



Tutorial – Movimento iModBot@ipleiria.pt

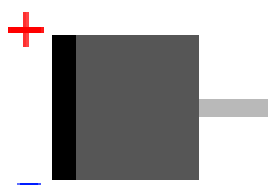


Introdução

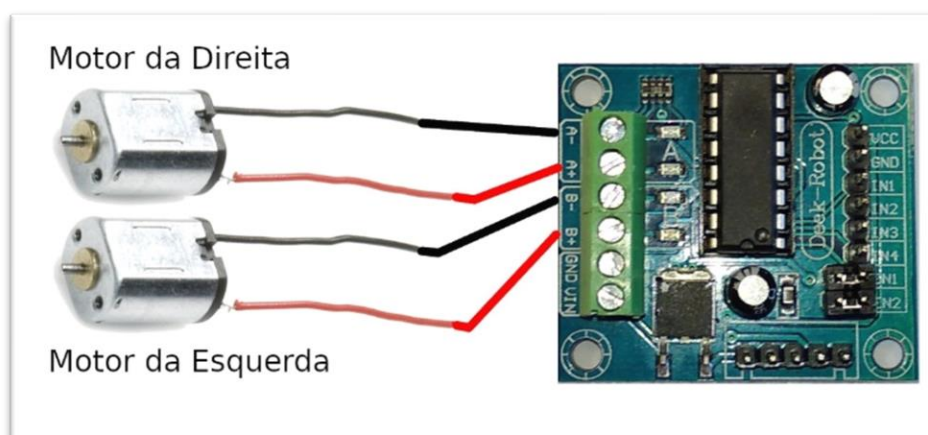
O robô **iModBot** é um veículo elétrico de pequenas proporções controlado pelo microcontrolador ESP32. Este microcontrolador pode ser programado usando a linguagem C/C++ e o software gratuito Arduino IDE.

Este tutorial vai focar-se no movimento do robô.

O robô possui duas rodas que lhe permitem mover em qualquer direção. Cada roda possui um motor elétrico cuja direção é controlada pela sua polarização.



Este motor é conectado ao módulo L293D como descrito no guia de montagem:



O controlo dos motores é efetuado diretamente pelo circuito integrado L293D, sendo este controlado pelo microcontrolador ESP32. A velocidade de cada roda é controlada por modulação de largura de pulso (PWM).

São usados dois pinos para controlar cada motor, sendo assim usados quatro pinos no total para o controlo das duas rodas.

Estes quatro pinos, geralmente, podem ser qualquer GPIO (General Purpose Input Output). Cada módulo de desenvolvimento baseado no microcontrolador ESP32 possui pinos (GPIO) restritos e pinos disponíveis diferentes.

O módulo que é enviado juntamente com o kit **iModBot** é denominado “DOIT ESP32 DevKit V1”. Se for usada a biblioteca “RobotOnLine_Rev2” por defeito os pinos destinados ao controlo das rodas serão:

- GPIO 4 e GPIO 16, para a roda direita;
- GPIO 17 e GPIO 18, para a roda esquerda.

Número do GPIO:	Estado lógico:	Efeito:
4	1	Motor direito recua.
16	0	
4	0	Motor direito avança.
16	1	
17	1	Motor esquerdo avança.
18	0	
17	0	Motor esquerdo recua.
18	1	

Usando as funções proporcionadas pelo ambiente Arduino um estado lógico “1” pode ser causado no pino “4” usando “**digitalWrite(4, HIGH);**”, da mesma maneira um estado lógico “0” pode ser causado usando “**digitalWrite(4, LOW);**”.

Para parar um motor é necessário colocar ambos os pinos no estado lógico “0”.

Para poder controlar a velocidade a que o motor gira é apenas necessário substituir o estado lógico “1”, indicado na tabela acima, por um sinal PWM.

A biblioteca “RobotOnLine_Rev2” usa, por defeito, os seguintes parâmetros para o sinal PWM:

- Frequência de 5kHz.
- Resolução de 8 bits.

Ex 1 - Controlo do movimento simples.

Descrição

Este exemplo permite mover o robô em qualquer direção sem controlo da velocidade.

