Adafruit LTR390 UV Sensor

Created by Abigail Torres

https://learn.adafruit.com/adafruit-ltr390-uv-sensor
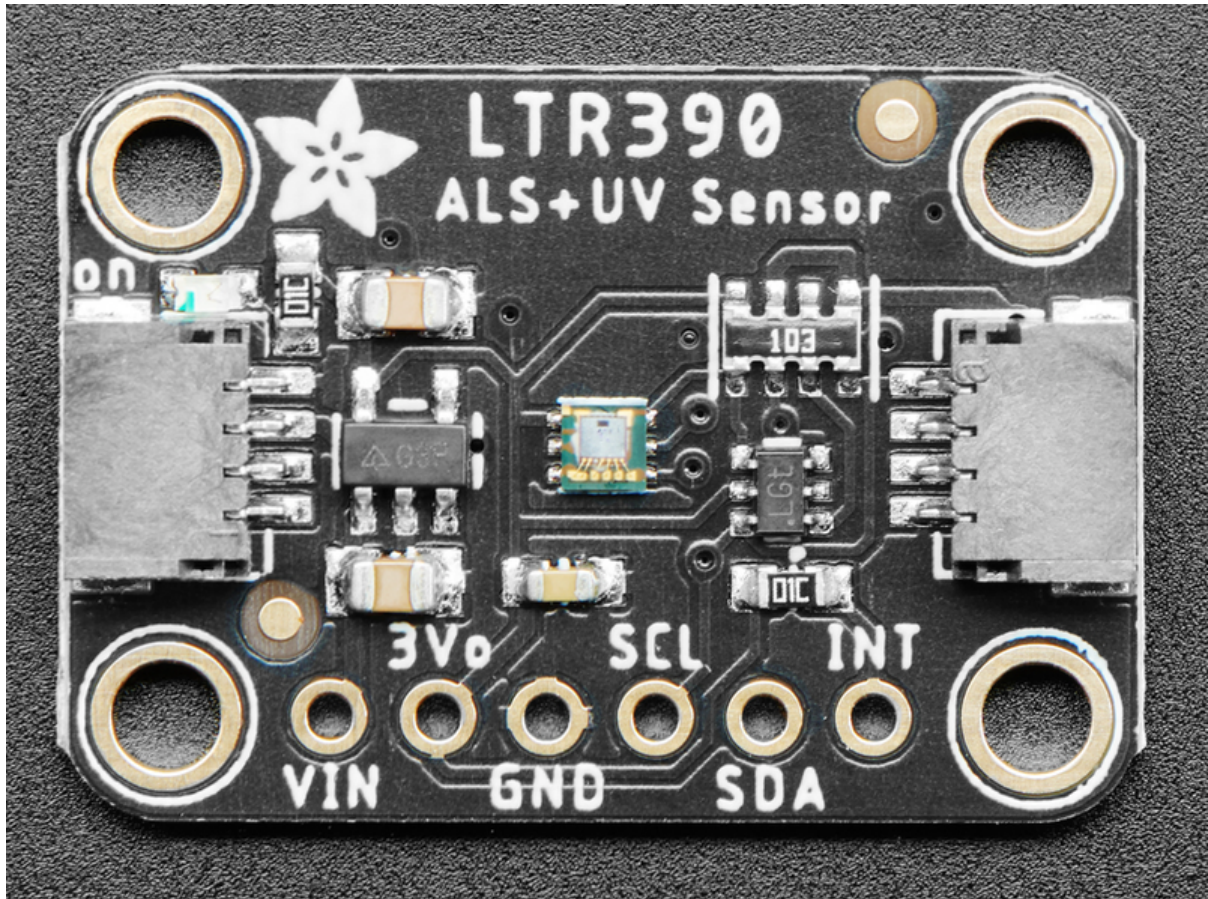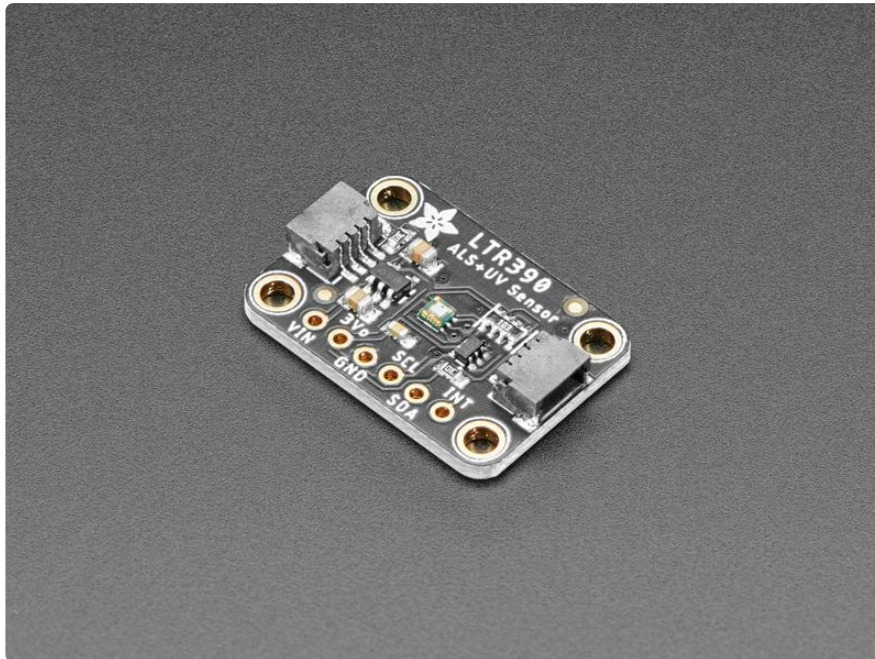
Last updated on 2024-01-05 01:46:11 PM EST

