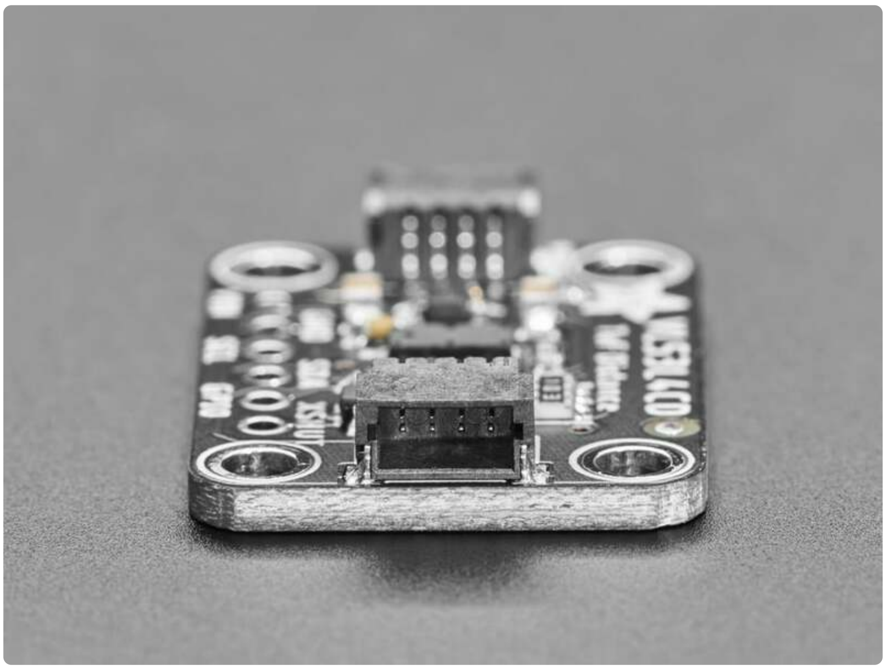
This breakout ships with a protector over the sensor. It must be removed before use! See details below.



This is another 'big sister' of the VL6180X ToF sensor (http://adafru.it/3316) and can handle about ~1 to 1300mm of range distance: basically it combines the short range of the VL6180X in that it can sense up to the body of the sensor, with the increased range of the VL53L0X (http://adafru.it/3317), which can handle about 50mm to 1200mm of range distance.



The sensor is small and easy to use in any robotics or interactive project. Since it needs 2.8V power and logic, we put the little fellow on a breakout board with a regulator and level shifting. You can use it with any 3-5V power or logic microcontroller with no worries. Works great with the **3.3V logic level of a Feather or Raspberry Pi, or the 5v level of a Metro 328 or Arduino Uno**. This breakout is ready to work with most common microcontrollers or SBCs. And since it speaks I2C, you can easily connect it up with two data wires plus power and ground.



As if that weren't enough, we've also added 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 micro with a plug-and-play cable to get ToF data ASAP.

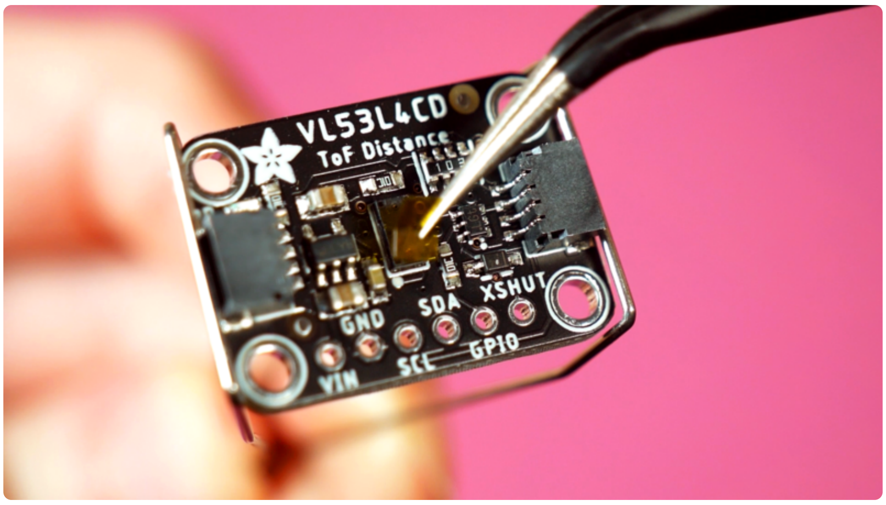
For a no-solder experience, just wire up to your favorite micro, like the STM32F405 Feather (http://adafru.it/4382) using a STEMMA QT adapter cable. (https://adafru.it/JnB) The Stemma QT connectors also mean the VL53L4CD can be used with our various associated accessories. (https://adafru.it/Ft6) QT Cable is not included, but we have a variety in the shop (https://adafru.it/17VE)

Communicating to the sensor is done over I2C with an API written by ST, they have an Arduino library with an example for communication here (https://adafru.it/Z3D).

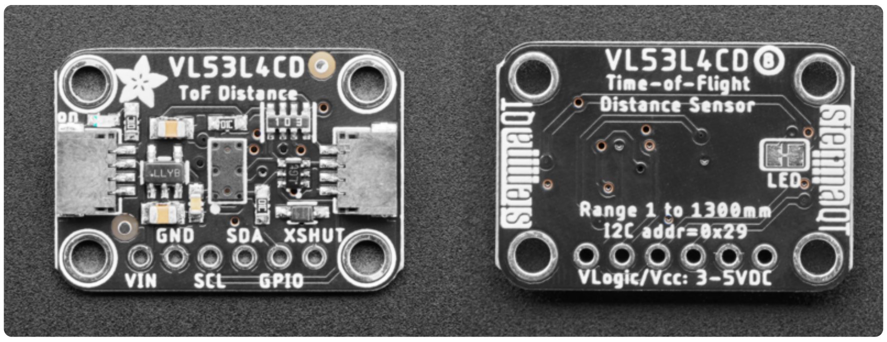
Removing the Protective Tape

Be careful when removing the tape! You don't want to damage the sensor.

Using tweezers (or some other appropriate tool), CAREFULLY remove the protective tape over the sensor, as seen in the image below. There is a small tab on the side of the tape that you can use to remove it.



Pinouts



This breakout ships with a protector over the sensor. It must be removed before use! See details at the bottom of the Overview page.

Power Pins

- **VIN** - This is the power pin. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 3V microcontroller like a Feather M4, use 3V, or for a 5V microcontroller like Arduino, use 5V.
- **GND** - This is common ground for power and logic.

I2C Logic Pins

The default I2C address for the VL53L4CD is **0x29**.

- **SCL** - I2C clock pin, connect to your microcontroller I2C clock line. There's a **10K pullup** on this pin.

- **SDA** - I2C data pin, connect to your microcontroller I2C data line. There's a **10K pullup** on this pin.
- **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/JRA).

Other Pins

- **GPIO** - This is the interrupt output pin, it is 2.8V logic level output - it can be read by 3.3V and most 5V logic microcontrollers.
- **XSHUT** - This is the shutdown pin. It is active low, and is logic-level shifted so you can use 3V or 5V logic.

