COLORIMETER

SEN-Color

1. GENERAL INFORMATION

Dear customer,
thank you very much for choosing our product.
In the following, we will introduce you to what to observe while setting up and using this product.
Should you encounter any unexpected problems during use, please do not hesitate to contact us.

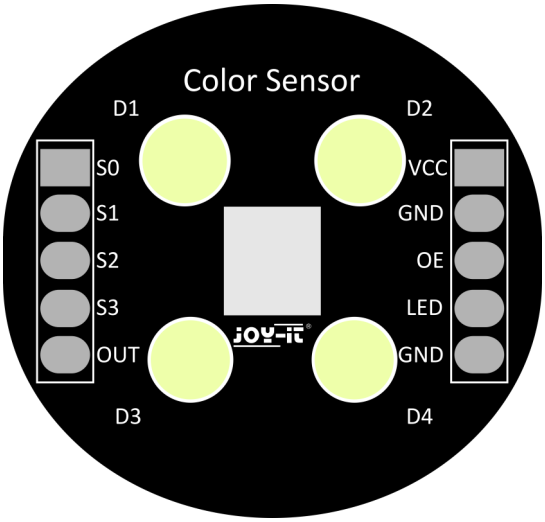
1.1 DESCRIPTION OF THE SENSOR

The TCS3200 chip is designed to detect the color of the light falling on it. It has an array of photodiodes (a matrix of 8x8, for a total of 64 sensors). These photodiodes are covered with four types of filters. Sixteen sensors have a RED filter and thus can only measure the red component in the incident light. Another sixteen have a GREEN filter and sixteen have a BLUE filter. Each visible color can be divided into three basic colors. So these three types of filtered sensors help measure the weighting of each of the primary colors in the incident light. The remaining 16 sensors have a clear filter.

The TCS3200 converts the intensity of the incident lighting into a f requency. The output waveform is a square wave with 50% duty cycle. You can use the timer of a MCU to measure the period of the pulse to determine the frequency. The output of the TCS3200 is available on one line, which means that only 1 pin is available as output.

The intensity of the red, green, blue and clear channels can be determined by using the inputs, S2 and S3, which are used to select the sensors, to provide their output on the output line. That means, the inputs S2 and S3 are used to control what kind of sensors give their signal to the output line. There are also two other inputs S0 and S1, which are used to regulate the output frequency. That means that with these inputs the intensity of the frequency of the channels RED, GREEN, BLUE and CLEAR can be regulated.

1.2 PIN ASSIGNMENT OF THE SENSOR



Pin Name	Description
GND	Ground of the power supply. All voltages are referenced to GND
OE	Enable for OUT (active low)
OUT	Output frequency
S0, S1	Inputs for selecting the scaling of the output frequency
S2, S3	Inputs for selecting the photodiode type
VDD	Supply voltage

1.3 COLOR FILTER AND FREQUENCY SCALING

When you select a color filter for the TCS3200, it can only pass that particular color, so it blocks other colors during this time. The intensity of the red, green, blue and clear channels can be determined by using the inputs, S2 and S3, which are used to select the sensors, to provide their output on the output line.
The TCS3200 has four photodiode types. Red, blue, green and clear, which greatly simplifies the amplitude of the incident light to increase the accuracy and simplify the optics when the light is projected onto the TCS3200. You can select the different types of photodiode by combining different combinations of S2 and S3.

