Robótica III

Professores

Antonio Fernando Traina – Professor da FATEC – Franca

Doutor em Física Aplica Computacional - IFSC-USP,

aftraina@gmail.com

Roseli Aparecida Romero – Coordenadora do Curso

Professora ICMC-USP,

rafrance@icmc.usp.br

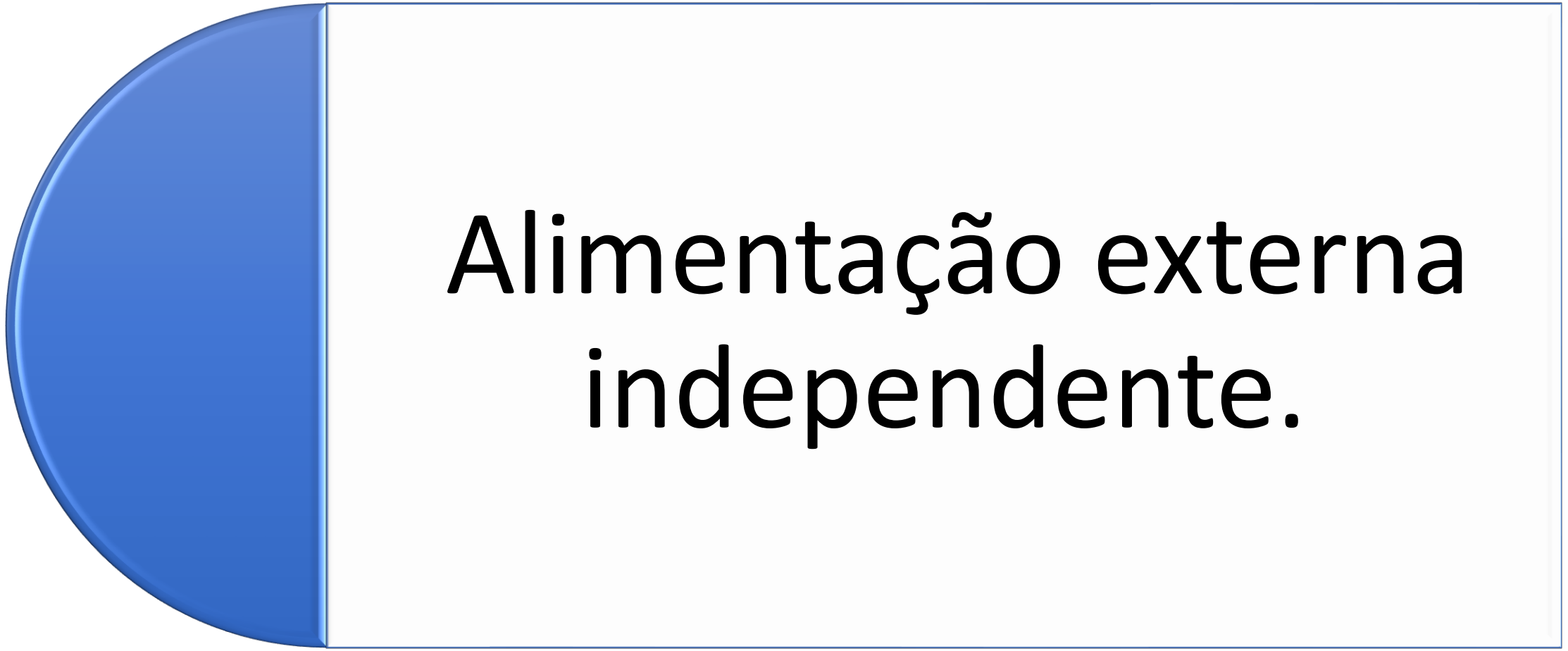
abril -2019

Robótica III
Aula Extra - Dicas

abril -13/2019

Dicas

Motor



Programação

Baseado no programa criado por Michael McRoberts e disponível no livro **Arduíno Básico:**

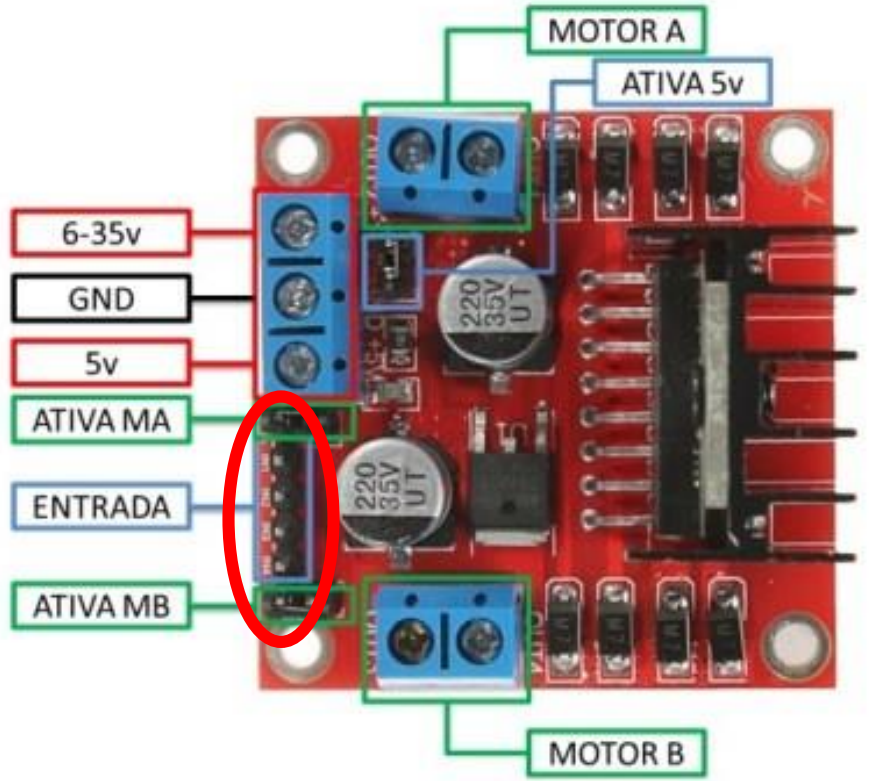
- Usar as funções da **biblioteca AFMotor**, responsável por comandar o motor shield.
- Baixar a biblioteca AFMotor do site da disciplina.
- Descompacte a pasta,
- renomeie para **AFMotor**,
- coloque essa pasta dentro da pasta **LIBRARIES** do programa (IDE) do seu Arduino.
- Sair e carregar a IDE novamente para que a biblioteca seja reconhecida pelo programa.

