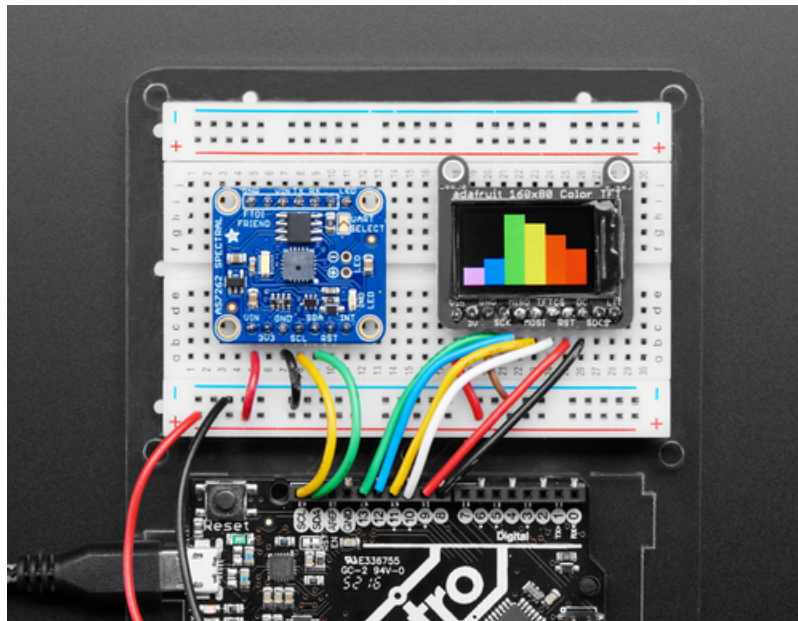


## Adafruit AS7262 6-channel Visible Light Sensor

Created by Dean Miller

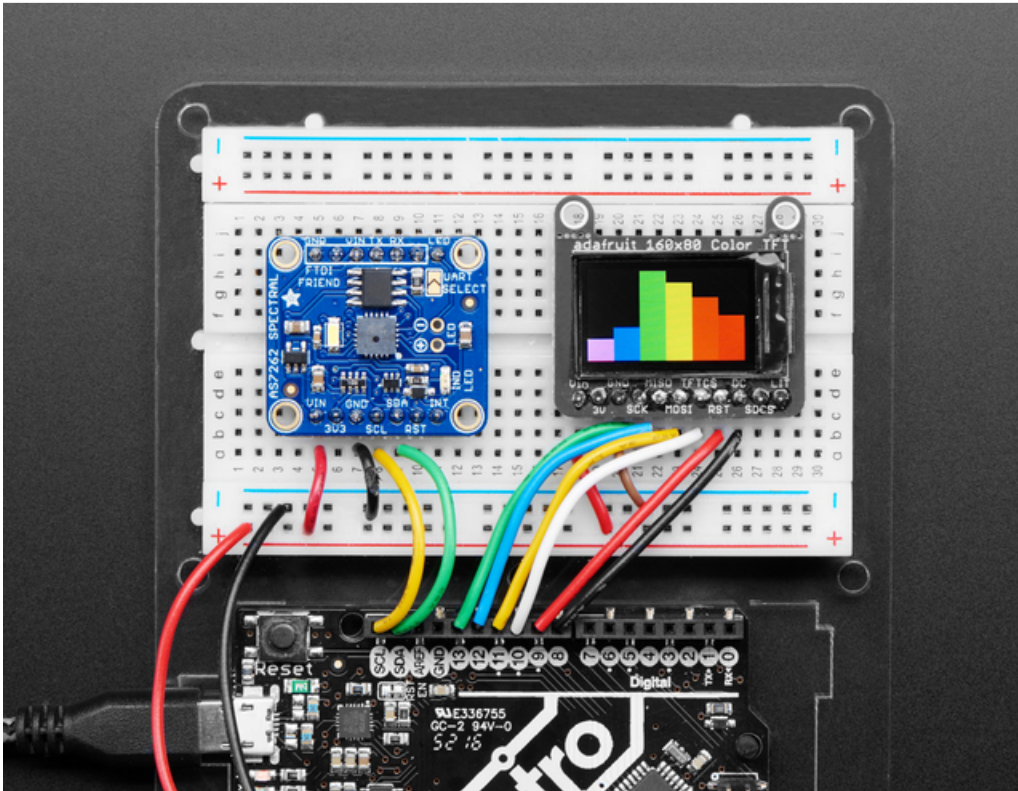


Last updated on 2018-03-28 08:29:27 PM UTC

## Guide Contents

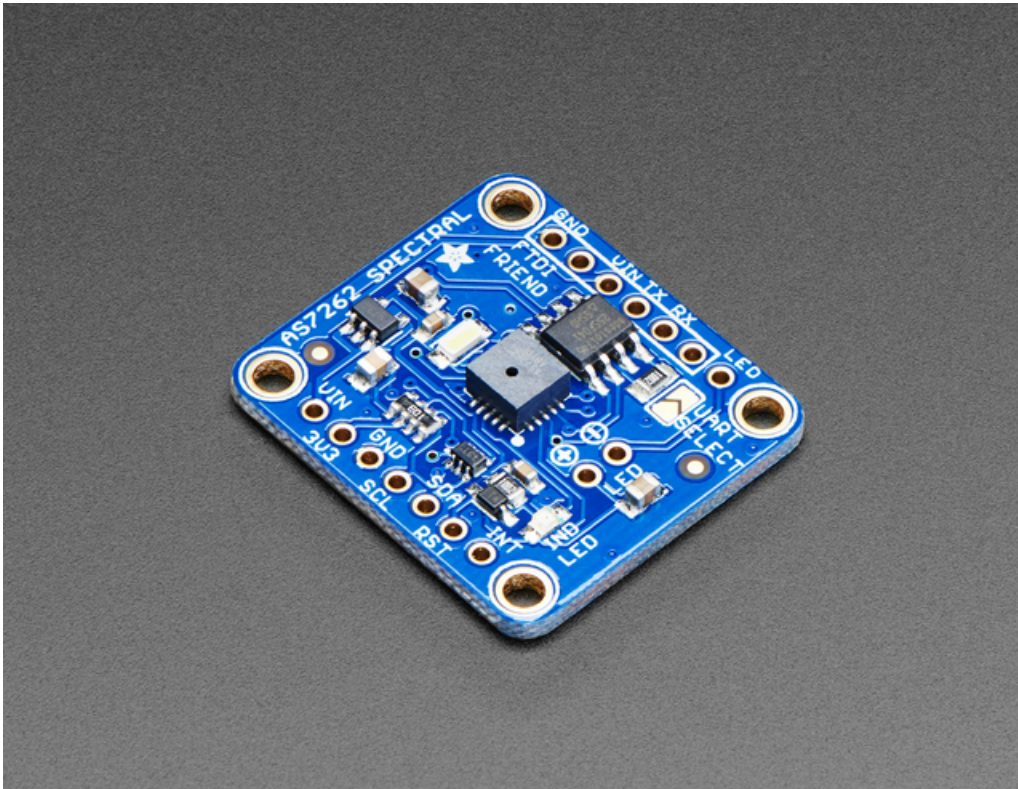
Guide Contents	2
Overview	3
Pinouts	6
Power Pins:	6
Logic pins:	6
UART Logic pins:	6
Arduino Wiring & Test	8
I2C Wiring	8
Download Adafruit_AS726x library	8
Load Test Example	9
TFT Demo	9
Adafruit 0.96" 160x80 Color TFT Display w/ MicroSD Card Breakout	10
CircuitPython Wiring & Test	13
I2C Wiring	13
Usage	14
CircuitPython Library Docs	17
Downloads	18
Documents	18
Schematic	18
Dimensions	18

