HANOI UNIVERSITY OF SCIENE AND TECHNOLOGY

SCHOOL OF MECHANICAL ENGINEERING

FACULTY OF MECHATRONIC ENGINEERING

-----*-----

SENSORS AND SIGNAL PROCESSING

SUBJECT

UTILIZING COLOR SENSORS

TEACHER: Ph.D Hoàng Sỹ Tuấn

Ph.D Trương Công Tuấn

STUDENT : Nguyễn Tiến Đức - 20215692

Triệu Đình Hoài - 20215707

CLASS : 144028

Contents :

SUMMARY OF PROJECT

1.Outline the objectives, significance, and requirements of project.

The course aims to provide for students basic knowledge and fundamental of sensors, along with signal processing circuits, post-sensor signal processing methods and sensor pairing with processing stages. next in measurement, control and mechatronic systems. Common types of sensors are classified and presented in groups with similar characteristics in circuit and signal processing methods after the sensor, in order to help students easily grasp and synthesize knowledge. The content of the module is the basis for students to carry out projects and other specialized subjects.

2.Summary of Achievements:

Students will gain numerous benefits when using color sensors in the Sensors and Signal Processing course. Specifically, students will: Understanding the principles of operation and applications of color sensors.

Developing programming skills and signal processing abilities.

Examples of applications of color sensors:

- Color detection.

- Object classification.

- Motion tracking.

3.The tasks

NO	FULL NAME	Tasks	NOTES
1	Nguyễn Tiến Đức	Analyze the operating principle, draw the circuit, writing report ,calculation …	Leader
2	Triệu Đình Hoài	Programming , calculation, experiment …	

PROBLEMS

Concept

How to classify product in the industry , quality control of the product?

We can use color sensors

In this project , we work with and operate the TCS 230 color sensor

1. TCS 230 sensor

Description

The TCS230 programmable color light-to-frequency converter combines configurable silicon photodiodes and a current-to-frequency converter on single monolithic CMOS integrated circuit. The output is a square wave (50% duty cycle) with frequency directly proportional to light intensity (irradiance). The full-scale output frequency can be scaled by one of three preset values via two control input pins. Digital inputs and digital output allow direct interface to a microcontroller or other logic circuitry. Output enable (OE) places the output in the high-impedance state for multiple-unit sharing of a microcontroller input line. The light-to-frequency converter reads an 8 x 8 array of photodiodes. Sixteen photodiodes have blue filters, 16 photodiodes have green filters, 16 photodiodes have red filters, and 16 photodiodes are clear with no filters. The four types (colors) of photodiodes are interdigitated to minimize the effect of non-uniformity of incident irradiance. All 16 photodiodes of the same color are connected in parallel and which type of photodiode the device uses during operation is pin-selectable. Photodiodes are 120 µm x 120 µm in size and are on 144-µm centers.

Features:

- High - Resolution Conversion of Light

- Intensity to Frequency

- Programmable Color and Full-Scale Output TFrequency

- Communicates Directly With a Microcontroller

- Single-Supply Operation (2.7 V to 5.5 V)

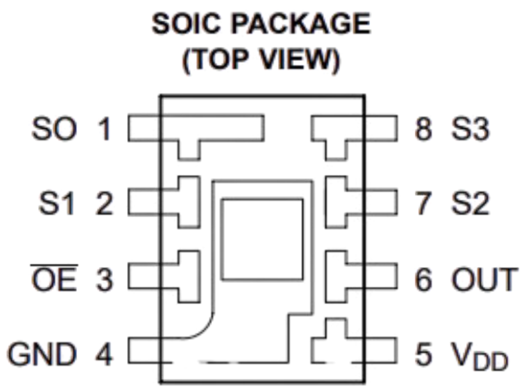
- Power Down Feature

- Nonlinearity Error Typically 0.2% at 50 kHz

- Stable 200 ppm/°C Temperature Coefficient

- Low - Profile Surface-Mount Package

Terminal



Terminal	No	I/O	Description
GND	4		Power supply ground. All voltages are referenced to GND.
OE	3	I	Enable for fo (active low).
OUT	6	0	Output frequency (fo).
SO, Sl	1, 2	I	Output frequency scaling selection inputs.
S2, S3	7, 8	I	Photodiode type selection inputs.
VDD	5		Supply voltage

The operating principle

The light-to-electric signal conversion process in the TCS230 sensor can be broken down into two main steps:

Step 1: Light Absorption

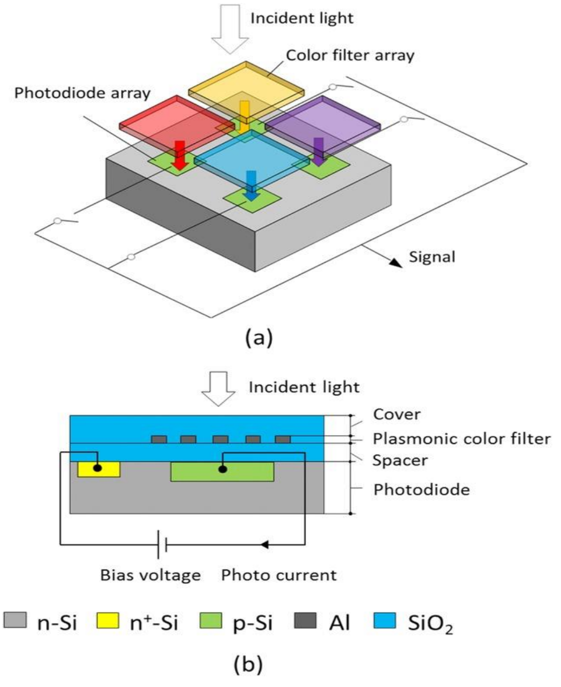
When light strikes the sensor, it gets absorbed by electrons within the material.

An absorbing electron receives energy from a photon of light, entering an excited state.

