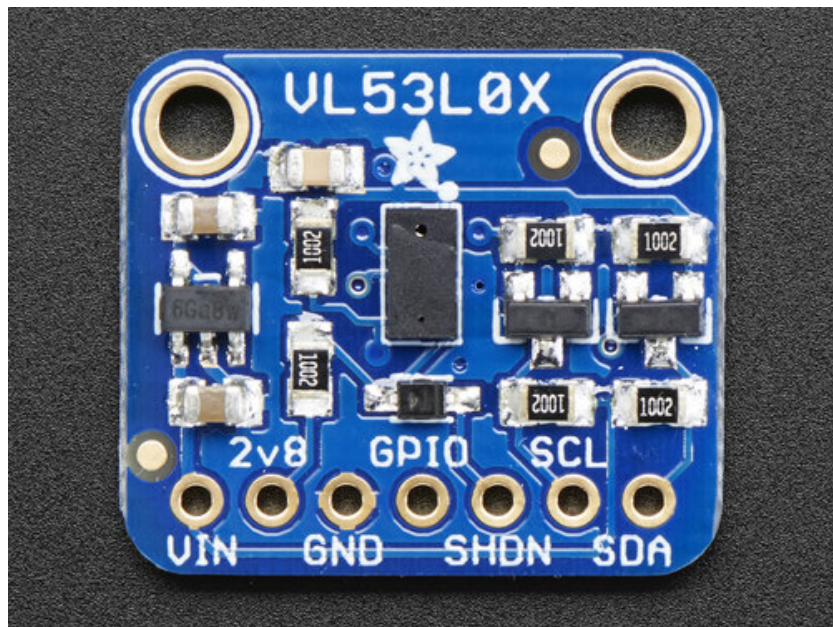


Adafruit VL53L0X Time of Flight Micro-LIDAR Distance Sensor Breakout

Created by lady ada

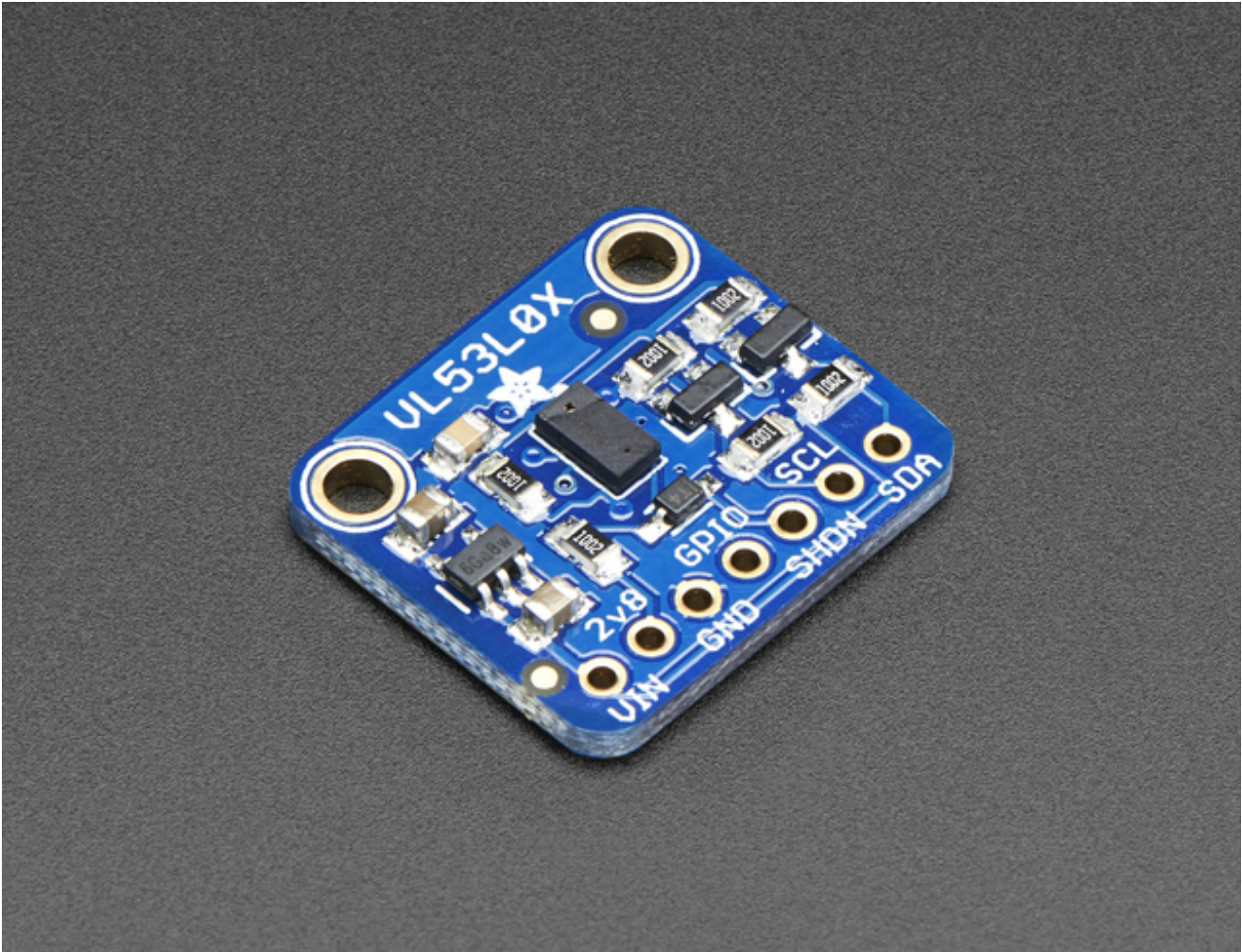


Last updated on 2017-08-18 05:33:43 PM UTC

Guide Contents

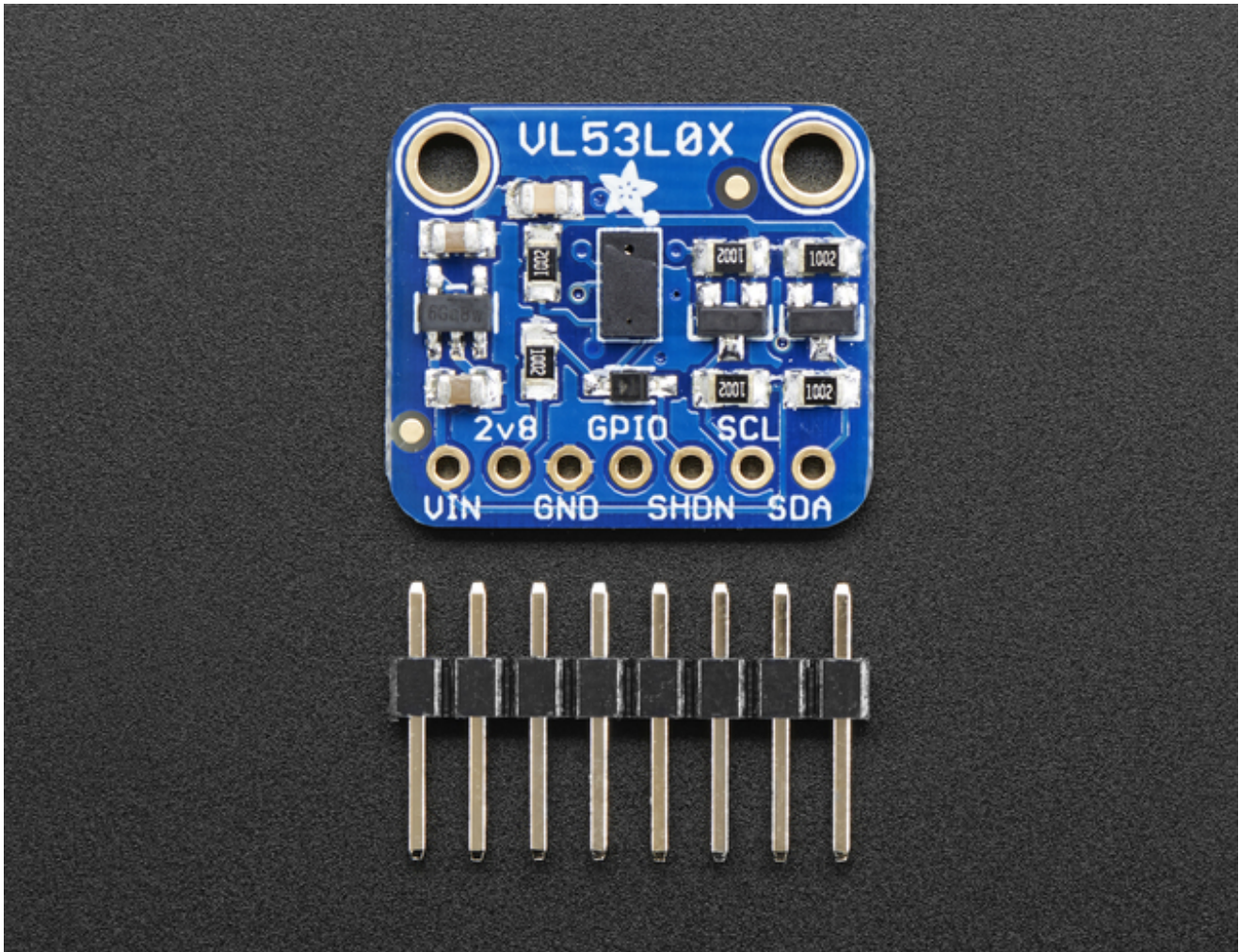
Guide Contents	2
Overview	3
Sensing Capabilities	5
Pinouts	9
Power Pins:	9
I2C Logic pins:	10
Control Pins:	10
Assembly	11
Prepare the header strip:	11
Solder!	13
Wiring & Test	15
Download Adafruit_VL53L0X	16
Load Demo	16
Connecting Multiple Sensors	18
Downloads	20
Files & Datasheets	20
Schematic & Fabrication Print	20

Overview

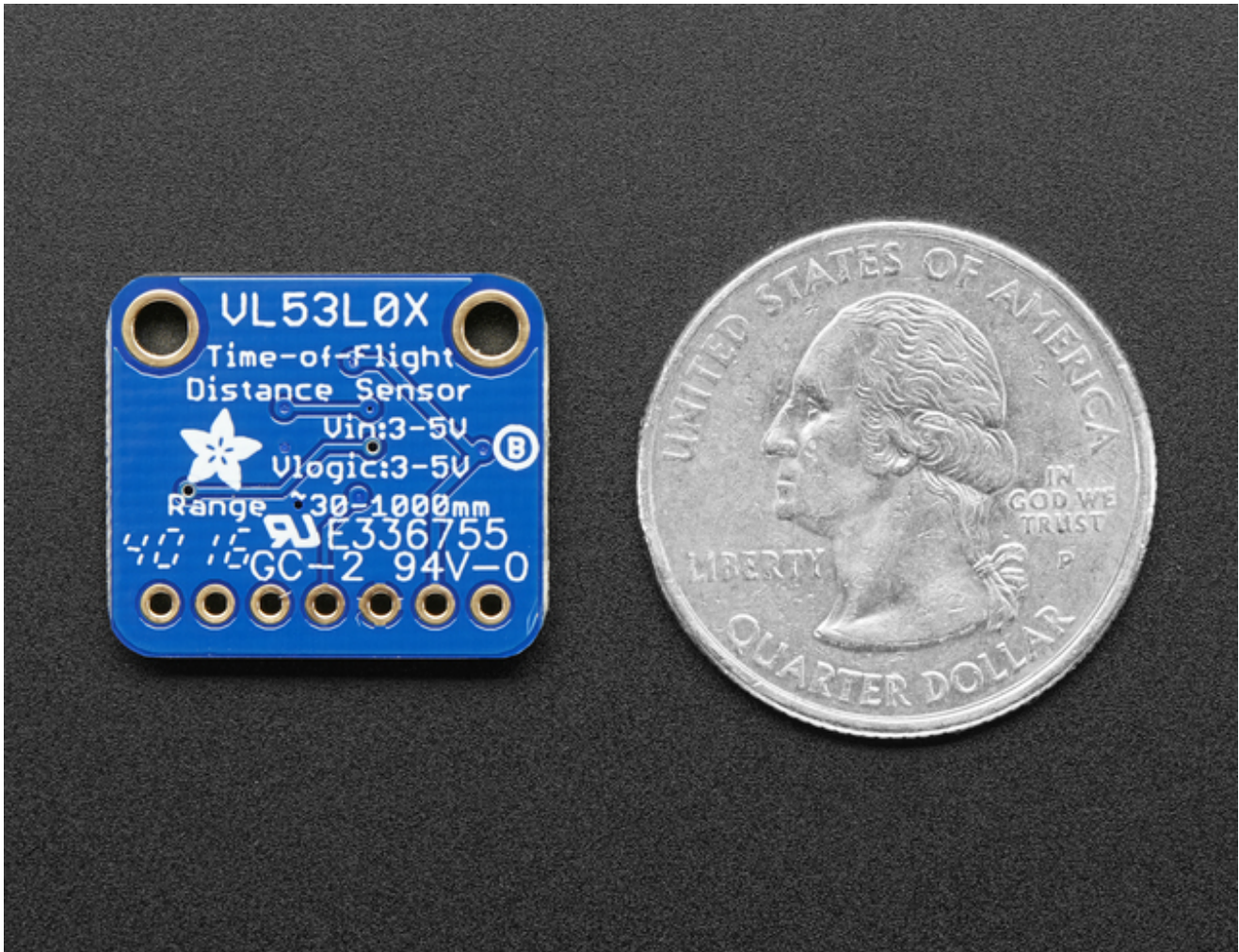


The VL53L0X is a *Time of Flight* distance sensor like no other you've used! The sensor contains a very tiny invisible laser source, and a matching sensor. The VL53L0X can detect the "time of flight", or how long the light has taken to bounce back to the sensor. Since it uses a very narrow light source, 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 that try to measure the amount of light bounced, the VL53L0x 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 is the 'big sister' of the VL6180X ToF sensor, and can handle about 50 - 1200 mm of range distance. If you need a smaller/closer range, check out the VL6180X which can measure 5mm to 200mm.