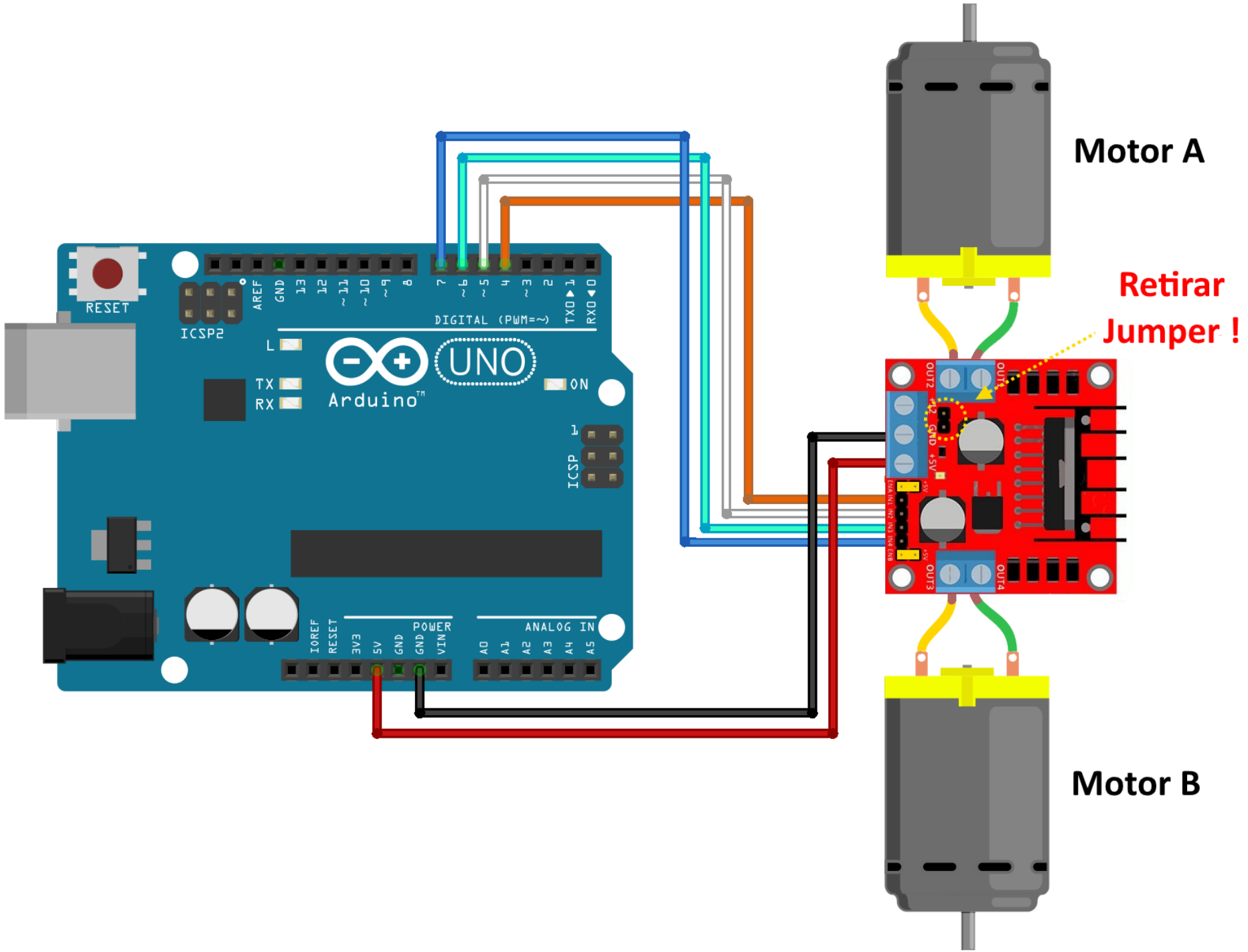
https://github.com/adafruit/Adafruit-Motor-Shield-library/zipball/master

IN 1, 2,3, 4

IN1 e IN2:são utilizados para controlar o sentido do motor A;

IN3 e IN4: são utilizados para controlar o sentido do motor B;




Motor A
Retirar
Jumper !
Motor B
RESET
ICSP2
DIGITAL (PWM=~)
UNO
Arduino™
ON
TX
RX
ICSP
POWER
ANALOG IN
OUT1
OUT2
OUT3
OUT4
+5V
GND

Código teste 1

Programando Ponte H com Arduino: teste

Teste o seu módulo carregando o programa abaixo, que vai servir para os mostrar o funcionamento do motor com ponte H.

O programa gira o motor A no sentido horário, depois desliga esse motor e gira o motor B no mesmo sentido. Depois, repete esse procedimento no sentido anti-horário.

Definições de variaveis:

```
//Programa : Motor shield com sensor
//TCRT5000    -   Adaptacoes : FILIPEFLOP
//Baseado no programa original de Michael
//McRoberts

#include <AFMotor.h>
 AF_DCMotor motor_esq(1); //Seleciona o
//motor 1

AF_DCMotor motor_dir(4); //Seleciona o motor 4
 int SENSOR1, SENSOR2, SENSOR3;

 //deslocamentos de calibracao
int leftOffset = 0, rightOffset = 0, centre = 0;
```

```
//pinos para a velocidade e direcao do motor
int speed1 = 3, speed2 = 11, direction1 = 12,
direction2 = 13;


//velocidade inicial e deslocamento de rotacao
int startSpeed = 70, rotate = 30;


//limiar do sensor
int threshold = 5;


//velocidades iniciais dos motores esquerdo e direito
int left = startSpeed, right = startSpeed;
```

Código calibrate()

```
//Rotina de calibracao do sensor
void calibrate() {
//Executa 10 vezes para obter uma media
            for (int x=0; x<10; x++) {
                  delay(100);
                   SENSOR1 = analogRead(0);
                   SENSOR2 = analogRead(1);
                   SENSOR3 = analogRead(2);
                   leftOffset = leftOffset + SENSOR1;
                   centre = centre + SENSOR2;
                   rightOffset = rightOffset + SENSOR3;
                   delay(100);
             }
}
```

```
 //obtem a media para cada sensor
 leftOffset = leftOffset /10;
 rightOffset = rightOffset /10;
 centre = centre / 10;

 //calcula os deslocamentos para os sensores
//esquerdo e direito
 leftOffset = centre - leftOffset;
 rightOffset = centre - rightOffset;
```

Código setup()

```
void setup()
{
  calibrate();
  delay(3000);
}
```

Código loop() 1/3

```
void loop() {
 //Gira o Motor A no sentido horario
        digitalWrite(IN1, HIGH);
        digitalWrite(IN2, LOW);
        delay(2000);
 //Para o motor A
        digitalWrite(IN1, HIGH);
        digitalWrite(IN2, HIGH);
        delay(500); void loop()
```

Código loop() 2/3

```
…
{
  //utiliza a mesma velocidade em ambos os motores
  left = startSpeed;
  right = startSpeed;

  //le os sensores e adiciona os deslocamentos
  SENSOR1 = analogRead(0) + leftOffset;
  SENSOR2 = analogRead(1);
  SENSOR3 = analogRead(2) + rightOffset;
```

Código loop() 3/3

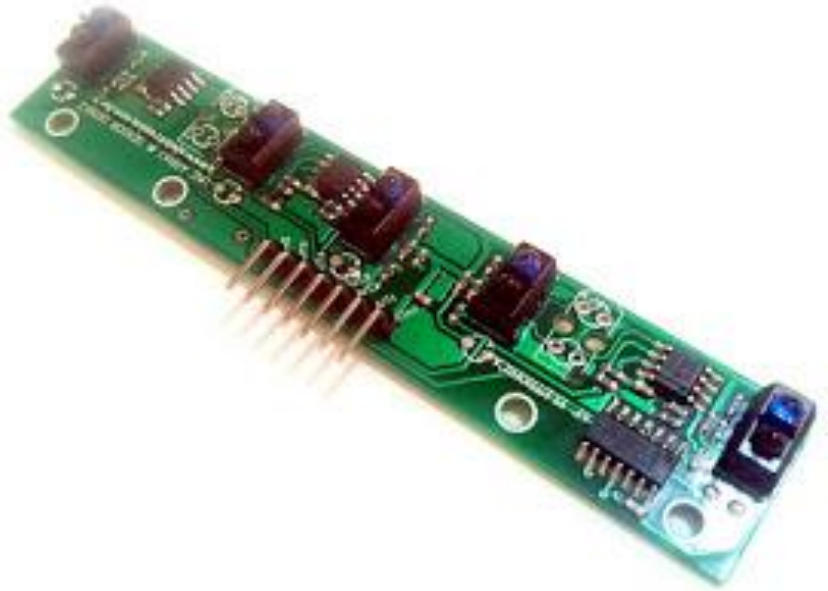
```
 //Se SENSOR1 for maior do que o sensor do centro + limiar, // vire para a direita
if (SENSOR1 > SENSOR2+threshold)
  {
    left = startSpeed + rotate;
    right = startSpeed - rotate;
  }
```

```
 //Se SENSOR3 for maior do que o sensor do centro + limiar,
// vire para a esquerda
  if (SENSOR3 > (SENSOR2+threshold))
  {
    left = startSpeed - rotate;
    right = startSpeed + rotate;
  }

  //Envia os valores de velocidade para os motores
  motor_esq.setSpeed(left);
  motor_esq.run(FORWARD);
  motor_dir.setSpeed(right);
  motor_dir.run(FORWARD);
}
```

