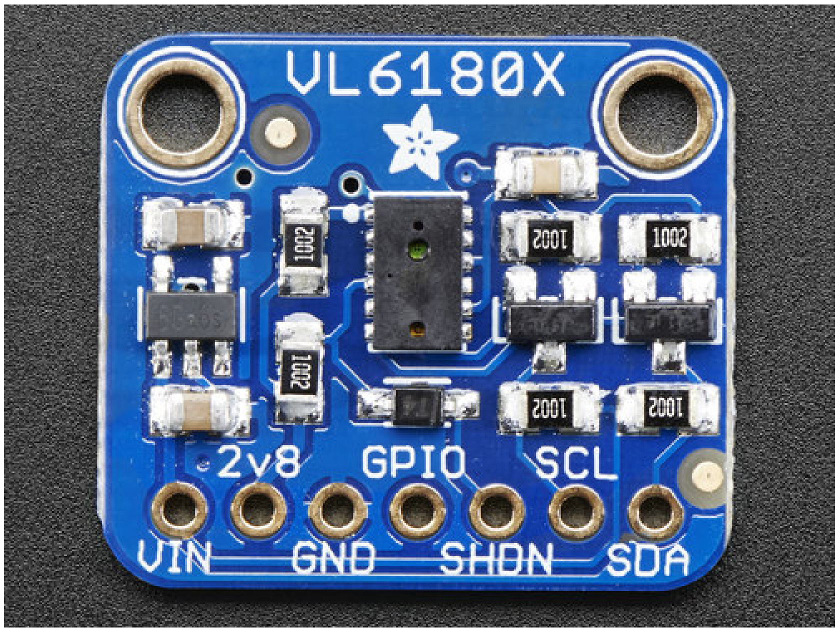
Adafruit VL6180X Time of Flight Micro-LIDAR Distance Sensor Breakout