Table of Contents

Overview

The LTR390 () is one of the few low-cost UV sensors available, and it's a pretty nice one! With both ambient light and UVA sensing with a peak spectral response between 300 and 350nm. You can use it for measuring how much sun you can get before needing to covering up.
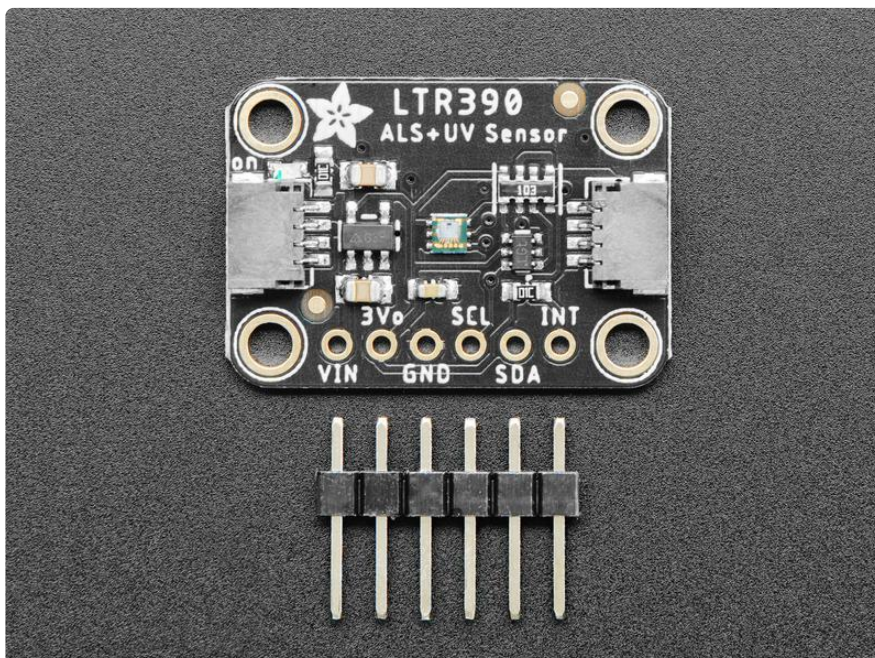
Unlike sensors that estimate UV Index readings from visible and IR light levels such as the Si1145 (http://adafru.it/1777), the LTR390 incorporates two sensors, one for visible and another specifically designed to measure UV light levels. The sensor's

CircuitPython library includes routines to derive the UV Index value from raw UV measurements as well as calculating the ambient light Lux level.

