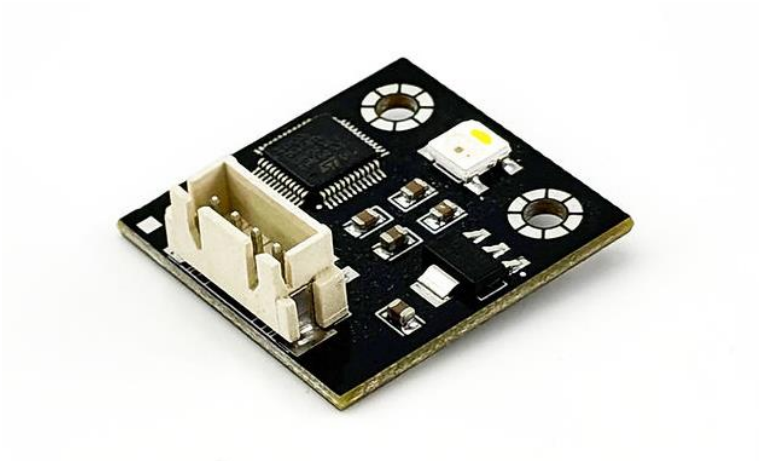
Color Sensor module



Introduction

The TCS34725 RGB Color Sensor Module is a complete optical color detection solution built around the TAOS TCS34725 IC. Featuring an integrated IR-blocking filter, an onboard RGBW LED, and a neutral white LED, the module measures reflected ambient or target light across Red, Green, Blue, and Clear (unfiltered) channels to deliver accurate digital RGB color values.

Features

Embedded Controller: Onboard STM32L051 ARM Cortex-M0+ microcontroller for real-time sensor sampling and color processing.

Optical Sensing: Integrated TAOS TCS34725 IC with on-chip IR-blocking filter to minimize infrared noise.

Visual Feedback: Top-mounted RGBW LED provides direct visual output matching detected colors.

Controlled Illumination: Bottom-mounted white LED provides consistent light supplementation for target objects

Multi-Format Processing: Real-time output of full-scale RGB, HSL, raw RGBC, and discrete color index parameters.

Connector Interface: Standard XH2.54 4-pin polarized I²C connector.

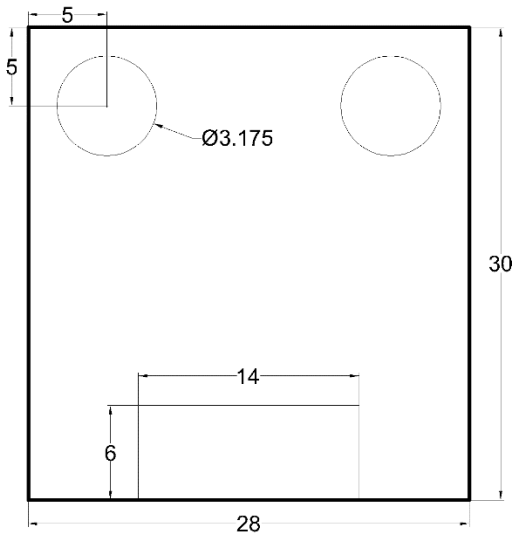
Printed on 05 August 2026

Features

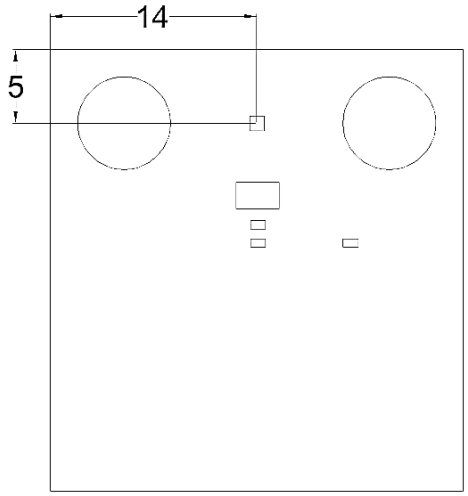
Power supply voltage : 4.5 ~ 5.5V DC
Operating current : ~ 6.2μA
Operating temperature : -40 ~ 85°C
Operating Humidity : <90%
Weight : 4.5 g
Interface : I²C Standard-mode 7-bits address 0x11
Size : 30 x 28 x 16mm
Detect distance : 1.5 – 7 cm (recommend 2 cm)

