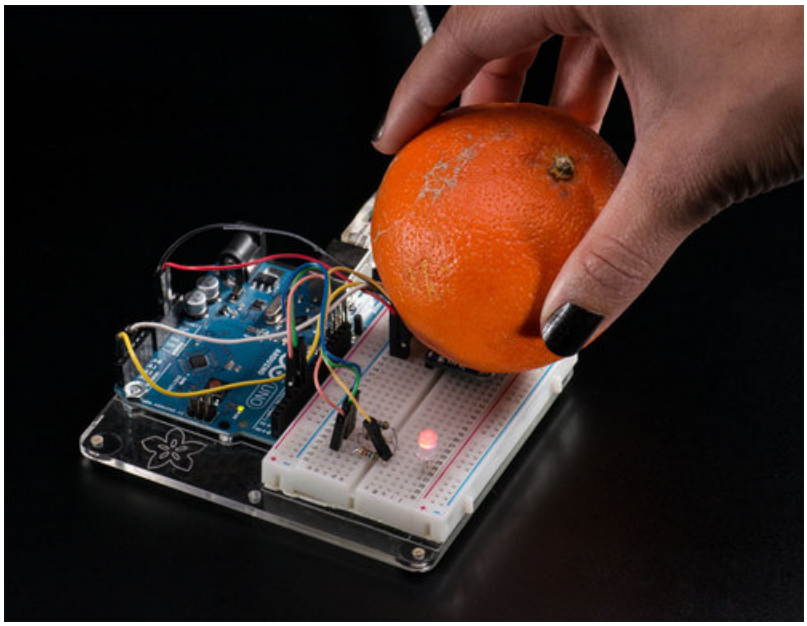
adafruit learning system

Adafruit Color Sensors

Created by Bill Earl



Last updated on 2018-11-05 03:48:12 PM UTC

Guide Contents

Guide Contents 2
Overview 3
Assembly and Wiring 5
Assembly (breakout version only) 5
- Position the header 5
- Position the Breakout 6
- And Solder! 7

Wiring 7
- Flora Wiring: 7
- Arduino Wiring: 8
- To control the LED 8

Python & CircuitPython 9
CircuitPython Microcontroller Wiring 9
Python Computer Wiring 9
CircuitPython Installation of TCS34725 Library 9
Python Installation of TCS34725 Library 10
CircuitPython & Python Usage 10
Full Example Code 11
Python Docs 12
Arduino Code 13
Download Adafruit_TCS34725 13
- Test the Sensor 13
- ColorView! 14
- ColorView Components 15
- ColorView Wiring 16

Library Reference 17
Construction and Initialization: 17
Gain and Integration Time: 17
Light Readings and Calculations: 17
Interrupts and LED control: 18
Use it with Processing! 19
- Load ColorView on the Arduino 19
- Load ColorView.pde in Processing 19
- Edit the Serial Port 20
- And Run! 21

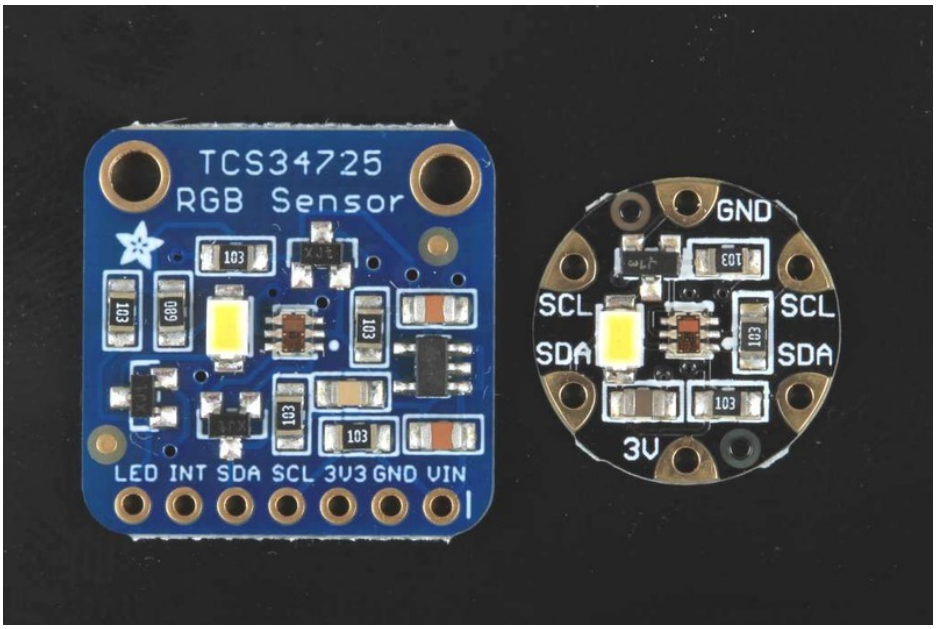
Downloads 22
Files 22
Breakout Board Version 22
Flora Sewable Version 23

