

Aula 2 – Motores, Ponte H e Sensores

Curso: Introdução à Robótica Móvel

PatoBots - UTFPR



CRONOGRAMA

- Motores.
- Acionamento do Motor.
- Ponte H.
- Acionamento da Ponte H.
- Tipos de Sensores: Sensor Óptico Reflexivo e Sensor de Distância Ultrassônico.
- Leitura dos Sensores.



MOTORES



MOTOR DC (Corrente Contínua)

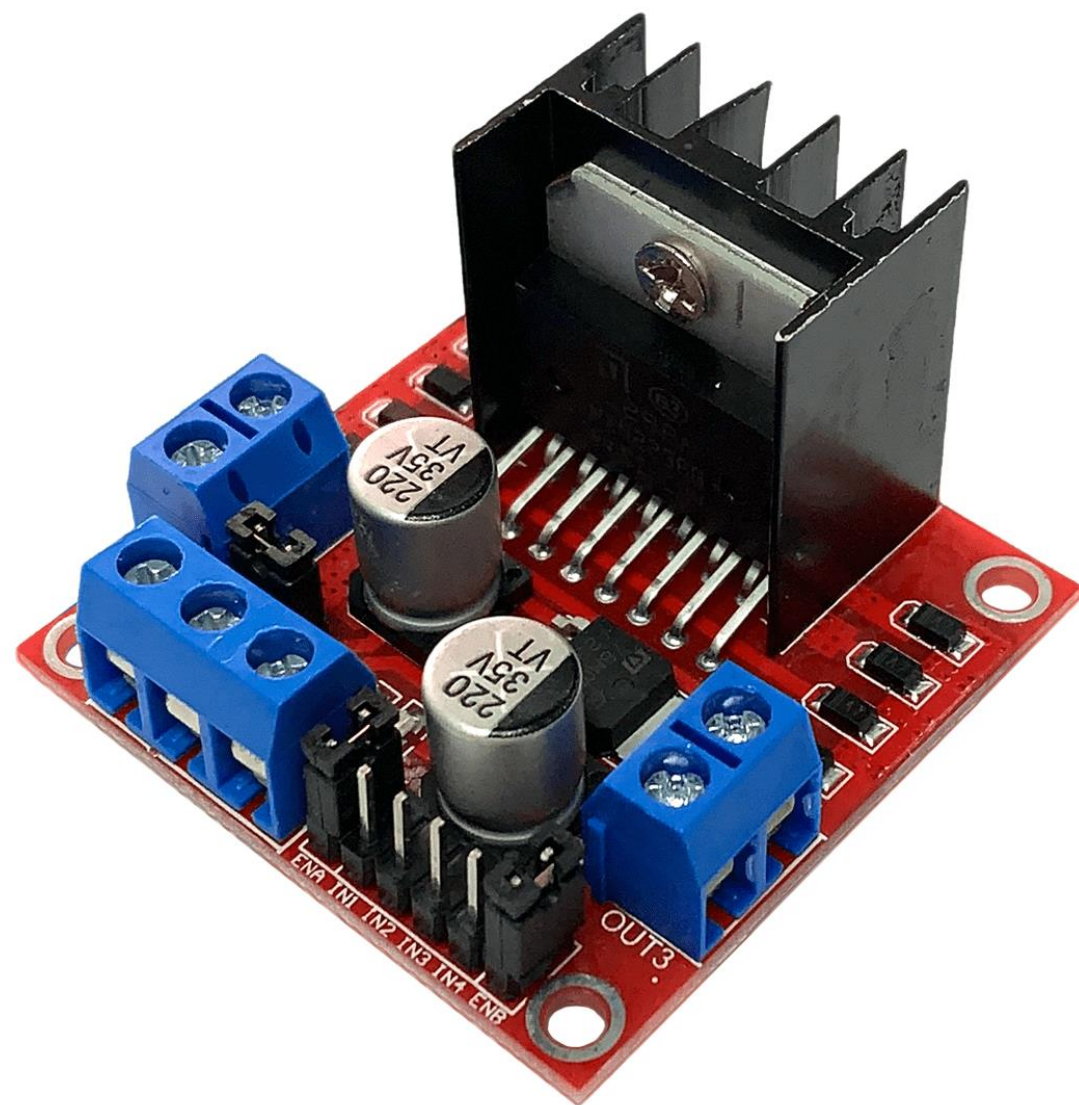


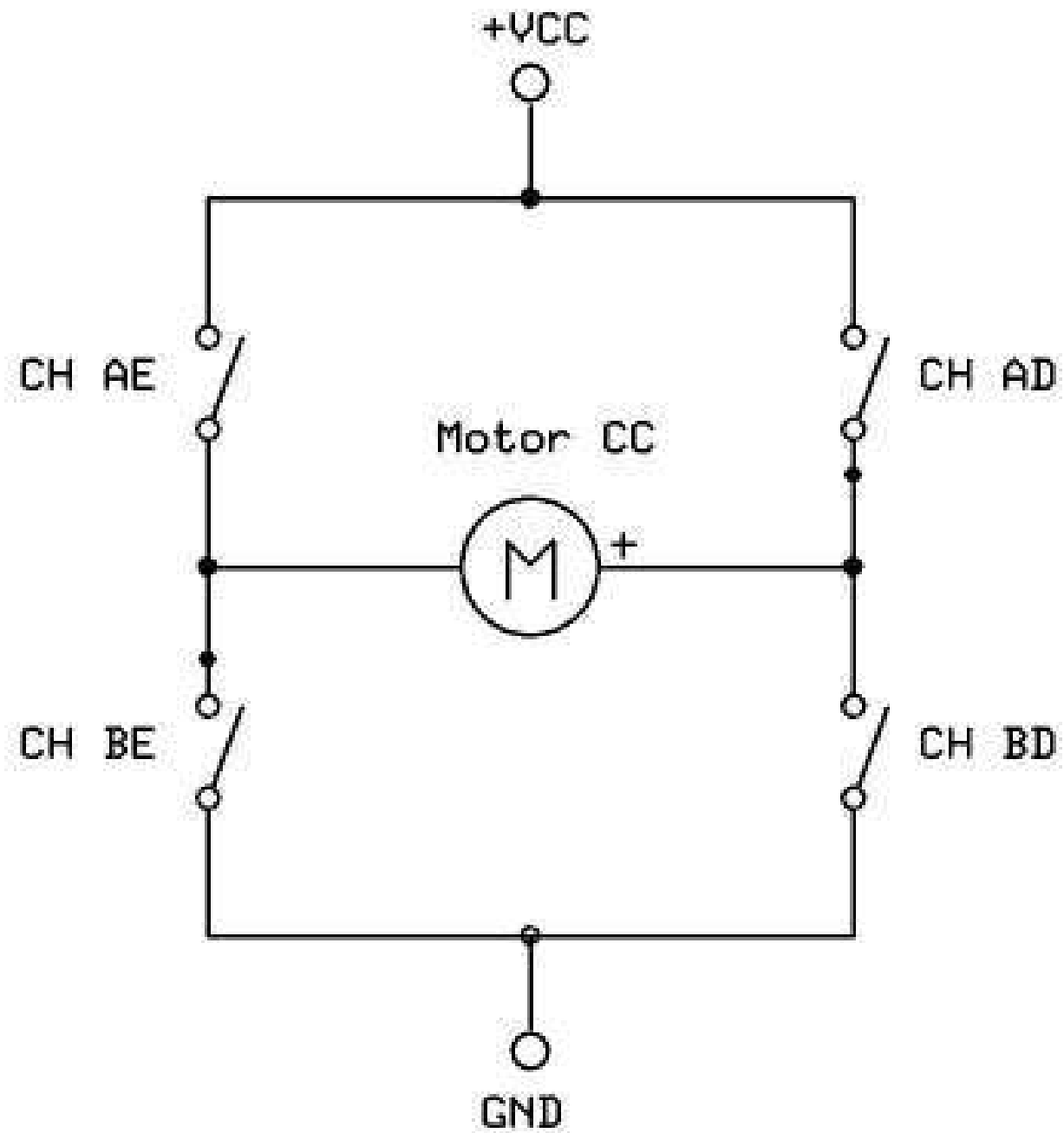
- Dispositivo que transforma energia elétrica em energia rotacional mecânica.
- Teste de Giro invertendo polaridade