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. Each order comes with a small piece of header. Solder the header onto your breakout board with your iron and some solder and wire it up for instant distance-sensing-success!

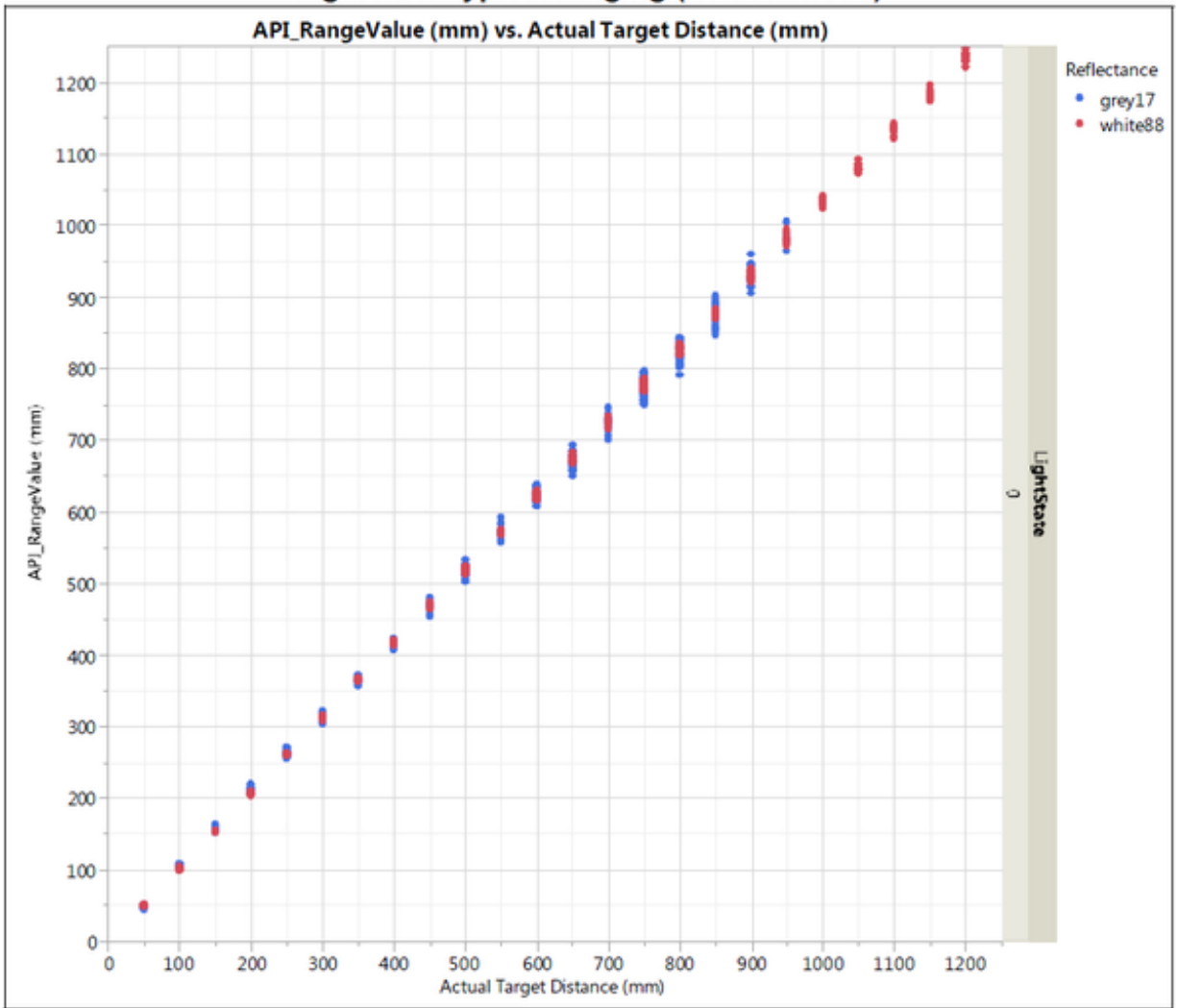


Communicating to the sensor is done over I2C with an API written by ST, so its not too hard to port it to your favorite microcontroller. We've written a wrapper library for Arduino so you can use it with any of your Arduino-compatible boards.

Sensing Capablities

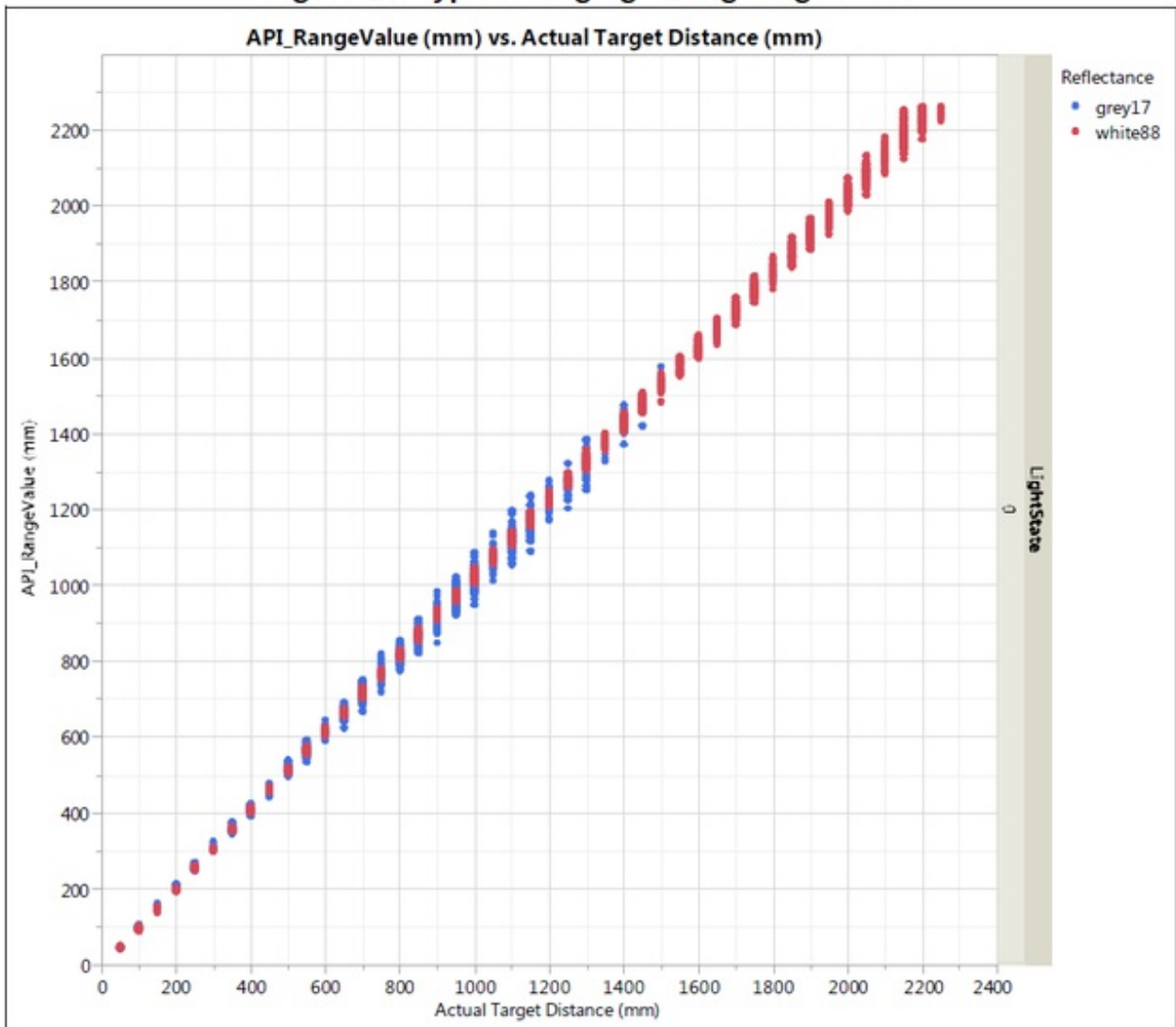
The sensor can measure approximately 50mm to 1.2 meter in default mode.

