Adafruit VEML7700 Ambient Light Sensor

Created by Kattni Rembor

https://learn.adafruit.com/adafruit-veml7700
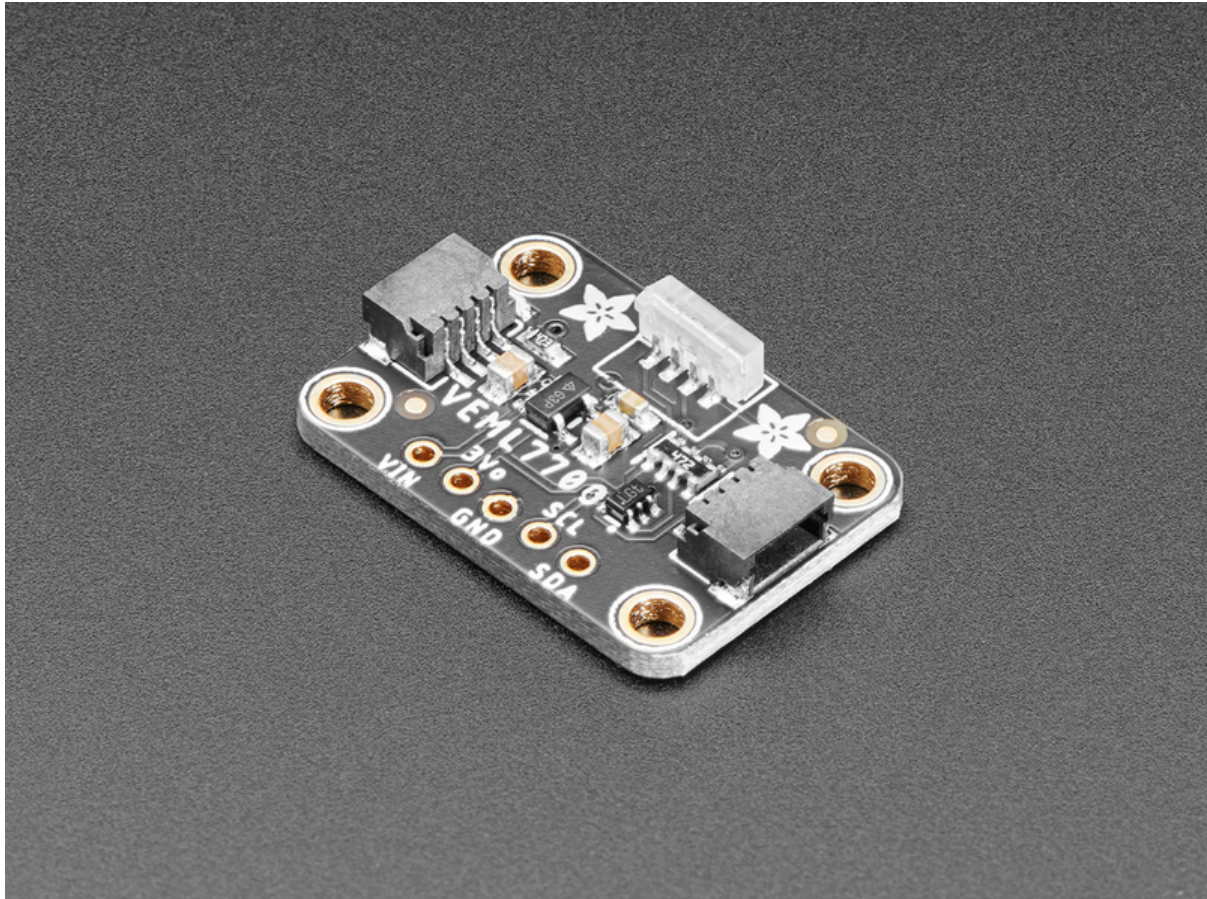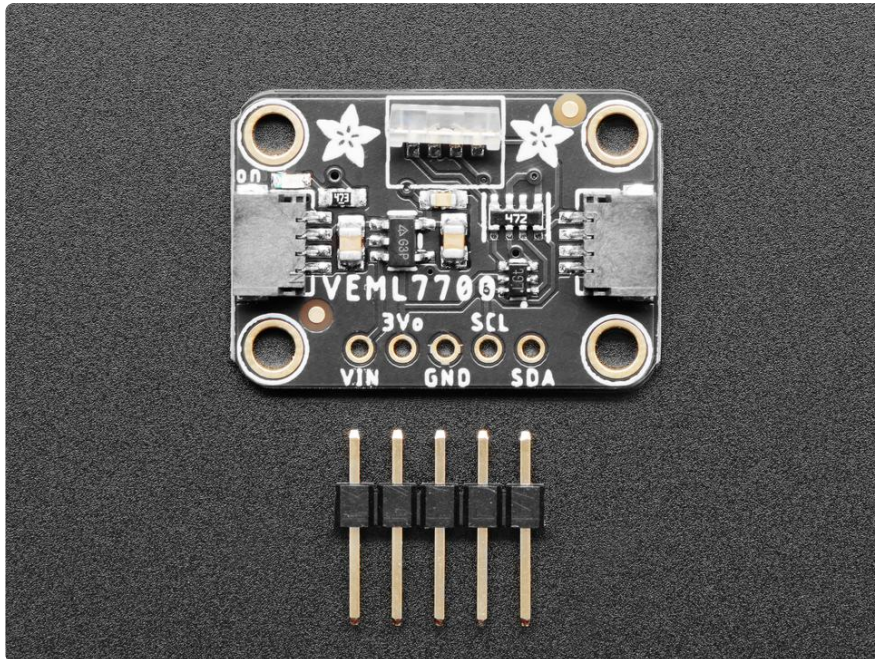
Last updated on 2024-06-03 02:43:25 PM EDT

Table of Contents

Overview

Vishay has a lot of light sensors out there, and this is a nice simple lux sensor that's easy to add to any microcontroller. Most light sensors just give you a number for brighter/darker ambient lighting. The VEML7700 makes your life easier by calculating the lux, which is an SI unit for light. You'll get more consistent readings between multiple sensors because you aren't dealing with some unitless values.

The sensor has 16-bit dynamic range for ambient light detection from 0 lux to about 120k lux with resolution down to 0.0036 lx/ct, with software-adjustable gain and integration times.
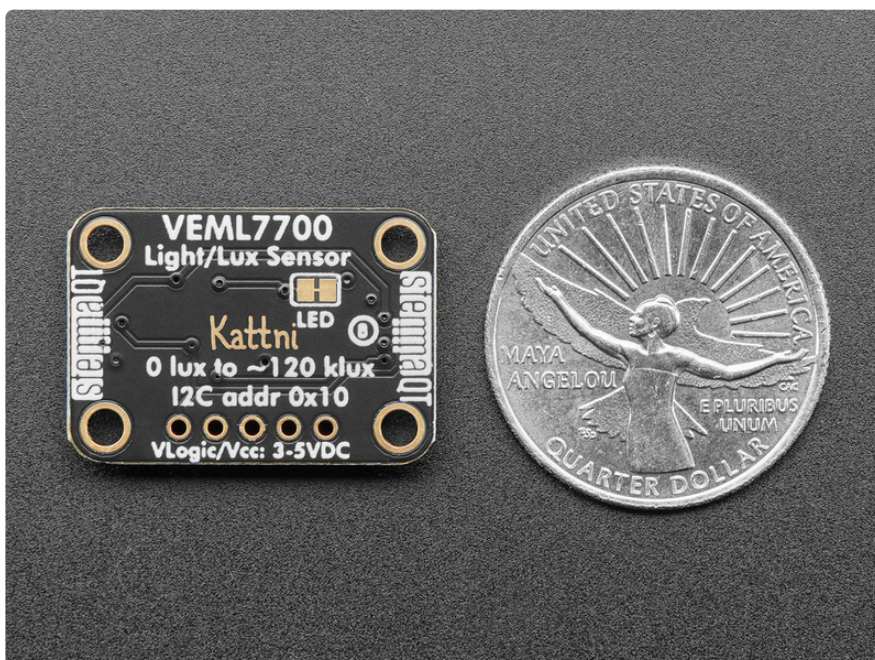
Interfacing is easy - this sensor uses plain, universal I2C. We put this sensor on a breakout board with a 3.3V regulator and logic level shifter so you can use it with 3.3V or 5V power/logic microcontrollers. We have written libraries for Arduino (https://adafru.it/EoE) (C/C++) as well as CircuitPython (Python 3) (https://adafru.it/EoF) so you can use this sensor with just about any kind of device, even a Raspberry Pi!