Created by lady ada

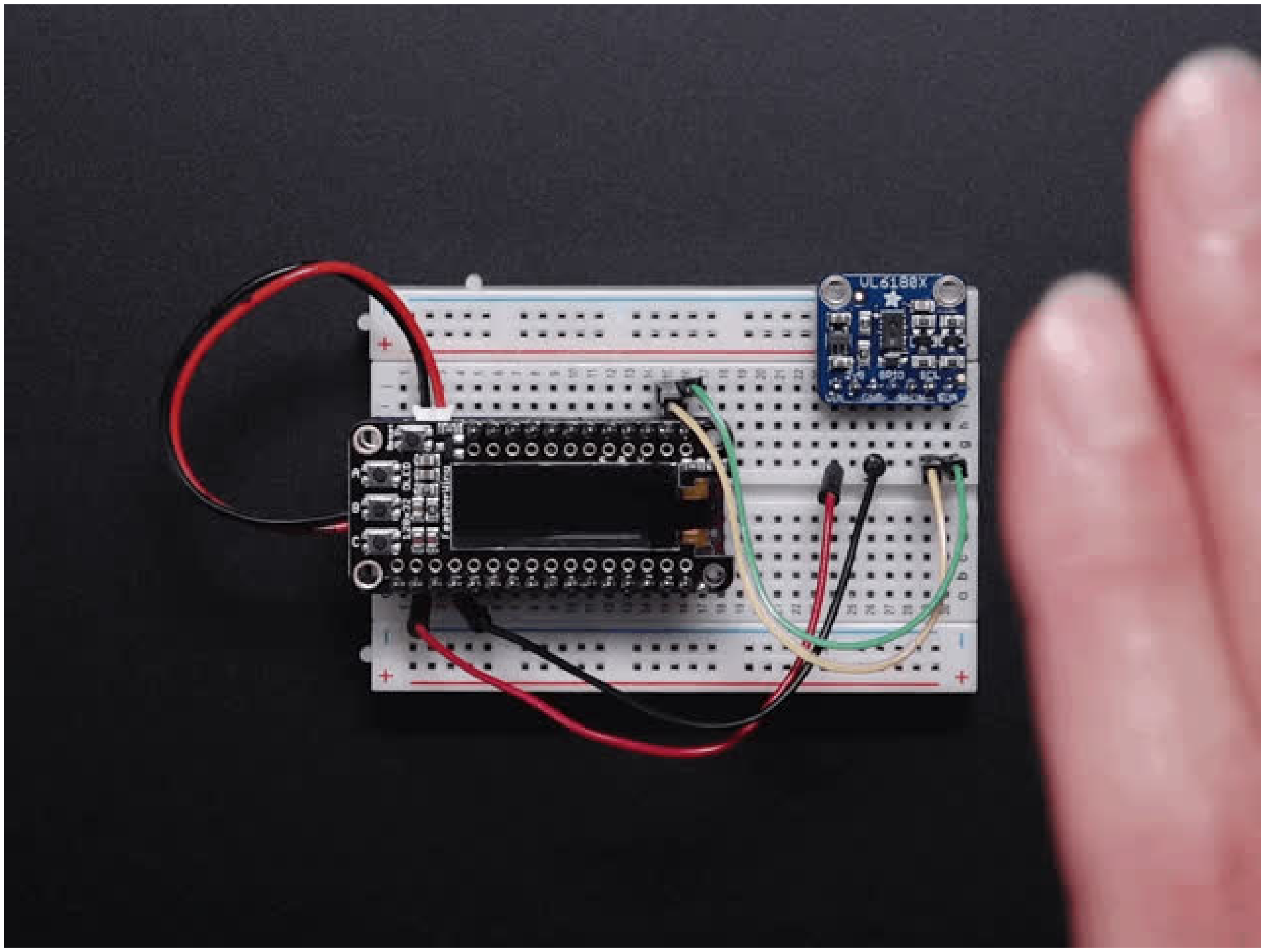


Last updated on 2016-11-30 07:49:35 PM UTC

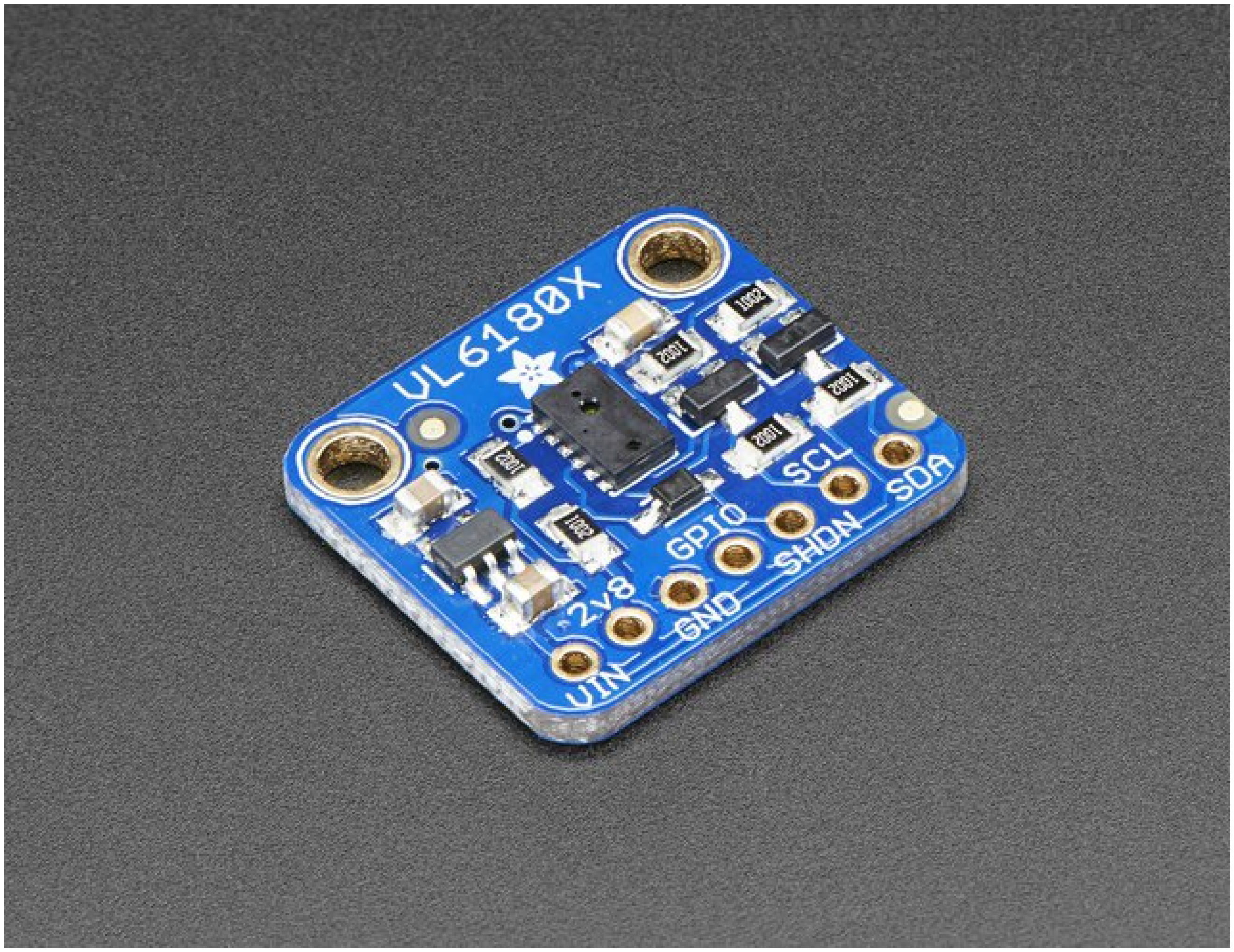
Guide Contents

Guide Contents 2
Overview 3
Pinouts 7
  Power Pins: 7
  I2C Logic pins: 8
Control Pins: 8
Assembly 9
  Prepare the header strip: 9
  Solder! 11
Wiring & Test 13
Download Adafruit_VL6180X 14
Load Demo 14
Library Reference 16
Reading Distance 16
Reading Light / Lux 17
Downloads 18
Files & Datasheets 18
Schematic & Fabrication Print 18

