



## Tutorial – Movimento iModBot@ipleiria.pt



# Introdução

O robô iModBot@ipleiria.pt é um veículo elétrico de pequenas proporções controlado pelo microcontrolador ESP32. Este microcontrolador pode ser programado usando a linguagem C/C++ e o software gratuito Arduino IDE.

Este tutorial vai focar-se na leitura dos vários sensores do robô.

O robô possui os seguintes tipos de sensores:

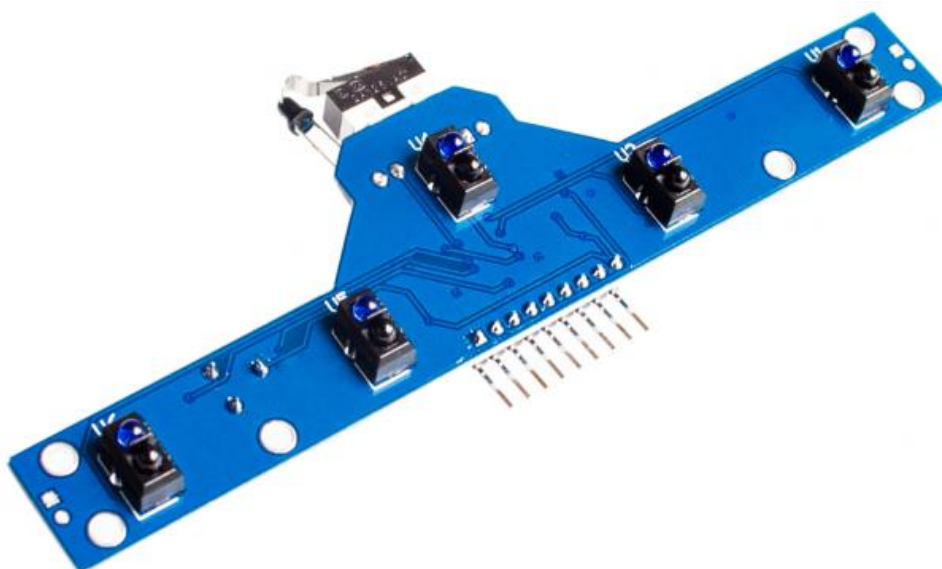
- Ultrassons, para detetar obstáculos distantes;
- Fim de curso, para detetar uma colisão;
- Sensores infravermelho (IR), para detetar um percurso (Linha perta desenhada numa superfície branca) e também para detetar obstáculos a curta distância;
- Encoder óticos, para obter um feedback da posição das rodas.



O sensor de ultrassons devolve uma distância à qual se encontra um obstáculo. Esta informação será útil para evitar que o robô colida com um obstáculo.

Ligações deste módulo:

Echo	D12
Trig	D14



O sensor fim de curso é um botão (switch, denominado CLP) colocado na parte da frente do módulo de sensores IR.

Os dois LED emissor e recetor infravermelho, denominado sensor “Near”, acima do switch permitem detetar obstáculos a curta distância, a luz natural poderá causar leituras erradas no recetor infravermelho.

Os cinco sensores infravermelho na parte inferior da placa, denominados S1, S2 ... S5, permitem detetar uma linha preta numa superfície clara (branca).

Ligações deste módulo:

S1	D25
S2	D33
S3	D32
S4	D35
S5	D34
CLP	VN
Near	VP



Os sensores/encoder óticos servem para obter um feedback da rotação das rodas podendo deste modo obter a distância percorrida e saber se uma está a rodar mais rapidamente que a outra.

Ligações deste módulo:

Encoder da roda esquerda	D0	D26
Encoder da roda direita	D0	D27

Deve haver uma peça semelhante a esta no eixo da roda:



Esta peça permite ao Encoder fazer a contagem dos pulsos da roda. Cada vez que uma roda dá uma volta completa (360°) a contagem deve aumentar 40 pulsos.

A contagem dos pulsos aumenta sempre, independentemente da direção em que as rodas se deslocam.