Dimensions (unit: mm)

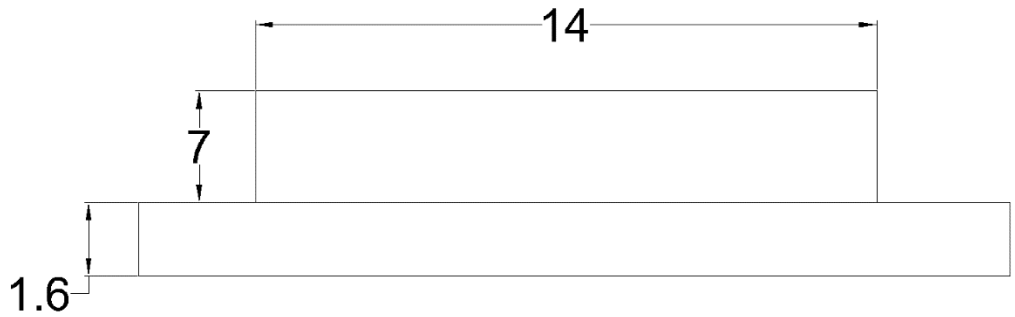
Top view:



Bottom view:



Side view:



I²C

This device, functioning as a slave, will be connected to the master device via I²C, under master control through a serial interface.

This device is compliant with I²C-Bus Specification. As an I²C compatible device, this device has a 7-bit serial address and supports I²C protocols. This device supports standard mode 100kHz.

The default I²C address is 0x11
If other I²C address options are required, please contact factory.

I²C Address Map:

Address	Data (unsigned integer)				Access
0x01	[7:0] COLOR_IDX				Read only
0x02	[127:96] BLUE_RAW	[95:64] GREEN_RAW	[63:32] RED_RAW	[31:0] CLEAR_RAW	Read only
0x03	[31:24] LIGHTNESS	[23:16] SATURATION	[15:0] HUE		Read only
0x04	Turn on the white led				Write only
0x05	Turn off the white led				Write only
0x06	Turn on RGBW led (show the color detected)				Write only
0x07	Turn off RGBW led (not show the color detected)				Write only
0x08	[23:16] BLUE	[15:8] GREEN	[7:0] RED		Read only

I²C

Address Description:

1. Adress 0x01 – Get color index

   COLOR_IDX

   Type: Read Only

   Description:

   It will return a color index range from 0 to 7

Color	Index
Black	0
White	1
Grey	2
Red	3
Green	4
Blue	5
Yellow	6
Cyan	7

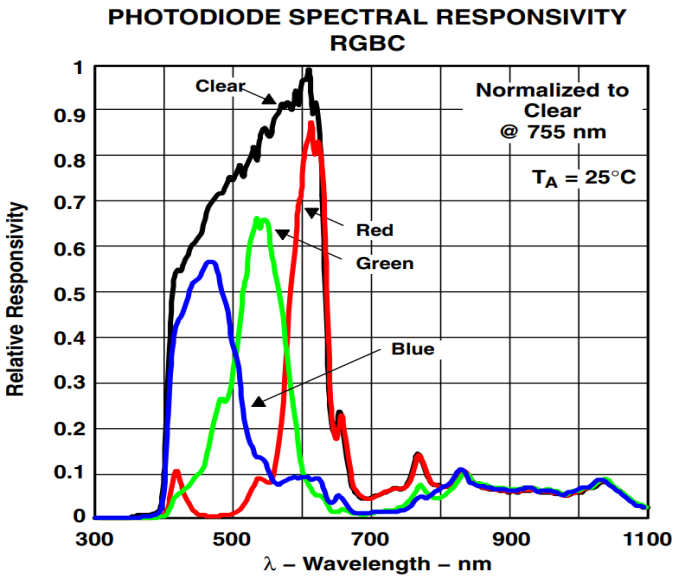
2. Adress 0x02 – Get RGBC value

   BLUE_RAW, GREEN_RAW, RED_RAW, CLEAR_RAW

   Type: Read Only

   Description:

   It will return the raw photodiode spectral responsivity data from the sensor IC. The output data of each channel saturates at 0 and 65536.

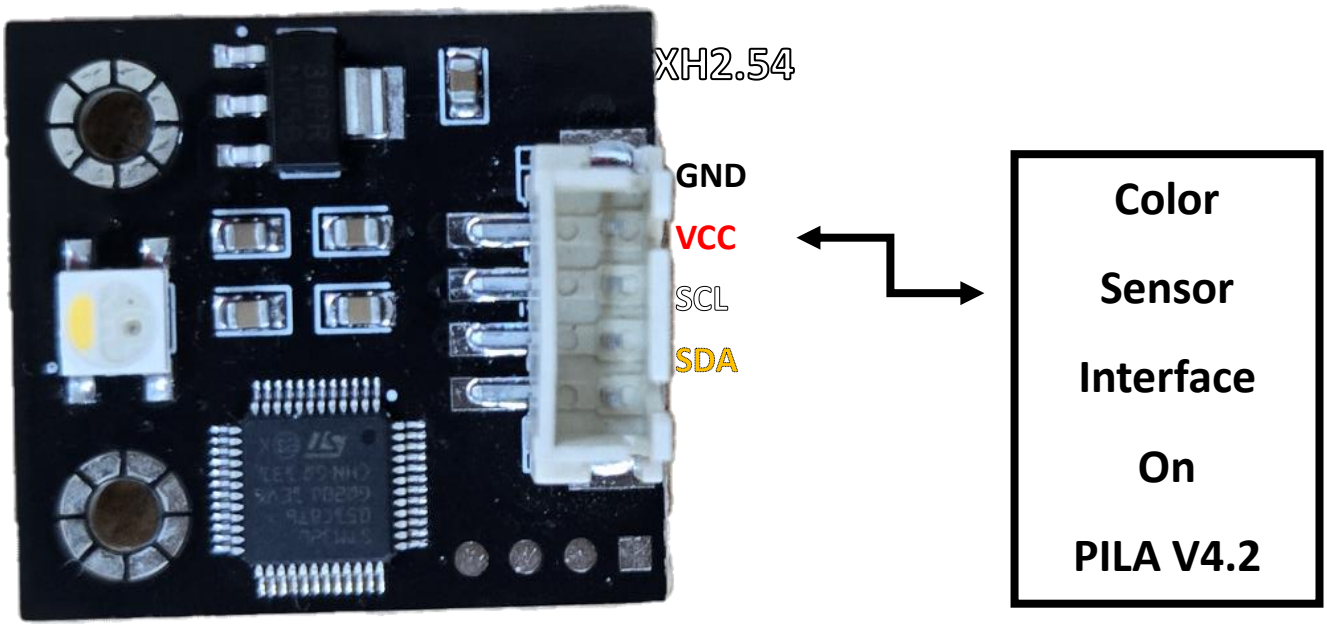


I²C

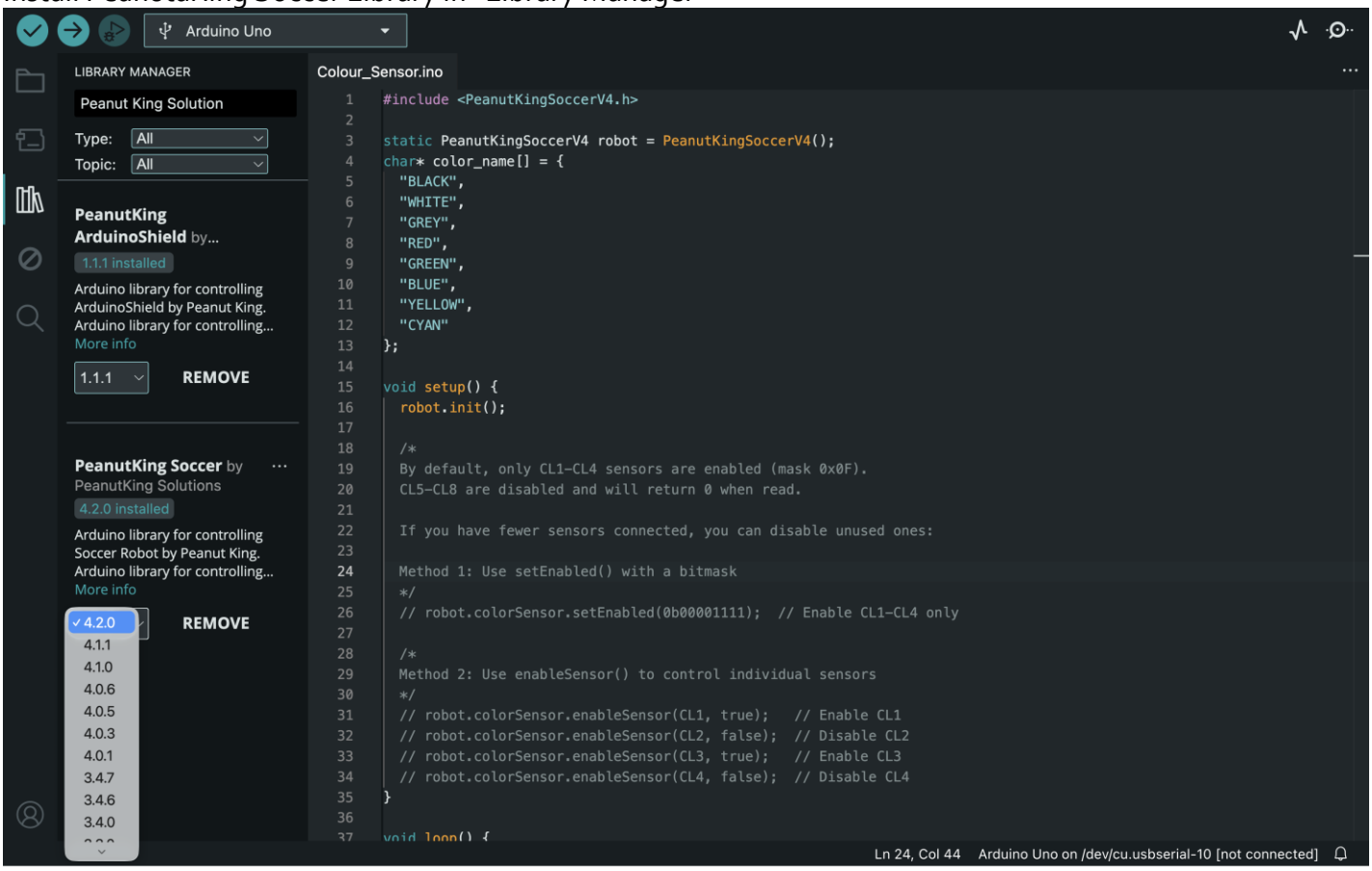
3. Adress 0x03 – Get HSL value
   LIGHTNESS, SATURATION, HUE
   Type: Read Only
   Description:
   It will return the calculated HSL value.
   The output data range of Hue is from 0~360.
   The output data range of Saturation is from 0~100.
   The output data range of Lightness is from 0~100.
4. Adress 0x04 – Turn on the bottom white led
   Description:
   By accessing this address, the board will turn on its white LED at the bottom.
5. Adress 0x05 – Turn off the bottom white led
   Description:
   By accessing this address, the board will turn off its white LED at the bottom.
6. Adress 0x06 – Turn on the top RGBW led
   Description:
   By accessing this address, the board will turn on its RGBW LED on the top. It will show the result of the color detected from address 0x01.
7. Adress 0x07 – Turn off the top RGBW led
   Description:
   By accessing this address, the board will turn on its RGBW LED on the top.
8. Register 0x08 – Get the RGB value
   BLUE, GREEN, RED
   Type: Read Only
   Description:
   It will return the calculated RGB value.
   The output data range of Red channel is from 0~255.
   The output data range of Green channel is from 0~255.

