ELETRÔNICA ÔMEGA



E-BOOK: TANQUE COM ESP32

Sumário

1 INTRODUÇÃO 2
2 CONCEITOS IMPORTANTES 3
3 COMPOSIÇÃO DO ESP32 13
4 BAIXANDO A IDE DO ARDUINO 16
5 CONFIGURANDO O ESP32 NA IDE 19
6 PONTE H 22
7 MOTOR DC 25
8 OUTROS COMPONENTES 26
9 MONTAGEM DO ROBÔ 27
10 CIRCUITO 28
11 CÓDIGO 36
12 POSSÍVEIS PROBLEMAS 43
13 FINALIZAÇÃO 44

1 Introdução

Neste E-book, você aprenderá a montar o nosso mais recente projeto: o Tanque Omega. Esse veículo é equipado com uma placa **ESP32** e permite controle remoto sem fio via Bluetooth utilizando um controle de **PS4**. Você poderá explorar todo o processo de construção deste incrível robô.

Tanque ESP32

Tanque robótico controlado via **Controle de PS4** com **ESP32** e corpo impresso em 3D, equipado com **4 motores DC** e **ponte H** e **servo motores** Movimenta-se em terrenos variados com esteiras robustas.

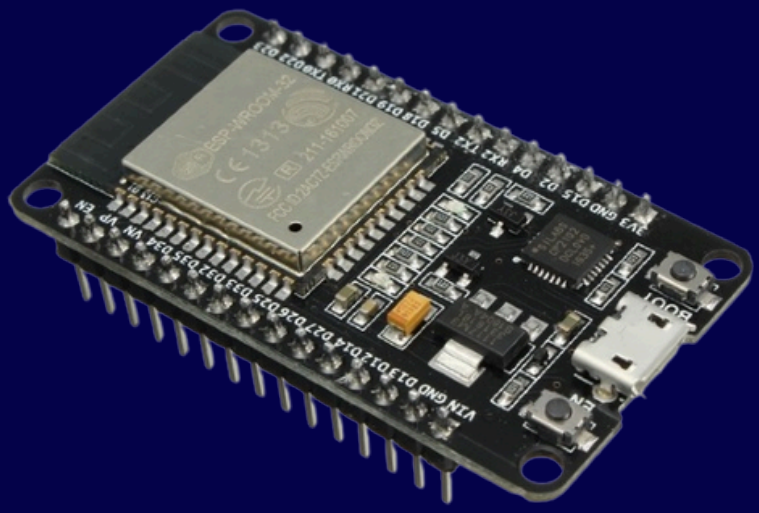


Você pode adquirir esse Kit
Clicando Aqui !!

2 Conceitos Importantes

O que é ESP-32 ??

Em nossos projetos vamos utilizar o **ESP-32**, ele nada mais é do que uma placa de desenvolvimento de código aberto que utiliza o chip **ESP32**. É muito semelhante ao **Arduino Uno**, visto que ambos são **microcontroladores**. A vantagem de se utilizar o ESP-32 é ele possuir **Wi-Fi** e **Bluetooth** nativo, dessa forma essa placa é ideal para projetos de automação residencial e internet das coisas.



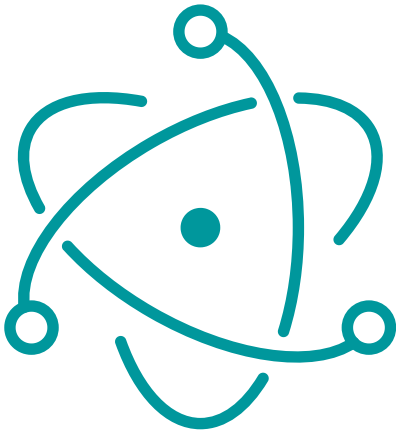
O **ESP-32** possui 36 pinos digitais e 16 podem ser utilizados como saída PWM (controle de dispositivos variando a intensidade).

Outro diferencial é a possibilidade de fazer a **programação** da placa via **OTA** (Over The Air), assim, através do **Wi-Fi** é possível **programar** o **ESP-32**.

Além disso, é possível programar a placa com as linguagens de programação: **Lua**, **Python**, **JavaScript** ou até mesmo com a **IDE do Arduino**.

Entradas e Saídas

Assim como o **Arduino**, o **ESP-32** possui entradas e saídas.

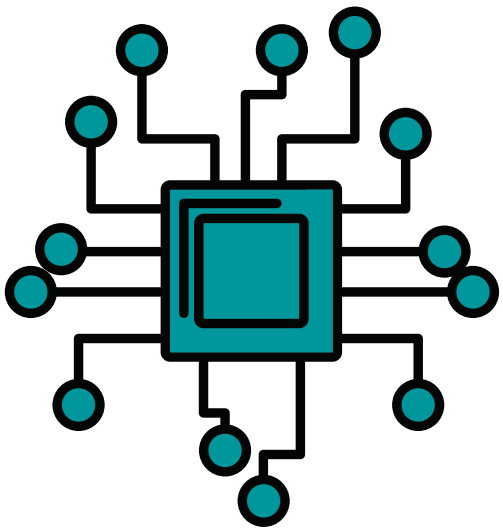


De forma resumida as entradas são por exemplo apertar um botão, e assim, quando ele é acionado é enviado um sinal para o **microcontroldor** do **ESP-32** que pode ativar algo, como por exemplo um **LED** ou um **motor**.

Já as saídas são exatamente o que é ativado pelo Arduino ou ESP-32, como um **LED**, **motor**, **buzzer**, entre outros.

Microcontroladores

Os **microcontroladores** são uma junção de Software e Hardware que através da programação conseguimos controlá-los para desempenhar tarefas. Ele é um tipo especial de circuito integrado, visto que conseguimos programar os **microcontroladores** para realizar tarefas específicas. Em breve vamos descobrir diversas coisas que podemos fazer utilizando **microcontroladores**. No caso do **ESP** o microcontrolador utilizado é o **ESP-32**.



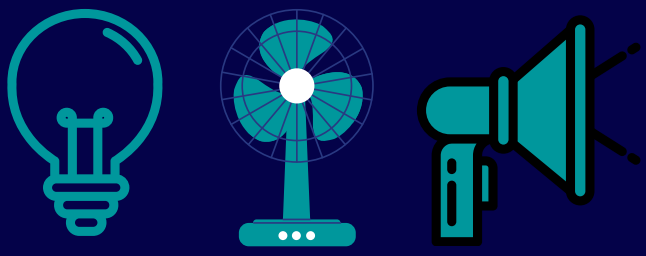
Sinais Digitais

Como visto anteriormente, o **Arduino Uno** possui **14 Portas Digitais**, mas afinal, você sabe o que é um **Sinal Digital** ?



O **Sinal Digital** possui uma quantidade limitada, geralmente é representado por dois níveis (Alto ou Baixo). Assim é definido para instantes de tempo e o conjunto de valores que podem assumir é limitada. Podemos usar como **exemplos** de **Saídas Digitais**:

- Lâmpadas;
- Buzinas;
- Ventiladores.



Observe que todos esses equipamentos estão ou ligados ou desligados, não há uma variação contínua.

E também existem as Entradas Digitais, por exemplo:

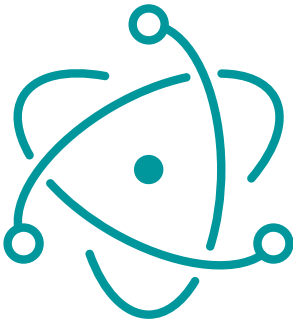
- Chaves seletoras;
- Fins de Curso;
- Botões.



Sinais Analógicos

Além de portas digitais, o **ESP** também possui **Portas Analógicas**. O **Sinal Analógico** é um sinal que pode assumir infinitos valores em um intervalo de tempo. **Exemplos** de **Sinais Analógicos** são:

- **Sensores de Temperatura;**
- **Sensores de Umidade;**
- **Sensores de Pressão;**
- **Potenciômetros.**



Simplificando...

Para simplificar, pense no interruptor e um dimmer de uma lâmpada…

Ou o interruptor está **acionado** e a lâmpada está **acesa** ou está **desacionado** e a lâmpada **apagada**, não existe outro estado. Dessa forma o interruptor é considerado um Sinal Digital.