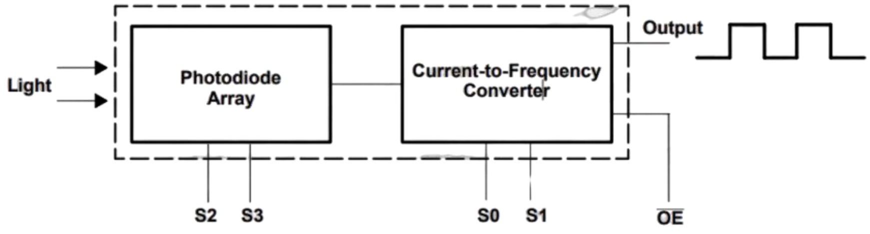
Step 2: Signal Generation

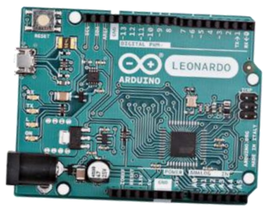
The excited electrons release their gained energy as an electric current.

This current is then converted into an electrical signal.



Function block diagram





Digital inputs and digital output allow direct interface to a microcontroller or other logic circuitry.

Output enable (/OE) places the output in the high-impedance state for multiple-unit sharing of a microcontroller input line.

The output is a square wave (50% duty cycle) with frequency (fO) directly proportional to light intensity (irradiance).

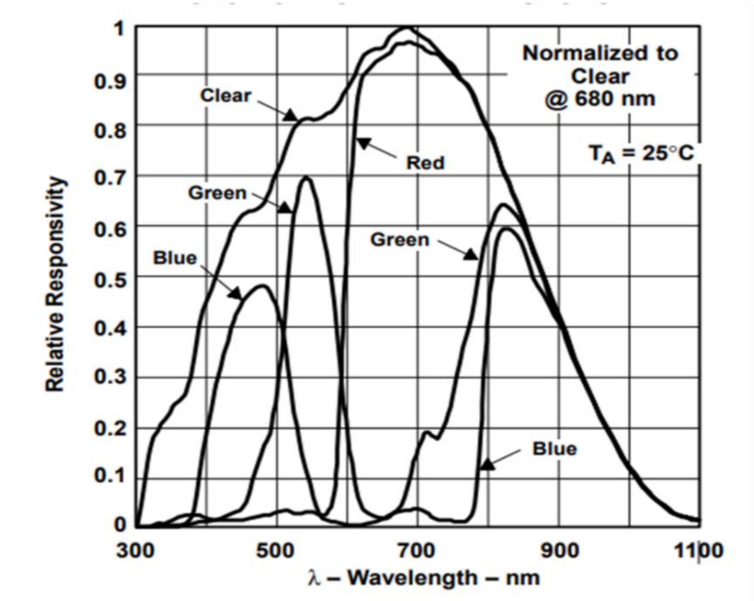
The full-scale output frequency can be scaled by one of three preset values and off, via two control input pins (S0, S1).

Four types of photodiodes - Red, Green, Blue and Clear (no filter) - are pin-selectable (S2, S3) to read the individual components of the color detected

Selectable options of output frequency scaling

S0	S1	OUTPUT FREQUENCY SCALING (fo)		S2	S3	PHOTODIODE TYPE
L	L	Power down		L	L	Red
L	H	2%		L	H	Blue
H	L	20%		H	L	Clear (no filter)
H	H	100%		H	H	Green

Photodiode spectral responsivity



The table above shows the spectral response of a photodiode. The spectral response is a graph that shows the intensity of light emitted or absorbed by a material at different wavelengths. In this case, the photodiode is stimulated by light with wavelengths from 300 to

1100 nanometers (nm).The x-axis of the graph is the wavelength of light, expressed in units of nanometers. The y-axis is the relative sensitivity of the photodiode, expressed as a number from 0 to 1. The number 1 corresponds to the highest sensitivity, while the number 0 corresponds to the lowest sensitivity.As you can see, the photodiode is most sensitive to light with a wavelength of 680 nm. This is the wavelength of red light. The photodiode is also sensitive to light with a wavelength of 500 nm, corresponding to green light. However, its sensitivity to blue and violet light is much lower.The spectral response of a photodiode can be used to identify the type of photodiode. For example, photodiodes with the highest sensitivity to red light are often used in applications such as color sensors.

Operating Characteristics at VDD = 5 V, TA = 25°C, S0 = H, S1 = H

PARAMETER	TEST CONDITIONS	CLEAR PHOTODIODE S2 = H, S3 = L			BLUE PHOTODIODE S2 = L, S3 = H			GREEN PHOTODIODE S2 = H, S3 = H			RED PHOTODIODE S2 = L, S3 = L			UNIT
		MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	
fO (Output frequency)	Ee = 47.2 mW/cm2, λp = 470 nm	16	20	24	11.2	16.4	21.6							kHz
	Ee = 40.4 mW/cm2, λp = 524 nm	16	20	24				8	13.6	19.2				kHz
	Ee = 34.6 mW/cm2, λp = 640 nm	16	20	24							14	19	24	kHz

f_D (Dark frequency)	$E_e = 0$	2 12	2 12	2 12	2 12	Hz
Irradiance R_e responsivity (Note 8)	λp = 470 nm	424	348	81	26	Hz/ (mW/ cm2)
	λp = 524 nm	495	163	337	35	
	λp = 565 nm	532	37	309	91	
	λp = 640 nm	578	31	29	550	
Saturation irradiance (Note 9)	λp = 470 nm	1410	1720			mW/ cm2
	λp = 524 nm	1210		1780		
	λp = 565 nm	1130		1940		
	λp = 640 nm	1040			1090	
Illuminance R_v responsivity (Note 10)	λp = 470 nm	565	464	108	35	Hz/lx
	λp = 524 nm	95	31	65	7	
	λp = 565 nm	89	6	52	15	
	λp = 640 nm	373	20	19	355	
Nonlinearity (Note 11)	f_O = 0 to 5 kHz	± 0.1%	± 0.1%	± 0.1%	± 0.1%	%F.S.
	f_O = 0 to 50 kHz	± 0.2%	± 0.2 %	± 0.2 %	± 0.2 %	%F.S.
	f_O = 0 to 500 kHz	± 0.5%	± 0.5%	± 0.5%	± 0.5%	%F.S.
Recovery from power down		100	100	100	100	ms
Response time to out- put enable (OE)		100	100	100	100	ns