Figure 20. Typical ranging (default mode)



In 'long range' mode you can detect as far as 1.5 to 2 meters on a nice white reflective surface.

Figure 21. Typical ranging - Long range mode



Depending on ambient lighting and distance, you'll get 3 to 12% ranging accuracy - better lighting and shiny surfaces will give you best results. Some experimentation will be necessary since if the object absorbs the laser light you won't get good readings.

Table 11. Max ranging capabilities with 33ms timing budget

Target reflectance level (Full FOV)	Conditions	Indoor (2)	Outdoor overcast (2)
White Target (88%)	Typical	200cm+ (1)	80cm
	Minimum	120cm	60cm
Grey Target (17%)	Typical	80cm	50cm
	Minimum	70cm	40cm

Note (1): using long range API profile

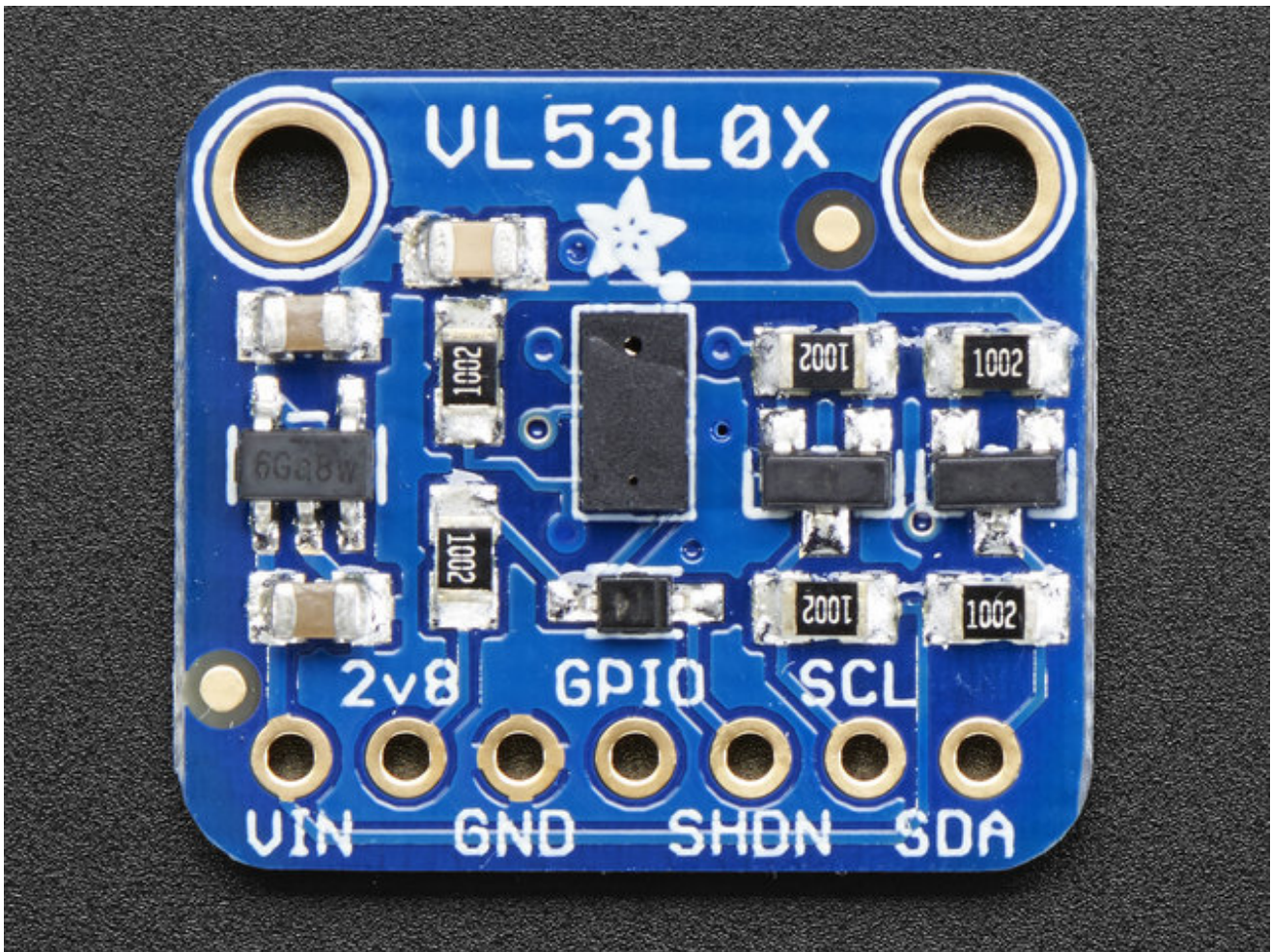
Table 12. Ranging accuracy

Target reflectance level (Full FOV)	Indoor (no infrared)			Outdoor		
	Distance	33ms	66ms	Distance	33ms	66ms
White Target (88%)	at 120cm	4%	3%	at 60cm	7%	6%
Grey Target (17%)	at 70cm	7%	6%	at 40cm	12%	9%

Pinouts

The VL53L0X 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 **0x29**. You *can* change it, but only in software. That means you have to wire the SHUTDOWN pin and hold all but one sensor in reset while you reconfigure one sensor at a time



Power Pins:

- **Vin** - this is the power pin. Since the chip uses 2.8 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

- **2v8** - this is the 2.8V output from the voltage regulator, you can grab up to 100mA from this if you like
- **GND** - common ground for power and logic