S2	S3	Photodiode type
Low	Low	Red
Low	High	Blue
High	Low	Clear
High	High	Green

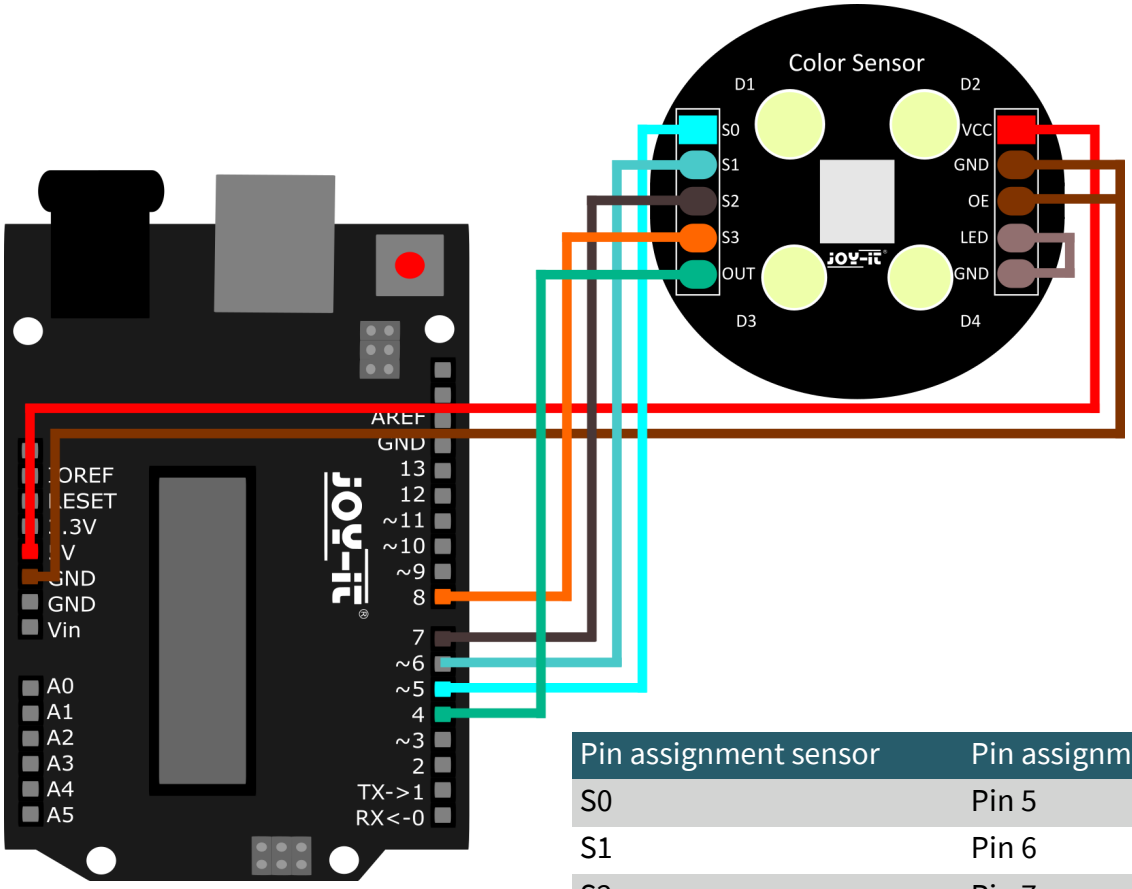
The TCS3200 can output the frequency of different measured square waves corresponding to different colors and light intensities. There is a relationship between the output and the light intensity, so the output of the sensor is affected by the frequency scaling of the measured light intensity. The range of the typical output frequency is 2 Hz ~ 500 kHz. We can apply different scaling factors with different combinations of S0 and S1. This scaling is noticeable by the fact that the size of the output values change, e.g. the scaling is used to adjust the values in certain applications so that they fit more into the frame of the application. In addition, the scaling can also be adjusted to influence the accuracy of the output values.

S0	S1	Output frequency scaling
Low	Low	Power down
Low	High	2%
High	Low	20%
High	High	100%

2. USE WITH THE ARDUINO

In the following you can see how to connect your color sensor to your Arduino. Please follow the connection table and the illustration.

Please note that the sensor should be calibrated before each use. More information about calibration can be found in section 4.1.



