Aula 1 – Motores, Ponte H e Sensores

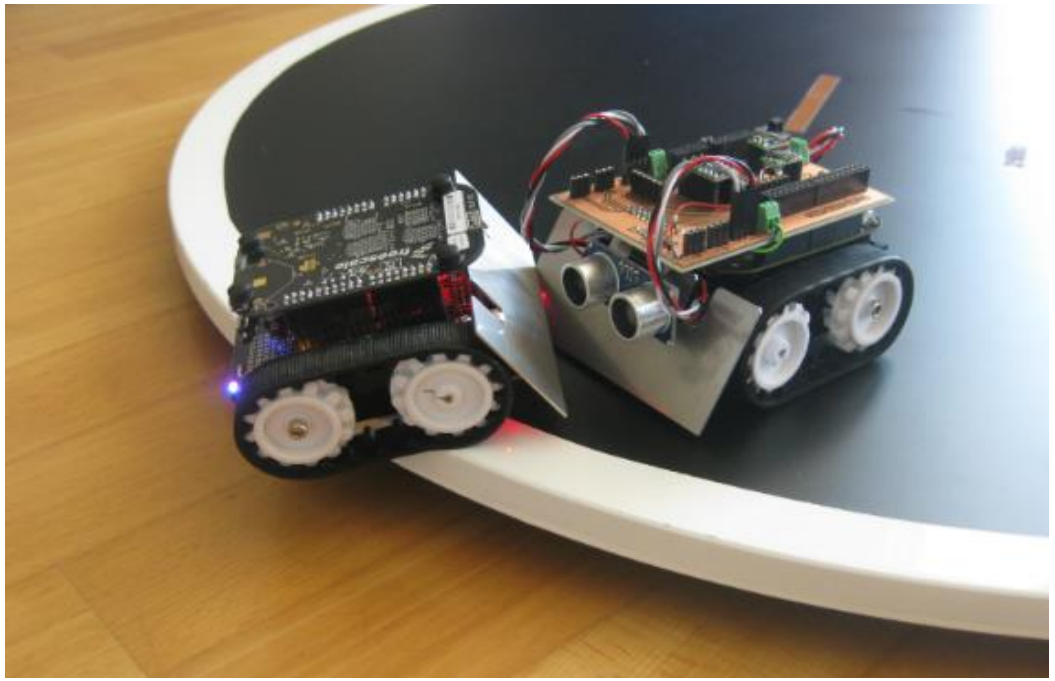
Curso: Introdução à Robótica Móvel

PatoBots - UTFPR

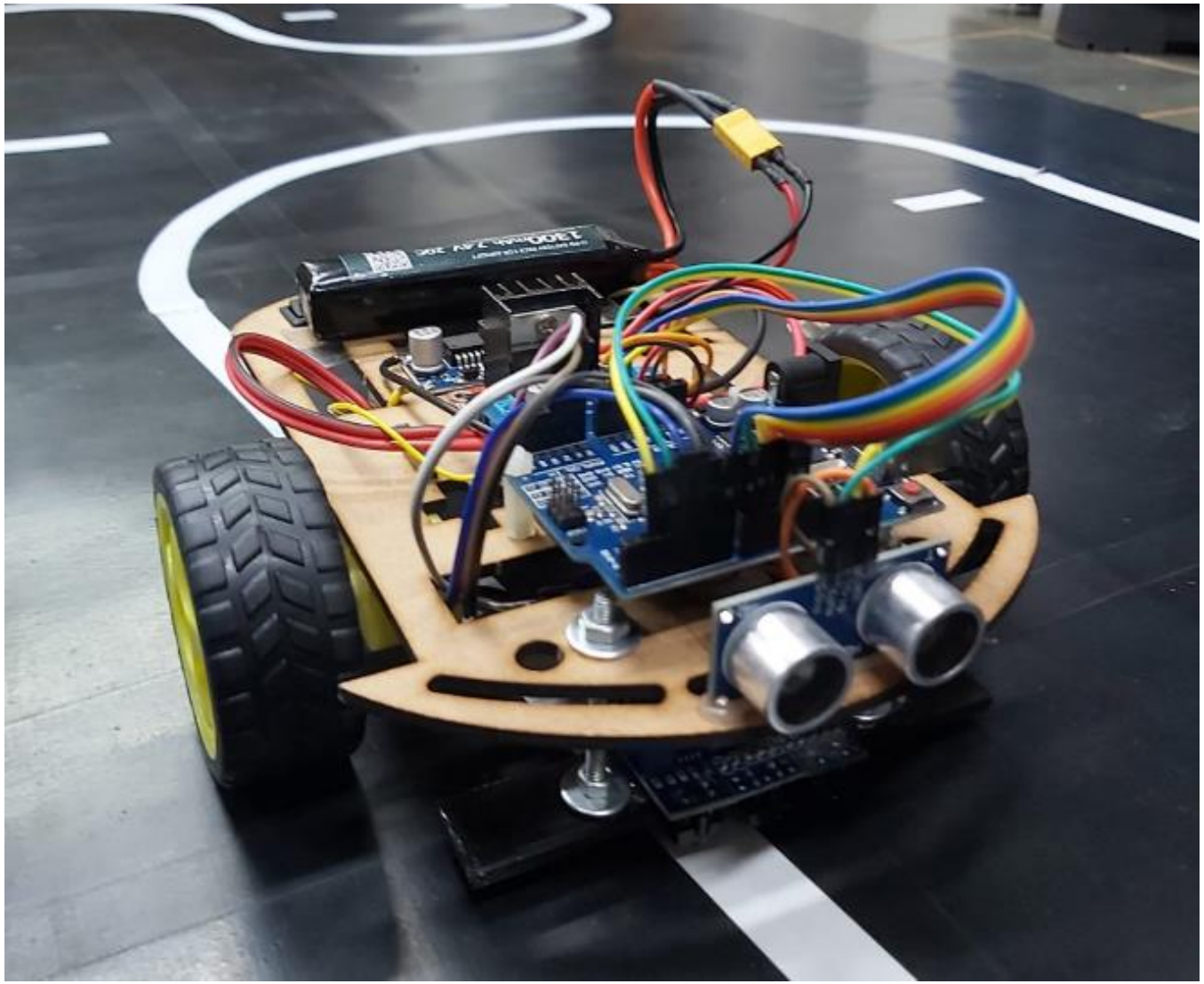
CRONOGRAMA

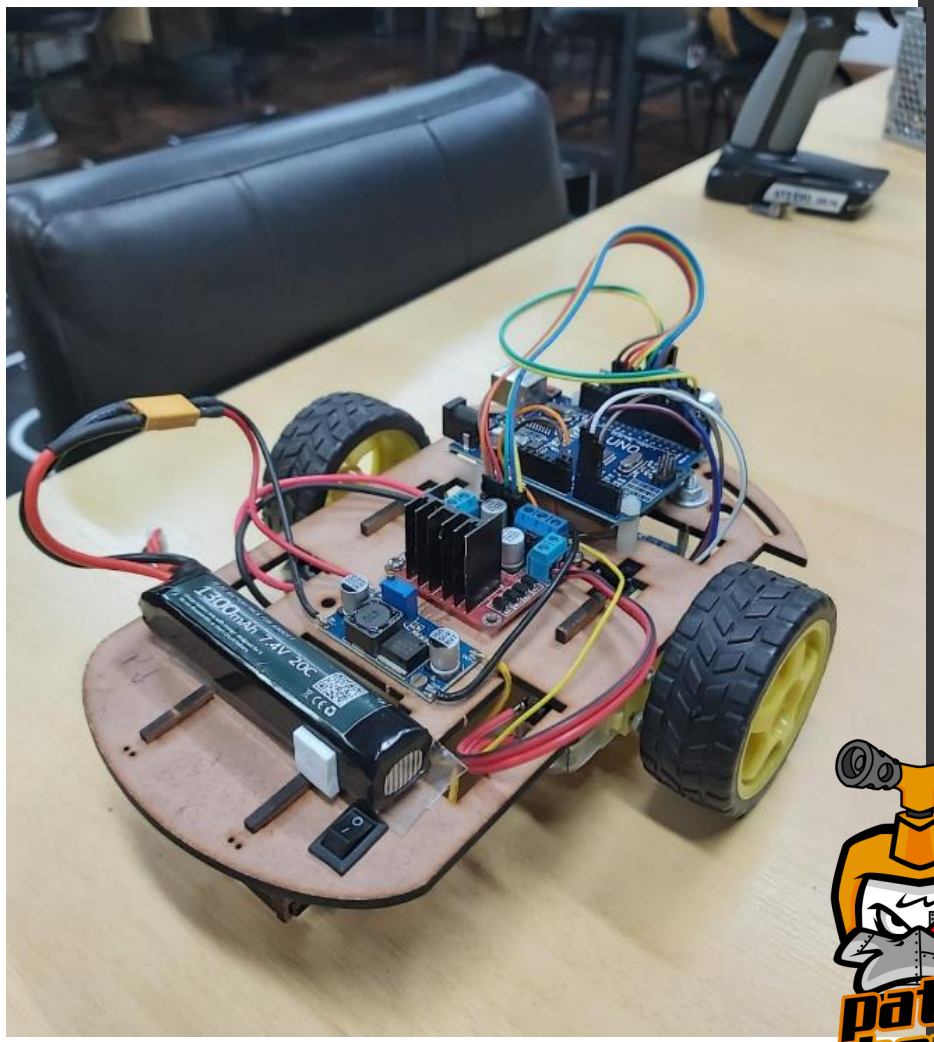
- Modalidades de competições de robótica móvel;
- Motores.
- Acionamento do Motor.
- Ponte H.
- Acionamento da Ponte H.
- Step-Up
- Tipos de Sensores: Sensor Óptico Reflexivo e Sensor de Distância Ultrassônico.
- Leitura dos Sensores.