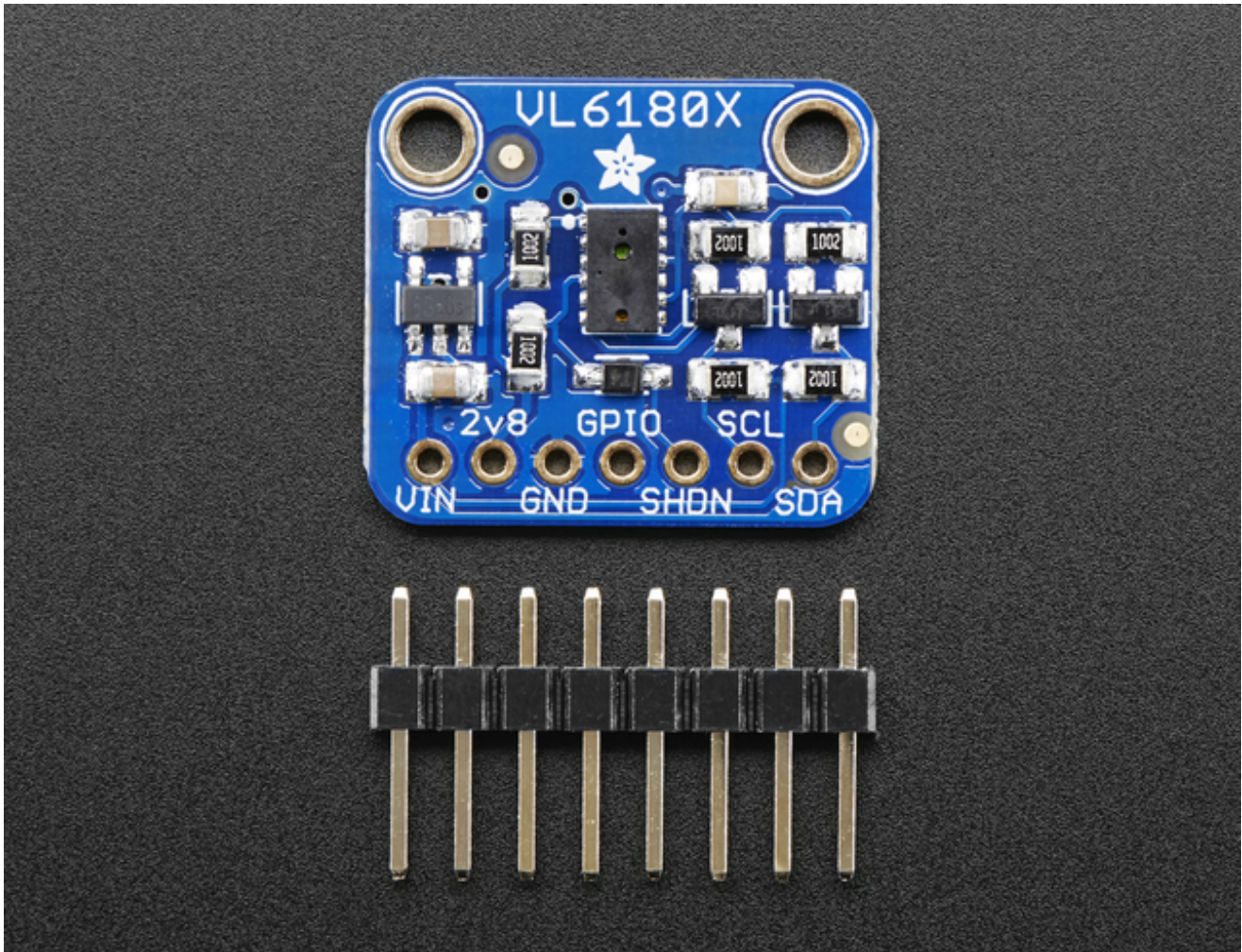
I2C Logic pins:

- **SCL** - I2C clock pin, connect to your microcontrollers I2C clock line.
- **SDA** - I2C data pin, connect to your microcontrollers I2C data line.

Control Pins:

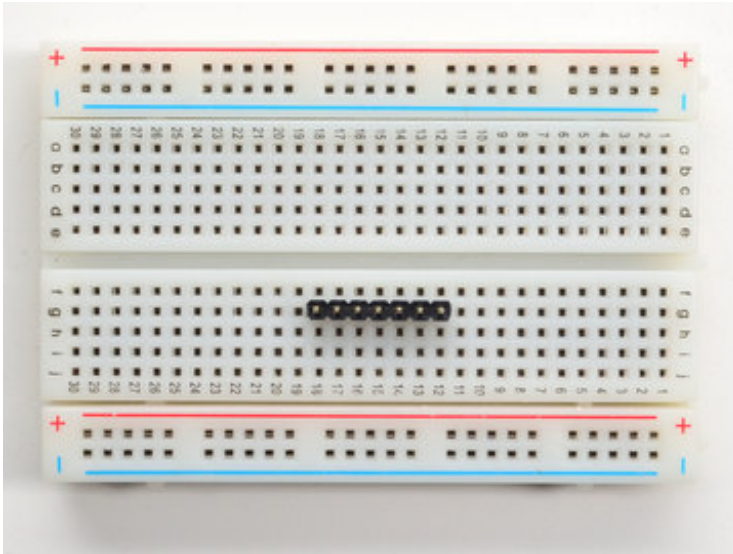
- **GPIO** - this is a pin that is used by the sensor to indicate that data is ready. It's useful for when doing continuous sensing. Note there is no level shifting on this pin, you may not be able to read the 2.8V-logic-level voltage on a 5V microcontroller (we could on an arduino UNO but no promises). Our library doesn't make use of this pin but for advanced users, it's there!
- **SHDN** - the shutdown pin for the sensor. By default it's pulled high. There's a level-shifting diode so you can use 3-5V logic on this pin. When the pin is pulled low, the sensor goes into shutdown mode.

Assembly



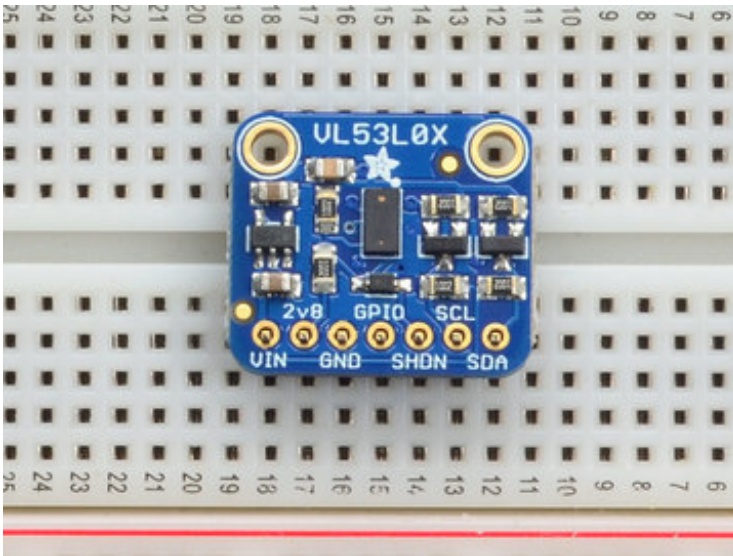
This page shows the VL53L0X or VL6180X sensor - the procedure is identical!