Pin assignment sensor	Pin assignment Arduino
S0	Pin 5
S1	Pin 6
S2	Pin 7
S3	Pin 7
OUT	Pin 4
VCC	5V
GND	GND
OE	GND
LED	GND (at the Sensor)
GND	LED (at the Sensor)

2.1 CODEEXAMPLE ARDUINO

After connecting the SEN-Color as shown above, you can now connect the Arduino to your PC via the included cable. After that you can open your Arduino IDE. If you don't have it installed yet, you can download and install it here. After that you can select your Arduino under **Tools > Port**. Under **Tools > Board** select **"Arduino Uno"** and under **Tools > Programmer** select **"ArduinoISP"**.

After the above steps you can now simply copy the following code into your IDE. After loading this code to your Arduino you can open the serial monitor which you can find in the upper right corner of your IDE. You will need to set the serial monitor to 9600 baud if it is not already set to 9600 baud. If all this has been done and the sensor is properly calibrated, you will now see your readings in the range of 0-255.

```
#include <math.h>
#define noPaverageseInterrupts 1

// Assignment of the sensor pins
#define S0 5
#define S1 6
#define S2 7
#define S3 8
#define sensorOut 4

/*Calibration values (must be updated before
  updated before each use)*/
int redMin = 0;
int redMax = 1;
int greenMin = 0;
int greenMax = 1;
int blueMin = 0;
int blueMax = 1;

int redColor = 0;
int greenColor = 0;
int blueColor = 0;

int redFrequency = 0;
int redEdgeTime = 0;
int greenFrequency = 0;
int greenEdgeTime = 0;
int blueFrequency = 0;
int blueEdgeTime = 0;

void setup()
{
  /*definition of the sensor pins*/
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(S3, OUTPUT);
  pinMode(sensorOut, INPUT);

  /*Scaling the output frequency
    S0/S1
    LOW/LOW=AUS, LOW/HIGH=2%,
    HIGH/LOW=20%, HIGH/HIGH=100%*/
  digitalWrite(S0, HIGH);
  digitalWrite(S1, HIGH);

  Serial.begin(9600);

}

//Initialization of the loop
void loop() {

  Serial.println("-------------------------------");

  {
    /*Determination of the photodiode type during measurement
      S2/S3
      LOW/LOW=RED, LOW/HIGH=BLUE,
      HIGH/HIGH=GREEN, HIGH/LOW=CLEAR*/
    digitalWrite(S2, LOW);
    digitalWrite(S3, LOW);
```