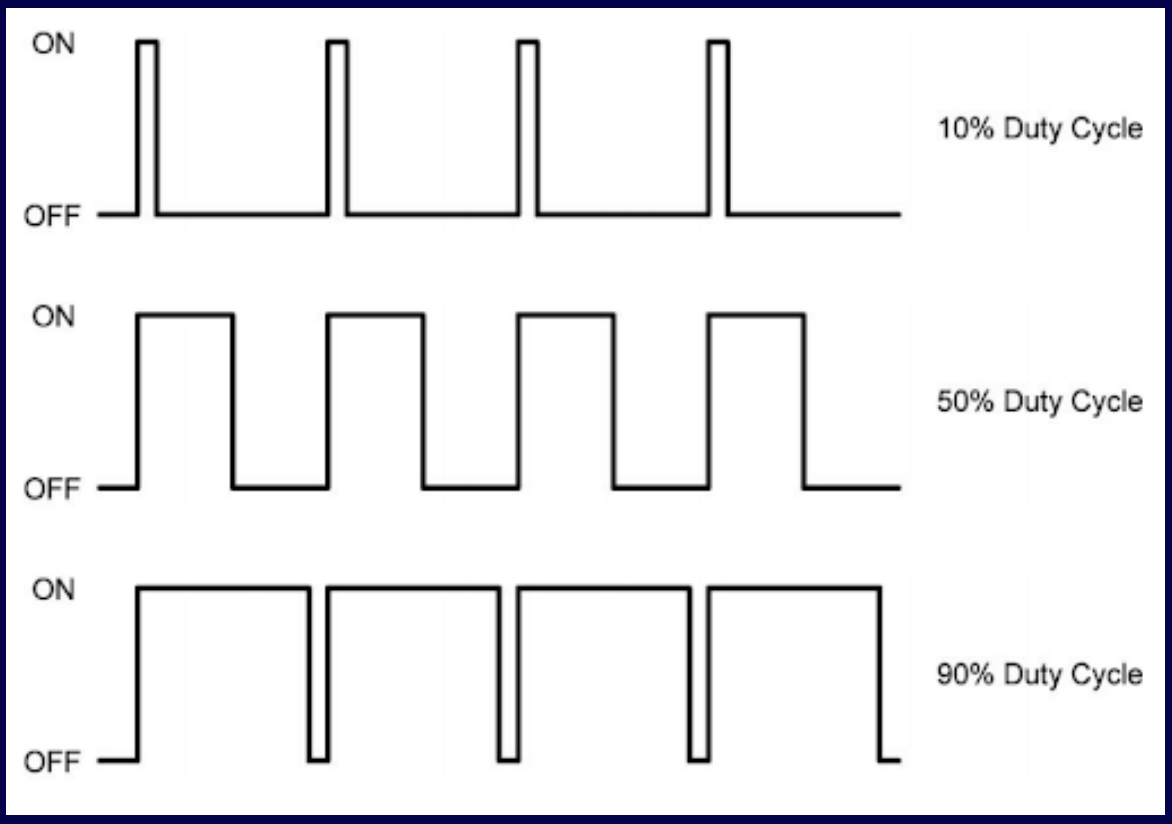
Já o **dimmer** controla a intensidade da lâmpada, girando o potenciômetro conseguimos regular a intensidade de brilho. Esse é considerado o sinal analógico, pois existem infinitos valores.

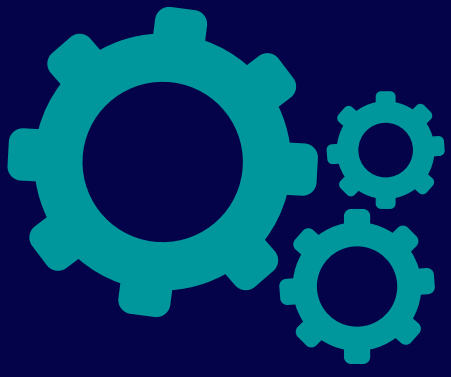




PWM

Outro conceito que aparecerá em nossos projetos é o **PWM**, que significa **Pulse Width Modulation**, traduzindo para o português fica Modulação por Largura de Pulso. Ele nada mais é que uma técnica desenvolvida para obter resultados analógicos por meios digitais. Essa técnica consiste na geração de uma onda quadrada em uma frequência muita alta em que pode ser controlada a percentagem do tempo em que a onda permanece em nível lógico alto. Esse tempo é chamado de **Duty Cycle** e sua alteração provoca mudança no valor médio da onda, indo desde 0V (0% de Duty Cycle) a 5V (100% de Duty Cycle) no caso do **Arduino**.





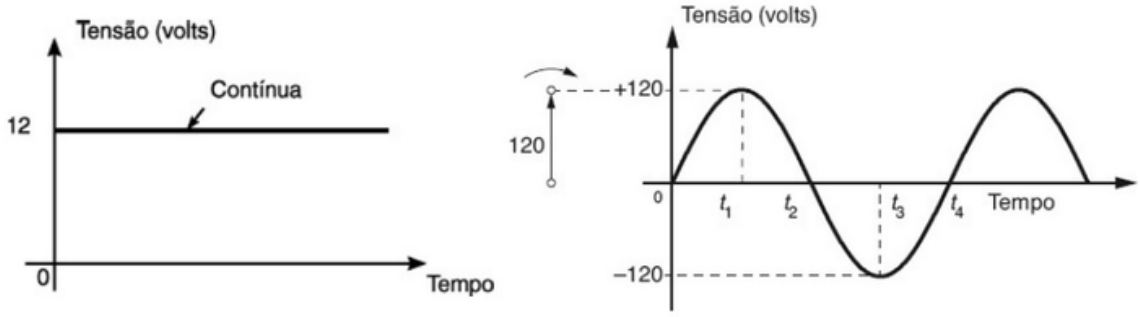
Tensão Elétrica

Também conhecida como **Diferença de Potencial**, a **tensão** é a grandeza que mede a diferença de potencial elétrico entre dois pontos. É a força necessária para mover os elétrons e criar assim uma **Corrente Elétrica**. O equipamento utilizado para medir a tensão é conhecido como **voltímetro**, no **Sistema Internacional** a unidade de medida é **Volts**, onde o símbolo é **V**.



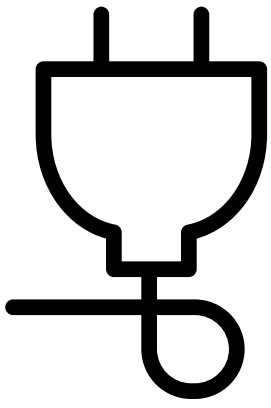
Tipos de Tensão

A tensão pode ser **Contínua** ou **Alternada**. Na contínua ela não muda de polaridade com o passar do tempo, já na alternada é ao contrário, ou seja, muda de polaridade com o passar do tempo.



Exemplos

- A **Contínua** não muda de polaridade com o passar do tempo. A **tensão** das **pilhas** é contínua.
- Já a **tensão Alternada** a polaridade será alternada de acordo com a frequência, no caso de uma tomada no Brasil, a frequência normal é de 60Hz, o que quer dizer que a polaridade desta tensão vai alternar 60 vezes por segundo.



Fórmulas

Podemos calcular a **Tensão Elétrica** utilizando a **Lei de Ohm**, onde a tensão é igual a resistência vezes a corrente elétrica.

Assim, temos a seguinte fórmula:

Tensão = Resistência x Corrente

Resumindo: $U = R \times I$

OBS: **U =** Tensão
R = Resistência
I = Corrente

Também é possível calcular a tensão se caso soubermos o valor da **Corrente** e da **Potência** elétrica, onde tensão elétrica é igual a potência dividida pela corrente.

Assim, temos a seguinte fórmula:

$$\text{Tensão} = \frac{\text{Potência}}{\text{Corrente}}$$

Resumindo: $U = \frac{P}{I}$

OBS: **U =** Tensão
P = Potência
I = Corrente

OBS: O termo **Voltagem** apesar de ser muito falado não existe, é apenas um termo popular.

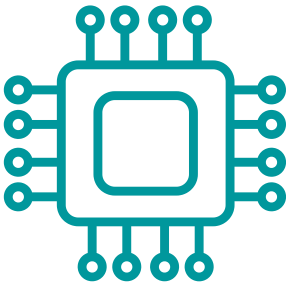
Corrente Elétrica

A **Corrente Elétrica** é um fenômeno físico onde ocorre o movimento de cargas elétricas, como os elétrons, que acontece no interior de diferentes materiais em razão da aplicação de uma diferença de potencial elétrico. A corrente elétrica é uma grandeza escalar, sua unidade de medida de acordo com o Sistema Internacional de Unidades, é o **Ampère**, cujo símbolo é **A**. Essa unidade mede o módulo da carga elétrica que atravessa a secção transversal de um condutor a cada segundo e por isso também pode ser escrita como **Coulombs Por Segundo**, onde o símbolo é **C/s**.

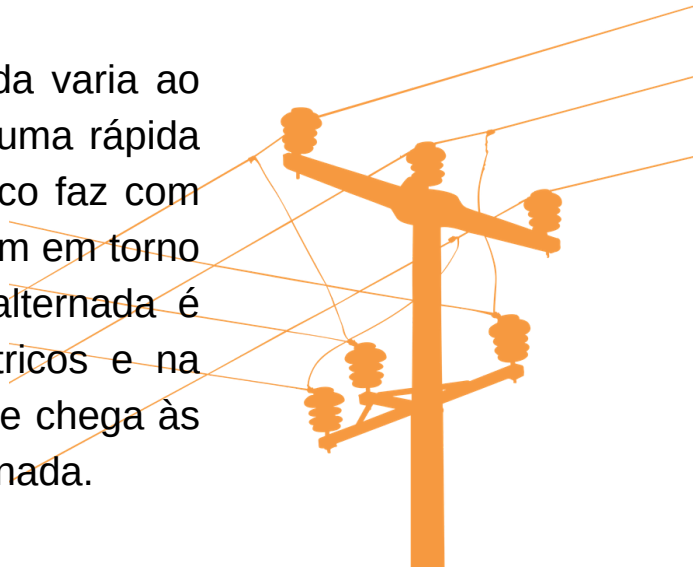
Tipos de Corrente

Assim como a **Tensão Elétrica**, existem dois tipos de corrente, a **Corrente Alternada (CA)** e a **Corrente Contínua (CC)**.