The output data range of Blue channel is from 0~255.

Application

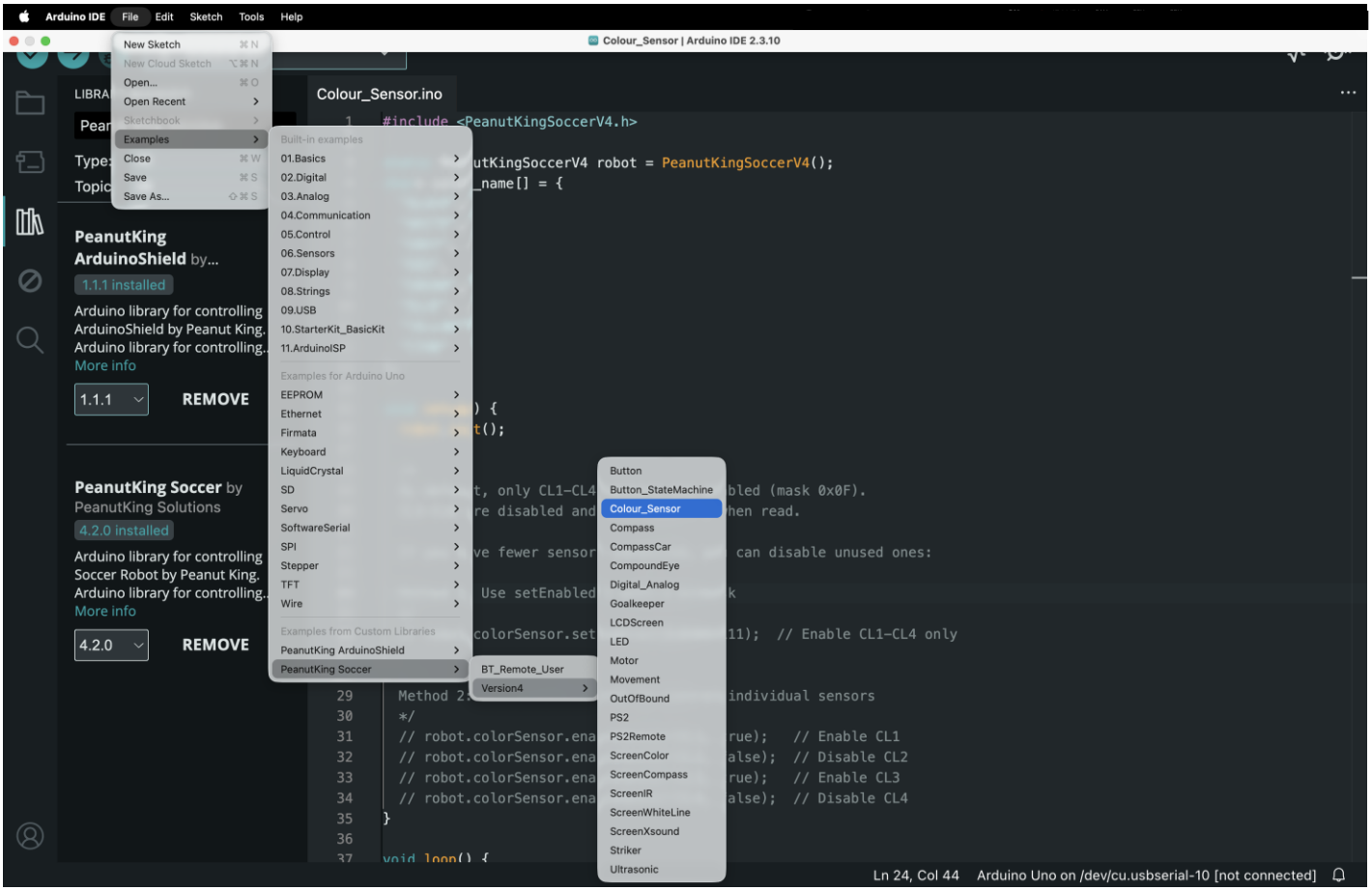


Install PeanutaKing Soccer Library in “Library Manager”



Application

Open the sample code in the following path
FIle→ Examples → PeanutKing Soccer → Version 4 → Color_Sensor



Application

```
#include <PeanutKingSoccerV4.h>

static PeanutKingSoccerV4 robot = PeanutKingSoccerV4();
char* color_name[] = {
 "BLACK",
 "WHITE",
 "GREY",
 "RED",
 "GREEN",
 "BLUE",
 "YELLOW",
 "CYAN"
};

void setup() {
 robot.init();

 /*
 By default, only CL1-CL4 sensors are enabled (mask 0x0F).
 CL5-CL8 are disabled and will return 0 when read.

 If you have fewer sensors connected, you can disable unused ones:

 Method 1: Use setEnabled() with a bitmask
 */
 // robot.colorSensor.setEnabled(0b00001111);  // Enable CL1-CL4 only

 /*
 Method 2: Use enableSensor() to control individual sensors
 */
 // robot.colorSensor.enableSensor(CL1, true);   // Enable CL1
 // robot.colorSensor.enableSensor(CL2, false);  // Disable CL2
 // robot.colorSensor.enableSensor(CL3, true);   // Enable CL3
 // robot.colorSensor.enableSensor(CL4, false);  // Disable CL4
}

void loop() {
 // Read color index from CL1 sensor
 CLR_SENSOR_ID colorSensor = CL1;

 // Read color index from CL1 sensor
 uint8_t colorIdx = robot.colorSensor.readColor(colorSensor);

 // Alternatively, use the wrapper function for compatibility:
 // uint8_t colorIdx = robot.getColorSensor(colorSensor);

 // Read RGB values from CL1 sensor
```

```