MODALIDADES



CARRINHO A SER USADO NAS AULAS





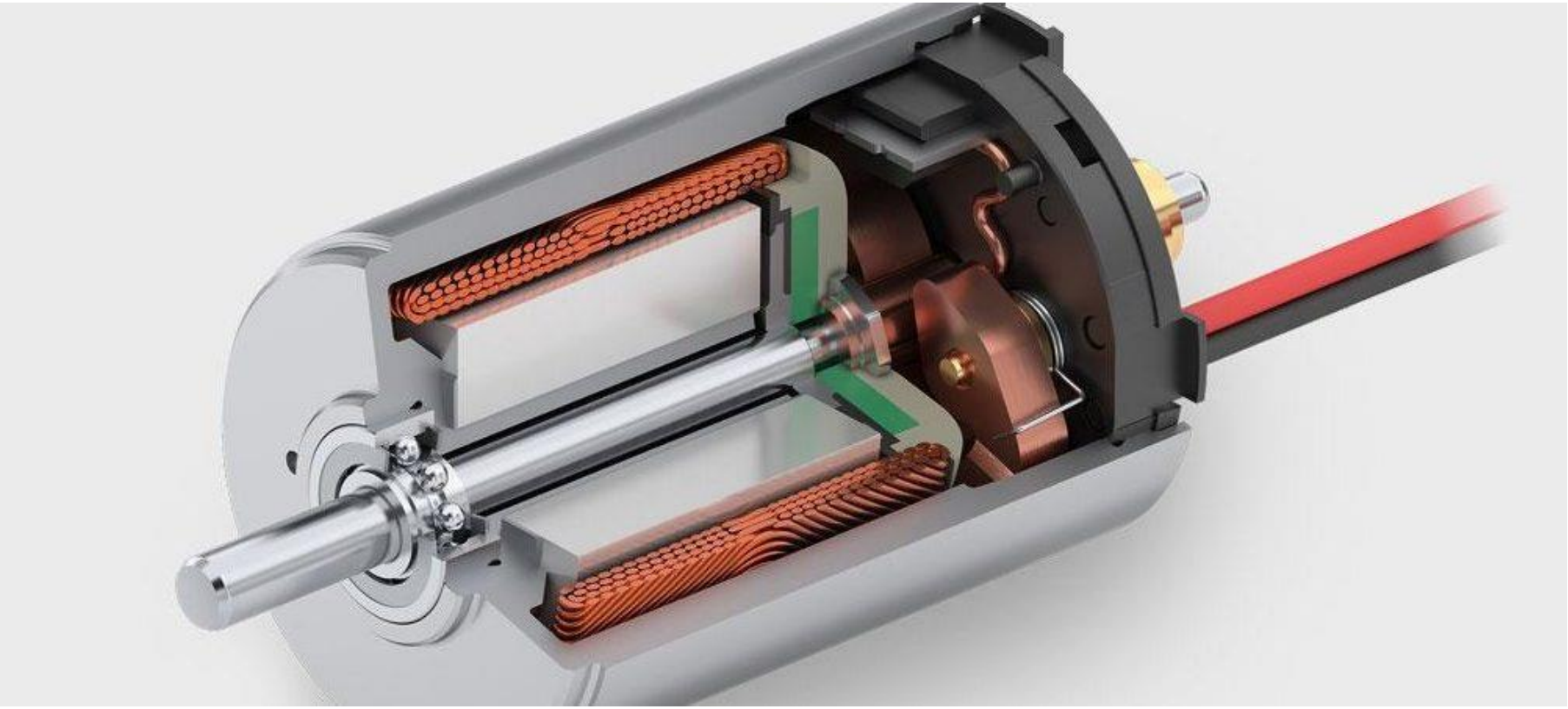
MOTORES

MOTOR DC (Corrente Contínua)

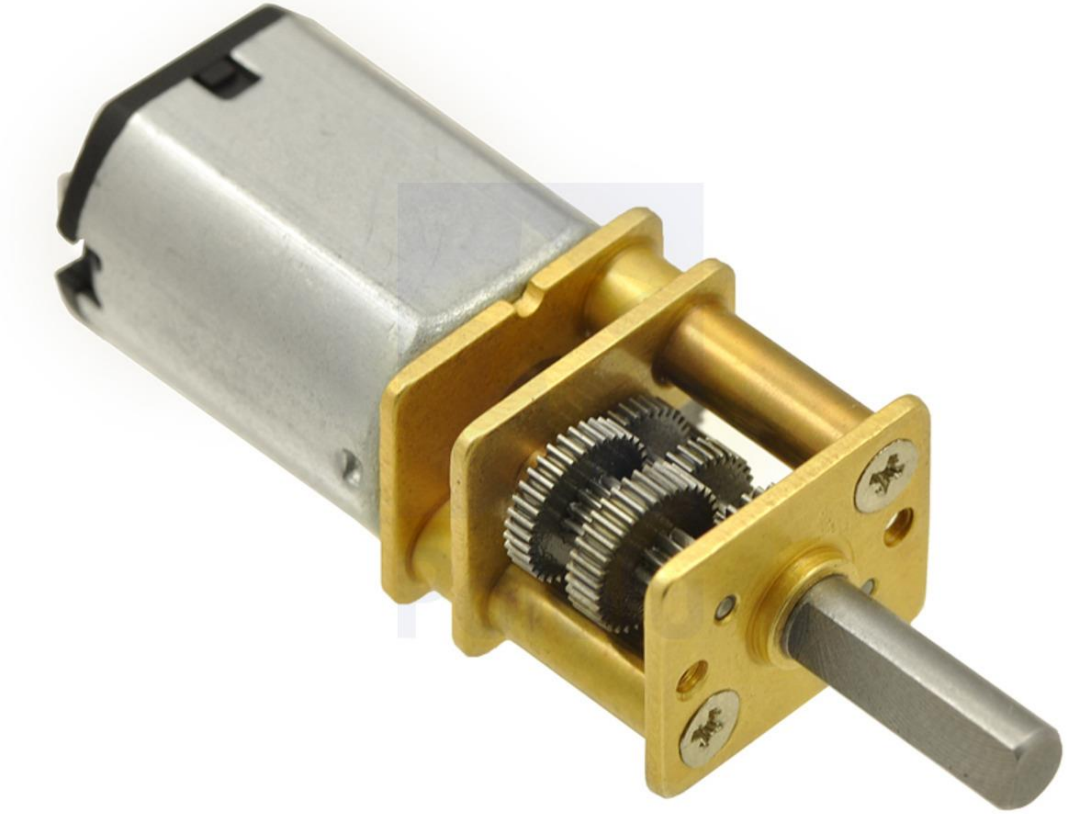
- Dispositivo que transforma energia elétrica em energia rotacional mecânica.

- Teste de Giro invertendo polaridade

MOTOR DC (Corrente Contínua)

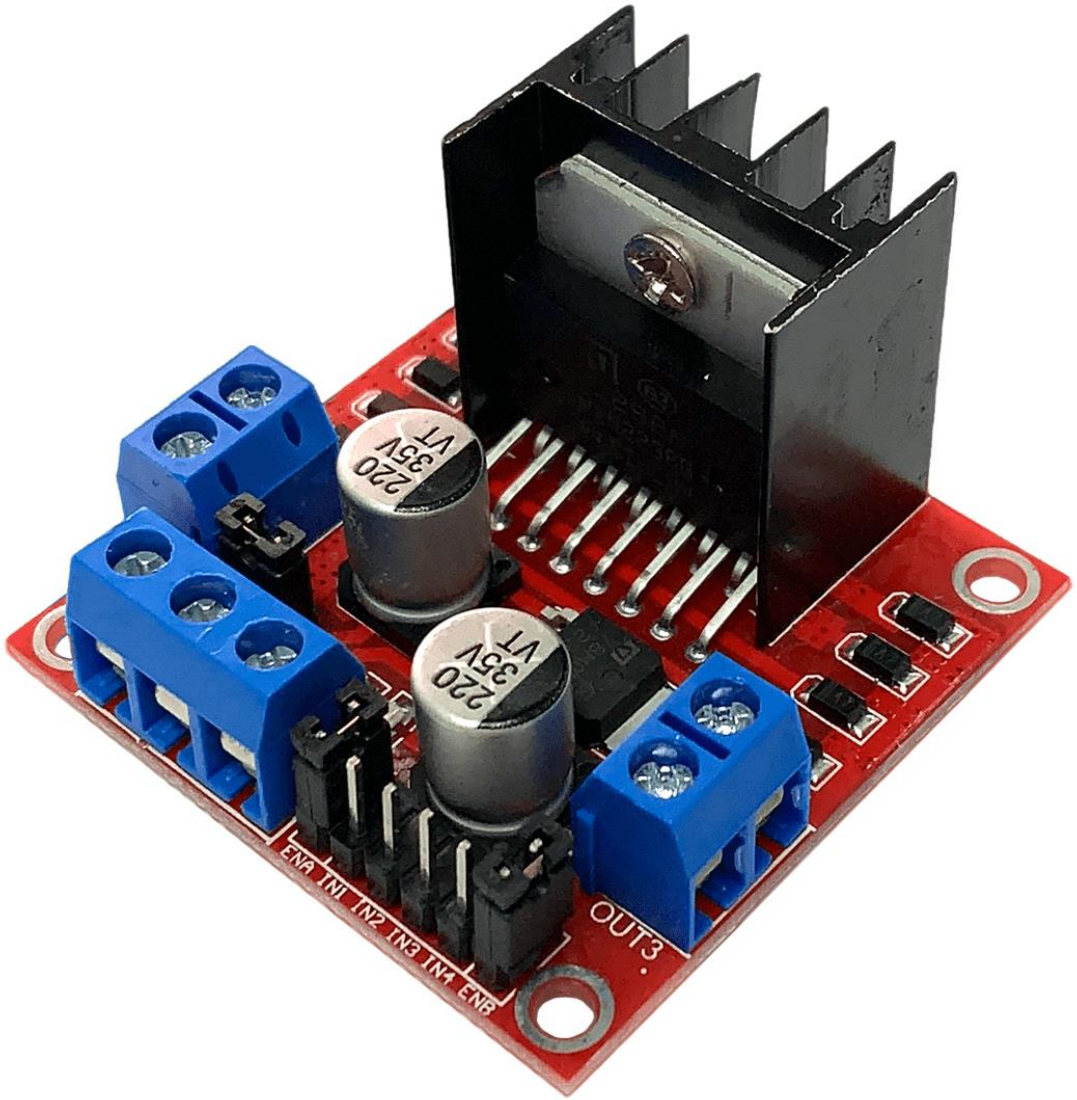


MOTOR DC (Corrente Contínua)