LED Jumper

- **LED jumper** - This jumper is located on the back of the board. Cut the trace on this jumper to cut power to the "on" LED.

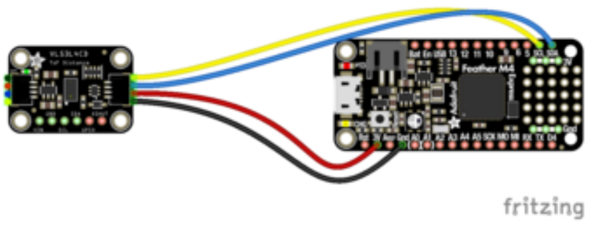
Python & CircuitPython

It's easy to use the **VL53L4CD** with Python or CircuitPython, and the Adafruit CircuitPython VL53L4CD (https://adafru.it/Z6d) module. This module allows you to easily write Python code that reads the distance from the **VL53L4CD** 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).

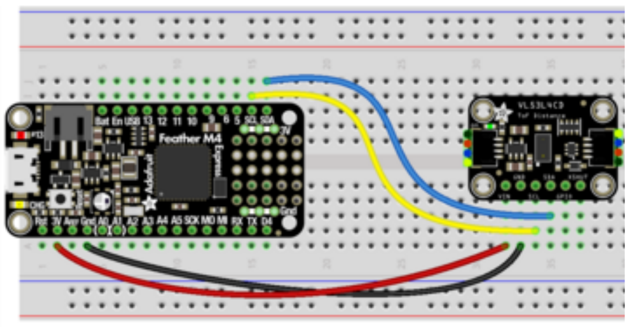
CircuitPython Microcontroller Wiring

First wire up a VL53L4CD to your board exactly as shown below. Here's an example of wiring a Feather M4 to the sensor 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 the standard **0.100" pitch** headers to wire it up on a breadboard:

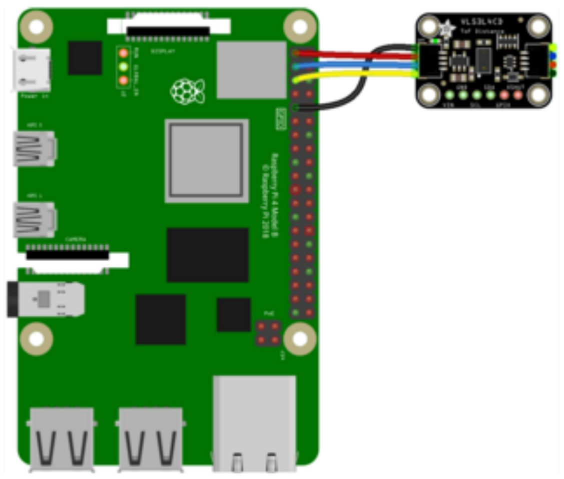


Board 3V to **sensor VIN (red wire)**
Board GND to **sensor GND (black wire)**
Board SCL to **sensor SCL (yellow wire)**
Board SDA to **sensor SDA (blue wire)**

Python Computer Wiring

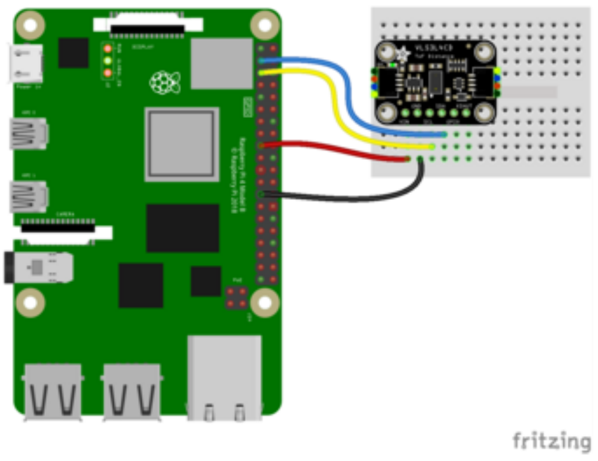
Since there's dozens of Linux computers/boards you can use, we will show wiring for Raspberry Pi. For other platforms, please visit the guide for CircuitPython on Linux to see whether your platform is supported (https://adafru.it/BSN).

Here's the Raspberry Pi wired to the sensor using I2C 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 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 VL53L4CD 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-vl53l4cd`

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!

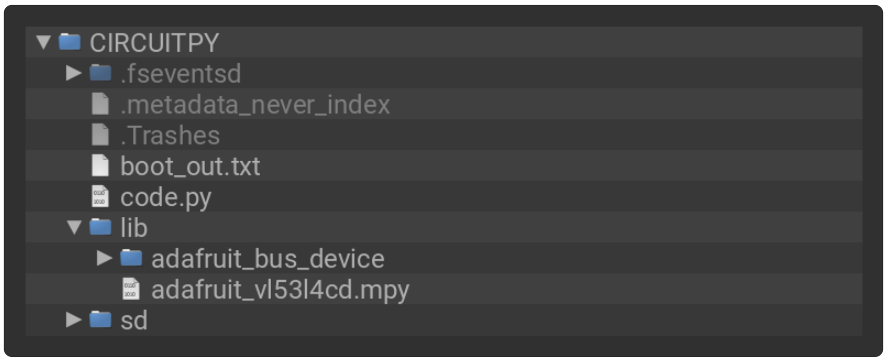
CircuitPython Usage

To use with CircuitPython, you need to first install the VL53L4CD 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 folder and file:

- **adafruit_bus_device/**
- **adafruit_vl53l4cd.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:

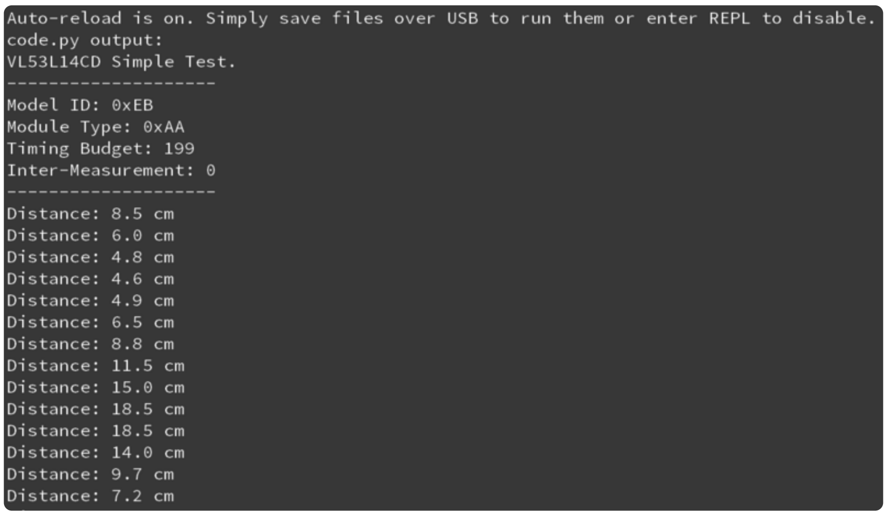
`python3 code.py`

Example Code

```
# SPDX-FileCopyrightText: 2017 Scott Shawcroft, written for Adafruit Industries
# SPDX-FileCopyrightText: Copyright (c) 2022 Carter Nelson for Adafruit Industries
#
# SPDX-License-Identifier: Unlicense
# SPDX-FileCopyrightText: 2017 Scott Shawcroft, written for Adafruit Industries
# SPDX-FileCopyrightText: Copyright (c) 2021 Carter Nelson for Adafruit Industries
#
# SPDX-License-Identifier: Unlicense
# Simple demo of the VL53L4CD distance sensor.
# Will print the sensed range/distance every second.

import board
import adafruit_vl53l4cd

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

vl53 = adafruit_vl53l4cd.VL53L4CD(i2c)

# OPTIONAL: can set non-default values
vl53.inter_measurement = 0
vl53.timing_budget = 200

print("VL53L4CD Simple Test.")
print("--------------------")
model_id, module_type = vl53.model_info
print("Model ID: 0x{:0X}".format(model_id))
print("Module Type: 0x{:0X}".format(module_type))
print("Timing Budget: {}".format(vl53.timing_budget))
print("Inter-Measurement: {}".format(vl53.inter_measurement))
print("--------------------")

vl53.start_ranging()

while True:
    while not vl53.data_ready:
        pass
    vl53.clear_interrupt()
    print("Distance: {} cm".format(vl53.distance))
```

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.



```
Auto-reload is on. Simply save files over USB to run them or enter REPL to disable.
code.py output:
VL53L14CD Simple Test.
--------------------
Model ID: 0xEB
Module Type: 0xAA
Timing Budget: 199
Inter-Measurement: 0
--------------------
Distance: 8.5 cm
Distance: 6.0 cm
Distance: 4.8 cm
Distance: 4.6 cm
Distance: 4.9 cm
Distance: 6.5 cm
Distance: 8.8 cm
Distance: 11.5 cm
Distance: 15.0 cm
Distance: 18.5 cm
Distance: 18.5 cm
Distance: 14.0 cm
Distance: 9.7 cm
Distance: 7.2 cm
```

Now try holding your hand in front of the sensor, and moving it closer and further away to see the values change!

First you import the necessary modules and libraries. Then you instantiate the sensor on I2C.

Then you're ready to read data from the sensor, including the initial information printed to the serial console.

Finally, inside the loop, you check the distance every second.

That's all there is to using the VL53L4CD with CircuitPython!

Python Docs

Python Docs (https://adafru.it/Z3E)

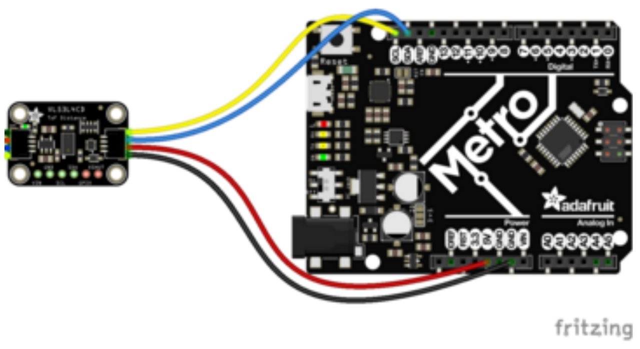
Arduino

Using the VL53L4CD with Arduino involves wiring up the sensor to your Arduino-compatible microcontroller, installing the STM32duino VL53L4CD (https://adafru.it/Z3D) library and running the provided example code.

Wiring

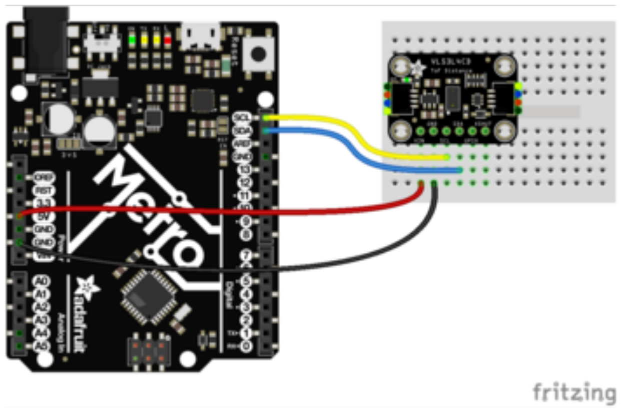
Wire as shown for a **5V** board like an Uno. If you are using a **3V** board, like an Adafruit Feather, wire the board's 3V pin to the VL53L4CD VIN.

Here is an Adafruit Metro wired up to the VL53L4CD 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)**

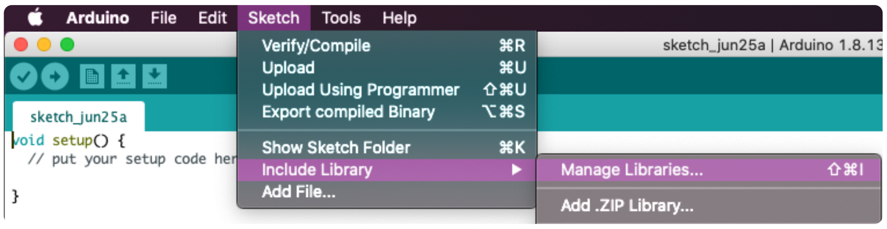
Here is an Adafruit Metro wired up using a solderless breadboard:



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)**

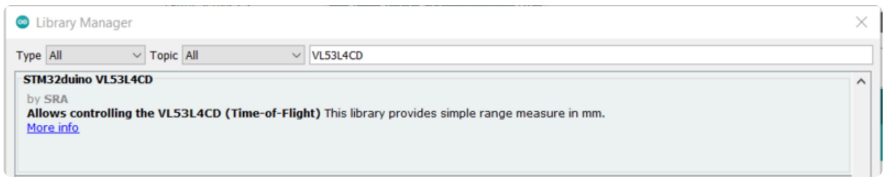
Library Installation

You can install the **VL53L4CD** library for Arduino using the Library Manager in the Arduino IDE.



```
void setup() {
  // put your setup code her
}
```

Click the **Manage Libraries ...** menu item, search for **VL53L4CD**, and select the **STM32duino VL53L4CD** library:



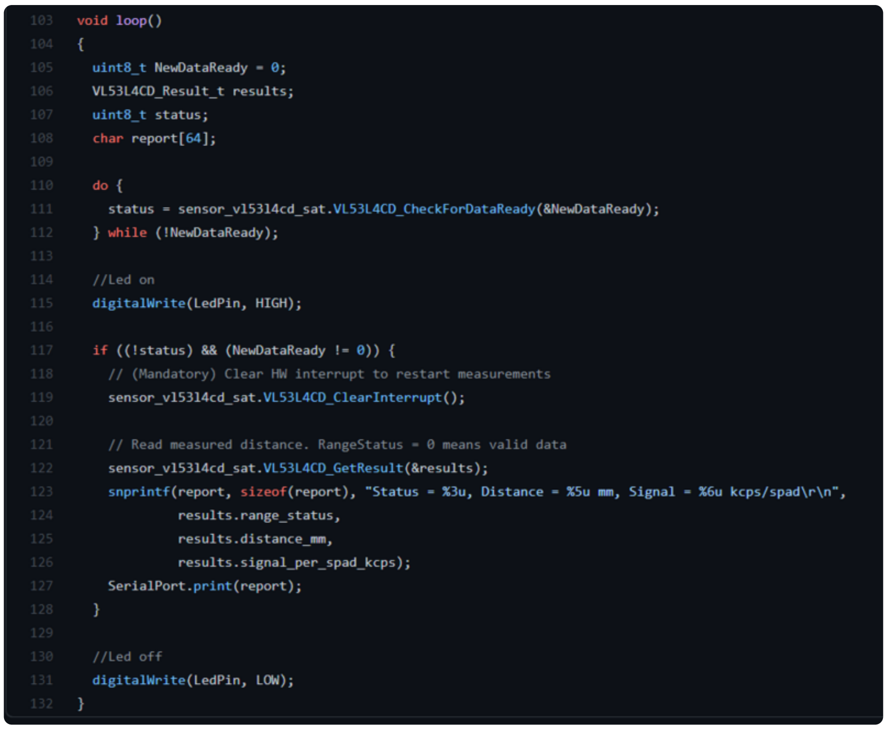
VL53L4CD

STM32duino VL53L4CD
by **SRA**
Allows controlling the VL53L4CD (Time-of-Flight) This library provides simple range measure in mm.
More info

Load Example

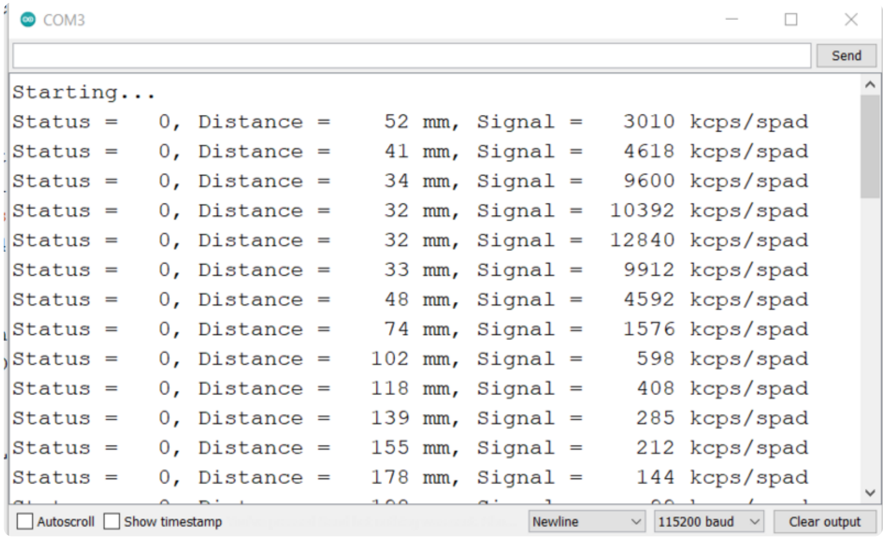
Open up **File -> Examples -> STM32duino VL53L4CD -> VL53L4CD_Sat_HelloWorld** and upload to your Arduino wired to the sensor.

Note: this is only the loop in the screenshot.



```
void loop()
{
  uint8_t NewDataReady = 0;
  VL53L4CD_Result_t results;
  uint8_t status;
  char report[64];

  do {
    status = sensor_vl53l4cd_sat.VL53L4CD_CheckForDataReady(&NewDataReady);
  } while (!NewDataReady);

  //Led on
  digitalWrite(LedPin, HIGH);

  if ((!status) && (NewDataReady != 0)) {
    // (Mandatory) Clear HW interrupt to restart measurements
    sensor_vl53l4cd_sat.VL53L4CD_ClearInterrupt();

    // Read measured distance. RangeStatus = 0 means valid data
    sensor_vl53l4cd_sat.VL53L4CD_GetResult(&results);
    snprintf(report, sizeof(report), "Status = %3u, Distance = %5u mm, Signal = %6u kcps/spad\r\n",
             results.range_status,
             results.distance_mm,
             results.signal_per_spad_kcps);
    SerialPort.print(report);
  }

  //Led off
  digitalWrite(LedPin, LOW);
}
```

Upload the sketch to your board and open up the Serial Monitor (**Tools -> Serial Monitor**) at 115200 baud. You should see the values from the sensor being printed out.

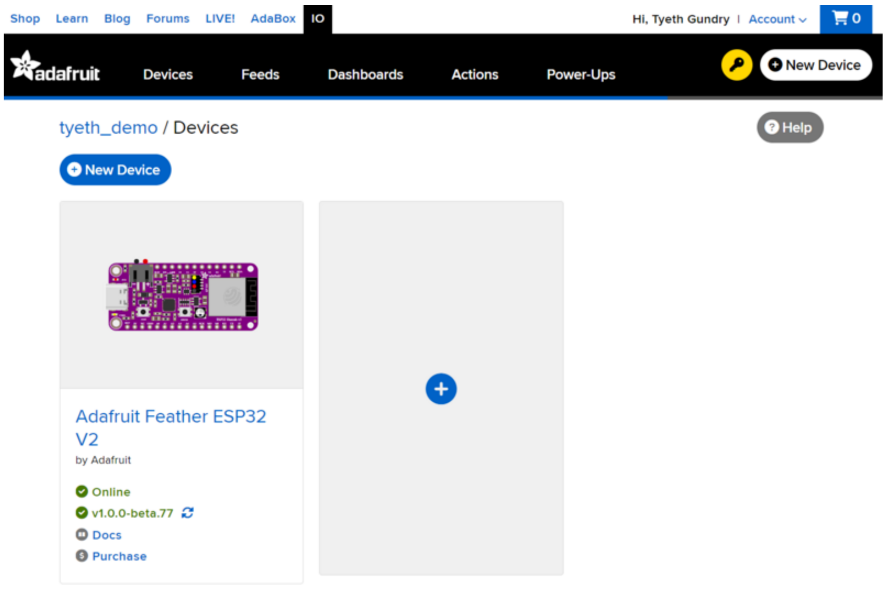


```
Starting...
Status =   0, Distance =    52 mm, Signal =   3010 kcps/spad
Status =   0, Distance =    41 mm, Signal =   4618 kcps/spad
Status =   0, Distance =    34 mm, Signal =   9600 kcps/spad
Status =   0, Distance =    32 mm, Signal =  10392 kcps/spad
Status =   0, Distance =    32 mm, Signal =  12840 kcps/spad
Status =   0, Distance =    33 mm, Signal =   9912 kcps/spad
Status =   0, Distance =    48 mm, Signal =   4592 kcps/spad
Status =   0, Distance =    74 mm, Signal =   1576 kcps/spad
Status =   0, Distance =   102 mm, Signal =    598 kcps/spad
Status =   0, Distance =   118 mm, Signal =    408 kcps/spad
Status =   0, Distance =   139 mm, Signal =    285 kcps/spad
Status =   0, Distance =   155 mm, Signal =    212 kcps/spad
Status =   0, Distance =   178 mm, Signal =    144 kcps/spad
```

Arduino Docs

Arduino Docs (https://adafru.it/Z3D)

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.