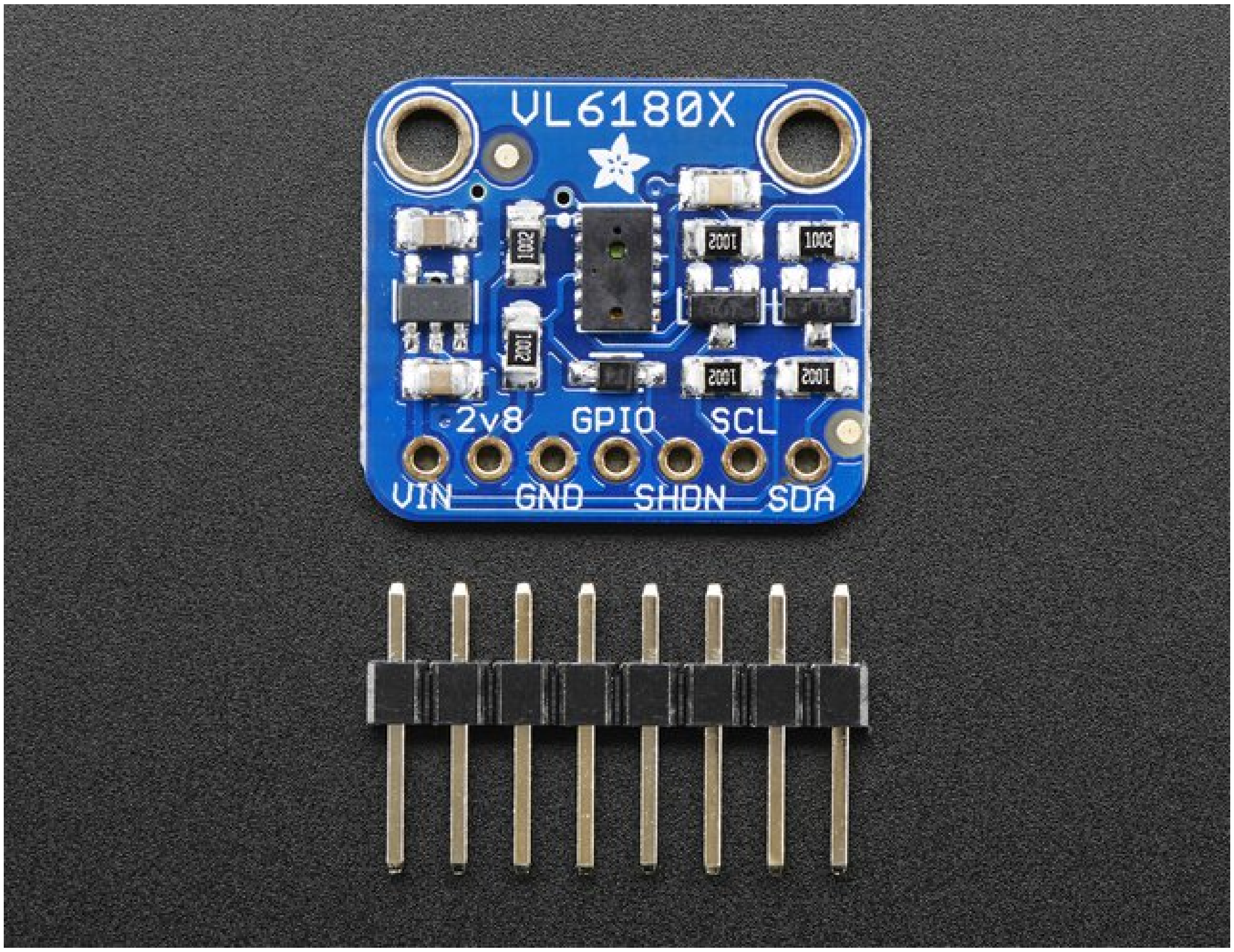
Overview



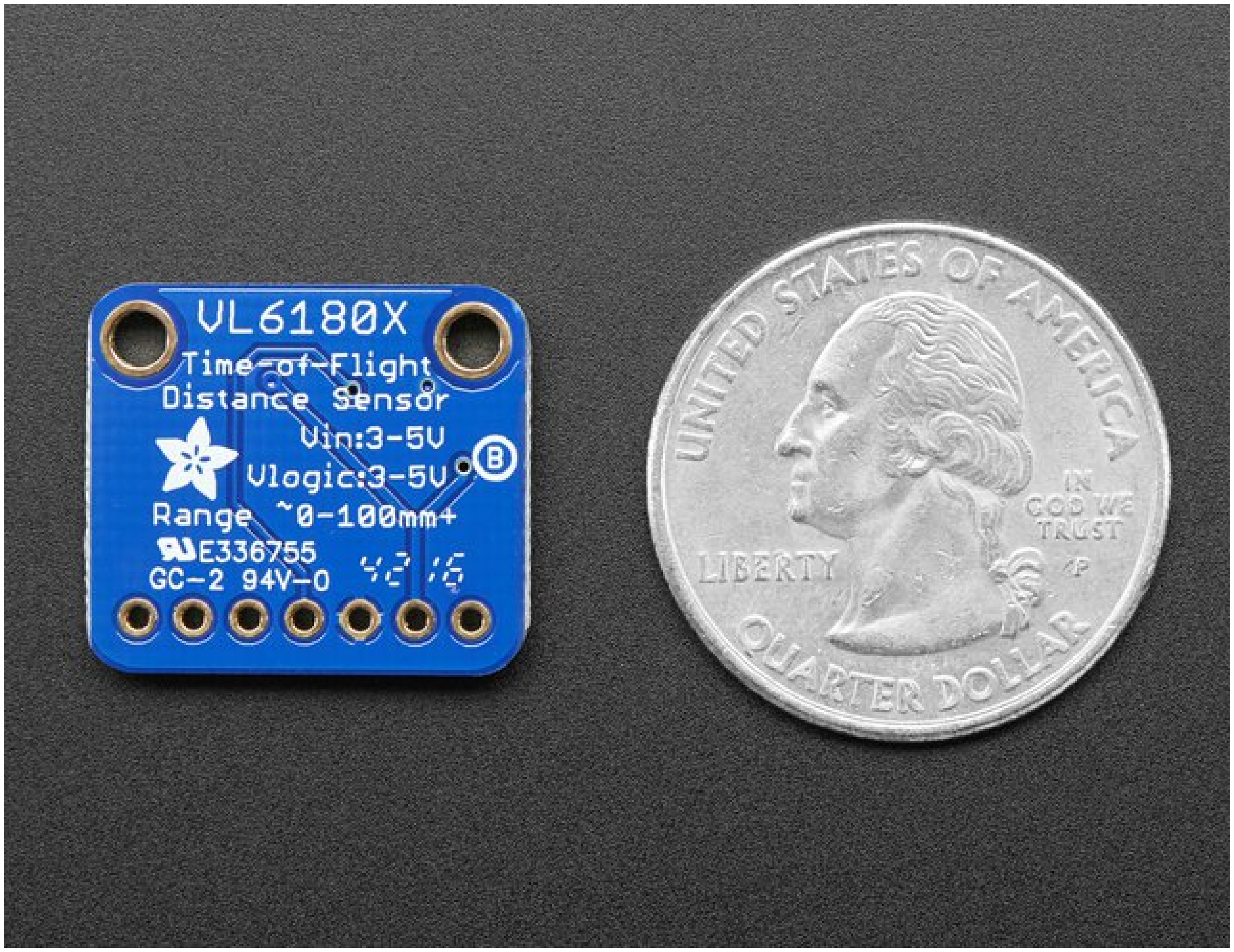
The VL6180X is a Time of Flight distance sensor like no other you've used! The sensor contains a very tiny invisible laser source, and a matching sensor. The VL6180X can detect the "time of flight", or how long the light has taken to bounce back to the sensor. Since it uses a very narrow light source,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



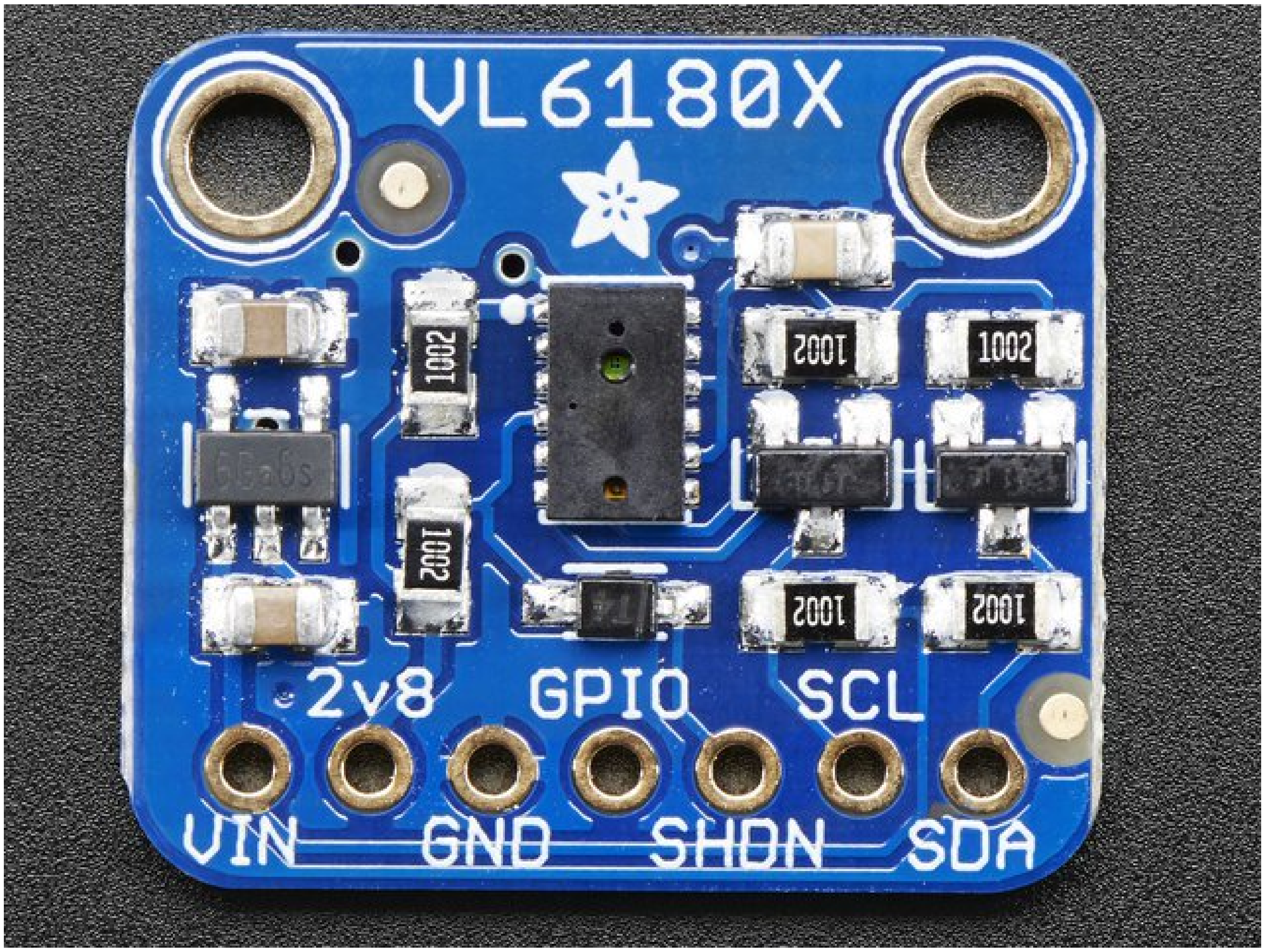
The sensor is small and easy to use in any robotics or interactive project. Since it needs 2.8V power and logic we put the little fellow on a breakout board with a regulator and level shifting. You can use it with any 3-5V power or logic microcontroller with no worries. Each order comes with a small 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 some simple commands. Most of the work is handled inside the sensor itself, so its very easy to port our Arduino library to another microcontroller.