www.pololu.com

PONTE H

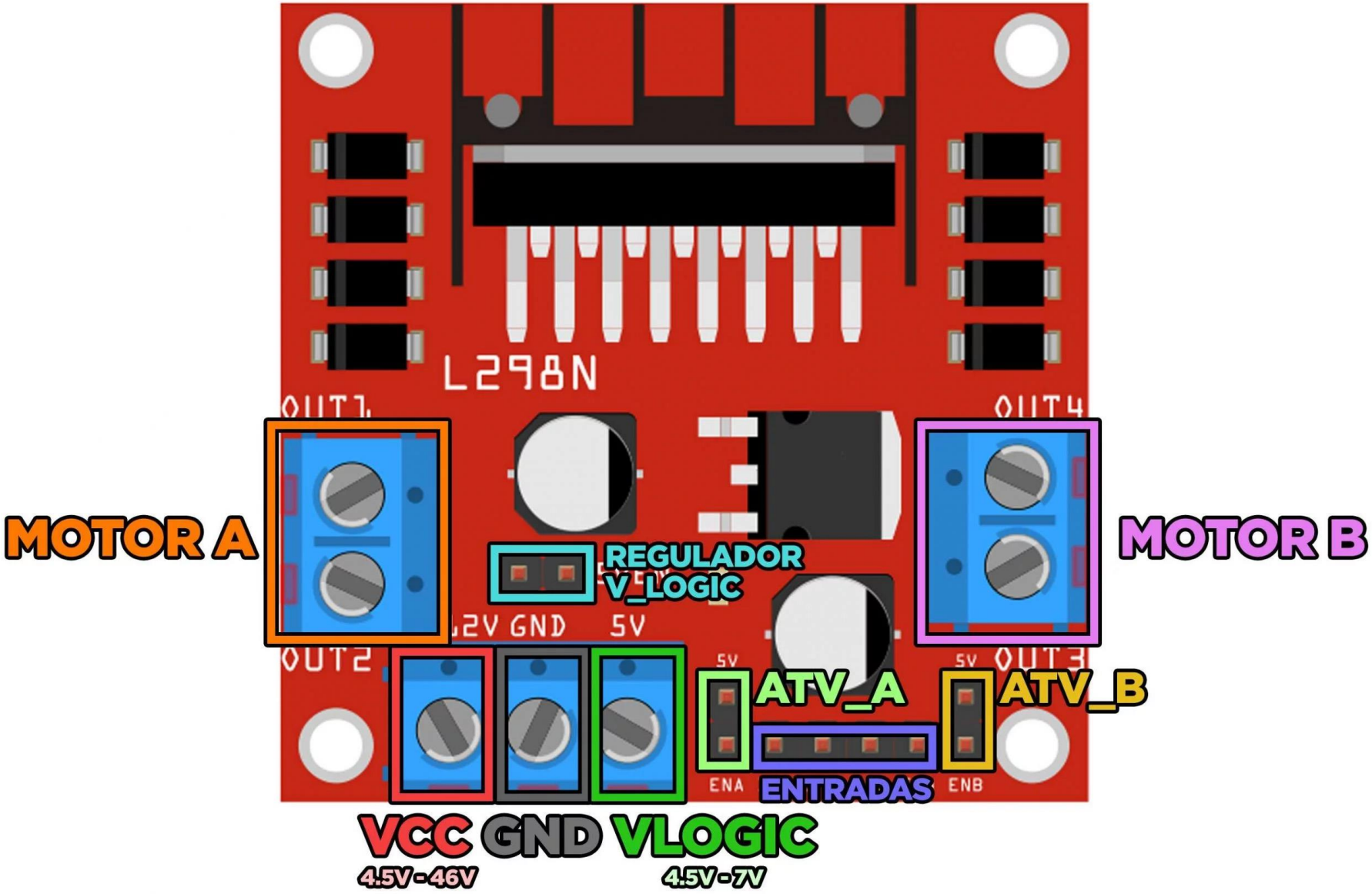

220 35V
220 35V
ENA IN1 IN2 IN3 IN4 ENB
OUT3.

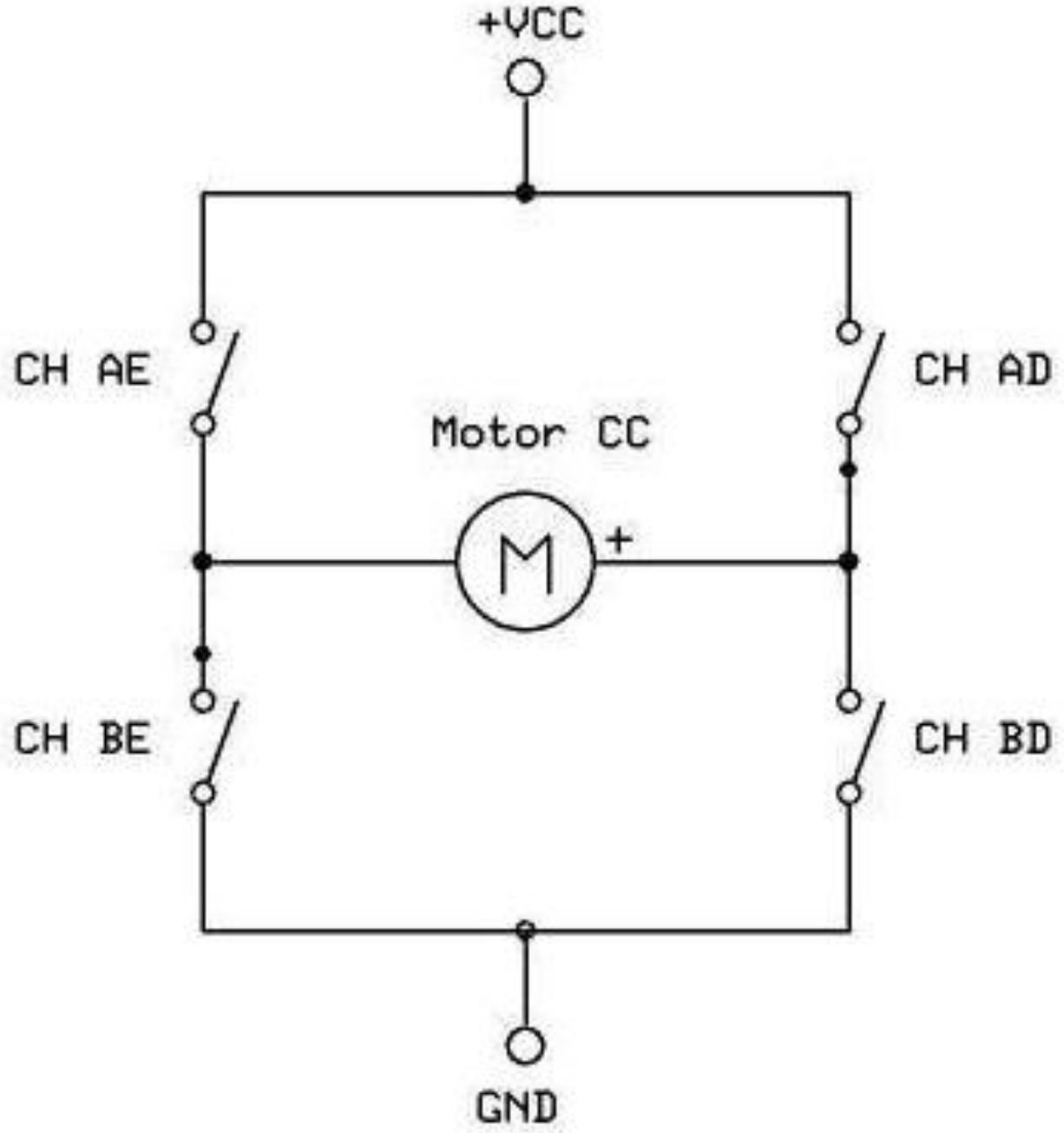
PONTE H

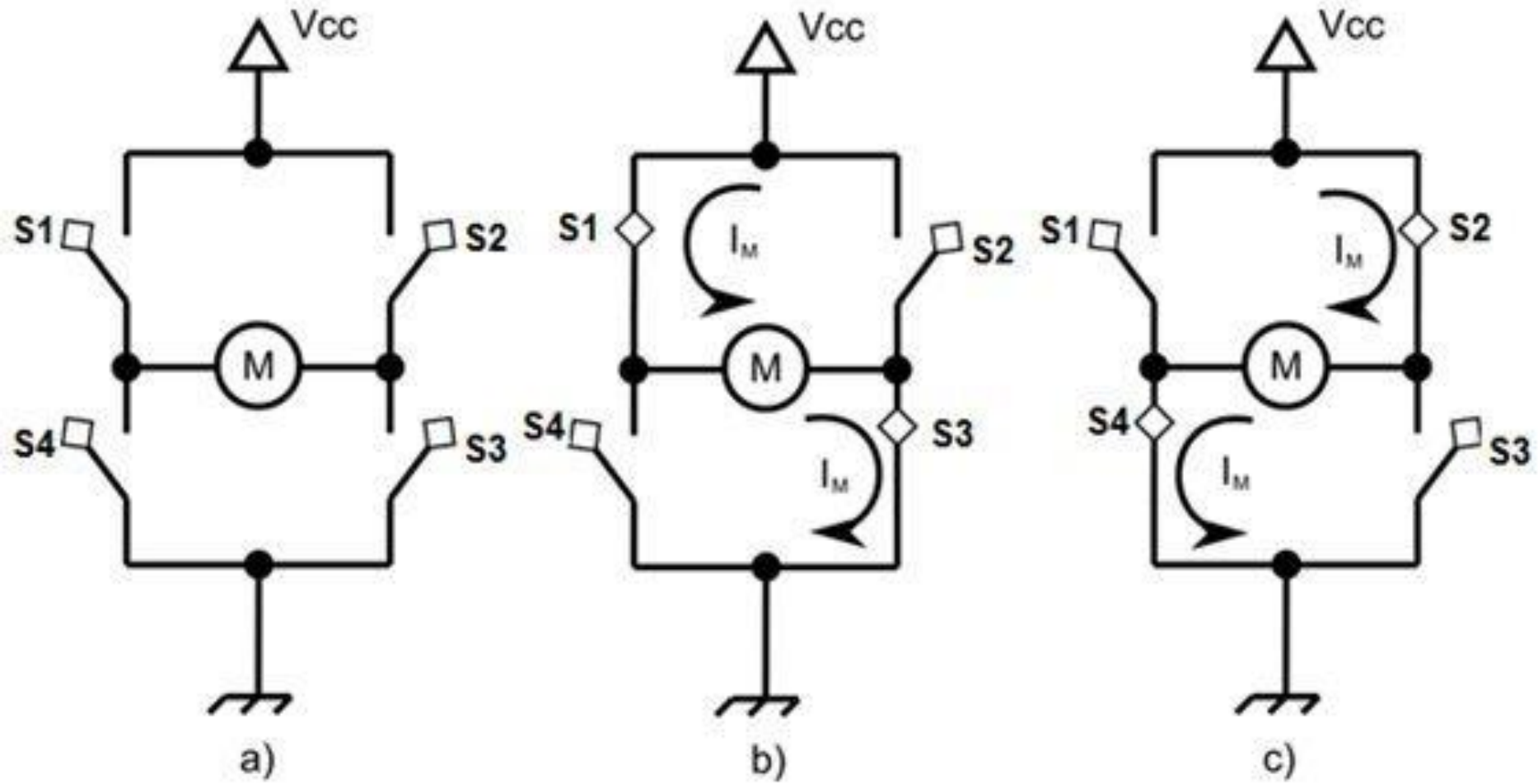
- Permite rotação do motor para ambos os lados.
- Utiliza comutação de chaves para variar estados de rotação.
- No módulo Ponte H L298N, a comutação é eletrônica via um chip embarcado.
- Diferencia-se pela comutação eletrônica em comparação com a mecânica.
- O chip na placa realiza a comutação de chaves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