rgb_t rgb = robot.colorSensor.readRGB(colorSensor);

// Alternatively, use the wrapper function:
// rgb_t rgb = robot.getColorSensorRGB(colorSensor);

// Read HSL values from CL1 sensor
hsl_t hsl = robot.colorSensor.readHSL(colorSensor);

// Alternatively, use the wrapper function:
// hsl_t hsl = robot.getColorSensorHSL(colorSensor);

// Read RGBC raw values from CL1 sensor
rgbc_t rgbc = robot.colorSensor.readRGBRaw(colorSensor);

// Or you can read all sensors at once using dataFetch()
// robot.dataFetch();
// uint8_t colorIdx = robot.colorRGB[colorSensor];
// hsl_t hsl = robot.colorHSL[colorSensor];

// Print the readings to the Serial Monitor
Serial.print("CL1 - Color: ");
if (colorIdx <= 7) {
  Serial.print(color_name[colorIdx]);
} else {
  Serial.print("UNKNOWN");
}

// Print RGB values
Serial.print(", RGB: ");
Serial.print(rgb.r);
Serial.print(", ");
Serial.print(rgb.g);
Serial.print(", ");
Serial.print(rgb.b);
// Print HSL values
Serial.print(", HSL: ");
Serial.print(hsl.h);
Serial.print(", ");
Serial.print(hsl.s);
Serial.print(", ");
Serial.print(hsl.l);
// Print RGBC raw values
Serial.print(", RGBC: ");
Serial.print(rgbc.r);
Serial.print(", ");
Serial.print(rgbc.g);
Serial.print(", ");
Serial.print(rgbc.b);
Serial.print(", ");
```

```
  Serial.print(rgbc.c);
  // Print white line detection
  Serial.print(", White: ");
  Serial.print(robot.colorSensor.isWhiteLine(colorSensor) ? "YES" : "NO");
  Serial.println();

  delay(100);
}
```

Application

Install the extension in Makecode for micro:bit with the following url.