Prepare the header strip:



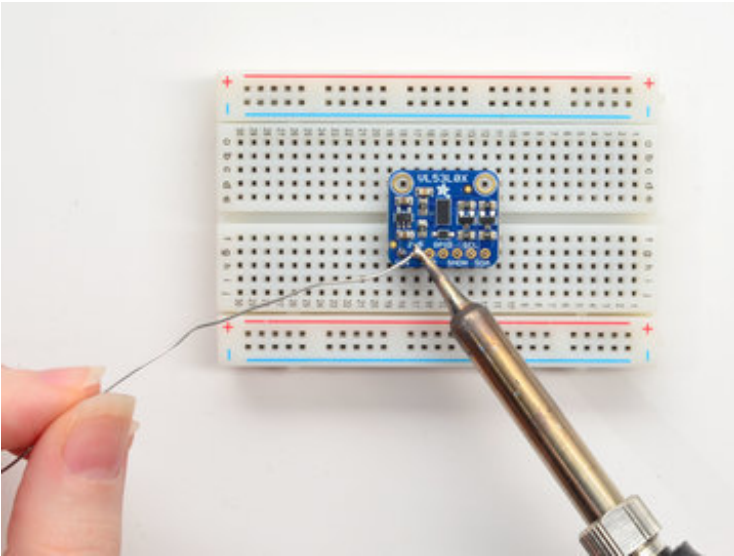
Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - **long pins down**

•

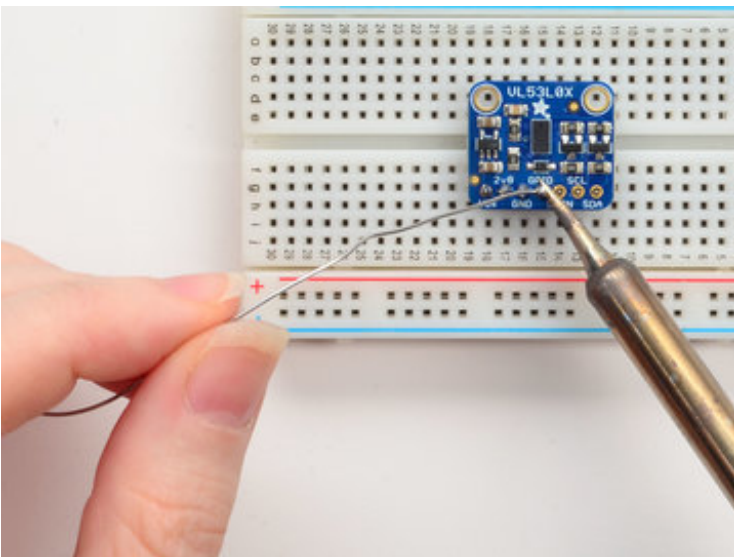


Place the Breakout board on top so the shorter ends of the pins line up though all the pads

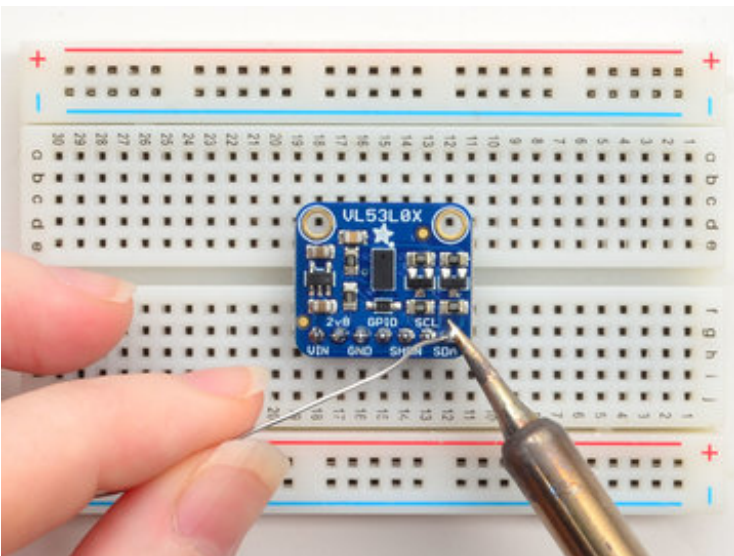
•



•



•

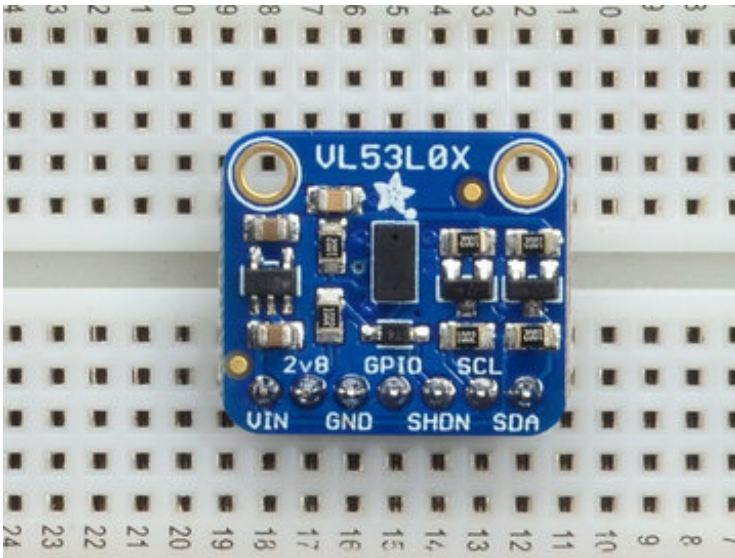


•

Solder!

Be sure to solder all pins for reliable electrical contact.