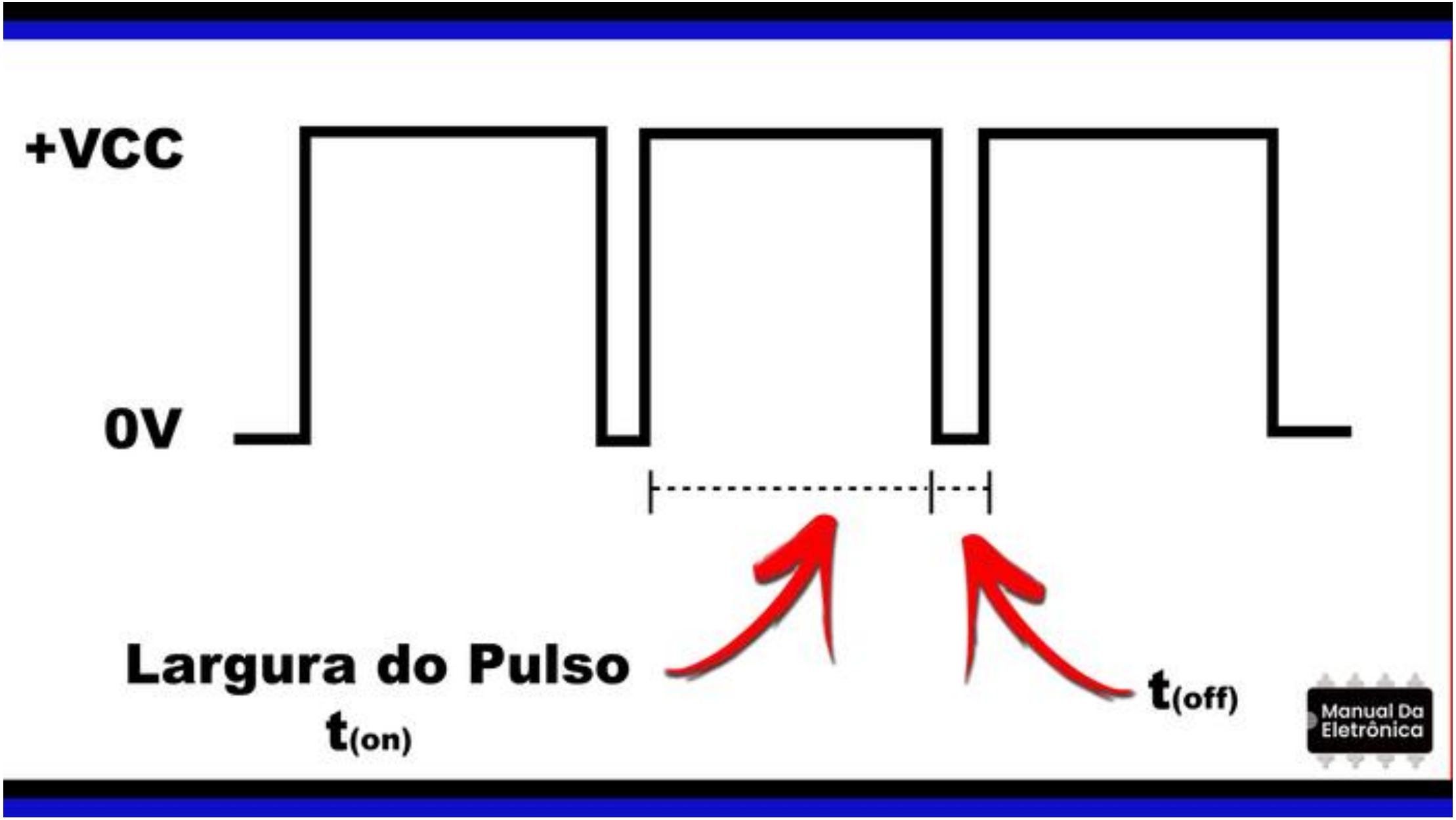
PONTE H

- 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 Vcc pode resultar em danos ao motor e possíveis acidentes.


L298N
OUT1
OUT4
OUT2
OUT3
MOTOR A
MOTOR B
REGULADOR
V_LOGIC
12V GND 5V
5V
5V
ATV_A
ATV_B
ENA
ENTRADAS
ENB
VCC GND VLOGIC
4.5V - 46V
4.5V - 7V

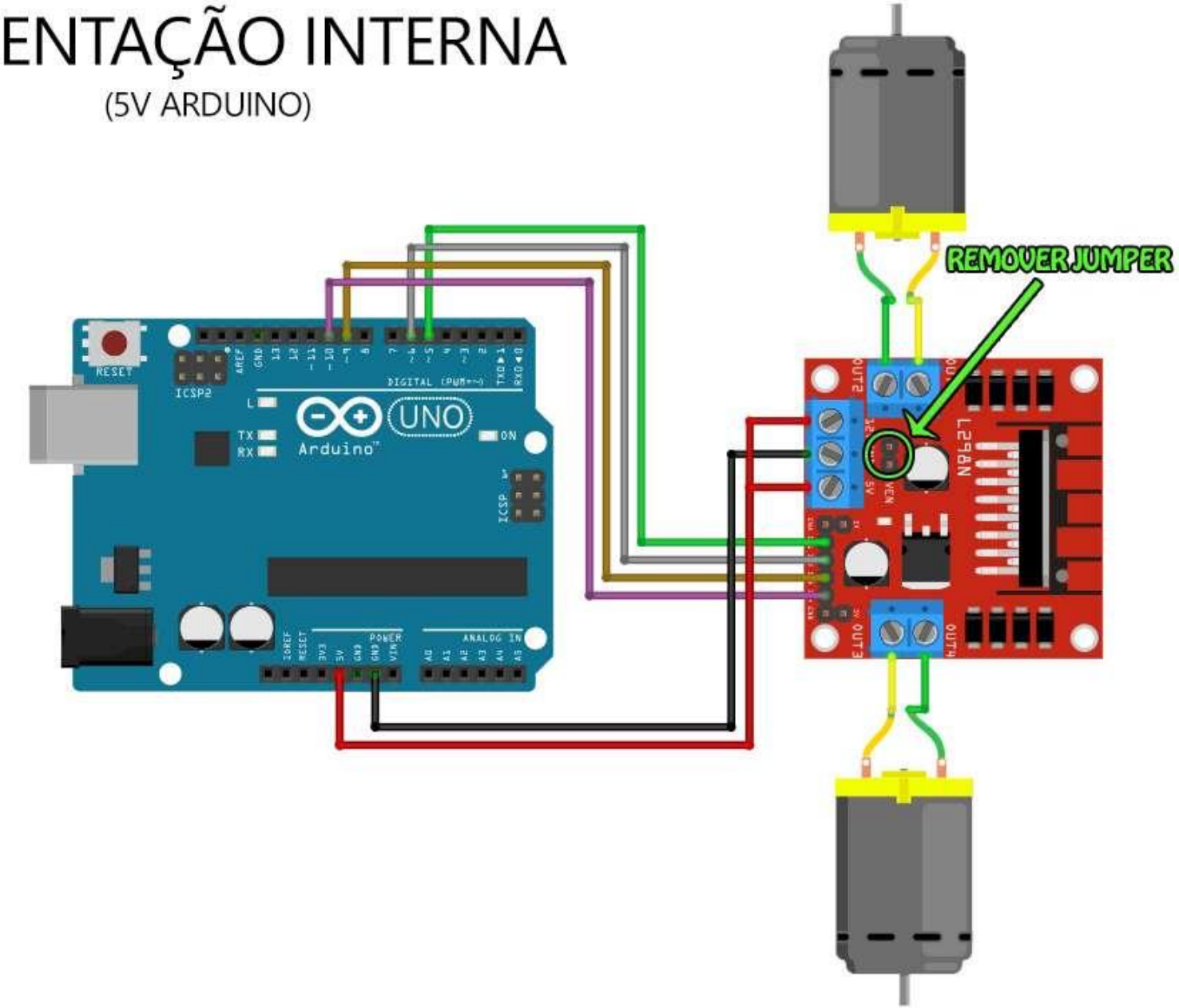

+VCC
CH AE
CH AD
Motor CC
M
+
CH BE
CH BD
GND


Vcc
S1
S2
M
S4
S3
a)
Vcc
S1
S2
I_M
M
S4
S3
I_M
b)
Vcc
S1
S2
I_M
M
S4
S3
I_M
c)


+VCC
0V
Largura do Pulso
t(on)
t(off)
Manual Da Eletrônica

ALIMENTAÇÃO INTERNA

(5V ARDUINO)



```
/* Iremos fazer uma classe para facilitar o uso da ponte H L298N
na manipulação dos motores na função Setup e Loop.*/

class DCMotor {
  int spd = 255, pin1, pin2;

  public:

    void Pinout(int in1, int in2){ /* Pinout é o método para a declaração
      dos pinos que vão controlar o objeto motor*/
      pin1 = in1;
      pin2 = in2;
      pinMode(pin1, OUTPUT);
      pinMode(pin2, OUTPUT);
      }
    void Speed(int in1){ /* Speed é o método que irá ser responsável
      por salvar a velocidade de atuação do motor*/
      spd = in1;
      }
    void Forward(){ // Forward é o método para fazer o motor girar para frente
      analogWrite(pin1, spd);
      digitalWrite(pin2, LOW);
      }
    void Backward(){ // Backward é o método para fazer o motor girar para trás
      digitalWrite(pin1, LOW);
      analogWrite(pin2, spd);
      }
```

```
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  }
```

```
}

void loop() {
  Motor1.Speed(200); /* A velocidade do motor pode variar de 0 a 255,
  onde 255 é a velocidade máxima.*/
  Motor2.Speed(200);

  Motor1.Forward(); // Comando para o motor ir para frente
  Motor2.Forward();
  delay(1000);
  Motor1.Backward(); // Comando para o motor ir para trás
  Motor2.Backward();
  delay(1000);
  Motor1.Stop(); // Comando para o motor parar
  Motor2.Stop();
  delay(500);
}
```