- **Corrente Contínua:** Não tem variação ao longo do tempo e se mantém praticamente constante, os elétrons são forçados a deslocar-se em sentido único. Esse tipo de corrente é comum em dispositivos que utilizam baixas tensões, como eletrônicos em geral.



- **Corrente Alternada:** A corrente alternada varia ao longo do tempo. Nesse tipo de corrente, uma rápida inversão de polaridade do potencial elétrico faz com que os elétrons se desloquem em vai e vem em torno de uma posição fixa. Corrente elétrica alternada é utilizada principalmente em motores elétricos e na transmissão de eletricidade, a corrente que chega às nossas casas é uma corrente elétrica alternada.



Fórmulas

Para se saber qual a intensidade da **Corrente Elétrica** em um condutor, é possível utilizar a equação de carga elétrica pelo tempo.

Assim, temos a seguinte fórmula:

$$\text{Intensidade de Corrente} = \frac{\text{Carga elétrica}}{\text{Intervalo de tempo}}$$



Resumindo: $I = \frac{Q}{\Delta t}$

OBS: **Q =** Carga elétrica
Δt = Intervalo de tempo
I = Corrente

Utilizando a **Tensão Elétrica** e a **Resistência** também é possível calcularmos a corrente, onde corrente é igual tensão elétrica dividido pela resistência.

Assim, temos a seguinte fórmula:

$$\text{Corrente} = \frac{\text{Tensão}}{\text{Resistência}}$$



Resumindo: $I = \frac{U}{R}$

OBS: **U =** Tensão
R = Resistência
I = Corrente

Resistência Elétrica

A **Resistência Elétrica** é definida como a capacidade que um corpo tem de opor-se à passagem da **Corrente Elétrica**. A unidade de medida da resistência no **Sistema Internacional** é o **Ohm (Ω)**, e representa a razão **volt/Ampére**.

Fórmulas

Para descobrir a **Resistência Elétrica** podemos utilizar a **Primeira Lei de Ohm**.

Assim, temos a seguinte fórmula:

$$\text{Resistência} = \frac{\text{Tensão}}{\text{Corrente}}$$

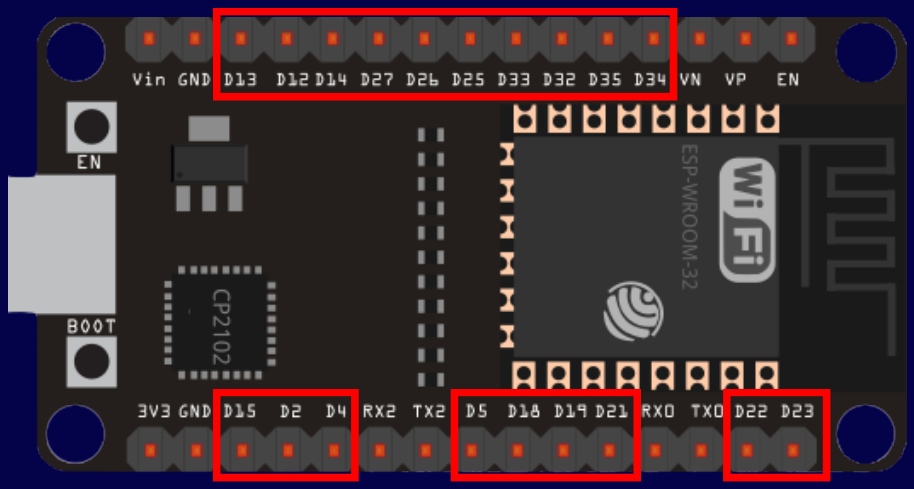
Resumindo: $R = \frac{U}{I}$

OBS: **U =** Tensão
R = Resistência
I = Corrente

3 Composição do ESP-32

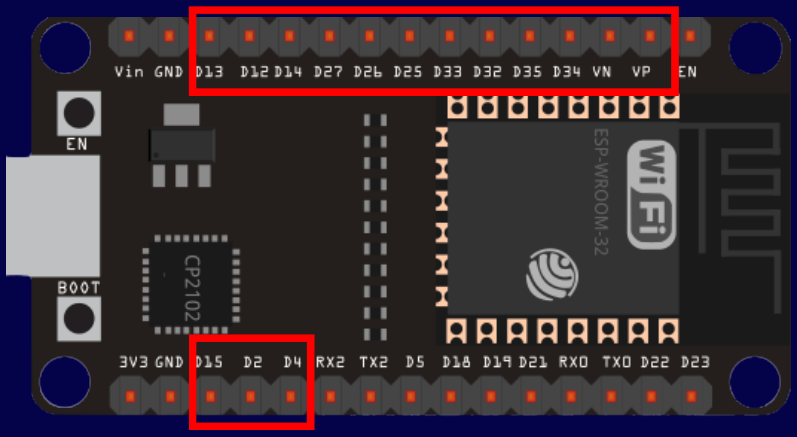
Bem, agora que já conhecemos os principais conceitos que vão ser utilizados, vamos conhecer um pouco mais sobre o **NodeMCU**.

Pinos Digitais



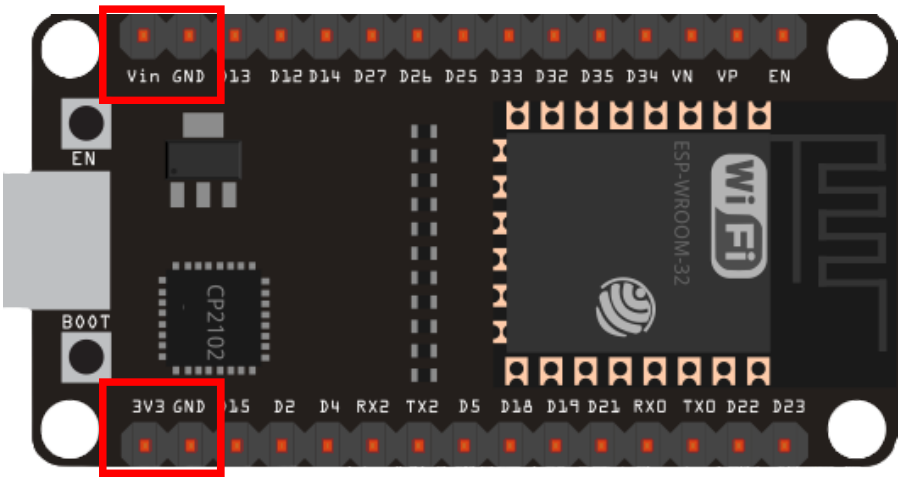
As portas **D2** a **D35** são os pinos de entradas e saídas digitais, usadas para conexões de **módulos** e **sensores**.

Pino Analógico



Os pinos **D13** a **VP** e **D15** a **D4** funcionam como pinos analógicos

Pinos para Alimentação

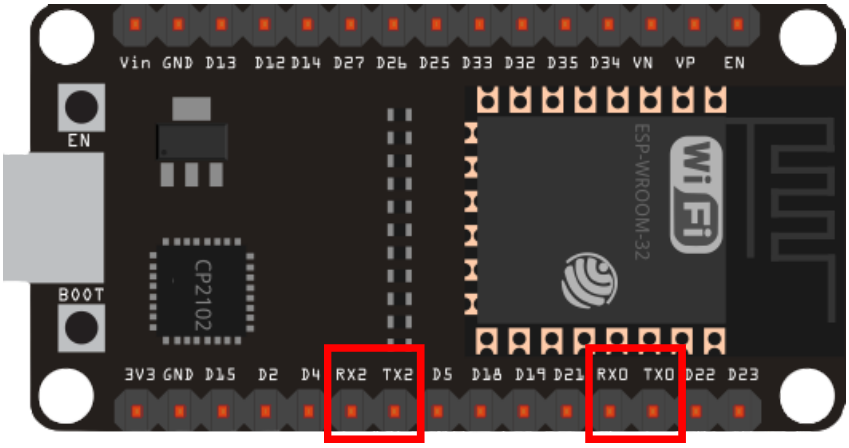


Pinos GND: É o terra que funciona como o negativo para todos os circuitos e dispositivos externos. O GND é comum a todas as portas, ou seja, ele é compartilhado, diferente das portas digitais.

Pino 3V3: Pino que fornece 3,3 volts para a alimentação de dispositivos externos.

Pino Vin: A porta Vin fornece a mesma tensão que é recebida pelo pino de alimentação externo.