This is Kattni's first PCB design for Adafruit, it's even signed on the back!
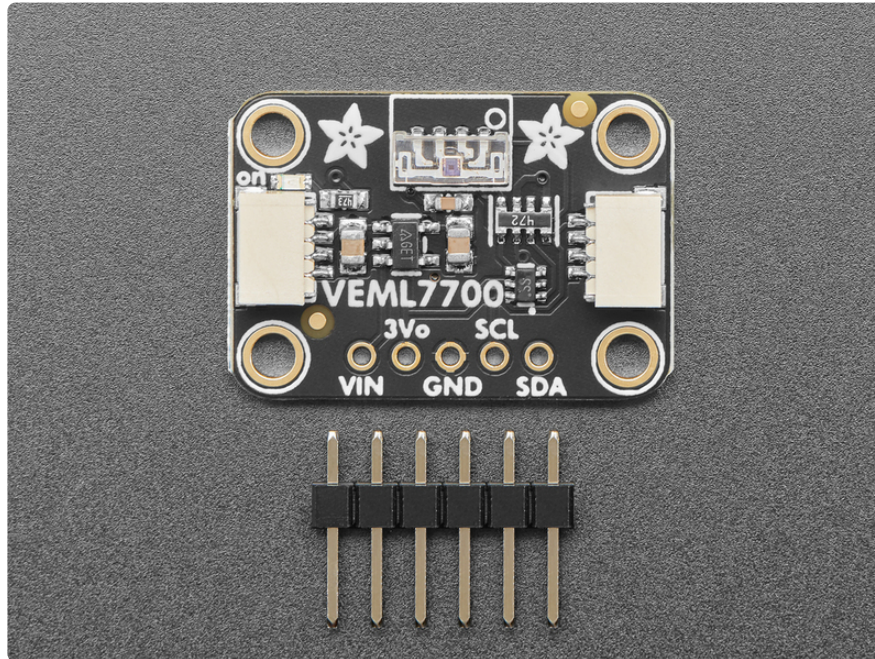
As with all Adafruit breakouts, we've done the work to make this handy light sensor super easy to use. We've put it on a breakout board with the required support circuitry and connectors to make it easy to work with. Since I2C is supported, we've added SparkFun Qwiic (https://adafru.it/Fpw) compatible STEMMA QT (https://adafru.it/Ft4) JST SH connectors that allow you to get going **without needing to solder.** Just use a STEMMA QT adapter cable (http://adafru.it/4209), plug it into your favorite microcontroller or Blinka supported SBC and you're ready to rock! QT Cable is not included, but we have a variety in the shop (https://adafru.it/17VE).

> There are two versions of this board - the STEMMA QT version shown above, and the original header-only version shown below. Code works the same on both!
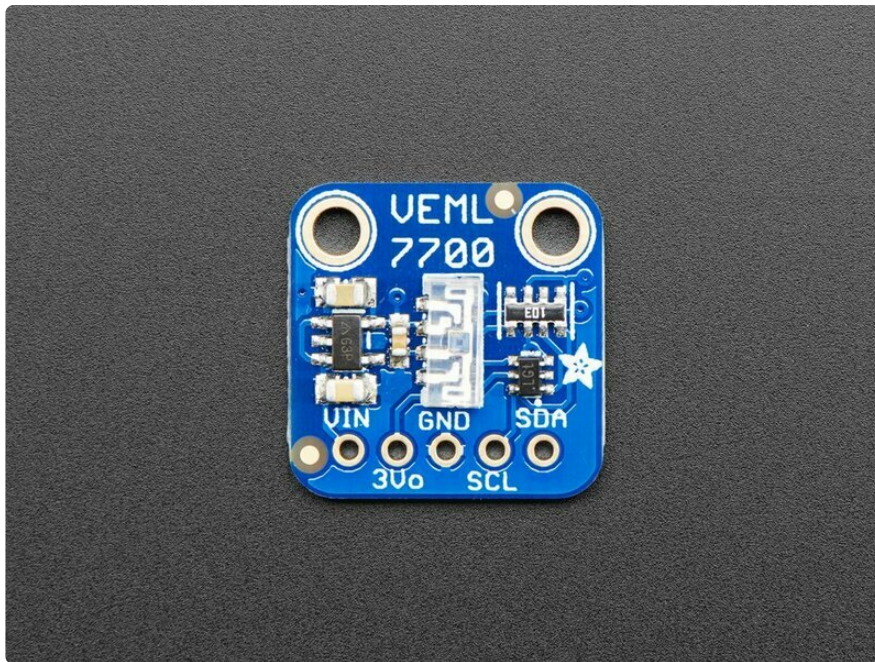
Pinouts
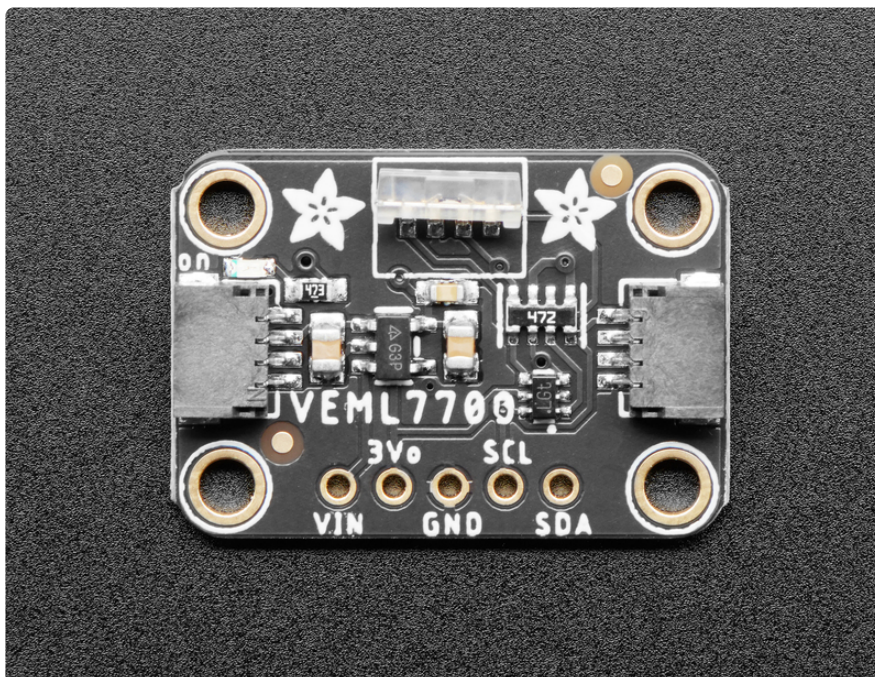
Power Pins

