Overview

- Sensor Chip: TCS3200 (RGB color detector)
- Detects static color, the output is a square wave with frequency directly proportional to incident light intensity
- Supports fill-light by onboard LEDs

Specifications

- Power: 2.7V ~ 5.5V
- Ideal detection distance: 10mm
- Dimension: 36.0mm * 20.6mm
- Mounting holes size: 2.0mm

Working principle

TCS3200D contains four types of filters: red filter, green filter, blue filter, and clear with no filter. When the sensor is illuminated by a ray of light, the types of filters (blue, green, red, or clear) used by the device can be selected by two logic inputs, S2 and S3. Table 2 illustrates the relationship among S2, S3, and filter types.

S2	S3	Filter type
L	L	Red
L	H	Blue
H	L	Clear (no filter)
H	H	Green

TCS3200D outputs a square wave (50% duty cycle) with a frequency corresponding to light intensity and color, and the frequency is directly proportional to light intensity. The typical output frequency of TCS3200D is in a range of 2Hz to 500KHz. Users can control frequency values of 100%, 20%, and 2% by two programmable outputs, S0 and S1, as Table 3 shown.

S0	S1	Output Frequency Scaling
L	L	Power down
L	H	2%
H	L	20%
H	H	100%

TCS3200D has different sensitivities to red, green, and blue. As a result, the RGB output of pure white is not always 255. Therefore, a white balance adjustment is required after power-up within 2 seconds. Here are the processes.

1. Place a white paper at the top of the sensor at a distance of 1cm, and input a High-level voltage to the LED port to light up 4 bright white LED indicators.
2. The program selects R, G, and B filters respectively, and measures the corresponding RGB values of red, green, and blue.
3. Calculate 3 adjustment parameters corresponding to red, green, and blue respectively, and perform automatic white balance adjustment.

Applications

This module can be applied to color sorting, environmental light induction and calibration, test strip reading, color matching test, etc.

Pinouts

In the case of working with an MCU:

- VCC ↔ 2.7V ~ 5.5V
- GND ↔ power supply ground
- LED ↔ MCU.IO (controlling the 4 white LEDs)

- OUT ↔ MCU.IO (RGB color output frequency)
- S0/S1 ↔ MCU.IO (Output frequency scaling selection inputs)
- S2/S3 ↔ MCU.IO (Photodiode type selection inputs)

S0/S1 selects the output frequency scaling factor, S2/S3 selects the color filter of red, green, blue, then OUT outputs a square wave with a frequency proportional to the selected color intensity. The detected color can be determined by the ratio of the intensity of red, green, and blue.

Note:

- Avoids environment light noise
- White balance is required when the first time using the module, reseting the module, or changing the light source.

Get Started at Pico

If you are the first time o use the Pico, you need to first learn how to get started at the Pico before you run other examples.

The Pico supports C and the Micropython which requires different firmware.