Overview

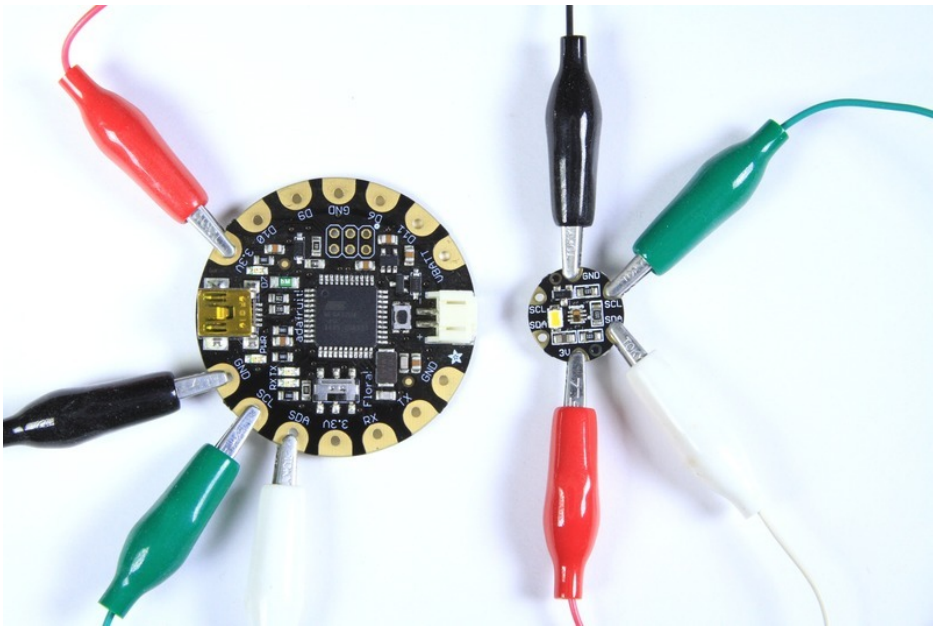
Your electronics can now see in dazzling color with this lovely color light sensor. We found the best color sensor on the market, the TCS34725, which has RGB and Clear light sensing elements. An IR blocking filter, integrated on-chip and localized to the color sensing photodiodes, minimizes the IR spectral component of the incoming light and allows color measurements to be made accurately. The filter means you'll get much truer color than most sensors, since humans don't see IR. The sensor also has an incredible 3,800,000:1 dynamic range with adjustable integration time and gain so it is suited for use behind darkened glass.

We add supporting circuitry as well, such as a 3.3V regulator so you can power the breakout with 3-5VDC safely and level shifting for the I2C pins so they can be used with 3.3V or 5V logic. Finally, we specified a nice neutral 4150°K temperature LED with a MOSFET driver onboard to illuminate what you're trying to sense. The LED can be easily turned on or off by any logic level output.

For more flexibility, we've made two different versions of this board: A breadboard-friendly breakout, and a wearable version designed to work with the Flora wearable platform.


TCS34725
RGB Sensor
LED INT SDA SCL 3V3 GND VIN
GND
SCL
SDA
SCL
SDA
3V

Assembly and Wiring

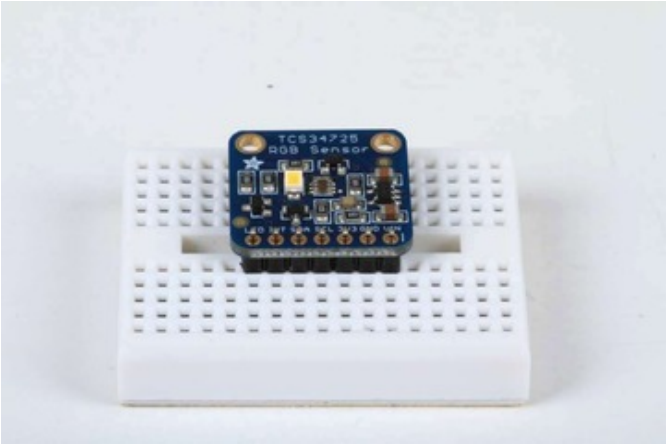


Both color sensors come with all surface mount components pre-soldered. The breakout-board version comes with an optional header for breadboard use. Soldering the header is a simple process:

Assembly (breakout version only)

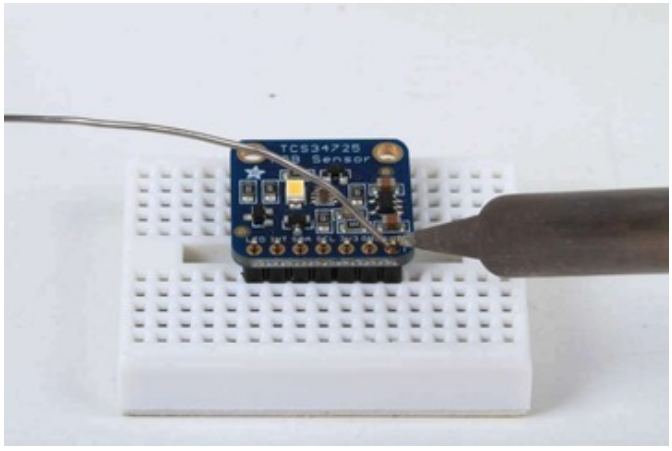
Position the header

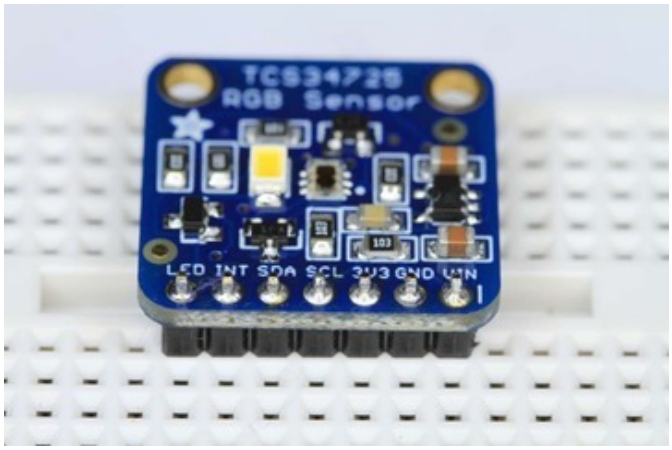
Trim the header to length if necessary and insert it (long pins down) into your breadboard.



Position the Breakout

Place the breakout over the exposed short end of the header pins.



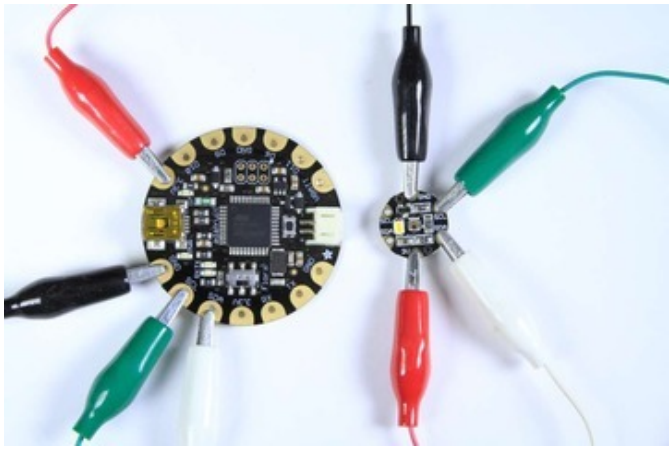


And Solder!