## Overview

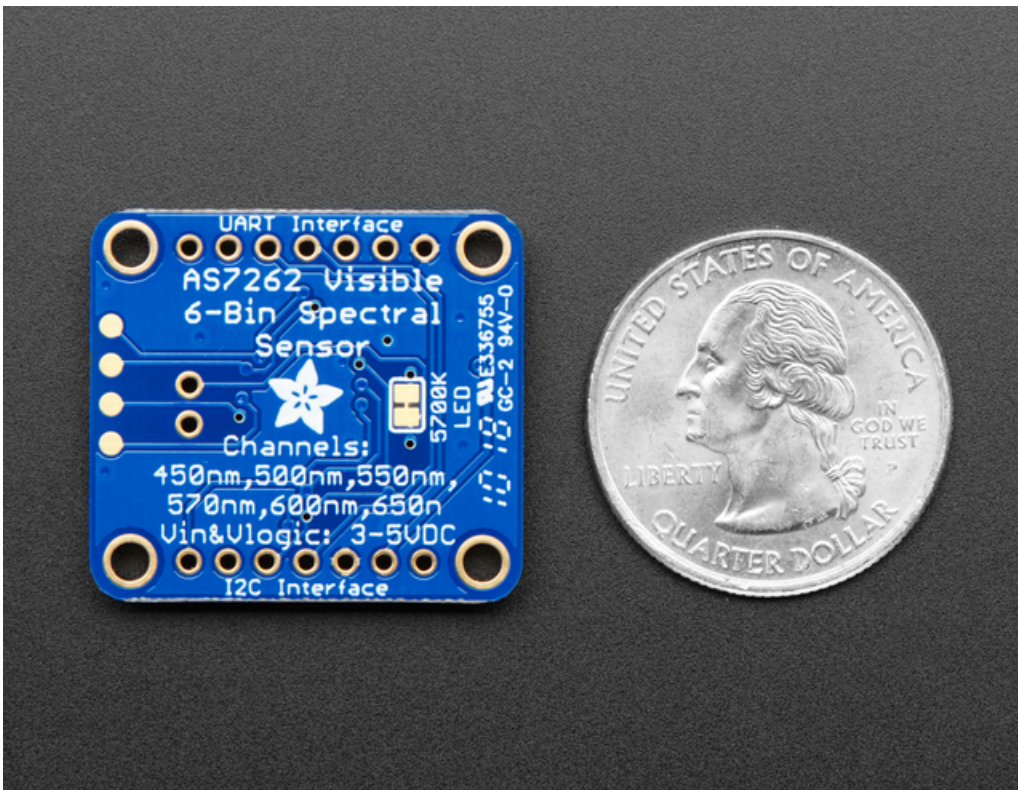


*The colors of the rainbow, so pretty in the sky  
Are also in this sensor, controlled by you and !!*

Detect trees of green, and red roses too with the new **Adafruit AS7262 6-Channel Visible Light / Color Sensor Breakout**. This sensor from AMS has 6 integrated visible light sensing channels for red, orange, yellow, green, blue, and violet. These channels can be read via the I2C bus as either raw 16-bit values or calibrated floating-point values. There is also an on-board temperature sensor that can be used to read the temperature of the chip, and a powerful LED flash to reflect light off objects for better color detection.