Pinouts

The VL6180X 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 I2C address is **0x29** and you *can't* change it!



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

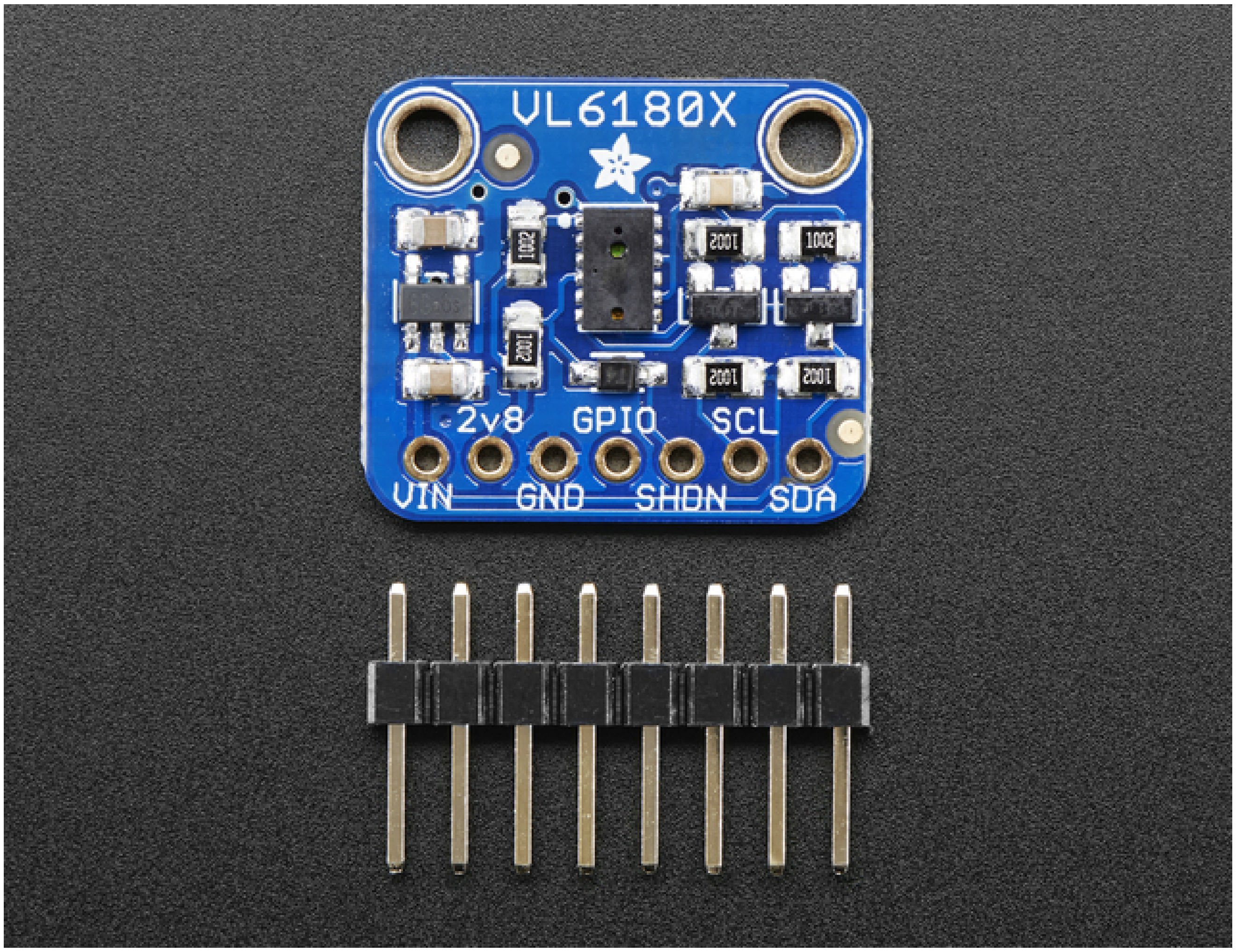
I2C Logic pins:

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

Control Pins:

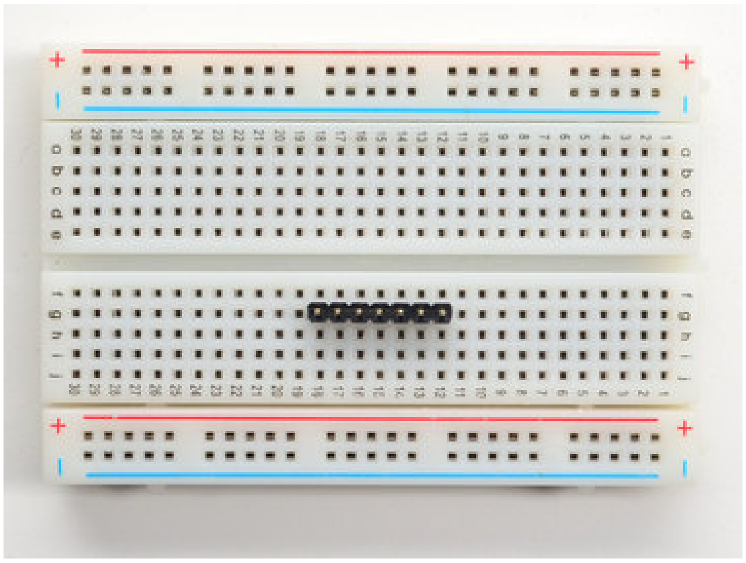
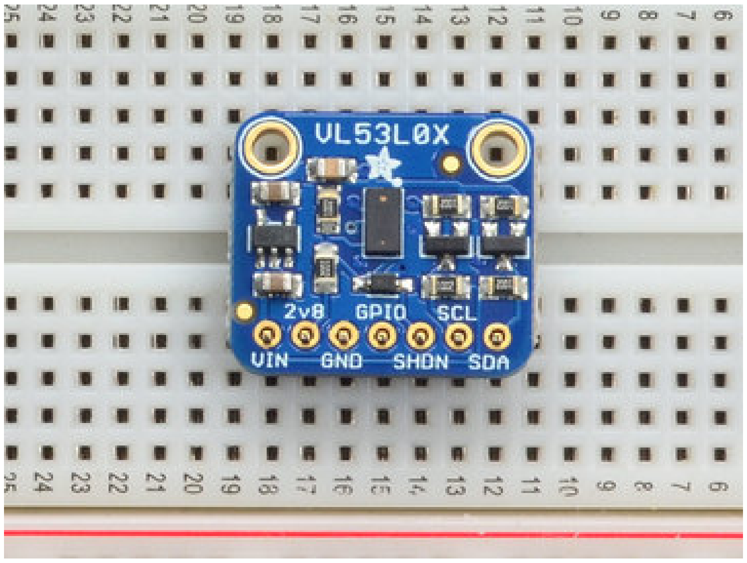
- **GPIO** - this is a pin that is used by the sensor to indicate that data is ready. It's useful for when doing continuous sensing. Note there is no level shifting on this pin, you may not be able to read the 2.8V-logic-level voltage on a 5V microcontroller (we could on an arduino UNO but no promises). 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