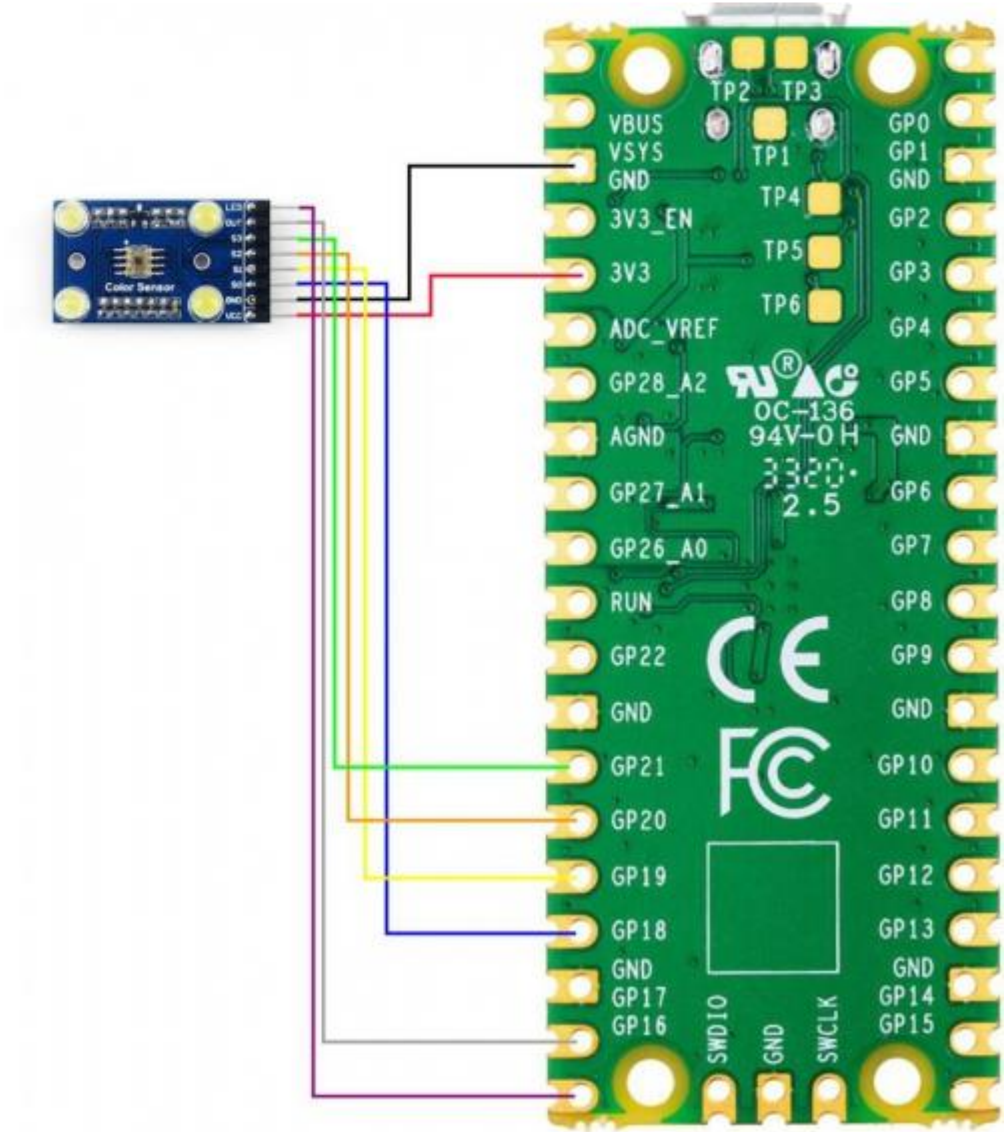
In most of the examples archives (Demo codes), we provide a Micropython firmware (uf2 file), we recommend you to use the provided firmware to test the board. Because the codes may run abnormally with different firmware.

If the board is tested to be workable, you can also download the newest firmware from Raspberry Pi and do further programming.

- If you want to use C codes, please refer to the C/C++ Guides
    - Raspberry Pi Pico C/C++ SDK
- If you want to use Micropython codes, please refer to the Micropython Guides.
    - Raspberry Pi Pico Python SDK
- The link of the newest Micropython firmware of Pico
    - Micropython firmware of Raspberry Pi Pico
- Official Guides of Raspberry Pi Pico
    - Document of Raspberry Pi Pico

Hardware Connection

Sensor	Pico	Description
VCC	3.3V	Power input
GMD	GND	Power ground
S0	GP18	Combined with S1, select output frequency scaling
S1	GP19	Combined with S2, select output frequency scaling
S2	GP20	Combined with S3, select output frequency scaling
S3	GP21	Combined with S2, select output frequency scaling
OUT	GP17	Read the output frequency of RGB
LED	GP16	Control the states of 4 LED indicators



Download examples

Use the Raspberry Pi as the host device. Open a terminal and run the following commands to download the example.

```
sudo apt-get install p7zip-full
cd ~
```

```
sudo wget  https://www.waveshare.com/w/upload/3/33/Color-Sensor-code.7z
7z x Color-Sensor-code.7z -o./Color-Sensor-code
```

Examples

C codes

- go into the c directory

```
cd ~/Color-Sensor-code/Pico/c/build
```

- Add the path of SDK

```
export PICO_SDK_PATH=../../pico-sdk
```

Note that if the path of your SDK is different, you need to modify the command and use the correct path to export

- Generate Makefile and build

```
cmake ..
make -j9
```

- After building, a uf2 file is generated
- Press and hold the button of Pico, connect it to Raspberry Pi then release the button.
- Copy/Drag the uf2 file to the portable disk (RPI-RP2) recognized