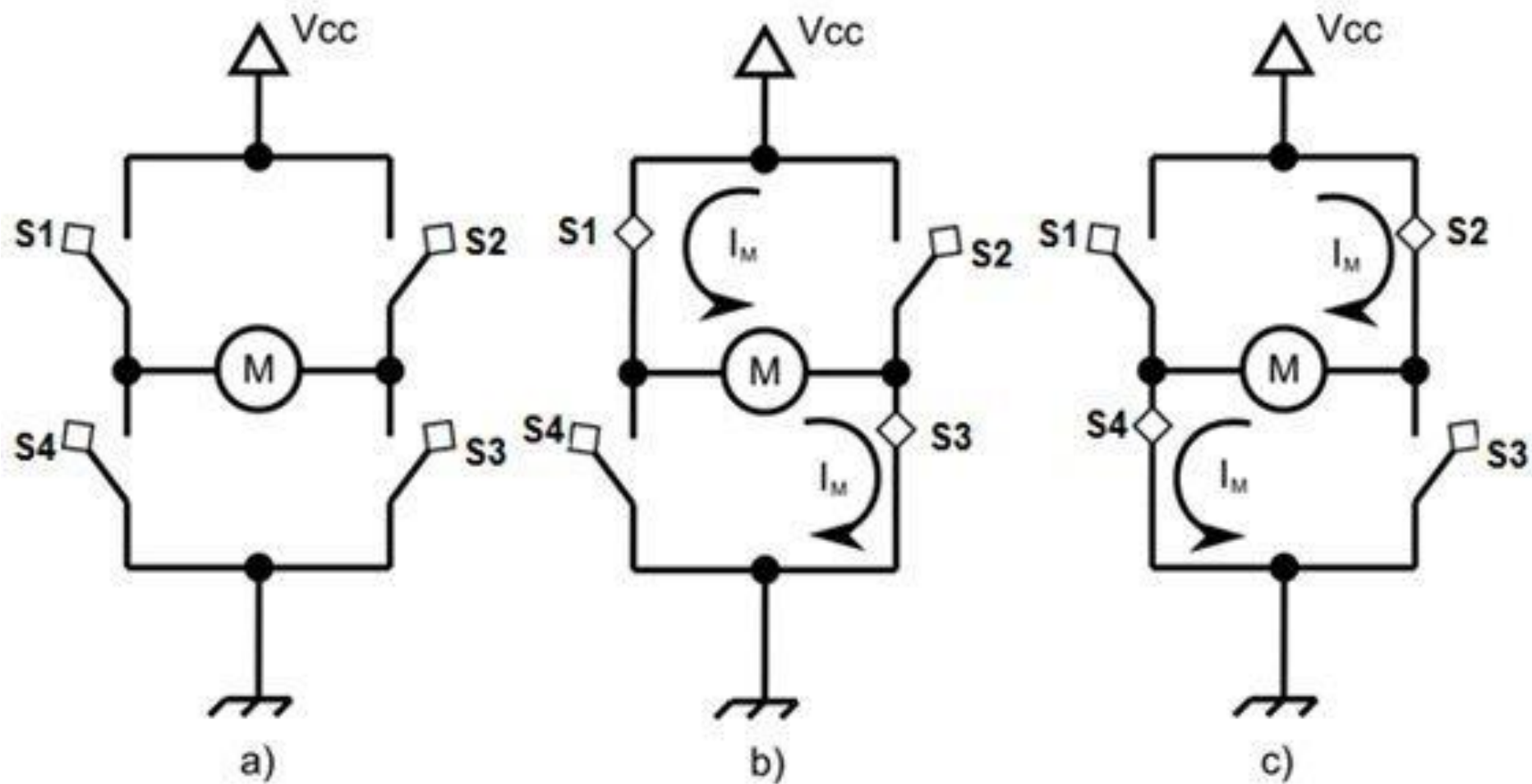


PONTE H



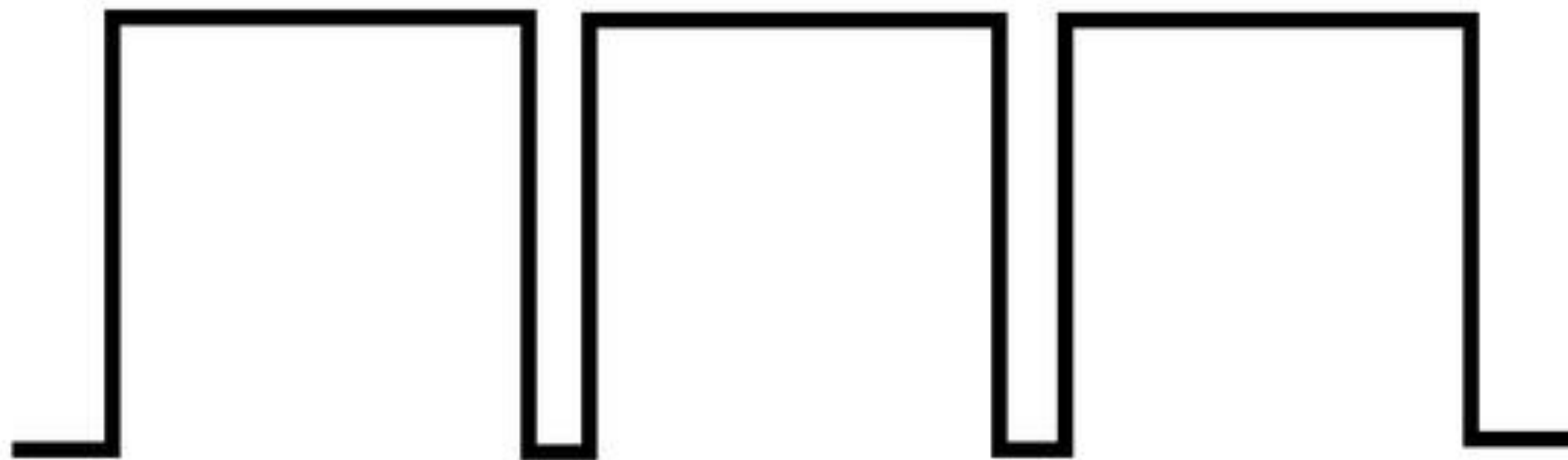






+VCC

0V



Largura do Pulso

$t_{(on)}$

$t_{(off)}$

Manual Da Eletrônica



PONTE H

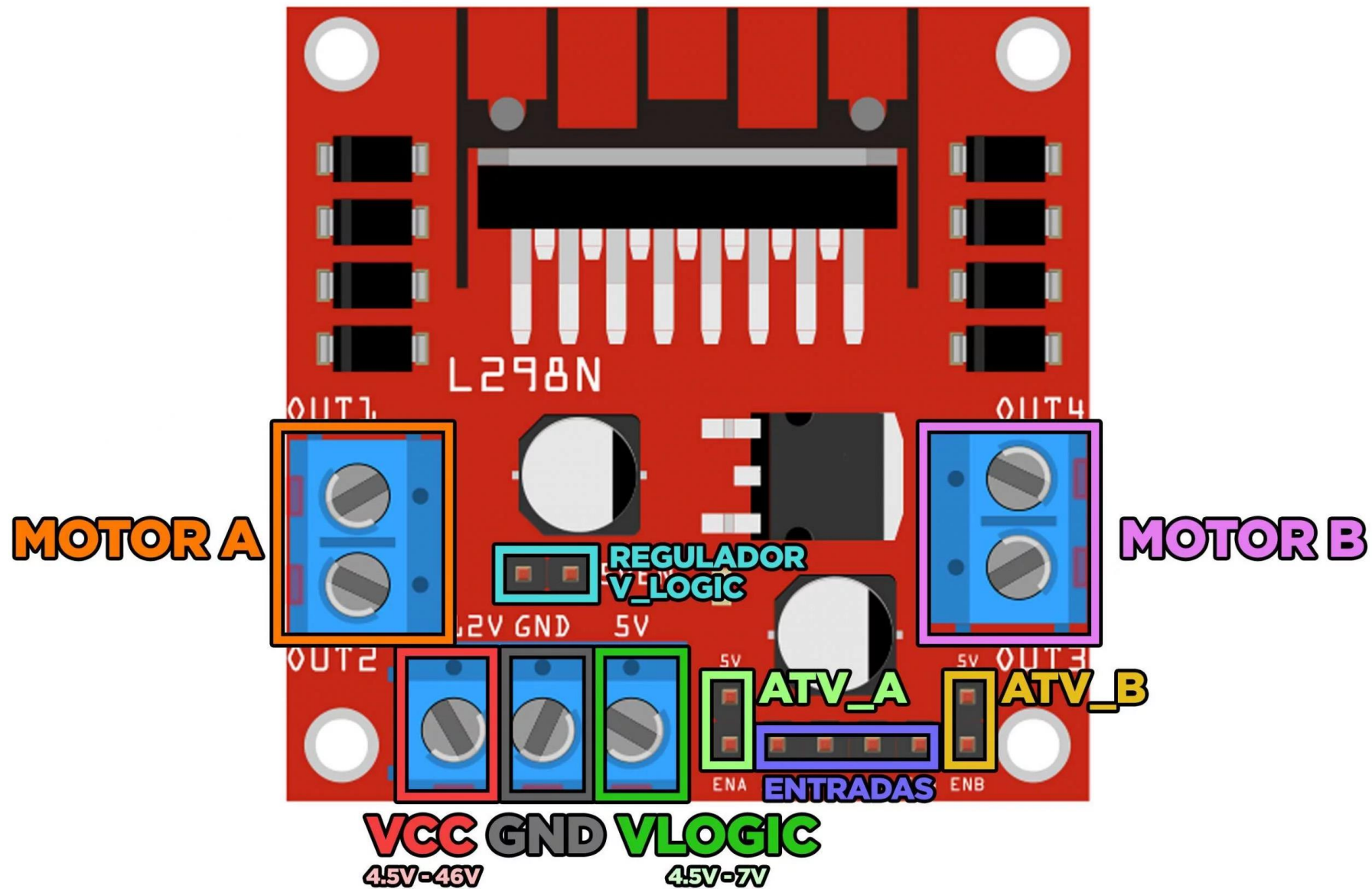
- Permite rotação do motor para ambos os lados.
- Utiliza comutação de chaves para variar estados de rotação.
- No módulo Ponte H L298N, a comutação é eletrônica via um chip embarcado.
- Diferencia-se pela comutação eletrônica em comparação com a mecânica.
- O chip na placa realiza a comutação de chaves de forma eficiente.