Módulo Seguidor de Linha Line Array Sensor TCRT5000

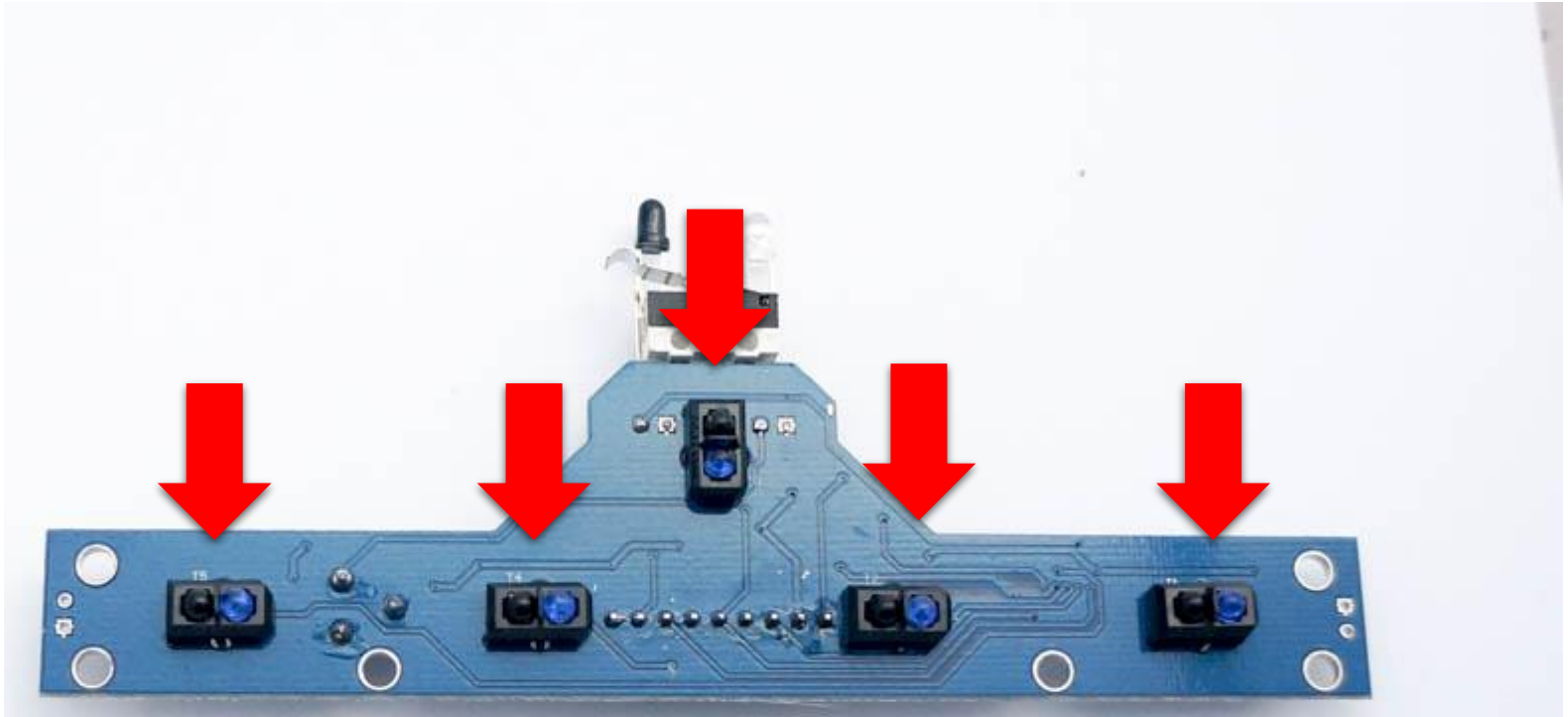
Infravermelho



Módulo Seguidor de Linha

Possui 5 sensores:

- A distância dos sensores são como mostrado na imagem abaixo
- Os sensores laterais para o do centro é de aproximadamente 46mm



Especificações e características:

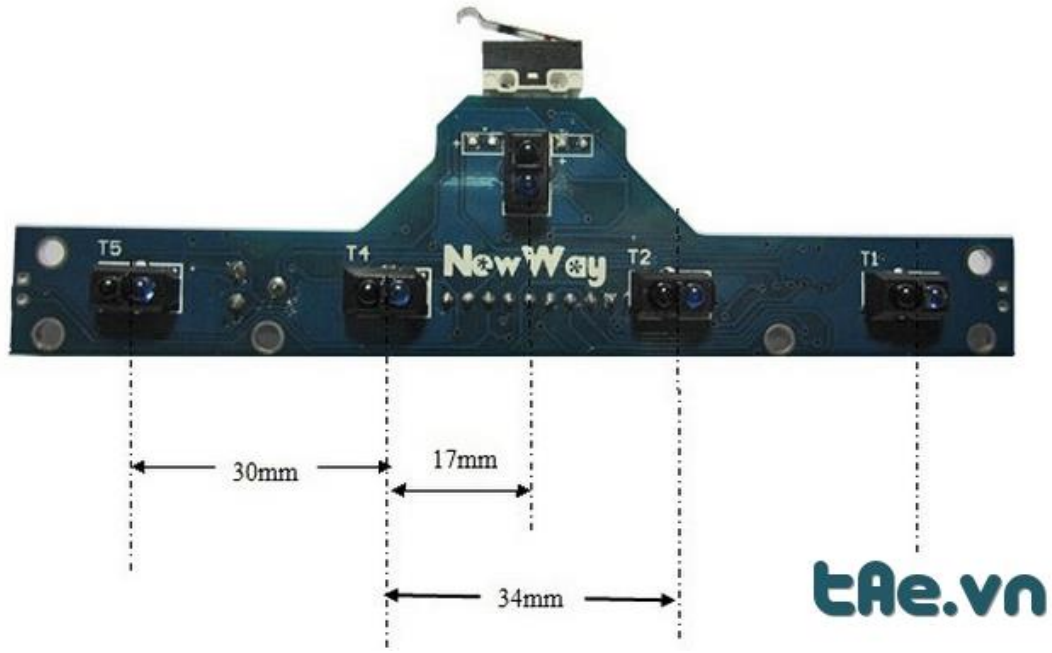
Dimensões da placa:

~125mm x 42mm x 13mm

Características Elétricas:

Alimentação: 5Vdc

Consumo: < 100mA



Código teste

Programando o sensor infravermelho TCRT5000 Arduino em linha: teste

Agora vamos testar o funcionamento do sensor infravermelho em linha.

Definições de variaveis:

```
// Programa : Teste Line Array
int valorSensorA0 = A0;
int valorSensorA1 = A1;
int valorSensorA2 = A2;
int valorSensorA3 = A3;
int valorSensorA4 = A4;
```

Código setup()

```
void setup()   {
       lcd.begin(16, 2);
       lcd.clear();
       lcd.setCursor(0, 0);
       lcd.print("  AFEletronica  ");
       lcd.setCursor(0, 1);
       lcd.print("   LineArray    ");
       Serial.begin(9600);
       Serial.print("AFEletronica LineArray");
       Serial.print("\n");
       delay(200);
      }
```

Código loop() 1/2

```
void loop()
{
  int SensorIR1 = analogRead(valorSensorA0);
  int SensorIR2 = analogRead(valorSensorA1);
  int SensorIR3 = analogRead(valorSensorA2);
  int SensorIR4 = analogRead(valorSensorA3);
  int SensorIR5 = analogRead(valorSensorA4);
```

Código loop() 2/2

```
   Serial.print(F(" SensorIR1-A1 ==>  "));
   Serial.println(SensorIR2);
   delay(200);
   Serial.print(F(" SensorIR1-A2 ==>  "));
   Serial.println(SensorIR3);
   delay(200);
{
  int SensorIR1 = analogRead(valorSensorA0);
  int SensorIR2 = analogRead(valorSensorA1);
  int SensorIR3 = analogRead(valorSensorA2);
  int SensorIR4 = analogRead(valorSensorA3);
  int SensorIR5 = analogRead(valorSensorA4);
```

```
   Serial.print(F(" SensorIR1-A3 ==>  "));
   Serial.println(SensorIR4);
   delay(200);
   Serial.print(F(" SensorIR1-A4 ==>  "));
   Serial.println(SensorIR5);
   delay(200);

}
```

Controle da velocidade dos motores por PWM

Lembrando:

O nome PWM é uma sigla que significa *Pulse Width Modulation* (Modulação em Largura de Pulso).

