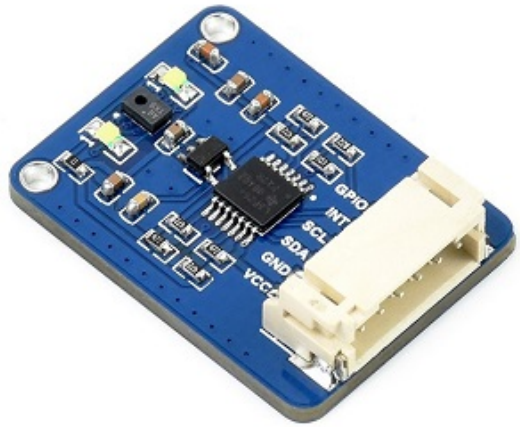
AS7341 Spectral Color Sensor

来自Waveshare Wiki
跳转至： 导航、 搜索

说明

产品概述

AS7341是以AS7341可见光谱传感IC为核心的传感器，它能够感知环境中不同波段的可见光成分值，在灵敏度、准确度上也均比较可观，同时介于它的体积非常小，如果你将用来做一个微型的光谱分析仪，它将是一个非常好的选择。



功能简介

特性	光谱传感器
接口	I2C

产品特性

- 强大的AS7341 IC内部集成了8个可见光通道、1个闪烁通道、1个NIR通道和1个未加滤光片的CLEAR通道，光谱响应波长约350nm-1000nm
- 内置6个独立的16位ADC，可以同时工作并行处理数据，互不影响
- 集成了一个专用通道，可以检测特定频率的环境光闪烁
- 板载了两颗高亮LED，可在弱光环境下补光
- 具有光谱中断检测，可编程上下限阈值
- 带有通用输入/输出GPIO口
- 板载电平转换电路，可兼容3.3V/5V的工作电平
- 提供完善的配套资料手册(Raspberry/Arduino/STM32示例程序和用户手册等)

产品参数

- 工作电压: 3.3V/5V
- 工作电流: 20mA(不打开LED) 70mA（打开LED）
- 传感器: AS7341
- 逻辑电压: 3.3V/5V
- 通信接口: I2C
- 产品尺寸: 30.5mm x 23mm
- 固定孔径: 2.0mm

接口说明

- 引脚功能

引脚号	标识	管脚描述
1	VCC	3.3V/5V电源正
2	GND	电源地
3	SDA	I2C数据线
4	SCL	I2C时钟线
5	INT	中断输出引脚
6	GPIO	通用输入/输出

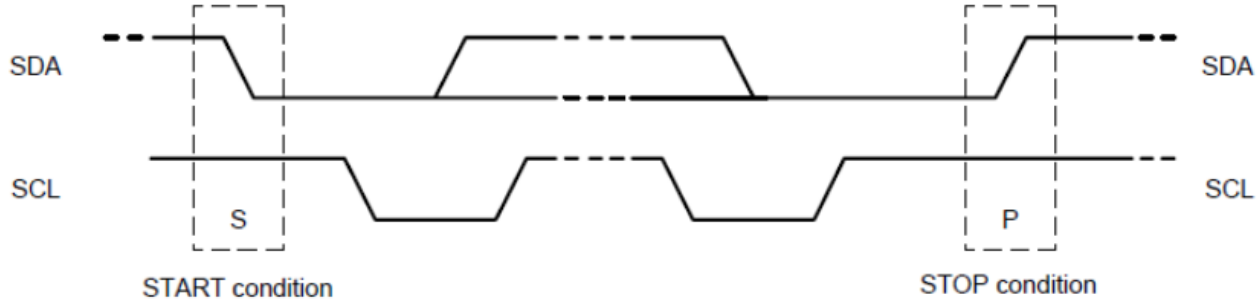
硬件说明

芯片

本产品采用AS7341-DLGM为核心，是一个用于光谱识别和颜色匹配应用的11通道IC。光谱响应定义在波长约350nm到1000nm,6个通道可以由独立的adc并行处理，而其他通道都是 可通过多路复用器访问。
AS7341通过纳米光学沉积干涉滤波器将滤波器集成到标准CMOS硅中，该技术及其封装提供了一个内置的光圈来控制进入传感器阵列的光线。控制和光谱数据访问通过串行I²C接口实现。

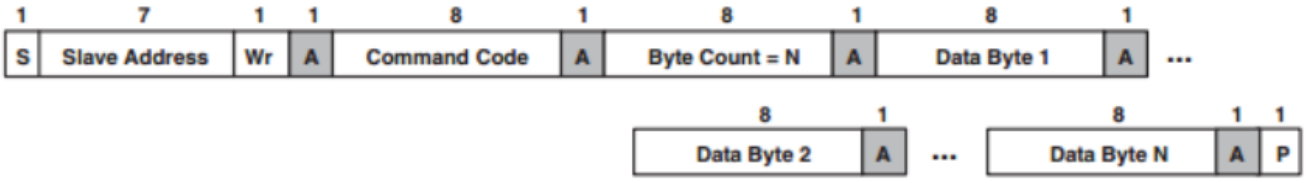
通信协议

从上面功能引脚表得知使用的是I2C通信，I2C 通信，一条数据线，一条时钟线。 I2C 总线在传送数据过程中共有三种类型信号：开始信号、结束信号和应答信号。



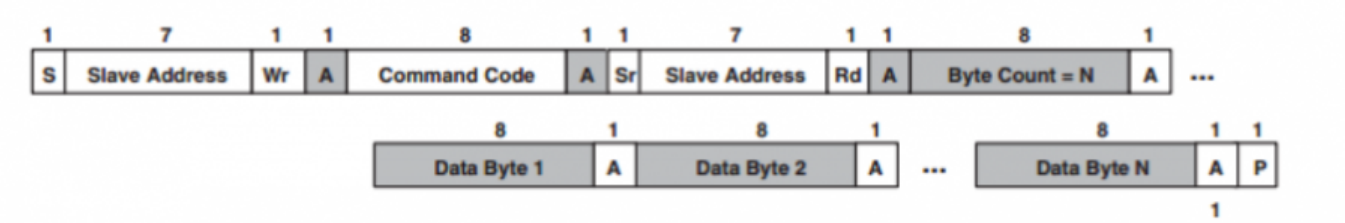
开始信号：SCL 为高电平时， SDA 由高电平向低电平跳变，开始传送数据。
结束信号：SCL 为高电平时， SDA 由低电平向高电平跳变，结束传送数据。
应答信号：接收数据的 IC 在接收到 8bit 数据后，向发送数据的 IC 发出特定的低电平脉冲，表示已收到数据。