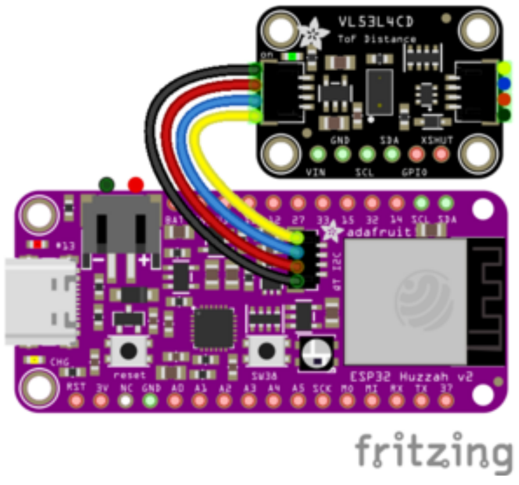
If you've never used WipperSnapper, click below to read through the quick start guide before continuing.

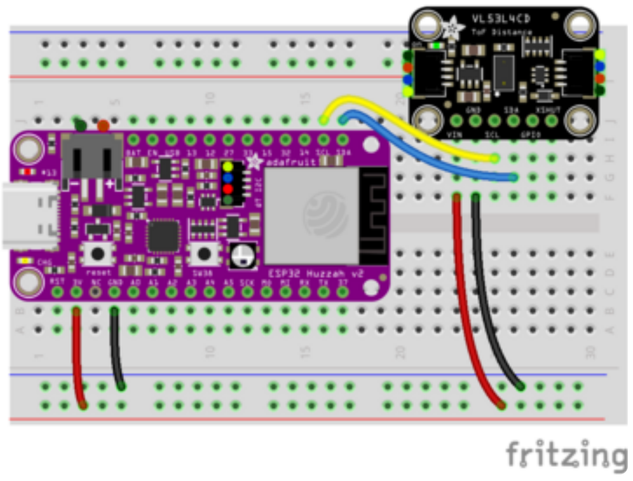
Quickstart: Adafruit IO WipperSnapper

https://adafru.it/Vfd

Wiring

First, wire up an VL53L4CD to your board exactly as follows. Here is an example of the VL53L4CD 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)



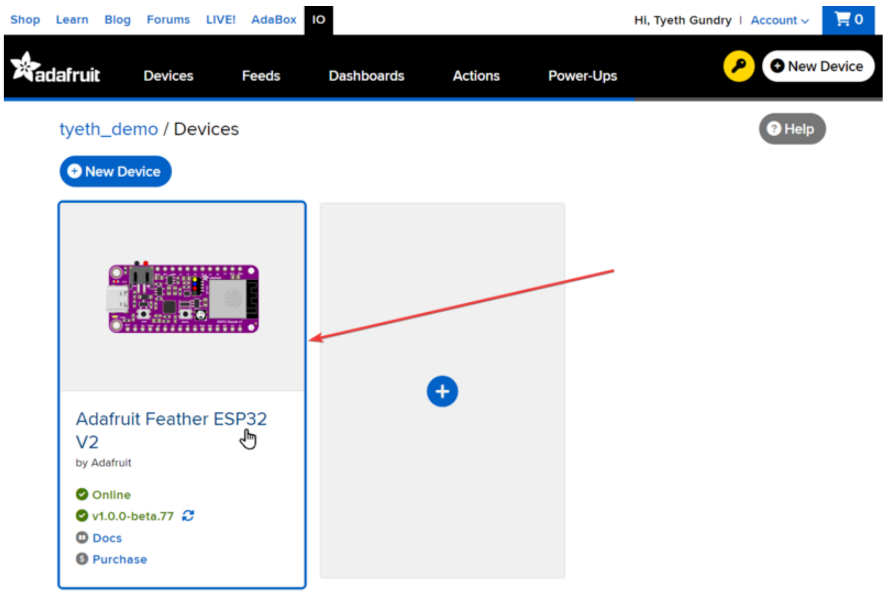


Board 3V to **sensor VIN (red wire on STEMMA QT)**
Board GND to **sensor GND (black wire on STEMMA QT)**
Board SCL to **sensor SCL (yellow wire on STEMMA QT)**
Board SDA to **sensor SDA (blue wire on STEMMA QT)**

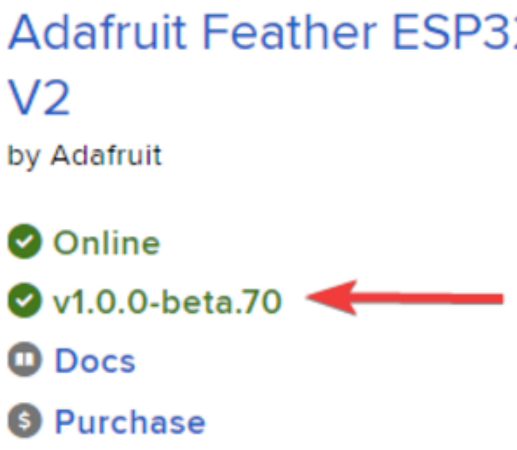
Usage

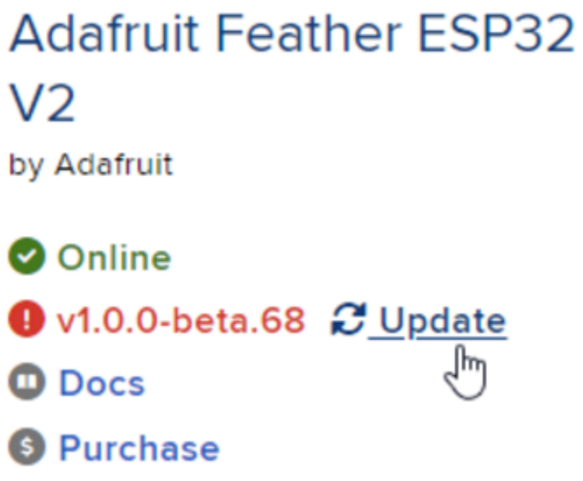
Connect your board to Adafruit IO Wippersnapper and navigate to the WipperSnapper board list **(https://adafru.it/TAu).**

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.