NOTES:

8. Irradiance responsivity Re is characterized over the range from zero to 5 kHz.

9. Saturation irradiance = (full-scale frequency)/(irradiance responsivity).

10. Illuminance responsivity Rv is calculated from the irradiance responsivity by using the LED luminous efficacy values stated in notes

4, 5, and 6 and using 1 lx = 1 lm/m2.

11. Nonlinearity is defined as the deviation of fO from a straight line between zero and full scale, expressed as a percent of full scale

INTERPRETING SENSOR DATA

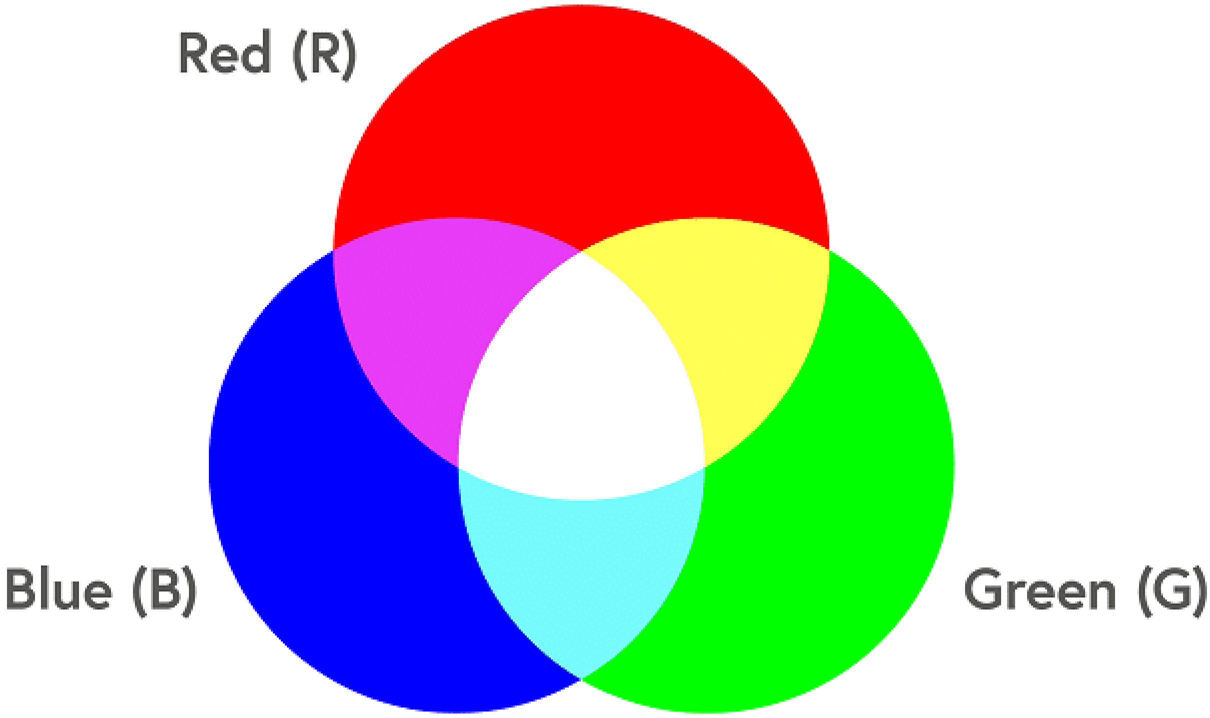
The output is a square wave (50% duty cycle) with frequency (fO) directly proportional to light intensity:

$$Fo=fd+(Re)(Ee)$$

where fo is the output frequency; fd is the output frequency for dark condition (when Ee = 0); Re is the device responsivity for a given wavelength of light in kHz/(mW/cm^2); Ee is the incident irradiance in mW/cm^2.

fd is an output frequency resulting from leakage currents. As shown in the equation above, this frequency represents a light-independent term in the total output frequency fo. At very low light levels (dark colors), this dark frequency can be a significant portion of fo. The dark frequency is also temperature dependent.

As fo is directly proportional to frequency, it is possible to map between the frequency and RGB color value (0-255 for each of R, G and B) using linear interpolation.



Two points on the RGB line are well determined – pure Black (RGB 0, 0, 0) and pure White (255, 255, 255). The values returned by the sensor can be read using easily obtainable color swatches:

A black color card gives us the dark condition constant fd. This is the origin (zero value) for the RGB straight line conversion.

A white color card gives us the extreme RGB point fw, also known as white balance. Knowing fd, this value can be used to scale all intermediate frequencies to a corresponding RGB value.

The proportional relationship is expressed by the standard straight line equation y = mx + b where y is the reading obtained (in our case fO)

x is the normalised RGB value

b is the value of y when x is 0 (in our case fD)

m is the slope, or proportionality constant, of the line (in our case $\frac{fw-fd}{255}$).

The resulting equation is

$$fo = fd + x\,\frac{fw-fd}{255}$$

or, rearranging to give us the desired RGB value

$$x = 255\frac{f_0 - fd}{(fw - fd)}$$

Hardware components

Number	Name	Quantity
1	ARDUINO LEONARDO	1
2	LCD 1602	1
3	MODULE I2C	1
4	JUMPER WIRES	11
5	TCS 230	1

1.ARDUINO LEONARDO

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing.

Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts a like

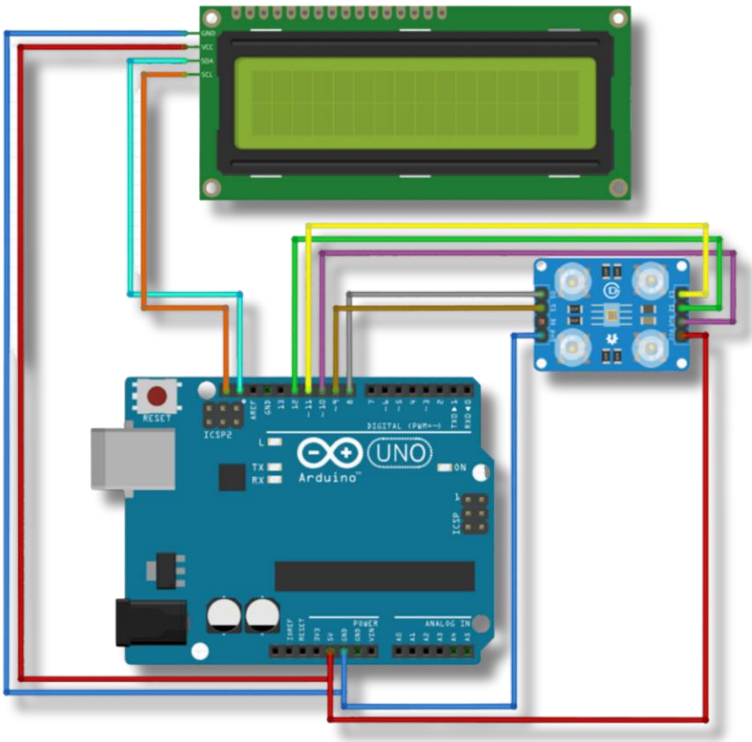
2.LCD 1602 + I2C

LCD modules are very commonly used in most embedded projects, the reason being its cheap price, availability and programmer friendly. Most of us would have come across these displays in our day to day life, either at PCO's or calculators. The appearance and the pinouts have already been visualized above now let us get a bit technical.

16×2 LCD is named so because; it has 16 Columns and 2 Rows. There are a lot of combinations available like, 8×1, 8×2, 10×2, 16×1, etc. but the most used one is the 16×2 LCD. So, it will have (16×2=32) 32 characters in total and each character will be made of 5×8 Pixel Dots. A Single character with all its Pixels is shown in the below picture.

Now, we know that each character has (5×8=40) 40 Pixels and for 32 Characters we will have (32×40) 1280 Pixels. Further, the LCD should also be instructed about the Position of the Pixels. Hence it will be a hectic task to handle everything with the help of MCU, hence an Interface IC like HD44780is used, which is mounted on the backside of the LCD Module itself. The function of this IC is to get the Commands and Data from the MCU and process them to display meaningful information onto our LCD Screen. You can learn how to interface an LCD using the above mentioned links. If you are an advanced programmer and would like to create your own library for interfacing your Microcontroller with this LCD module then you have to understand the HD44780 IC is working and commands which can be found its datasheet.

Schematic



Source code :

First we need to define the pins to which the sensor is connected and define a variable for reading the frequency. In the setup section we need to define the four control pins as outputs and the sensor output as an Arduino input. Here we also need to set the frequency-scaling, for this example I will set it to 20%, and start the serial communication for displaying the results in the Serial Monitor.

In the loop section, we will start with reading the red filtered photodiodes. For that purpose we will set the two control pins S2 and S3 to low logic level. Then using the "pulseIn()" function we will read the output frequency and put it into the variable "frequency". Using the Serial.print() function we will print the result on the serial monitor. The same procedure goes for the two other colors, we just need to adjust the control pins for the appropriate color.1

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
#define S0 4
#define S1 5
#define S2 6
```

```
#define S3 7
#define sensorOut 8
int rLED = 9;
int gLED = 10;
int bLED = 11;
int PW = 0;
void setup()
{
   lcd.init();
  //Bật đèn nền
  lcd.backlight();
  // Xóa màn hình LCD
  pinMode(S0, OUTPUT);
  pinMode(S1, OUTPUT);
  pinMode(S2, OUTPUT);
  pinMode(S3, OUTPUT);
  pinMode(sensorOut, INPUT);
  // Setting frequency scaling to 20%
  digitalWrite(S0, HIGH);
  digitalWrite(S1, LOW);
   Serial.begin(9600);
}
void loop(){
  String colour = readcolour();
   lcd.clear();
  lcd.setCursor(6, 1);
  lcd.print(    colour);
  }
String readcolour()
{
  // Setting red photodiodes to be read
  digitalWrite(S2, LOW);
  digitalWrite(S3, LOW);
  PW = pulseIn(sensorOut, LOW);  // Reading the output pulse width
  int R = PW;
  // Printing the value on the serial monitor
   lcd.setCursor(0, 0);
  lcd.print("R=");         //printing name
  lcd.print(PW);              //printing RED color pulse width
  lcd.print(" ");
  delay(500);
  // Setting Green photodiodes to be read
  digitalWrite(S2, HIGH);
  digitalWrite(S3, HIGH);
  PW = pulseIn(sensorOut, LOW);  // Reading the output pulse width
  int G = PW;
  lcd.setCursor(6,0);
  // Printing the value on the serial monitor
  lcd.print("G=");          //printing name
  lcd.print(PW);              //printing GREEN color pulse width
```