- I2C写数据时序



首先主机（即树莓派，后面统称为主机）会发送一个开始信号，然后将其 I2C 的 7 位地址与写操作位组合成 8 位的数据发送给从机（即 TSL2581 传感器模块,后面统称为从机），从机接收到后会响应一个应答信号，主机此时将命令寄存器地址发送给从机，从机接收到发送响应信号，此时主机发送命令寄存的值，从机回应一个响应信号，直到主机发送一个停止信号，此次 I2C 写数据操作结束

- I2C读数据时序



首先主机会发送一个开始信号，然后将其 I2C 的 7 位地址与写操作位组合成 8 位的数据发送给从机，从机接收到后会响应一个应答信号，主机此时将命令寄存器地址发送给从机，从机接收到发送响应信号，此时主机重新发送一个开始信号，并且将其 7 位地址和读操作位组合成 8 位的数据发送给从机，从机接收到信号后发送响应信号，再将其寄存器中的值发送给主机，主机端给予响应信号，直到主机端发送停止信号，此次通信结束。

- I2C地址

AS7341的I2C设备地址为0X39

9.1 I²C Address

Figure 26:
AS7341 I²C Slave Address

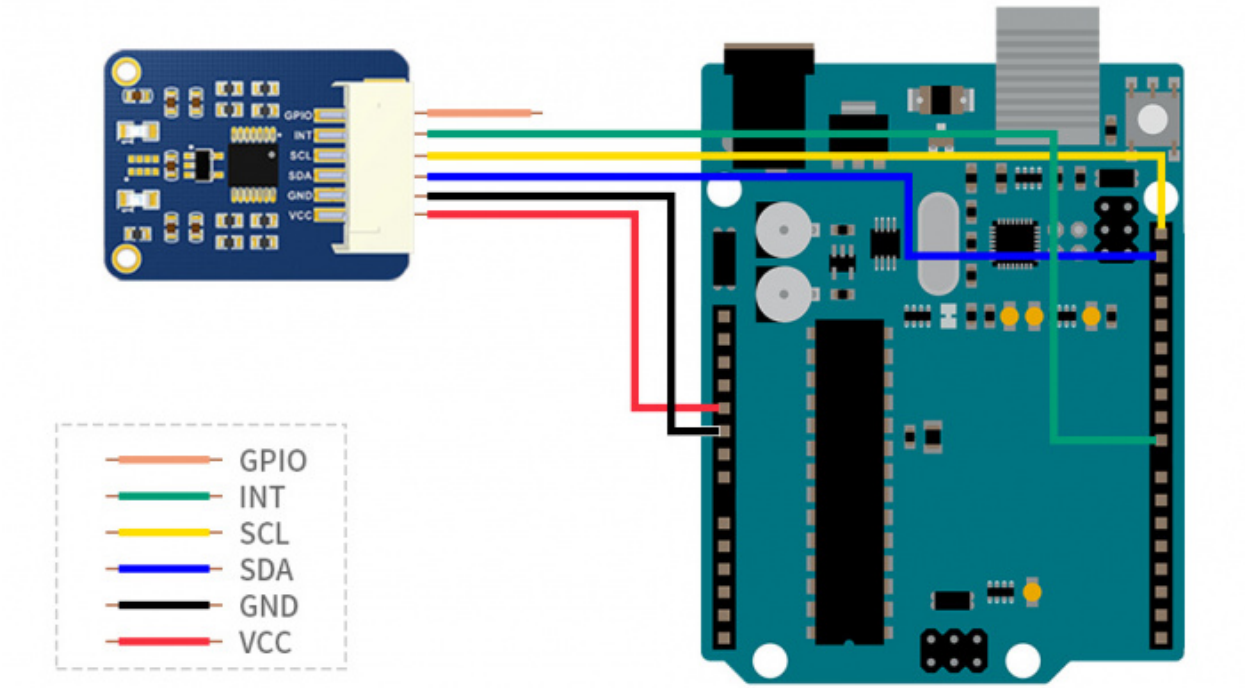
Device	I²C Address
AS7341	0x39

AS7341数据手册第21页

Arduino

本例程在Arduino UNO上测试，如使用了其他型号的Arduino，请注意相关引脚连接是否正确