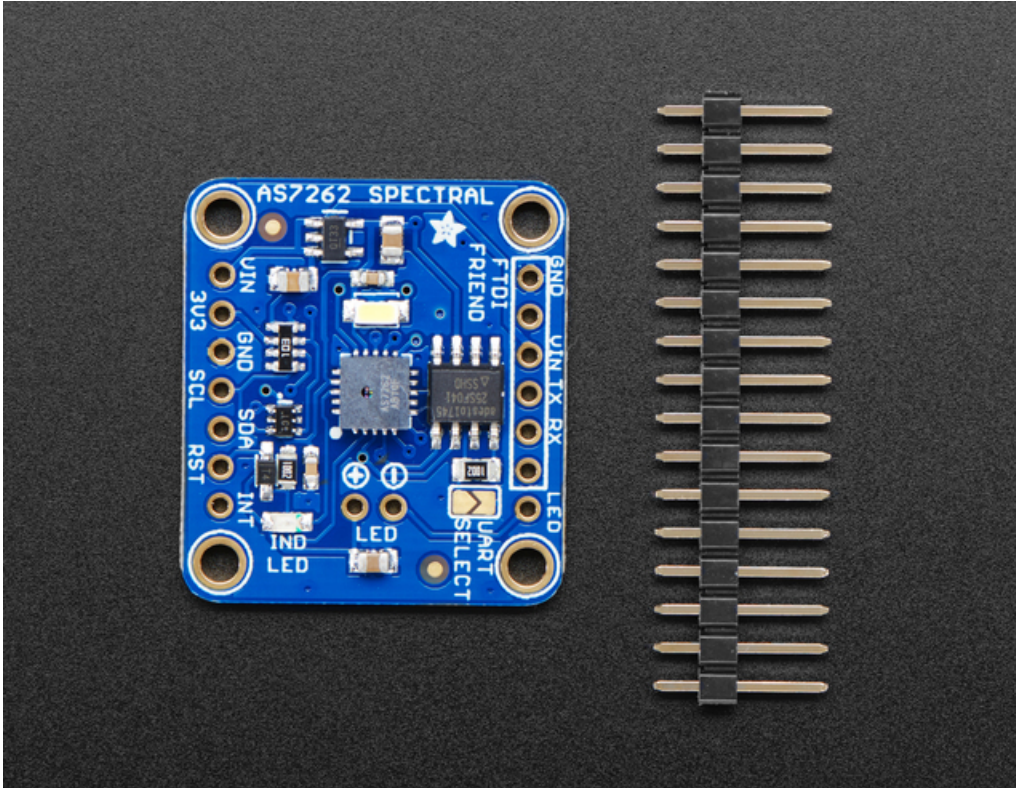


The 6 channels on the AS7262 are realized by silicon interference filters at 450nm, 500nm, 550nm, 570nm, 600nm, and 650nm. This breakout uses the I2C interface on the chip by default, but a UART interface that accepts AT commands is also selectable.

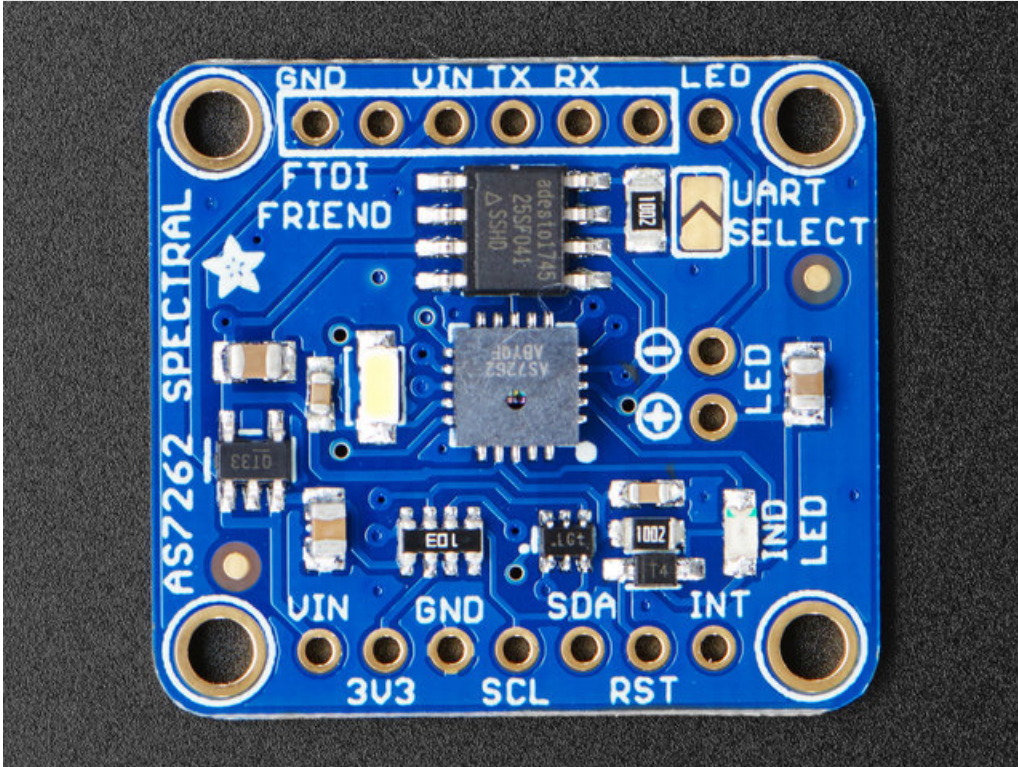


We've placed the sensor on a PCB for you and included an SPI flash chip pre-programmed with the device firmware, a 3.3V regulator, I2C level shifting, and the recommended LED with 5700K color temperature. Both the I2C and UART interfaces are broken out, making this breakout an excellent all-in-one solution for all your color-sensing needs. The pinout for the UART interface is plug-and-play compatible with the Adafruit FTDI Friend.