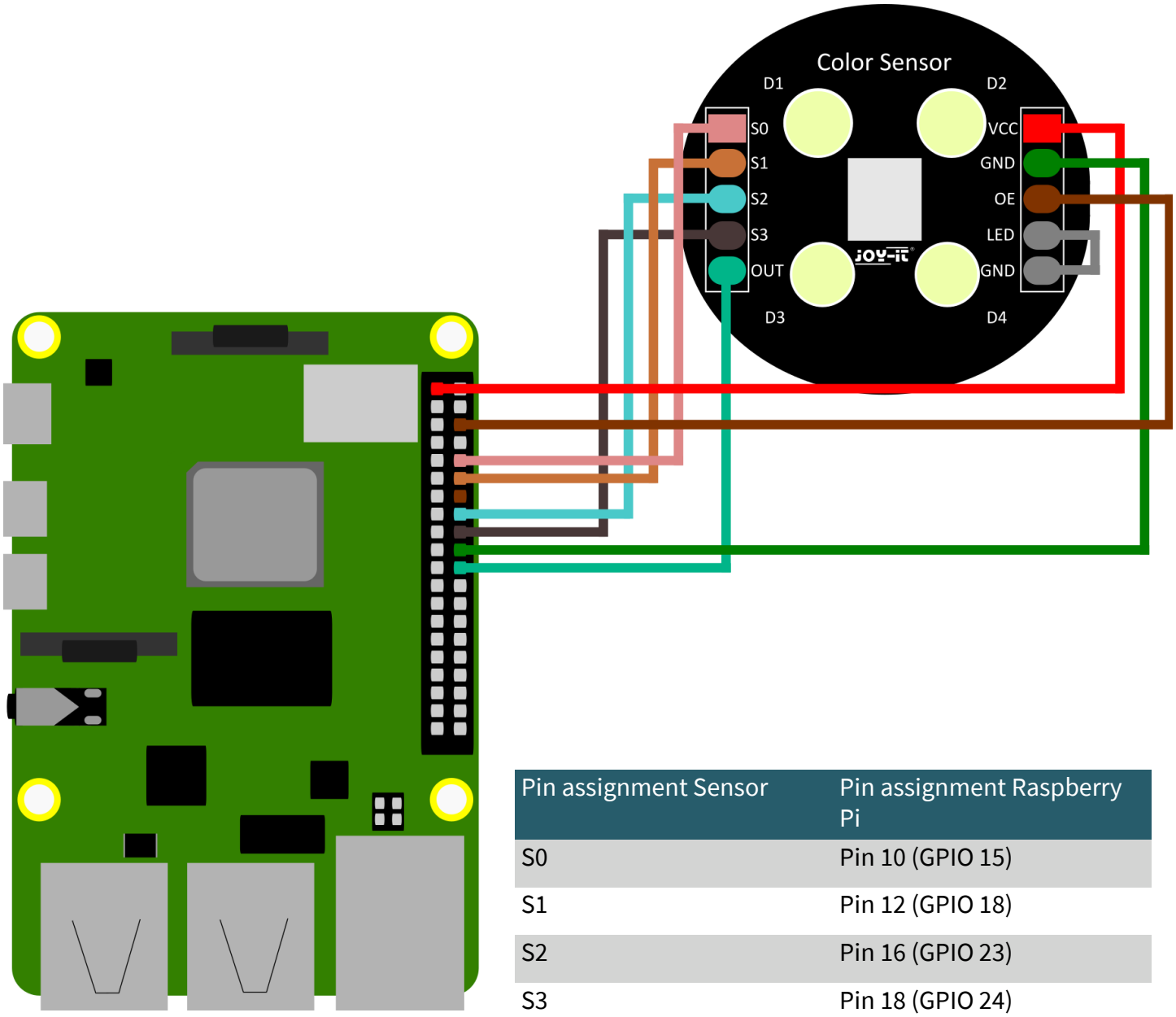
```
    /*Frequency measurement of the specified color and its as-
signment to an RGB value between 0-255*/
    float(redEdgeTime) = pulseIn(sensorOut, HIGH) + pulseIn
(sensorOut, LOW);
    float(redFrequency) = (1 / (redEdgeTime / 1000000));
    redColor = map(redFrequency, redMax, redMin, 255, 0);
    if (redColor > 255) {
      redColor = 255;
    }
    if (redColor < 0) {
      redColor = 0;
    }
    /*Output of frequency mapped to 0-255*/
    Serial.print("Red Frequency: ");
    Serial.println(redFrequency);
    Serial.print("R = ");
    Serial.println(redColor);
    delay(100);
  }

  {
    digitalWrite(S2, HIGH);
    digitalWrite(S3, HIGH);
    /*Frequency measurement of the specified color and its as-
signment to an RGB value between 0-255*/
    float(greenEdgeTime) = pulseIn(sensorOut, HIGH) + pulseIn
(sensorOut, LOW);
    float(greenFrequency) = (1 / (greenEdgeTime / 1000000));
    greenColor = map(greenFrequency, greenMax, greenMin, 255,
0);
    if (greenColor > 255) {
      greenColor = 255;
    }
    if (greenColor < 0) {
      greenColor = 0;
    }
    /*Output of frequency mapped to 0-255*/
    Serial.print("Green Frequency: ");
    Serial.println(greenFrequency);
    Serial.print("G = ");
    Serial.println(greenColor);
    delay(100);
  }

  {
    digitalWrite(S2, LOW);
    digitalWrite(S3, HIGH);
    /*Frequency measurement of the specified color and its as-
signment to an RGB value between 0-255*/
    float(blueEdgeTime) = pulseIn(sensorOut, HIGH) + pulseIn
(sensorOut, LOW);
    float(blueFrequency) = (1 / (blueEdgeTime / 1000000));
    blueColor = map(blueFrequency, blueMax, blueMin, 255, 0);
    if (blueColor > 255) {
      blueColor = 255;
    }
    if (blueColor < 0) {
      blueColor = 0;
    }
    /*Output of frequency mapped to 0-255*/
    Serial.print("Blue Frequency: ");
    Serial.println(blueFrequency);
    Serial.print("B = ");
    Serial.println(blueColor);
```

```
    delay(100);
  }

  Serial.println("-----------------------------");
  delay(100);

  /*Determination of the measured color by comparison
  with the values of the other measured colors*/
  if (redColor > greenColor and redColor > blueColor) {
    Serial.println("Red detected ");
    delay(100);
  }

  if (greenColor > redColor and greenColor > blueColor) {
    Serial.println("Green detected ");
    delay(100);
  }

  if (blueColor > redColor and blueColor > greenColor) {
    Serial.println("Blue detected ");
    delay(100);
  }

  if (blueColor > 0 and greenColor > 0 and redColor > 0) {
    Serial.println("Please place an Object in front of the
Sensor");
    delay(100);
  }

  Serial.println("-----------------------------");
  delay(1000);
}
```