Solder all pins to ensure good electrical contact.

Wiring

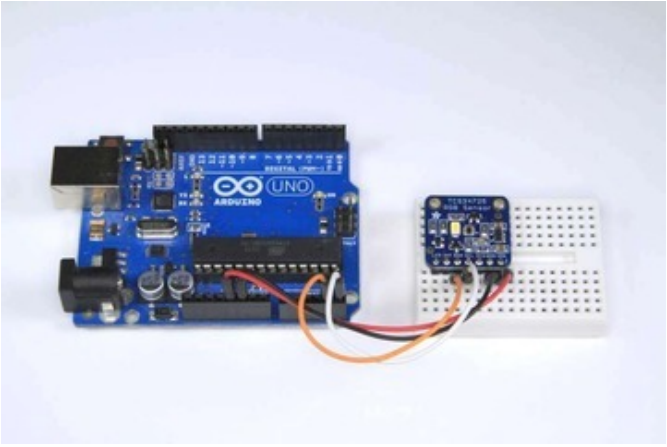
These sensors communicate via a 2-wire I2C interface. To connect to the processor, you need a total of just 4 wires.

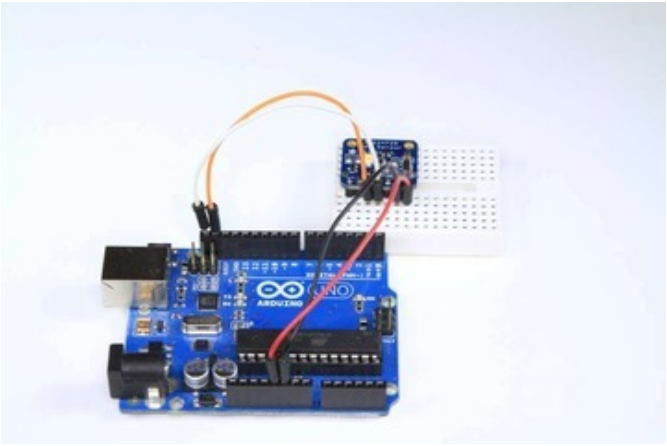


Flora Wiring:

Connect from:

- **3.3v -> 3v (red wire)**
- **GND -> GND (black wire)**
- **SDA -> SDA (white wire)**
- **SCL -> SCL (green wire)**





Arduino Wiring:

Connect jumpers from:

- **5v -> VIN (red wire)**
- **GND -> GND (black wire)**
- **SDA -> SDA (orange wire)**
- **SCL -> SCL (white wire)**

Note: On older Arduinos such as the Duemilanove and pre R3 UNOs, SDA is on Analog 4 and SCL is on Analog 5.

On pre-R2 Megas, SDA is on Digtital 20 and SCL is on digital 21.

For the Leonardo, SDA is digital pin 2 and SCL is digital pin 3.

To control the LED

(Breakout version only) - The LED pin can be pulled low to turn off the LED. This can be done in three ways:

1. Wire directly to ground to turn it off completely.
2. Wire to a spare digital pin and control it with digitalWrite().
3. Wire the LED pin to the INT pin and control with setInterrupt() (See Library Reference for details).

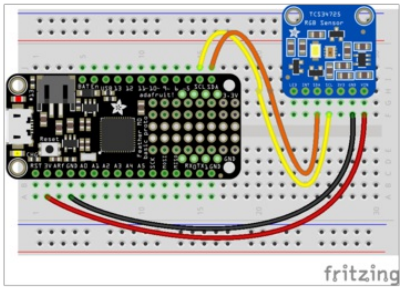
Python & CircuitPython

It's easy to use the TCS34725 sensor with Python and CircuitPython, and the Adafruit CircuitPython TCS34725 (https://adafru.it/C5G) module. This module allows you to easily write Python code that reads the color 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 TCS34725 to your board exactly as shown on the previous pages for Arduino. Here's an example of wiring a Feather M0 to the sensor with an I2C connection:

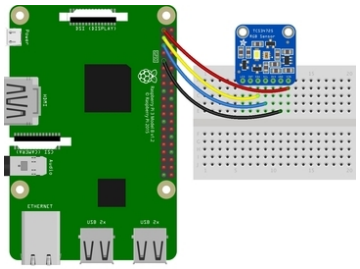


- **Board 3V** to **sensor VIN**
- **Board GND** to **sensor GND**
- **Board SCL** to **sensor SCL**
- **Board SDA** to **sensor SDA**

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**
- **Pi GND** to **sensor GND**
- **Pi SCL** to **sensor SCL**
- **Pi SDA** to **sensor SDA**

CircuitPython Installation of TCS34725 Library

You'll need to install the Adafruit CircuitPython TCS34725 (https://adafru.it/C5G) 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/zdx). Our introduction guide has a great page on how to install the library bundle (https://adafru.it/ABU) for both express and non-express boards.

Remember for non-express boards like the, you'll need to manually install the necessary libraries from the bundle:

- **adafruit_tcs34725.mpy**
- **adafruit_bus_device**

Before continuing make sure your board's lib folder or root filesystem has the **adafruit_tcs34725.mpy,** and **adafruit_bus_device** 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 TCS34725 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-tcs34725`

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 color and more 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_tcs34725
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_tcs34725.TCS34725(i2c)
```

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

- **color_rgb_bytes** - A 3-tuple of the red, green, blue color values. These are returned as bytes from 0 to 255 (0 is low intensity, 255 is maximum intensity).
- **temperature** - The color temperature in Kelvin detected by the sensor. This is computed from the color and might not be super accurate!
- **lux** - The light intensity in lux detected by the sensor. This is computed from the color and might not be super accurate!