We've also prepared software libraries to get you up and running in Arduino or CircuitPython with just a few lines of code!



## Pinouts



This sensor has 4 mounting holes and 2 header breakout strips.

### Power Pins:

- **Vin** - this is the power pin. Since the sensor uses 3.3V, we have included an onboard voltage regulator 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

### Logic pins:

- **SCL** - this is the I2C clock pin, connect to your microcontrollers I2C clock line. There is a 10K pullup on this pin and it is level shifted so you can use 3 - 5VDC. If UART mode is selected, this pin acts as RX.
- **SDA** - this is the I2C data pin, connect to your microcontrollers I2C data line. There is a 10K pullup on this pin and it is level shifted so you can use 3 - 5VDC. If UART mode is selected, this pin acts as TX.
- **RST** - this is the reset pin. When it is pulled to ground the sensor resets itself. This pin is level shifted so you can use 3-5VDC logic.

Solder closed the **UART SELECT** jumper to switch the sensor into UART mode.

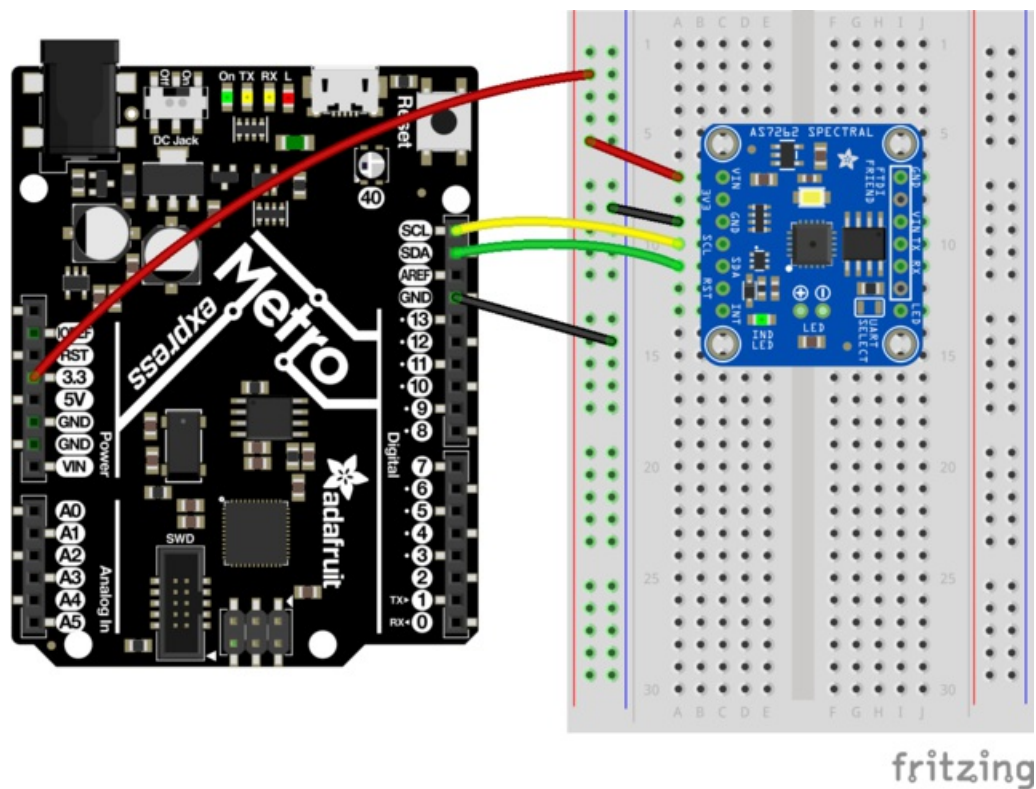
### UART Logic pins:

- **TX** - this is the UART transmit pin, connect to your microcontrollers UART RX line. There is a 10K pullup on this pin and it is level shifted so you can use 3 - 5VDC. If I2C mode is selected, this pin acts as SCL.
- **RX** - this is the UART receive pin, connect to your microcontrollers UART TX line. There is a 10K pullup on this pin

and it is level shifted so you can use 3 - 5VDC. If I2C mode is selected, this pin acts as SDA.

## Arduino Wiring & Test

You can easily wire this breakout to any microcontroller, we'll be using an Adafruit Metro M0 Express (Arduino compatible) with the Arduino IDE. But, you can use any other kind of microcontroller as well as long as it has I2C clock and I2C data lines.



## I2C Wiring

- 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**

This sensor uses I2C address **0x49**.

## Download Adafruit\_AS726x library

To begin reading sensor data, you will need to download Adafruit\_AS726x from our github repository. You can do that by visiting the github repo and manually downloading or, easier, just click this button to download the zip

Download Adafruit AS726x Library

<https://adafru.it/AUn>



Rename the uncompressed folder **Adafruit\_AS726x** and check that the **Adafruit\_AS726x** folder contains **Adafruit\_AS726x.cpp** and **Adafruit\_AS726x.h**

Place the **Adafruit\_AS726x** library folder your **arduinosketchfolder/libraries/** folder. You may need to create the **libraries** subfolder if its your first library. Restart the IDE.

