OVERVIEW

This is a color sensor module based on TCS34725, will output RGB data and light intensity through the I2C interface. Its advantages include high sensitivity, wide dynamic range, accurate measuring, etc.

SPECIFICATION

<table>
<thead>
<tr>
<th>Working voltage:</th>
<th>3.3V/5V</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controller:</td>
<td>TCS34725FN</td>
</tr>
<tr>
<td>IO voltage:</td>
<td>3.3V/5V</td>
</tr>
<tr>
<td>Interface:</td>
<td>I2C</td>
</tr>
<tr>
<td>Dimension:</td>
<td>27 x 20(mm)</td>
</tr>
</tbody>
</table>

INTERFACE

<table>
<thead>
<tr>
<th>PIN</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCC</td>
<td>3.3V/5V</td>
</tr>
<tr>
<td>GND</td>
<td>Ground</td>
</tr>
<tr>
<td>SDA</td>
<td>I2C Data Input</td>
</tr>
<tr>
<td>SCL</td>
<td>I2C Clock Input</td>
</tr>
<tr>
<td>INT</td>
<td>Interrupt Output (Open drain output)</td>
</tr>
<tr>
<td>LED</td>
<td>LED</td>
</tr>
</tbody>
</table>
CONTENTS

Overview ...........................................................................................................................................1

Specification ......................................................................................................................................1

Interface ...........................................................................................................................................1

Hardware ..........................................................................................................................................3

Controller .........................................................................................................................................3

Communication protocol ..................................................................................................................3

I2C write ...........................................................................................................................................3

I2C Read ...........................................................................................................................................4

I2C address .......................................................................................................................................4

How to use .......................................................................................................................................5

Download examples ..........................................................................................................................5

examples ...........................................................................................................................................5

Raspberry Pi ......................................................................................................................................5

STM32 ................................................................................................................................................10

Arduino .............................................................................................................................................12

FAQ ..................................................................................................................................................14
HARDWARE

CONTROLLER

TCS34725 is used for color sensing. TCS34725 is an I2C bus-based color light-to-digital converter with IR filter, provides a digital return of red, green, blue (RGB) and clear light sensing values. The high sensitivity, wide dynamic range and IR blocking filter make the TCS34725 an ideal color sensor solution for use under varying lighting conditions and through attenuating materials.

COMMUNICATION PROTOCOL

I2C bus has two lines, one is data line (SDA) and another is lock line (SDL). There are three kinds of signals when communicating, Start signal, Stop signal and Answer signal.

Start signal: When SCL is High, SDA change from High to Low, it start to transmit data

Stop signal: When SCL is High, SDA change from Low to High, it stop transmitting.

Answer signal: Every time IC send back a certain Low plus to sender after it receives 8 bits data.

I2C WRITE
When working, Raspberry Pi (hereafter named as Master) will first send a Start signal, then send a byte to TCS34725 (hereafter named as Slaver), whose first 7 bits are address of Slaver and 1 bit write bit. Slave response with Answer signal every time it receives any data. Master send command register address to Slaver, then data of command register. Stop signals is sent to slave to stop communicating.

**I2C READ**

When working, Master will first send a Start signal, then send a byte to Slaver, whose first 7 bits are address of Slaver and 1 bit write bit. Slave response with Answer signal every time it receives any data. Master send command register address to Slaver. After that, Mater will send a Start signal again, and then send a byte (7 bits address and 1 bit read bit) to Slaver. Slaver response and send data of the register to Master, master answer as well. Stop signals will be sent to stop communicating.