The LTR390 also has a much much simpler I2C interface so you can run it on the Arduino or Python microcontrollers/microcomputers with ease. Unlike the GUVA analog sensor (http://adafru.it/1918), the biasing and ADC is all internal so you don't need an ADC.
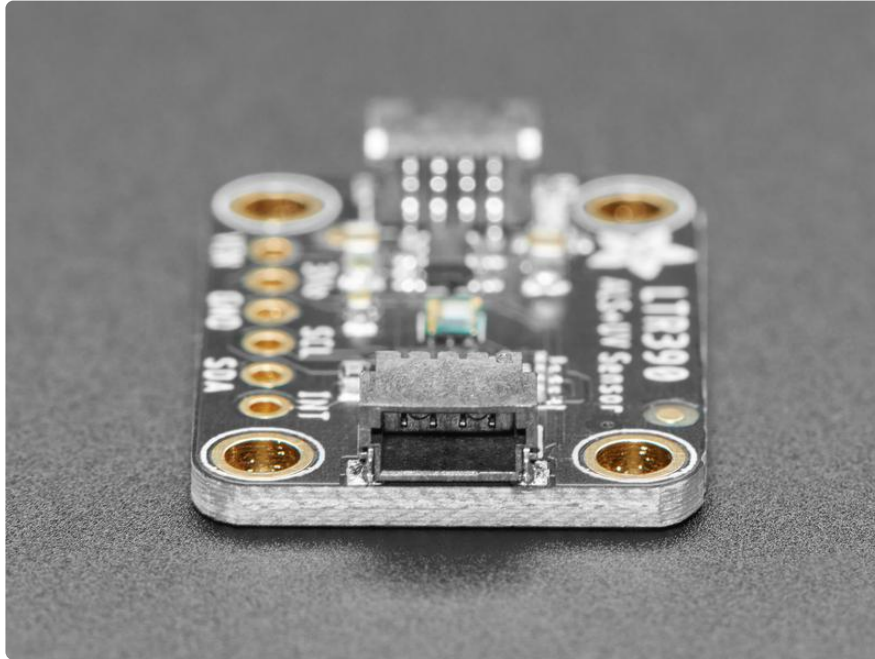
To make using it as easy as possible, we've put the LTR390 on a breakout PCB in our Stemma QT form factor () with a sprinkle of support circuitry to give you options when testing. You can either use a breadboard or the SparkFun qwiic () compatible ST EMMA QT () connectors, and compatibility with 5V voltage levels as commonly found on Arduinos (), as well as 3.3V logic used by many other boards like the Raspberry Pi or our Feathers. QT Cable is not included, but we have a variety in the shop () for quick plug-and-play support
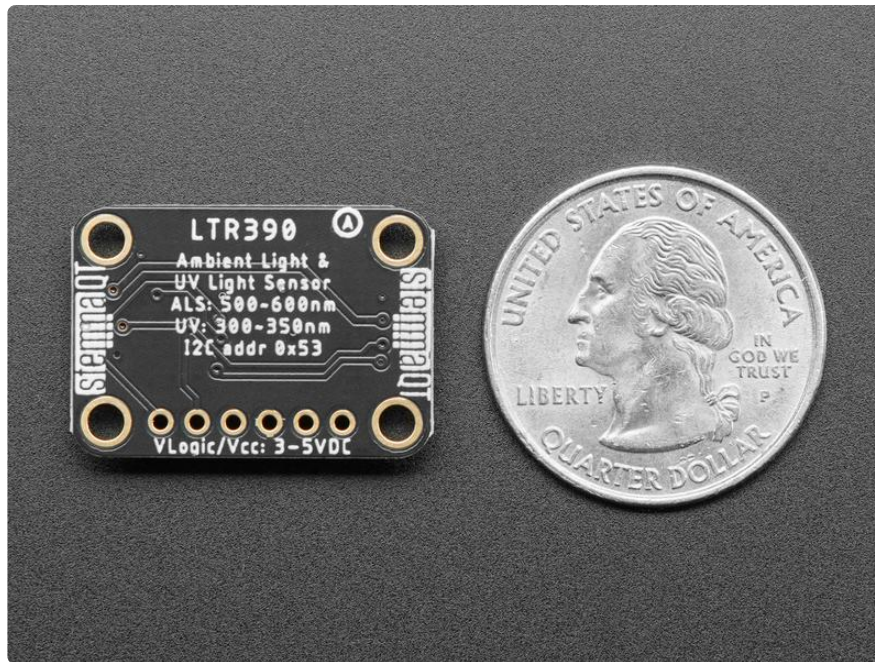
Pinouts
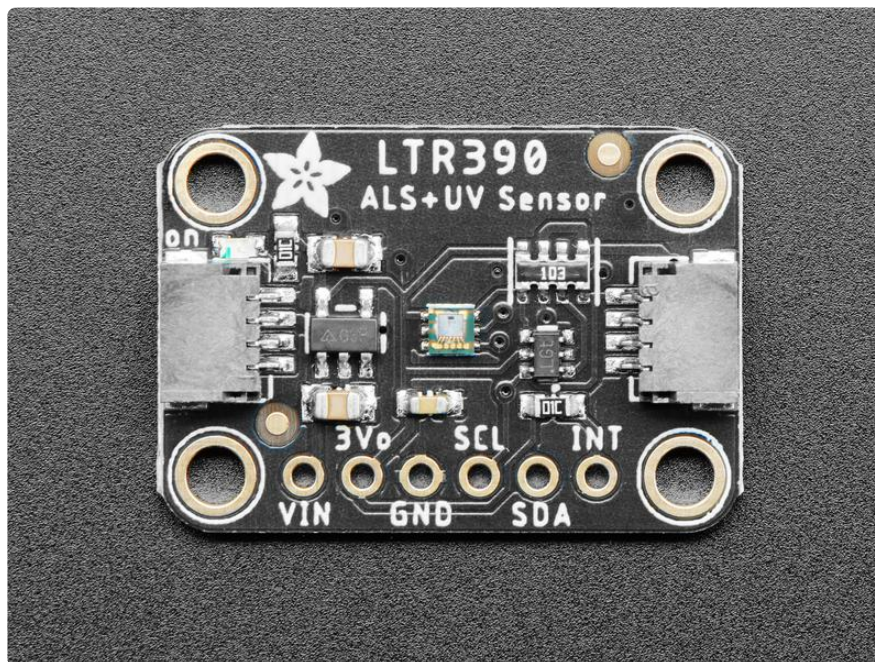
Power Pins