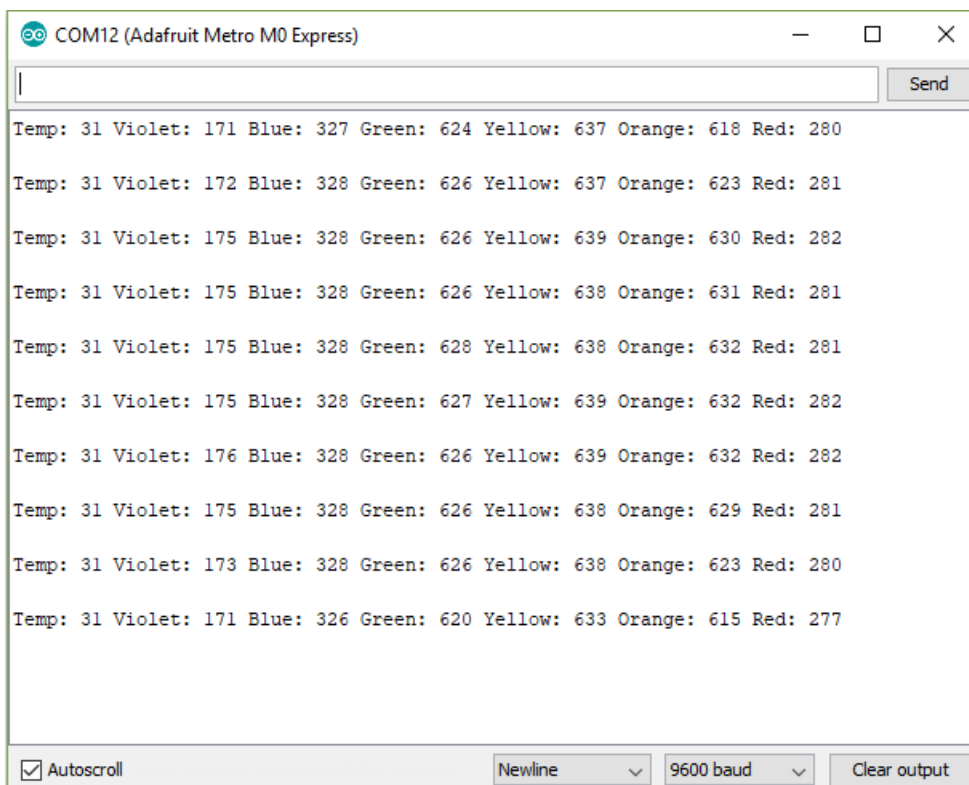
We also have a great tutorial on Arduino library installation at:  
<http://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use>

## Load Test Example

---

Open up **File->Examples->Adafruit\_AS726x->AS7262\_test** and upload to your Arduino wired up to the sensor. This example connects to the sensor and starts taking readings.

Once uploaded to your Arduino, open up the serial console at 9600 baud speed to see the readings. Your sensor will read the temperature and it's 6 visible light channels. Your serial monitor will look something like this:



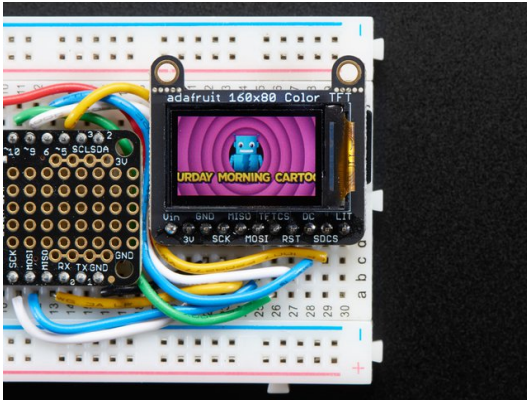
The screenshot shows a serial monitor window titled "COM12 (Adafruit Metro M0 Express)". The window contains a text area with the following data: Temp: 31 Violet: 171 Blue: 327 Green: 624 Yellow: 637 Orange: 618 Red: 280. Below the text area are controls for "Autoscroll" (checked), "Newline" (dropdown), "9600 baud" (dropdown), and "Clear output".

```
Temp: 31 Violet: 171 Blue: 327 Green: 624 Yellow: 637 Orange: 618 Red: 280
Temp: 31 Violet: 172 Blue: 328 Green: 626 Yellow: 637 Orange: 623 Red: 281
Temp: 31 Violet: 175 Blue: 328 Green: 626 Yellow: 639 Orange: 630 Red: 282
Temp: 31 Violet: 175 Blue: 328 Green: 626 Yellow: 638 Orange: 631 Red: 281
Temp: 31 Violet: 175 Blue: 328 Green: 628 Yellow: 638 Orange: 632 Red: 281
Temp: 31 Violet: 175 Blue: 328 Green: 627 Yellow: 639 Orange: 632 Red: 282
Temp: 31 Violet: 176 Blue: 328 Green: 626 Yellow: 639 Orange: 632 Red: 282
Temp: 31 Violet: 175 Blue: 328 Green: 626 Yellow: 638 Orange: 629 Red: 281
Temp: 31 Violet: 173 Blue: 328 Green: 626 Yellow: 638 Orange: 623 Red: 280
Temp: 31 Violet: 171 Blue: 326 Green: 620 Yellow: 633 Orange: 615 Red: 277
```

## TFT Demo

---

You can create a nice bar graph of the sensors 6 color channels using a 0.90 inch TFT from Adafruit.