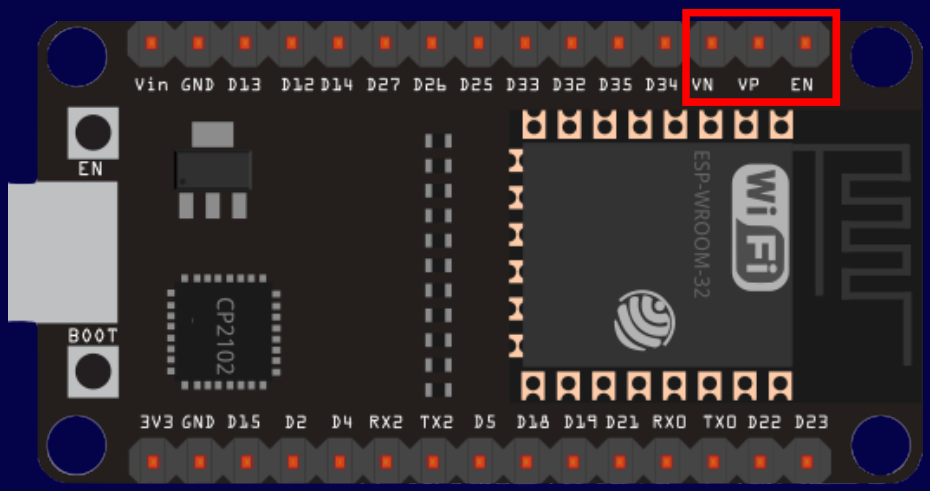
Pinos para Comunicação



RX: Receiver, recebe bits.
TX: Transmitter, envia bits.

São as portas utilizada para fazer a **Comunicação Serial**.

Outros Pinos



Pino EN: O chip ESP32 é habilitado quando o pino EN está em nível lógico **alto**. Quando está em nível lógico **baixo**, o chip funciona com a potência mínima.

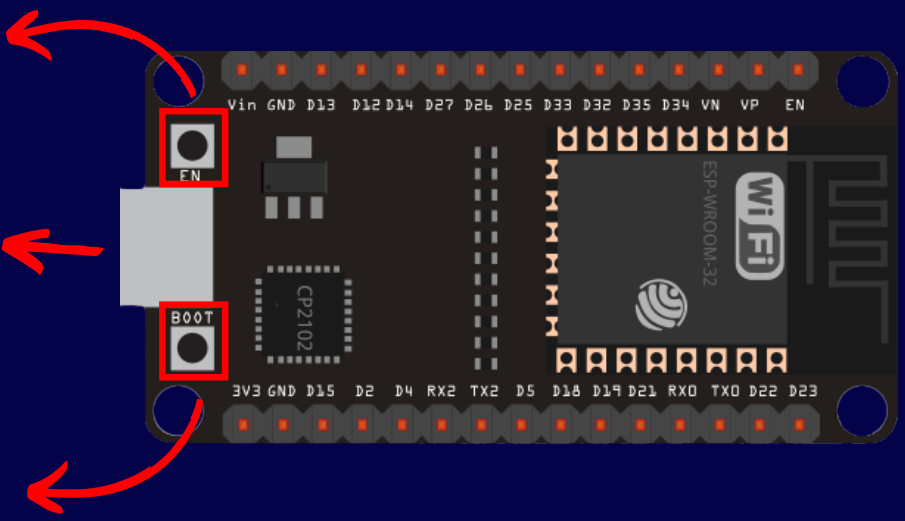
Pino VP: Sensor VP.

Pino VN: Sensor VN.

EN: É usado para reiniciar o NodeMCU.

Entrada Micro Usb: Usada para a alimentação da placa e para enviar a programação.

BOOT: permite a gravação do programa no ESP32.

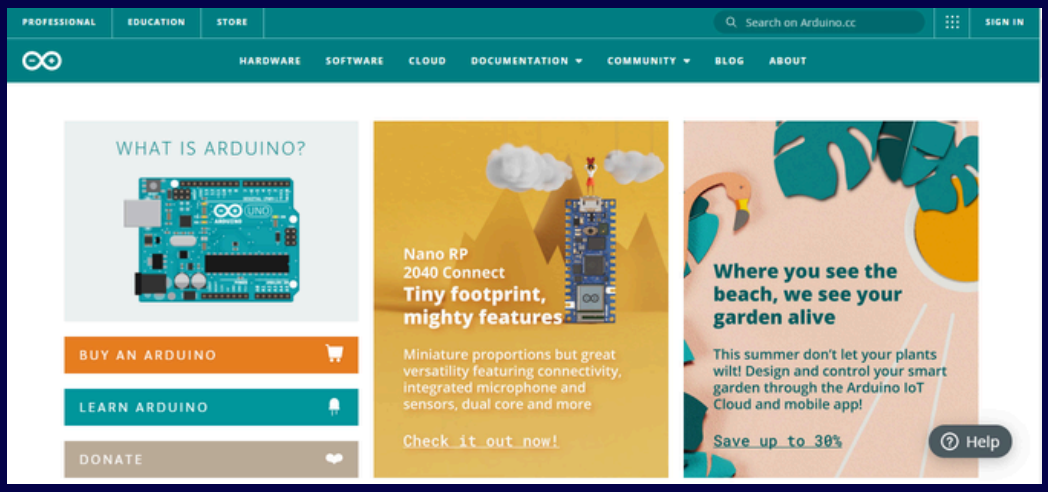


4 Baixando a IDE do Arduino

Para **programarmos** nossa placa vamos utilizar a **IDE do Arduino** !

Clique Aqui para baixar a **IDE do Arduino**. Caso tenha dúvidas, siga o passo a passo abaixo.

1 Acesse o site oficial do Arduino Clicando aqui.



2 Clique em **Software**.



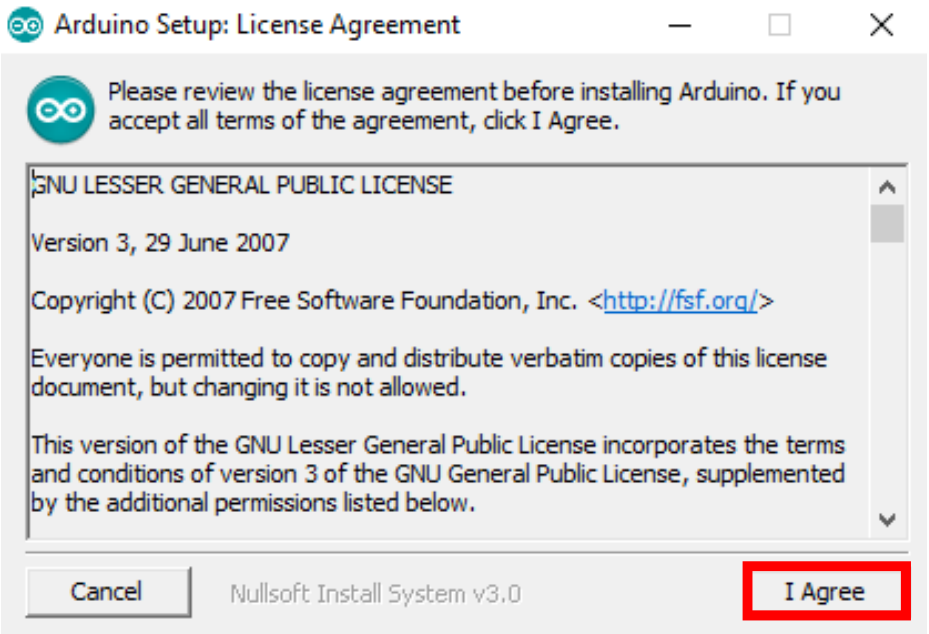
Aqui é onde vamos fazer o Download da **IDE do Arduino**. Selecione qual sistema operacional você utiliza.

Caso possua **Windows 7** ou uma **versão superior**, selecione a primeira opção: **Windows Win 7 and newer**.

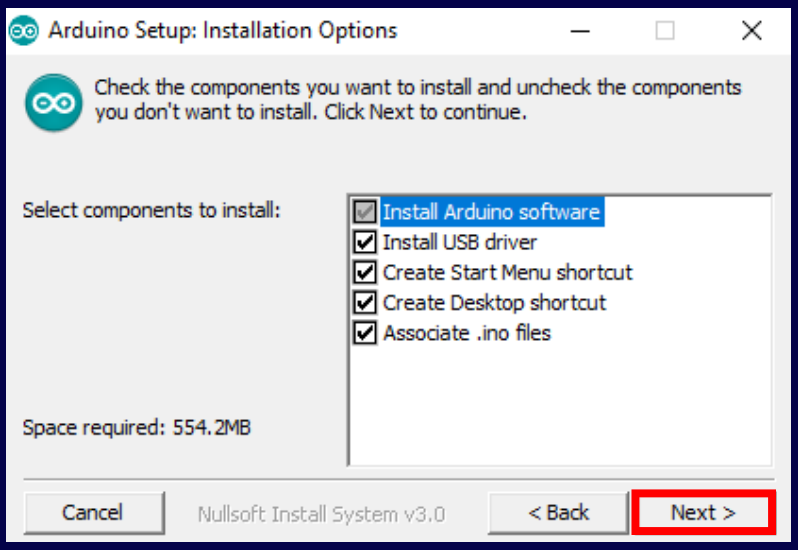


Se quiser, você pode fazer uma doação para ajudar o desenvolvimento da **Arduino IDE**. Caso contrário, basta clicar em **Just Download** e o programa começará a baixar.

