

# Relatório Técnico — Sensor de Cor GY-33 (TCS34725)

Ana Beatriz Barbosa Yoshida — RA: 245609 — @beatrizbarbosay  
Julio Nunes Avelar — RA: 241163 — @JN513  
Turma EA701 — 2025S2

11 de Novembro de 2025

**Repositório:** [https://github.com/JN513/sensor\\_tcs34725\\_yoshida\\_avelar](https://github.com/JN513/sensor_tcs34725_yoshida_avelar)

## 1 Escopo e Objetivos

O presente experimento teve como objetivo desenvolver um sistema embarcado capaz de detectar e identificar cores utilizando o sensor óptico **GY-33 (TCS34725)** acoplado à placa **BitDogLab**. O projeto foi implementado em linguagem **C/C++** utilizando o **Pico SDK**, explorando os recursos de comunicação digital **I2C** e controle de brilho via **PWM por hardware** do microcontrolador **RP2040**.

Os principais objetivos técnicos foram:

- Implementar comunicação I2C com o sensor TCS34725;
- Calibrar e ler os canais de cor (R, G, B e C);
- Converter os valores normalizados para os formatos RGB (0–255) e HSV;
- Controlar o LED RGB da BitDogLab via PWM;
- Classificar a cor dominante de acordo com faixas de matiz (Hue);
- Reproduzir a cor detectada no LED RGB, garantindo coerência visual.

O sistema foi considerado funcional quando a cor reproduzida pelo LED correspondia à cor observada no objeto e a classificação textual coincidiu com o esperado.

## 2 Metodologia e Implementação

### 2.1 Arquitetura do Sistema

O sistema é composto por dois módulos principais:

- Sensor de cor (GY-33 / TCS34725)** — responsável pela leitura das intensidades de luz refletida nos canais vermelho, verde, azul e claro, comunicando-se com o microcontrolador via I2C.
- LED RGB da BitDogLab** — reproduz a cor detectada por meio de três canais PWM independentes (R, G e B).



## 2.2 Ligações Elétricas

Pino Sensor	GPIO BitDogLab	Função	Observação
VCC	3.3V	Alimentação	Compatível 3.3V–5V
GND	GND	Terra comum	—
SDA	GPIO0	I2C0 SDA	Dados
SCL	GPIO1	I2C0 SCL	Clock

O LED RGB integrado à BitDogLab utiliza os pinos:

$$R = \text{GPIO13}, \quad G = \text{GPIO11}, \quad B = \text{GPIO12}$$

Ajustes de brilho foram necessários para evitar mistura excessiva de branco.

## 2.3 Desenvolvimento de Software

O firmware foi desenvolvido utilizando o **Pico SDK v2.2.0**, com as seguintes bibliotecas:

- `hardware_i2c.h` — comunicação com o sensor;
- `hardware_pwm.h` — controle do LED RGB;
- `pico/stdlib.h` e `stdio.h` — depuração e temporização.

**Fluxo lógico do programa:**

1. Inicializa periféricos (UART, I2C e PWM);
2. Lê valores dos canais R, G, B e C;
3. Normaliza os dados com base no canal “clear”;
4. Converte os valores RGB para HSV;
5. Classifica a cor dominante de acordo com o matiz (Hue);
6. Atualiza o LED RGB via PWM;
7. Exibe no terminal a cor detectada e seu matiz.

## 2.4 Conversão RGB–HSV e Classificação de Cores

Foi implementado um algoritmo de conversão RGB–HSV para detecção perceptiva de cores. A classificação foi baseada em faixas de matiz (Hue):

Faixa de Hue (°)	Cor Dominante
0–15 ou >345	Vermelho
15–45	Laranja
45–70	Amarelo
70–160	Verde
160–200	Ciano
200–260	Azul
260–320	Roxo
320–345	Magenta

As faixas foram ajustadas empiricamente para melhorar a distinção entre tons intermediários.

## 2.5 Controle PWM do LED RGB

Cada canal foi configurado com frequência de 1 kHz e resolução de 16 bits. Os valores normalizados (0–255) são convertidos em *duty cycle* proporcional, invertido devido ao LED de ânodo comum. Foi realizada calibração manual para equilibrar o brilho entre canais.