REGULADOR DE TENSÃO STEP-UP

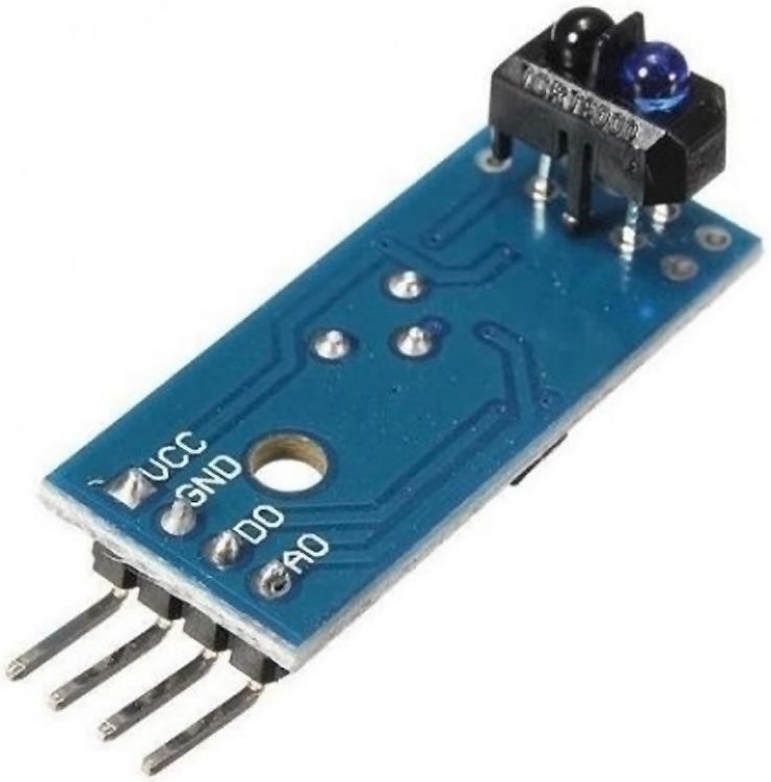
REGULADOR DE TENSÃO STEP-UP

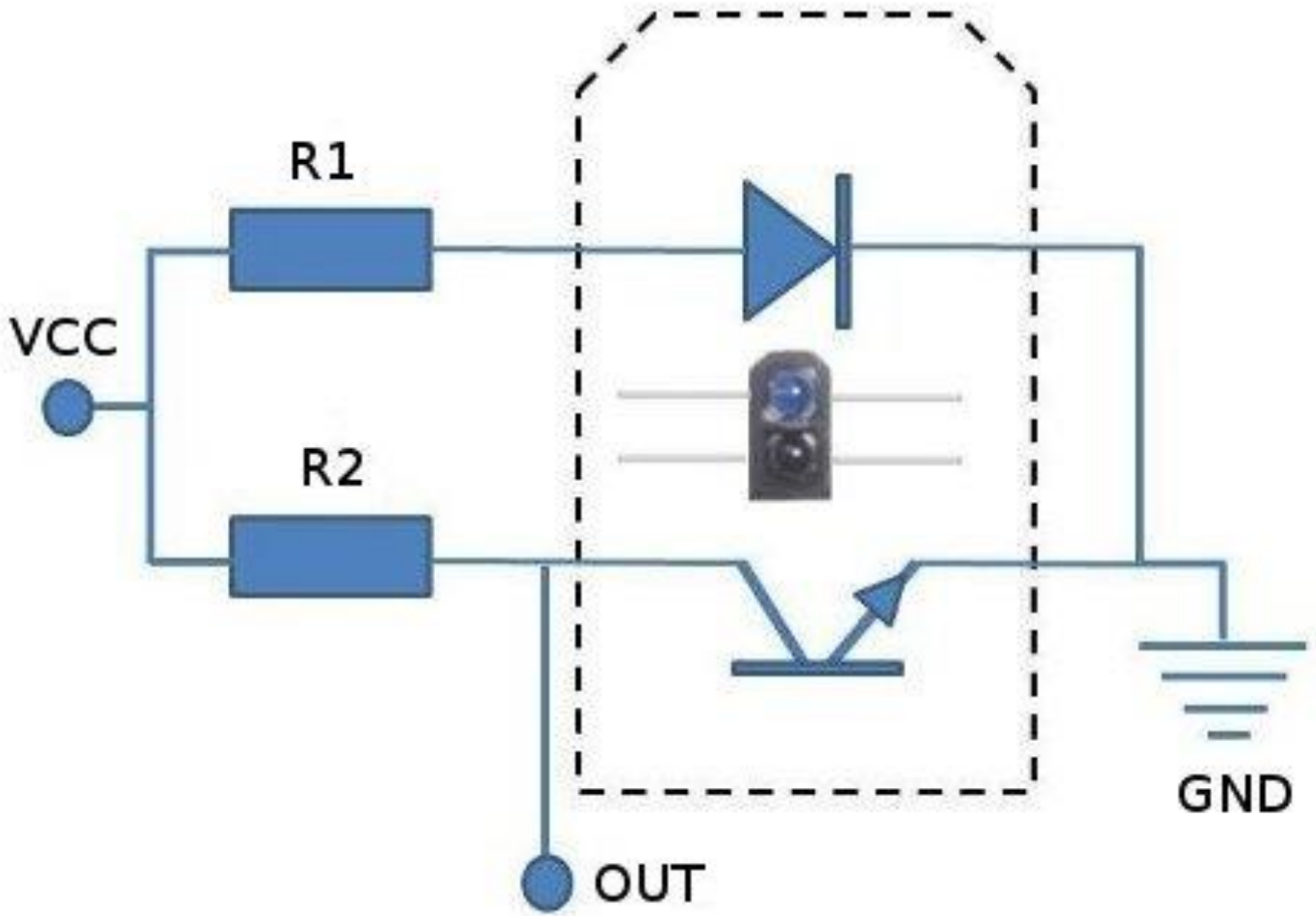
REGULADOR DE TENSÃO STEP-UP

- Dispositivo que tem como base semicondutores, tais como diodos e circuitos integrados, que tem a função de aumentar ou diminuir a tensão de saída de um circuito elétrico.
- Um regulador de tensão é incapaz de gerar energia.

SENSOR ÓPTICO REFLEXIVO

SENSOR ÓPTICO REFLEXIVO




R1
VCC
R2
OUT
GND

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 maior 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.