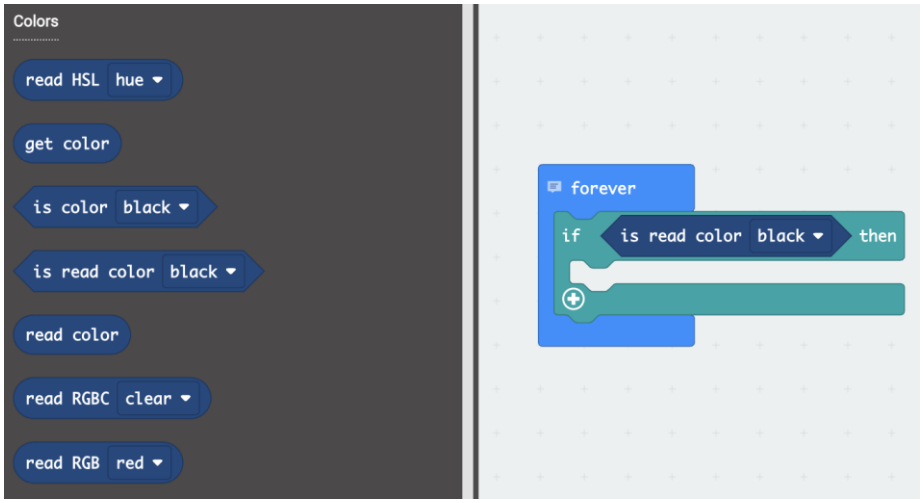
https://github.com/peanut-king-solution/pxt-pks-shield

In "Edu Kit", pull the following blocks out

"get color": Identifies and returns the discrete color classification or index of. the target object.

"read RGB": Fetches raw or processed Red, Green, and Blue (RGB) intensity. channel values.

Boolean Color Evaluation: Supports conditional control logic using the "is color" function to verify specific. target colors (e.g., Black, Green, Red) directly within application if-else structures.



Important Information and Disclaimer

The information in this statement reflects Peanut King Solution Limited's knowledge and belief as of the date it is provided. Peanut King Solution Limited relies on information from third parties and does not guarantee its accuracy. Efforts are ongoing to better integrate third-party information. Peanut King Solution Limited has taken and continues to take reasonable steps to provide accurate and representative information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. Certain information is considered proprietary by Peanut King Solution Limited and its suppliers, so CAS numbers and other limited information may not be available for release.

NOTICE

Peanut King Solution Limited reserves the right to make changes to the products in this document to improve performance or for any other reason, or to discontinue them without notice. Customers should contact Peanut King Solution Limited for the latest product information before placing orders or designing Peanut King Solution Limited products into systems. Peanut King Solution Limited assumes no responsibility for the use of any products or circuits described in this document or customer product design, does not convey any license under any patent or other right, and does not guarantee that the circuits are free of patent infringement. Peanut King Solution Limited makes no claim regarding the suitability of its products for any particular purpose and assumes no liability arising from the use of any product or circuit, specifically disclaiming any and all liability, including consequential or incidental damages.

PEANUT KING SOLUTION LIMITED PRODUCTS ARE NOT DESIGNED OR INTENDED FOR USE IN CRITICAL APPLICATIONS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY OR DEATH. THE USE OF PEANUT KING SOLUTION LIMITED PRODUCTS IN LIFE SUPPORT SYSTEMS IS NOT AUTHORIZED.