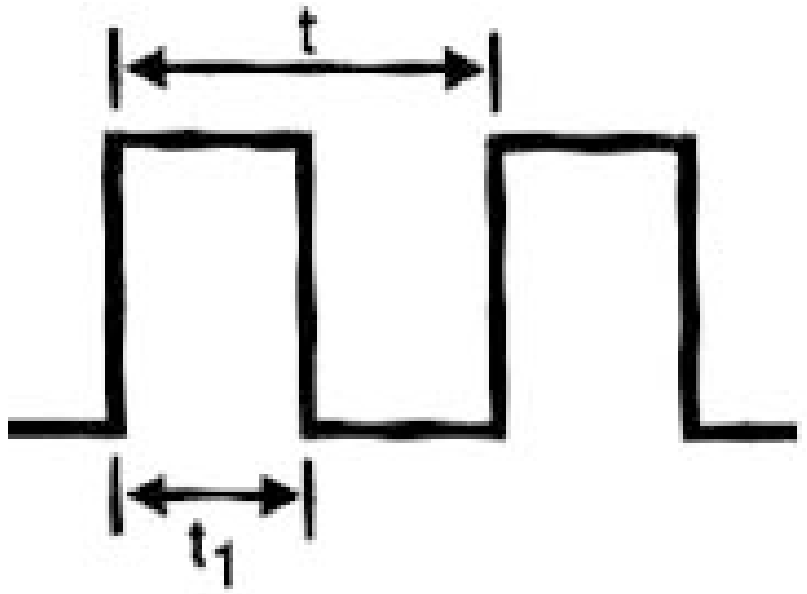
O PWM trata-se de uma modulação digital / em sinal digital, ou seja, trabalha com somente dois níveis de sinal: **on** (ou ‘1’) e off (ou ‘0’).

Em termos simples, o PWM controla quanto tempo um sinal digital fica em **on** em uma determinada janela de tempo fixa (chamada de período).

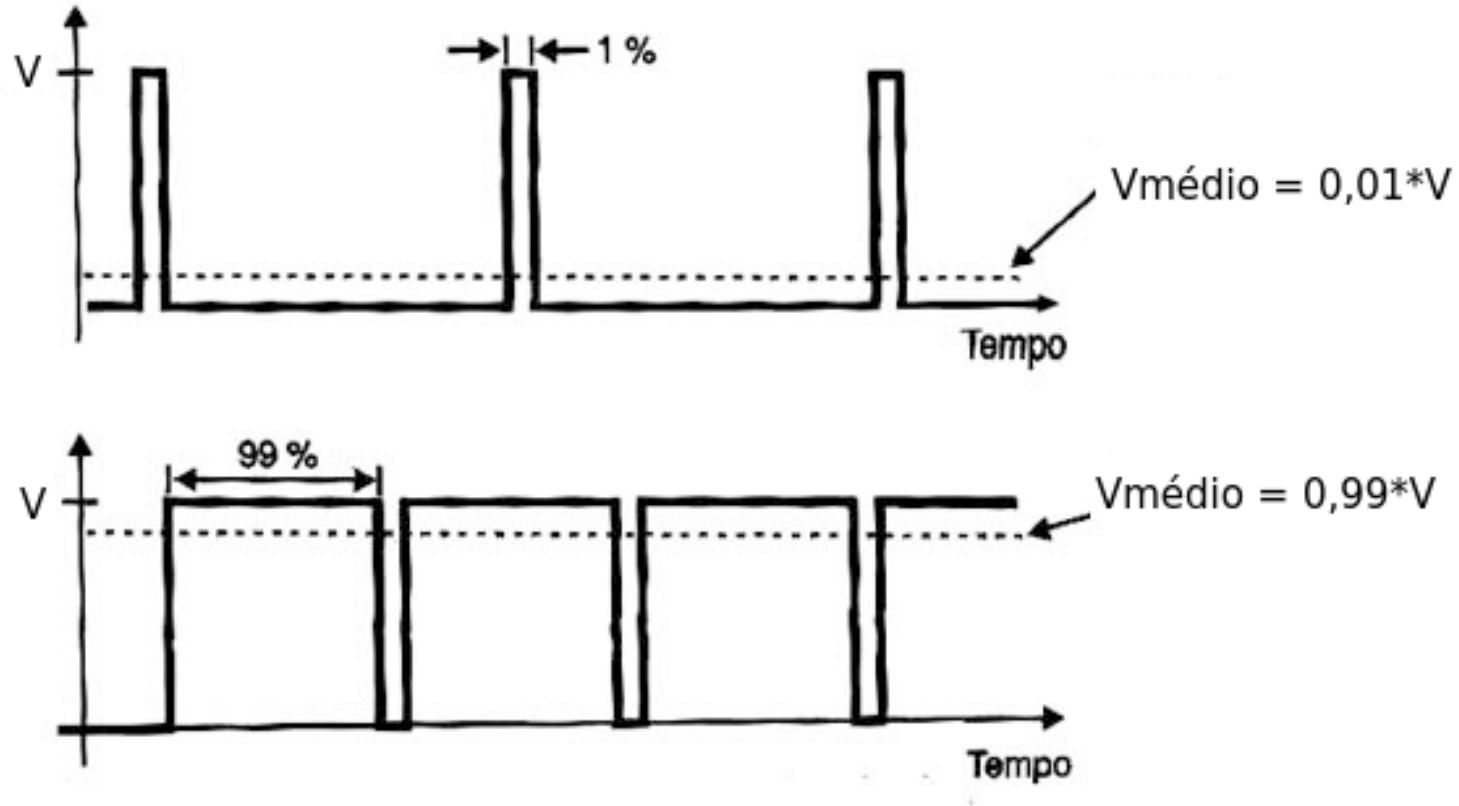
PWM

A esta porção de tempo em que o sinal fica em **on** no período, é o Duty Cycle.

O Duty Cycle é tratado na forma porcentagem, uma fração do período total do sinal.



Duty cycle



Relação

Duty cycle de 0%: tensão média igual a 0V

Duty cycle de 50%: tensão média igual a metade da tensão do sinal lógico.

Duty cycle de 100%: tensão média igual a tensão do sinal lógico.

$$DutyCycle = \frac{t_{on}}{t_{on} + t_{off}} \cdot 100$$

Relação entre velocidade de motor DC e PWM

No caso dos motores DC, a “velocidade de giro” (RPM) pode ser controlada variando-se sua tensão de alimentação.

Além disso, o RPM é diretamente proporcional à tensão aplicada. Portanto, variando-se a tensão média de alimentação do motor, varia-se o RPM do mesmo.

PWM no Arduino

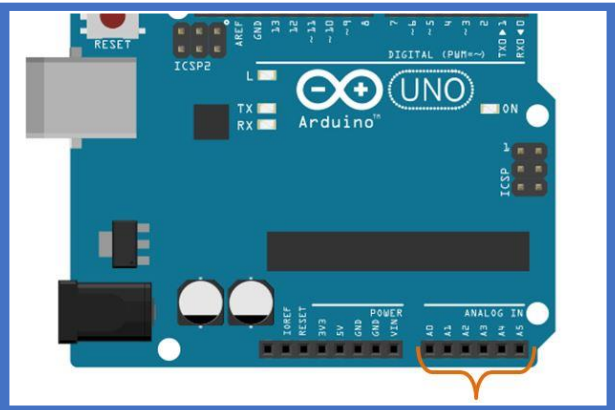
A programação do Arduino já possui uma função nativa para gerar um sinal PWM em alguns dos pinos

- (aqueles identificados com um símbolo/identificação ~).

Fácil utilização do mesmo com pouquíssimo custo em termos computacionais (uso de memória Flash, RAM e processamento) e escrita de códigos-fonte mais limpos e curtos.