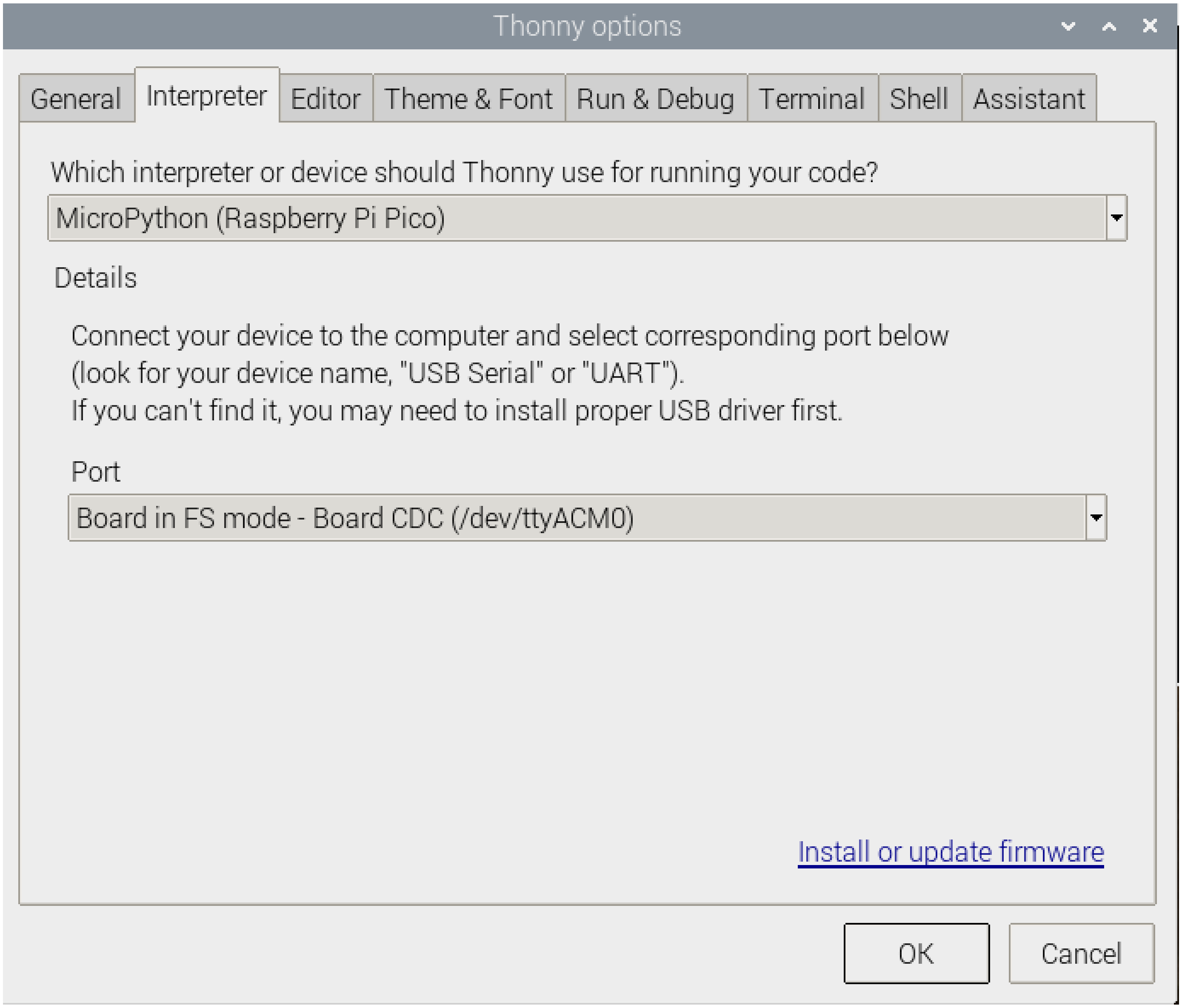
```
cp main.uf2 /media/pi/RPI-RP2/
```

Micropython codes

- Flash the Micropython firmware first
- Open the Thonny IDE (Menu->Programming->Thonny Python IDE).
- 【Optional】If the Thonny IDE in the Raspberry Pi is not the new version that supports Pico, please upgrade it first.

```
sudo apt upgrade thonny
```

- Configure Interpreter, choose Tools->Options... -> Interpreter, choose MicroPython(Raspberry Pi Pico) and the ttyACM0 port.