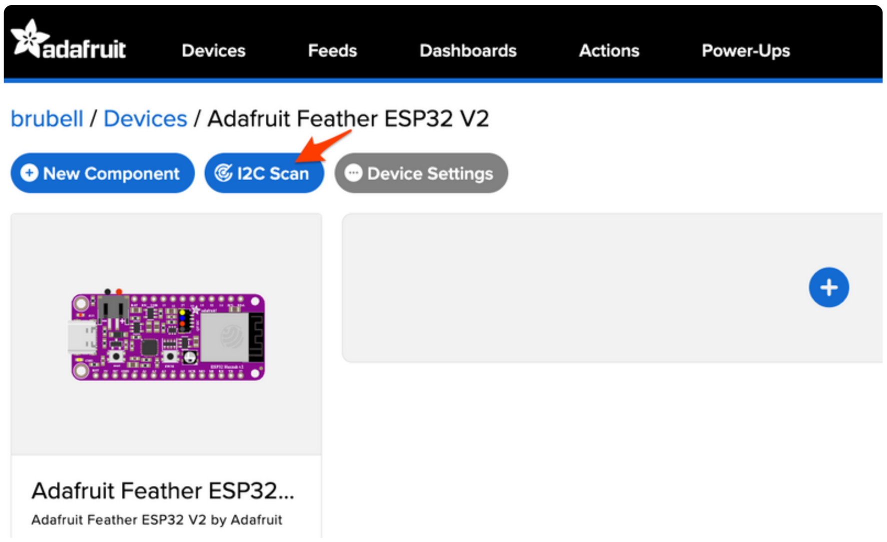


On the device page, quickly **check that you're running the latest version of the WipperSnapper firmware**.

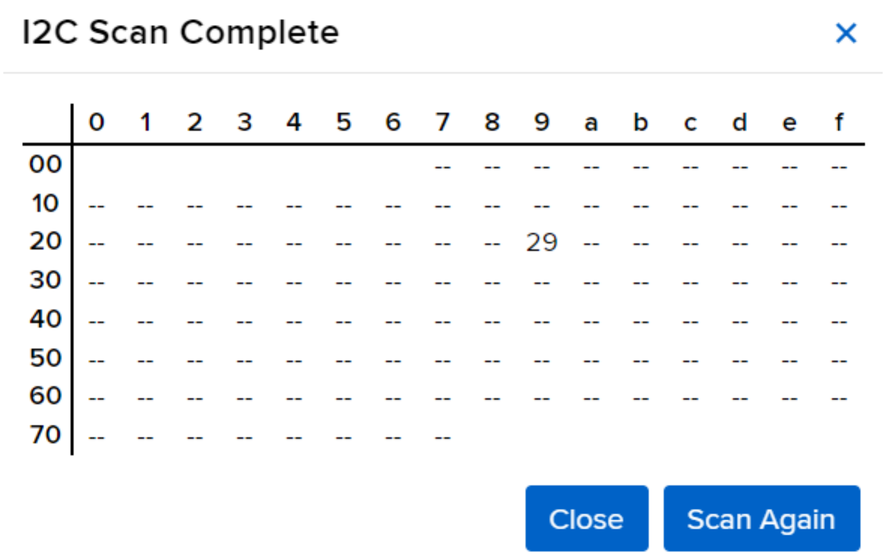
The device tile on the left indicates the version number of the firmware running on the connected board.

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.


Devices
Feeds
Dashboards
Actions
Power-Ups
brubell / Devices / Adafruit Feather ESP32 V2
New Component
I2C Scan
Device Settings
Adafruit Feather ESP32...
Adafruit Feather ESP32 V2 by Adafruit

You should see the VL53L4CD's default I2C address of `0x29` 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	--	--	--	--	--	--	--	--	--	29	--	--	--	--	--	--
30	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
40	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
50	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
60	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--
70	--	--	--	--	--	--	--	--								

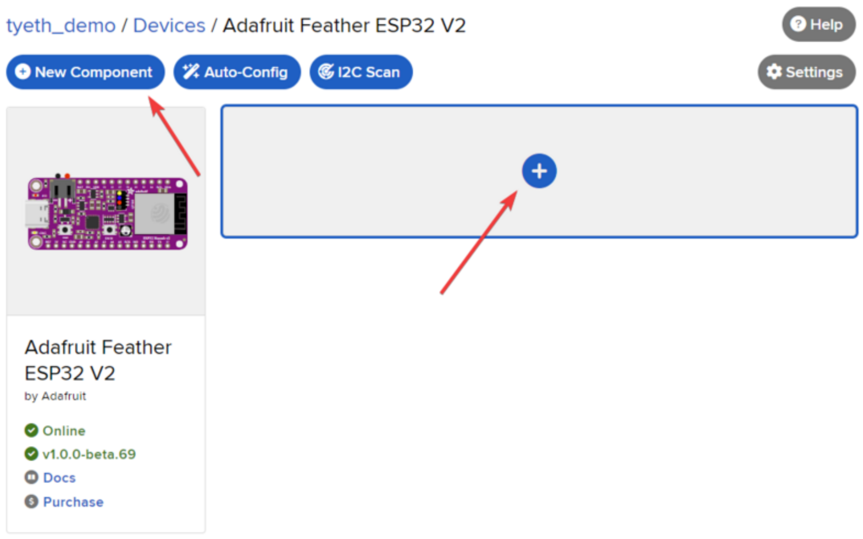
I don't see the sensor's I2C address listed!

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

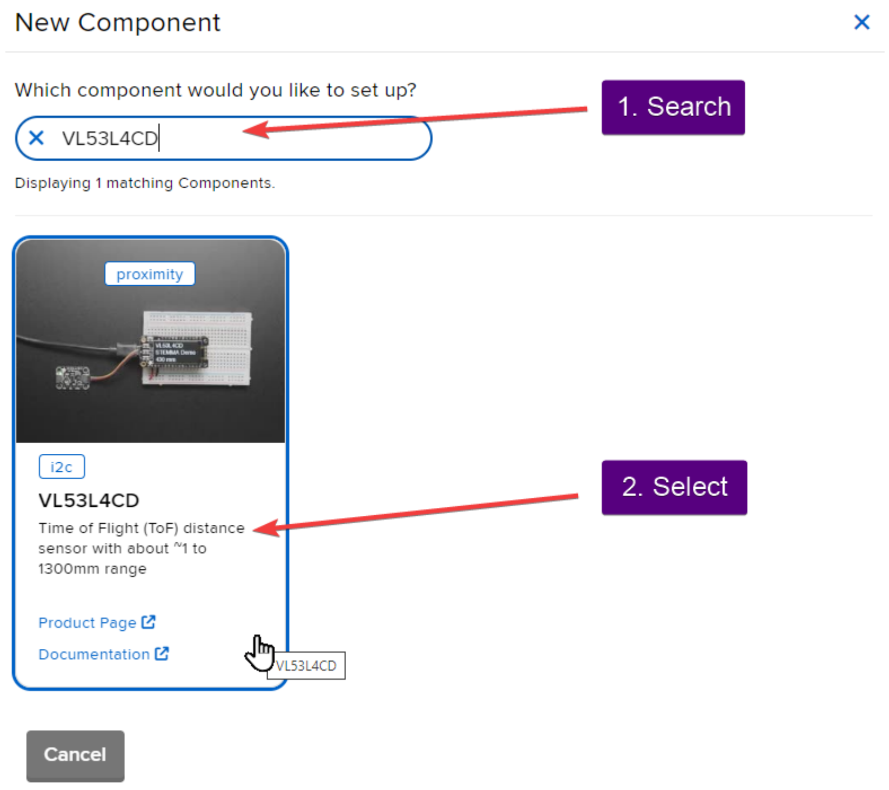
Then, reset the board and let it re-connect to Adafruit IO WipperSnapper.