Função **analogWrite()**,

Função analogwrite



analogWrite(PINO, VALOR_ANALOGICO);

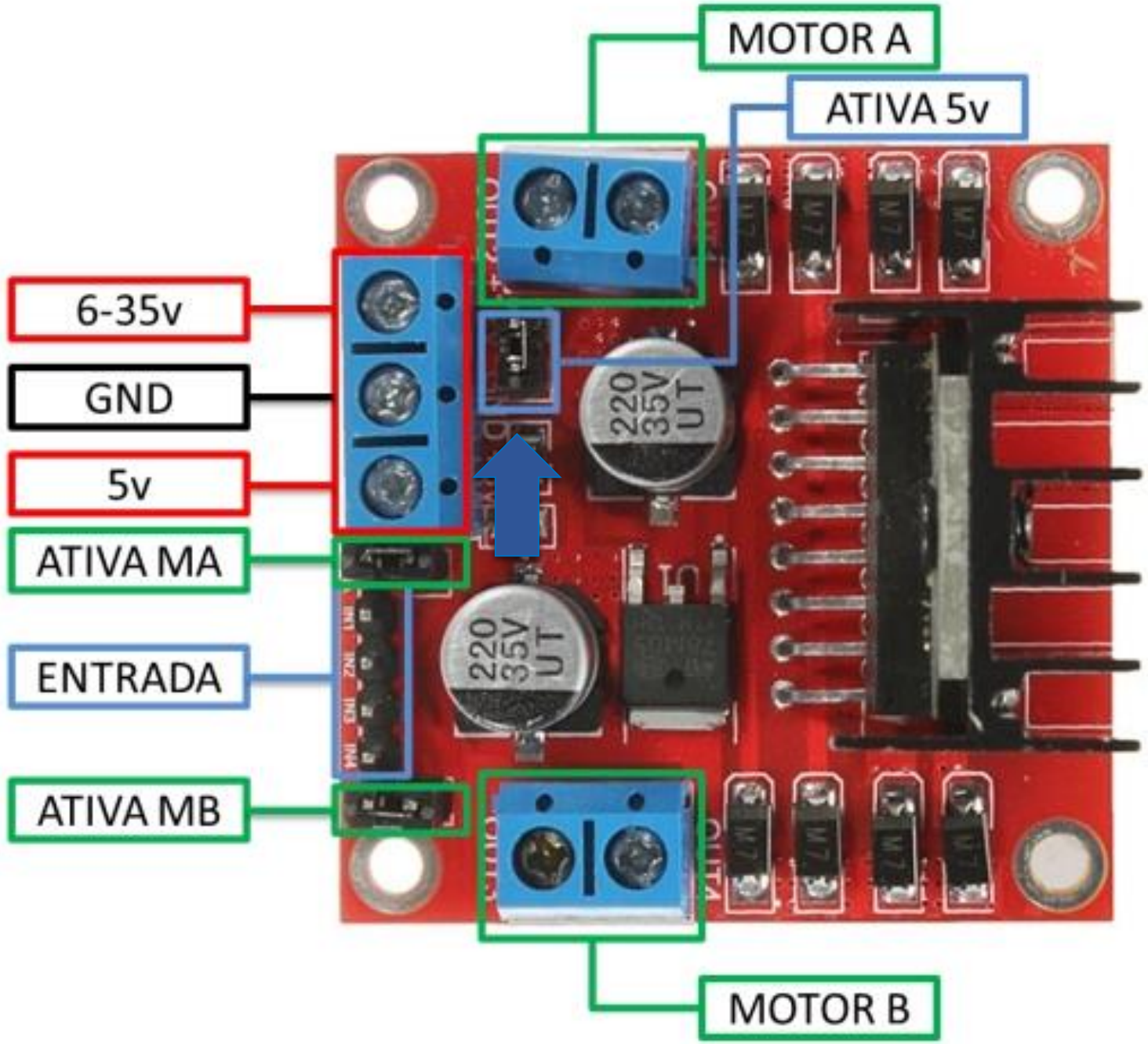
Função analogwrite

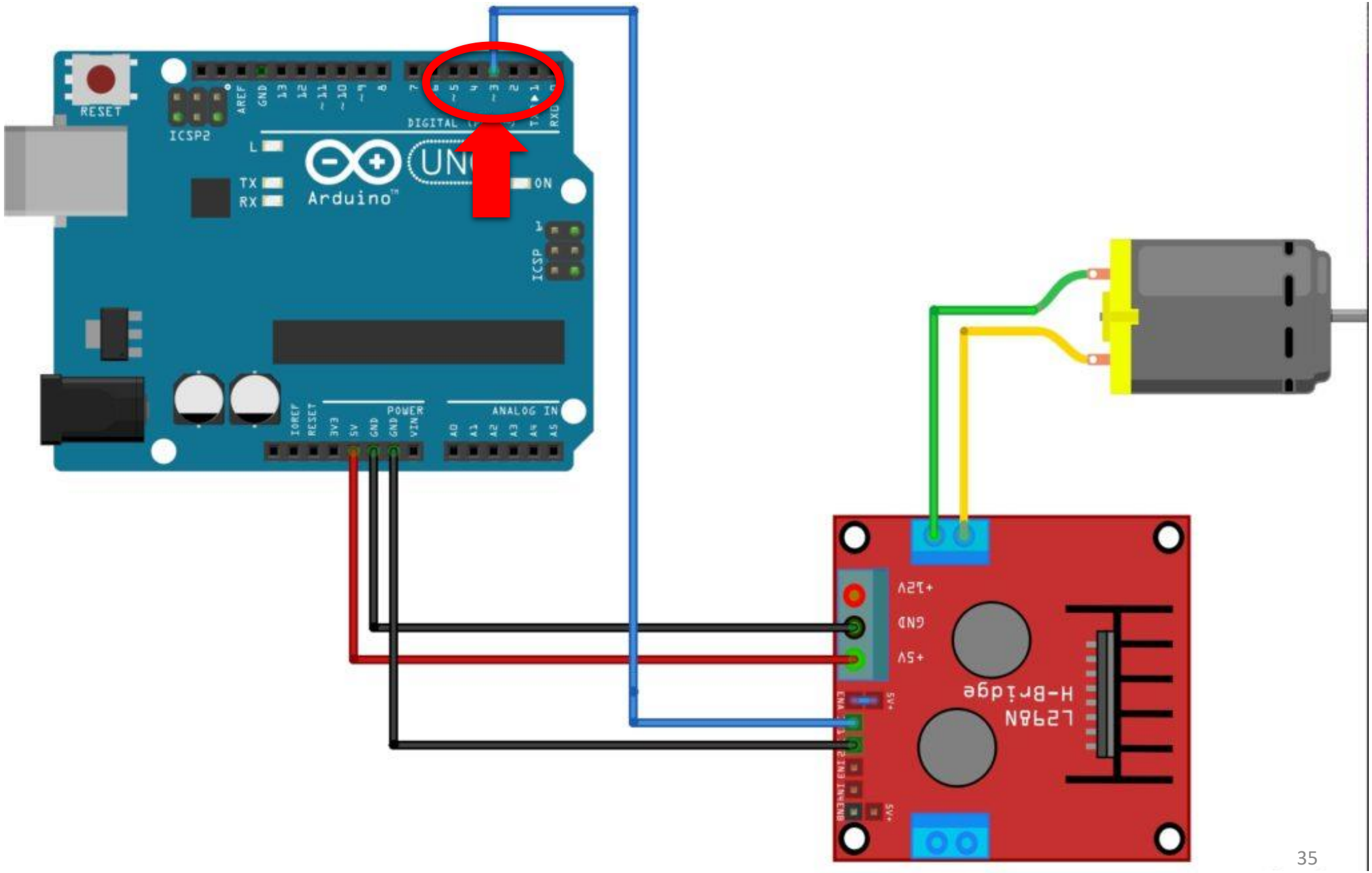
Onde:

PINO: número do pino do Arduino o qual se deseja que o sinal PWM seja gerado.
Conforme dito anteriormente, somente pinos com símbolo/identificação ~ podem gerar sinais PWM.

VALOR_ANALOGICO: valor (de 0 a 255), proporcional ao Duty Cycle a ser gerado.
Ou seja, para Duty Cycle de 100%, deve-se utilizar valor 255, já para Duty Cycle 20% deve-se utilizar o valor 51 e assim por diante.