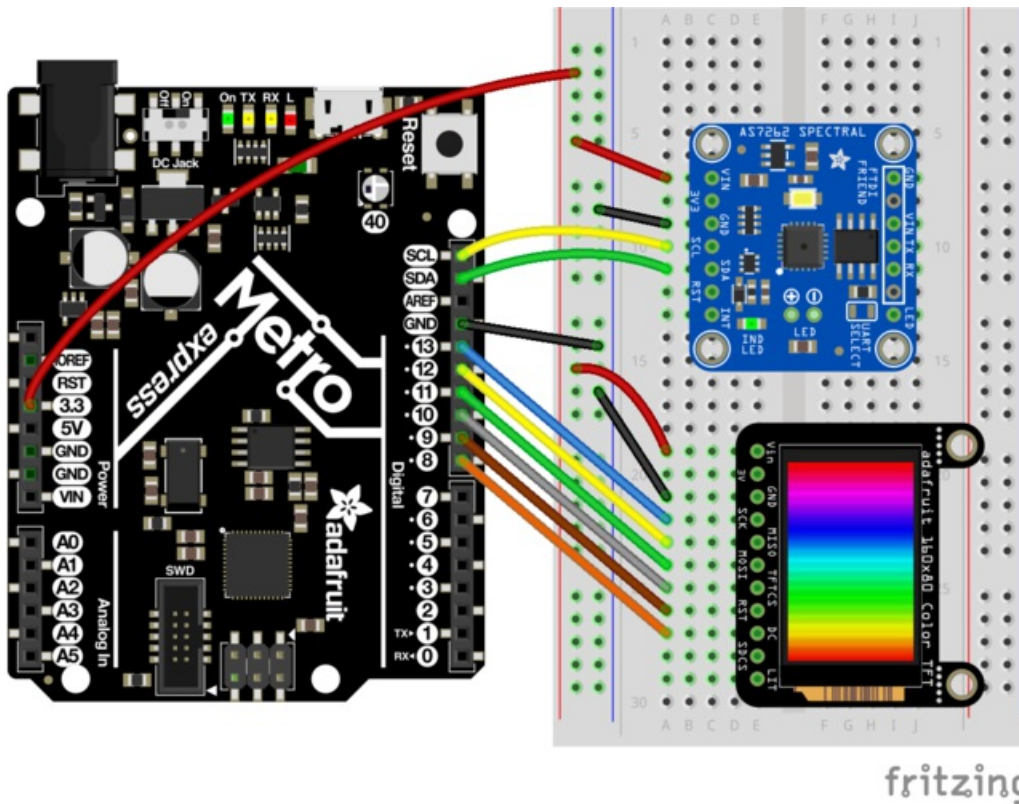


Adafruit 0.96" 160x80 Color TFT Display w/ MicroSD Card Breakout

\$19.95  
IN STOCK

ADD TO CART

Wire the TFT up as shown:



fritzing

Upload the following code to your board through the Arduino IDE (the code is also available under **File->Examples->Adafruit\_AS726x->color\_graph**)

```

/*****
 * This is a library for the Adafruit AS7262 6-Channel Visible Light Sensor
 *
 * This sketch reads the sensor and creates a color bar graph on a tiny TFT
 *
 * Designed specifically to work with the Adafruit AS7262 breakout and 160x18 tft
 * ----> http://www.adafruit.com/products/3779
 * ----> http://www.adafruit.com/product/3533
 *
 * These sensors use I2C to communicate. The device's I2C address is 0x49
 * Adafruit invests time and resources providing this open source code,
 * please support Adafruit and open source hardware by purchasing products
 *****/

```

please support Adafruit and open-source hardware by purchasing products from Adafruit!

Written by Dean Miller for Adafruit Industries.

BSD license, all text above must be included in any redistribution

\*\*\*\*\*/

```
#include <Wire.h>
#include "Adafruit_AS726x.h"

#include <Adafruit_GFX.h> // Core graphics library
#include <Adafruit_ST7735.h> // Hardware-specific library
#include <SPI.h>

// For the breakout, you can use any 2 or 3 pins
// These pins will also work for the 1.8" TFT shield
#define TFT_CS 10
#define TFT_RST 9 // you can also connect this to the Arduino reset
// in which case, set this #define pin to -1!
#define TFT_DC 8

#define SENSOR_MAX 5000

#define BLACK 0x0000
#define GRAY 0x8410
#define WHITE 0xFFFF
#define RED 0xF800
#define ORANGE 0xFA60
#define YELLOW 0xFFE0
#define LIME 0x07FF
#define GREEN 0x07E0
#define CYAN 0x07FF
#define AQUA 0x04FF
#define BLUE 0x001F
#define MAGENTA 0xF81F
#define PINK 0xF8FF

uint16_t colors[] = {
  MAGENTA,
  BLUE,
  GREEN,
  YELLOW,
  ORANGE,
  RED
};

Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_RST);

//create the object
Adafruit_AS726x ams;

//buffer to hold raw values (these aren't used by default in this example)
//uint16_t sensorValues[AS726x_NUM_CHANNELS];

//buffer to hold calibrated values
float calibratedValues[AS726x_NUM_CHANNELS];

uint16_t barWidth;
```

```

void setup() {

  Serial.begin(9600);

  tft.initR(INITR_MINI160x80); // initialize a ST7735S chip, mini display
  tft.setRotation(3);

  tft.fillScreen(ST7735_BLACK);

  barWidth = tft.width() / AS726x_NUM_CHANNELS;

  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);

  //begin and make sure we can talk to the sensor
  if(!ams.begin()){
    Serial.println("could not connect to sensor! Please check your wiring.");
    while(1);
  }

  ams.setConversionType(MODE_2);

  //uncomment this if you want to use the driver LED (off by default)
  //ams.drV0n();
}

void loop() {

  if(ams.dataReady()){

    //read the values!
    //ams.readRawValues(sensorValues);
    ams.readCalibratedValues(calibratedValues);

    for(int i=0; i<AS726x_NUM_CHANNELS; i++){
      uint16_t height = map(calibratedValues[i], 0, SENSOR_MAX, 0, tft.height());

      tft.fillRect(barWidth * i, 0, barWidth, tft.height() - height, ST7735_BLACK);
      tft.fillRect(barWidth * i, tft.height() - height, barWidth, height, colors[i]);
    }
  }
}