PONTE H

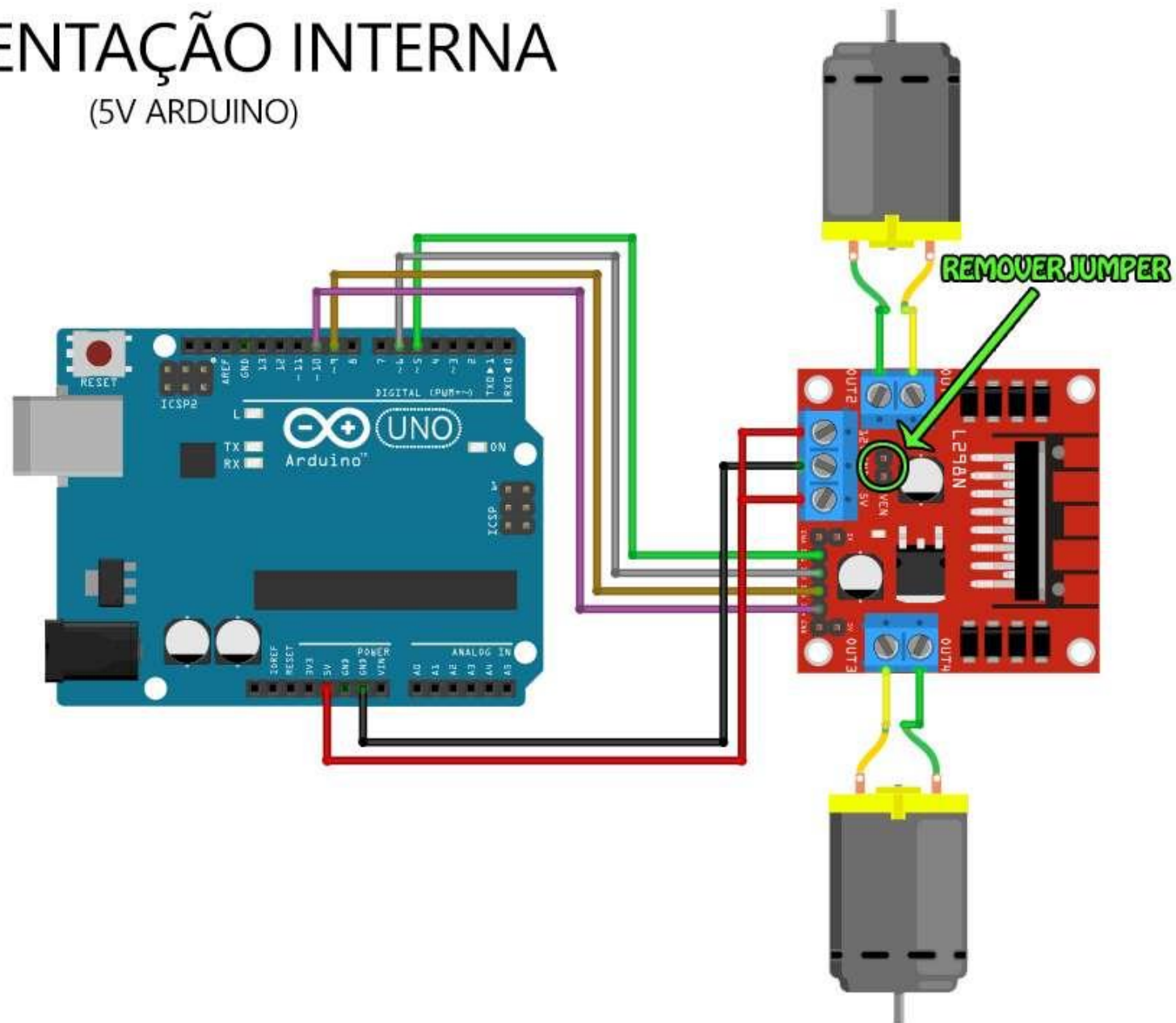
- Permite maior dissipação de potência em comparação com acionamento direto por microprocessadores.
- A ponte H L298N fornece 2A por canal, superando a capacidade de alguns mA por portas de circuitos microprocessados.
- Cuidado necessário com a tensão de operação do motor.
- Exceder o limite de alimentação V_{cc} pode resultar em danos ao motor e possíveis acidentes.





ALIMENTAÇÃO INTERNA

(5V ARDUINO)



```
1  /* Iremos fazer uma classe para facilitar o uso da ponte H L298N
2  na manipulação dos motores na função Setup e Loop.*/
3
4  class DCMotor {
5      int spd = 255, pin1, pin2;
6
7      public:
8
9          void Pinout(int in1, int in2){ /* Pinout é o método para a declaração
10 dos pinos que vão controlar o objeto motor*/
11     pin1 = in1;
12     pin2 = in2;
13     pinMode(pin1, OUTPUT);
14     pinMode(pin2, OUTPUT);
15     }
16     void Speed(int in1){ /* Speed é o método que irá ser responsável
17 por salvar a velocidade de atuação do motor*/
18     spd = in1;
19     }
20     void Forward(){ // Forward é o método para fazer o motor girar para frente
21     analogWrite(pin1, spd);
22     digitalWrite(pin2, LOW);
23     }
24     void Backward(){ // Backward é o método para fazer o motor girar para trás
25     digitalWrite(pin1, LOW);
26     analogWrite(pin2, spd);
27     }
```



```
26     analogWrite(pin2, spd);
27     }
28     void Stop(){ // Stop é o metodo para fazer o motor ficar parado.
29         digitalWrite(pin1, LOW);
30         digitalWrite(pin2, LOW);
31     }
32 };
33
34     DCMotor Motor1, Motor2; /* Criação de dois objetos motores, já que usaremos dois motores,
35 e eles já estão prontos para receber os comandos já configurados acima. */
36
37 void setup() {
38     Motor1.Pinout(5,6); // Seleção dos pinos que cada motor usará, como descrito na classe.
39     Motor2.Pinout(9,10);
40 }
```