NÃO SE ESQUEÇA de colocar um jumper em ATIVA MA




RESET
ICSP2
Arduino
DIGITAL
POWER
ANALOG IN
+12V
GND
+5V
L298N
H-Bridge

Código teste 1

Definições de variaveis:

```
/* Controle de velocidade de motor DC com PWM
#define PINO_PWM     3
//pino do Arduino que terá a ligação para o driver de motor (ponte H) L298N)
#define TEMPO_NA_MESMA_VELOCIDADE     300
//tempo (ms) em que o motor ficara na mesma velocidade
```

Código setup()

```
void setup()
{
//configura como saída pino terá a ligação para o
//driver de motor (ponte H) L298N)
    pinMode(PINO_PWM, OUTPUT);
}
```

Código loop()

```
void loop()   {
    int valor_pwm = 0;
//variavel que armazena o valor do PWM (0..255 -> 0%..100% da rotação do motor)
    //Aumento de velocidade (0%..100%)
    for (valor_pwm = 0; valor_pwm < 256; valor_pwm++)
    {
        analogWrite(PINO_PWM, valor_pwm);
        delay(TEMPO_NA_MESMA_VELOCIDADE);
    }
     //Diminuição de velocidade (100%..0%)
    for (valor_pwm = 255; valor_pwm >= 0; valor_pwm--)
    {
        analogWrite(PINO_PWM, valor_pwm);
        delay(TEMPO_NA_MESMA_VELOCIDADE);
    }
}
```

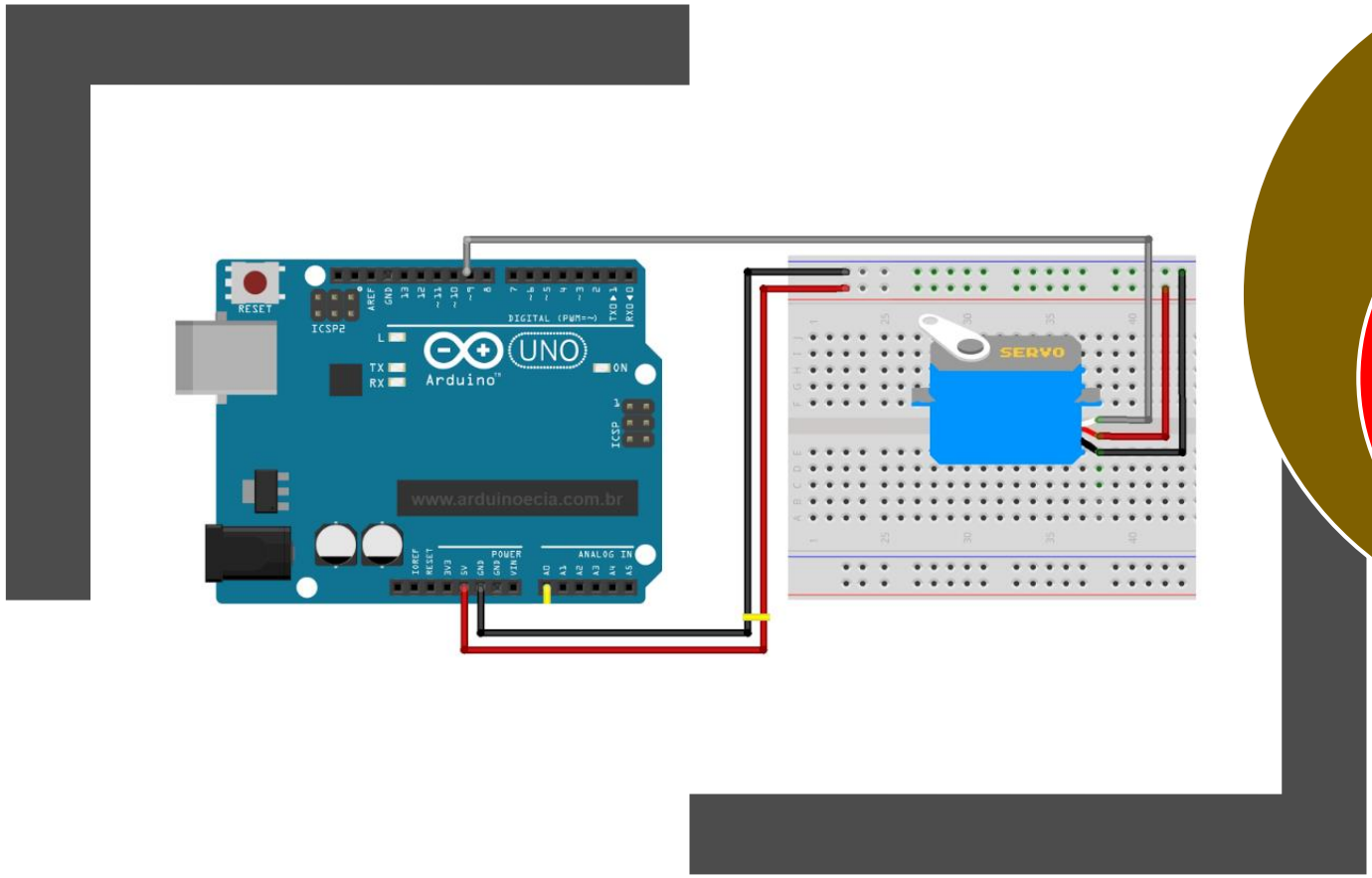
Atuador: Micro Servo Motor

Sg90

Especificações:

Voltagem de Operação: 3,0 – 7,2v
Velocidade: 0,12 seg/60Graus (4,8v) sem carga
Torque: 1,2 kg.cm (4,8v) e 1,6 kg.cm (6,0v)
Temperatura de Operação.: -30C ~ +60C
Dimensões.: 32x30x12 mm
Tipo de Engrenagem: Nylon
Tamanho cabo: 245 mm
Peso: 9g

Ligação: Conecte a alimentação do Micro Servo 9g ao Arduino.



Fio Marrom com GND,

Fio Vermelho com 5v e

Fio Laranja na Porta Digital 10.

Código teste

Com biblioteca

Definições de variaveis:

```
//Programa : Controle Atuador C Microservo - Versão 1.0
include <Servo.h>
 #define SERVO 10          // Porta Digital 10 PWM
 Servo s;                  // Variável Servo
int pos;                   // Posição Servo
```

Servo()

Código setup()

```
void setup ()
{
        s.attach(SERVO);
        Serial.begin(9600);
        s.write(0);                    // Inicia motor posição zero
}
```

Código loop()

```
void loop()  {
  for(pos = 0; pos < 90; pos++)
  {
      s.write(pos);
     delay(15);
  }
delay(1000);
  for(pos = 90; pos >= 0; pos--)
  {
    s.write(pos);
    delay(15);
  }
}
```

Entradas e saídas analógicas

Pinos de entrada e “saída” analógica

Entradas Analógicas

ARDUINO – PINOS ANALÓGICOS

- 6 PINOS ANALOGICOS – (A0, A1, A2, A3, A4 E A5)
- CONVERSOR ANALÓGICO DIGITAL (CAD) - TRADUZIR VALORES DE 0 A 5V EM 0 A 1023
- CAD – CONVERSÃO DA TENSÃO E SETA 10 BITS DE ACORDO COM A TENSÃO (10 BITS DE RESOLUÇÃO

$2^{10} = 1024$ bits

As grandezas analógicas são aquelas que podem assumir diversos valores de mplitude dentro de uma faixa de valores.