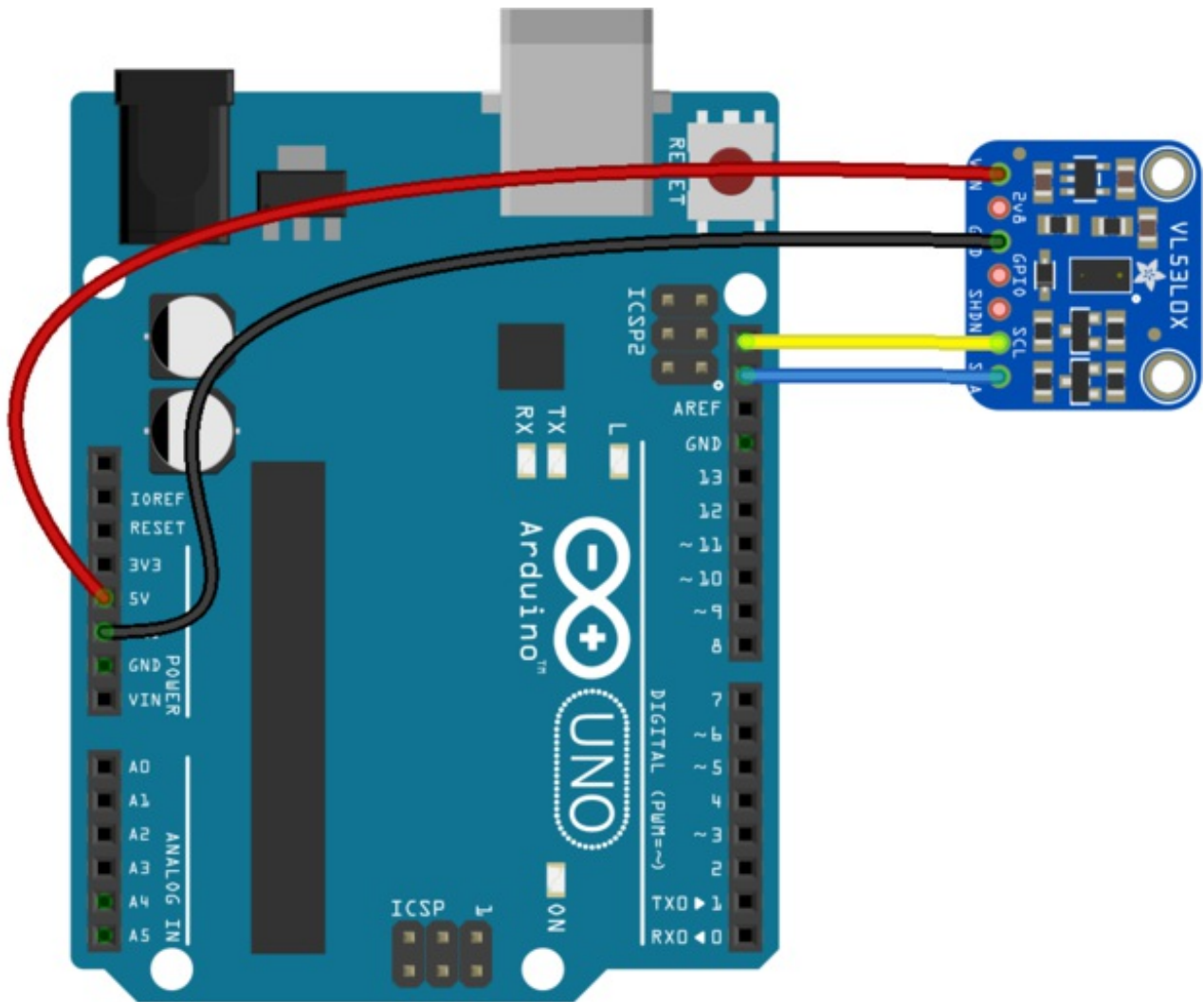
(For tips on soldering, be sure to check out our [Guide to Excellent Soldering](http://adafruit.it/aTk) (<http://adafruit.it/aTk>)).



OK You're done! Check your work over and continue on to the next steps

Wiring & Test

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 API code. We strongly recommend using an Arduino to start though!



fritzing

- Connect **Vin** to the power supply, 3-5V is fine. 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
- Connect the **SCL** pin to the I2C clock **SCL** pin on your Arduino. 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. 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 VL53L0X has a default I2C address of **0x29**!

You *can* change it, but only in software. That means you have to wire the SHUTDOWN pin and hold all but one sensor in reset while you reconfigure one sensor at a time

Download Adafruit_VL53L0X

To begin reading sensor data, you will need to [download Adafruit_VL53L0X Library from our github repository \(http://adafru.it/sDz\)](http://adafru.it/sDz). You can do that by visiting the github repo and manually downloading or, easier, just click this button to download the zip

[Download Adafruit VL53L0X library](http://adafru.it/sDA)
<http://adafru.it/sDA>

Rename the uncompressed folder **Adafruit_VL53L0X** and check that the **Adafruit_VL53L0X** folder contains **library.properties**

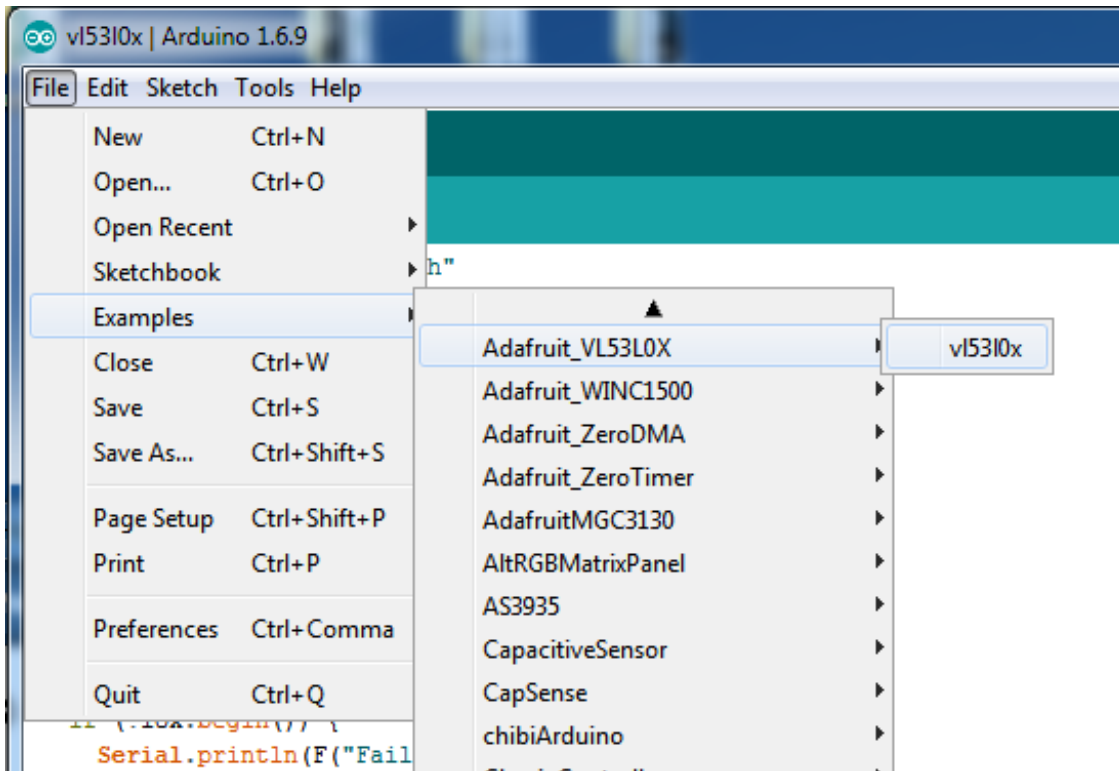
Place the **Adafruit_VL53L0X** library folder your **arduinofolder/libraries/** folder. You may need to create the **libraries** subfolder if its your first library. Restart the IDE.