硬件连接

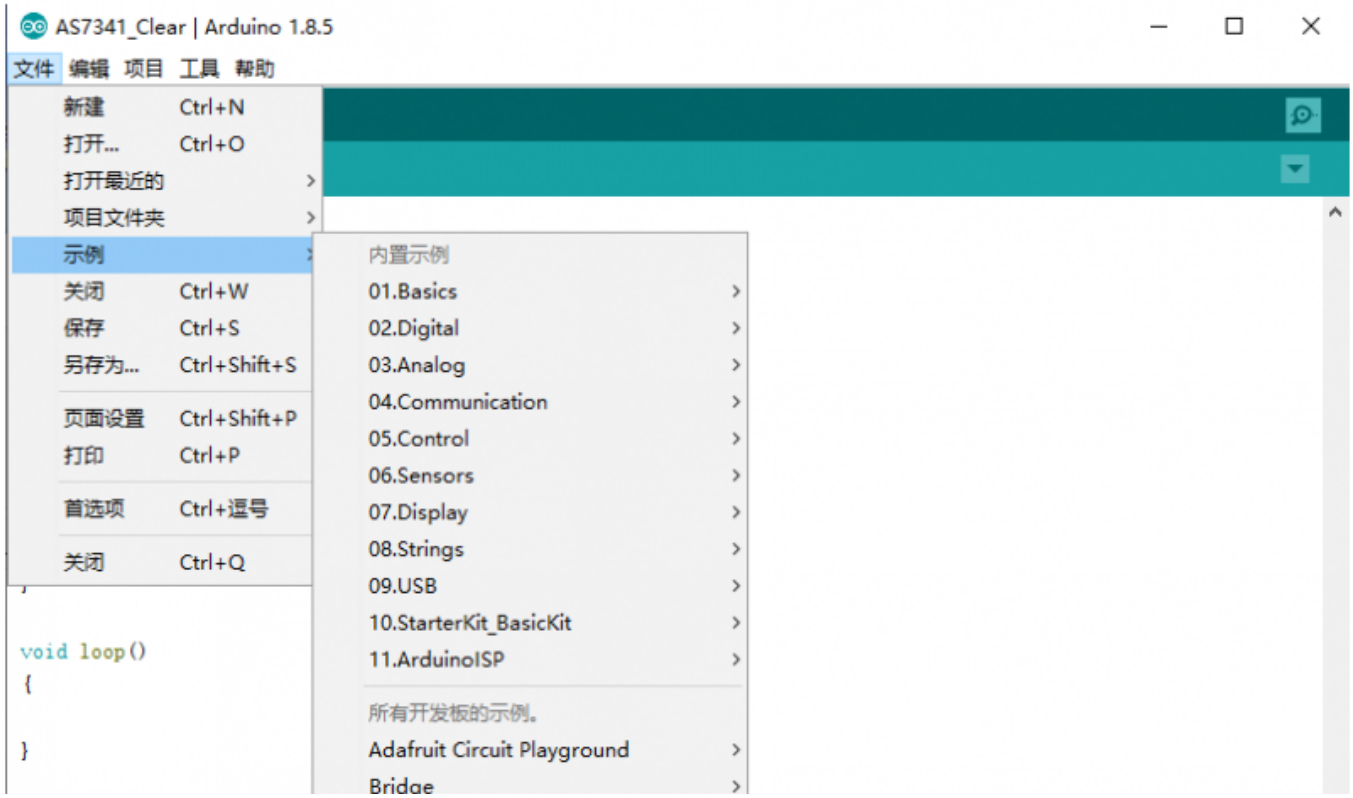


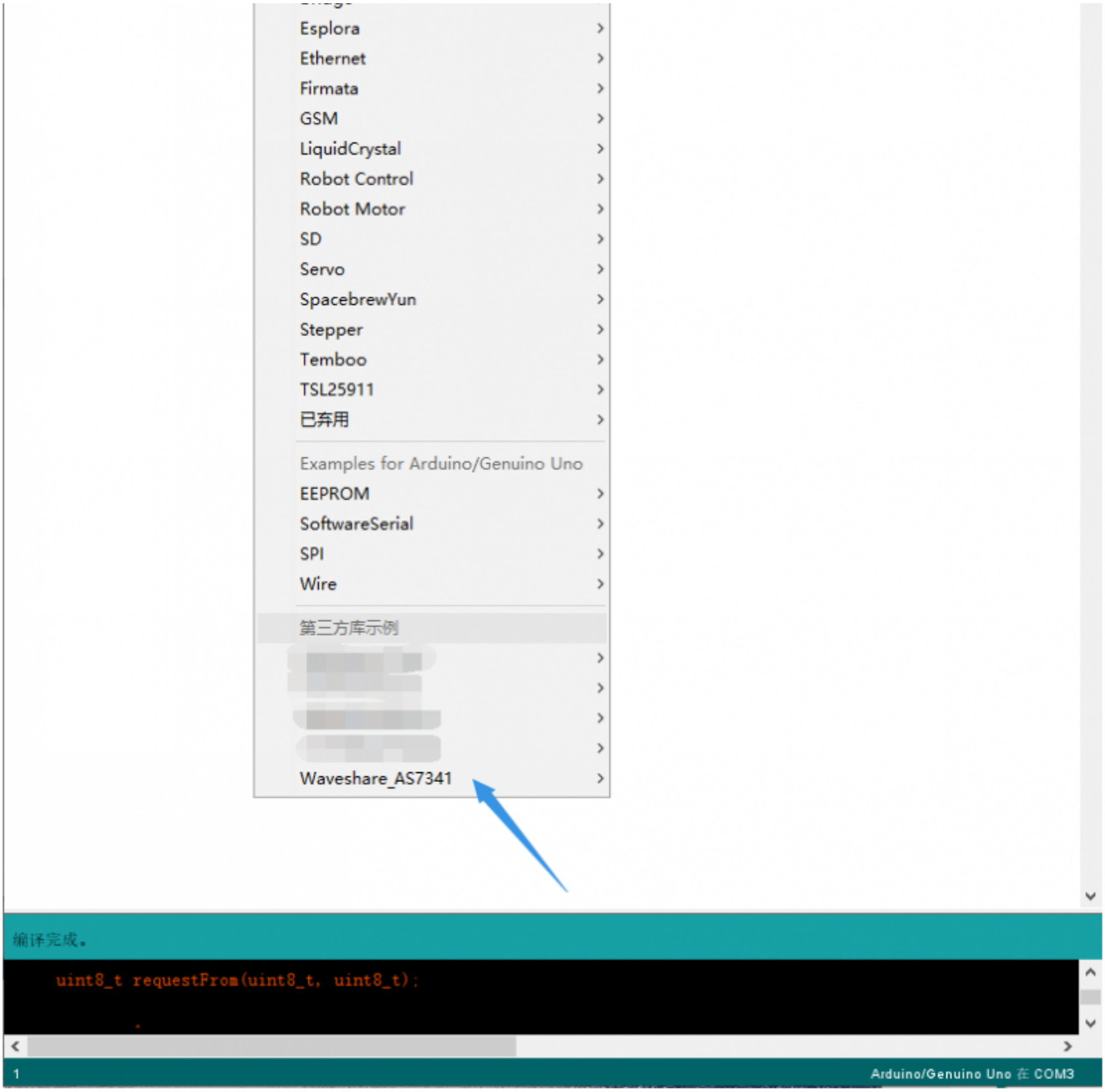
安装编译软件（windows教程）

arduino IDE 安装教程

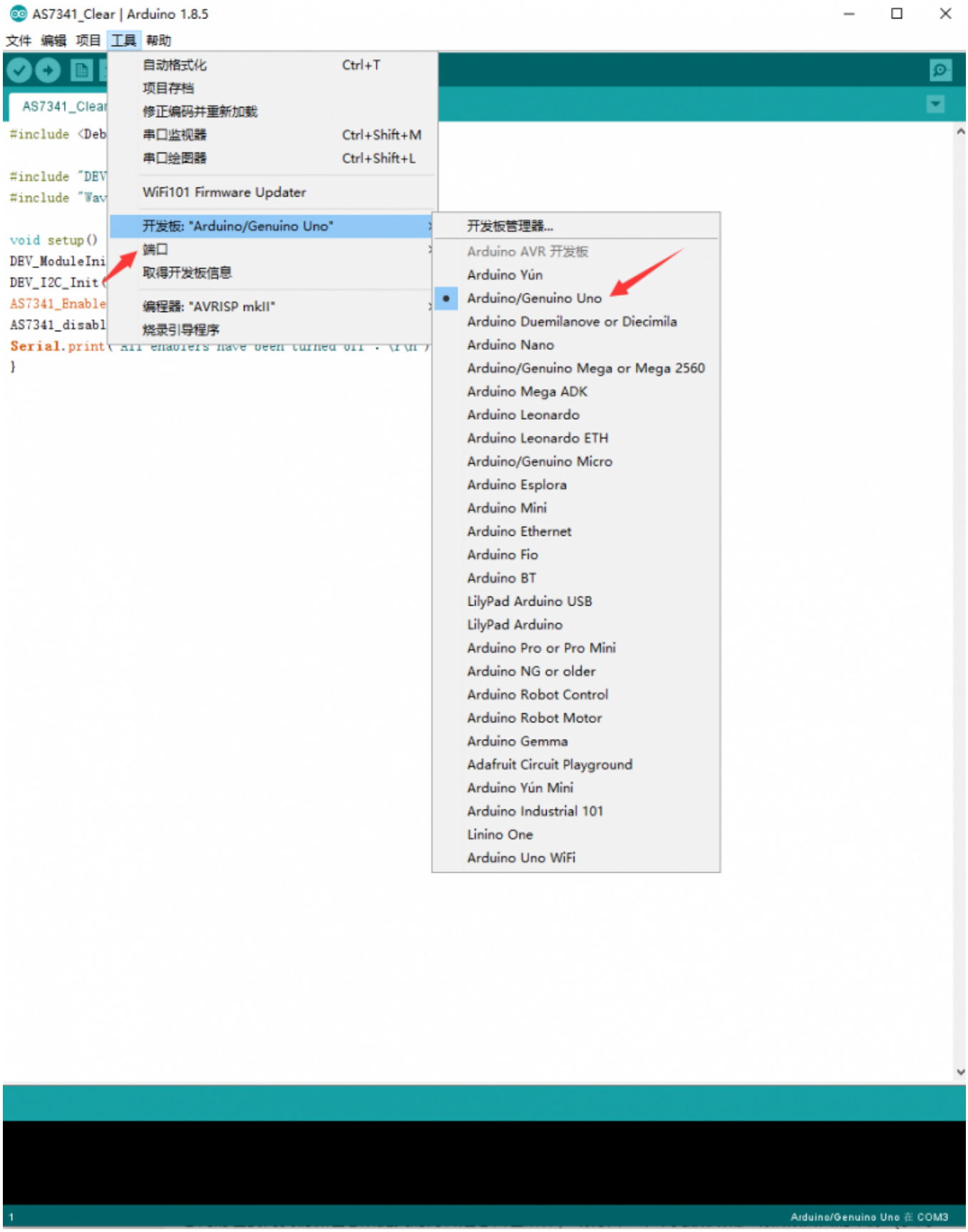
验证程序

在产品百科界面下载程序，然后解压。Arduino程序位于 ~/Arduino/... 把Arduino目录下的文件夹Waveshare_AS7341复制到Arduino安装目录的libraries下，一般是C:\Users\XXX\Documents\Arduino\libraries 或者 C:\Program Files (x86)\Arduino\libraries 打开Arduino IDE: 点击 文件(flie)->示例(example) 加载库，查看是否有Waveshare_AS7341选项，若有说明库导入成功，打开example中的ino工程文件，如图所示：

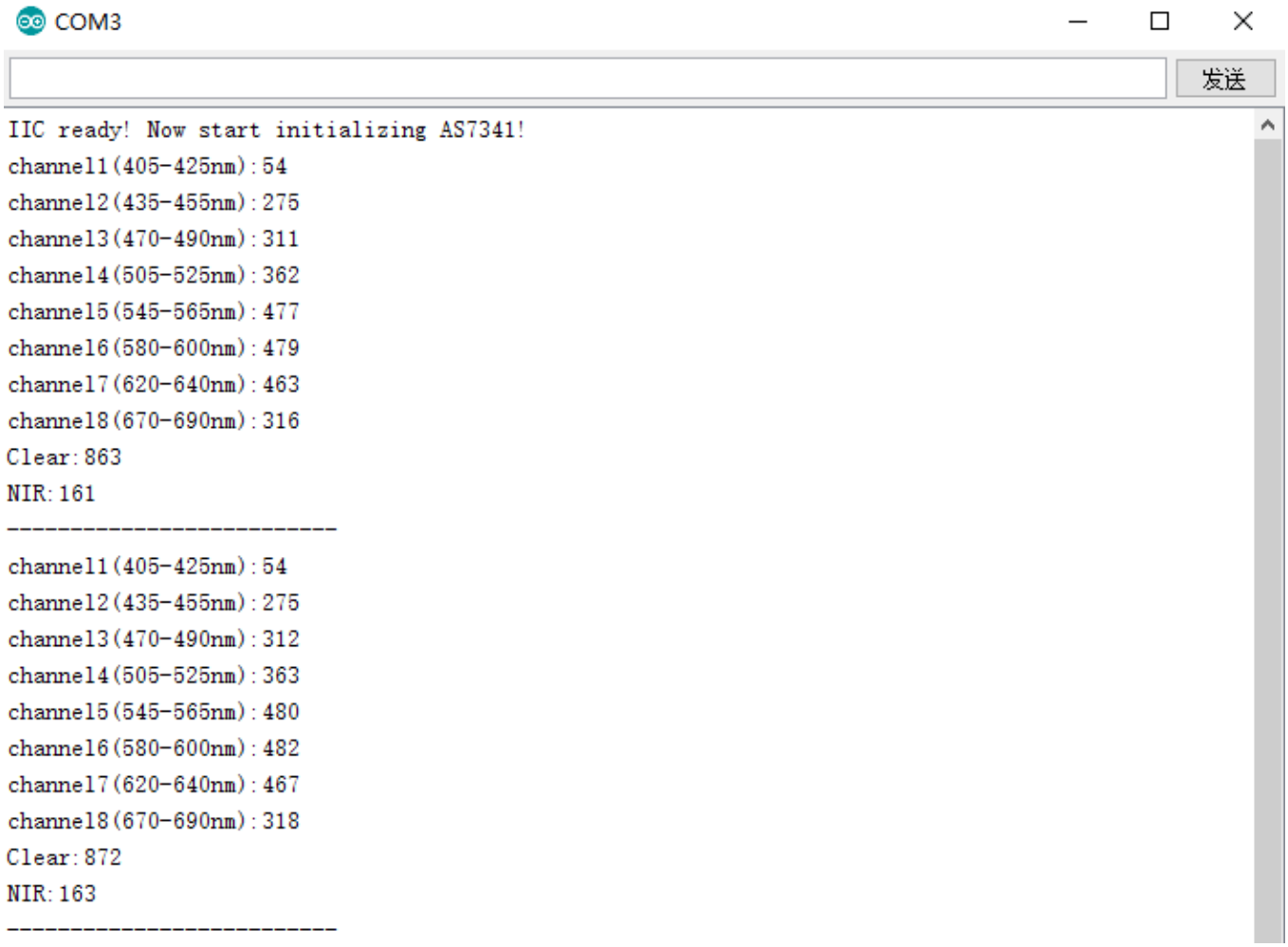




开发板选择相应的型号，选择相应的COM口，编译程序，下载到UNO上,打开串口监视器


AS7341_Clear | Arduino 1.8.5
文件 编辑 项目 工具 帮助
自动格式化 Ctrl+T
项目存档
修正编码并重新加载
串口监视器 Ctrl+Shift+M
串口绘图器 Ctrl+Shift+L
WiFi101 Firmware Updater
开发板: "Arduino/Genuino Uno"
端口
取得开发板信息
编程器: "AVRISP mkII"
烧录引导程序
开发板管理器...
Arduino AVR 开发板
Arduino Yún
Arduino/Genuino Uno
Arduino Duemilanove or Diecimila
Arduino Nano
Arduino/Genuino Mega or Mega 2560
Arduino Mega ADK
Arduino Leonardo
Arduino Leonardo ETH
Arduino/Genuino Micro
Arduino Esplora
Arduino Mini
Arduino Ethernet
Arduino Fio
Arduino BT
LilyPad Arduino USB
LilyPad Arduino
Arduino Pro or Pro Mini
Arduino NG or older
Arduino Robot Control
Arduino Robot Motor
Arduino Gemma
Adafruit Circuit Playground
Arduino Yún Mini
Arduino Industrial 101
Linino One
Arduino Uno WiFi
Arduino/Genuino Uno 在 COM3