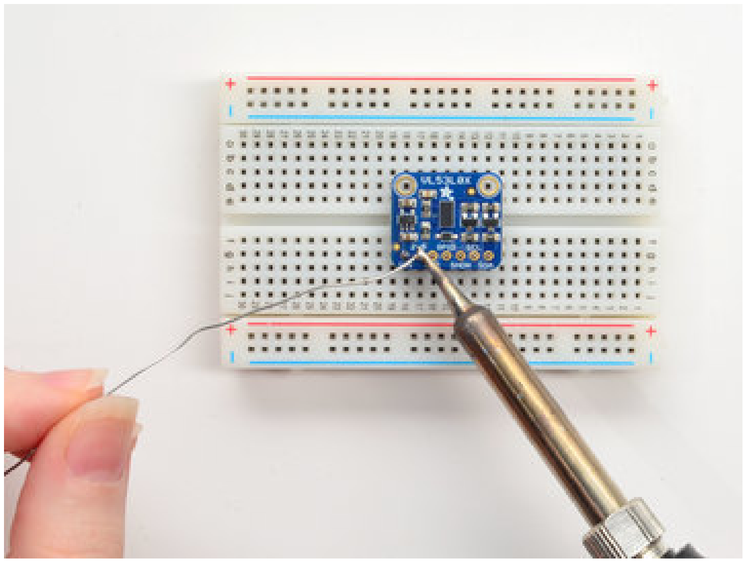


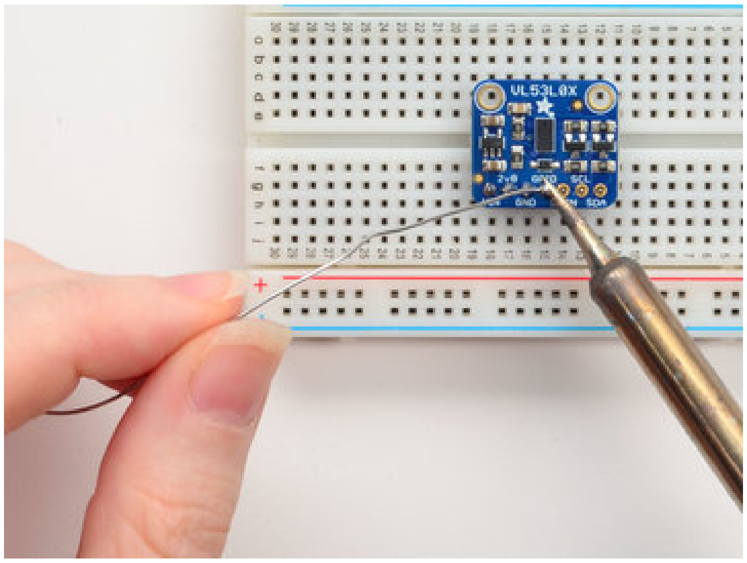
This page shows the VL53L0X or VL6180X sensor - the procedure is identical!

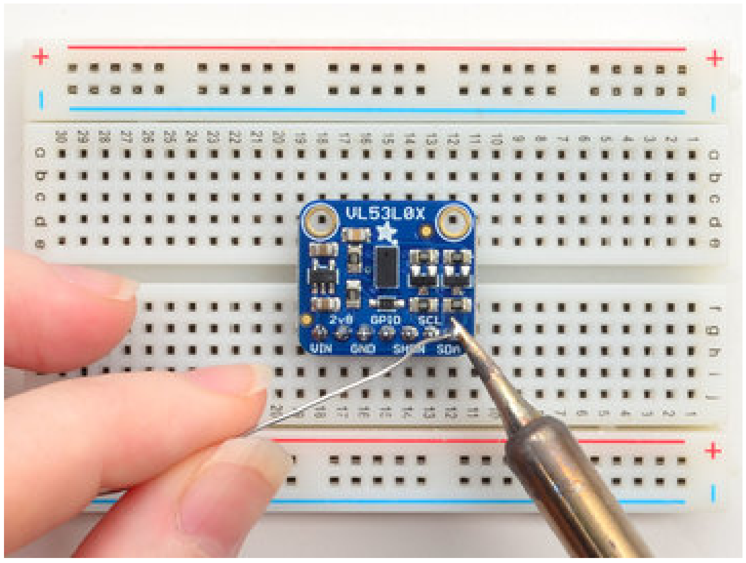
Prepare the header strip:

- 
  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

- 

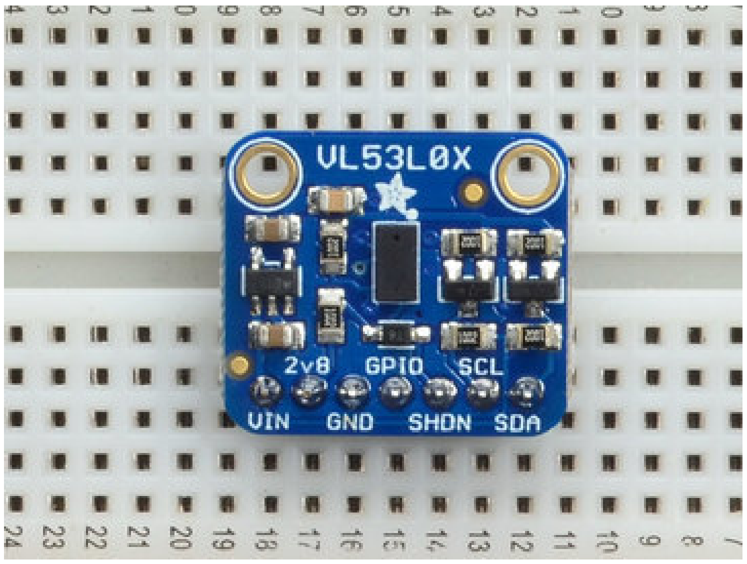
- 

- 

Solder!

Be sure to solder all pins for reliable electrical contact.