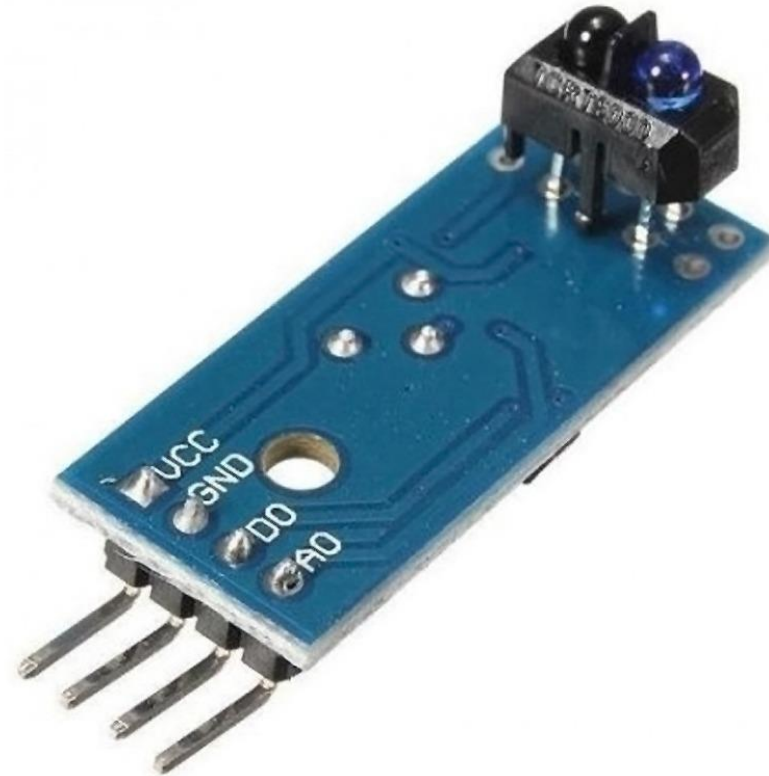

```
40 }  
41  
42 void loop() {  
43     Motor1.Speed(200); /* A velocidade do motor pode variar de 0 a 255,  
44     onde 255 é a velocidade máxima.*/  
45     Motor2.Speed(200);  
46  
47     Motor1.Forward(); // Comando para o motor ir para frente  
48     Motor2.Forward();  
49     delay(1000);  
50     Motor1.Backward(); // Comando para o motor ir para trás  
51     Motor2.Backward();  
52     delay(1000);  
53     Motor1.Stop(); // Comando para o motor parar  
54     Motor2.Stop();  
55     delay(500);  
56 }
```

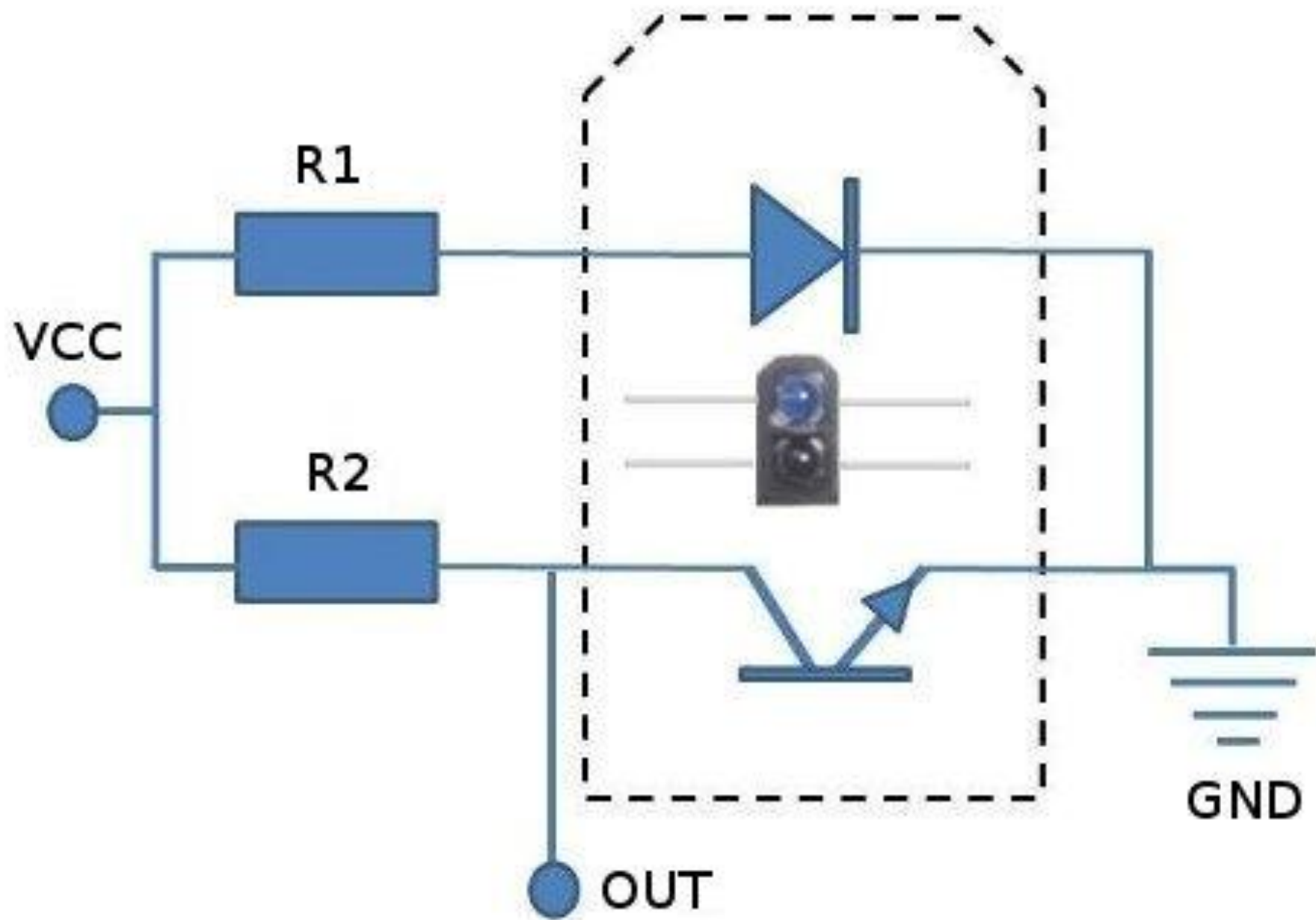


SENSOR ÓPTICO REFLEXIVO



SENSOR ÓPTICO REFLEXIVO





SENSOR ÓPTICO REFLEXIVO

- Esse sensor tem dois componentes: Um sensor infravermelho (Azul) e um fototransistor (Preto) divididos por uma divisória preta
- Quando algum objeto se aproxima do sensor, a luz infravermelha é refletida no objeto, passa para o outro lado, ativando o resistor e fazendo a leitura.
- Quanto mais reflexivo o material (como por exemplo a cor branca), maior será o sinal enviado, e quanto menos reflexivo (como por exemplo a cor preta), o sinal enviado será próximo de zero.



SENSOR ÓPTICO REFLEXIVO

- Ele tem dois modos de leitura: Analógico e Digital;
- O Modo Analógico vai enviar a quantidade de reflexão que o sensor está lendo;
- O Modo Digital vai enviar se o sensor está lendo a reflexão ou não;