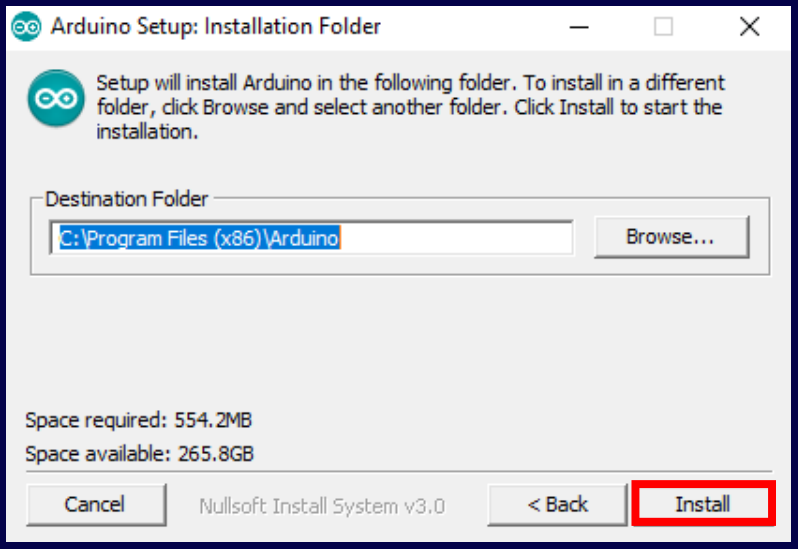
3 Após o Download do instalador da IDE terminar, **clique duas vezes sobre o ícone** gerado para abri-lo.



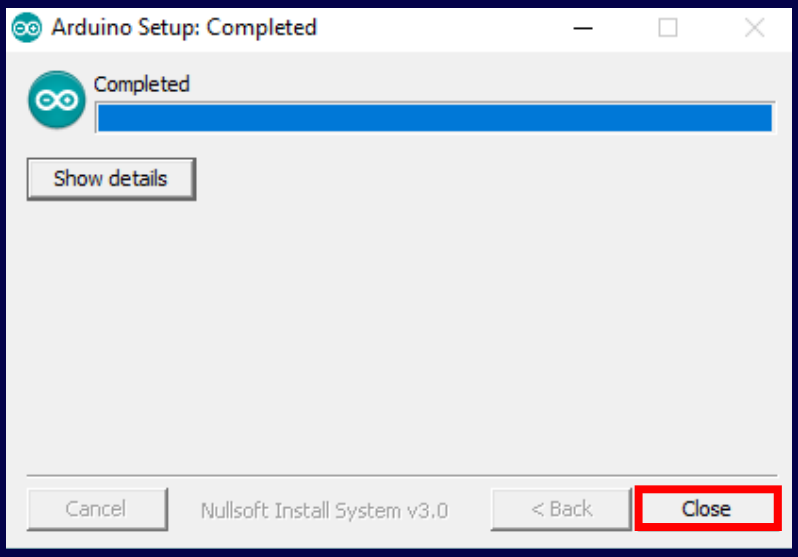
Clique em **I Agree**.



Selecione todas as caixas e clique em **Next**.



Escolha o local onde a IDE vai ser instalada em seu computador e clique em **Install**. A instalação começará.



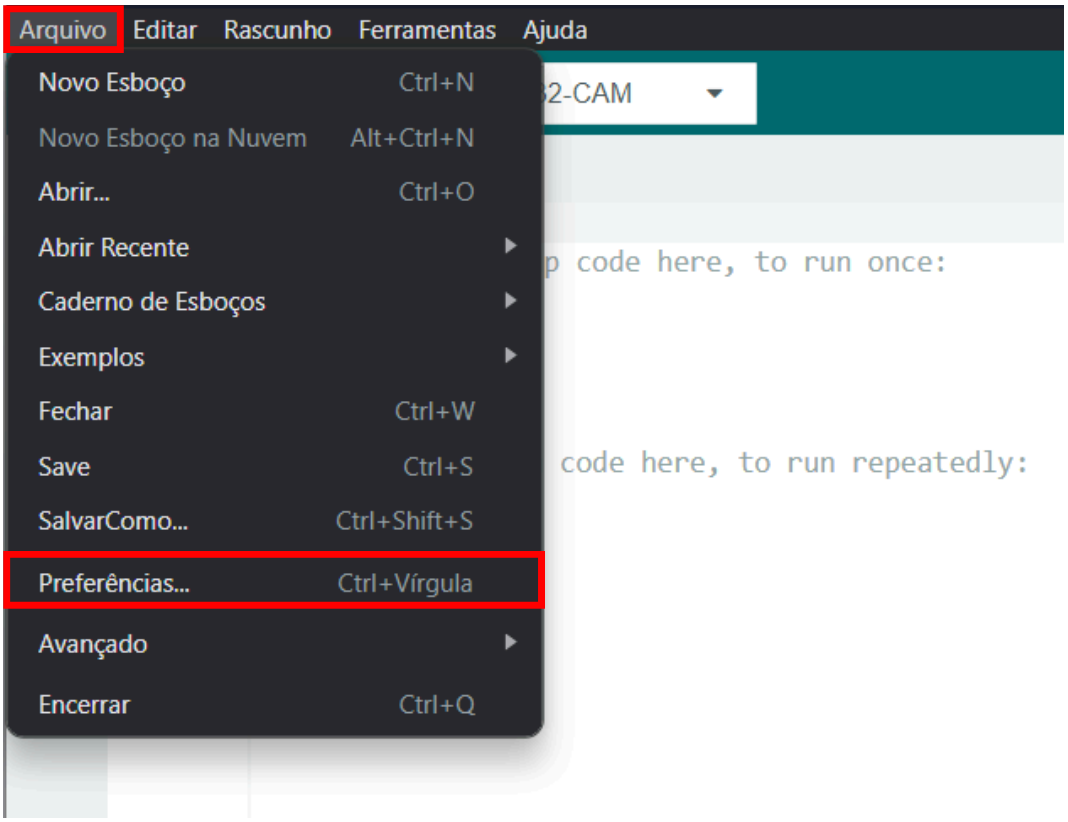
Quando a instalação for concluída clique em **Close**.

Pronto, agora é só programar !

5 Configurando o ESP32 na IDE

Precisamos configurar a **IDE do Arduino** para conseguirmos programar o **ESP32**.

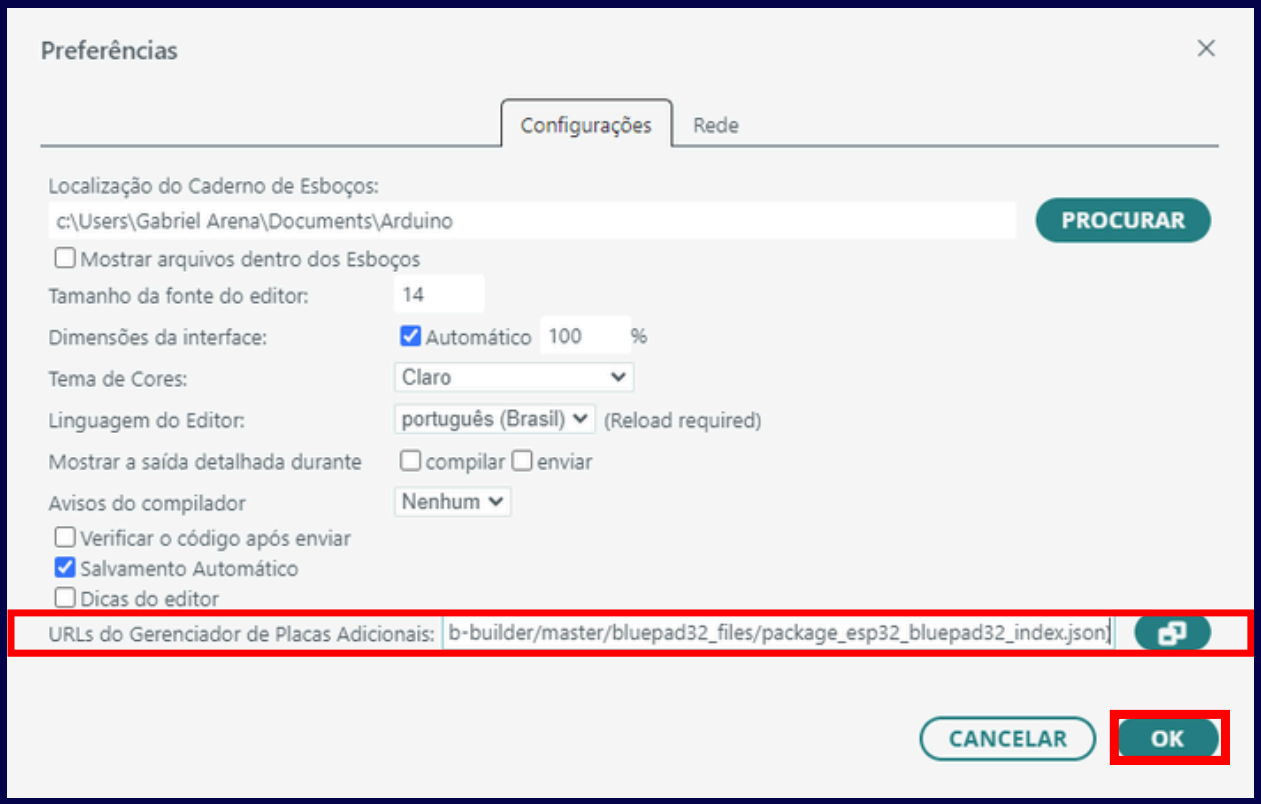
1 Abra a **IDE do Arduino** e clique em **Arquivo** e depois em **Preferências**.