**I2C ADDRESS**

The I2C device address of TCS34725 is 0x29
TCS34725 Color Sensor User Manual

<table>
<thead>
<tr>
<th>Device</th>
<th>Address</th>
<th>Package-Leads</th>
<th>Interface Description</th>
<th>Ordering Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCS34725</td>
<td>0x29</td>
<td>FN–6</td>
<td>$V_{BUS} = V_{DD}$ Interface</td>
<td>TCS34725FN</td>
</tr>
<tr>
<td>TCS34727</td>
<td>0x29</td>
<td>FN–6</td>
<td>$V_{BUS} = 1.8$ V Interface</td>
<td>TCS34727FN</td>
</tr>
</tbody>
</table>

**TCS3472 datasheet page 34**

Note: 0x29 is 7bit in fact, therefore, when you set the I2C address, you should left-shift one bit, turn it to 0x52

**HOW TO USE**

**DOWNLOAD EXAMPLES**

Find and download examples from Waveshare wiki:

**Resources**

- User Manual
- **Demo code**
- Schematic

**Datasheet**

Extract the 7z you get:

- Arduino: examples for Arduino
- Raspberry Pi: examples for Raspberry Pi (wiringPi, python, bcm2835)

**EXAMPLES**

**RASPBERRY PI**

Insert the SD card (Raspbian installed)
Copy the Raspberry Pi examples to SD card:

Insert SD card to Raspberry Pi and power on, you can find the folder is listed in /boot

Copy the examples to /home/pi and change its permission:

---

**INSTALL LIBRARIES**

To run the examples, you need to first install libraries (wiringPi, bcm2835 and python) and enable I2C interface, otherwise example cannot work properly.

**BCM2835**


Download the library from bcm2835 libraries and install:

```
wget http://www.airspayce.com/mikem/bcm2835/bcm2835-1.58.tar.gz
sudo tar zxf bcm2835-1.xx.tar.gz
```
cd bcm2835-1.xx

sudo ./configure

make

sudo make check

sudo make install

**Note:** The xx is the version number you download, for example, if the version you download is bcm2835-1.52. then the command you should execute is sudo tar zxfv bcm2835-1.52.tar.gz

**wiringPi libraries:**

sudo apt-get install git

sudo git clone git://git.drogon.net/wiringPi

cd wiringPi

sudo ./build

**Python libraries:**

sudo apt-get install python-pip

sudo pip install RPi.GPIO

sudo pip install spidev

sudo apt-get install python-imaging

sudo apt-get install python-smbus

**Enable I2C interface:**

sudo raspi-config
Reboot Raspberry Pi and check I2C devices:

```
sudo reboot
i2cdetect -y 1
```
<table>
<thead>
<tr>
<th>TCS34725 Color Sensor</th>
<th>Raspberry Pi</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCC</td>
<td>3.3V</td>
</tr>
<tr>
<td>GND</td>
<td>GND</td>
</tr>
<tr>
<td>SDA</td>
<td>SDA</td>
</tr>
<tr>
<td>SCL</td>
<td>SCL</td>
</tr>
<tr>
<td>INT</td>
<td>17</td>
</tr>
<tr>
<td>LED</td>
<td>18</td>
</tr>
</tbody>
</table>

**RUNNING EXAMPLE**

**BCM2835 example**

```
cd bcm2835
sudo ./main
```

**WiringPi example**

```
cd wiringpi
sudo ./main
```

**python example**

```
cd python
sudo python main.py
```

Note: If you get error information that files are not exist when running BCM2835 or wiringpi example, please execute make command and try again.

**EXPECTED RESULT**
The expected result of three examples are similar, here we take python codes as example:

R, G, B value are printed in RGB888 format (DEC), C is light value without processing.

RGB565 and RGB888 are HEX data printed in certain format. LUX is light value processed. CT is color temperature. (https://en.wikipedia.org/wiki/Color_temperature)

If you want to measure CT, please turn off LED. INT is interrupt, 1: light value is over threshold.

You can turn the RGB value to color with tools below:

https://www.waveshare.com/w/upload/5/53/Infrared-Temperature-Sensor-Code.7z

STM32
Open STM32 project with Keil uVision5. The example is based on HAL libraries.