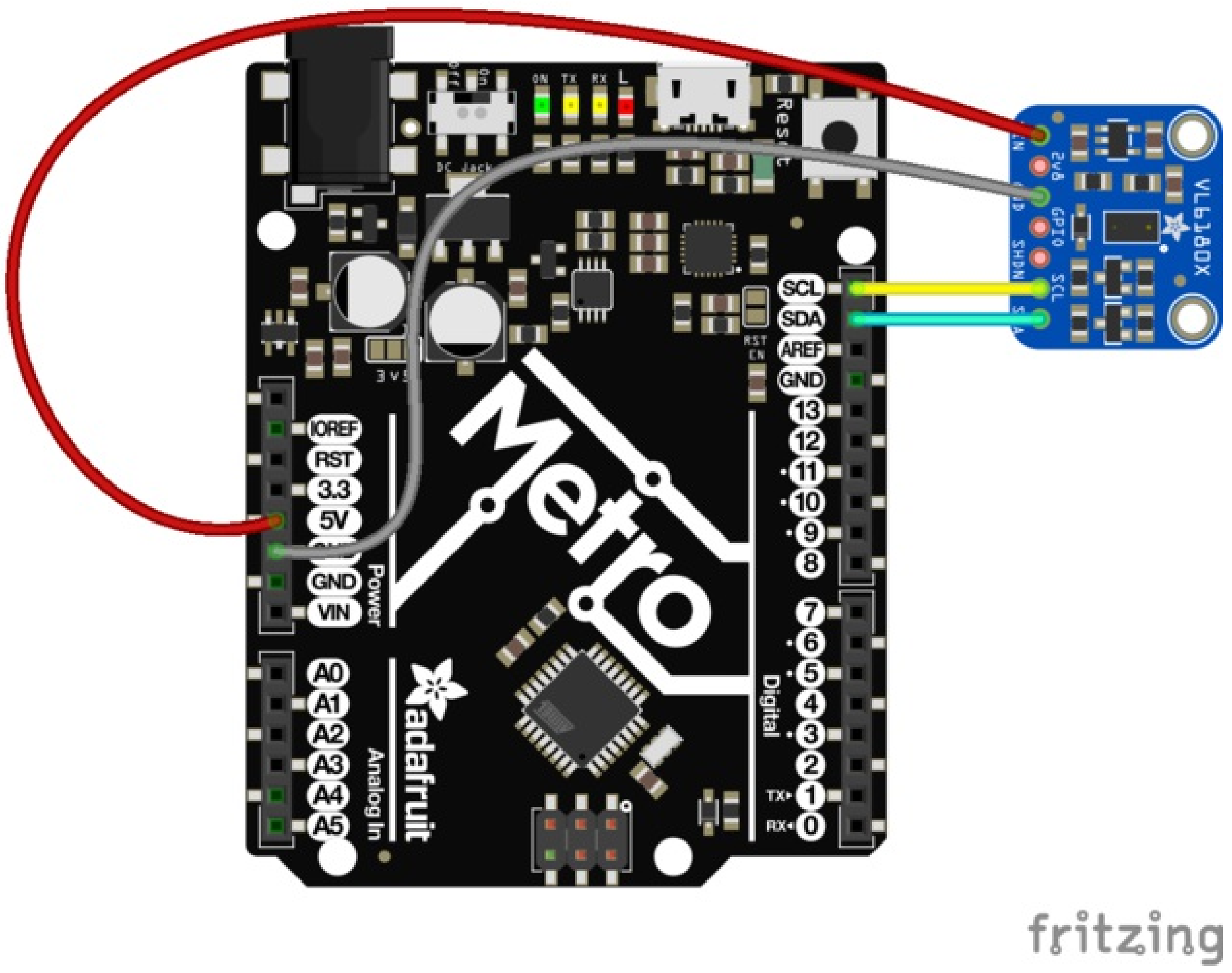
(For tips on soldering, be sure to check out our Guide to Excellent Soldering (http://adafru.it/aTk)).

- 

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!



- 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 VL6180X has a default I2C address of 0x29!

Download Adafruit_VL6180X

To begin reading sensor data, you will need todownload Adafruit_VL6180X Library from our github repository (http://adafru.it/sld). You can do that by visiting the github repo and manually downloading or, easier, just click this button to download the zip

Download Adafruit VL6180X library
http://adafru.it/sle

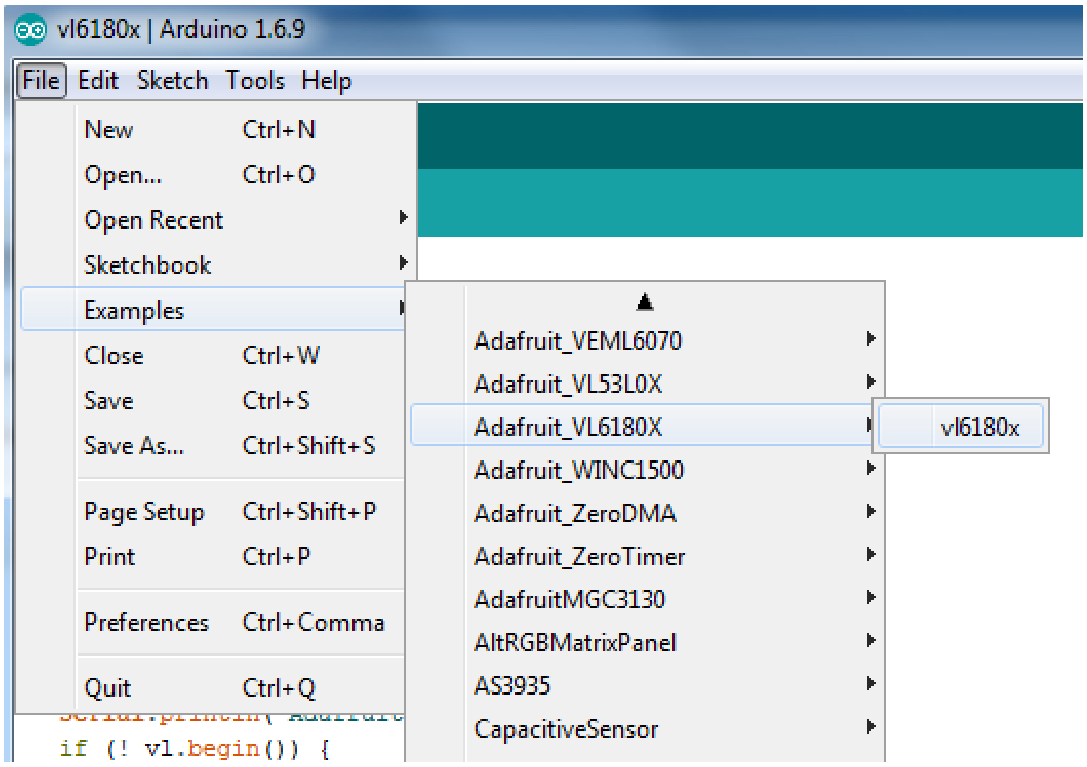
Rename the uncompressed folder **Adafruit_VL6180X** and check that the **Adafruit_VL6180X** folder contains **library.properties**

Place the **Adafruit_VL6180X** library folder your ***arduinosketchfolder*/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)