## 3 Resultados e Análise

Os testes foram conduzidos sob iluminação branca ambiente. O sistema foi capaz de identificar e reproduzir as cores principais com boa correspondência entre leitura e exibição.

Cor Observada	Hue (°)	Cor Detectada	RGB Reproduzido
Garrafinha vermelha	~5	Vermelho	(255, 0, 0)
Camisa verde	~130	Verde	(0, 255, 0)
Camisa azul	~230	Azul	(0, 0, 255)
Garrafinha amarela	~55	Amarelo	(255, 255, 0)
Mesa branca	—	Branco/Cinza	(255, 255, 255)

**Observações:**

- Em tons claros, o LED tende ao branco ou cinza;
- A detecção de preto exigiu limiar mínimo no canal “clear”.

A figura ?? mostra a montagem do sistema, enquanto a figura 1 apresenta a saída no terminal durante os testes.

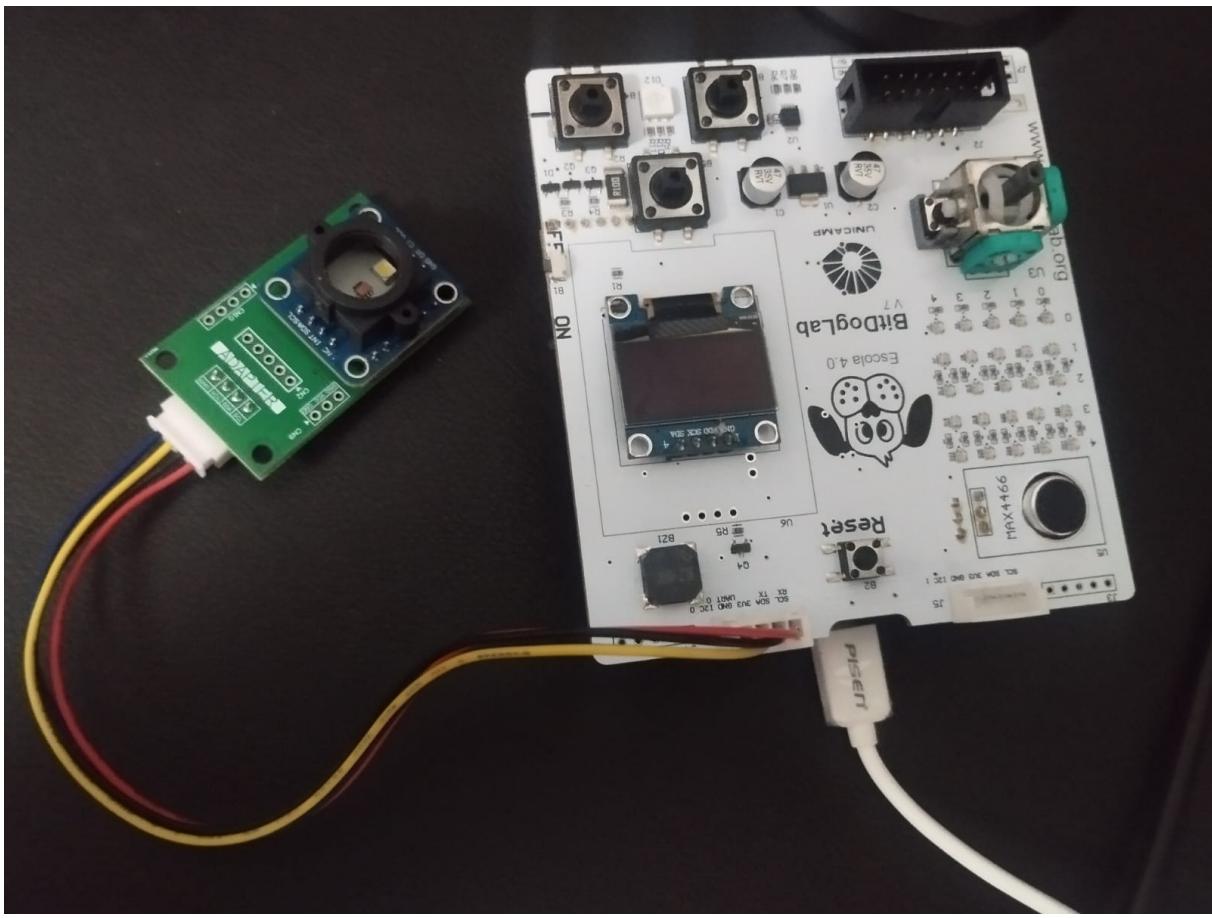


Figura 1: Montagem do sistema na BitDogLab.