- 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 microcontroller 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 - I2C clock pin, connect to your microcontroller I2C clock line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
- SDA - I2C data pin, connect to your microcontroller I2C data line. This pin is level shifted so you can use 3-5V logic, and there's a 10K pullup on this pin.
- STEMMA QT () - These connectors allow you to connectors to dev boards with S TEMMA QT connectors or to other things with various associated accessories ()

Other Pins

INT -This is the primary interrupt pin. You can setup the LTR390 to pull this low when certain conditions are met such as new measurement data being available. Consult the datasheet () for usage

Arduino

Using the LTR390 with Arduino is a simple matter of wiring up the sensor to your Arduino-compatible microcontroller, installing the Adafruit LTR390 () library we've written, and running the provided example code.

I2C Wiring

Here is how to wire up the sensor using one of the STEMMA QT () connectors. The examples show a Metro but wiring will work the same for an Arduino or other compatible board.
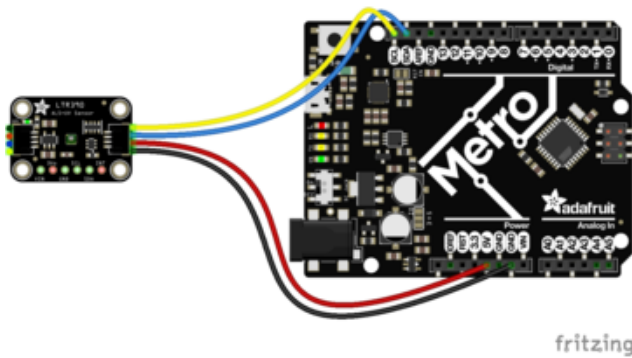
Connect board VIN (red wire) to Arduino 5V if you are running a 5V board Arduino (Uno, etc.). If your board is 3V, connect to that instead.
Connect board GND (black wire) to Arduino GND
Connect board SCL (yellow wire) to Arduino SCL
Connect board SDA (blue wire) to Arduino SDA