```

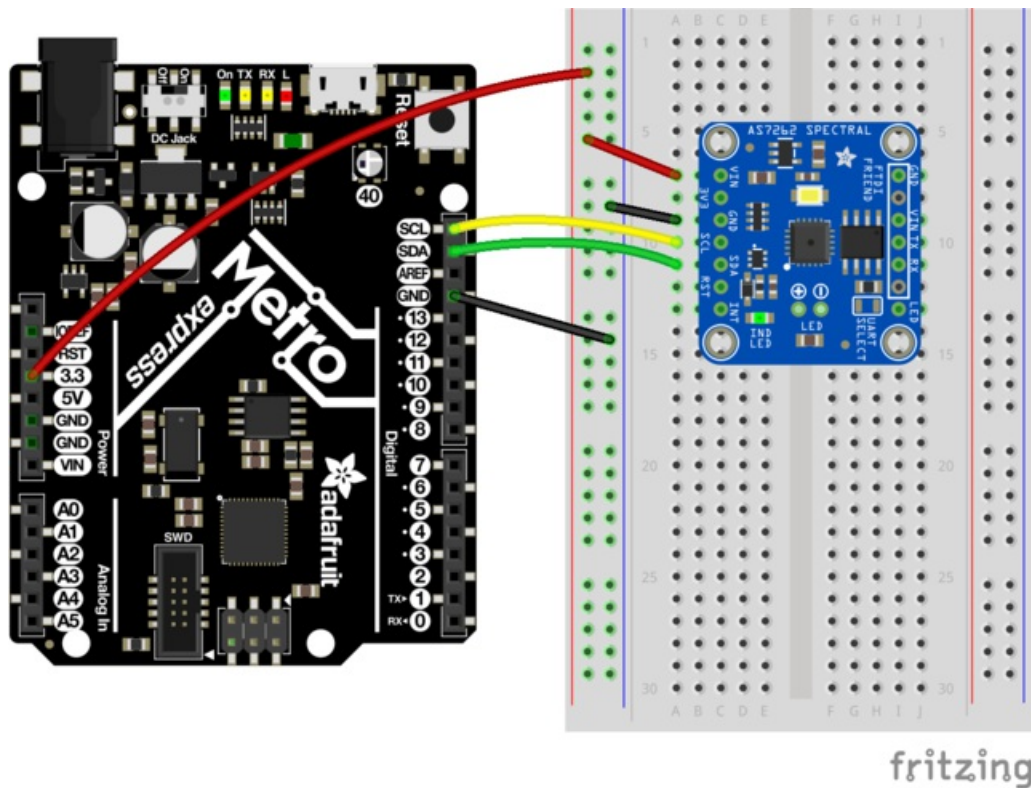
## CircuitPython Wiring & Test

You can easily wire this breakout to a microcontroller running CircuitPython. We will be using a Metro M0 Express.

### I2C Wiring

- Connect **Vin** to the power supply, 3-5V is fine.
- Connect **GND** to common power/data ground
- Connect the **SCL** pin to the I2C clock **SCL** pin on your Feather or Metro M0.  
On a Gemma M0 this would be **Pad #2/ A1**
- Connect the **SDA** pin to the I2C data **SDA** pin on your Feather or Metro M0.  
On an Gemma M0 this would be **Pad #0/A2**
- Connect the **WAKE** pin to ground.

This sensor uses I2C address **0x49**.



Next you'll need to install the [Adafruit\\_CircuitPython\\_AS726x library](#) 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 introduction guide has [a great page on how to install the library bundle](#) for both express and non-express boards.

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

- `adafruit_as726x.mpy`
- `adafruit_bus_device`
- `adafruit_register`

You can also download the `adafruit_as726x.mpy` from [it's release page on Github](#).

Before continuing make sure your board's lib folder or root filesystem has the `adafruit_as726x.mpy`, `adafruit_register`, and `adafruit_bus_device` files and folders copied over.

Next [connect to the board's serial REPL](#) so you are at the CircuitPython `>>>` prompt.

## Usage

---

To demonstrate usage we will initialize the sensor and read it's onboard temperature sensor 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
from adafruit_as726x import Adafruit_AS726x

i2c = busio.I2C(board.SCL, board.SDA)
sensor = Adafruit_AS726x(i2c)
```

Remember if you're using a board that doesn't support hardware I2C (like the ESP8266) you need to use the `bitbangio` module instead:

```
i2c = bitbangio.I2C(board.SCL, board.SDA)
```

now you can read the temperature property from the sensor:

```
print('Temperature: {0}C'.format(sensor.temperature))
```

As long as you get a reasonable temperature (usually around 28 degrees C) you know your sensor is wired up correctly and working!

Below is a complete example that reads all color channels and prints them out as a graph in the REPL. Save this as `main.py` on your board and open the REPL to see the output. Remember to change to the `bitbangio` module if necessary for your board!