Which interpreter or device should Thonny use for running your code?

MicroPython (Raspberry Pi Pico)

Details

Connect your device to the computer and select corresponding port below
(look for your device name, "USB Serial" or "UART").
If you can't find it, you may need to install proper USB driver first.

Port

Board in FS mode - Board CDC (/dev/ttyACM0)

- Click File -> Open.. and browser the micropython codes (Color Sensor.py) to run the codes

Expected result

- Place the white paper 1cm above the four bright white LEDs of the sensor
- Power up the development board. At this time, the LED port is set to a high level, and the four bright white LEDs light up
- Wait for more than 2 seconds after power-on (the white balance is being adjusted at this time)

- After the adjustment, the sensor is facing the object to be measured, and the serial port outputs the corresponding RGB data.

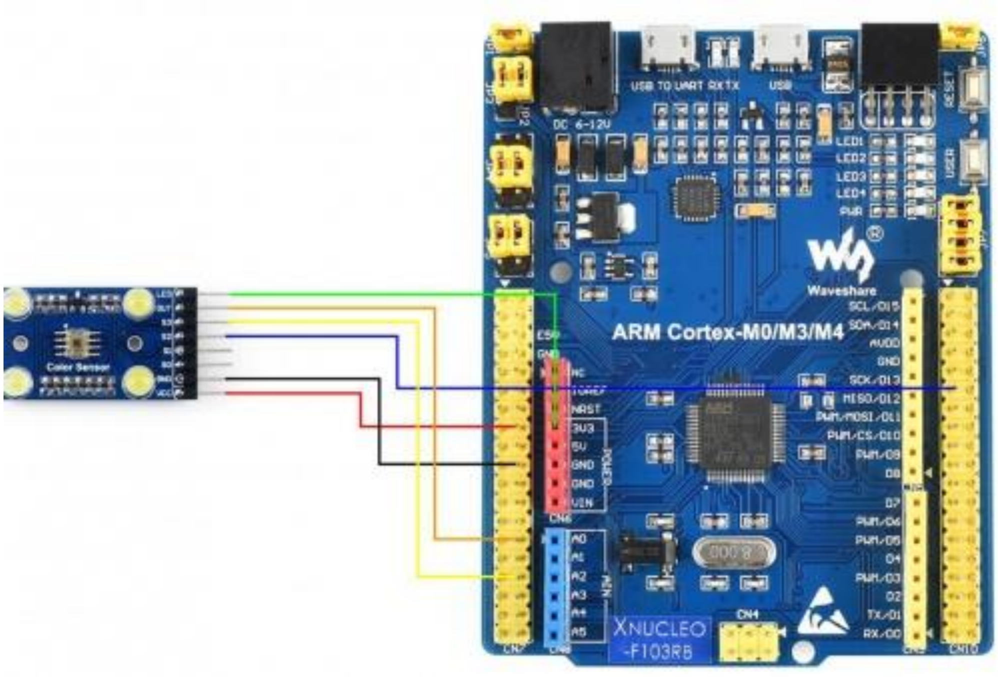
Notices

- Light interference from outside should be avoided, which may affect the result of color identification. It is recommended to place the light source and Color Sensor in a close, light reflection-free box for testing.
- White balance adjustment is required whenever the Color Sensor module is reset or the light source is changed.

The STM32 examples are based on the STM32F103RBT6 and the STM32H743. The connection provided below is based on the STM32F103RB. If you need to use other STM32 boards, you may need to change the hardware connection and port the code yourself.