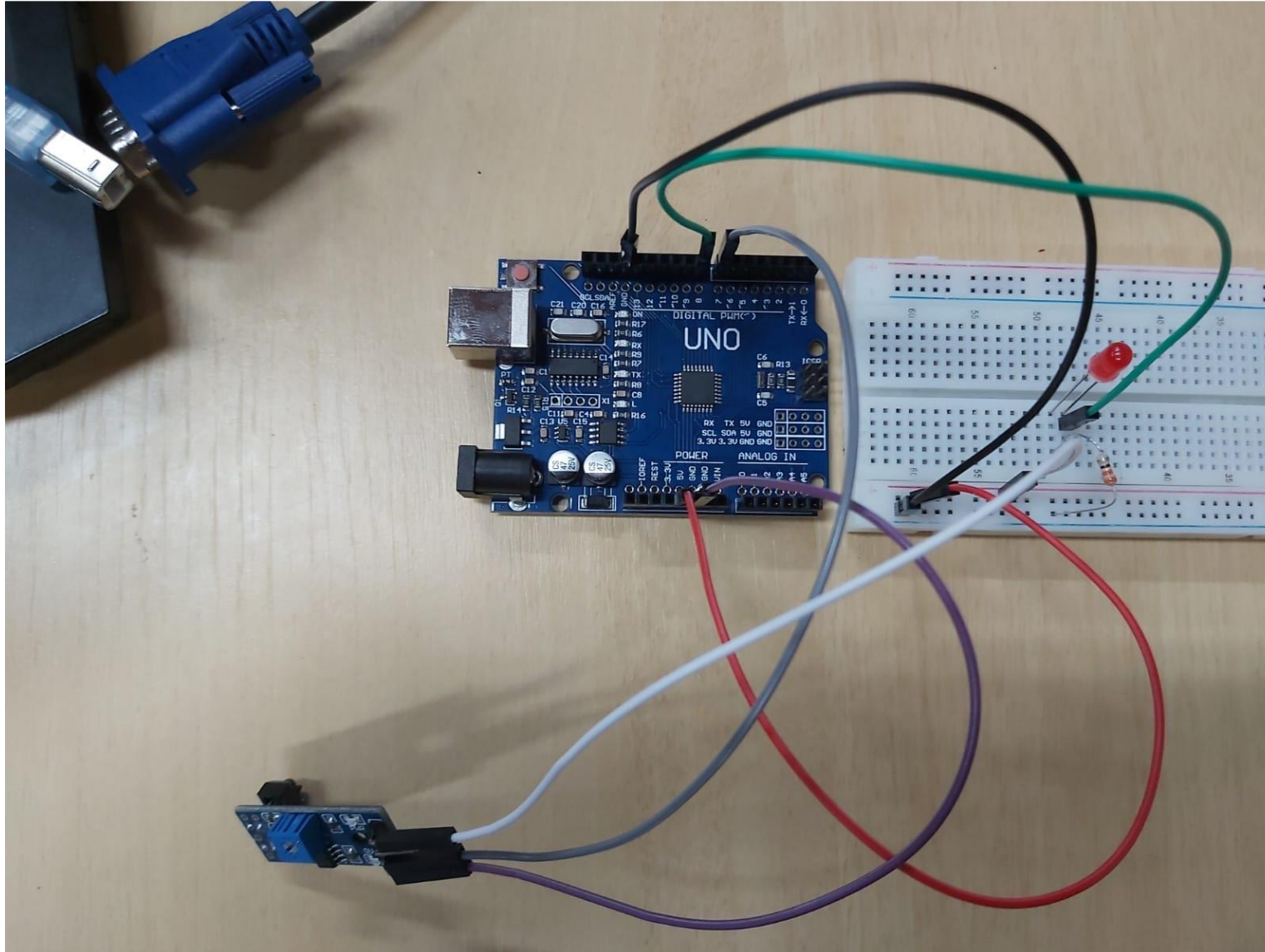


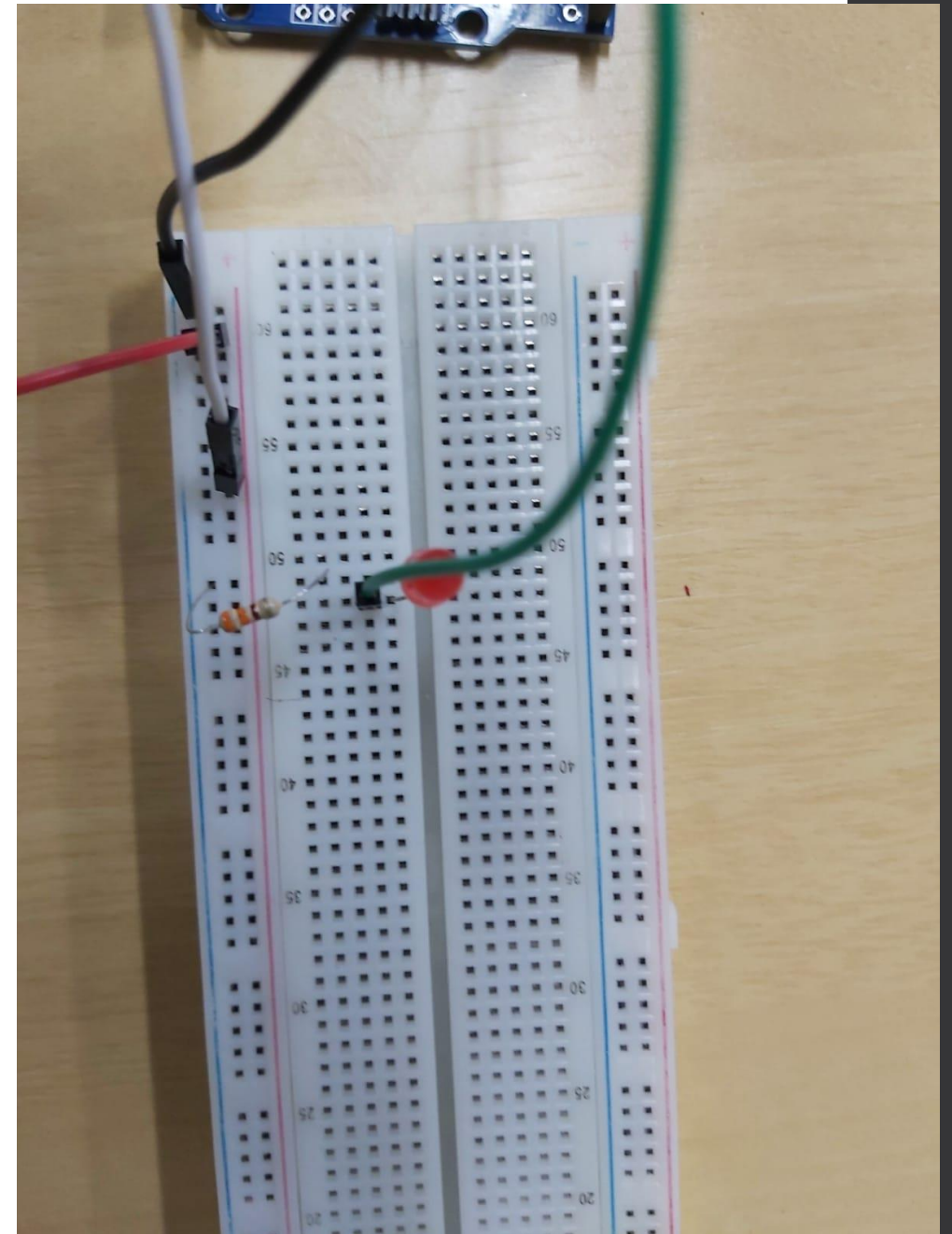
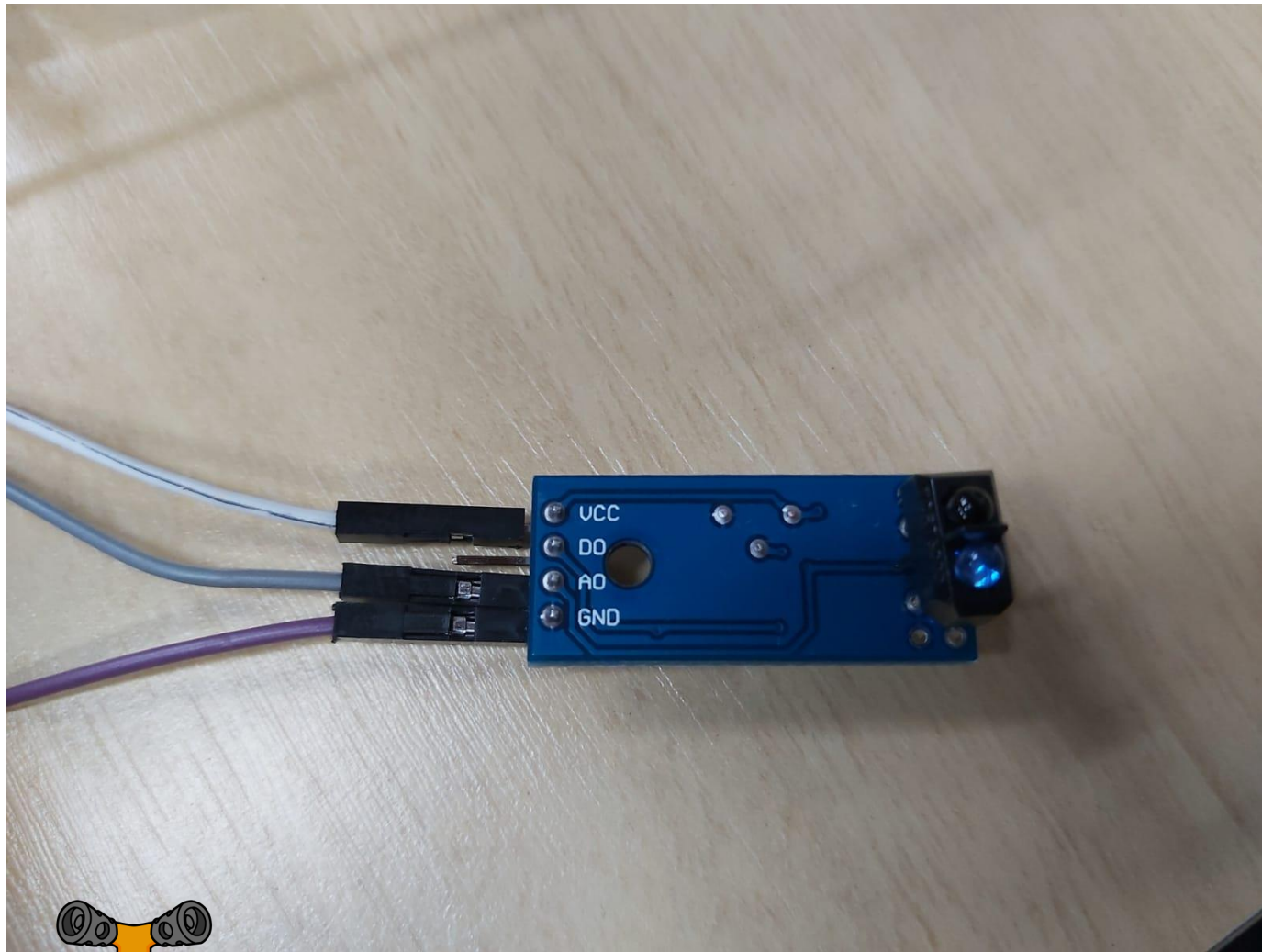
SENSOR ÓPTICO REFLEXIVO