```

import time

import board
import busio

from adafruit_as726x import Adafruit_AS726x

#maximum value for sensor reading
max_val = 16000

#max number of characters in each graph
max_graph = 80

def graph_map(x):
    return min(int(x * max_graph / max_val), max_graph)

# Initialize I2C bus and sensor.
i2c = busio.I2C(board.SCL, board.SDA)
sensor = Adafruit_AS726x(i2c)

sensor.conversion_mode = sensor.MODE_2

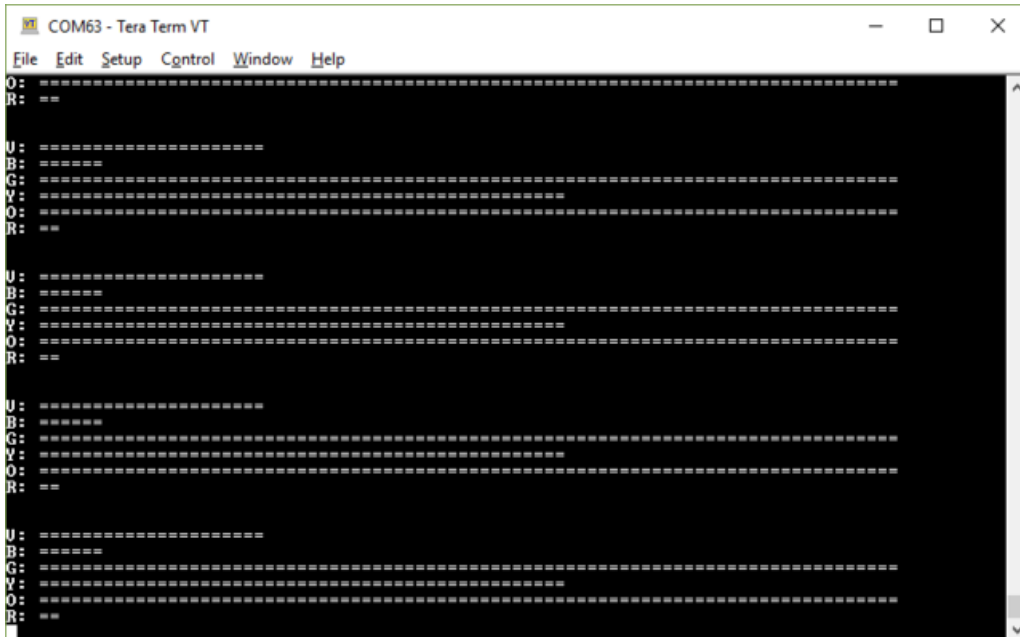
while True:
    # Wait for data to be ready
    while not sensor.data_ready:
        time.sleep(.1)

    #plot plot the data
    print("\n")
    print("V: " + graph_map(sensor.violet)*'=')
    print("B: " + graph_map(sensor.blue)*'=')
    print("G: " + graph_map(sensor.green)*'=')
    print("Y: " + graph_map(sensor.yellow)*'=')
    print("O: " + graph_map(sensor.orange)*'=')
    print("R: " + graph_map(sensor.red)*'=')

    time.sleep(1)

```

Running the above code should something like this in your REPL:







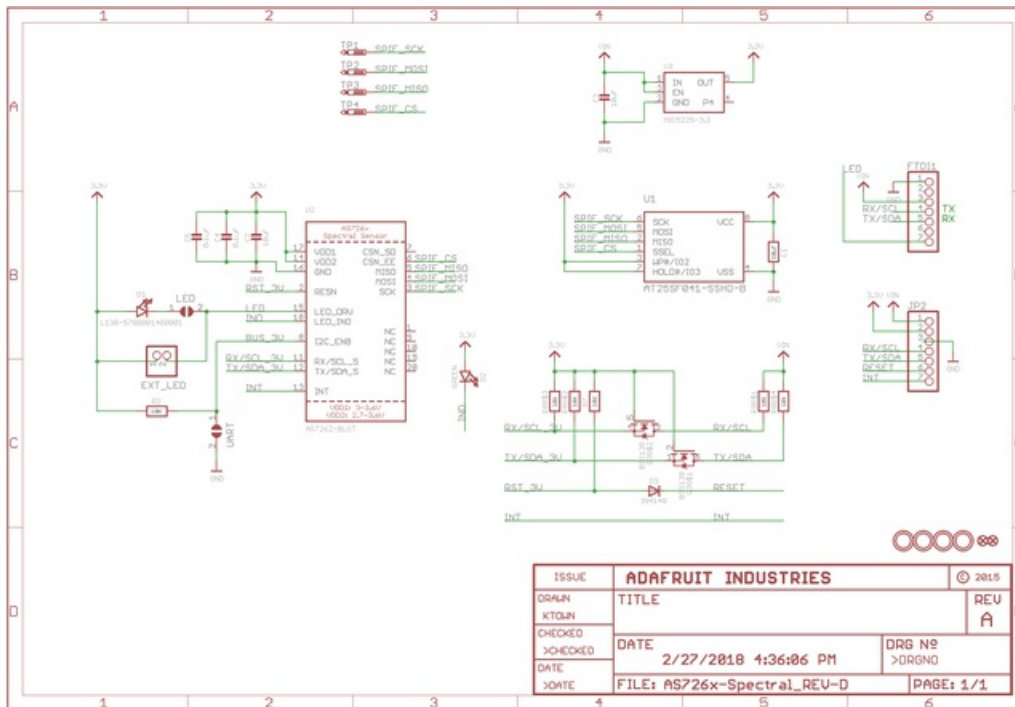
# Downloads

## Documents

- [AS726x Datasheet](#)
- [AS726x Arduino Driver](#)
- [AS726x CircuitPython Driver](#)
- [Fritzing object in the Adafruit Fritzing Library](#)
- [AS7262 Breakout PCB files \(EAGLE format\)](#)

## Schematic

click to enlarge



## Dimensions

in inches. Click to enlarge