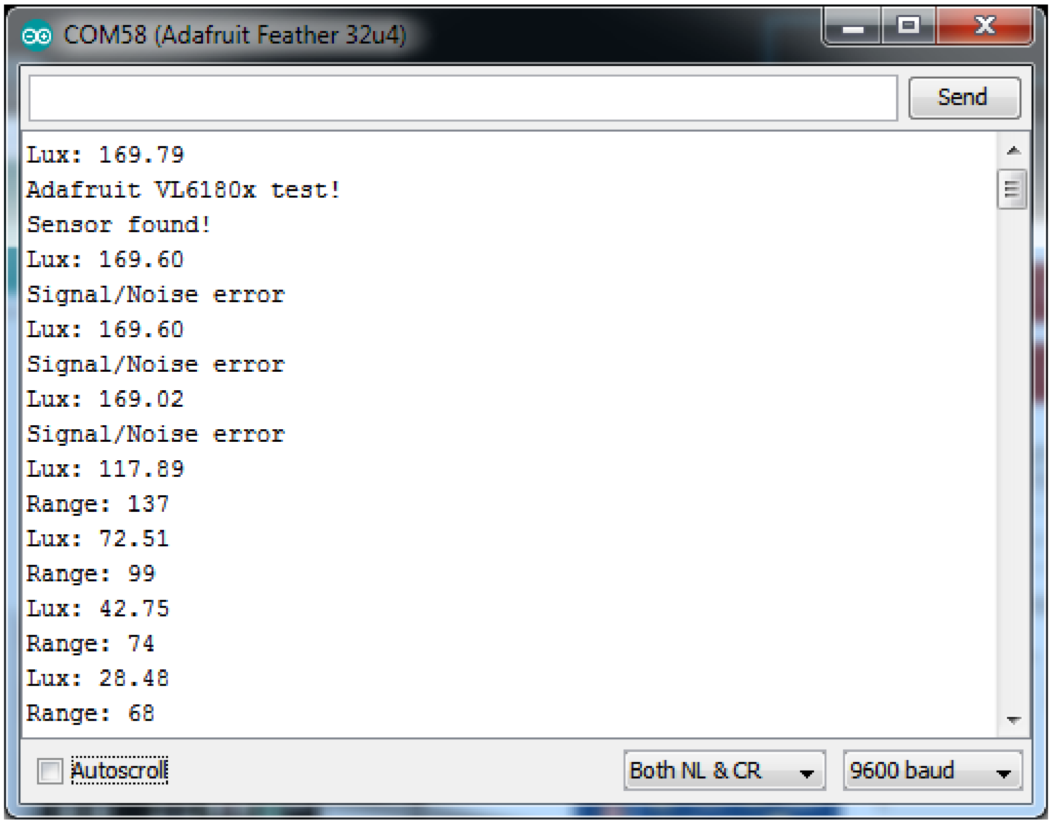
Load Demo

Open up **File->Examples->Adafruit_VL6180X->vl6180x** and upload to your Arduino wired up to the sensor



```
if (! vl.begin()) {
```

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



```
Lux: 169.79
Adafruit VL6180x test!
Sensor found!
Lux: 169.60
Signal/Noise error
Lux: 169.60
Signal/Noise error
Lux: 169.02
Signal/Noise error
Lux: 117.89
Range: 137
Lux: 72.51
Range: 99
Lux: 42.75
Range: 74
Lux: 28.48
Range: 68
```

Move your hand up and down to read the sensor data, the range readings are in millimeters and the light sensors in Lux. Note that when nothing is detected, it will print out the error which may vary.

Library Reference

You can start out by creating a Adafruit_VL6180X object using the default I2C:

```
Adafruit_VL6180X vl = Adafruit_VL6180X();
```

Once started, you can initialize the sensor which will also do some simple initializations and settings using **begin()**. Note that this will return True if the sensor was found and initialized. If not, it will return False.

```
vl.begin()
```

Reading Distance

Reading the range/distance is easy, just call **readRange()**

```
vl.readRange()
```

which will return the range in millimeters. If you get 0 or a value over 200 there's likely an error. Either way, before you trust that reading make sure to ask the sensor if the last reading had an error:

```
uint8_t status = vl.readRangeStatus()
```

The status will be **0** on no error, anything else is an error. Here's a simple code chunk that will decode the error!

```
 if  ((status >= VL6180X_ERROR_SYSERR_1) && (status <= VL6180X_ERROR_SYSERR_5)) {
  Serial.println("System error");
 }
 else if (status == VL6180X_ERROR_ECEFAIL) {
  Serial.println("ECE failure");
 }
 else if (status == VL6180X_ERROR_NOCONVERGE) {
  Serial.println("No convergence");
 }
 else if (status == VL6180X_ERROR_RANGEIGNORE) {
  Serial.println("Ignoring range");
 }
 else if (status == VL6180X_ERROR_SNR) {
  Serial.println("Signal/Noise error");
 }
 else if (status == VL6180X_ERROR_RAWUFLOW) {
  Serial.println("Raw reading underflow");
 }
 else if (status == VL6180X_ERROR_RAWOFLOW) {
  Serial.println("Raw reading overflow");
 }
 else if (status == VL6180X_ERROR_RANGEUFLOW) {
  Serial.println("Range reading underflow");
 }
 else if (status == VL6180X_ERROR_RANGEOFLOW) {
  Serial.println("Range reading overflow");
 }
```

Reading Light / Lux

You can also read a light reading from the sensor with

```
vl.readLux(GAIN)
```

which will return a semi-calibrated Lux reading. You can use different Gain settings to get a different range. For better results at low light, use higher gain. For better results at high light, use a lower gain.

Here's the gain's available:

- **VL6180X_ALS_GAIN_1** - gain of 1x
- **VL6180X_ALS_GAIN_1_25** - gain of 1.25x
- **VL6180X_ALS_GAIN_1_67** - gain of 1.67x
- **VL6180X_ALS_GAIN_2_5** - gain of 2.5x
- **VL6180X_ALS_GAIN_5** - gain of 5x
- **VL6180X_ALS_GAIN_10** - gain of 10x
- **VL6180X_ALS_GAIN_20** - gain of 20x
- **VL6180X_ALS_GAIN_40** - gain of 40x