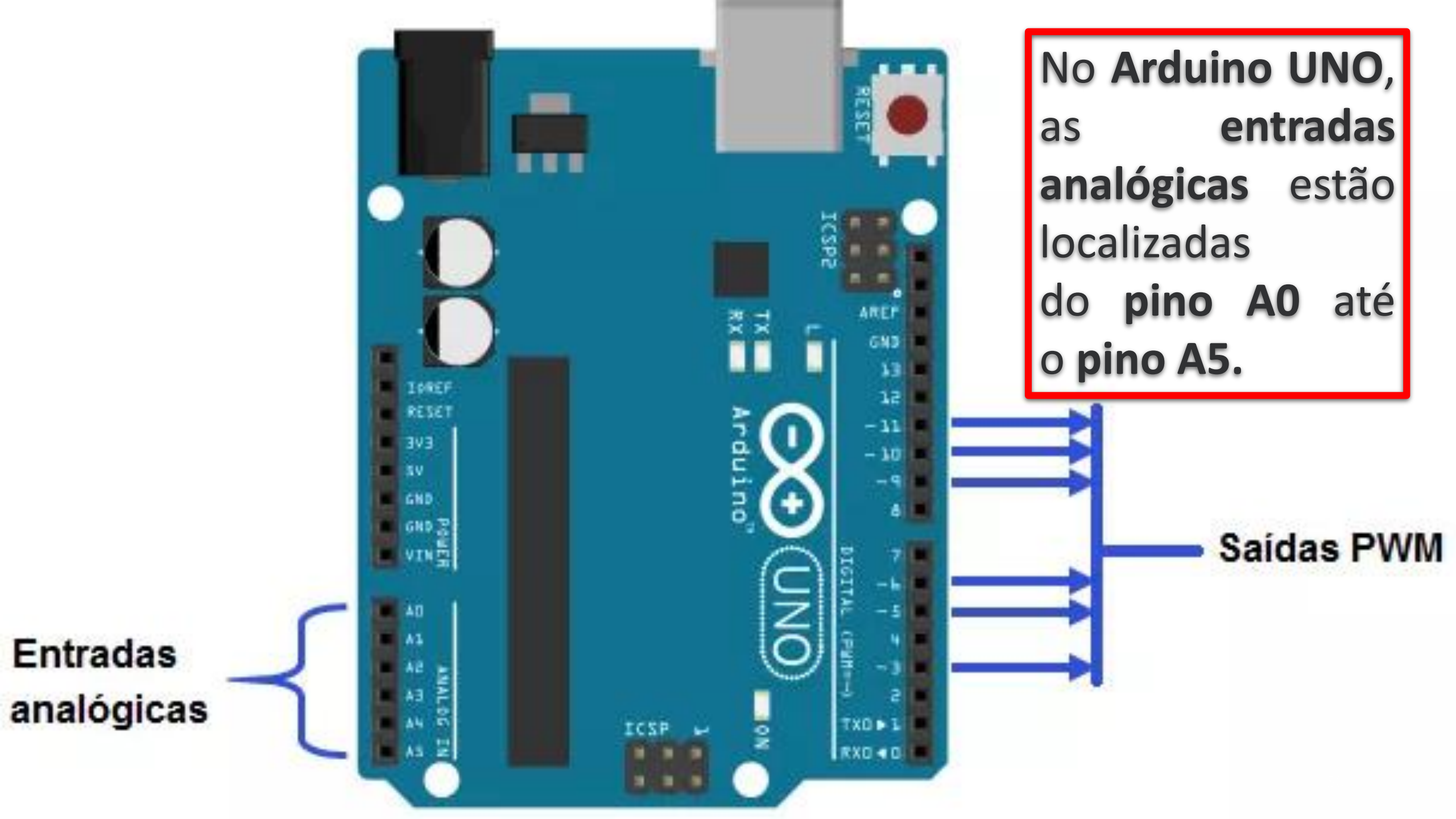
Um exemplo é o velocímetro de um carro, que por sua vez, pode ser considerado analógico, pois o ponteiro gira continuamente conforme o automóvel acelera ou freia.

Pinos de entrada e “saída” analógica

Existem pinos destinados a lidar com este tipo de grandeza, onde, alguns são utilizados como entradas analógicas, isto é, possuem a função de receber dados provenientes de grandezas analógicas

Outros, possuem a função de produzir informações que simulam o comportamento de grandezas analógicas, aqui estamos falando da utilização de uma técnica chamada **PWM**.

No **Arduino UNO**, as **entradas analógicas** estão localizadas do **pino A0** até o **pino A5.**

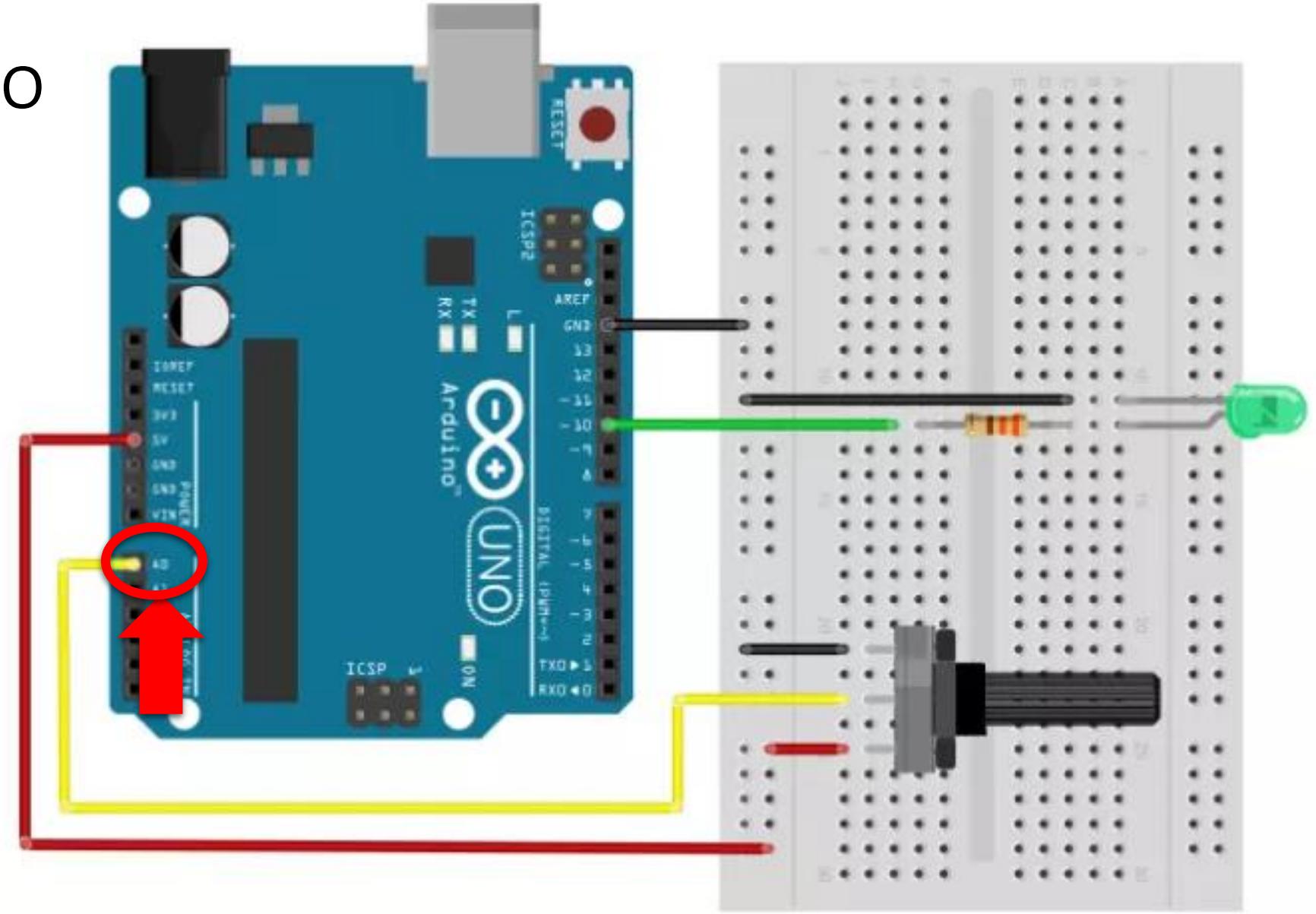


PWM

São os elementos que podem atuar fornecendo sinais que simulam o comportamento de uma grandeza analógica, são alguns dos pinos caracterizados como **pinos de entrada/saída digital**

De maneira específica, os pinos 3,5,6,9,10 e 11, os que tem um sinal de ~.

Exemplo


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
TX
RX
L
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A0
A1
ANALOG IN
DIGITAL (PWM~)
Arduino
UNO