```
lcd.print(" ");
delay(500);
// Setting Blue photodiodes to be read
digitalWrite(S2, LOW);
digitalWrite(S3, HIGH);
PW = pulseIn(sensorOut, LOW);  // Reading the output pulse width
int B = PW;
// Printing the value on the serial monitor
lcd.setCursor(0, 1);
lcd.print("B=");          //printing name
lcd.print(PW);              //printing BLUE color pulse width
delay(500);
String colour;
if(R <= 30 && G <= 35 && B <= 25)
{
  colour = "   White";
}
else if (R > 200  && G > 200 && B > 200 )
{
  colour = "   Black";
}
else if(R >= 50 && R <= 150 && G >= 30 && G <= 90 && B >= 50 && B <= 110)
{
  colour = "   Green";
}
else if ( R >= 20 && R <= 50 && G >= 80 && G <= 140 && B >= 60 && B <= 110)
{
  colour = "   Red";
}
else if (R >= 26 && R <= 50 && G >= 35 && G <= 65 && B >= 25 && B <= 50 )
{
  colour = "   Pink";
}
else if (R >= 50 && R <= 90 && G >= 30 && G <= 50 && B >= 20 && B <=40)
{
  colour = "   Blue";
}
else if (R >= 25 && R <= 40 && G >= 30 && G <= 45 && B >= 50 && B <= 70)
{
  colour = "   Yellow";
}
else if (R >= 25 && R <= 40 && G >= 50 && G <= 70 && B >= 60 && B <= 90)
{
  colour = "   Orange";
}
else if (R >= 60 && R <= 80 && G >= 90 && G <= 110 && B >= 90 && B <= 110)
{
  colour = "   Brown";
}
else
```

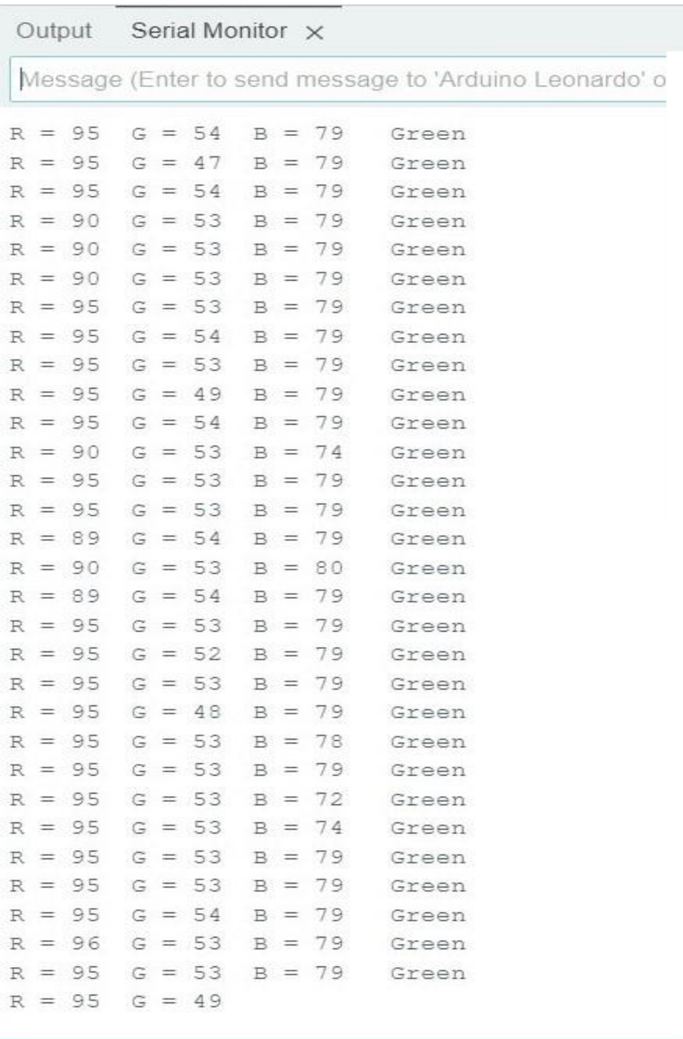
```
  {
    colour = "   Unknown";
  }
   return colour;
  }
```

RESULT

Now if we run the Serial Monitor we will start getting some values. These values depend on the selected frequency-scaling, as well as from the surrounding lighting

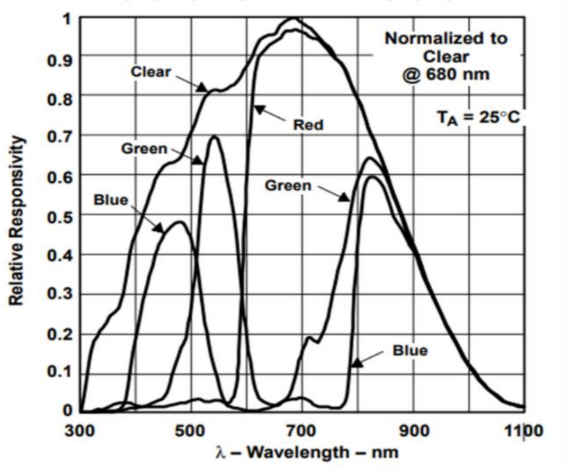
Note here that three values differ due to the different sensitivity of each photodiode type, as seen from the photodiode spectral responsivity diagram from the datasheet of the sensor.

Nevertheless, now let's see how the values react when we will bring different colors in front of the sensor. So for example, if we bring red color, the initial value will drop down, in my case from around 49 to around 54.