Here is how to wire the sensor to a board using a solderless breadboard:
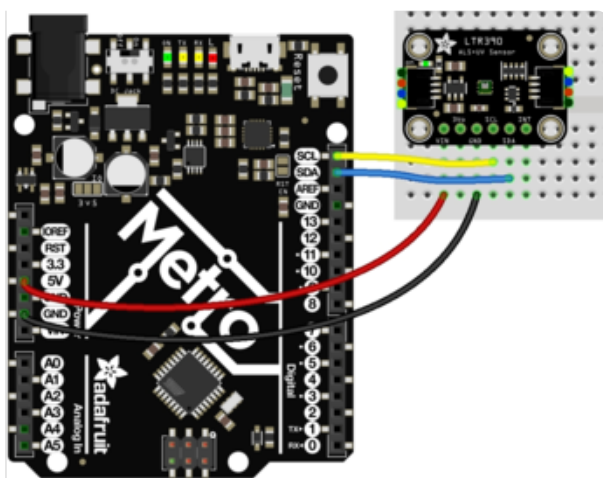
Connect board VIN (red wire) to Arduino 5V if you are running a 5V board Arduino (Uno, etc.). If your board is 3V, connect to that instead.
Connect board GND (black wire) to Arduino GND
Connect board SCL (yellow wire) to Arduino SCL
Connect board SDA (blue wire) to Arduino SDA

Library Installation

You can install the Adafruit LTR390 library for Arduino using the Library Manager in the Arduino IDE.

Click the Manage Libraries ... menu item, search for Adafruit LTR390 , and select the A dafruit LTR390 library:

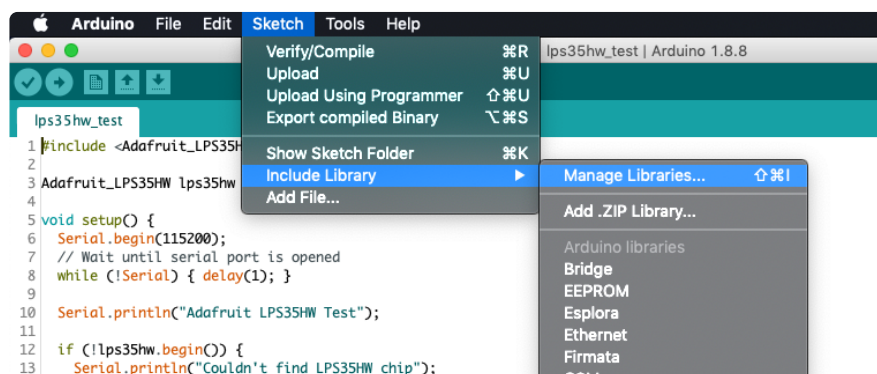Next, follow the same process for the Adafruit BusIO library.
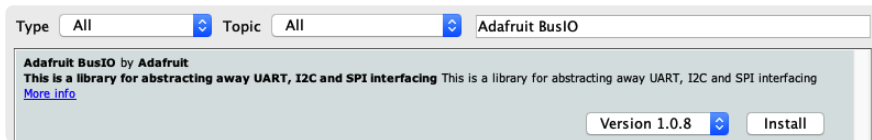
Load Example

Open up File -> Examples -> Adafruit LTR390 -> LTR390_test
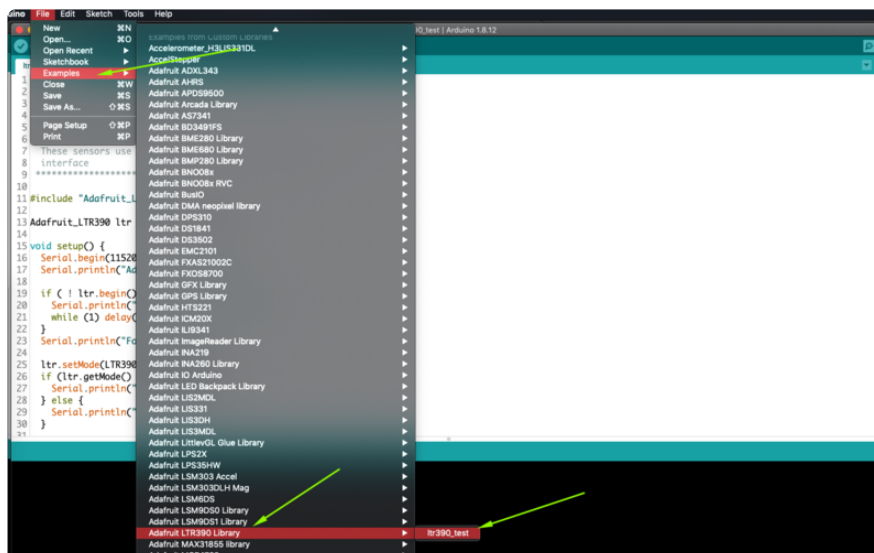
After opening the demo file, upload to your Arduino wired up to the sensor. Once you upload the code, you will see the UV data values being printed when you open the Serial Monitor (Tools->Serial Monitor) at 115200 baud, similar to this:

Example Code

```
/*************************************************
  This is an example for the LTR390 UV Sensor

  Designed specifically to work with the LTR390 UV sensor from Adafruit
  ----> https://www.adafruit.com
```

```
   These sensors use I2C to communicate, 2 pins are required to
   interface
 *****************************************************/

#include "Adafruit_LTR390.h"

Adafruit_LTR390 ltr = Adafruit_LTR390();

void setup() {
  Serial.begin(115200);
  Serial.println("Adafruit LTR-390 test");

  if ( ! ltr.begin() ) {
    Serial.println("Couldn't find LTR sensor!");
    while (1) delay(10);
  }
  Serial.println("Found LTR sensor!");

  ltr.setMode(LTR390_MODE_UVS);
  if (ltr.getMode() == LTR390_MODE_ALS) {
    Serial.println("In ALS mode");
  } else {
    Serial.println("In UVS mode");
  }

  ltr.setGain(LTR390_GAIN_3);
  Serial.print("Gain : ");
  switch (ltr.getGain()) {
    case LTR390_GAIN_1: Serial.println(1); break;
    case LTR390_GAIN_3: Serial.println(3); break;
    case LTR390_GAIN_6: Serial.println(6); break;
    case LTR390_GAIN_9: Serial.println(9); break;
    case LTR390_GAIN_18: Serial.println(18); break;
  }

  ltr.setResolution(LTR390_RESOLUTION_16BIT);
  Serial.print("Resolution : ");
  switch (ltr.getResolution()) {
    case LTR390_RESOLUTION_13BIT: Serial.println(13); break;
    case LTR390_RESOLUTION_16BIT: Serial.println(16); break;
    case LTR390_RESOLUTION_17BIT: Serial.println(17); break;
    case LTR390_RESOLUTION_18BIT: Serial.println(18); break;
    case LTR390_RESOLUTION_19BIT: Serial.println(19); break;
    case LTR390_RESOLUTION_20BIT: Serial.println(20); break;
  }

  ltr.setThresholds(100, 1000);
  ltr.configInterrupt(true, LTR390_MODE_UVS);
}

void loop() {
  if (ltr.newDataAvailable()) {
      Serial.print("UV data: ");
      Serial.print(ltr.readUVS());
  }

  delay(100);
}
```

Arduino Docs

Arduino Docs ()

Python & CircuitPython

It's easy to use the LTR390 with Python or CircuitPython, and the Adafruit CircuitPython LTR390 () module. This module allows you to easily write Python code that reads uv and ambient light from the LTR390 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 ().

CircuitPython Microcontroller Wiring

First wire up a LTR390 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 () connectors:
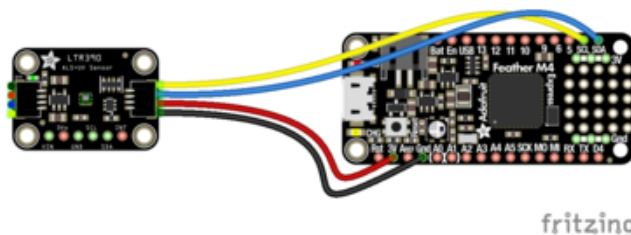
Board 3V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
Board SDA to sensor SDA (blue wire)