Hardware connection

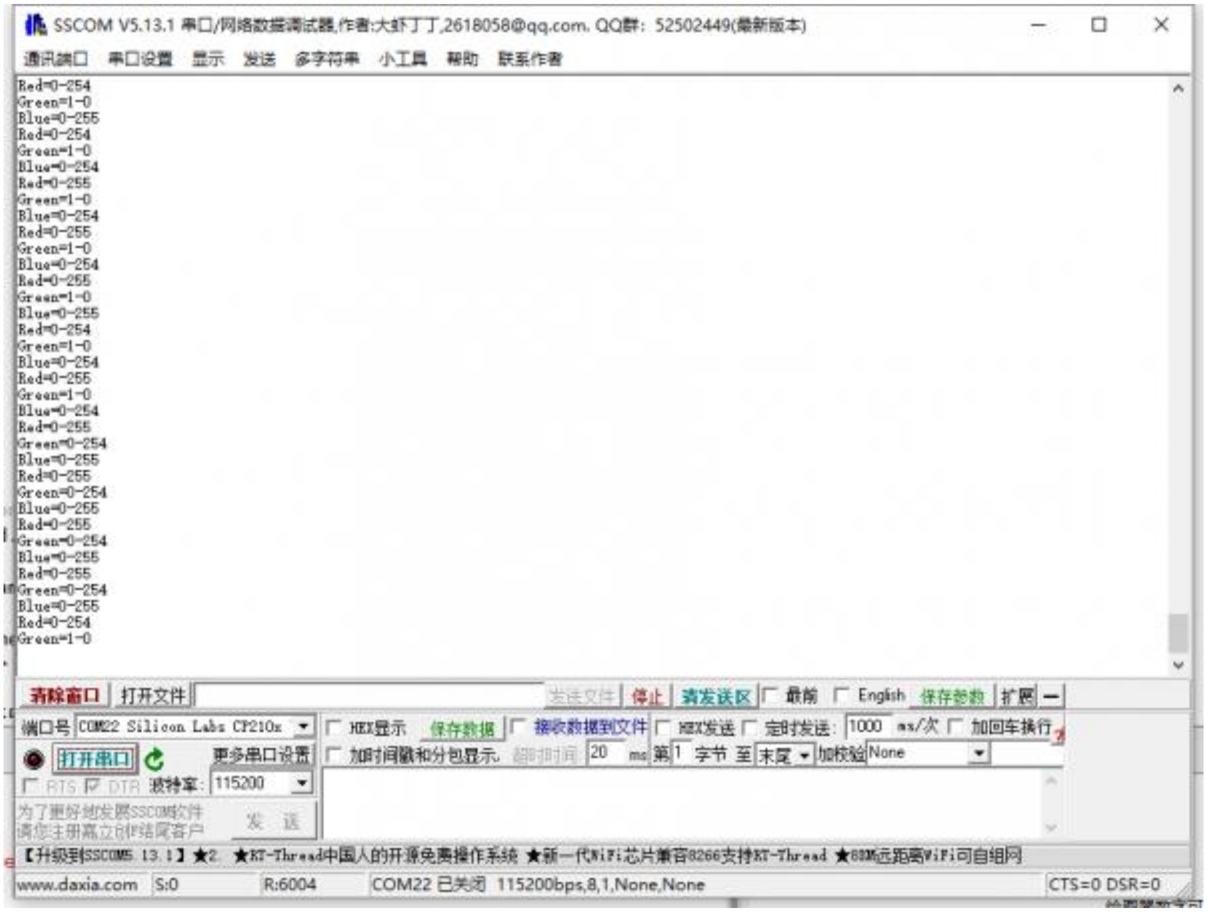
Sensor	STM32	Description
VCC	3.3V	Power input
GMD	GND	Power ground
S0	NC	Combined with S1, select output frequency scaling
S1	NC	Combined with S2, select output frequency scaling
S2	GPIOA.5	Combined with S3, select output frequency scaling
S3	GPIOA.4	Combined with S2, select output frequency scaling
OUT	GPIOA.0	Read the output frequency of RGB
LED	3.3V	Control the states of 4 LED indicators



Examples

The examples are developed based on the HAL libraries. Download the Demo codes archive to your PC. Unzip and find the STM32 project from Color-Sensor-code\STM32\STM32F103RB\MDK-ARM.

- Open the Color Sensor.uvprojx file by Keil
- Build and the project
- Program the project to your STM32 board.
- Connect the UART1 of your STM32 board to the PC and check the serial data by SSCOM software.