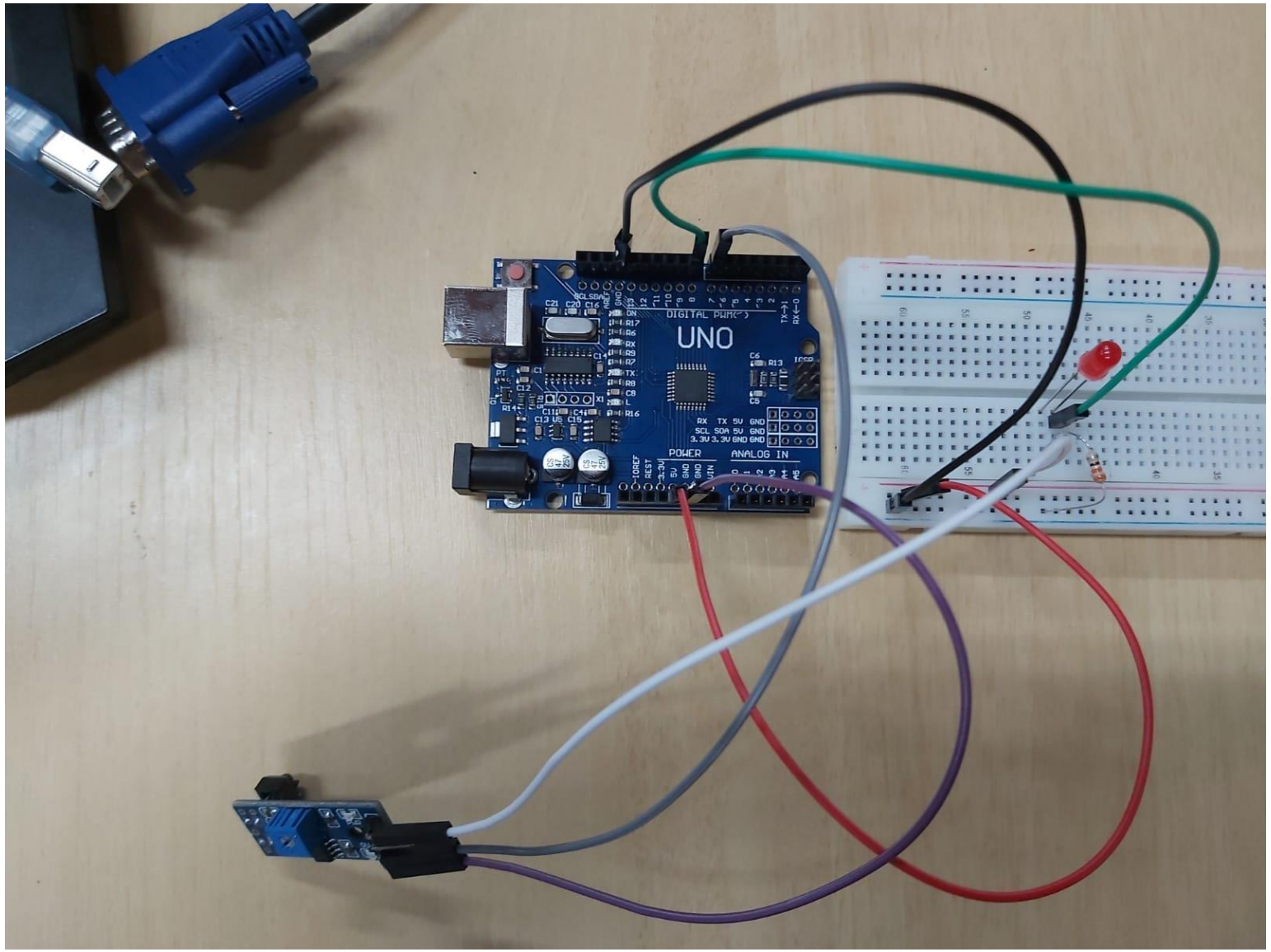
SENSOR ÓPTICO REFLEXIVO

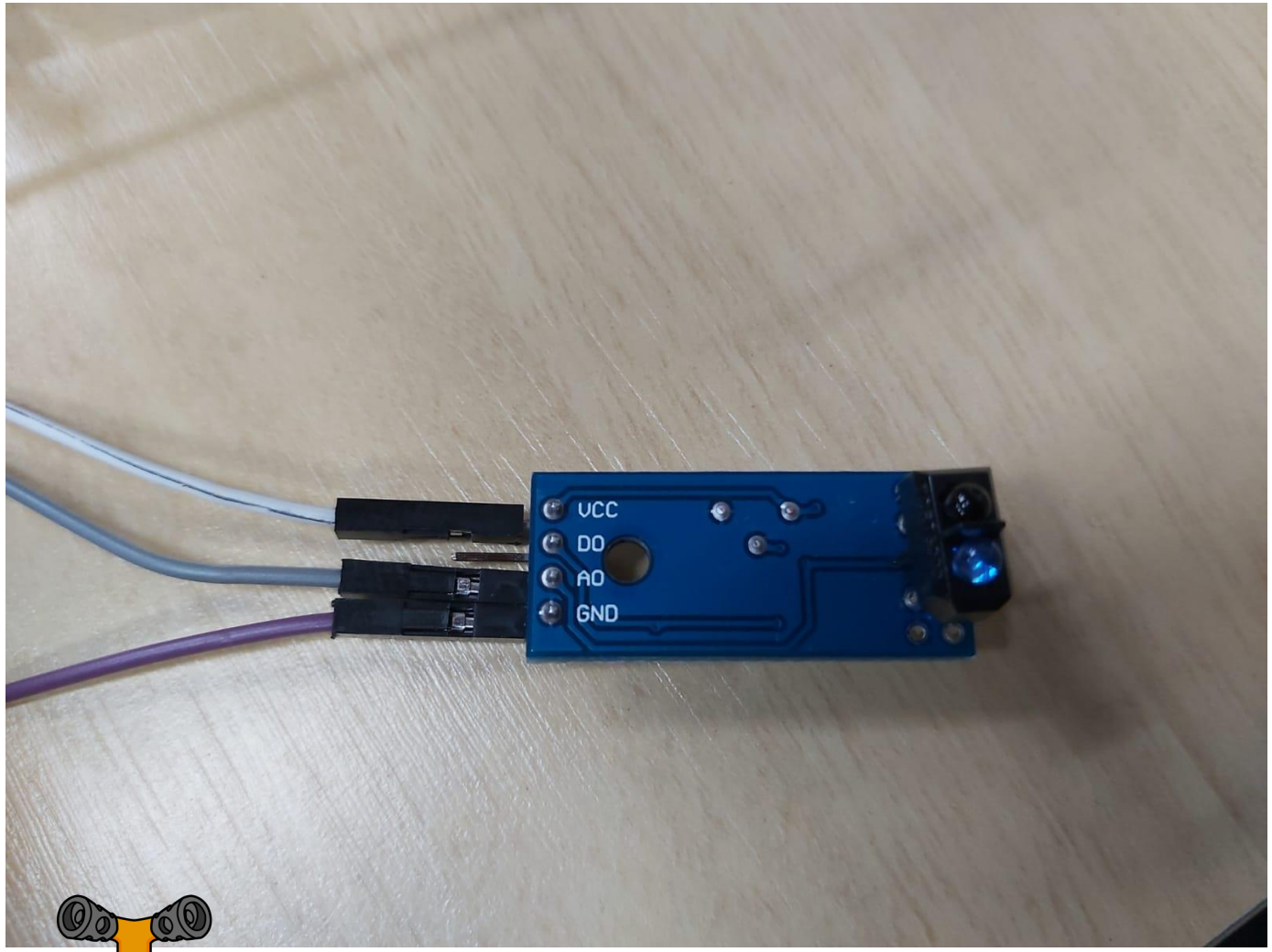
- Ele tem dois modos de leitura: Analógico e Digital;
- O Modo Analógico vai enviar a quantidade de reflexão que o sensor está lendo;
- O Modo Digital vai enviar se o sensor está lendo a reflexão ou não;

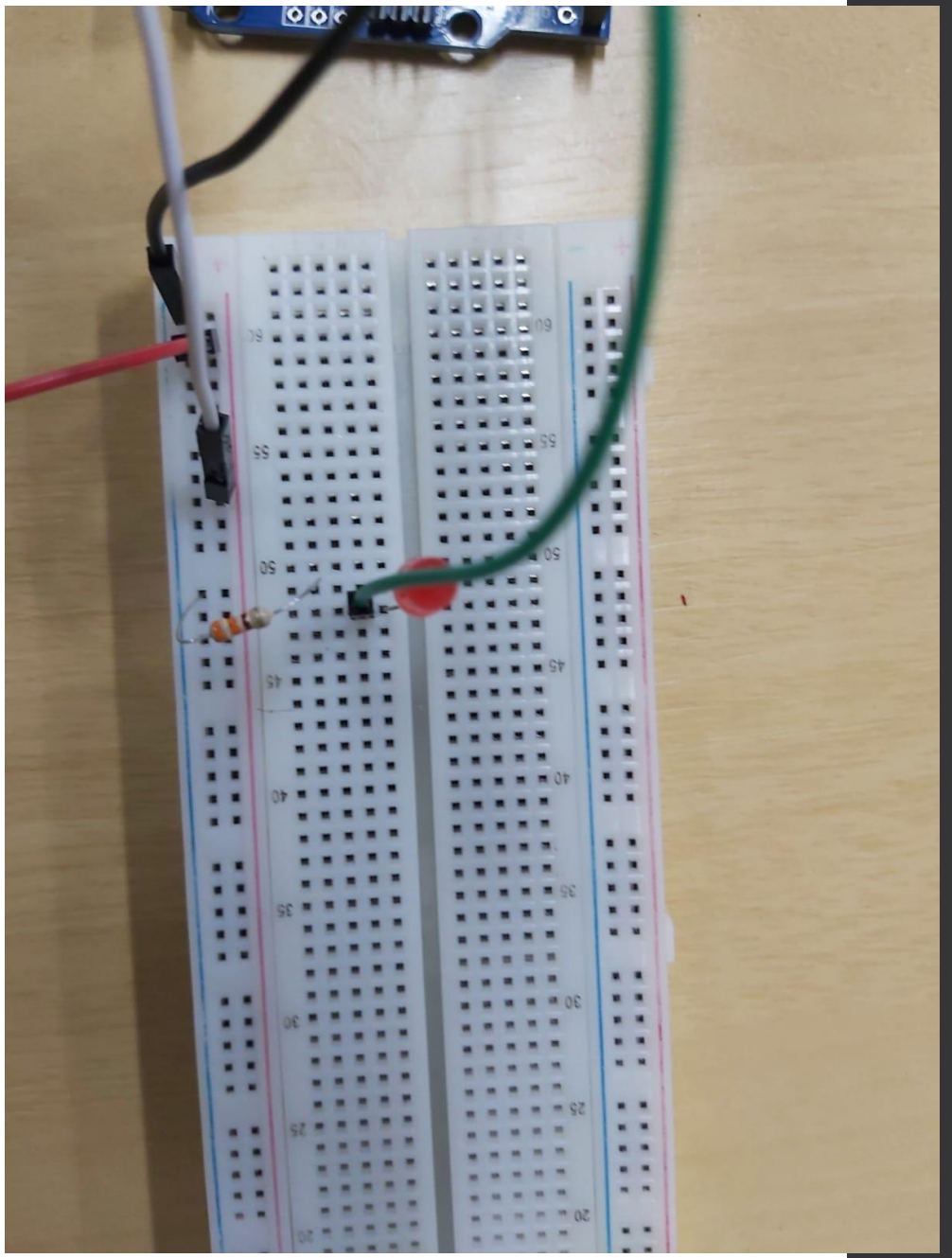
SENSOR ÓPTICO REFLEXIVO

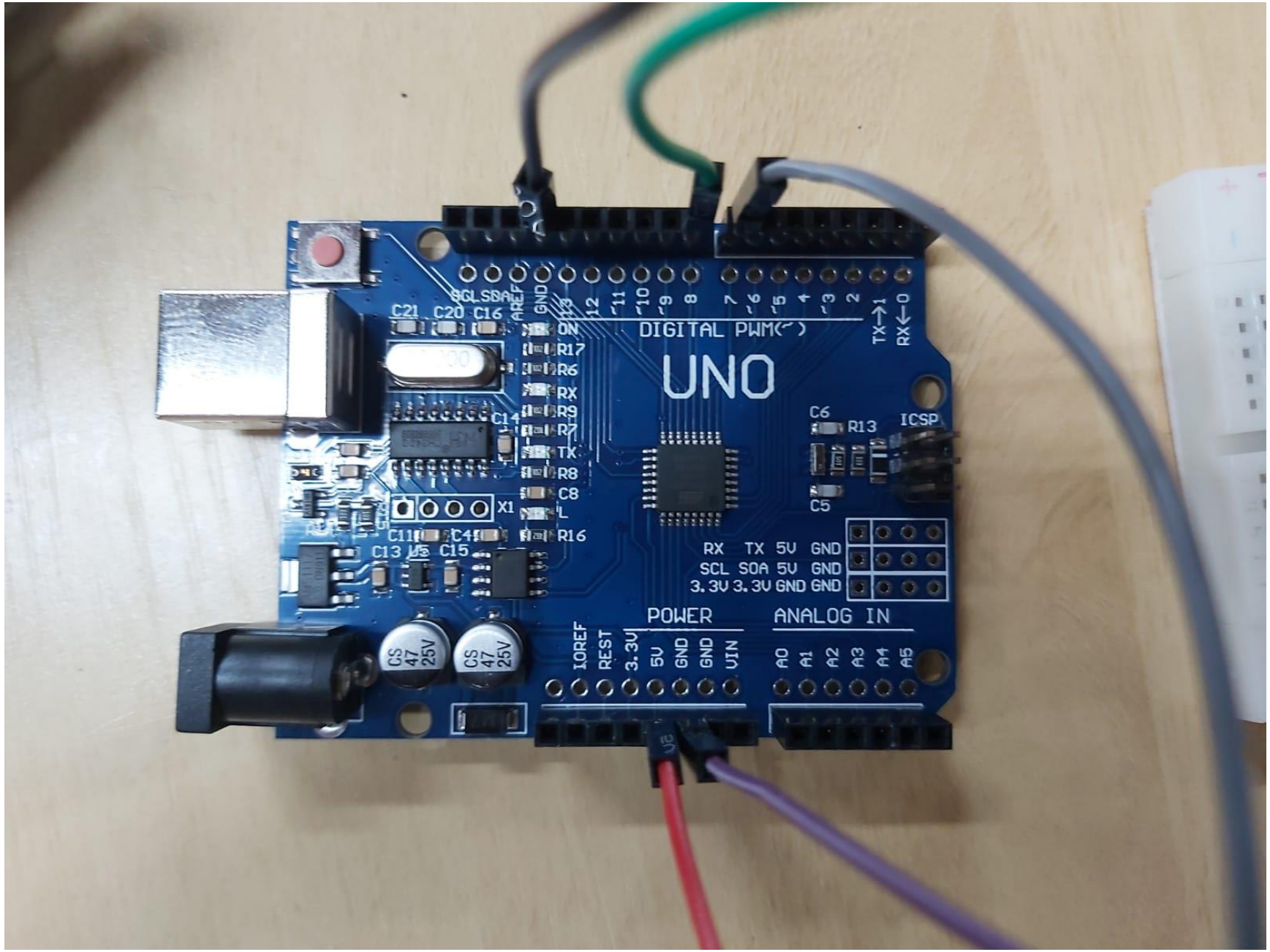
- Este sensor óptico reflexivo está soldado em uma placa que faz aumentar a distância qual o sensor consegue refletir seu infravermelho, também acertando uma maior precisão.
- Na parte de cima da placa tem um potenciômetro que ajusta a precisão da leitura do sensor, regulando o quanto de sinal ele envia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