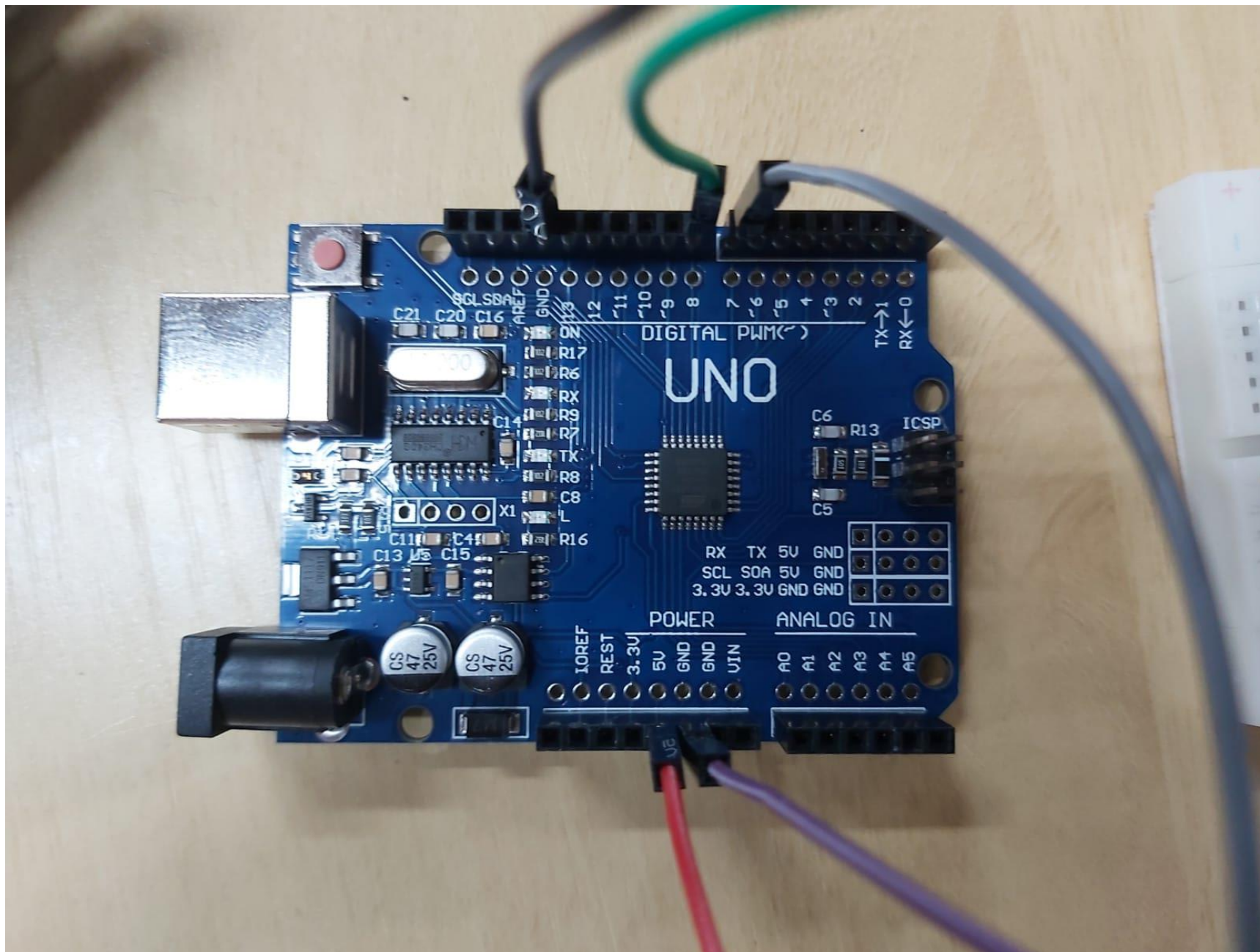
- Este sensor óptico reflexivo está soldado em uma placa que faz aumentar a distância qual o sensor consegue refletir seu infravermelho, também acertando uma maior precisão.
- Na parte de cima da placa tem um potenciômetro que ajusta a precisão da leitura do sensor, regulando o quanto de sinal ele envia pela distância.