3. USE WITH THE RASPBERRY PI

In the following you can see how to connect your color sensor to your Raspberry Pi. Please follow the connection table and the illustration (for more accurate measurement results we recommend to use a voltage translator, like our KY-051, and then run the sensor with 5 V).

Please note that the sensor should be calibrated before each use. For more information about calibration, please refer to section 4.1.

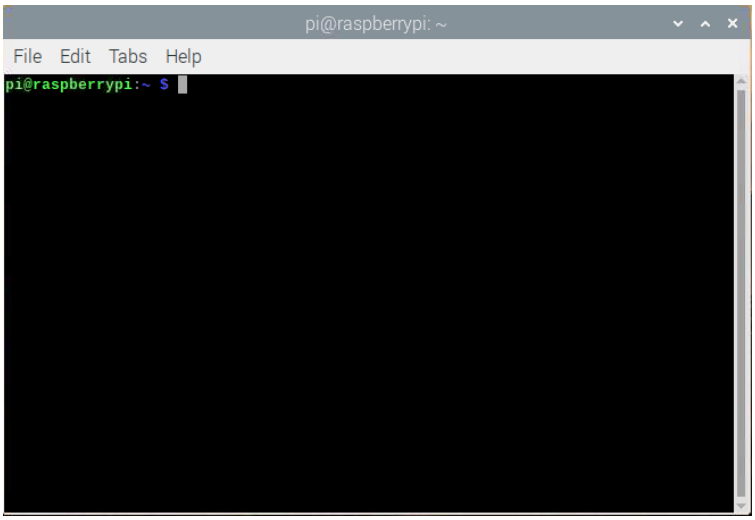


Pin assignment Sensor	Pin assignment Raspberry Pi
S0	Pin 10 (GPIO 15)
S1	Pin 12 (GPIO 18)
S2	Pin 16 (GPIO 23)
S3	Pin 18 (GPIO 24)
VCC	Pin 1 / 3V3
GND	Pin 20 / GND
OE	Pin 6 / GND
LED	GND (at the Sensor)
GND	LED (at the Sensor)
OUT	Pin 22 (GPIO 25)

3.1. CODEEXAMPLE RASPBERRY PI

After connecting the SEN-Color as shown before, we will now connect the Raspberry Pi to a monitor and we will connect a keyboard and a mouse. After setting up the Raspberry Pi for the first time, we will now open the console.