EXEMPLO DE LEITURA DE SENSOR ÓPTICO REFLEXIVO: DIGITAL



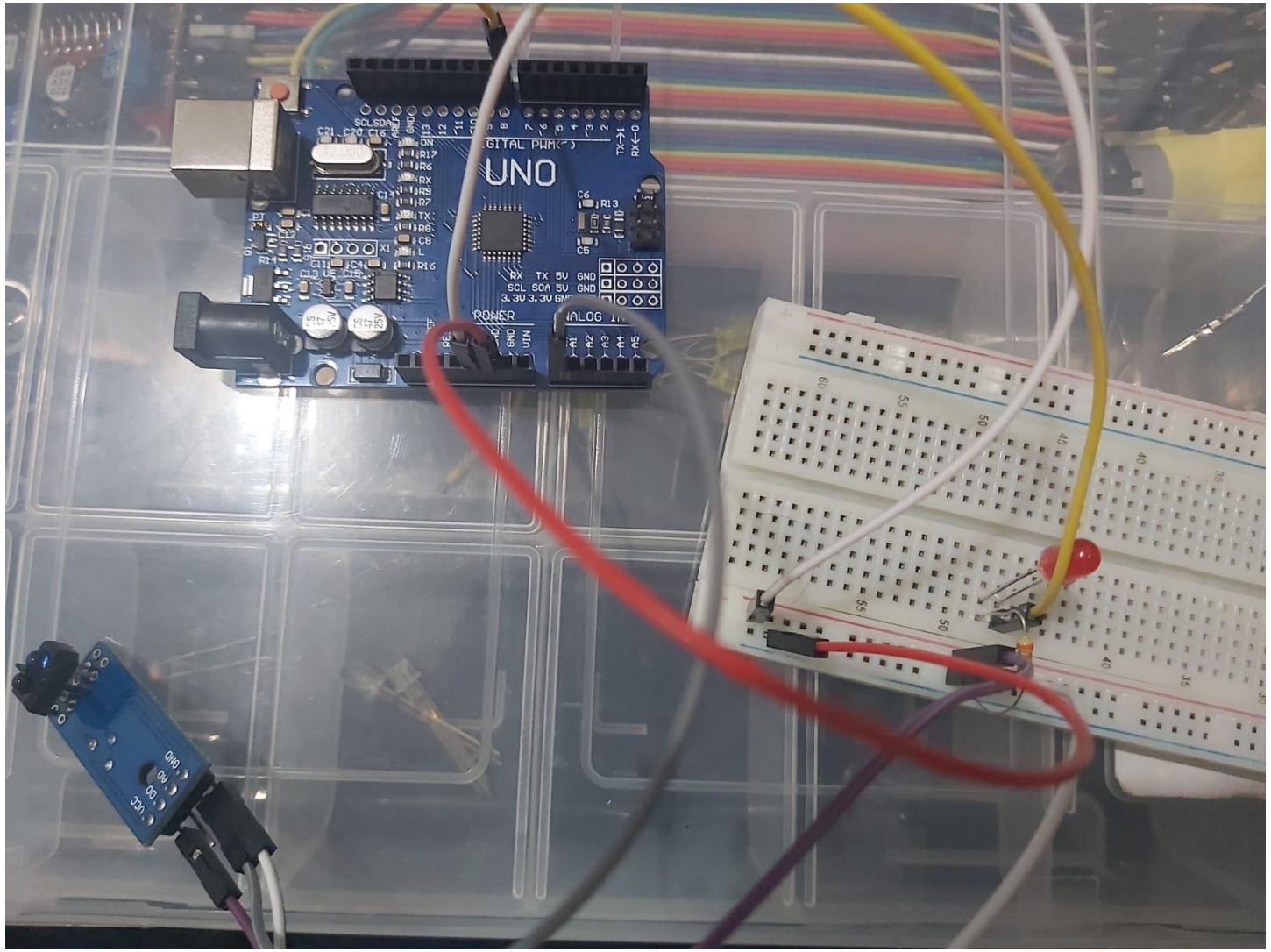

VCC
DO
AO
GND

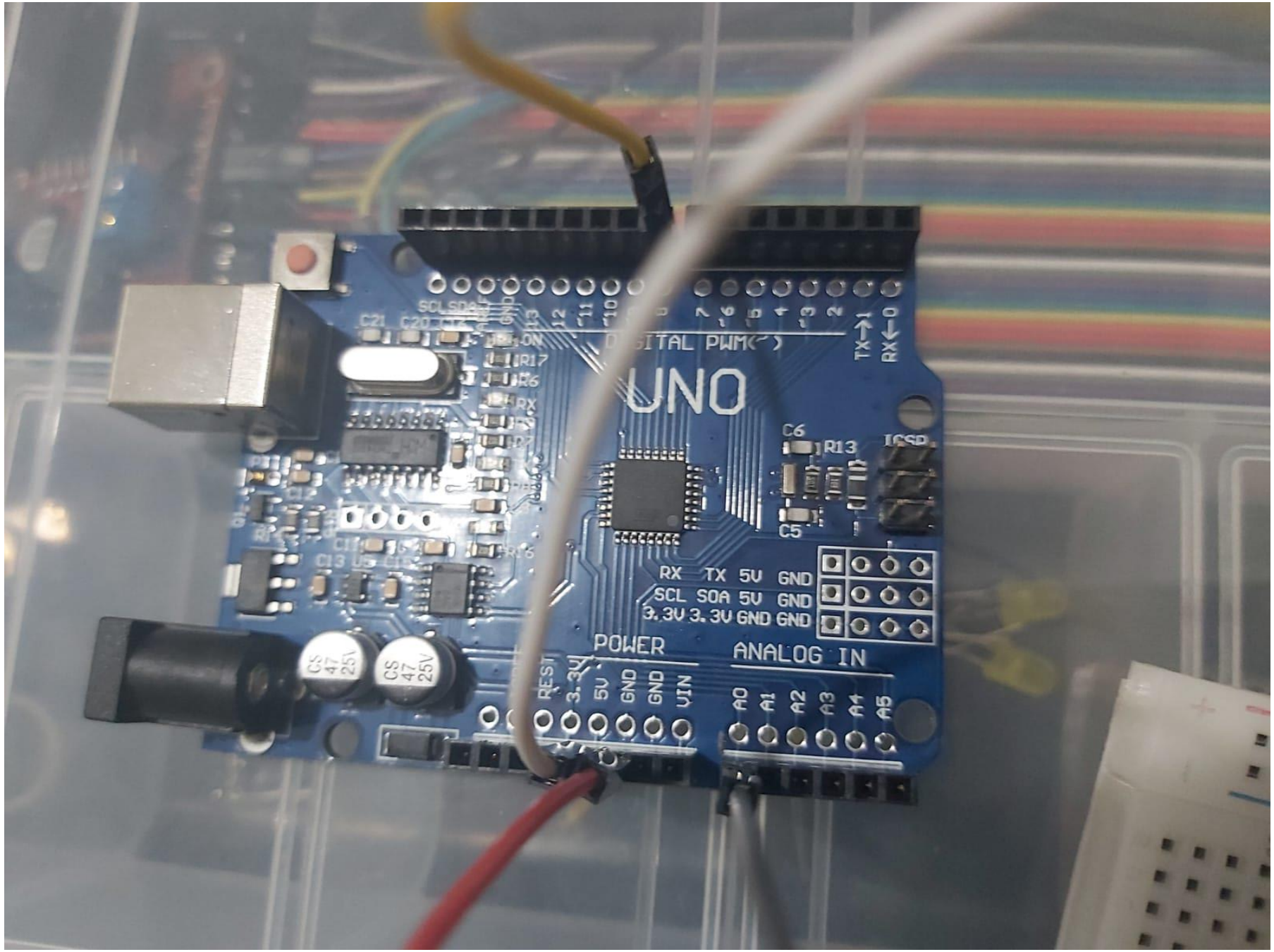



UNO
DIGITAL PWM(~)
POWER
ANALOG IN
ICSP
RX TX 5V GND
SCL SDA 5V GND
3.3V 3.3V GND GND
IOREF
RESET
3.3V
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5
TX→1
RX←0

```
int led = 8; //Pino do led
int sensor = 7;   //Ligado ao pino "coletor" do sensor óptico
int leitura = 0;       //Armazena informações sobre a leitura do sensor

void setup()
{
  pinMode(led, OUTPUT); //Define o pino do led como saida
  pinMode(sensor, INPUT);   //Define o pino do sensor como entrada
}

void loop()
{
  //Le as informações do pino do sensor
  leitura = digitalRead(sensor);

  if (leitura != 1) //Verifica se o objeto foi detectado
  {
    digitalWrite(led, 1);
  }
  else{
    digitalWrite(led, 0);
  }
}
```