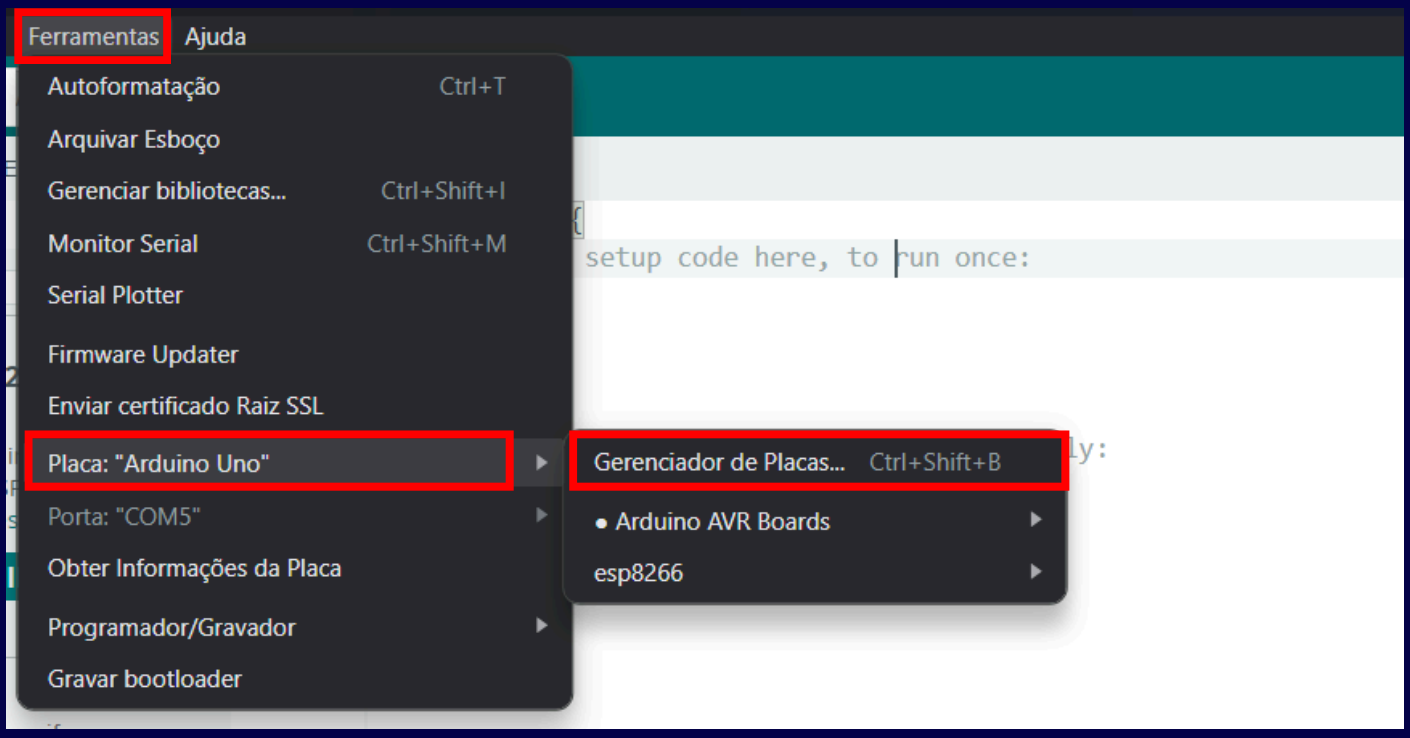
2 Copie o link:

https://raw.githubusercontent.com/ricardoquesada/esp32-arduino-lib-builder/master/bluepad32_files/package_esp32_bluepad32_index.json

Cole no campo **URLs adicionais de Gerenciadores de Placas**, depois clique em **OK**.



3 Clique em **Ferramenta**, depois em **Placa** e em **Gerenciador de Placas**.

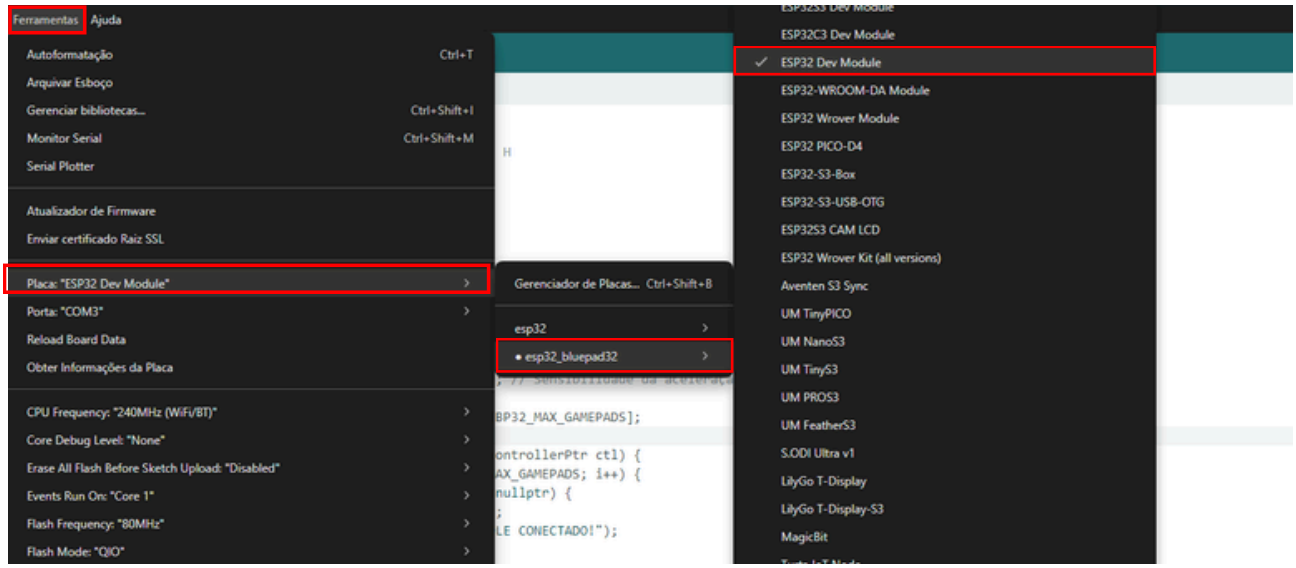


Busque na **barra de pesquisa** por **esp32_bluepad32**. selecione a opção feita por **Ricardo Quesada**.



Clique em **Instalar**.

Clique em **Ferramentas**, depois em **Placa**, selecione **ESP32_bluepad32** e clique em **ESP32 Dev Module**.



6 Ponte H L298N

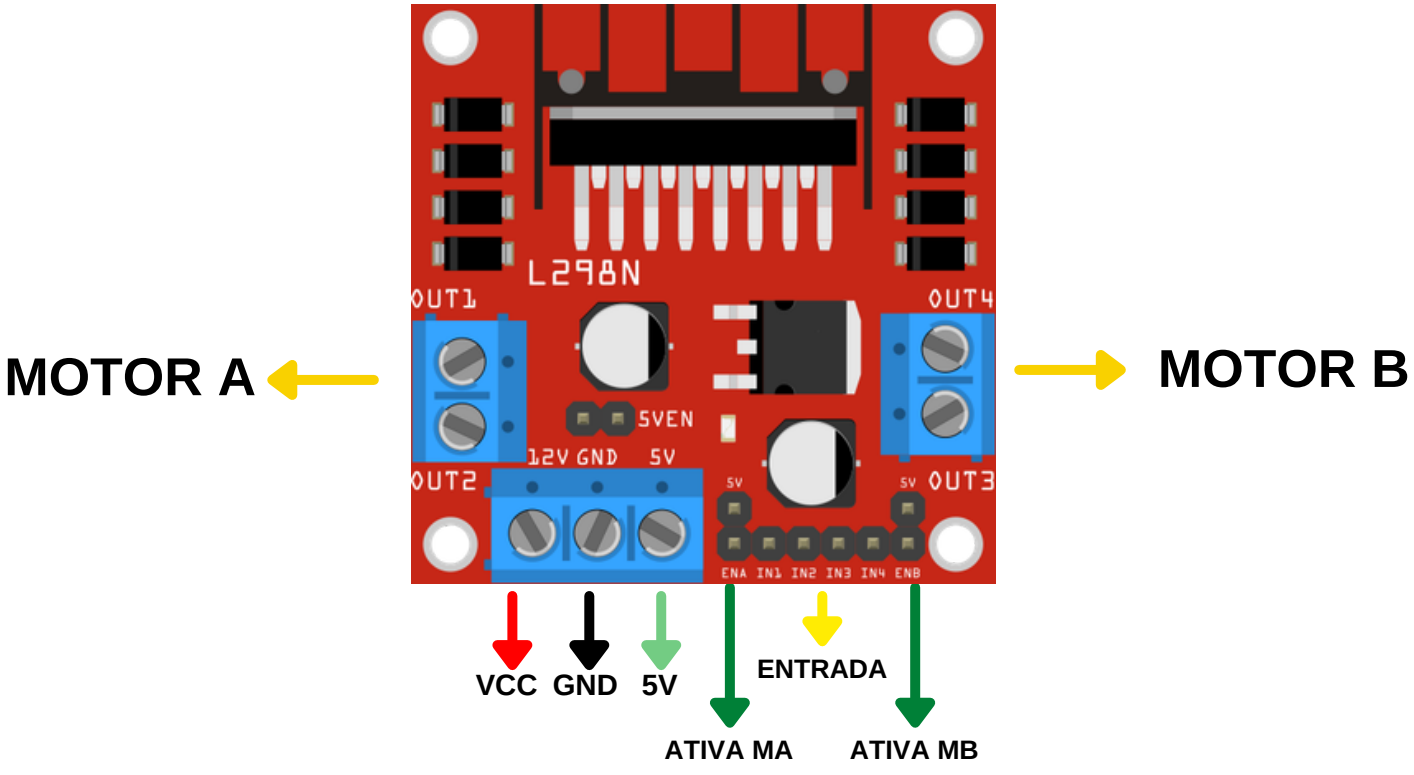
Em muitos projetos de **robótica**, é necessário utilizar **motores DC** (corrente contínua) para realizar movimentos. No entanto, esses motores são frequentemente **cargas indutivas** e exigem uma quantidade de corrente que ultrapassa a capacidade das portas do ESP. Portanto, é importante não conectar um motor que exige uma **corrente superior a 40 mA** diretamente nas portas digitais do ESP, pois isso pode danificar a placa. Para lidar com a alta corrente exigida pelos motores, é necessário utilizar transistores. A **Ponte H** é um circuito composto por **4 transistores** que permite controlar motores, não apenas em relação à sua direção, mas também em relação às suas velocidades. Além disso, a utilização da **Ponte H** é relativamente simples no ESP.