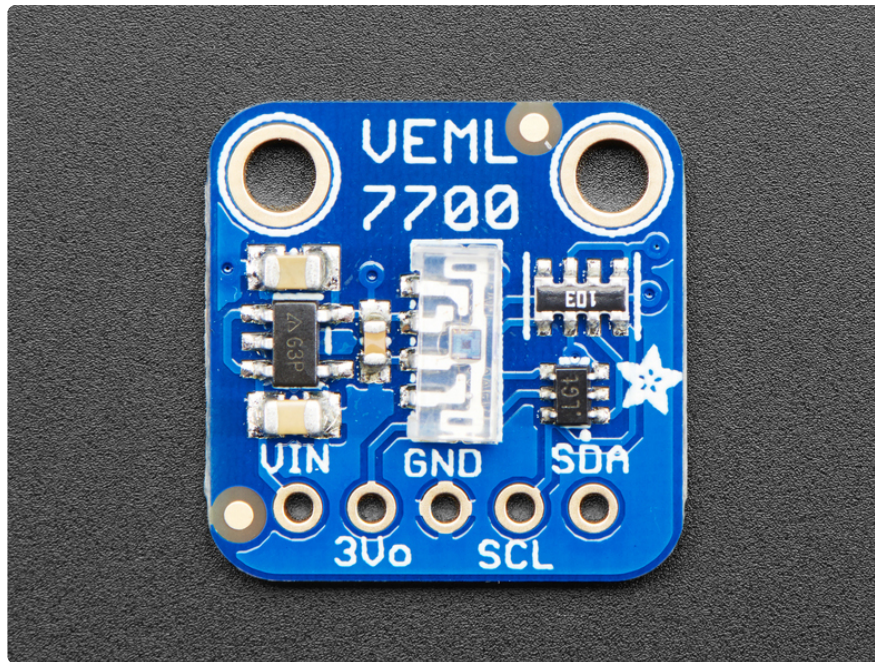- **Vin** - this is the power pin. Since the sensor chip uses 3 VDC, we have included a voltage regulator on board that will take 3-5VDC and safely convert it down. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V micro like Arduino, use 5V
- **3Vo** - 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 development boards with **STEMMA QT** connectors, or to other things, with various associated accessories (https://adafru.it/Ft6).

Power LED and LED Jumper

- **Power LED** - In the upper right corner, above the STEMMA connector, on the front of the board, is the power LED, labeled **on**. It is a green LED.
- **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.
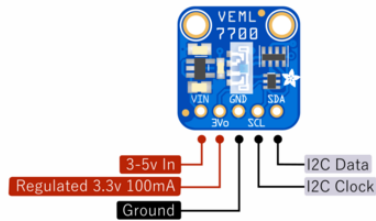
Assembly
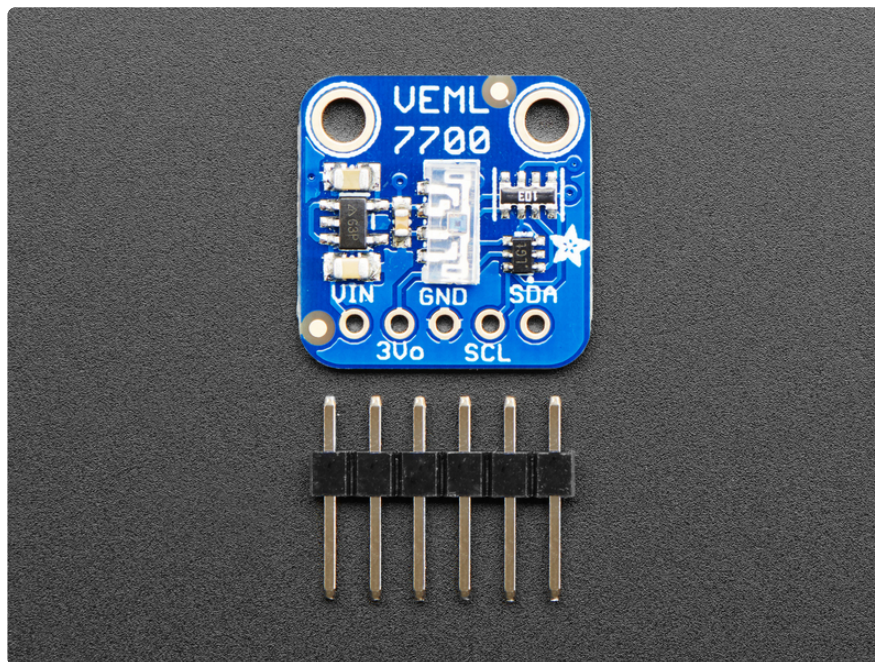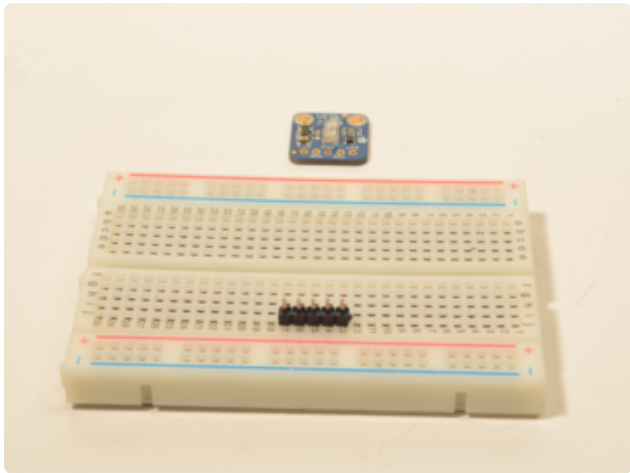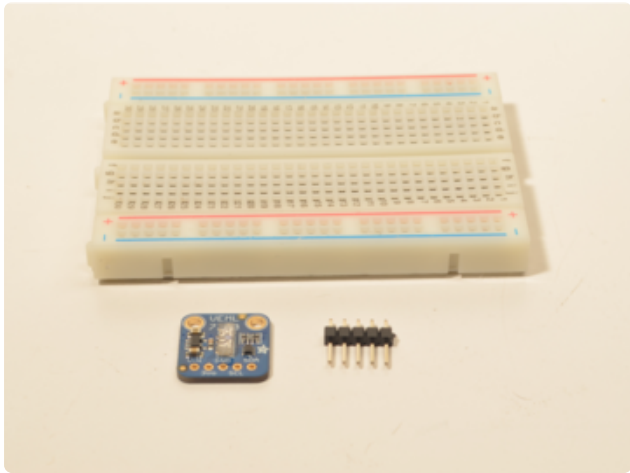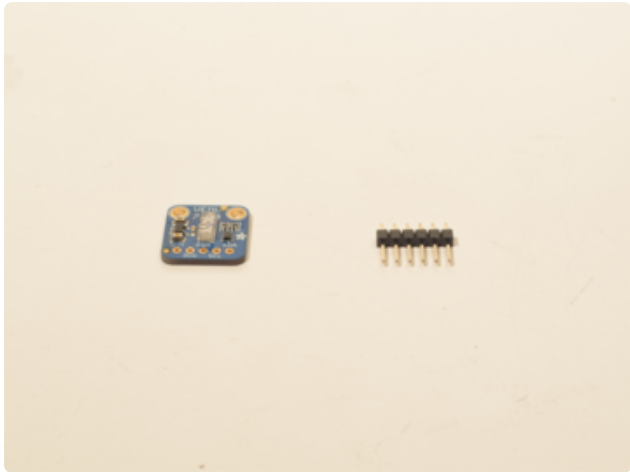
Prepare the header strip:
Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - **long pins down**
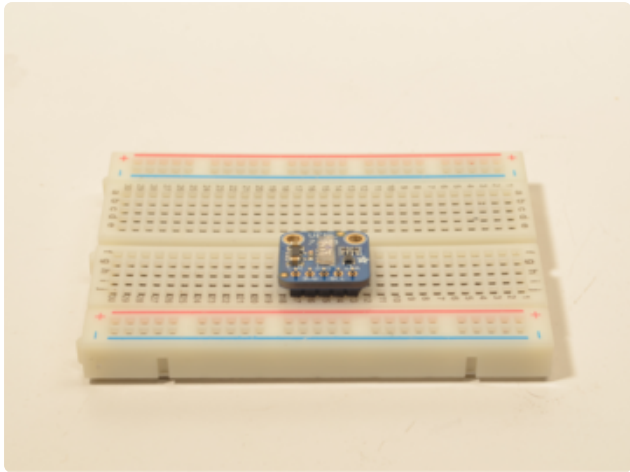
Add the breakout board:

Place the breakout board over the pins so that the short pins poke through the breakout pads
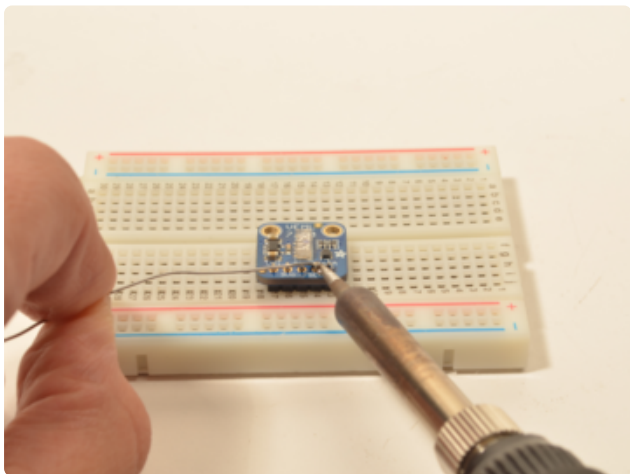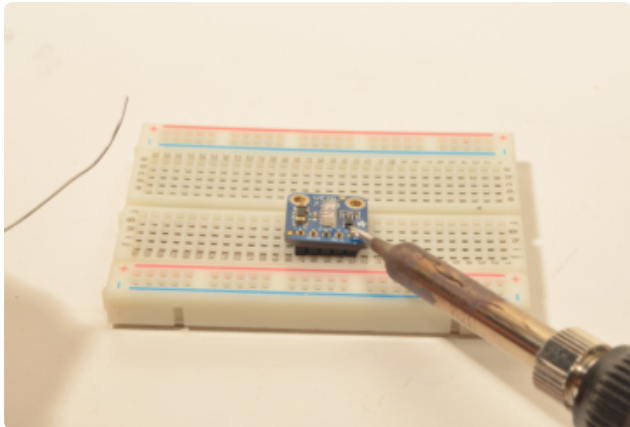
And Solder!

Be sure to solder all 5 pins for reliable electrical contact.

(For tips on soldering, be sure to check out our Guide to Excellent Soldering (https://adafru.it/aTk)).
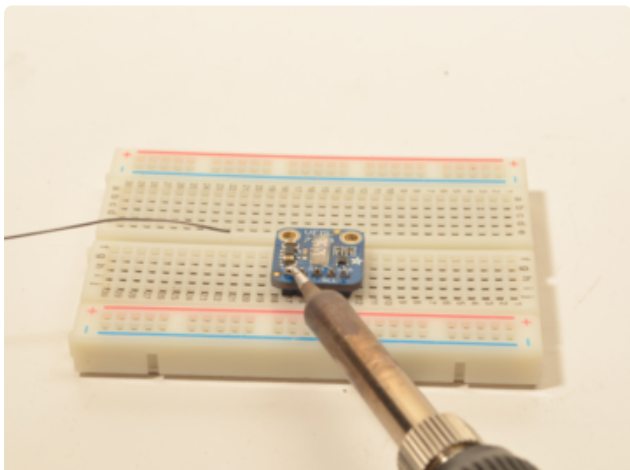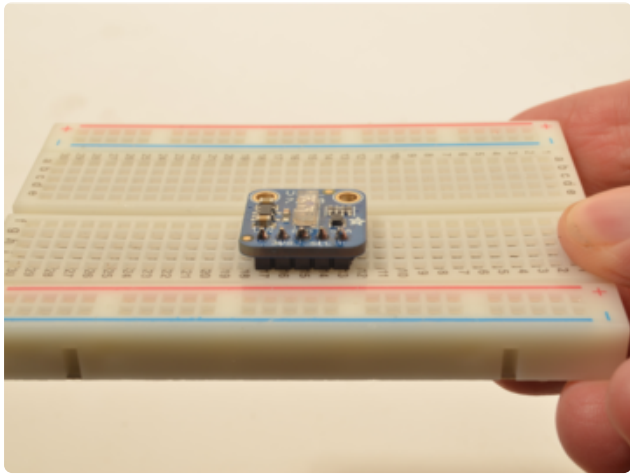
You're done! Check your solder joints visually and continue onto the next steps

Arduino

Wiring

Connecting the VEML7700 to your Feather or Arduino is easy:
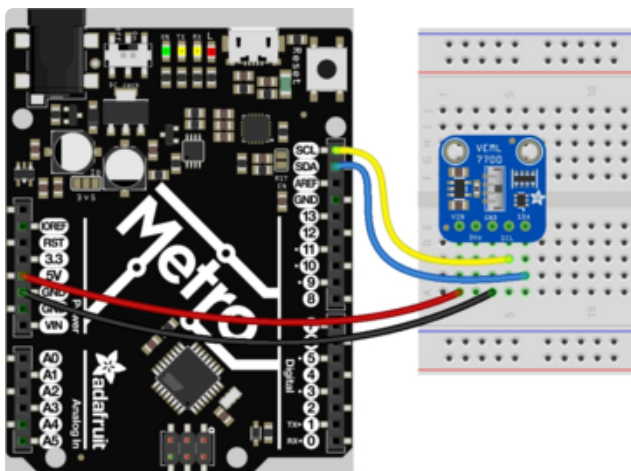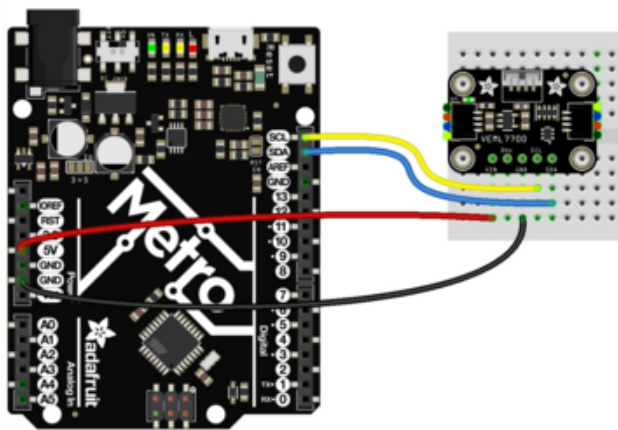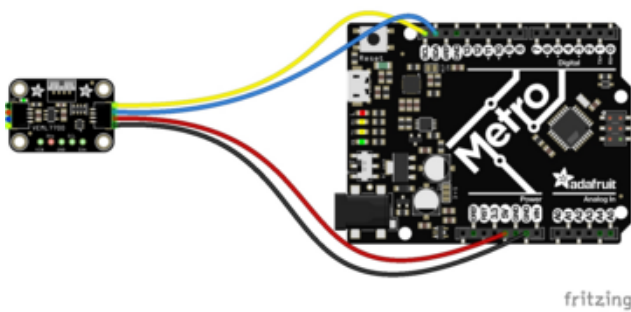
If you are running a Feather (3.3V), connect **Feather 3V** to **board VIN (red wire on STEMMA QT version)**

If you are running a 5V Arduino (Uno, etc.), connect **Arduino 5V** to **board VIN (red wire on STEMMA QT version)**

Connect **Feather or Arduino GND** to **board GND (black wire on STEMMA QT version)**

Connect **Feather or Arduino SCL** to **board SCL (yellow wire on STEMMA QT version)**

Connect **Feather or Arduino SDA** to **board SDA (blue wire on STEMMA QT version)**

The final results should resemble the illustration above, showing an Adafruit Metro development board.