O número de pulsos também pode ser usado para calcular a distância percorrida pela roda, para tal precisamos de medir o diâmetro (d) da roda. Após sabermos o diâmetro vamos calcular o perímetro (P) usando a fórmula  $P = \pi * d$ . Após conhecermos o perímetro sabemos que 40 pulsos equivalem ao perímetro calculado, desta forma a distância percorrida (D) pode ser calculada da seguinte maneira:

$$D = \frac{\text{pulsos} * P}{40}$$
 em que P é o perímetro da roda, “pulsos” o número de pulsos contados e D a distância percorrida.

**Ex1 - Usar sensor de ultrassons com a biblioteca****Descrição**

Este exemplo obter a distância, em cm, à qual se encontra um obstáculo e apresentar a mesma no monitor série.

**Sintaxe**

*Distance(); = devolve o valor, em cm, da distancia medida.*

**Parâmetros**

Nenhum

**Respostas**

Nenhum

**Exemplo:**

```
#include <robotOnLine.h>

robotOnLine robot;

void setup()
{
  robot.begin();

  Serial.begin(115200);
}

void loop()
{
  byte dist = robot.distance();
  Serial.println(dist);

  delay(250);
}
```

## Ex2 - Ler sensor CLP, Near, S1, S2, S3, S4 e S5

### Descrição

Este exemplo permite obter o estado de cada pino e apresentar o mesmo no monitor série.

### Sintaxe

```
readCLP();
readNear();
readS1();
readS2();
readS3();
readS4();
readS5();
```

### Parâmetros

state = variável que recebe o valor da função, 1 ou 0, consoante o estado do pino.

### Respostas

Nenhum

### Exemplo:

```
#include <robotOnLine.h>

robotOnLine robot;

void setup()
{
    robot.begin();

    Serial.begin(115200);
}

void loop()
{
    bool state = 0;

    state = robot.readCLP();
    Serial.print("Estado do pino CLP: ");
    Serial.println(state);

    state = robot.readNear();
    Serial.print("Estado do pino Near: ");
    Serial.println(state);

    state = robot.readS1();
    Serial.print("Estado do pino S1: ");
    Serial.println(state);

    state = robot.readS2();
    Serial.print("Estado do pino S2: ");
    Serial.println(state);

    state = robot.readS3();
    Serial.print("Estado do pino S3: ");
    Serial.println(state);

    state = robot.readS4();
    Serial.print("Estado do pino S4: ");
    Serial.println(state);

    state = robot.readS5();
    Serial.print("Estado do pino S5: ");
    Serial.println(state);

    delay(250);
}
```

### Ex3 - Ler e reiniciar a contagem de pulsos dos Encoders.

#### Descrição

Este exemplo permite ler o número de pulsos contados pelos Encoder óticos e apresentar os mesmos no monitor série.

#### Sintaxe

*getLeftEncoderCount();/\*  
devolve um número de  
pulsos contados pelo  
Encoder da roda  
esquerda \*/*

*getRightEncoderCount();  
/\* devolve um número de  
pulsos contados pelo  
Encoder da roda direita  
\*/*

*clearEncoderCount(); /\*  
coloca a contagem de  
pulsos a zero \*/*

#### Parâmetros

Nenhum

#### Respostas

“short” é um tipo de variável com metade do tamanho da variável de tipo “int”

#### Exemplo:

```
#include <robotOnLine.h>

robotOnLine robot;

short leftPulse = 0;
short rightPulse = 0;

void setup()
{
    robot.begin();

    Serial.begin(115200);
}

void loop()
{
    leftPulse = robot.getLeftEncoderCount();
    rightPulse = robot.getRightEncoderCount();

    Serial.print("Pulso da roda esquerda: ");
    Serial.print(leftPulse);
    Serial.print("\t");
    Serial.print("Pulso da roda direita: ");
    Serial.println(rightPulse);

    if( (leftPulse > 10000) || (rightPulse > 10000) )
    {
        robot.clearEncoderCount();
    }

    delay(250);
}
```