实验现象：



程序说明

例程说明

在AS7341目录下的example里存放着不同功能的测试工程，以下按照文件名对需要注意的地方依次说明

AS7341_Getdata用于获得10个通道的测试数据，AS7341只有6个独立ADC，但却有11个通道，这就必须用到多路复用器SMUX。 具体相关配置请对照数据手册参考代码
AS7341_Getdata中包括了打开补光LED及调节亮度的驱动代码

```
//AS7341_EnableLED(true);// LED ON or OFF
```

```
//AS7341_ControlLed(10);//Adjust the brightness of the LED lamp
```

如需使用到LED补光，注释这两行代码即可

AS7341_Getflicker用于检测100或120Hz的环境光闪烁，需要自行产生一个该频率的闪烁光，调节积分时间、增益等可检测到不同频率的闪烁

AS7341_Syns将传感器模式配置为SYNS模式，在该模式下，传感器的GPIO口需要接收一个下降沿信号才能触发测量，每一个下降沿触发一次测量
模块默认没有将GPIO口与某根引脚直接连接，在测试时，将GPIO口与开发板的3.3v或5V脚短暂接触再断开

产生一个下降沿信号即可

如在实际使用中需要用到这个模式，再将GPIO口连接到触发源上即可

```
while(!AS7341_MeasureComplete());//当GPIO接收到有效信号时循环跳出
```

AS7341_INT为光谱中断测试，设置中断产生的上下限阈值，同时可以设置中断触发的通道，通道选择可以是CH0-CH4中的某一个，当环境光变化导致中断被触发时，读取相关寄存器即可

```
AS7341_SetInterruptPersistence(0);//设置光谱中断持久性（通俗的讲就是产生中断的灵敏度）
```

```
AS7341_SetSpectralThresholdChannel(4);//设置检测中断的通道
```

AS7341_pinINT是对模块上INT引脚的实验，AS7341每次测量完成后，INT引脚就会变为低电平，配置相关寄存器可以设置传感器多久测量一次环境的光谱数据，INT引脚也因此会多久跳变一次。该例程中将测量时间设置为了1s，同时对INT引脚的电平状态做了监测。

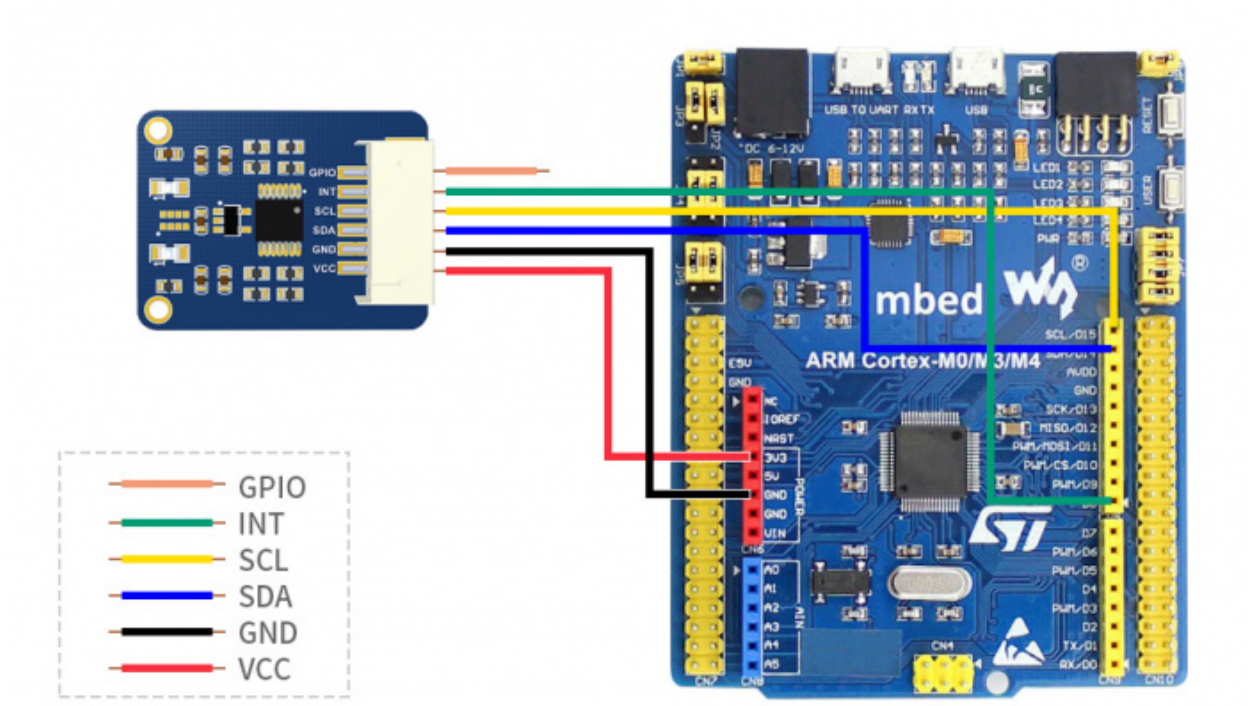
AS7341_Clear是对AS7341上述所有开启的寄存器使能位进行复位

STM32

本例程在NUCLEO-F103RB（芯片型号STM32RBT6）及 OpenH743I-C（芯片型号STM32H743IIT6）上验证通过，如需移植，请注意相关配置及连接方式

硬件连接

与XNUCLEO-F103RB的连接：



AS7341 Spectral Color Sensor	XNUCLEO-F103RB

VCC	3.3V/5V
GND	GND
SDA	SDA/D14/PB9
SCL	SCL/D15/PB8
INT	D8/PA9
GPIO	-

与OpenH743I-C 的连接：

AS7341 Spectral Color Sensor	OpenH743I-C
VCC	3.3V/5V
GND	GND
SDA	PD13(I2C4 SDA)
SCL	PD12(I2C4 SCL)
INT	PD11
GPIO	-