Sintaxe

`digitalWrite(pino, estado);`

Parâmetros

pino //número associado ao pino
estado //HIGH ou LOW

Respostas

Nenhum

Exemplo:

```
void setup()
{
  pinMode( 4,OUTPUT); //Direita
  pinMode(16,OUTPUT); //Direita
  pinMode(17,OUTPUT); //Esquerda
  pinMode(18,OUTPUT); //Esquerda
}

void loop()
{
  // Para seguir em frente
  digitalWrite( 4, LOW);
  digitalWrite(16, HIGH);

  digitalWrite(17, HIGH);
  digitalWrite(18, LOW);

  delay(1000);
  // Para recuar
  digitalWrite( 4, HIGH);
  digitalWrite(16, LOW);

  digitalWrite(17, LOW);
  digitalWrite(18, HIGH);

  delay(1000);

  // Rodar para a direita
  digitalWrite( 4, HIGH);
  digitalWrite(16, LOW);

  digitalWrite(17, HIGH);
  digitalWrite(18, LOW);

  delay(1000);

  // Rodar para a esquerda
  digitalWrite( 4, LOW);
  digitalWrite(16, HIGH);

  digitalWrite(17, LOW);
  digitalWrite(18, HIGH);

  delay(1000);

  //Parar os motores
  digitalWrite( 4, LOW);
  digitalWrite(16, LOW);

  digitalWrite(17, LOW);
  digitalWrite(18, LOW);

  delay(1000);
}
```

Ex 2 - Controlo usando a biblioteca

Descrição

Permite controlar o movimento do robô em qualquer direção e também o controlo da velocidade das rodas.

Sintaxe

```
forward(velocidade );  
reverse(velocidade);  
rotateLeft(velocidade);  
rotateRight(velocidade):  
stopMotors();
```

Parâmetros

Velocidade = é um número de 0 a 255.

Respostas

O número de bytes disponíveis para serem lidos

Exemplo:

```
#include <iModBot.h>  
  
iModBot robot;  
  
void setup()  
{  
  robot.begin();  
}  
  
void loop()  
{  
  // Velocidade  
  // Pode variar de 1 a 255  
  byte speed = 255;  
  
  // Para seguir em frente  
  robot.forward(speed);  
  
  delay(500);  
  
  // Para recuar  
  robot.reverse(speed);  
  
  delay(500);  
  
  // Rodar para a direita  
  robot.rotateRight(speed);  
  
  delay(500);  
  
  // Rodar para a esquerda  
  robot.rotateLeft(speed);  
  
  delay(500);  
  
  // Parar os motores  
  robot.stopMotors();  
  
  delay(1000);  
}
```

Ex 3 - Controlo individual de cada roda usando a biblioteca**Descrição**

Neste exemplo é possível controlar individualmente a direção de cada roda.

Sintaxe

```
rightWheel(velocidade);  
leftWheel(velocidade);
```

Parâmetros

Velocidade = número de -255 a 255, um valor negativo faz a roda recuar e vice-versa. Um 0 faz a roda parar.

Respostas

Nada

Exemplo:

```
#include <iModBot.h>  
  
iModBot robot;  
  
void setup()  
{  
    robot.begin();  
}  
  
void loop()  
{  
    // Mover roda esquerda em frente  
    robot.leftWheel(255);  
  
    delay(1000);  
  
    // Mover roda esquerda para trás  
    robot.leftWheel(-255);  
  
    delay(1000);  
  
    // Mover roda direita em frente  
    robot.rightWheel(255);  
  
    delay(1000);  
  
    // Mover roda direita para trás  
    robot.rightWheel(-255);  
  
    delay(1000);  
}
```

Ex 4 - Rodar o robô “n” graus usando a biblioteca

Descrição

Neste exemplo é possível rodar o robô “n” graus e escolher para que lado o mesmo deve rodar (esquerda ou direita).

Sintaxe

turnLeft (graus);
turnRight (graus);

Parâmetros

Graus = é um número de 1 a 360 que corresponde a quantos graus o robô irá rodar. Um 0 faz o robô rodar 90 graus.