Installation

You can install the **Adafruit VEML7700 Library** for Arduino using the Library Manager in the Arduino IDE:
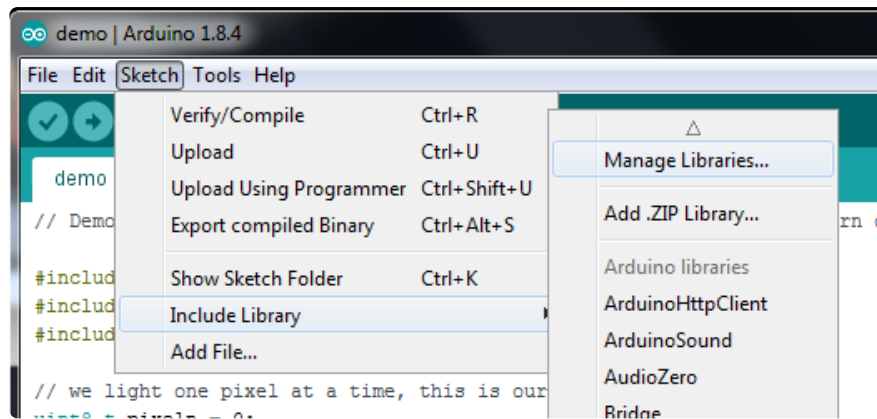
Click the **Manage Libraries ...** menu item, search for **Adafruit VEML7700**, and select the **Adafruit VEML7700** library:
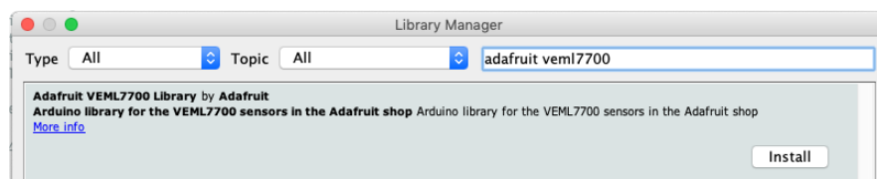
Then follow the same process for the **Adafruit BusIO** library.
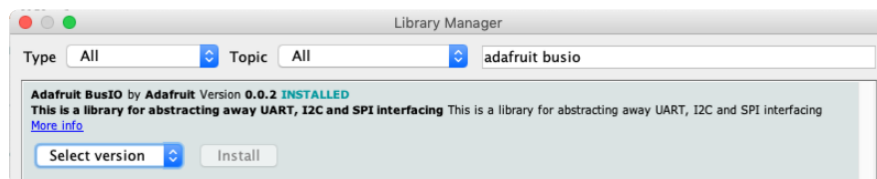
Load Example

Open up **File -> Examples -> Adafruit VEML7700 -> veml7700_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**). You should see the the values for Lux, white light, and raw ambient light levels.

Example Code

The following example code is part of the standard library, but illustrates how you can retrieve sensor data from the VEML7700 for the Lux, white light and raw ambient light values:

```
#include "Adafruit_VEML7700.h"
```

```cpp
Adafruit_VEML7700 veml = Adafruit_VEML7700();

void setup() {
  Serial.begin(115200);
  while (!Serial) { delay(10); }
  Serial.println("Adafruit VEML7700 Test");

  if (!veml.begin()) {
    Serial.println("Sensor not found");
    while (1);
  }
  Serial.println("Sensor found");

  // == OPTIONAL =====
  // Can set non-default gain and integration time to
  // adjust for different lighting conditions.
  // ==================
  // veml.setGain(VEML7700_GAIN_1_8);
  // veml.setIntegrationTime(VEML7700_IT_100MS);

  Serial.print(F("Gain: "));
  switch (veml.getGain()) {
    case VEML7700_GAIN_1: Serial.println("1"); break;
    case VEML7700_GAIN_2: Serial.println("2"); break;
    case VEML7700_GAIN_1_4: Serial.println("1/4"); break;
    case VEML7700_GAIN_1_8: Serial.println("1/8"); break;
  }

  Serial.print(F("Integration Time (ms): "));
  switch (veml.getIntegrationTime()) {
    case VEML7700_IT_25MS: Serial.println("25"); break;
    case VEML7700_IT_50MS: Serial.println("50"); break;
    case VEML7700_IT_100MS: Serial.println("100"); break;
    case VEML7700_IT_200MS: Serial.println("200"); break;
    case VEML7700_IT_400MS: Serial.println("400"); break;
    case VEML7700_IT_800MS: Serial.println("800"); break;
  }

  veml.setLowThreshold(10000);
  veml.setHighThreshold(20000);
  veml.interruptEnable(true);
}

void loop() {
  Serial.print("raw ALS: "); Serial.println(veml.readALS());
  Serial.print("raw white: "); Serial.println(veml.readWhite());
  Serial.print("lux: "); Serial.println(veml.readLux());

  uint16_t irq = veml.interruptStatus();
  if (irq & VEML7700_INTERRUPT_LOW) {
    Serial.println("** Low threshold");
  }
  if (irq & VEML7700_INTERRUPT_HIGH) {
    Serial.println("** High threshold");
  }
  delay(500);
}
```

You should get something resembling the following output when you open the Serial Monitor at 115200 baud:

```
/dev/cu.usbmodem1444220 (Adafruit Metro M0 Express)
                                                          Send
Lux: 443.43
White: 471.85
Raw ALS: 61587
Lux: 443.76
White: 471.85
Raw ALS: 61634
Lux: 444.24
White: 471.85
Raw ALS: 61700
Lux: 444.24
White: 471.85
Raw ALS: 61700
```
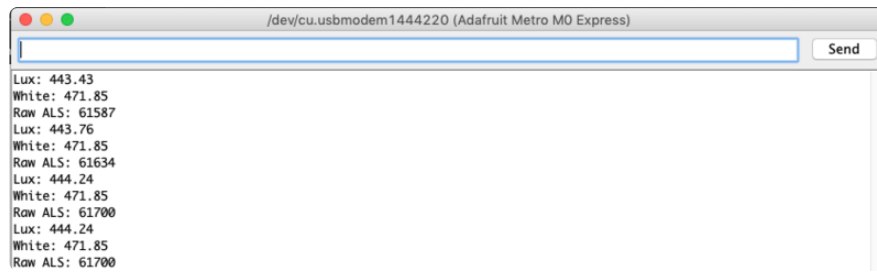
Arduino Docs

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

Python & CircuitPython

It's easy to use the VEML7700 sensor with CircuitPython and the Adafruit CircuitPython VEML7700 (https://adafru.it/EoF) module. This module allows you to easily write Python code that reads the ambient light levels, including Lux, from the 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 VEML7700 to your board exactly as follows. Here is an example of the VEML7700 wired to a Feather using I2C:
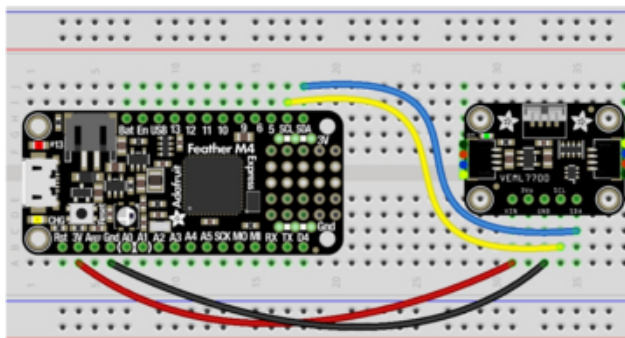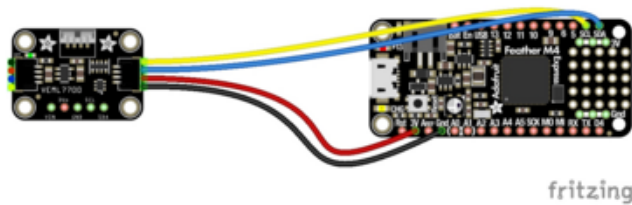
Board 3V to sensor VIN (red wire on STEMMA QT version)