程序说明

在产品百科界面下载程序，然后解压。STM32程序位于 ~/ STM32/... 中，可以看到NUCLEO-F103RB、OpenH743I-C两个文件夹

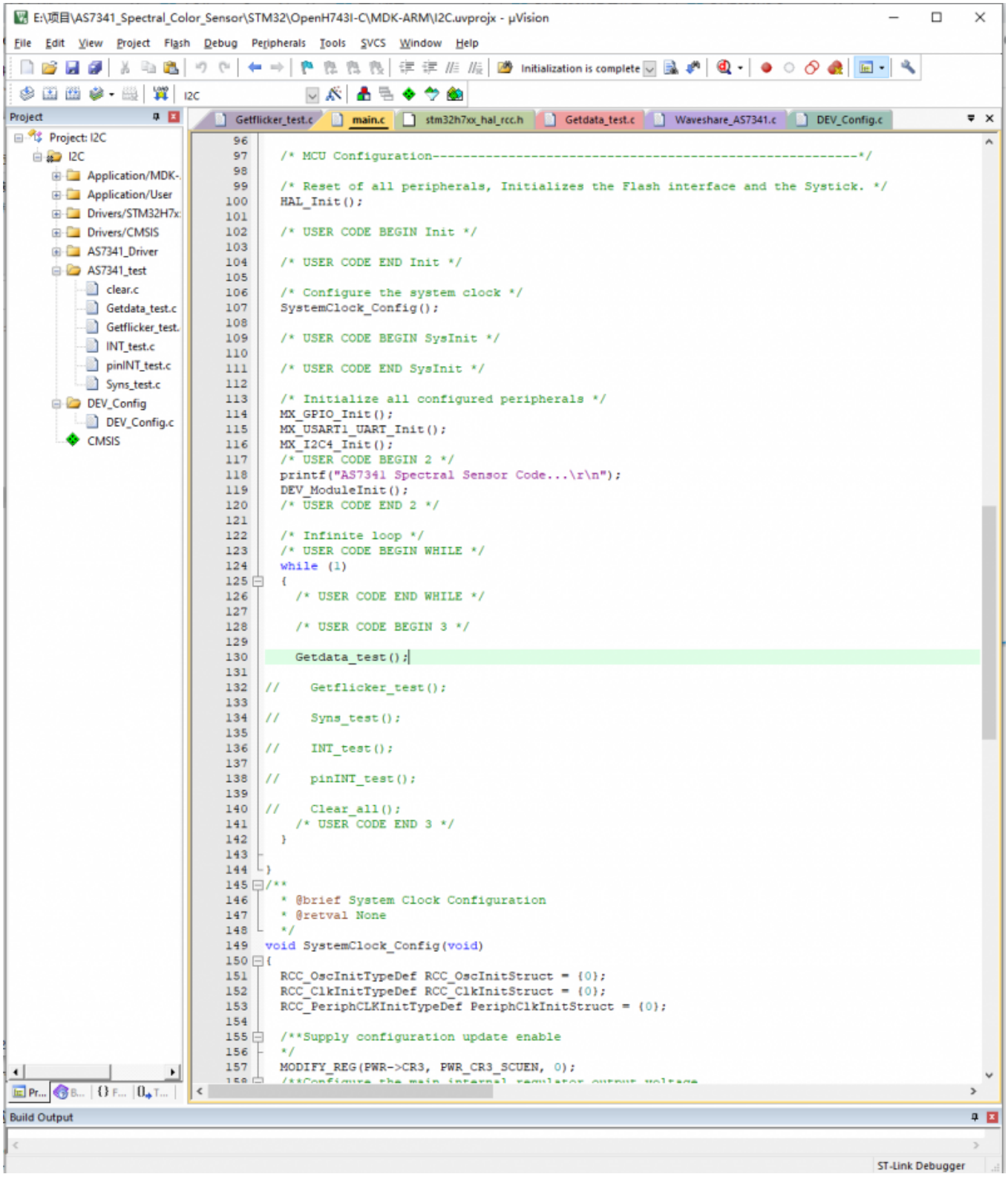
NUCLEO-F103RB

打开STM32中的\XNUCLEO-F103RB\MDK-ARM\demo.uvprojx，使用Keil uVision5打开。例程使用的是HAL库。如需换芯片或者想使用标准库你只需要更改DEV_Config.c和.h，实现里面的函数和宏定义即可。换芯片也可以使用STM32CubeMX进行配置。例程使用串口2（PA2,PA3）输出数据。串口波特率为115200，其他默认：数据位8位，停止位1位，没有校验。串口助手工具在资料文件夹里有提供。