SSCOM V5.13.1 串口/网络数据调试器,作者:大虾丁丁,2618058@qq.com. QQ群：52502449(最新版本)
通讯端口 串口设置 显示 发送 多字符串 小工具 帮助 联系作者
清除窗口 打开文件 发送文件 停止 清发送区 最前 English 保存参数 扩展
HEX显示 保存数据 接收数据到文件 HEX发送 定时发送 加回车换行
打开串口 更多串口设置 加时间戳和分包显示 加校验
发 送
【升级到SSCOM5.13.1】★RT-Thread中国人的开源免费操作系统 ★新一代WiFi芯片兼容8266支持RT-Thread ★8KM远距离WiFi可自组网

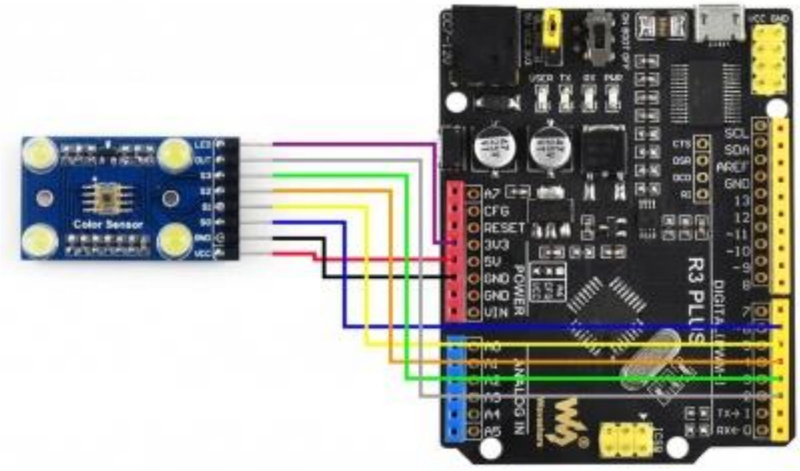
```
Red=0-254
Green=1-0
Blue=0-255
Red=0-254
Green=1-0
Blue=0-254
Red=0-255
Green=1-0
Blue=0-254
Red=0-255
Green=1-0
Blue=0-254
Red=0-255
Green=1-0
Blue=0-255
Red=0-254
Green=1-0
Blue=0-254
Red=0-255
Green=1-0
Blue=0-254
Red=0-255
Green=0-254
Blue=0-255
Red=0-255
Green=0-254
Blue=0-255
Red=0-255
Green=0-254
Blue=0-255
Red=0-255
Green=0-254
Blue=0-255
Red=0-254
Green=1-0
```

端口号 COM22 Silicon Labs CP210x
波特率：115200
定时发送：1000 ms/次
www.daxia.com S:0 R:6004 COM22 已关闭 115200bps,8,1,None,None CTS=0 DSR=0

The Arduino example is written for the Arduino UNO. If you want to connect it to other Arduino boards, you may need to change the connection.

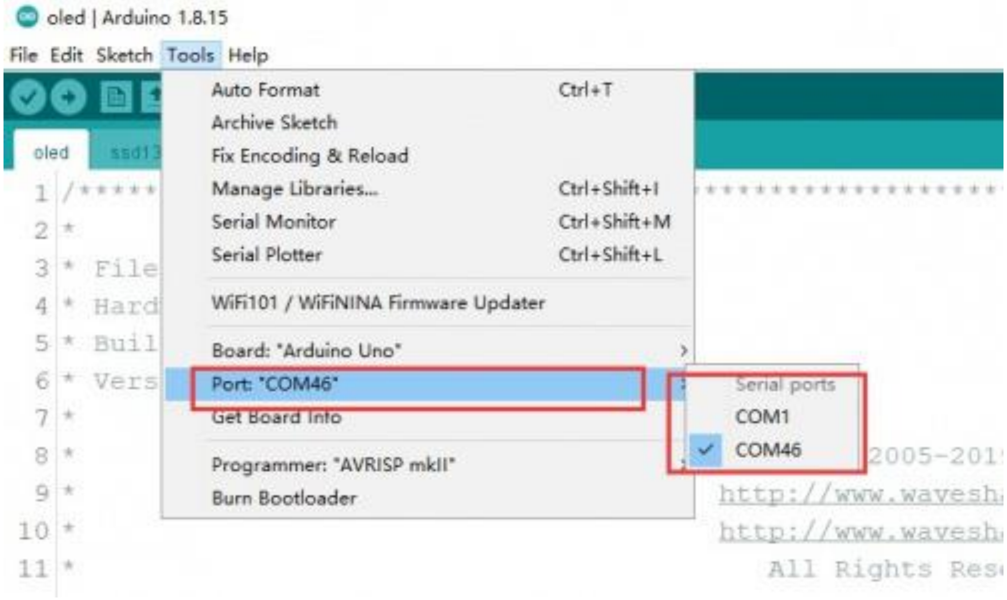
Hardware connection

Sensor	Arduino	Description
VCC	5V	Power input
GMD	GND	Power ground
S0	D6	Combined with S1, select output frequency scaling
S1	D5	Combined with S2, select output frequency scaling
S2	D4	Combined with S3, select output frequency scaling
S3	D3	Combined with S2, select output frequency scaling
OUT	D2	Read the output frequency of RGB
LED	GP16	Control the states of 4 LED indicators