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

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

Board SDA to sensor SDA (blue wire on STEMMA QT version)

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. 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 with I2C:
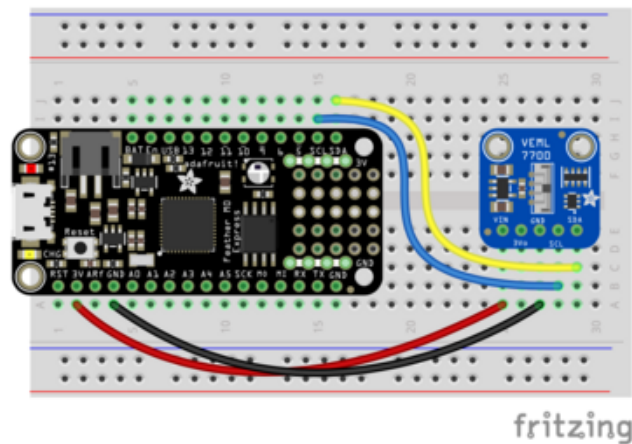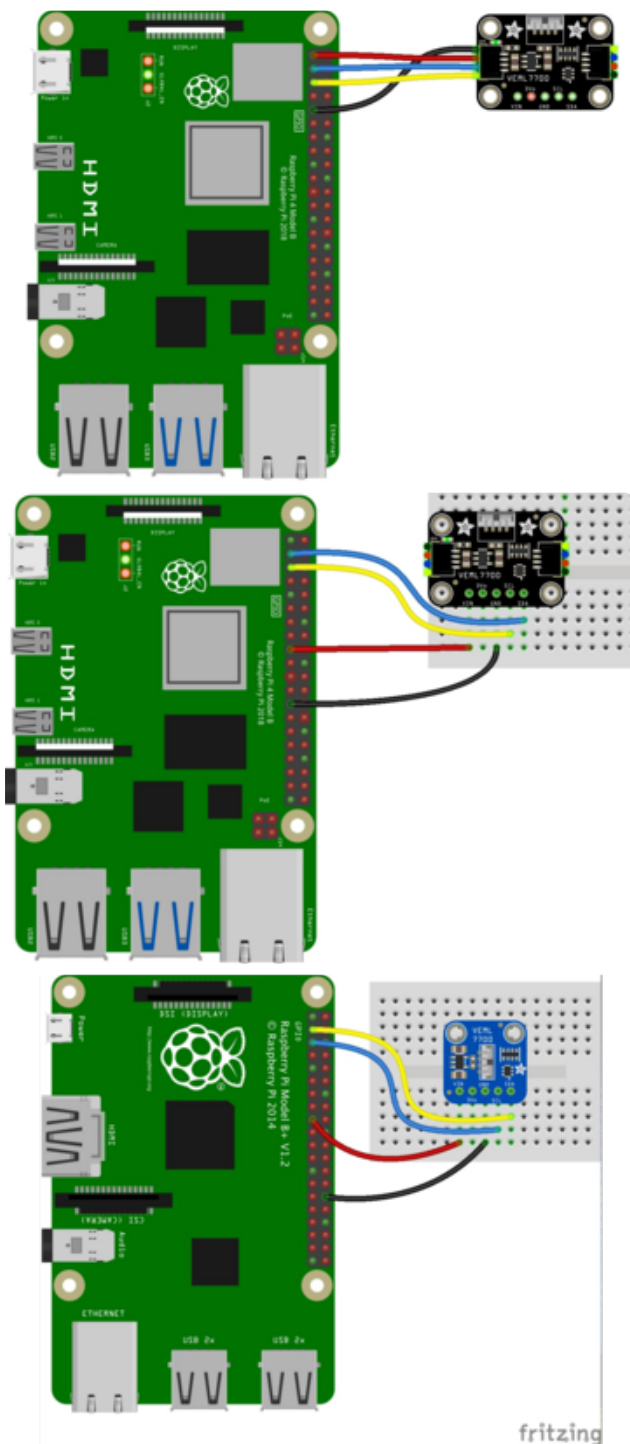
Pi **3V3** to sensor **VIN (red wire on STEMMA QT version)**
Pi **GND** to sensor **GND (black wire on STEMMA QT version)**
Pi **SCL** to sensor **SCL (yellow wire on STEMMA QT version)**
Pi **SDA** to sensor **SDA (blue wire on STEMMA QT version)**

CircuitPython Installation of VEML7700 Library

You'll need to install the Adafruit CircuitPython VEML7700 (https://adafru.it/EoF) library on your CircuitPython board.

First make sure you are running the latest version of Adafruit CircuitPython (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). Our CircuitPython starter guide has a great page on how to install the library bundle (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_veml7700.mpy**
- **adafruit_bus_device**
- **adafruit_register**

Before continuing make sure your board's lib folder or root filesystem has the **adafruit_veml7700.mpy, adafruit_bus_device**, and **adafruit_register** files and folders copied over.

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

Python Installation of VEML7700 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:

- `sudo pip3 install adafruit-circuitpython-veml7700`

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 ambient light levels 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_veml7700

i2c = busio.I2C(board.SCL, board.SDA)
veml7700 = adafruit_veml7700.VEML7700(i2c)
```

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

- **light** - The ambient light data.
- **lux** - The light levels in Lux.

For example to print ambient light levels and lux values:

```
print("Ambient light:", veml7700.light)
print("Lux:", veml7700.lux)
```


For more details, check out the library documentation (https://adafru.it/EoQ).

That's all there is to using the VEML7700 sensor with CircuitPython!

Full Example Code

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

import time
import board
import adafruit_veml7700

i2c = board.I2C()  # uses board.SCL and board.SDA
# i2c = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a
microcontroller
veml7700 = adafruit_veml7700.VEML7700(i2c)

while True:
    print("Ambient light:", veml7700.light)
    time.sleep(0.1)
```
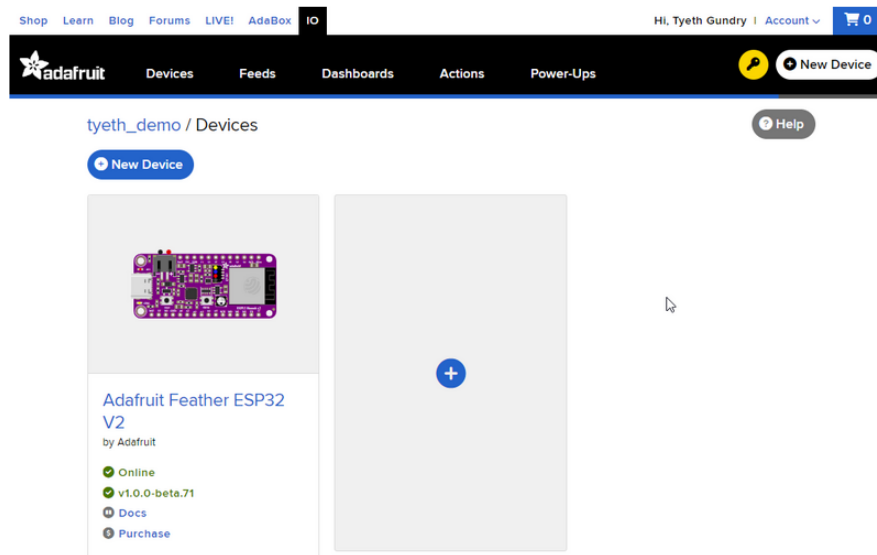
Python Docs