Vantagens de utilizar

O circuito integrado **L298N** é muito utilizado para controlar motores. Uma das vantagens do uso deste circuito é o menor espaço ocupado, a baixa complexidade do circuito e o fato de ele já possuir dois circuitos H, podendo assim, controlar dois motores. Outra vantagem do **L298N** é a resposta a sinais de **PWM**. Se forem utilizados sinais de **PWM**, é possível regular a tensão de saída, e dessa forma regular a velocidade dos motores.

Conhecendo o funcionamento...



- **Motor A e Motor B:** Borne para conexão dos motores DC.
- **VCC:** Bornes responsáveis pela alimentação dos motores.
- **GND:** É o GND da placa, que deverá ser o mesmo GND do Arduino. Por isso, é importante interligar os GNDs sempre com fios.

- **5V:** Em casos de não haver fonte de alimentação com mais de 6V podemos alimentar a placa com 5V por esta porta.

- **(Ativa MA) e (Ativa MB):** são os pinos responsáveis pelo controle PWM dos motores A e B. Se estiver com jumper, não haverá controle de velocidade.

- **(IN1 e IN2) e (IN3 e IN4):** São utilizados para controlar o sentido dos motores.

- **Ativa 5V:** É o Jumper responsável por regular a alimentação da **Ponte H**, caso o jumper esteja conectado, a alimentação deve ser de **6** a **12V**. Se o jumper está desconectado, é preciso utilizar a porta que fornece **5V** do ESP, porém é necessário se atentar a **tensão** do motor utilizado e a **corrente**, **caso contrário o ESP pode queimar**.

Controle dos motores

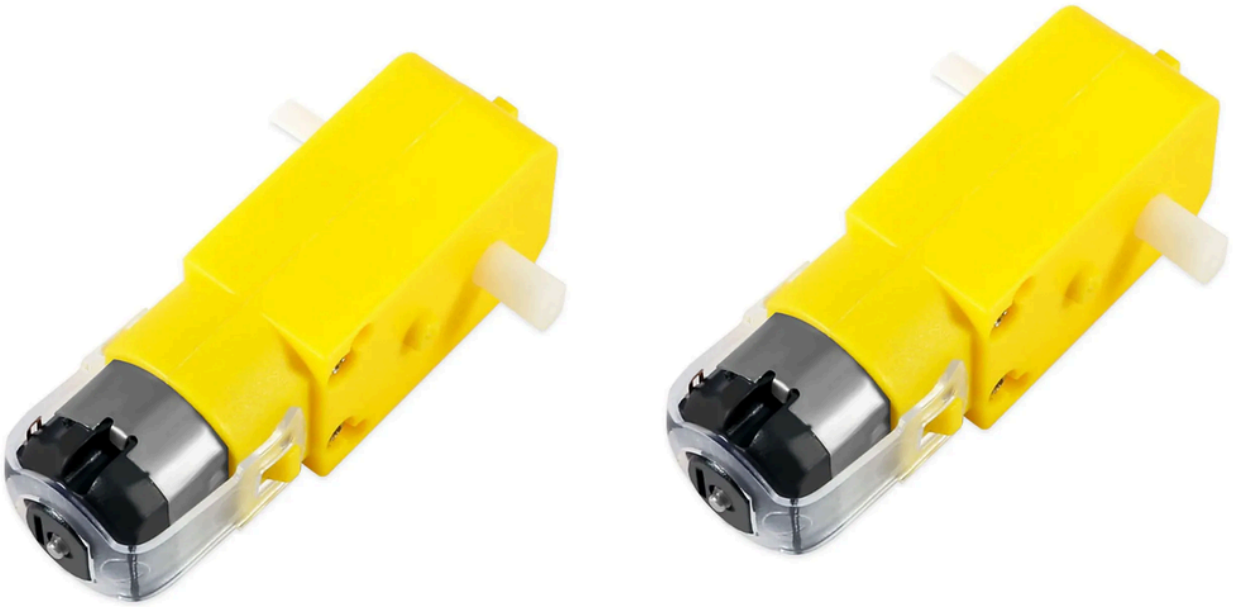
Para o **controle dos motores**, é necessário acionar os pinos da seguinte maneira:

Motor A	IN1	IN2
Horário	High (5V)	Low (GND)
Anti Horário	Low (GND)	High (5V)
Freio	Low (GND)	Low (GND)
Freio	High (5V)	High (5V)

A mesma forma se aplica para o motor **B**, **IN3** e **IN4**.

7 Motor DC

Um **Motor DC** é um dispositivo que converte energia elétrica de **corrente contínua** em energia mecânica. Esses motores são compostos por ímãs permanentes ou por um campo e armadura, que não possuem ímãs permanentes. Os motores de **corrente contínua** são amplamente utilizados e possuem diversas aplicações, incluindo brinquedos, eletrodomésticos, máquinas industriais, veículos elétricos e muitos outros. Os **motores DC** são especialmente populares em aplicações que exigem controle preciso de velocidade ou torque. Isso ocorre porque a velocidade de rotação do motor pode ser facilmente controlada variando a tensão ou corrente que é fornecida ao motor. Além disso, a simplicidade de design e a facilidade de manutenção tornam esses motores uma opção atraente em muitas aplicações.



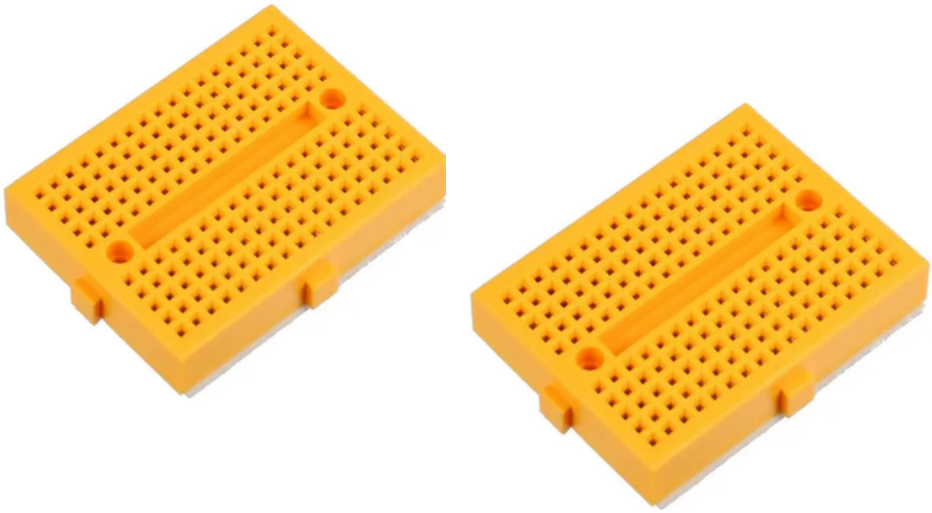
Em resumo, os **motores DC** são uma escolha popular em muitas áreas, devido à sua ampla gama de aplicações, controle preciso de velocidade e torque, simplicidade de design e facilidade de manutenção.