Respostas

Nada

Exemplo:

```
#include <iModBot.h>

iModBot robot;

void setup()
{
  robot.begin();
}

void loop()
{
  // Rodar para a esquerda 90 graus
  robot.turnLeft(90);

  delay(1000);

  // Rodar para a direita 270 graus
  robot.rightWheel(270);

  delay(1000);
}
```

Ex 5 - Controlar velocidade e movimento sem biblioteca

Descrição

Este exemplo permite controlar o movimento do robô sem recurso à biblioteca “RobotOnLine_Rev2”.

Sintaxe

ledcSetup(canalPWM, frequenciaPWM, resoluçãoPWM);

ledcWrite(canalPWM, PWM);

Parâmetros

canalPWM = canal PWM de 0 a 15.

frequenciaPWM = frequência de 1 Hz a 312.5kHz

resoluçãoPWM = resolução do pulso PWM

PWM = largura do pulso PWM (8 bits, 0 a 255 neste caso)

Returns

Exemplo:

```
#define PWM_Frequency 5000 // PWM
frequency, up to 312.5kHz
#define _IN1_PWM_channel 0 //
select the channel number
#define _IN2_PWM_channel 1 // there
are 15 channels,
#define _IN3_PWM_channel 2 //
choose between 0 to 15
#define _IN4_PWM_channel 3 // (not
related to the pins)
#define PWM_RESOLUTION 8 // 8
bits

//motor pins IN1&IN2 right engine
byte _IN1 = 4; //IO15 and 5 are
HIGH during boot.
byte _IN2 = 16;

//IN3&IN4 left engine
byte _IN3 = 17;
byte _IN4 = 18;

void setup()
{
    pinMode(4,OUTPUT);
    pinMode(16,OUTPUT);
    pinMode(17,OUTPUT);
    pinMode(18,OUTPUT);

    //setup PWM for motor pins
    ledcSetup(_IN1_PWM_channel,
    PWM_Frequency, PWM_RESOLUTION);
    ledcAttachPin(_IN1,
    _IN1_PWM_channel);

    ledcSetup(_IN2_PWM_channel,
    PWM_Frequency, PWM_RESOLUTION);
    ledcAttachPin(_IN2,
    _IN2_PWM_channel);

    ledcSetup(_IN3_PWM_channel,
    PWM_Frequency, PWM_RESOLUTION);
    ledcAttachPin(_IN3,
    _IN3_PWM_channel);

    ledcSetup(_IN4_PWM_channel,
    PWM_Frequency, PWM_RESOLUTION);
    ledcAttachPin(_IN4,
    _IN4_PWM_channel);

    ledcWrite(_IN1_PWM_channel, 0);
    ledcWrite(_IN2_PWM_channel, 0);
    ledcWrite(_IN3_PWM_channel, 0);
    ledcWrite(_IN4_PWM_channel, 0);
}

void loop()
```

```
{  
    byte speed = 255;  
  
    // move forward  
    //right engine forward  
    ledcWrite(_IN1_PWM_channel, 0);  
    ledcWrite(_IN2_PWM_channel,  
speed);  
    //left engine forward  
    ledcWrite(_IN3_PWM_channel,  
speed);  
    ledcWrite(_IN4_PWM_channel, 0);  
  
    delay(1000);  
  
    // turn left  
    //right engine forward  
    ledcWrite(_IN1_PWM_channel, 0);  
    ledcWrite(_IN2_PWM_channel,  
speed);  
    //left engine reverse  
    ledcWrite(_IN3_PWM_channel, 0);  
    ledcWrite(_IN4_PWM_channel,  
speed);  
  
    delay(1000);  
  
    // turn right  
    //right engine reverse  
    ledcWrite(_IN1_PWM_channel,  
speed);  
    ledcWrite(_IN2_PWM_channel, 0);  
    //left engine forward  
    ledcWrite(_IN3_PWM_channel,  
speed);  
    ledcWrite(_IN4_PWM_channel, 0);  
  
    delay(1000);  
  
    // reverse  
    //right engine reverse  
    ledcWrite(_IN1_PWM_channel,  
speed);  
    ledcWrite(_IN2_PWM_channel, 0);  
    //left engine reverse  
    ledcWrite(_IN3_PWM_channel, 0);  
    ledcWrite(_IN4_PWM_channel,  
speed);  
  
    delay(1000);  
}
```