```
Output    Serial Monitor ×
Message (Enter to send message to 'Arduino Leonardo' o
R = 95   G = 54   B = 79    Green
R = 95   G = 47   B = 79    Green
R = 95   G = 54   B = 79    Green
R = 90   G = 53   B = 79    Green
R = 90   G = 53   B = 79    Green
R = 90   G = 53   B = 79    Green
R = 95   G = 53   B = 79    Green
R = 95   G = 54   B = 79    Green
R = 95   G = 53   B = 79    Green
R = 95   G = 49   B = 79    Green
R = 95   G = 54   B = 79    Green
R = 90   G = 53   B = 74    Green
R = 95   G = 53   B = 79    Green
R = 95   G = 53   B = 79    Green
R = 89   G = 54   B = 79    Green
R = 90   G = 53   B = 80    Green
R = 89   G = 54   B = 79    Green
R = 95   G = 53   B = 79    Green
R = 95   G = 52   B = 79    Green
R = 95   G = 53   B = 79    Green
R = 95   G = 48   B = 79    Green
R = 95   G = 53   B = 78    Green
R = 95   G = 53   B = 79    Green
R = 95   G = 53   B = 72    Green
R = 95   G = 53   B = 74    Green
R = 95   G = 53   B = 79    Green
R = 95   G = 53   B = 79    Green
R = 95   G = 54   B = 79    Green
R = 96   G = 53   B = 79    Green
R = 95   G = 53   B = 79    Green
R = 95   G = 49
```

The result displayed on the Arduino



IDE's

Serial Monitor shows three basic color

parameters:

R (Red), G (Green), and B (Blue).

By combining these three values,

we obtain the desired color representation

Calibrate :

At the data collection step, you need to perform the following steps:

1. Prepare objects with basic colors (red, green, blue, white, black) or the colors you want to calibrate accurately.

2. Place these objects in front of the TCS230 sensor in a stable lighting condition.

3. Use the current code to measure the pulse width (PW) value of the sensor for each reference color.

4. Note the PW value of each reference color.

For example, to collect data for the color green, you can perform the following steps:

1. Prepare a green object.

2. Place this object in front of the TCS230 sensor in a stable lighting condition.

3. Run the code and record the PW value of the sensor.

4. Repeat steps 2-3 for other green objects.

After collecting data for all reference colors, you can use this data to determine the color threshold for each color.

Here are some tips for collecting data:

* Make sure that the colored objects are placed in front of the sensor in the same position. This will help ensure that the PW measurements are accurate.

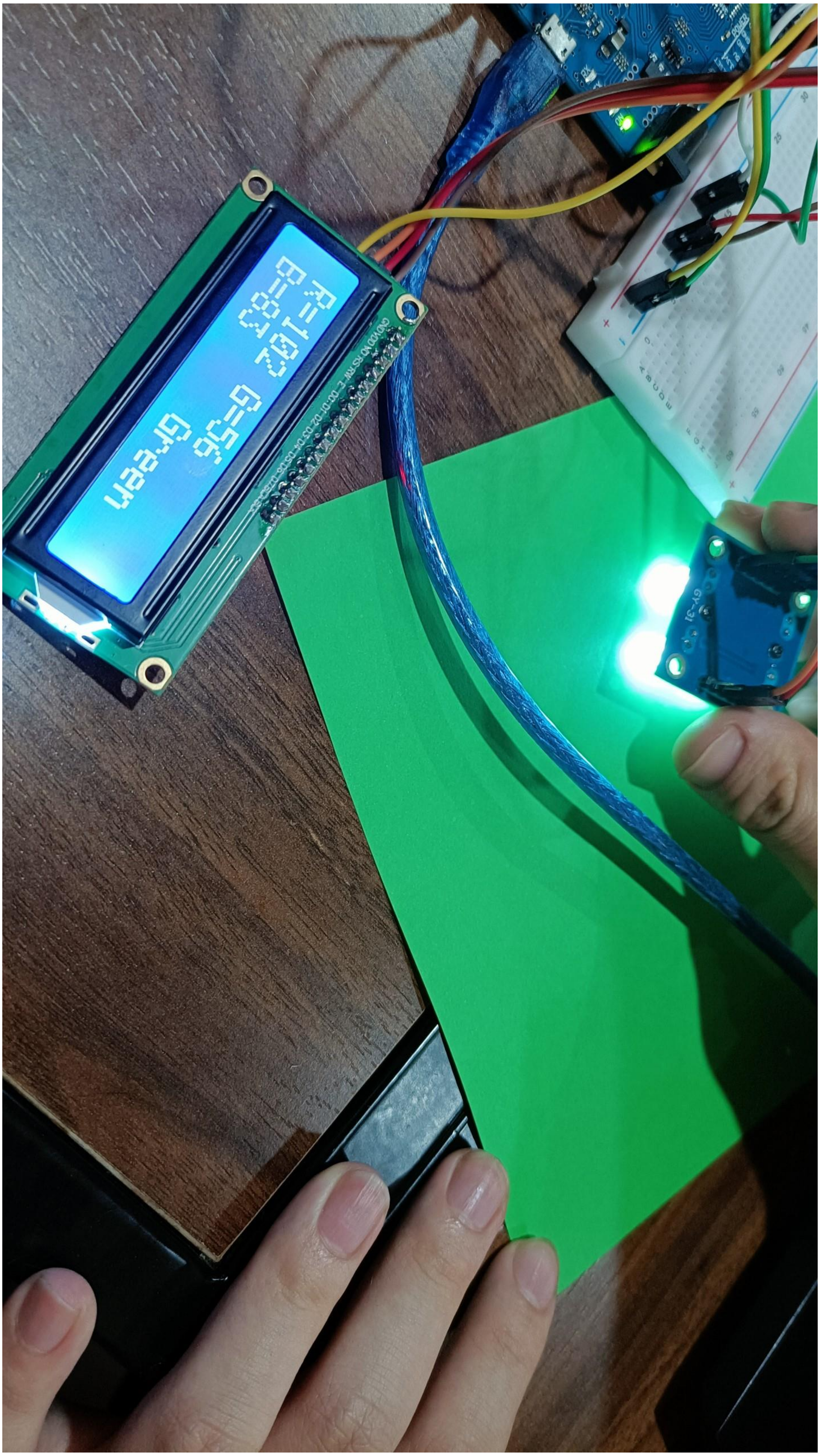
* If possible, try to collect data in a variety of lighting conditions. This will help you determine more accurate color thresholds for different lighting environments.

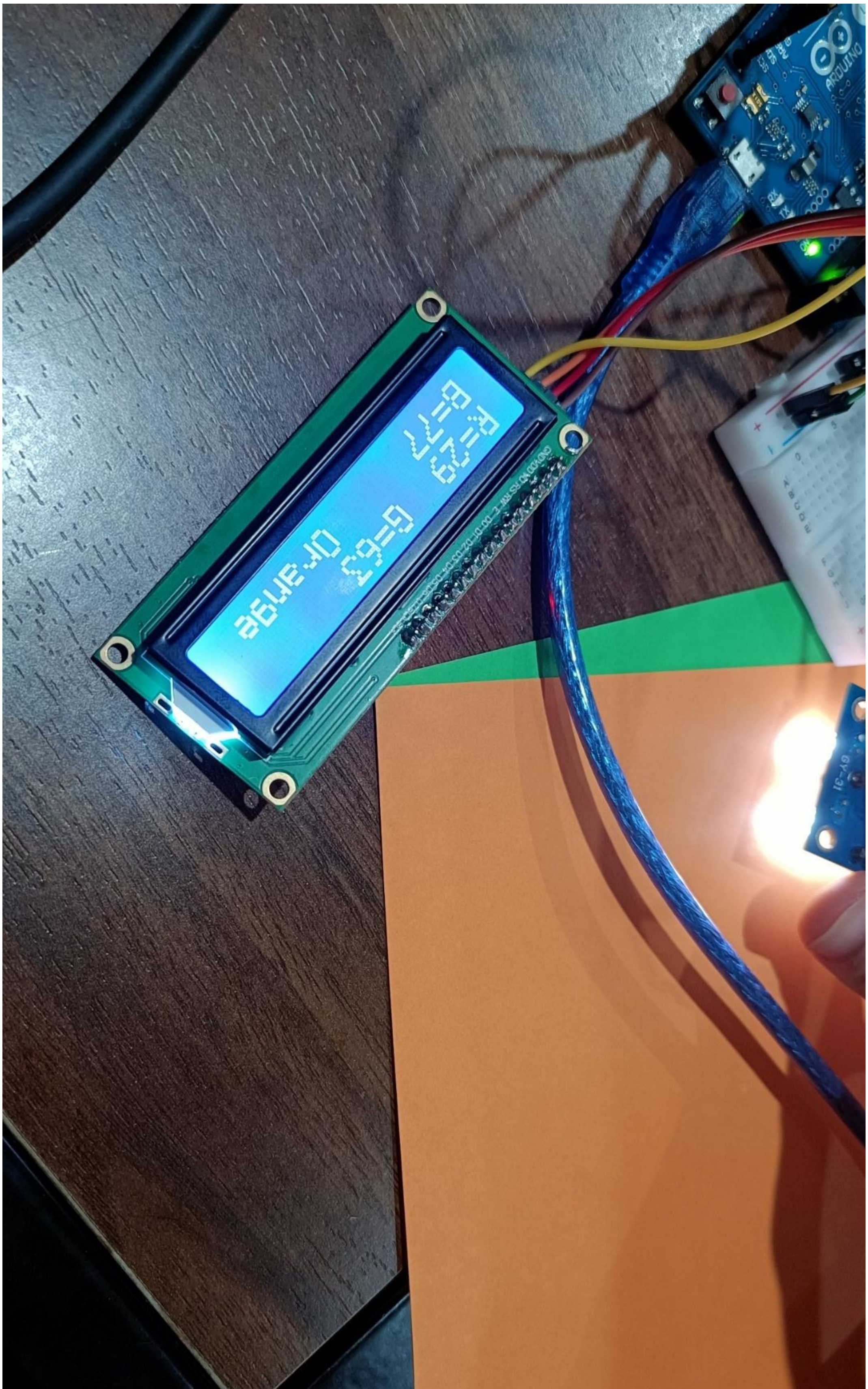
* Check the collected data to ensure that they are accurate. If you see any abnormal PW values, remove them from your dataset.

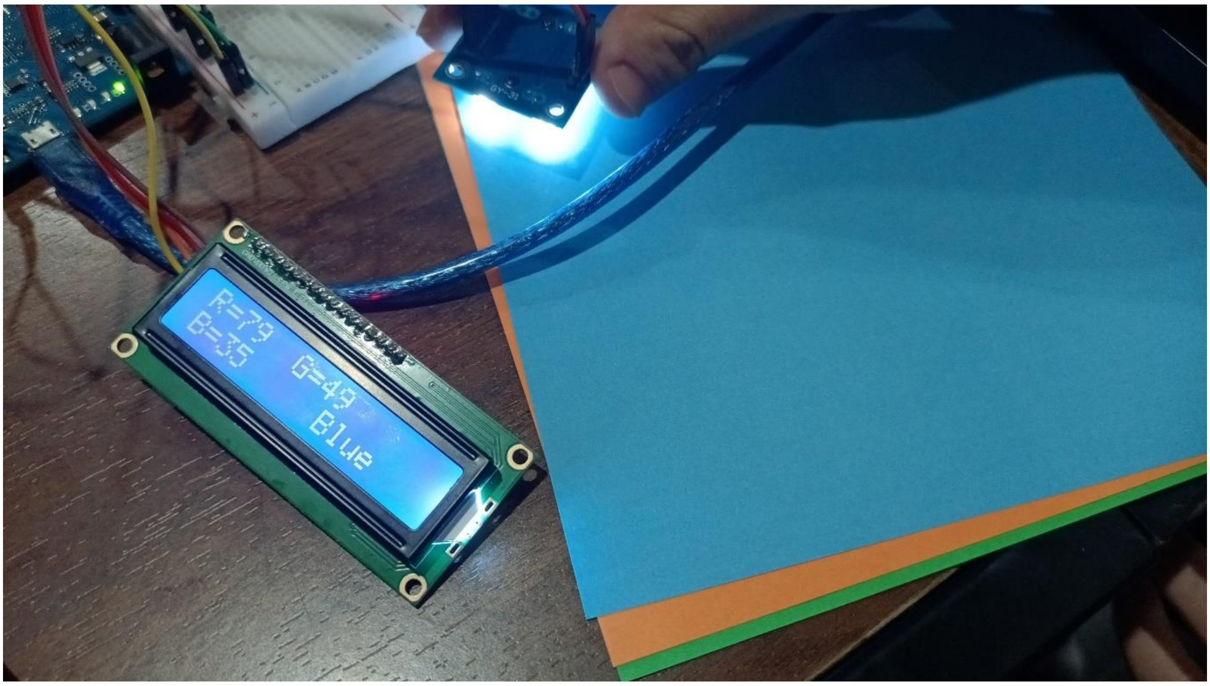
* You can use a spreadsheet or graphing software to help you analyze the data. This will help you identify trends and patterns in the data.

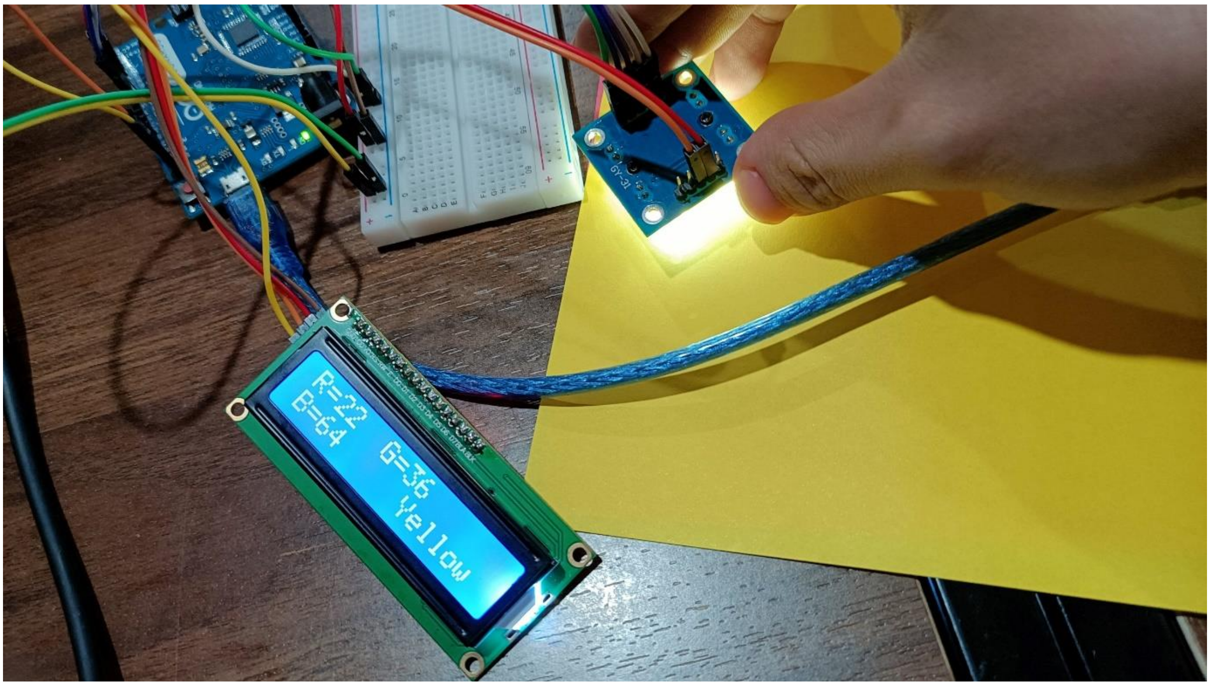
* You can use a statistical analysis tool to help you determine the best thresholds for each color.

R	G	B	Color
96	53	77	Green
91	47	83	Green
96	53	83	Green
91	54	83	Green
96	47	78	Green
97	53	83	Green
96	53	79	Green
97	53	83	Green
96	53	83	Green
97	53	83	Green


R=102 G=56
B=83 Green
GY-31


R=29
G=63
B=77
Orange
ARDUINO
GY-31


R=79
B=35 G=49
Blue


R=22
B=64 G=36
Yellow

CONCLUSION

In conclusion, the utilization of the TCS230 sensor in the context of signal processing has proven to be both insightful and promising. Through our experimentation, we successfully integrated the TCS230 sensor into our setup, aiming to explore its capabilities and applications.