```
print('Color: ({0}, {1}, {2})'.format(*sensor.color_rgb_bytes))
print('Temperature: {0}K'.format(sensor.temperature))
print('Lux: {0}'.format(sensor.lux))
```

```
>>> print('Color: ({0}, {1}, {2})'.format(*sensor.color_rgb_bytes))
Color: (37, 14, 4)
>>> print('Temperature: {0}K'.format(sensor.temperature))
Temperature: 2414.75K
>>> print('Lux: {0}'.format(sensor.lux))
Lux: 2174.72
>>>
```

In addition there are some properties you can both read and write to change how the sensor behaves:

- **integration_time** - The integration time of the sensor in milliseconds. Must be a value between 2.4 and 614.4.
- **gain** - The gain of the sensor, must be a value of 1, 4, 16, 60.

```
sensor.integration_time = 200
sensor.gain = 60
```

```
>>> sensor.integration_time = 200
>>> sensor.gain = 60
>>> print('Color: ({0}, {1}, {2})'.format(*sensor.color_rgb_bytes))
Color: (257, 25, 0)
>>> print('Temperature: {0}K'.format(sensor.temperature))
Temperature: 2555.53K
>>> print('Lux: {0}'.format(sensor.lux))
Lux: 3873.08
>>>
```

See the simpletest.py example (https://adafru.it/C5I) for a complete demo of printing the range every second. Save this as **code.py** on the board and examine the REPL output to see the range printed every second.

That's all there is to using the TCS34725 with CircuitPython!

Full Example Code

```
# Simple demo of the TCS34725 color sensor.
# Will detect the color from the sensor and print it out every second.
import time