After the above steps you should see the following window in front of you.



In the next step, we will now create a new file by adding

```
sudo nano SEN-Color.py
```

and copy the following code into the opened new file by going to **Edit > Paste.**

Please note that the calibration values must be updated before you can use the sensor properly!

```
import RPi.GPIO as GPIO
import time

#Allocation of the sensor pins
s0 = 15
s1 = 18
s2 = 23
s3 = 24
signal = 25

NUM_CYCLES = 500

GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)

def setup():
```

```
  GPIO.setup(signal,GPIO.IN, pull_up_down=GPIO.PUD_UP)
  GPIO.setup(s0,GPIO.OUT)
  GPIO.setup(s1,GPIO.OUT)
  GPIO.setup(s2,GPIO.OUT)
  GPIO.setup(s3,GPIO.OUT)

def calc(raw, cN, out):
  #Calibration values (must be updated before
  #each use)
  #SEE INSTRUCTIONS
  redMin = 0
  redMax = 1
  greenMin = 0
  greenMax = 1
  blueMin = 0
  blueMax = 1

  out_from = 255
  out_to = 0

  colourTime = 1
  #Frequency measurement of the specific color
  for impulse_count in range(NUM_CYCLES):
    GPIO.wait_for_edge(signal, GPIO.RISING)
    if colourTime is 1:
      start = time.time()
      colourTime = 0
  duration = time.time() - start
  raw  = (NUM_CYCLES / duration)
  print("red frequency - ",raw,"Hz")

  #assignment of the measured color to
  #an RGB value between 0-255
  out_range = out_from - out_to
  in_range = redMax - redMin
  in_val = raw - redMin
  val = (float(in_val) / in_range) * out_range
  out = out_to + val
  if out < 0:
    out = 0
  elif out > 255:
    out = 255
  print(cN,"= ",out)

def loop():
  temp = 1

  #Determination of the output frequency scaling
  #S0/S1
  #LOW/LOW=/, LOW/HIGH=2%,
  #HIGH/LOW=20%, HIGH/HIGH=100%
  GPIO.output(s0,GPIO.HIGH)
  GPIO.output(s1,GPIO.HIGH)

  while(1):

##################################################################
################################
    time.sleep(0.3)
    print("-----------------------------------------------")
    time.sleep(0.3)
```

```
    print("Measured frequencys of Red, Green and Blue in Hz")

    #Determination of the photodiode type
    #S2/S3
    #LOW/LOW=RED, LOW/HIGH=BLUE,
    #HIGH/HIGH=GREEN, HIGH/LOW=CLEAR

    GPIO.output(s2,GPIO.LOW)
    GPIO.output(s3,GPIO.LOW)
    time.sleep(0.3)
    redraw = 0
    out_val1 = 0
    calc(redraw, "R", out_val1)

#################################################################
################################

    GPIO.output(s2,GPIO.HIGH)
    GPIO.output(s3,GPIO.HIGH)
    time.sleep(0.3)
    greenraw = 0
    out_val2 = 0
    calc(greenraw, "G", out_val2)

#################################################################
################################

    GPIO.output(s2,GPIO.LOW)
    GPIO.output(s3,GPIO.HIGH)
    time.sleep(0.3)
    blueraw = 0
    out_val3 = 0
    calc(blueraw, "B", out_val3)

#################################################################
################################

    red = out_val1
    green = out_val2
    blue = out_val3

    time.sleep(0.3)
    print("-------------------------------------------------------
---")
    time.sleep(0.3)

    #Determination of the measured color by comparison
    #with the values of the other measured colors

    if red>green and blue:
      print("Red detected")
      temp=1
    elif green>red and blue:
      print("Green detected")
      temp=1
    elif blue>red and green:
      print("Blue detected")
      temp=1
```

```
    elif blue>red and green>red:
      print("Turquoise detected")
      temp=1
    elif blue>green and red>green:
      print("Purple detected")
      temp=1
    elif temp==1:
      print("place a object infront of the Sensor")
      temp=0

    time.sleep(0.3)

def endprogram():
    GPIO.cleanup()

if __name__=='__main__':

    setup()

    try:
        loop()

    except KeyboardInterrupt:
        endprogram
```