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

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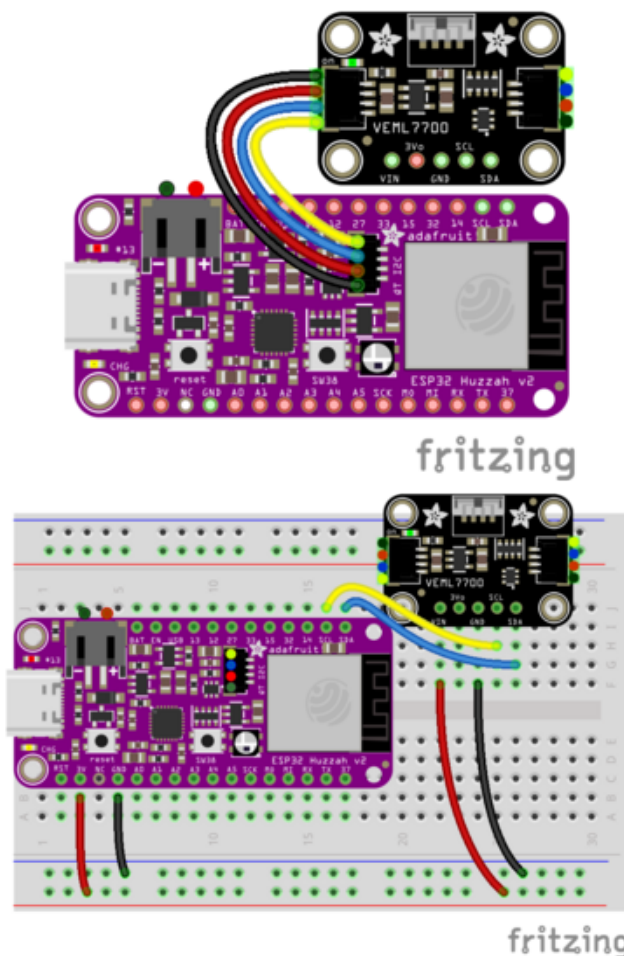
If you've never used WipperSnapper, click below to read through the quick start guide before continuing.

Wiring

First, wire up an VEML7700 to your board exactly as follows. Here is an example of the VEML7700 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)**
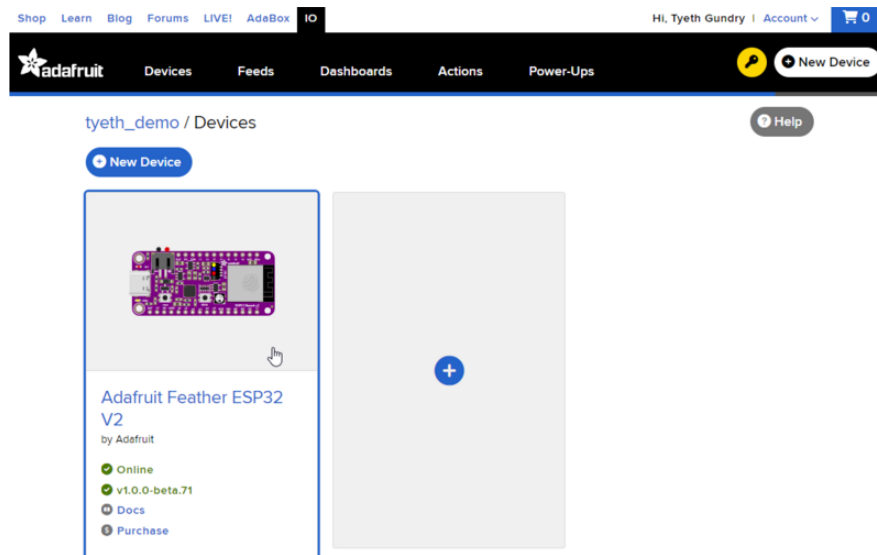**Board SCL** to **sensor SCL (yellow wire on STEMMA QT)**
Board SDA to **sensor SDA (blue wire on STEMMA QT)**

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) first.
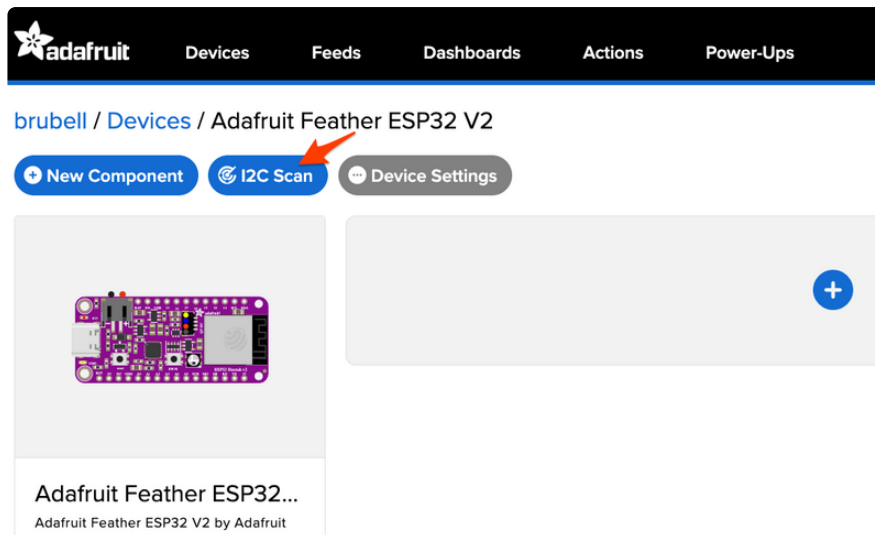

On the device page, quickly **check that you're running the latest version of the WipperSnapper firmware**.

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) 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 VEML7700's default I2C address of `0x10` pop-up in the I2C scan list.

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 `VEML7700` into the search bar, then select the **VEML7700** component.
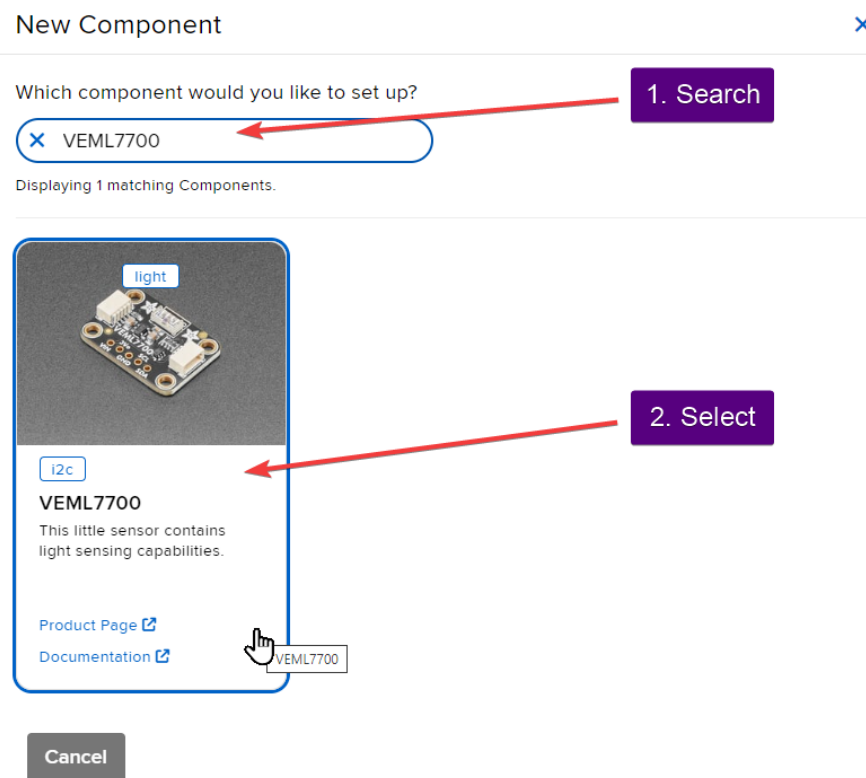
On the component configuration page, the VEML7700's sensor address should be listed along with the sensor's settings.