import board
import busio

import adafruit_tcs34725


# Initialize I2C bus and sensor.
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_tcs34725.TCS34725(i2c)

# Main loop reading color and printing it every second.
while True:
    # Read the color as RGB bytes (0-255 values).
    r, g, b = sensor.color_rgb_bytes
    print('Detected color: #{0:02X}{1:02X}{2:02X}'.format(r, g, b))
    # Read the color temperature and lux of the sensor too.
    try:
        temp = sensor.temperature
        lux = sensor.lux
        print('Temperature: {0}K Lux: {1}'.format(temp, lux))
    except ZeroDivisionError:
        print("No light to measure")
    # Delay for a second and repeat.
    time.sleep(1.0)
```

Python Docs

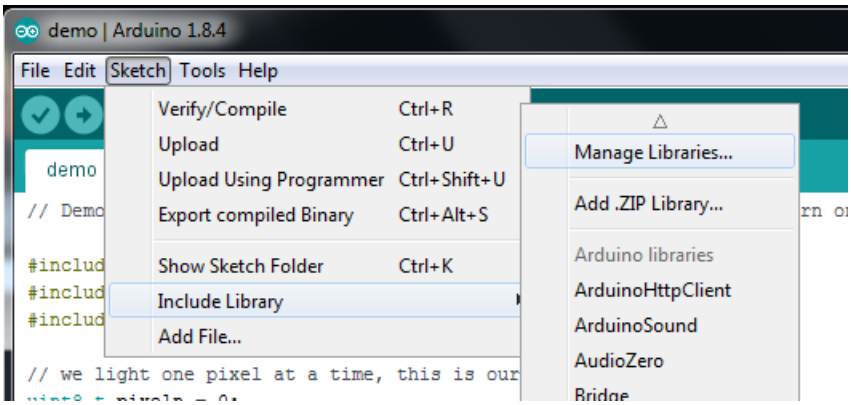
Python Docs (https://adafru.it/C5J)

Arduino Code

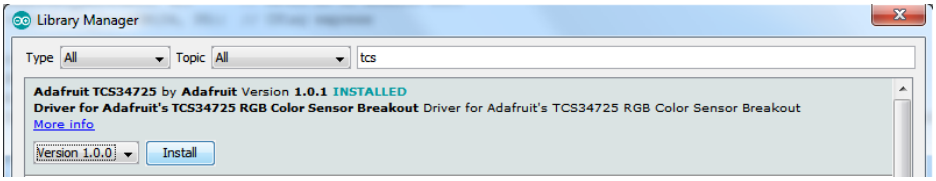
Download Adafruit_TCS34725

To begin reading sensor data, you will need to install the (https://adafru.it/sDz)Adafruit_TCS34725 (https://adafru.it/cb1)

The easiest way to do that is to open up the **Manage Libraries...** menu in the Arduino IDE



Then search for **Adafruit TCS34725** and click **Install**



We also have a great tutorial on Arduino library installation at:
http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use (https://adafru.it/aYM)

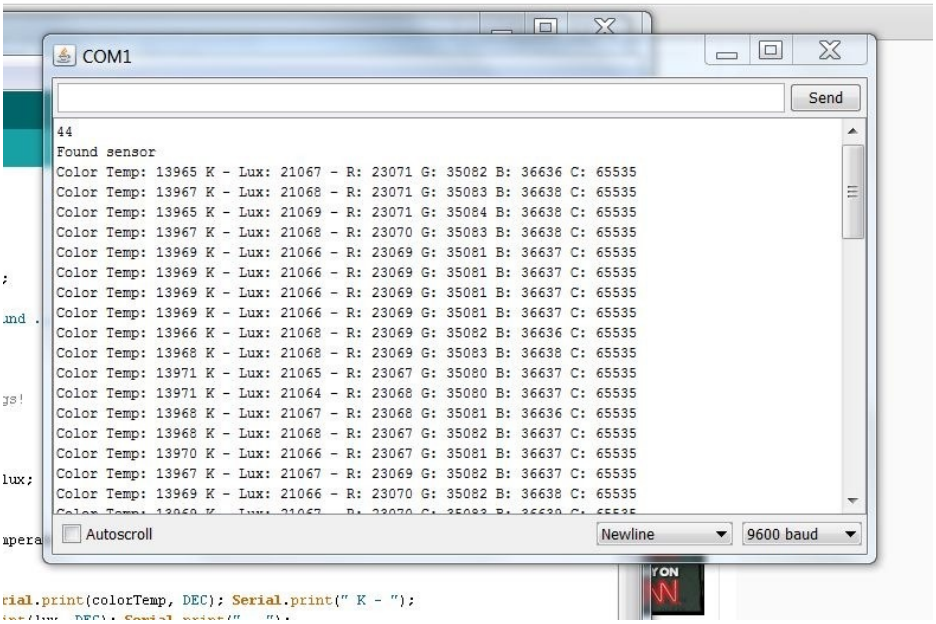
Test the Sensor

Run the TCS34725 test sketch to verify that your sensor is working properly.

Upload the sketch to your Aruduino or Flora and open the Serial Monitor to see the output. The sketch should print out basic color measurement parameters as shown below. Move the sensor around, cover it and/or expose it to different light sources to see how it reacts.

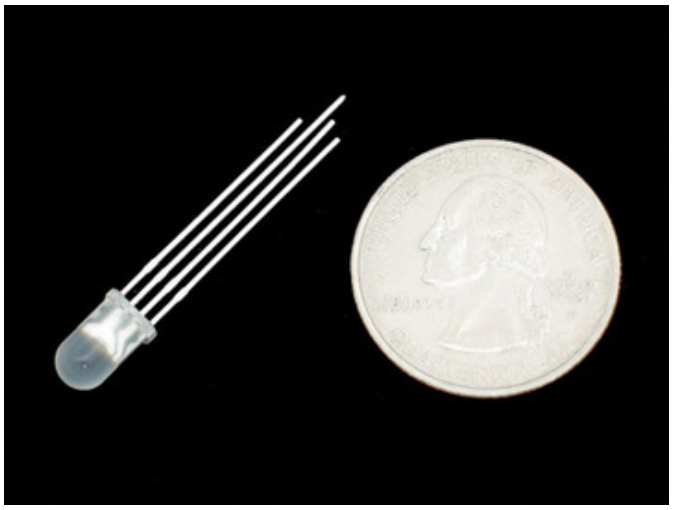
Color parameters reported are:

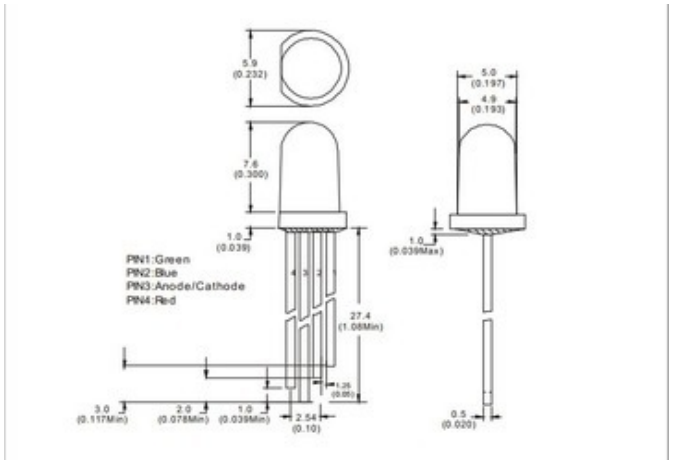
- **Color Temperature** (https://adafru.it/cb3) - measured in Kelvin