Monitor Mode	Serial	View Mode	Text	Port	/dev/ttyACM0 - Raspberry Pi	Baud rate	115200
RAW -> C:659	R:80	G:104	B:69		HEX -> #506845	COR DETECTADA: VERDE	
RAW -> C:659	R:80	G:104	B:69		HEX -> #506845	COR DETECTADA: VERDE	
RAW -> C:652	R:79	G:104	B:69		HEX -> #4F6845	COR DETECTADA: VERDE	
RAW -> C:664	R:80	G:105	B:69		HEX -> #506945	COR DETECTADA: VERDE	
RAW -> C:671	R:80	G:105	B:68		HEX -> #506944	COR DETECTADA: VERDE	
RAW -> C:714	R:81	G:105	B:67		HEX -> #516943	COR DETECTADA: VERDE	
RAW -> C:734	R:81	G:105	B:67		HEX -> #516943	COR DETECTADA: VERDE	
RAW -> C:733	R:81	G:105	B:67		HEX -> #516943	COR DETECTADA: VERDE	
RAW -> C:727	R:81	G:105	B:67		HEX -> #516943	COR DETECTADA: VERDE	
RAW -> C:754	R:82	G:106	B:66		HEX -> #526A42	COR DETECTADA: VERDE	
RAW -> C:744	R:81	G:106	B:66		HEX -> #516A42	COR DETECTADA: VERDE	
RAW -> C:725	R:81	G:106	B:67		HEX -> #516A43	COR DETECTADA: VERDE	
RAW -> C:737	R:81	G:106	B:66		HEX -> #516A42	COR DETECTADA: VERDE	
RAW -> C:741	R:81	G:106	B:66		HEX -> #516A42	COR DETECTADA: VERDE	
RAW -> C:602	R:77	G:105	B:71		HEX -> #4D6947	COR DETECTADA: VERDE	
RAW -> C:596	R:76	G:104	B:71		HEX -> #4C6847	COR DETECTADA: VERDE	
RAW -> C:592	R:76	G:105	B:71		HEX -> #4C6947	COR DETECTADA: VERDE	

Figura 2: Saída no terminal durante o teste.

## 4 Dificuldades e Soluções

- **Brilho excessivo no LED RGB:** calibração de brilho para evitar saturação branca.
- **Limites imprecisos entre cores próximas:** ampliação das faixas de Hue.
- **Influência da iluminação ambiente:** padronização da luz de teste.
- **Sensibilidade desigual dos canais RGB:** compensação manual de ganho (redução no canal verde).

## 5 Conclusões e Trabalhos Futuros

O projeto alcançou os objetivos propostos, demonstrando o funcionamento correto do sensor TCS34725 na detecção de cores. A integração entre leitura I2C, conversão HSV e controle PWM resultou em um sistema eficiente, didático e de boa responsividade.

**Trabalhos futuros sugeridos:**

- Implementar calibração automática com referência branca;
- Adicionar compensação de luminosidade ambiente (auto gain);
- Integrar exibição da cor detectada em display OLED;
- Expandir para reconhecimento de padrões multicoloridos;

## 6 Referências

- ams AG. *TCS34725 Color Light-to-Digital Converter with IR Filter — Datasheet*. Disponível em: <https://cdn-shop.adafruit.com/datasheets/TCS34725.pdf>
- Raspberry Pi Foundation. *Pico SDK — C/C++ Development Guide*. Disponível em: <https://www.raspberrypi.com/documentation/microcontrollers/>
- BitDogLab. *Documentação de Hardware — Mapeamento de GPIOs e periféricos*.
- Adafruit Industries. *TCS34725 Color Sensor Guide*. Disponível em: <https://learn.adafruit.com/adafruit-color-sensors>
- Repositório do projeto: [https://github.com/JN513/sensor\\_tcs34725\\_yoshida\\_avelar](https://github.com/JN513/sensor_tcs34725_yoshida_avelar)