The **Send Every** option is specific to each sensor's measurements. This option will tell the Feather how often it should read from the VEML7700 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.

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.
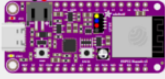

To view the data that has been logged from the sensor, click on the graph next to the sensor name.

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

Adjusting for Different Light Levels

While the VEML7700 is capable of being used over a wide range of ambient lighting conditions, from dark room to direct sunlight, getting good readings will typically require making some adjustments.

There are two main adjustments:

- **gain** - how much the signal is amplified
- **integration time** - how long the signal is built up

This Application Note from Vishay, the manufacturer of the VEML7700, contains good information:

<div align="center">

Designing the VEML7700 Into an Application

https://adafru.it/-IA

</div>

This table summarizes the maximum illumination (in terms of lux) that can be measured for given combinations of gain (**GAIN**) and integration time (**IT**).

IT (ms)	GAIN 2	GAIN 1	GAIN 1/4	GAIN 1/8
	MAXIMUM POSSIBLE ILLUMINATION			
800	236	472	1887	3775
400	472	944	3775	7550
200	944	1887	7550	15 099
100	1887	3775	15 099	30 199
50	3775	7550	30 199	60 398
25	7550	15 099	60 398	120 796

If the maximum illumination is known, then the gain and integration times can be manually set accordingly. In general, the raw ALS value (different than lux values in table) should be between 100 and 10000. Read the Application Note for details.

NOTE: The library defaults are **GAIN=1/8** and **IT=100ms**.

Non-Linear Correction

For lower light levels, the VEML7700 behaves linearly, so lux can be computed simply from the raw ALS reading. For higher levels, the behavior becomes increasingly non-linear. For those conditions, the Application Note provides a non-linear correction that can be applied to the computed lux value.

The Application Note seems to indicate that If the raw ALS value is over 100 with a gain of 1/8 and integration time of 100ms, then the correction should be applied.

The VEML7700 shows good linear behavior for lux levels from 0.0036 lx to about 1 klx.

A software flow may look like the flow chart diagram at the end of this note:

- Starting with the lowest gain (gain x 1/8), check the ALS counts. If ≤ 100 counts, increase the gain to 1/4.
- Check the ALS counts again. If they are still ≤ 100 counts, increase the gain to 1.
- Check the ALS counts again. If they are still ≤ 100 counts, increase the gain to 2.
- Check the ALS counts again. If they are still ≤ 100 counts, increase the integration time from 100 ms to 200 ms, and continue the procedure up to the longest integration time of 800 ms.

If the illumination value is > 100 counts (started with gain x 1/8), a correction formula may be applied to get rid of small non-linearity for very high light levels.

Although it's a little confusing as to what integration times this applies to.

Automatic Adjustment

The Application Note linked above provides a flow chart (Fig. 24) that outlines a method to automatically adjust gain and integration time based on the raw sensor readings. Additionally, the non-linear correction is applied as needed.

Currently, only the Arduino library provides an implementation of this. See the example here:

> ### veml7700_autolux.ino
>
> https://adafru.it/-lB

However, one could implement the same thing in their own code. The libraries provide the necessary methods for setting gain and integration time and reading the raw ALS values.

Downloads

Files

- VEML7700 Datasheet (https://adafru.it/EoR)
- EagleCAD files on GitHub (https://adafru.it/EoS)
- 3D Models of Right-Angled Sensor on GitHub (https://adafru.it/11sb)
- Fritzing object for STEMMA QT version in Adafruit Fritzing Library (https://adafru.it/Ytb)
- Fritzing object for original version from Adafruit Fritzing Library (https://adafru.it/EoT)

Schematic and Fab Print for STEMMA QT Version
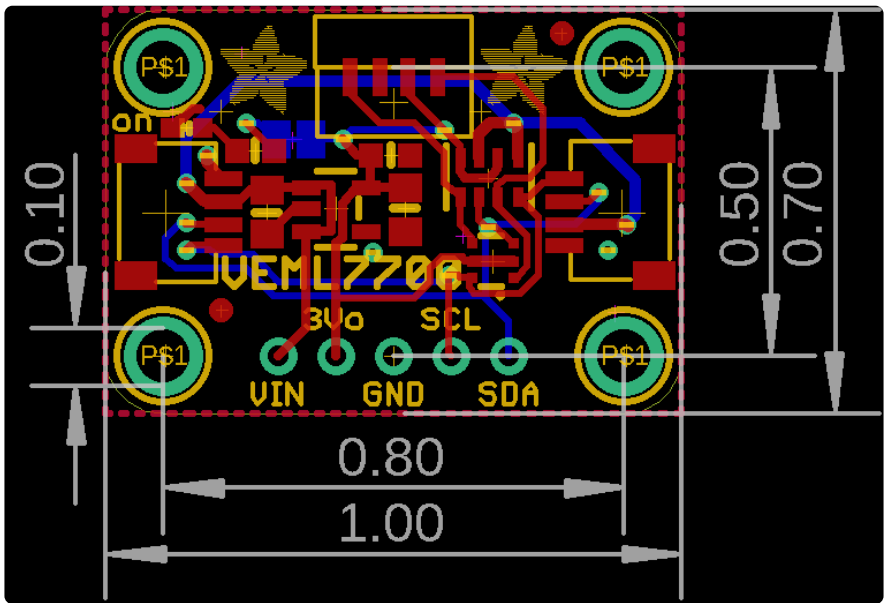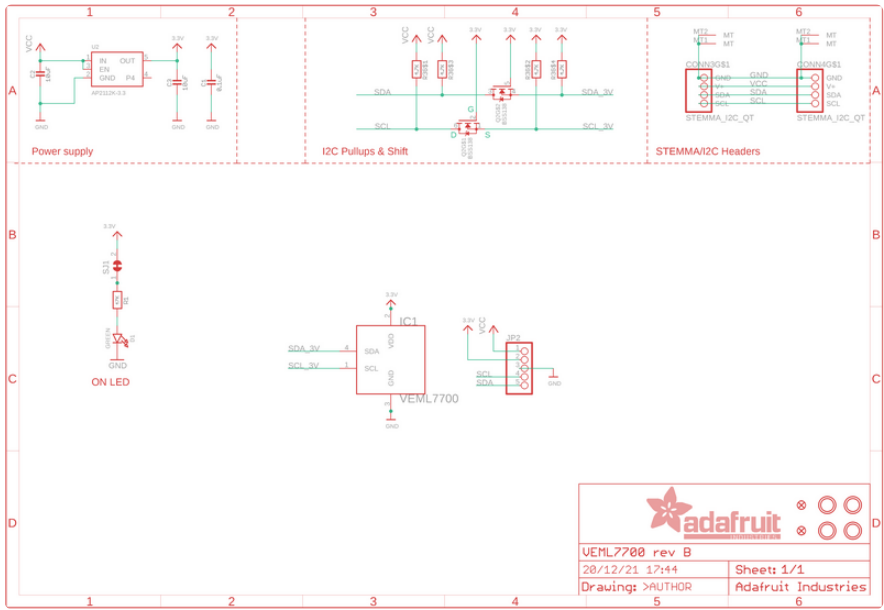
Schematic and Fab Print for Original Version