OpenH743I-C

打开STM32中的\OpenH743I-C\MDK-ARM\I2C.uvprojx，使用Keil uVision5打开。例程同样使用的是HAL库。

两者在芯片信号和外设配置等方面有所不同，但使用的测试程序是完全一样的 我们以OpenH743I-C为例，打开工程中的main.c

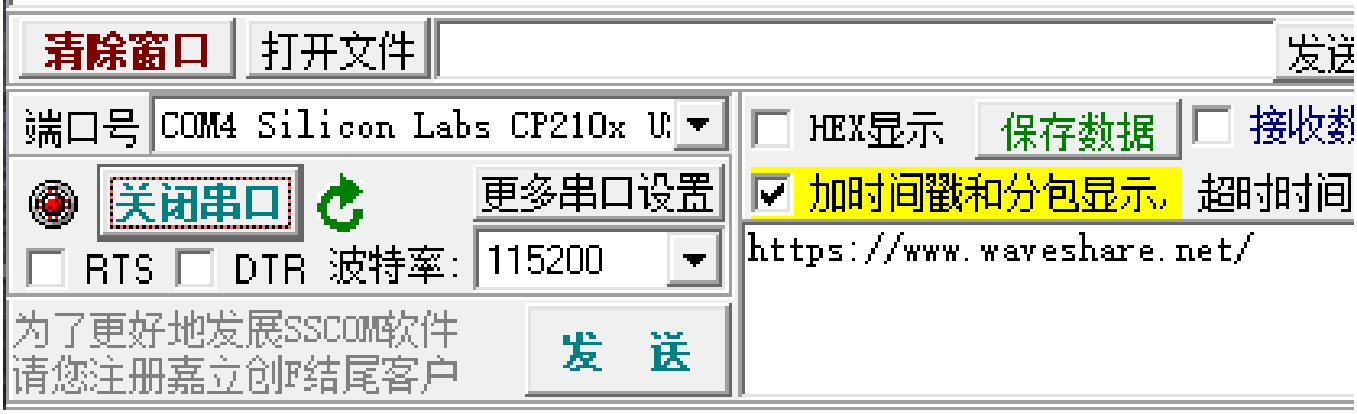


取消注释需要测试的程序，以图示程序为例，接上下载器，串口数据线接至USART1，点击编译下载验证即可

相关的程序用途及说明已经在Arduino教程里说明过了，可到Arduino章节页面查看，这里不再赘述

实验结果：

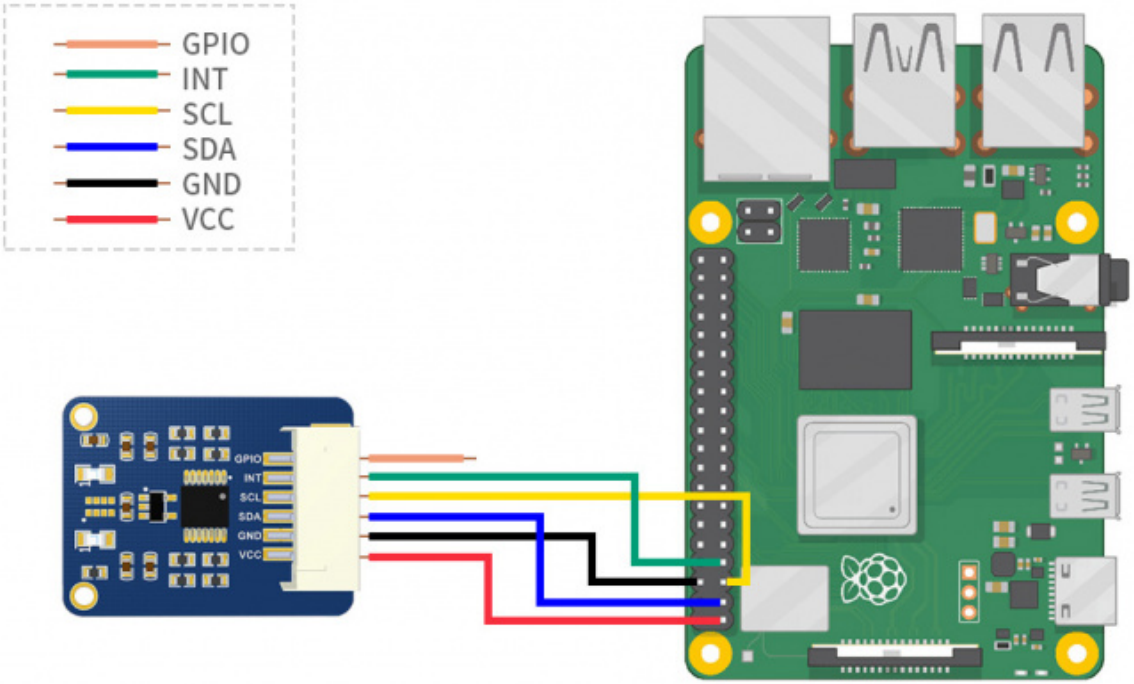
```
[16:02:28.939]收←◆channel1(405-425nm):
25
channel2(435-455nm):
80
channel3(470-490nm):
105
channel4(505-525nm):
146

[16:02:29.559]收←◆channel5(545-565nm):
192
channel6(580-600nm):
201
channel7(620-640nm):
193
channel8(670-690nm):
125
Clear:
577
NIR:
94

[16:02:30.681]收←◆channel1(405-425nm):
25
channel2(435-455nm):
80
channel3(470-490nm):
106
channel4(505-525nm):
147
```



Raspberry Pi

本例程使用的是Raspberry Pi 3 Model B，提供BCM2835、WiringPi、文件IO、RPI（Python）库例程

硬件连接



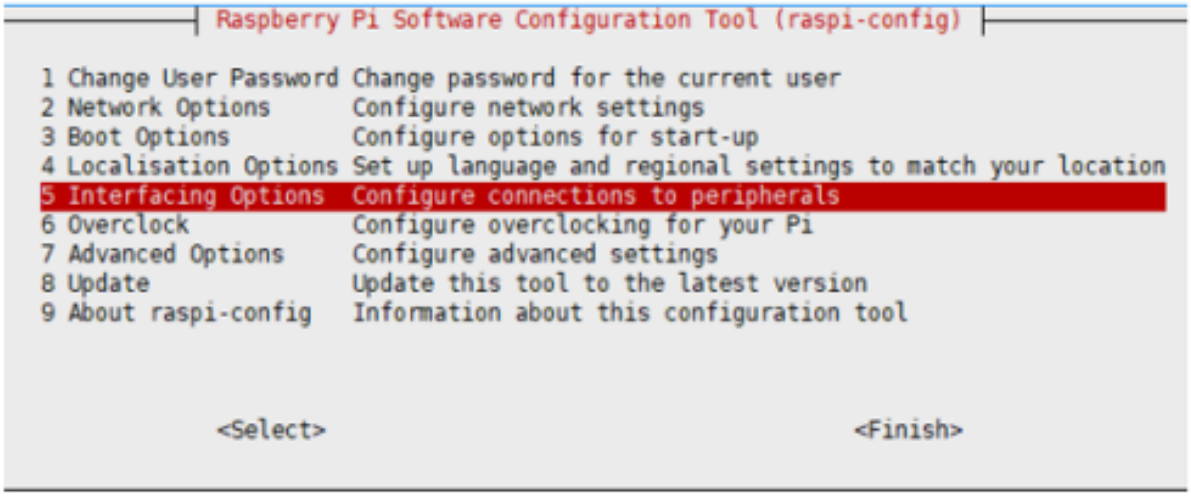
AS7341 Spectral Color Sensorr	树莓派（BCM）
VCC	3.3V/5V
GND	GND
SDA	SDA(2)
SCL	SCL(3)
INT	4
GPIO	-