You can also use the standard 0.100" pitch headers to wire it up on a breadboard:
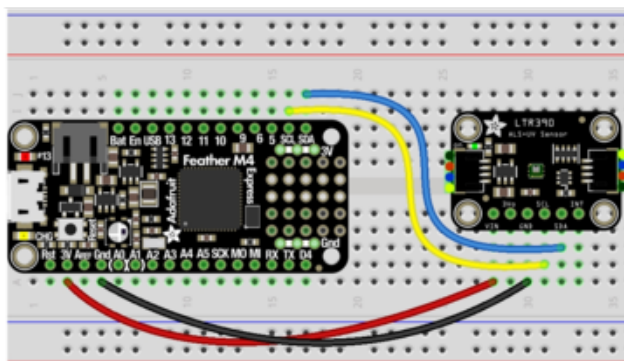
Board 3V to sensor VIN (red wire)
Board GND to sensor GND (black wire)
Board SCL to sensor SCL (yellow wire)
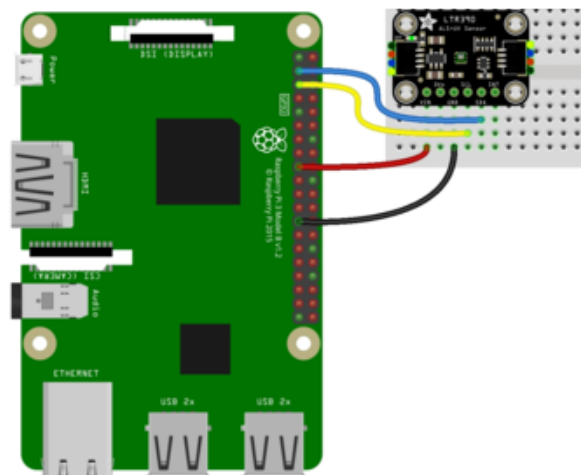Board SDA to sensor SDA (blue wire)

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 ().

Here's the Raspberry Pi wired to the sensor using I2C and a STEMMA QT () connector:
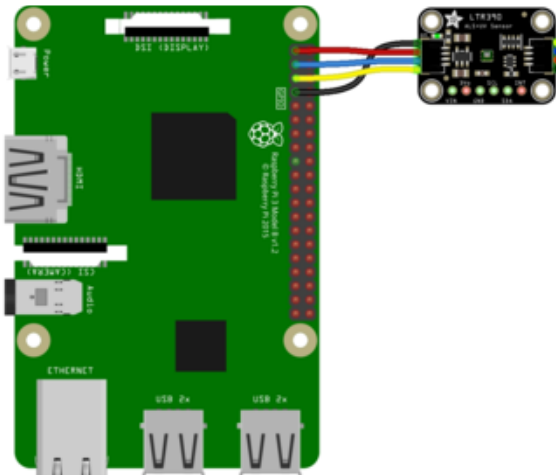
Pi 3V to sensor VCC (red wire)
Pi GND to sensor GND (black wire)
Pi SCL to sensor SCL (yellow wire)
Pi SDA to sensor SDA (blue wire)

Finally here is an example of how to wire up a Raspberry Pi to the sensor using a solderless breadboard

Pi 3V to sensor VCC (red wire)
Pi GND to sensor GND (black wire)
Pi SCL to sensor SCL (yellow wire)
Pi SDA to sensor SDA (blue wire)

CircuitPython Installation of LTR390 Library

You'll need to install the Adafruit CircuitPython LTR390 () library on your CircuitPython board.

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

Before continuing make sure your board's lib folder or root filesystem has the adafruit _LTR390.mpy file and the adafruit_register and adafruit_bus_device folders copied over.

When done, your CIRCUITPY drive should look like this:
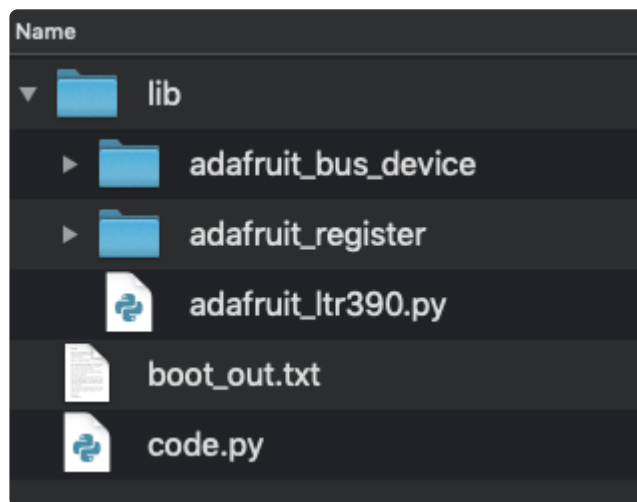
Next connect to the board's serial REPL ()so you are at the CircuitPython >>> prompt.