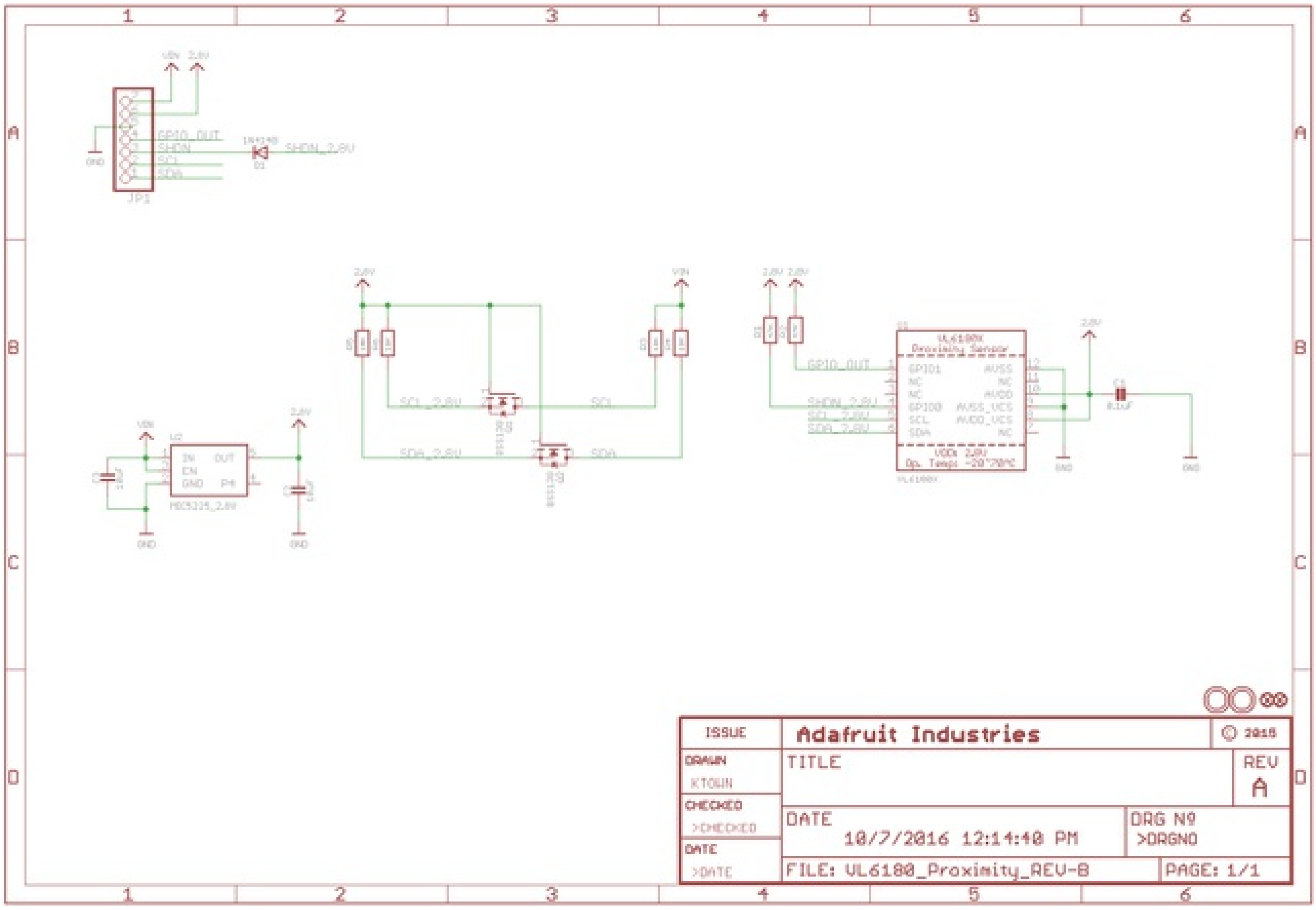
We suggest starting with a gain of 5x and then adjusting up or down as necessary

Downloads

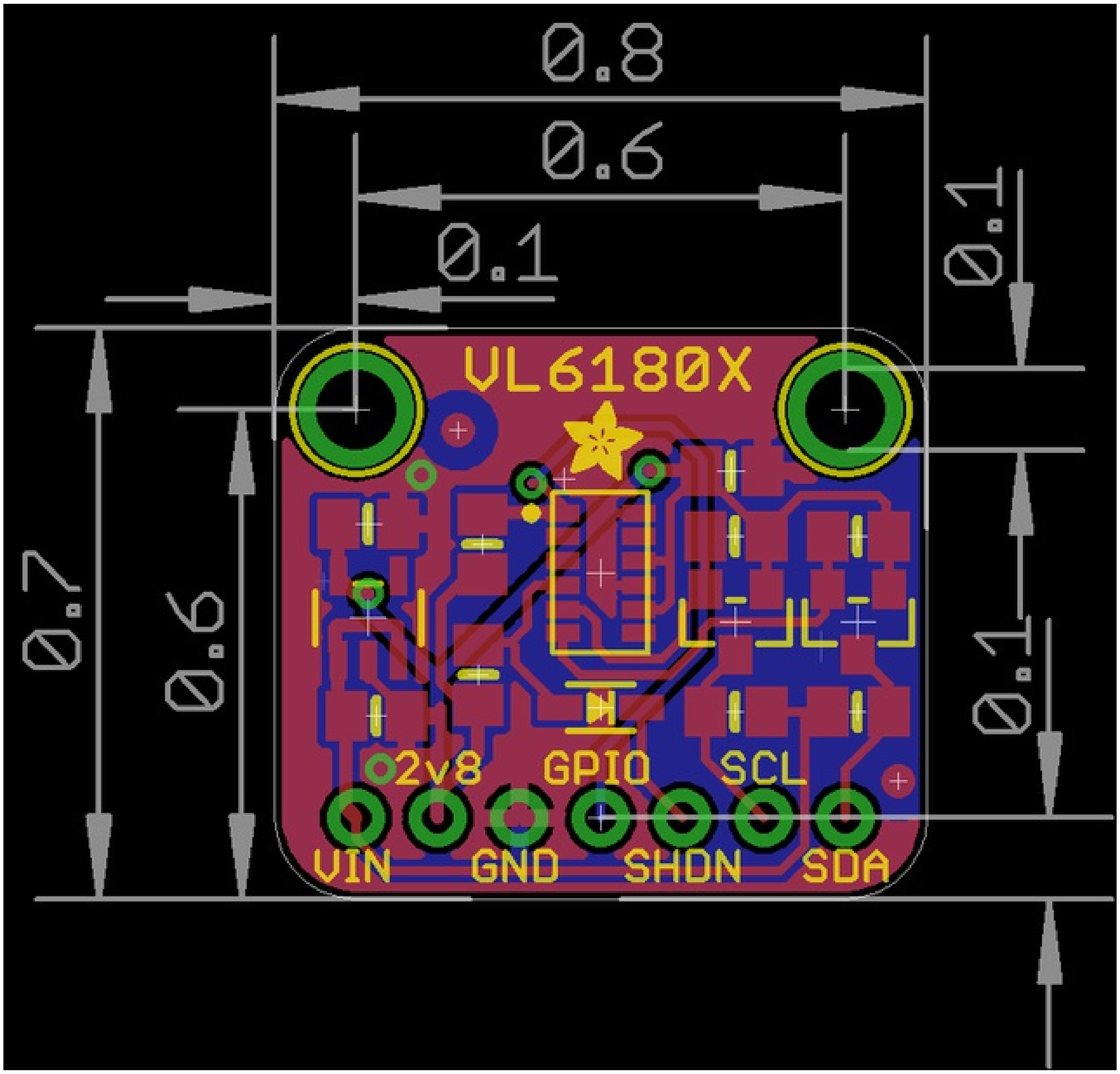
Files & Datasheets

- Datasheet for the VL6180X (http://adafru.it/sIa)
- ST product page (http://adafru.it/sIb) with more details including API downloads
- Fritzing object in Adafruit Fritzing library (http://adafru.it/aP3)
- EagleCAD PCB files in GitHub (http://adafru.it/sIc)

Schematic & Fabrication Print



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


0.8
0.6
0.1
0.1
0.7
0.6
0.1
VL6180X
2v8
GPIO
SCL
VIN
GND
SHDN
SDA