EXEMPLO DE LEITURA DE SENSOR ÓPTICO REFLEXIVO: DIGITAL



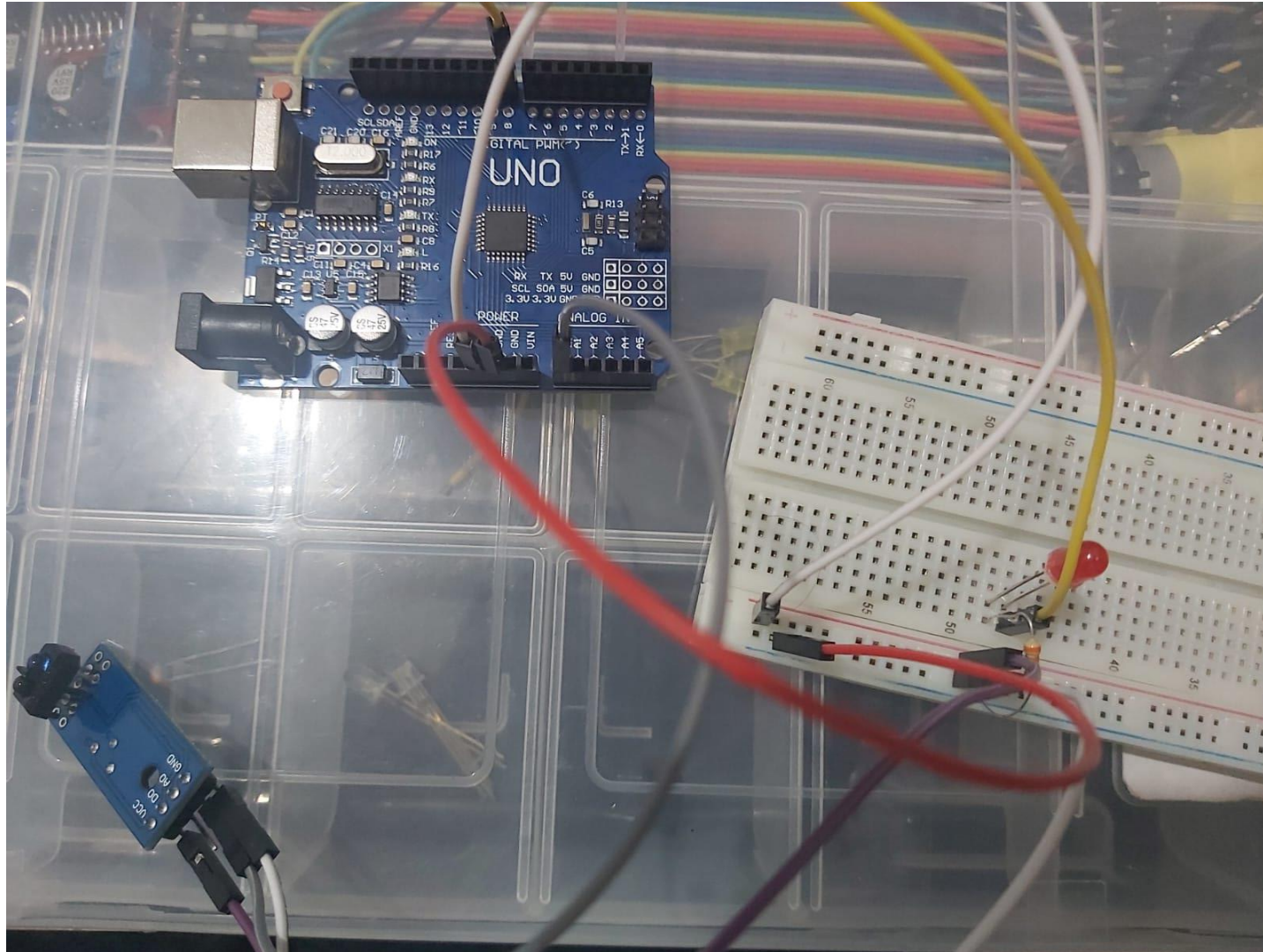


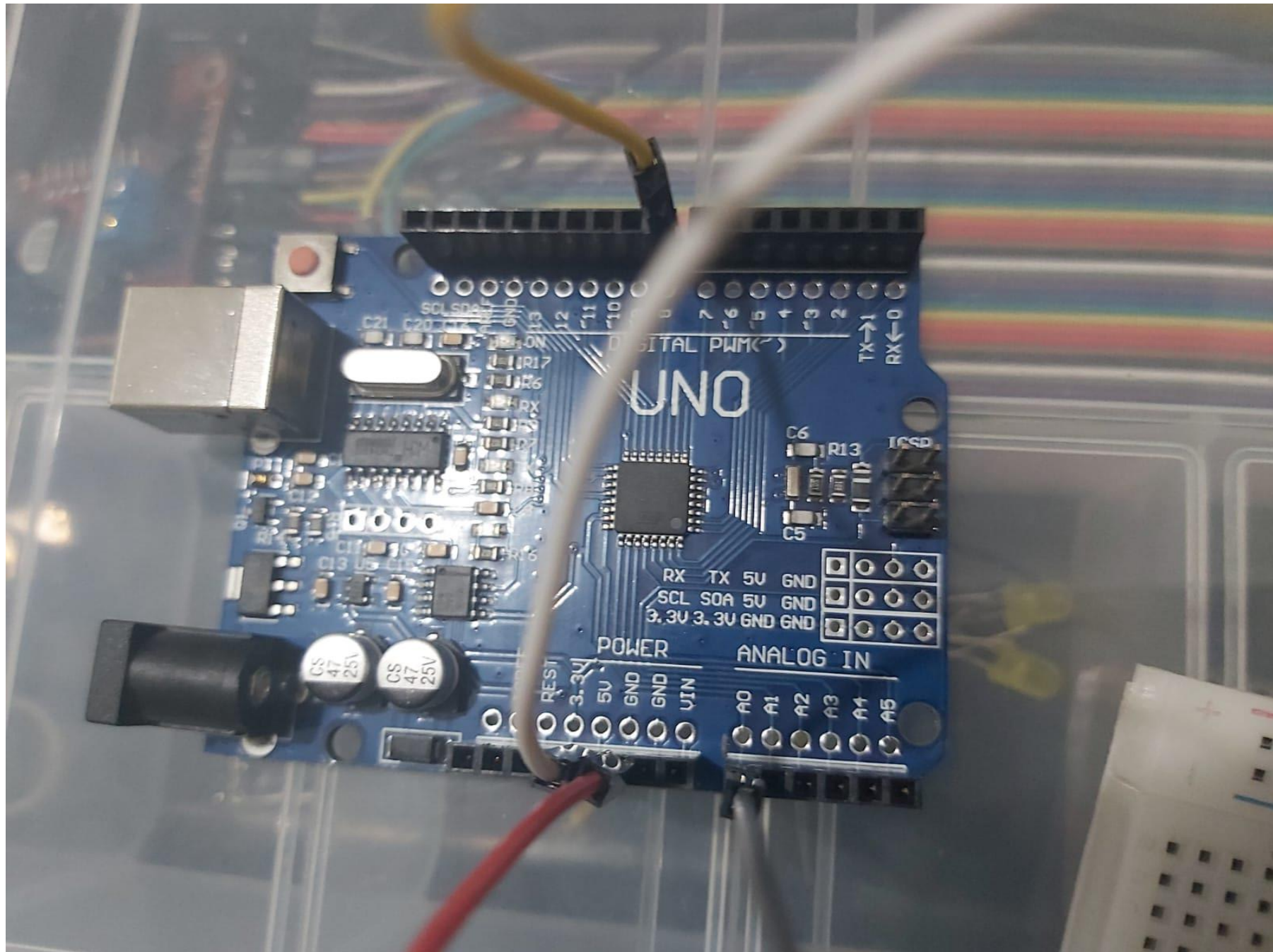


```
1 int led = 8; //Pino do led
2 int sensor = 7; //Ligado ao pino "coletor" do sensor óptico
3 int leitura = 0; //Armazena informações sobre a leitura do sensor
4
5 void setup()
6 {
7     pinMode(led, OUTPUT); //Define o pino do led como saída
8     pinMode(sensor, INPUT); //Define o pino do sensor como entrada
9 }
10
11 void loop()
12 {
13     //Le as informações do pino do sensor
14     leitura = digitalRead(sensor);
15
16     if (leitura != 1) //Verifica se o objeto foi detectado
17     {
18         digitalWrite(led, 1);
19     }
20     else{
21         digitalWrite(led, 0);
22     }
23 }
```



EXEMPLO DE LEITURA DE SENSOR ÓPTICO REFLEXIVO: ANALÓGICO





```
1 int led = 8; //Pino do led
2 int sensor = 0; //Ligado ao pino "coletor" do sensor óptico
3 int leitura; //Armazena informações sobre a leitura do sensor
4
5 void setup()
6 {
7     Serial.begin(9600);
8     pinMode(led, OUTPUT); //Define o pino do led como saída
9     pinMode(sensor, INPUT); //Define o pino do sensor como entrada
10 }
11
12 void loop()
13 {
14     //Le as informações do pino do sensor
15     leitura = analogRead(sensor);
16     Serial.println(leitura);
17     if (leitura < 341) //Verifica se o objeto foi detectado
18     {
19         digitalWrite(led, 1);
20     }
21     else{
22         digitalWrite(led, 0);
23     }
24 }
```



SENSOR DE DISTÂNCIA ULTRASSÔNICO





ULTRASSÔNICO

- Emite pulsos de ondas sonoras ultrassônicas.
- Mede o tempo que leva para os pulsos refletirem de um objeto de volta para o sensor.
- Alcance pode variar, geralmente entre 2cm a 4m com uma precisão de 3 milímetros.
- Sensível a condições ambientais, como temperatura e umidade.



ULTRASSÔNICO

- Não é afetado por cores ou texturas do objeto.
- Pode ser usado em diversas condições de iluminação.
- Pode ser influenciado por obstáculos que absorvem ou dispersam as ondas sonoras.
- Precisão pode diminuir em distâncias muito curtas.



ULTRASSÔNICO

- Detecção de objetos;
- Medição de distância;
- Garantindo de efetividade das máquinas agrícolas.