Examples

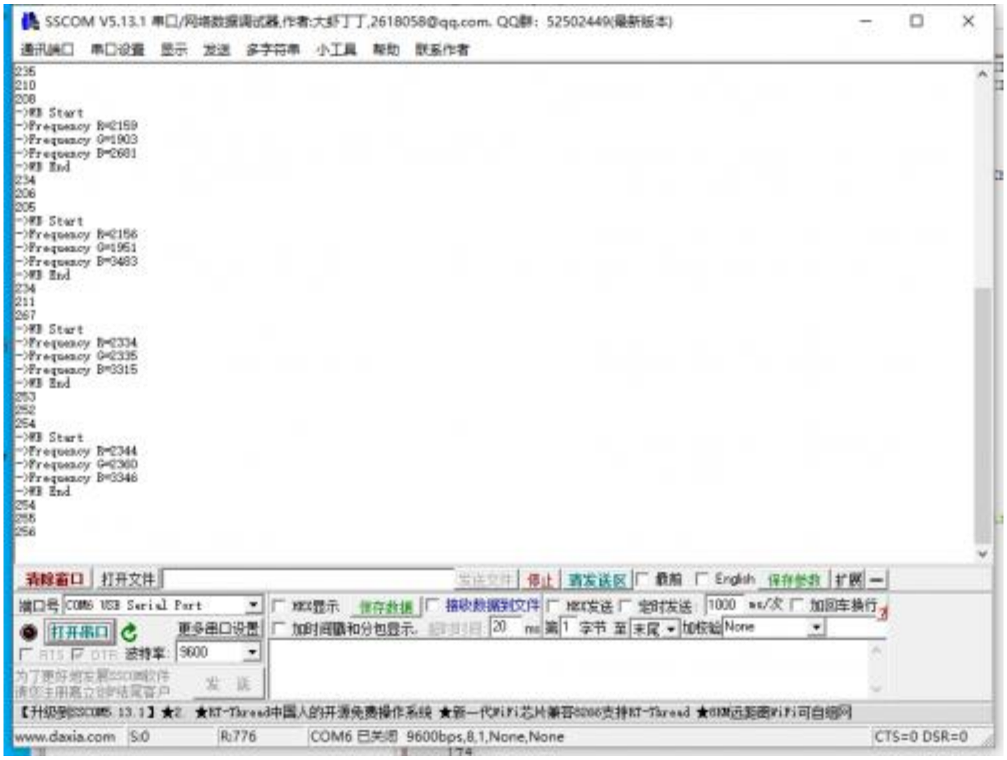
- Download the demo codes to your PC and unzip
- Install the Arduino IDE in your PC
- Go into Color-Sensor-code/Arduino/Color_Sensor
- Run the Color_Senosr.ino file
- Select the correct Board and the Port



Board: "Arduino Uno"

Port: "COM46"

- Build the project and upload it to the board.
- Open the serial monitor of the Arduino IDE or the SSCOM software and check the serial data.



```
236
210
208
->WB Start
->Frequency R=2159
->Frequency G=1903
->Frequency B=2681
->WB End
234
206
205
->WB Start
->Frequency R=2156
->Frequency G=1951
->Frequency B=3483
->WB End
234
211
267
->WB Start
->Frequency R=2334
->Frequency G=2335
->Frequency B=3315
->WB End
253
252
254
->WB Start
->Frequency R=2344
->Frequency G=2360
->Frequency B=3346
->WB End
254
255
256
```

www.daxia.com S:0 R:776 COM6 已关闭 9600bps,8,1,None,None CTS=0 DSR=0

Resources

- Datasheets
- User Manual
- Schematic
- Demo Code
- Software