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.



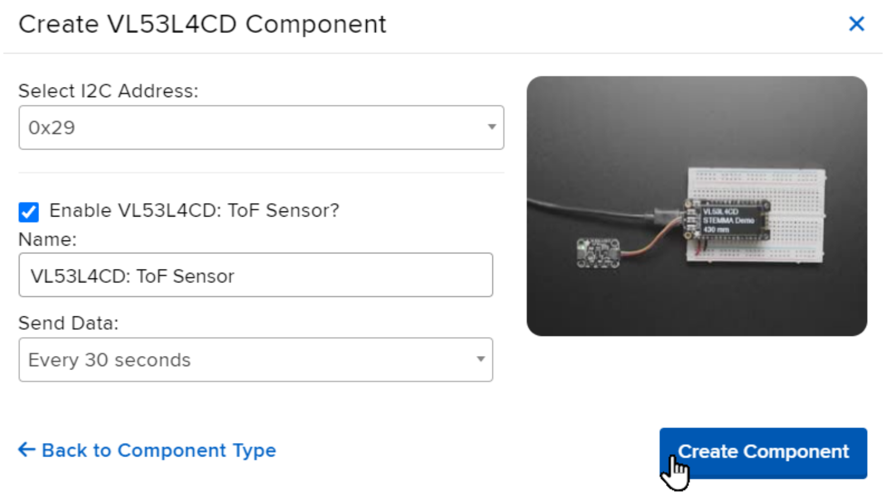
Adafruit IO supports a large amount of components. To quickly find your sensor, type `VL53L4CD` into the search bar, then select the **VL53L4CD** component.


New Component
Which component would you like to set up?
VL53L4CD
Displaying 1 matching Components.
1. Search
proximity
i2c
VL53L4CD
Time of Flight (ToF) distance sensor with about ~1 to 1300mm range
Product Page
Documentation
2. Select
VL53L4CD
Cancel

On the component configuration page, the VL53L4CD'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 VL53L4CD 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 VL53L4CD Component

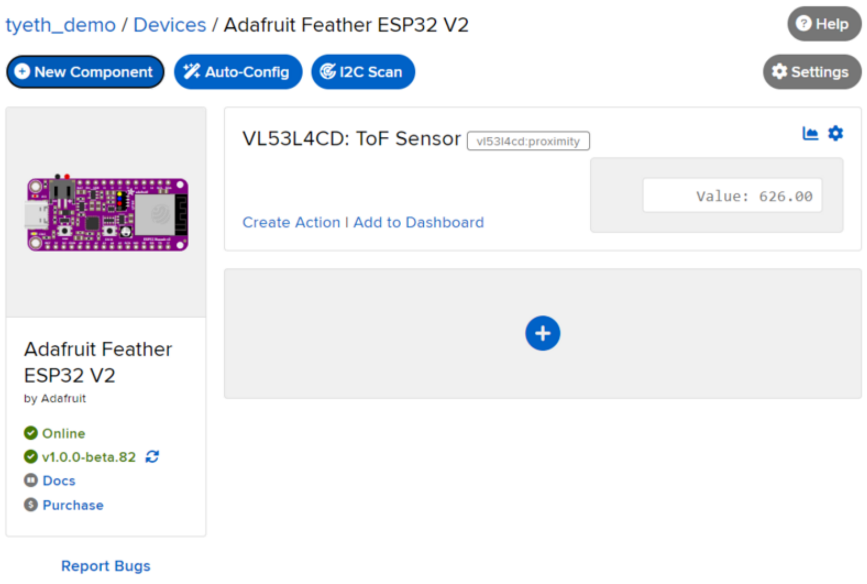
Select I2C Address: 0x29

✓ Enable VL53L4CD: ToF Sensor?

Name: VL53L4CD: ToF Sensor

Send Data: 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.


tyeth_demo / Devices / Adafruit Feather ESP32 V2

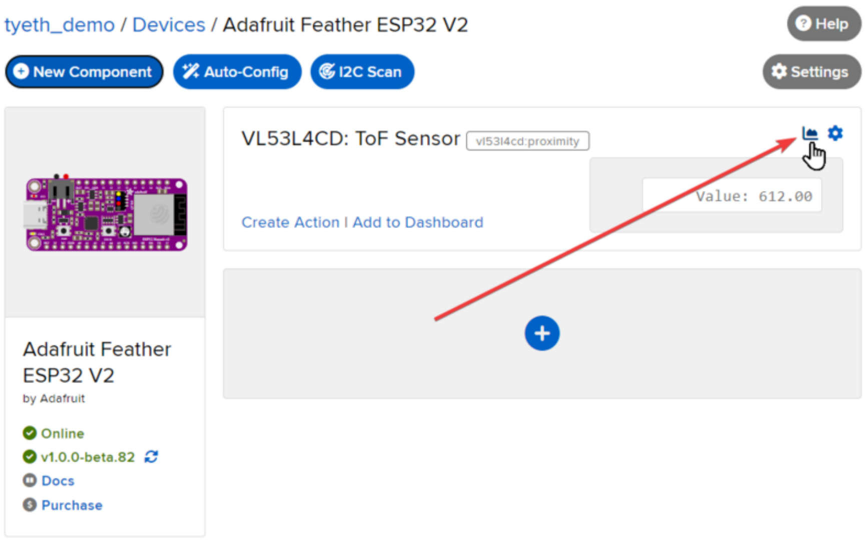
VL53L4CD: ToF Sensor vl53l4cd:proximity

Value: 626.00

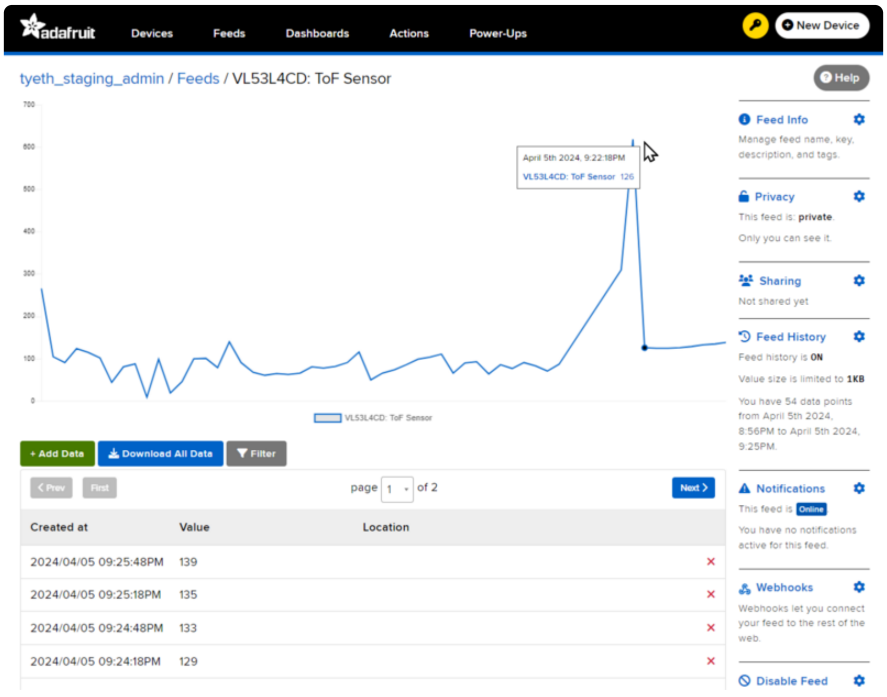
Adafruit Feather ESP32 V2
by Adafruit

Online
v1.0.0-beta.82

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).