The TCS230 sensor, with its principle of operation based on color sensing, demonstrated reliable performance in capturing and processing signals related to the intensity and frequency of incident light. The obtained results revealed a clear correlation between the sensor's output and the characteristics of the applied stimuli.

Our analysis of the data highlighted the sensor's strengths in accurately detecting and quantifying color variations. The versatility of the TCS230 was evident as it provided consistent results across a range of test scenarios. This reliability positions the sensor as a valuable tool for applications requiring precise color recognition.

However, it's essential to acknowledge certain limitations encountered during our experimentation. Factors such as ambient light conditions and sensor calibration proved to influence the sensor's output, prompting consideration for further refinement in real-world applications.

Despite these challenges, the TCS230 sensor holds great potential in various practical applications. Its effectiveness in color sensing opens avenues for integration into fields such as industrial automation, robotics, and quality control processes.

In the future, advancements in sensor calibration techniques and noise reduction methods could enhance the TCS230's performance further. Additionally, exploring collaborative applications with other sensors and technologies may unlock new possibilities for comprehensive signal processing solutions.

In summary, our exploration of the TCS230 sensor in the realm of signal processing has provided valuable insights into its capabilities and limitations. As we continue to refine and expand our understanding of this technology, the TCS230 stands poised as a reliable and adaptable tool for a wide range of applications in color sensing and signal processing.

REFERENCE MATERIALS

U can scan this qr

code to watch the video of TCS 230