Python Installation of LTR390 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 ()!

Once that's done, from your command line run the following command:

- `sudo pip3 install adafruit-circuitpython-ltr390`

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 UV and ambient light 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 board
import busio
import adafruit_ltr390

i2c = busio.I2C(board.SCL, board.SDA)

ltr = adafruit_ltr390.LTR390(i2c)
```

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

- uvs - The raw UV light measurement.
- light - The raw ambient light measurement.
- uvi - The calculated UV Index value.
- lux - The calculated Lux ambient light value.

```
>>> import board
>>> import busio
>>> import adafruit_ltr390
>>>
>>> i2c = busio.I2C(board.SCL, board.SDA)
>>>
>>> ltr = adafruit_ltr390.LTR390(i2c)
```

```
print("UV:", ltr.uvs, "\t\tAmbient Light:", ltr.light)
```

```
>>> print("UV:", ltr.uvs, "\t\tAmbient Light:", ltr.light)
UV: 2          Ambient Light: 2040
```

```
print("UV Index:", ltr.uvi, "\t\tLux:", ltr.lux)
```

```
>>> print("UV Index:", ltr.uvi, "\t\tLux:", ltr.lux)
UV Index: 0.0834783          Lux: 1628.0
```

Example Code

```
# SPDX-FileCopyrightText: 2021 by Bryan Siepert, written for Adafruit Industries
#
# SPDX-License-Identifier: Unlicense
import time
import board
import adafruit_ltr390

i2c = board.I2C()  # uses board.SCL and board.SDA
```

```
# i2c = board.STEMMA_I2C()  # For using the built-in STEMMA QT connector on a
microcontroller
ltr = adafruit_ltr390.LTR390(i2c)

while True:
    print("UV:", ltr.uvs, "\t\tAmbient Light:", ltr.light)
    print("UVI:", ltr.uvi, "\t\tLux:", ltr.lux)
    time.sleep(1.0)
```
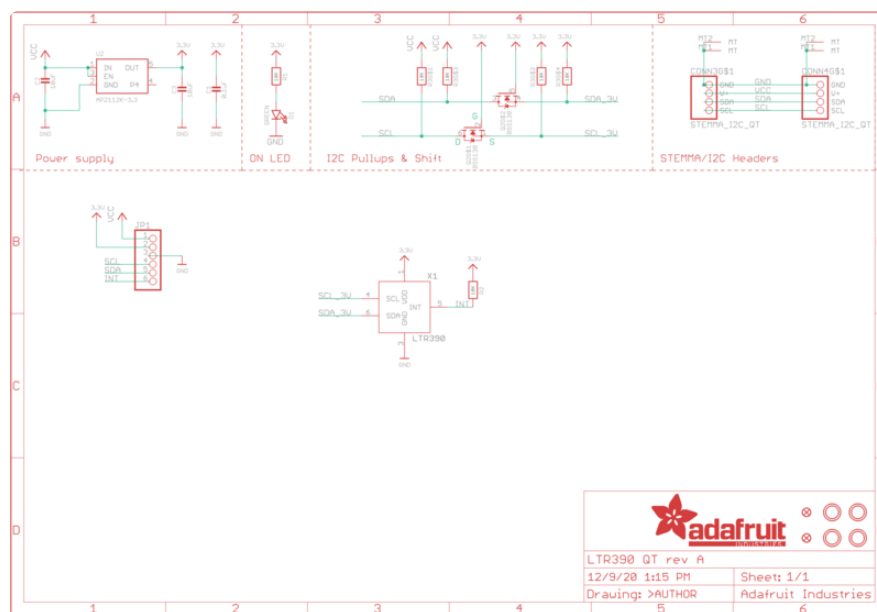
Python Docs

[Python Docs ()](#)

Downloads

Files

- [LTR390 Datasheet ()](#)
- [EagleCAD files on GitHub ()](#)
- [Fritzing object in the Adafruit Fritzing Library ()](#)

Schematic

Fab Print