Finally, we save and exit the file with **Ctrl+O ENTER Ctrl+X**

You then execute the code by entering

```
sudo python3 SEN-Color.py
```

Then to terminate the code use **CTRL+C.**

4. CALIBRATING THE SENSOR

When calibrating, it is important to ensure that the sensor is already in the lighting conditions in which it will later operate. It is therefore recommended to mount the sensor in its final working position before calibration. Since the sensor can also react sensitively to stray light, additional stray light protection can improve the measurement results if necessary.

4.1 CALIBRATING THE SENSOR ON THE ARDUINO/RASPBERRYPI

For the most accurate calibration possible, the color sensor should have a distance of approx. 4-5 cm to the measurement object during the calibration measurement. In addition, it is recommended to perform the calibration exclusively with the primary colors (red, green and blue). For this purpose, you can download and print our RGB calibration sheet.

During calibration, the minimum and maximum values are determined for each primary color and entered in the code.

```
#define S1 6
#define S2 7
#define S3 8
#define sensorOut 4

/*Calibration values (must be
  updated each use)*/
int redMin = 0;
int redMax = 1;
int greenMin = 0;
int greenMax = 1;
int blueMin = 0;
int blueMax = 1;
```

*Arduino code snippet

```
def loop():
  temp = 1
  #Calibration values (must be
  #updated before each use)
  #SEE INSTRUCTIONS
  redMin = 0
  redMax = 1
  greenMin = 0
  greenMax = 1
  blueMin = 0
  blueMax = 1
```

*Raspberry Pi code snippet

Now execute the code from 2.1 or 3.1. To determine the minimum measured values, measurements are performed without a target. Now take the measured values of the individual channels for the minimum calibration values.

Tip: To achieve an even better calibration you can also use the average of several measurements.

To determine the maximum calibration values, measurements are now performed with the basic colors as the measurement object. Here we recommend our RGB calibration sheet.

Hold the sensor one after the other in front of the three basic colors and adopt the values measured here for the maximum calibration values.

Tip: Again, you can improve the calibration by using the average of several measurements.

After the minimum and maximum calibration values have been entered into the original code, the code can be transferred again. The calibration is now complete and the sensor is ready for use.

5. ADDITIONAL INFORMATION

Our information and take-back obligations according to the Electrical and Electronic Equipment Act (ElektroG)

Symbol on electrical and electronic equipment:

This crossed-out dustbin means that electrical and electronic appliances do not belong in the household waste. You must return the old appliances to a collection point.
Before handing over waste batteries and accumulators that are not enclosed by waste equipment must be separated from it.

Return options:
As an end user, you can return your old device (which essentially fulfils the same function as the new device purchased from us) free of charge for disposal when you purchase a new device.
Small appliances with no external dimensions greater than 25 cm can be disposed of in normal household quantities independently of the purchase of a new appliance.

Possibility of return at our company location during opening hours:
SIMAC Electronics GmbH, Pascalstr. 8, D-47506 Neukirchen-Vluyn, Germany

Possibility of return in your area:
We will send you a parcel stamp with which you can return the device to us free of charge. Please contact us by email at Service@joy-it.net or by telephone.

Information on packaging:
If you do not have suitable packaging material or do not wish to use your own, please contact us and we will send you suitable packaging.

6. SUPPORT

If there are still any issues pending or problems arising after your purchase, we will support you by e-mail, telephone and with our ticket support system.

Email: **service@joy-it.net**

Ticket system: http://support.joy-it.net

Telephone: +49 (0)2845 98469-66 (10-17 o‘clock)

For further information please visit our website:

www.joy-it.net

Published: 18.08.2021