- **Lux (https://adafru.it/aKS)** - or Lumens (https://adafru.it/cb4)per Square Meter
- **R, G** and **B** (filtered) values
- **Clear** (unfiltered) value



ColorView!

The ColorView sketch demonstrates reflected-light measurement using the on-board led. The white led is used to illuminate nearby objects and the sensor measures the light reflected from the object. The ColorView sketch then uses the RGB outputs of the sensor to drive an RGB led to match the color that is seen by the sensor!

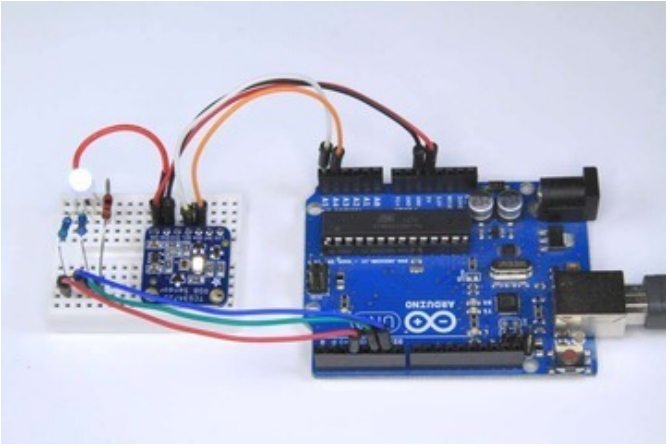




ColorView Components

In addition to a processor and a color sensor, you will need an RGB LED (http://adafru.it/159) and some resistors:

- 1x 1K ohm resistor (Brown, Black Red Gold)
- 2x 560 ohm resistor (Green Blue Brown Gold)

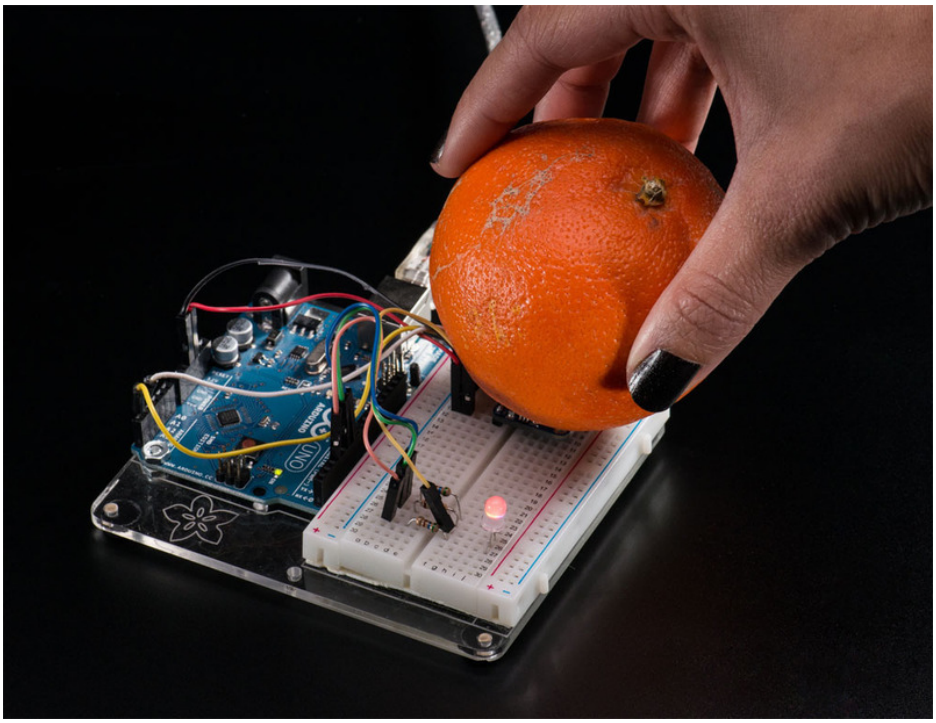


ColorView Wiring

In addition to the basic power and I2C wiring, you will need the following connections:

- LED common anode (long pin) -> 5v.
- LED Red Pin -> 1K resistor -> Arduino Pin 3
- LED Green Pin -> 560 ohm resistor -> Arduino Pin 5
- LED Blue Pin -> 560 ohm resistor -> Arduino Pin 6

Upload the ColorView sketch to your Arduino, then place different objects in front of the sensor. The LED color should match the color of the sensed object!



Library Reference

Construction and Initialization:

Adafruit_TCS34725(tcs34725IntegrationTime_t = TCS34725_INTEGRATIONTIME_2_4MS,

> **tcs34725Gain_t = TCS34725_GAIN_1X);**

> Declare a TCS34725 sensor with optional integration time and gain values.

boolean Adafruit_TCS34725::begin(void)

> Initialize the TCS34725 Color Sensor. Call this function before anything else.

Gain and Integration Time:

void Adafruit_TCS34725::setIntegrationTime(tcs34725IntegrationTime_t it)

> Sets the integration time for color samples from the sensor. Longer integration times can be used for increased sensitivity at low light levels. Valid integration times are:
>
> - TCS34725_INTEGRATIONTIME_2_4MS = 0xFF, /**< 2.4ms */
> - TCS34725_INTEGRATIONTIME_24MS = 0xF6, /**< 24ms */
> - TCS34725_INTEGRATIONTIME_50MS = 0xEB, /**< 50ms */
> - TCS34725_INTEGRATIONTIME_101MS = 0xD5, /**< 101ms */
> - TCS34725_INTEGRATIONTIME_154MS = 0xC0, /**< 154ms */
> - TCS34725_INTEGRATIONTIME_700MS = 0x00 /**< 700ms */

void Adafruit_TCS34725::setGain(tcs34725Gain_t gain)

> Sets the gain of the ADC to control the sensitivity of the sensor. Valid gain settings are:
>
> - TCS34725_GAIN_1X = 0x00, /**< No gain */
> - TCS34725_GAIN_4X = 0x01, /**< 2x gain */
> - TCS34725_GAIN_16X = 0x02, /**< 16x gain */
> - TCS34725_GAIN_60X = 0x03 /**< 60x gain */

Light Readings and Calculations:

void Adafruit_TCS34725::getRawData (uint16_t *r, uint16_t *g, uint16_t *b, uint16_t *c)

> Reads the raw sensor output for the Red, Green, Blue and Clear segments of the sensor.

uint16_t Adafruit_TCS34725::calculateColorTemperature(uint16_t r, uint16_t g, uint16_t b)

> Calculates the color temperature from the Red, Green and Blue components.

uint16_t Adafruit_TCS34725::calculateLux(uint16_t r, uint16_t g, uint16_t b)

> Calculates Lux from the Red, Green and Blue components.

Interrupts and LED control:

void Adafruit_TCS34725::setInterrupt(boolean i)

> Sets the sensor interrupt to generate an interrupt when the detected level is within the limits (see setIntLimits() below). The Int pin is only available on the breakout version.
>
> The boolean parameter can be used to control the LED. On the breakout version, you must connect the LED pin to the INT pin for LED control.
>
> - Passing "false" will enable the on-board led for reflected light measurement.
> - Passing "true" will turn the led off for incident light measurement.

void Adafruit_TCS34725::clearInterrupt(void)

> Clears the sensor interrupt.