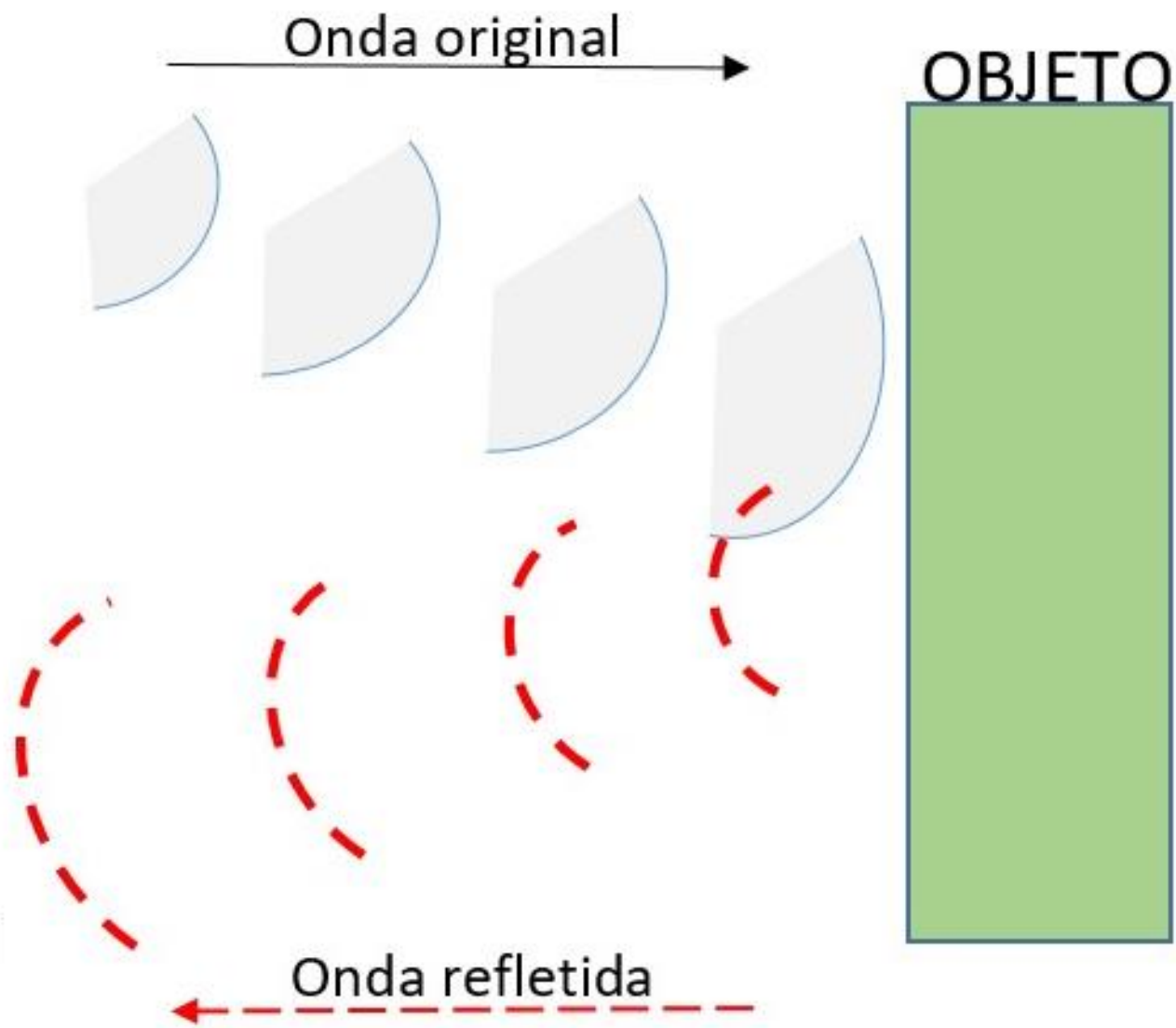


Transmissor

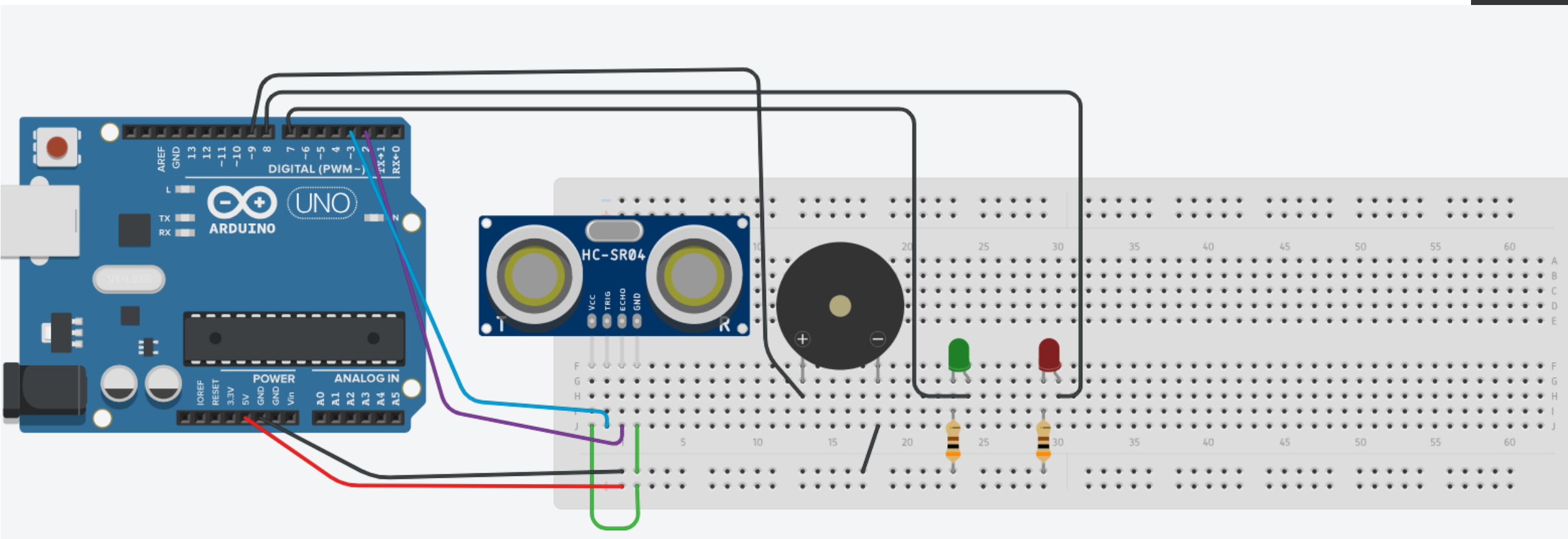


Receptor

Onda original



EXEMPLO DE LEITURA DE SENSOR ULTRASSÔNICO



```
1 //Configuração de Distancia Mínima em centímetros
2 const int distancia_carro = 10;
3
4 //Configurações de Portas do Arduino
5
6 //Sensor
7 const int TRIG = 3;
8 const int ECHO = 2;
9
10 //Demais componentes
11 const int ledGreen = 7;
12 const int ledRed = 8;
13 const int buzzer = 9;
14
15 // Variaveis para funcionamento do Buzzer
16 float seno;
17 int frequencia;
18
19 void setup() {
20     Serial.begin(9600);
21
22     // Configurações do Sensor
23     pinMode(TRIG, OUTPUT);
24     pinMode(ECHO, INPUT);
25
26     // Configurações do LED
27     pinMode(ledGreen, OUTPUT);
```



```
26 // Configurações do LED
27 pinMode(ledGreen, OUTPUT);
28 pinMode(ledRed, OUTPUT);
29
30 //Configurações do Buzzer
31 pinMode(buzzer, OUTPUT);
32
33 }
34
35 void loop() {
36     int distancia = sensor_morcego(TRIG,ECHO);
37
38     if(distancia <= distancia_carro){
39         Serial.print("Atenção: ");
40         Serial.print(distancia);
41         Serial.println("cm");
42         digitalWrite(ledGreen, LOW);
43         digitalWrite(ledRed, HIGH);
44         tocaBuzzer();
45     }
46     else{
47         Serial.print("Livre: ");
48         Serial.print(distancia);
49         Serial.println("cm");
50         digitalWrite(ledGreen, HIGH);
51         digitalWrite(ledRed, LOW);
```



```
51     digitalWrite(ledRed, LOW);
52     noTone(buzzer);
53 }
54 delay(100);
55
56 }
57
58 int sensor_morcego(int pinotrig, int pinoecho) {
59     digitalWrite(pinotrig, LOW);
60     delayMicroseconds(2);
61     digitalWrite(pinotrig, HIGH);
62     delayMicroseconds(10);
63     digitalWrite(pinotrig, LOW);
64
65     return pulseIn(pinoecho, HIGH) / 58;
66 }
67
68 //Função para execução do Alarme Sonoro
69 void tocaBuzzer() {
70     for(int x=0; x<180; x++) {
71         seno=(sin(x*3.1416/180));
72         frequencia = 2000+(int(seno*1000));
73         tone(buzzer, frequencia);
74         delay(2);
75     }
76
77 }
```