EXEMPLO DE LEITURA DE SENSOR ÓPTICO REFLEXIVO: ANALÓGICO




UNO
RX TX 5V GND
SCL SDA 5V GND
3.3V 3.3V GND GND
POWER
ANALOG IN
RESET
3.3V
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5
ICSP
C6
R13
C5

```arduino
int led = 8; //Pino do led
int sensor = 0;   //Ligado ao pino "coletor" do sensor óptico
int leitura;      //Armazena informações sobre a leitura do sensor

void setup()
{
  Serial.begin(9600);
  pinMode(led, OUTPUT); //Define o pino do led como saida
  pinMode(sensor, INPUT);   //Define o pino do sensor como entrada
}

void loop()
{
  //Le as informações do pino do sensor
  leitura = analogRead(sensor);
  Serial.println(leitura);
  if (leitura < 341) //Verifica se o objeto foi detectado
  {
    digitalWrite(led, 1);
  }
  else{
    digitalWrite(led, 0);
  }
}
```

SENSOR DE DISTÂNCIA ULTRASSÔNICO


HC-SR04
Vcc
Trig
Echo
Gnd
T
R

ULTRASSÔNICO

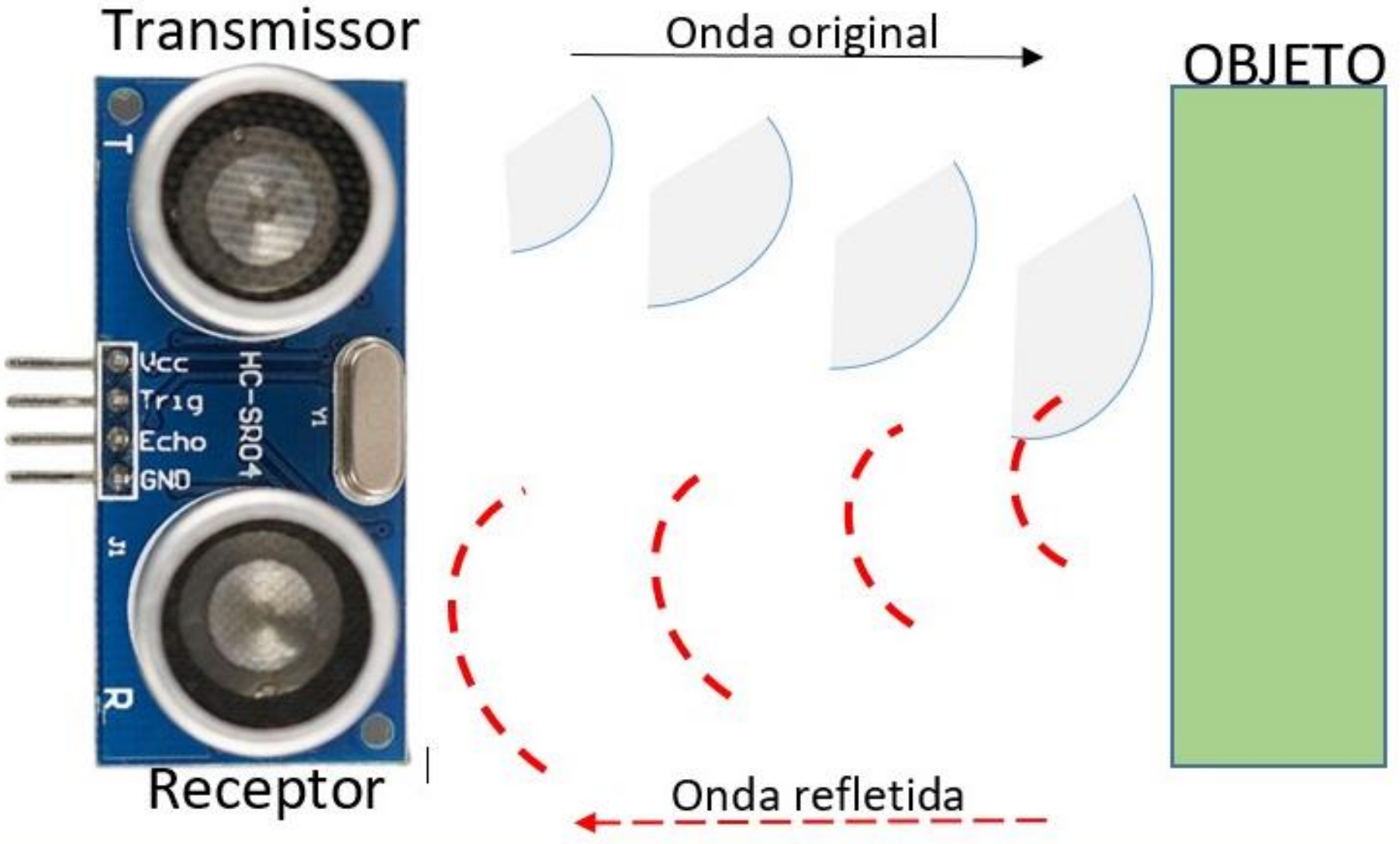
- Emite pulsos de ondas sonoras ultrassônicas.
- Mede o tempo que leva para os pulsos refletirem de um objeto de volta para o sensor.
- Alcance pode variar, geralmente entre 2cm a 4m com com uma precisão de 3 milímetros.
- Sensível a condições ambientais, como temperatura e umidade.

ULTRASSÔNICO

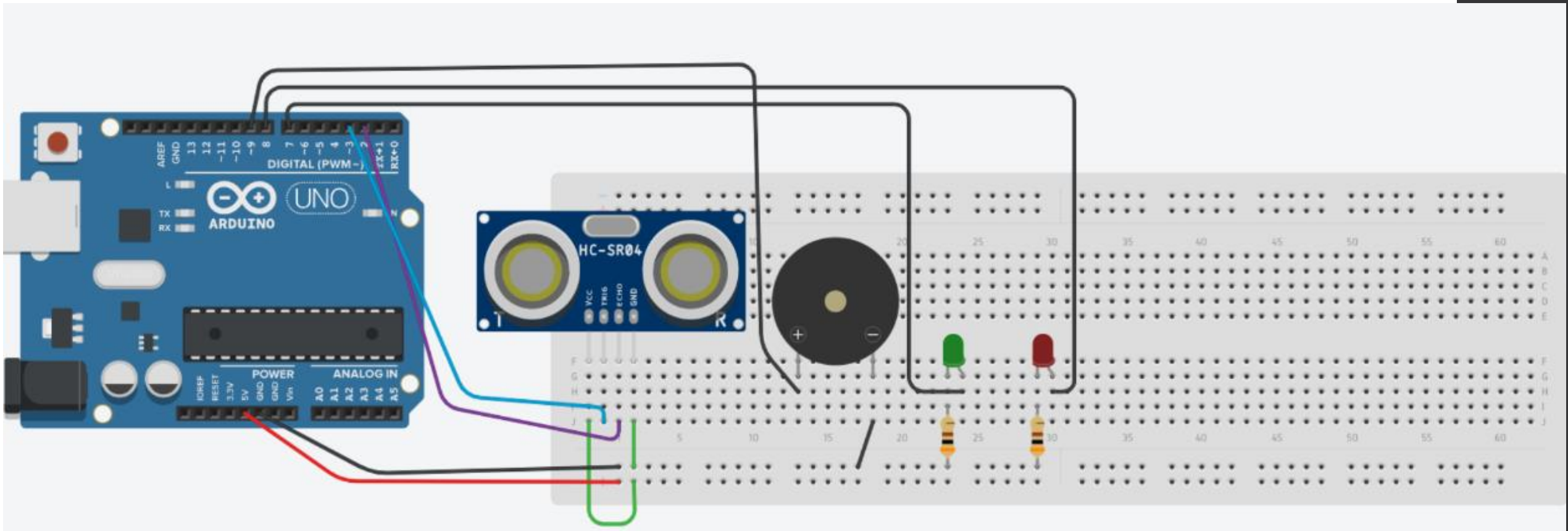
- Não é afetado por cores ou texturas do objeto.
- Pode ser usado em diversas condições de iluminação.
- Pode ser influenciado por obstáculos que absorvem ou dispersam as ondas sonoras.
- Precisão pode diminuir em distâncias muito curtas.

ULTRASSÔNICO

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


Transmissor
Onda original
OBJETO
T
Vcc
Trig
Echo
GND
HC-SR04
Y1
J1
R
Receptor
Onda refletida

EXEMPLO DE LEITURA DE SENSOR ULTRASSONICO



```
//Configuração de Distancia Mínima em centimetros
const int distancia_carro = 10;

//Configurações de Portas do Arduino

//Sensor
const int TRIG = 3;
const int ECHO = 2;

//Demais componentes
const int ledGreen = 7;
const int ledRed = 8;
const int buzzer = 9;

// Variaveis para funcionamento do Buzzer
float seno;
int frequencia;

void setup() {
  Serial.begin(9600);

  // Configurações do Sensor
  pinMode(TRIG,OUTPUT);
  pinMode(ECHO,INPUT);

  // Configurações do LED
  pinMode(ledGreen, OUTPUT);
```

```
  // Configurações do LED
  pinMode(ledGreen, OUTPUT);
  pinMode(ledRed, OUTPUT);

  //Configurações do Buzzer
  pinMode(buzzer, OUTPUT);

}

void loop() {
  int distancia = sensor_morcego(TRIG,ECHO);

    if(distancia <= distancia_carro){
      Serial.print("Atenção: ");
      Serial.print(distancia);
      Serial.println("cm");
      digitalWrite(ledGreen, LOW);
      digitalWrite(ledRed, HIGH);
      tocaBuzzer();
    }
    else{
      Serial.print("Livre: ");
      Serial.print(distancia);
      Serial.println("cm");
      digitalWrite(ledGreen, HIGH);
      digitalWrite(ledRed, LOW);
```

```
        digitalWrite(ledRed, LOW);
        noTone(buzzer);
      }
  delay(100);

}

int sensor_morcego(int pinotrig,int pinoecho){
  digitalWrite(pinotrig,LOW);
  delayMicroseconds(2);
  digitalWrite(pinotrig,HIGH);
  delayMicroseconds(10);
  digitalWrite(pinotrig,LOW);

  return pulseIn(pinoecho,HIGH)/58;
}

//Função para execução do Alarme Sonoro
void tocaBuzzer(){
 for(int x=0;x<180;x++){
  seno=(sin(x*3.1416/180));
  frequencia = 2000+(int(seno*1000));
  tone(buzzer,frequencia);
  delay(2);
}

}
```