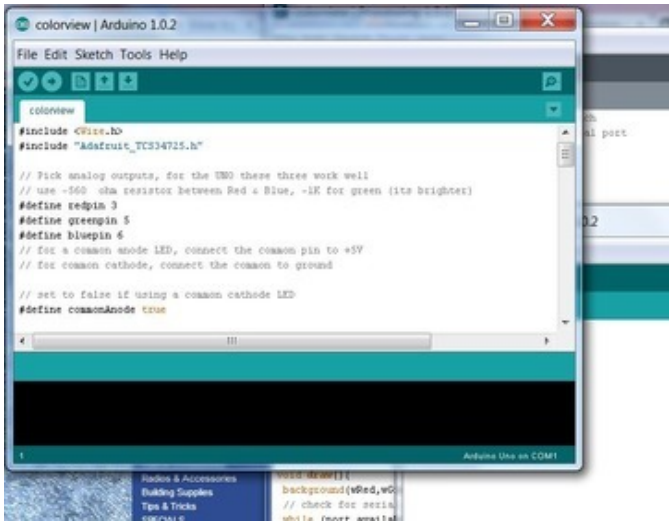
void Adafruit_TCS34725::setIntLimits(uint16_t low, uint16_t high)

> Sets the high and low threshold levels for interrupts. For more detail on the operation of interrupts, please refer to the data sheet (https://adafru.it/cb6).

Use it with Processing!

The Adafruit_TCS34725 Library includes a processing sketch to communicate with the ColorView Arduino sketch and display color on your computer screen in real time

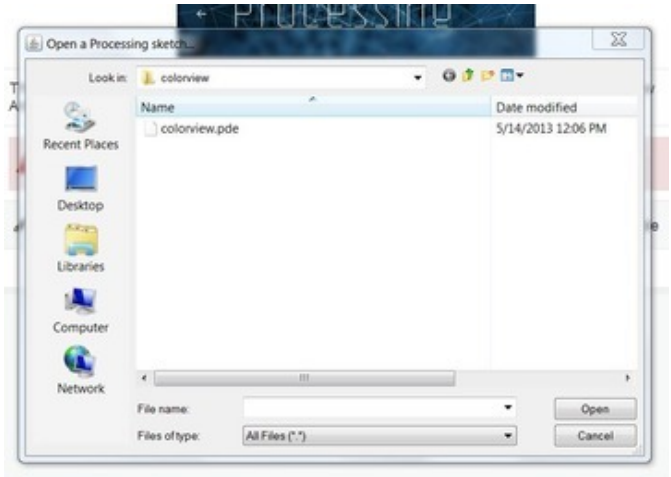
The Processing Sketch only works with Processing 1.5.1. It is not compatible with Processing version 2.0!



Load ColorView on the Arduino

Open the ColorView example sketch and upload it to your Arduino.

Make note of the serial port used by your Arduino.

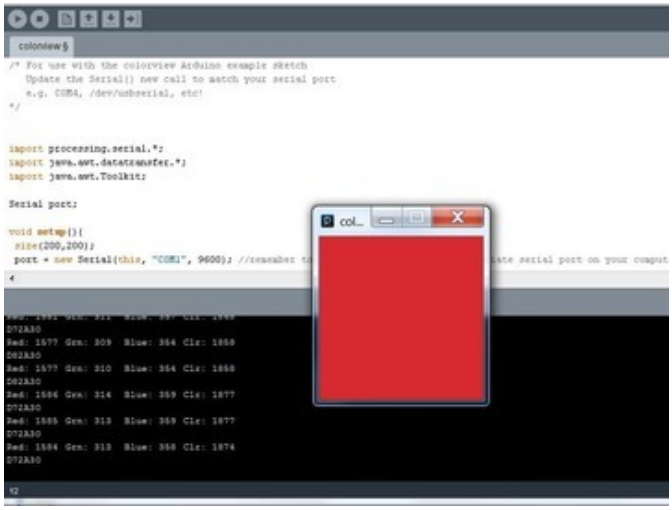


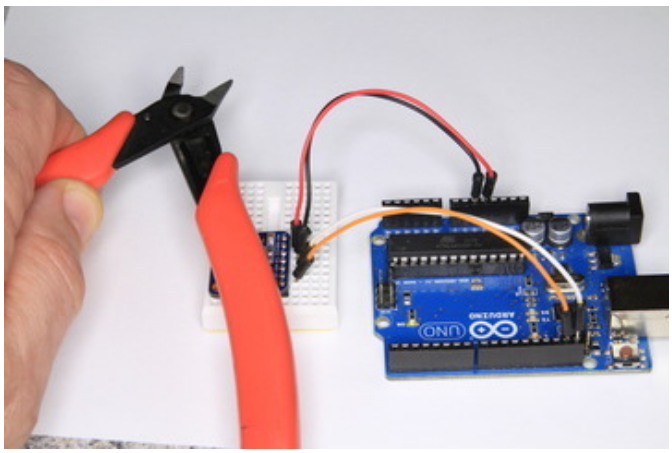
Load ColorView.pde in Processing

Navigate to the "Processing" folder inside the Adafruit_TCS34725 Library folder and open "ColorView.pde".

Edit the Serial Port

Find the line where the Serial port is opened and edit it to use the same port as your Arduino.





And Run!

When you run the processing sketch, it will display the sensor text output and pop up a window with a color patch matching the color seen by your sensor.

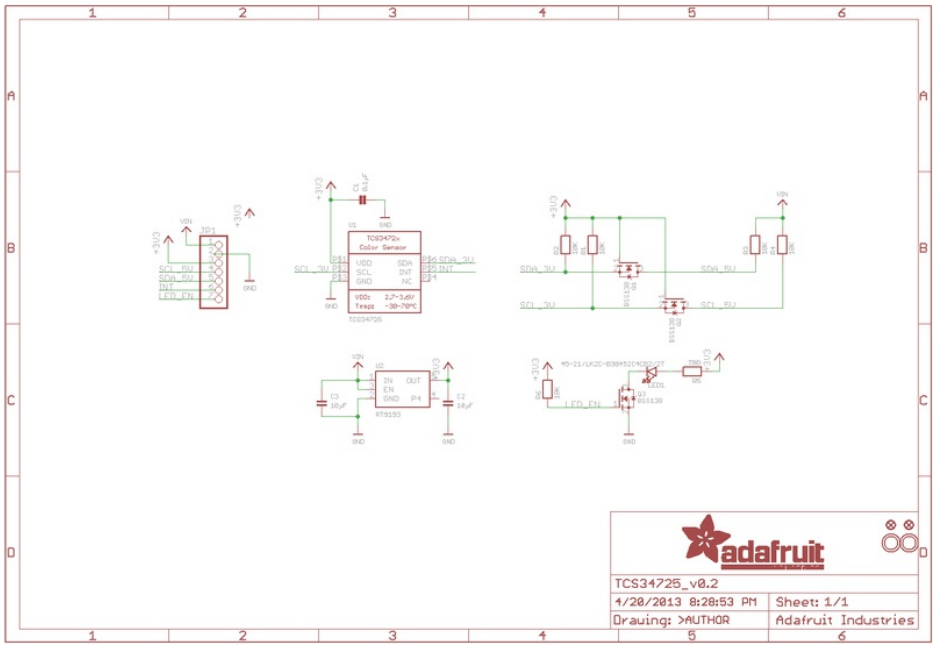
Downloads

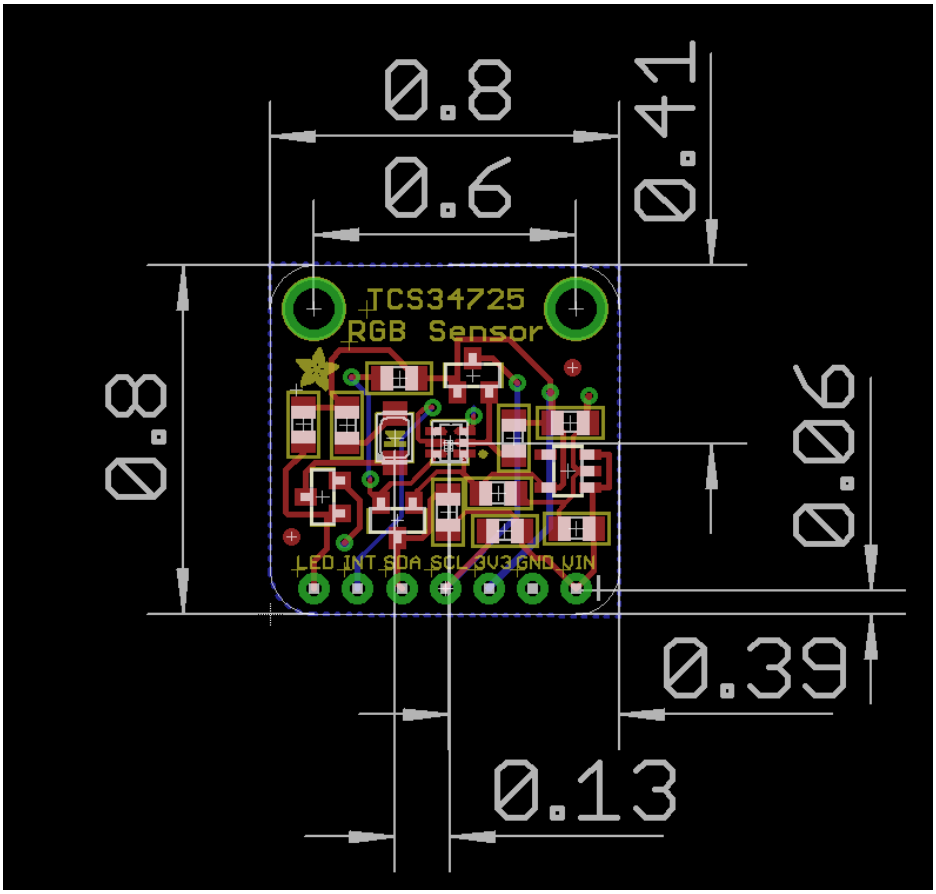
Files

- Adafruit TCS34725 Arduino Library (https://adafru.it/cb1)
- TCS34725 Data Sheet (https://adafru.it/cb6)
- Fritzing objects in Adafruit Fritzing Library (https://adafru.it/aP3)
- EagleCAD PCB files for Breakout version (https://adafru.it/rCS)
- EagleCAD PCB files for Flora version (https://adafru.it/rCU)

Breakout Board Version

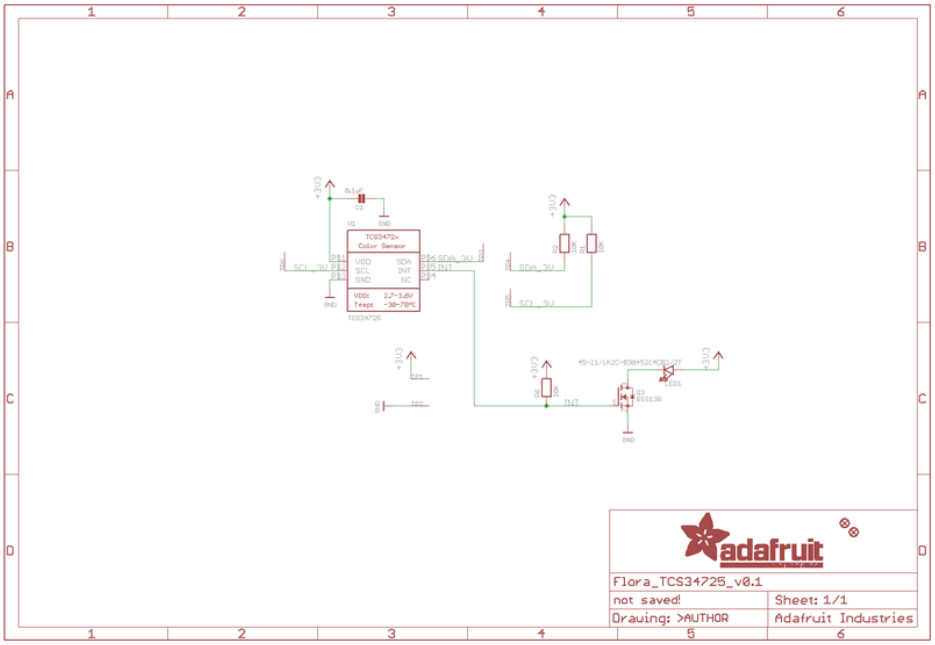
Schematic and fabrication print

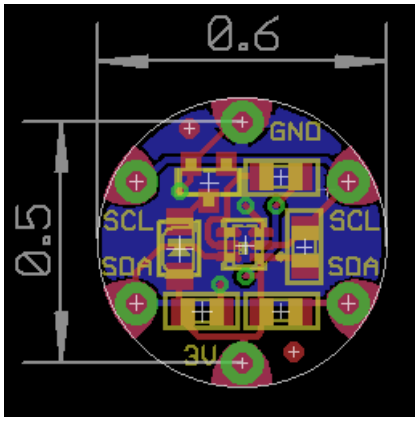




Flora Sewable Version

Schematic and fabrication print




0.6
0.5
GND
SCL
SDA
SCL
SDA
3V