Manage feed name, key, description, and tags.

This feed is: **private**.

Only you can see it.

Not shared yet

Feed history is **ON**

Value size is limited to **1KB**

You have 54 data points from April 5th 2024, 8:56PM to April 5th 2024, 9:25PM.

This feed is Online.

You have no notifications active for this feed.

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

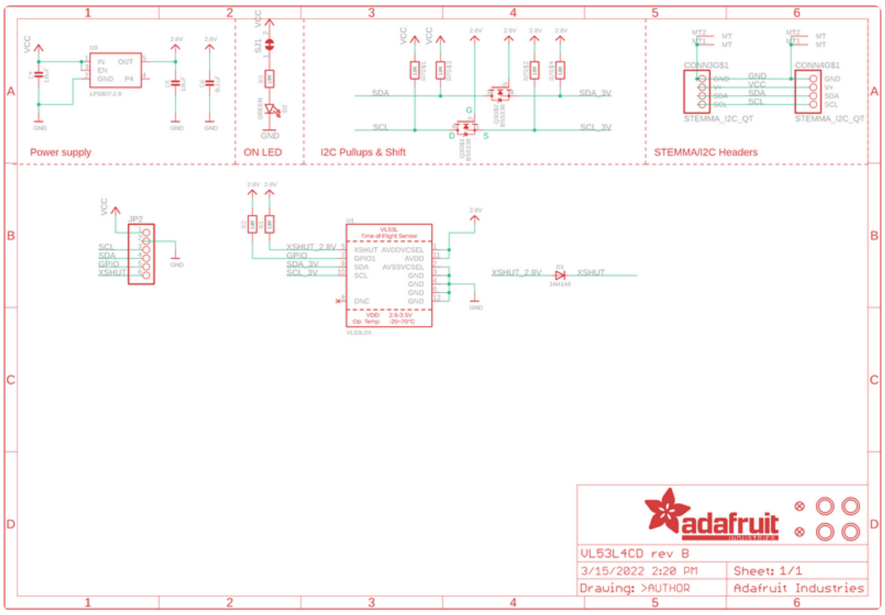
Created at	Value	Location
2024/04/05 09:25:48PM	139	
2024/04/05 09:25:18PM	135	
2024/04/05 09:24:48PM	133	
2024/04/05 09:24:18PM	129	

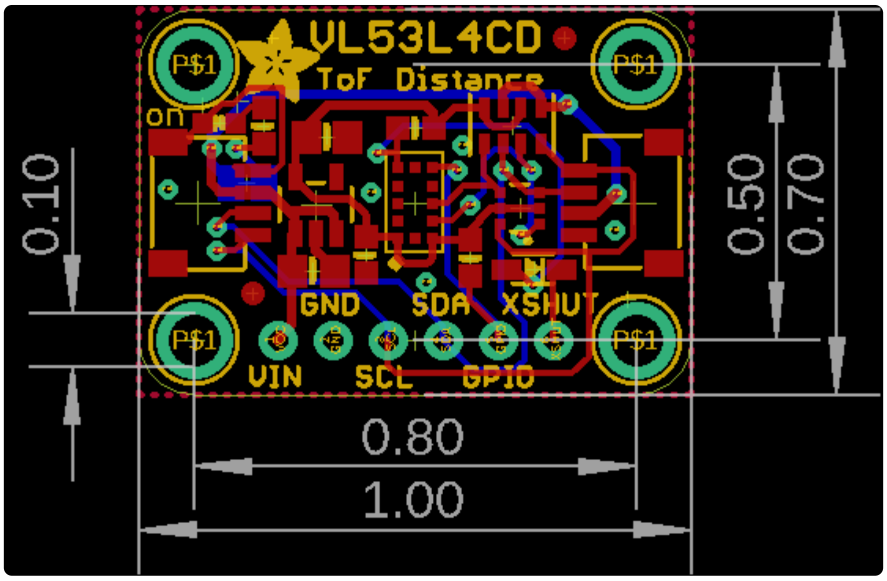
Downloads

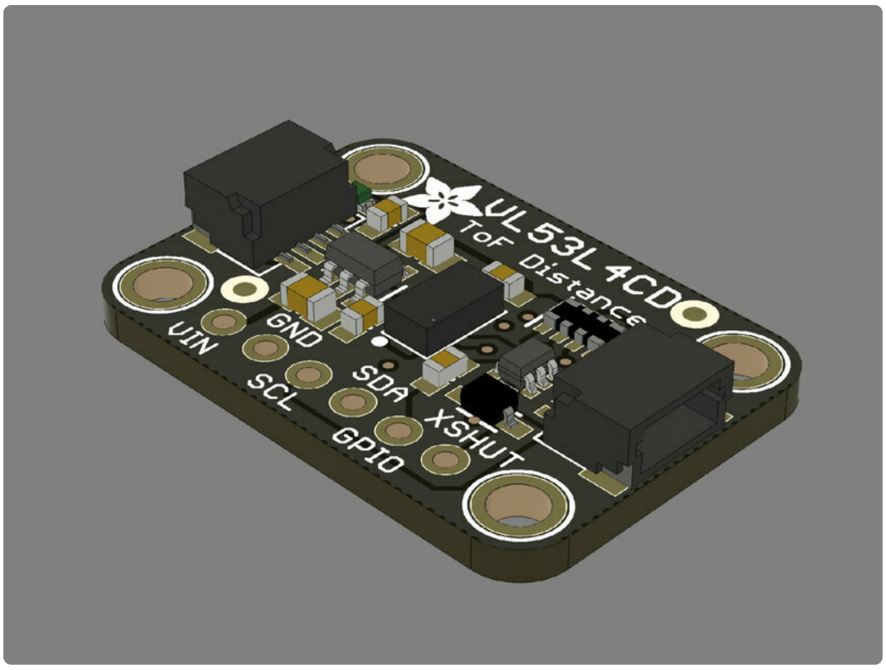
Files

- VL53L4CD Datasheet (https://adafru.it/Z6a)
- EagleCAD PCB files on GitHub (https://adafru.it/Z6b)
- 3D Models on GitHub (https://adafru.it/ZEd)
- Fritzing object in the Adafruit Fritzing Library (https://adafru.it/Z6c)

Schematic and Fab Print






VL53L4CD
ToF Distance
VIN
GND
SCL
SDA
GPIO
XSHUT