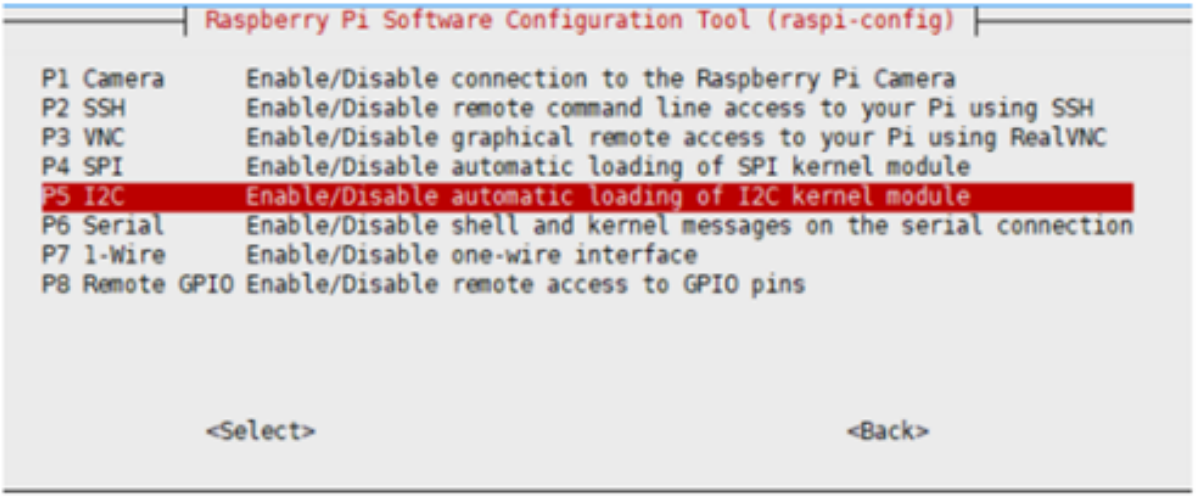
树莓派使用

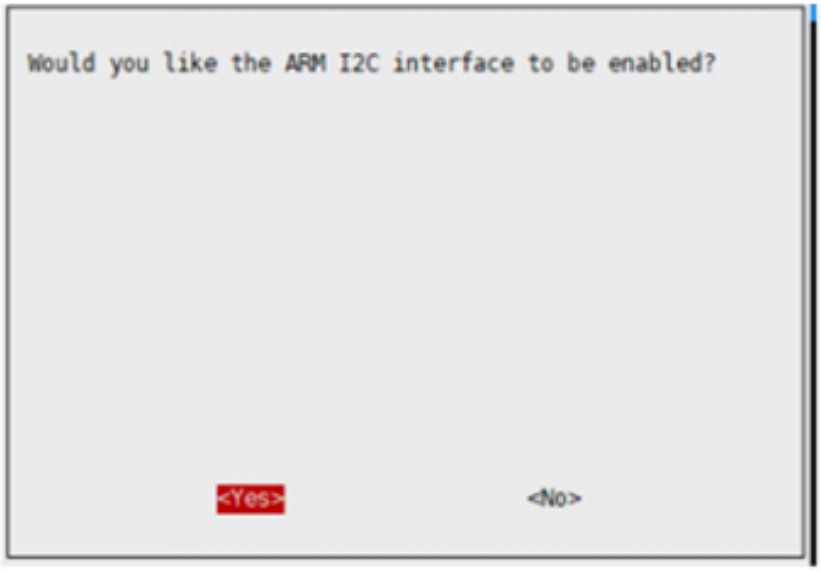
开启I2C接口

- 在终端执行:

```
sudo raspi-config
#选择 Interfacing Options -> I2C ->yes 启动 i2C 内核驱动
```







- 然后重启树莓派

```
sudo reboot
```

安装库

- 安装BCM2835，打开树莓派终端，并运行下指令

```
wget http://www.airspayce.com/mikem/bcm2835/bcm2835-1.60.tar.gz
tar zxvf bcm2835-1.60.tar.gz
cd bcm2835-1.60/
sudo ./configure
sudo make
sudo make check
sudo make install
```

- 安装wiringpi

```
sudo apt-get install wiringpi
#对于树莓派4B可能需要进行升级：
cd /tmp
wget https://project-downloads.drogon.net/wiringpi-latest.deb
sudo dpkg -i wiringpi-latest.deb
gpio -v
```

下载并运行测试例程

```
sudo apt-get install p7zip-full
wget https://www.waveshare.net/w/upload/b/b3/AS7341_Spectral_Color_Sensor_code.7z
7z x AS7341_Spectral_Color_Sensor_code.7z -r -o./AS7341_Spectral_Color_Sensor_code
sudo chmod 777 -R  AS7341_Spectral_Color_Sensor_code
```

C程序

```
cd AS7341_Spectral_Color_Sensor_code/RaspberryPi/c
make clean
make
```

执行

```
sudo ./main data
```

来验证测试程序

- 注意：这里的data可换为flicker、syns、int、pinint、clear以验证不同的测试程序

data 对应 Arduino 的 AS7341_Getdata 例程
flicker 对应 Arduino 的 AS7341_Getflicker 例程
syns 对应 Arduino 的 AS7341_Syns 例程
int 对应 Arduino 的 AS7341_INT 例程
pinint 对应 Arduino 的 AS7341_pinINT 例程
clear 对应 Arduino 的 AS7341_Clear 例程

- 以执行sudo ./main data为例，测试结果为：

```
pi@raspberrypi:~/AS7341_Spectral_Color_Sensor_code/AS7341_Spectral_Color_Sensor_code/RaspberryPi/c $ sudo
 ./main data
USE_DEV_LIB
Current environment: Raspbian
DEV I2C Device
DEV I2C Device
Initialization is complete!
channel1(405-425nm):
37
channel2(435-455nm):
99
channel3(470-490nm):
191
channel4(505-525nm):
164
channel5(545-565nm):
238
channel6(580-600nm):
260
channel7(620-640nm):
197
channel8(670-690nm):
117
Clear:
973
NIR:
111
-------------------------
```

python程序

```
cd
cd AS7341_Spectral_Color_Sensor_code/AS7341_Spectral_Color_Sensor_code/RaspberryPi/python/examp
les
```

执行

```
sudo python data.py
```

来验证测试程序

- 注意：这里的data可换为flicker、syns、int、pinint、clear以验证不同的测试程序

data 对应 Arduino 的 AS7341_Getdata 例程
flicker 对应 Arduino 的 AS7341_Getflicker 例程
syns 对应 Arduino 的 AS7341_Syns 例程
int 对应 Arduino 的 AS7341_INT 例程
pinint 对应 Arduino 的 AS7341_pinINT 例程
clear 对应 Arduino 的 AS7341_Clear 例程

- 以执行data.py为例，测试结果为：

```
pi@raspberrypi:~ $ cd AS7341_Spectral_Color_Sensor_code/AS7341_Spectral_Color_Sensor_code/RaspberryPi/python/examples
pi@raspberrypi:~/AS7341_Spectral_Color_Sensor_code/AS7341_Spectral_Color_Sensor_code/RaspberryPi/python/examples $ sud
o python data.py
channel1(405-425nm):

259

channel2(435-455nm):

338

channel3(470-490nm):

202

channel4(505-525nm):

123

channel5(545-565nm):

258

channel6(580-600nm):

338

channel7(620-640nm):

202

channel8(670-690nm):

123

Clear:

1368

NIR:

149

-----------------------
```

程序说明

所有测试程序的功能及需要注意的地方已在Arduino教程中介绍过，同样的，当执行sudo ./main syns 或python syns.py时，需要给GPIO口上拉再下拉产生一个下降沿信号，可将GPIO脚接到短暂的接触高电平引脚再放开以此来产生一个下降沿信号

资料

文档

- 原理图

程序

- 示例程序

软件

- Arduino IDE
- 串口调试助手

数据手册

- AS7341

FAQ

问题：为什么我按教程中所说的上电操作之后，串口只打印一点提示语就不再输出任何数据或者输出的数据都为0?

请检查硬件连接是否OK，尤其是SDA和SCL的线序不要搞反，并将传感器断电后重新接上并重新运行程序