We also have a great tutorial on Arduino library installation at:

<http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use> (<http://adafru.it/aYM>)

Load Demo

Open up **File->Examples->Adafruit_VL53L0X->vl53l0x** and upload to your Arduino wired up to the sensor



Thats it! Now open up the serial terminal window at 115200 speed to begin the test.

```
COM58 (Adafruit Feather 32u4)
VL53L0X API Simple Ranging example

Reading a measurement... out of range
Reading a measurement... out of range
Reading a measurement... out of range
Reading a measurement... out of range
Reading a measurement... out of range
Reading a measurement... Distance (mm): 61
Reading a measurement... Distance (mm): 41
Reading a measurement... Distance (mm): 33
Reading a measurement... Distance (mm): 50
Reading a measurement... Distance (mm): 89
Reading a measurement... Distance (mm): 135
Reading a measurement... Distance (mm): 195
Reading a measurement... Distance (mm): 220
Reading a measurement... Distance (mm): 160
Reading a measurement... Distance (mm): 97
Reading a measurement... Distance (mm): 64
Reading a measurement... Distance (mm): 81
Reading a measurement... Distance (mm): 240
Reading a measurement... Distance (mm): 349
Reading a measurement... Distance (mm): 398
Reading a measurement... Distance (mm): 438
Reading a measurement... Distance (mm): 433
Reading a measurement... Distance (mm): 453
Reading a measurement... Distance (mm): 454
Reading a measurement... Distance (mm): 462
Reading a measurement... Distance (mm): 452
```

Move your hand up and down to read the sensor data. Note that when nothing is detected, it will say the reading is out of range

Connecting Multiple Sensors

I2C only allows one address-per-device so you have to make sure each I2C device has a unique address. The default address for the VL53L0X is **0x29** but you *can* change this in software.

To set the new address you can do it one of two ways. During initialization, instead of calling `lox.begin()`, call `lox.begin(0x30)` to set the address to 0x30. Or you can, later, call `lox.setAddress(0x30)` at any time.

The good news is its easy to change, the annoying part is each *other* sensor has to be in shutdown. You can shutdown each sensor by wiring up to the **XSHUT** pin to a microcontroller pin. Then perform something like this pseudo-code:

1. Reset all sensors by setting all of their XSHUT pins low for delay(10), then set all XSHUT high to bring out of reset
2. Keep sensor #1 awake by keeping XSHUT pin high
3. Put all other sensors into shutdown by pulling XSHUT pins low
4. Initialize sensor #1 with `lox.begin(new_i2c_address)` Pick any number but 0x29 and it must be under 0x7F. Going with 0x30 to 0x3F is probably OK.
5. Keep sensor #1 awake, and now bring sensor #2 out of reset by setting its XSHUT pin high.
6. Initialize sensor #2 with `lox.begin(new_i2c_address)` Pick any number but 0x29 and whatever you set the first sensor to
7. Repeat for each sensor, turning each one on, setting a unique address.

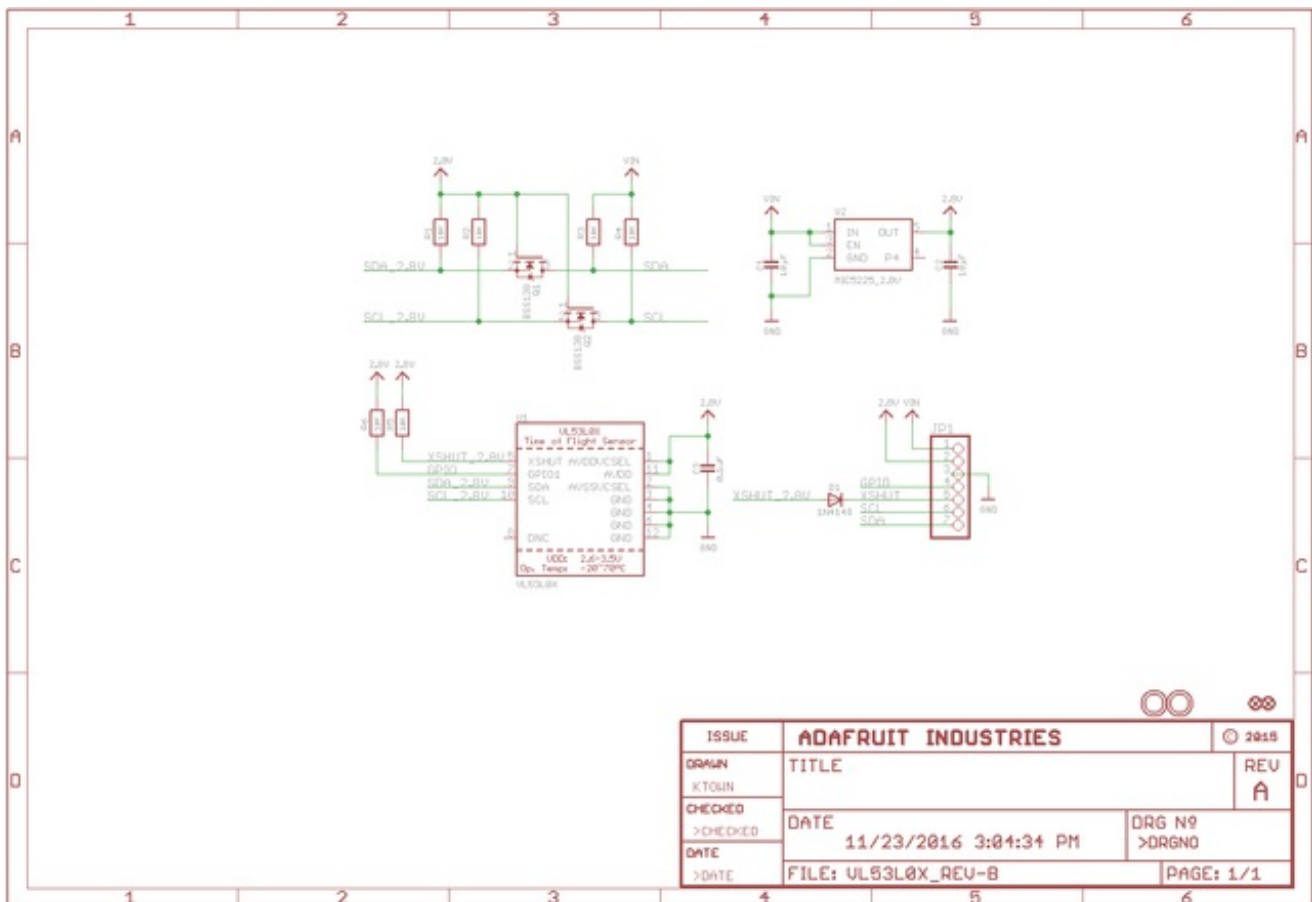
Note you must do this *every* time you turn on the power, the addresses are not permanent!

Downloads

Files & Datasheets

- [Datasheet for the VL53L0X](http://adafru.it/sDw) (<http://adafru.it/sDw>)
- [ST product page](http://adafru.it/sDx) (<http://adafru.it/sDx>) with more details including API downloads
- [Fritzing object in Adafruit Fritzing library](http://adafru.it/aP3) (<http://adafru.it/aP3>)
- [EagleCAD PCB files in GitHub](http://adafru.it/sDy) (<http://adafru.it/sDy>)
- [Adafruit VL53L0X library](http://adafru.it/sDz) (<http://adafru.it/sDz>)
- [Pololu VL53L0X library](http://adafru.it/sTf) (<http://adafru.it/sTf>)

Schematic & Fabrication Print



Use M2.5 or #2-56 screws to mount