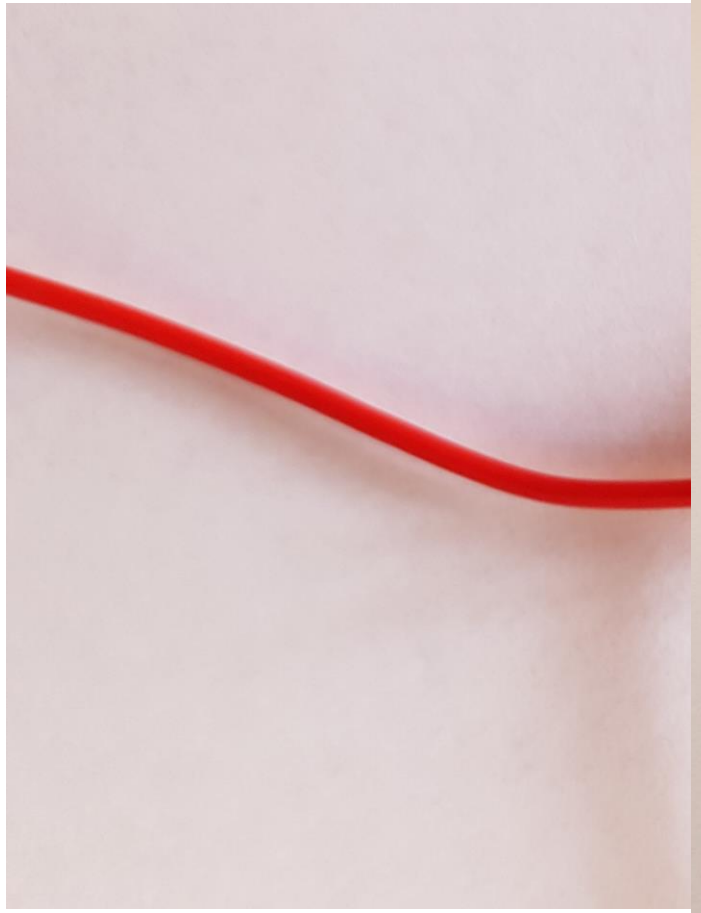
Exemplo

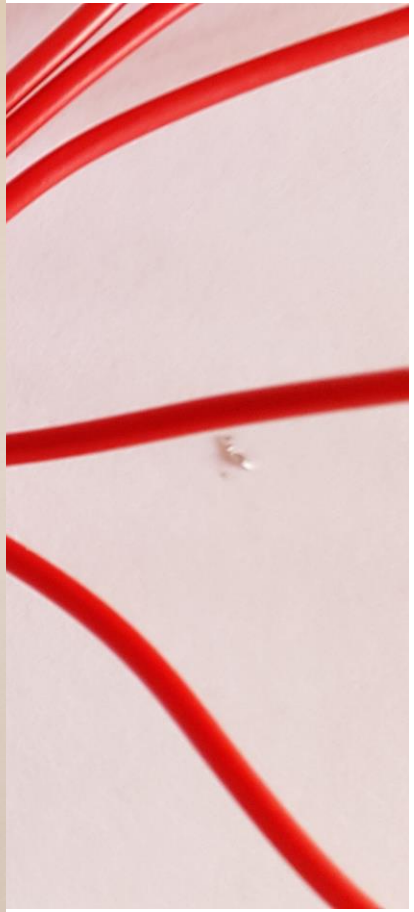
```
unsigned int potenciometro
unsigned int pwm
 void setup()
{
   pinMode(10,OUTPUT);
}
 void loop()
{
   potenciometro = analogRead(A0);
     pwm = map(potenciometro,0,1023,0,255);
     analogWrite(10,pwm);
 }
```

Dicas

Fita dupla face

Fios soldados





Dicas

Carrinho não roda – Este é um problema complexo, pois podem ser infinitas possibilidades.

Tente isolar os componentes e testar 1 por 1, principalmente os motores e a ponte H.

Em alguns casos pode ser problemas de aterramento da bateria também.

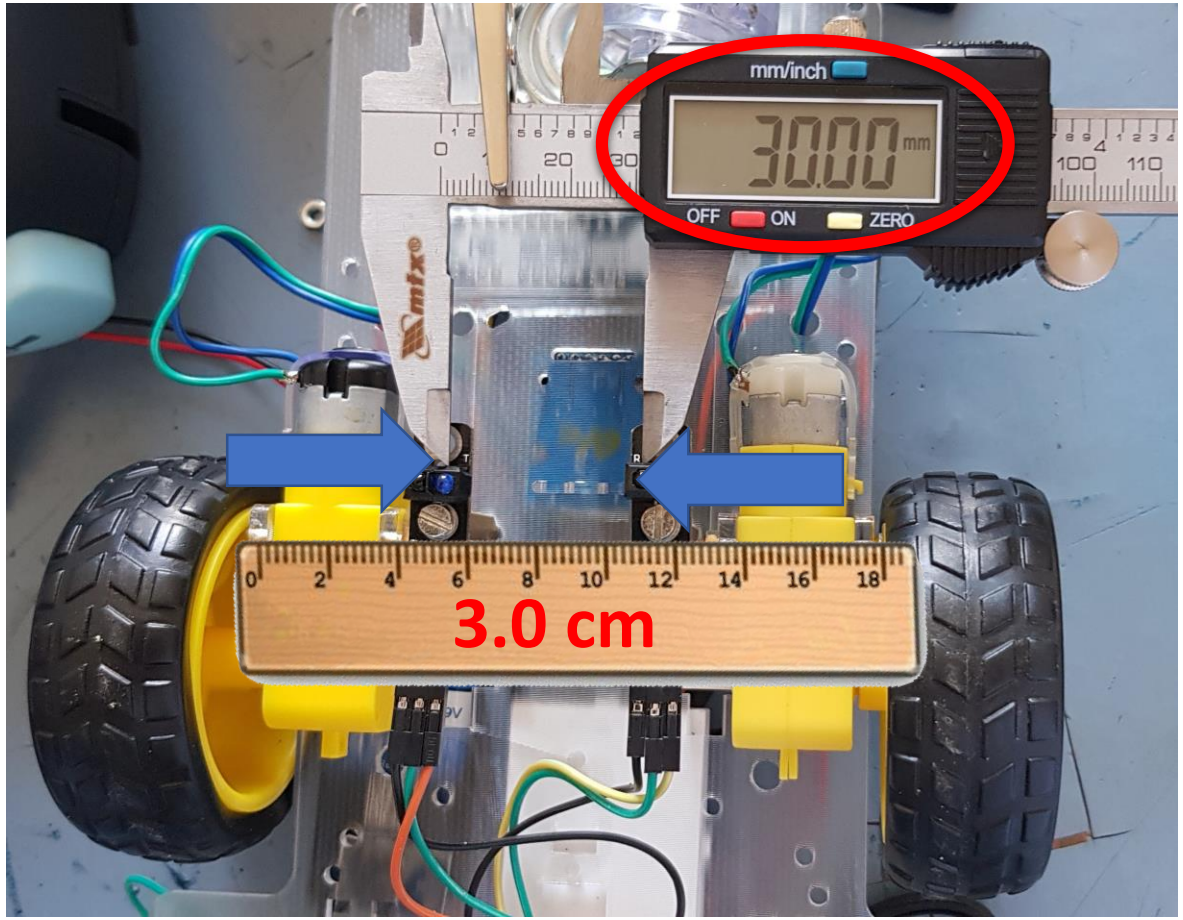
Dicas

Bateria fraca – Os testes podem ter consumido a bateria e talvez seja necessário o uso de uma nova. Baterias abaixo de 6,5 Volts já começam a diminuir a eficiência do carrinho.

Carrinho saindo da pista – Isso pode acontecer por ele estar rápido de mais ou por falha do infravermelho.

Se o problema for com o contraste da pista (talvez parte dela esteja mais escura) use 2 LEDs de alto brilho na frente do carrinho para iluminar a pista próximo aos sensores.

Distância entre os sensores infravermelhos



Referencias

https://www.arduino.cc/reference/pt/

https://www.filipeflop.com/blog/motor-dc-arduino-ponte-h-l298n/

https://portal.vidadesilicio.com.br/robo-seguidor-de-linha-sensor-infravermelho-e-pwm/

https://www.filipeflop.com/blog/micro-servo-motor-9g-sg90-com-arduino-uno/

http://blogmasterwalkershop.com.br/arduino/arduino-utilizando-o-sensor-reflexivo-tcrt5000/

http://blog.eletrogate.com/sensor-ultrassonico-hc-sr04-com-arduino/

https://www.filipeflop.com/blog/sensor-de-cor-tcs3200-rgb-arduino/

https://www.arduinoecia.com.br/2013/10/sensor-optico-reflexivo-tcrt5000.html

Referencias

http://blog.eletrogate.com/sensor-de-cor-tcs230-com-arduino/

https://portal.vidadesilicio.com.br/dht11-dht22-sensor-de-umidade-e-temperatura/

https://portal.vidadesilicio.com.br/hc-sr04-sensor-ultrassonico/

https://fritzenlab.com.br/2016/11/sensor-infravermelho-seguidor-linha/

https://www.filipeflop.com/blog/projeto-robo-seguidor-de-linha-arduino/

https://www.arduinoecia.com.br/2013/10/sensor-optico-reflexivo-tcrt5000.html

https://portal.vidadesilicio.com.br/robo-seguidor-de-linha-sensor-infravermelho-e-pwm/

https://www.embarcados.com.br/arduino-comunicacao-serial/

Fim da Aula II