Development board used is Waveshare XNUCLEO-F103RB, the chip is STM32F103RBT6. Example uses UART2 (PA2, PA3) to print data, 115200, 8N1.

### HARDWARE CONNECTION

<table>
<thead>
<tr>
<th>TCS34725 Color Sensor</th>
<th>STM32</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCC</td>
<td>3.3V</td>
</tr>
<tr>
<td>GND</td>
<td>GND</td>
</tr>
<tr>
<td>SDA</td>
<td>SDA/D14/PB9</td>
</tr>
<tr>
<td>SCL</td>
<td>SCL/D15/PB8</td>
</tr>
<tr>
<td>INT</td>
<td>D8/PA9</td>
</tr>
<tr>
<td>LED</td>
<td>PWM1/D9/PC7</td>
</tr>
</tbody>
</table>

### EXPECTED RESULT

This is the output when testing red

![](image)

You can turn the RGB value to color with tools below:

ARDUINO

The development board used is Waveshare UNO PLUS(Compatible with Arduino UNO), set serial monitor to 115200

HARDWARE CONNECTION

<table>
<thead>
<tr>
<th>TCS34725 Color Sensor</th>
<th>Arduino</th>
</tr>
</thead>
<tbody>
<tr>
<td>VCC</td>
<td>3.3V/5V</td>
</tr>
<tr>
<td>GND</td>
<td>GND</td>
</tr>
<tr>
<td>SDA</td>
<td>SDA</td>
</tr>
<tr>
<td>SCL</td>
<td>SCL</td>
</tr>
<tr>
<td>INT</td>
<td>D8</td>
</tr>
<tr>
<td>LED</td>
<td>D9</td>
</tr>
</tbody>
</table>

EXPECTED RESULT

This is the output when testing red
You can turn the RGB value to color with tools below:

https://www.waveshare.com/w/upload/5/53/Infrared-Temperature-Sensor-Code.7z
FAQ

1. Q: Raspberry Pi example initializing failed?

   A: Please check if you connect sensor correctly, and check i2C device with command `i2cdetect -y 1`

   ![Image of i2cdetect output]

   Please correct connecting and restart

2. Q: What happened when running example by mistake?

   A: If you find that python or bcm2835 examples cannot work properly after running wiringpi codes, please just restart Raspberry Pi can test again

3. Q: Data output are incorrect when using STM32 and Arduino examples?

   A: Please check if you choose the correct COM port (according to device manager).

   If all the setting are correct, please exchange RXD and TXD and try again.
4. Q: Why the RGB data outputted are all 0

![RGB data outputted are all 0](image)

A: Please check if you connect device correctly then press reset button

![TCS34725 initialization error!!](image)

5. Q: The RGB data output are all 253?

![RGB data output are all 253](image)

A: The light intensity value is over measure range, you can try to modify the gain parameter in initial codes, or add statement

```
TCS34725_Set_Gain(TCS34725_GAIN_16X)  following initialize part.
```

6. Q: Color detect is abnormal after modifying integrate time

A: The integrate time is relate to maximum data of RGB channels. If the color turns darker or lighter after modification, please try to change the brightness of LED

7. Q: Why interrupt cannot be triggered or be triggered all the time after modifying integrate time
A: Interrupt is relate to data of Clear channel. Data of Clear channel is influenced by integrate time. When gain is 60:

<table>
<thead>
<tr>
<th>Integrate time</th>
<th>Max value of Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.4ms</td>
<td>1024</td>
</tr>
<tr>
<td>24ms</td>
<td>10240</td>
</tr>
<tr>
<td>50ms</td>
<td>5400</td>
</tr>
<tr>
<td>101ms</td>
<td>21504</td>
</tr>
<tr>
<td>154ms</td>
<td>65535</td>
</tr>
<tr>
<td>700ms</td>
<td>65535</td>
</tr>
</tbody>
</table>

Therefore, you should modify the threshold value if sample rate is fast. And please increase brightness of LED when you set integrate time to 2.4ms.