Lembre-se: é necessário soldar os fios nos motores. O passo a passo para esse procedimento está disponível no vídeo da **página 27.**

8 Outros Componentes

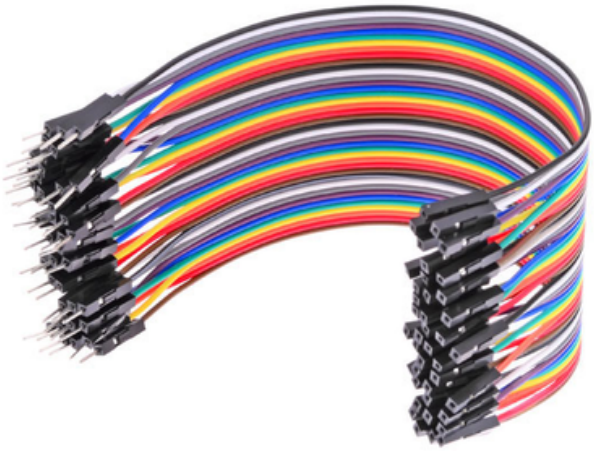
Protoboard

Nada mais é do que uma placa com furos e conexões condutoras utilizada para a montagem de protótipos e projetos que estão na fase inicial. A vantagem de se utilizar uma **Protoboard** é que podemos montar os componentes direto nele, assim eliminamos a necessidade de utilizar a solda.



Jumpers

Os Jumpers nada mais são do que condutores para ligar um ponto a outro.



9 Montagem do Tanque

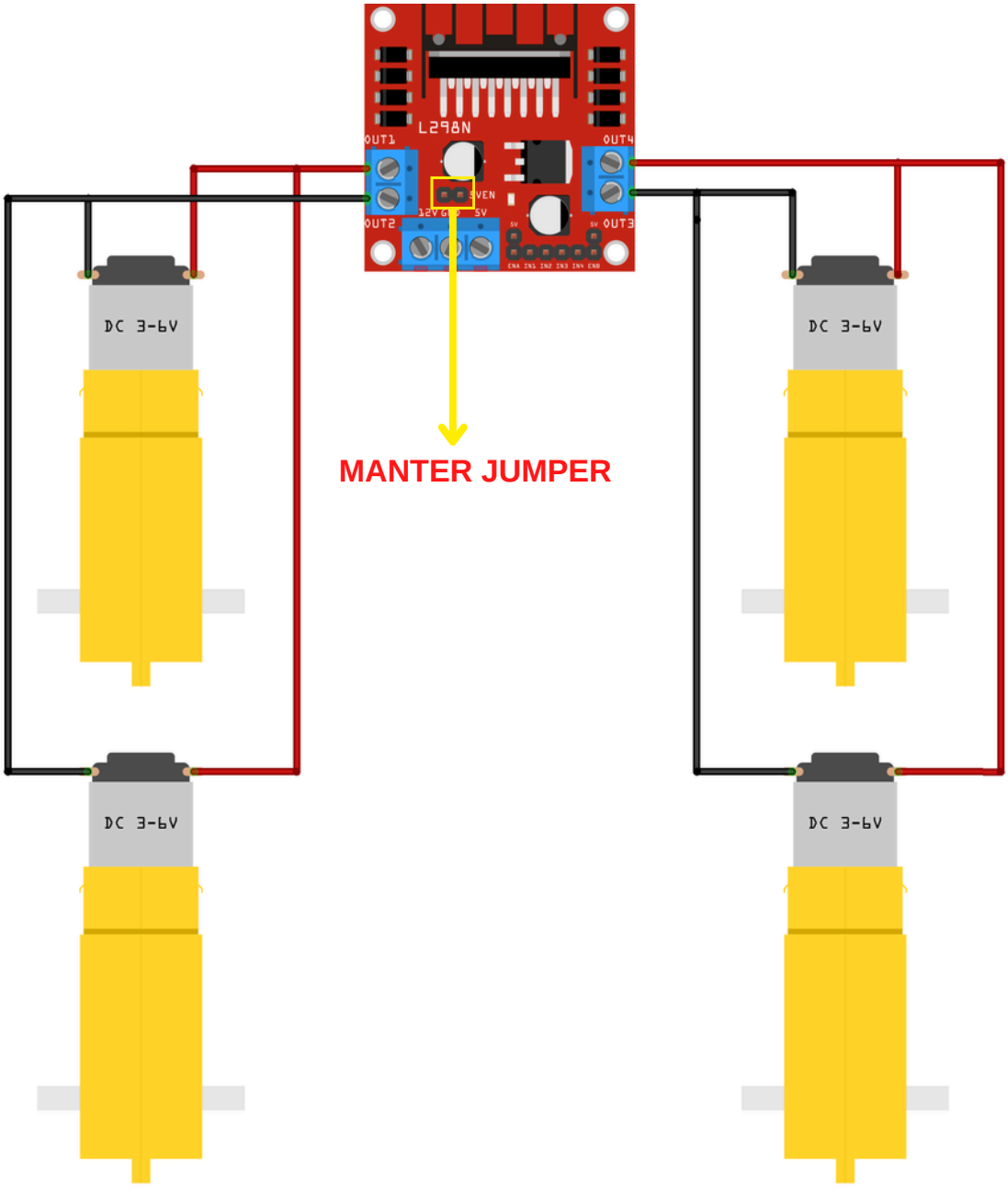
Para a montagem do tanque, assista ao nosso vídeo com o passo a passo completo:



10 Circuito

Ligação dos Motores

Lembre-se que os fios dos motores devem estar ligados **iguais ao da imagem**, na mesma ordem, para que robô ande no **sentido correto**.



OUT1: Fios **positivos** dos motores do lado esquerdo.

OUT2: Fio **negativo** dos motor do lado esquerdo.

OUT3: Fio **negativos** do motores do lado direito.

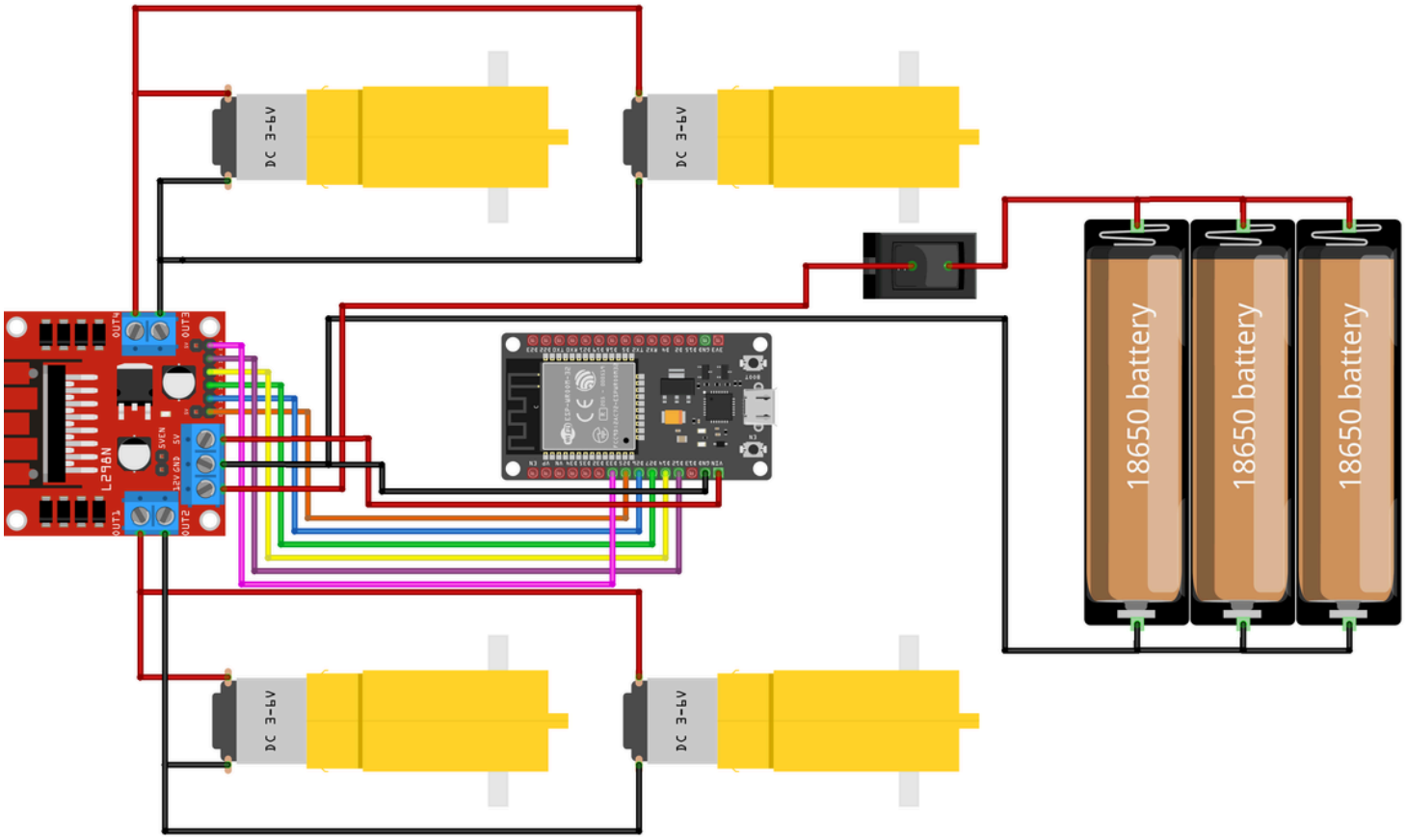
OUT4: Fios **positivos** dos motores do lado direito.

Caso os motores **não rodem no sentido correto**, basta inverter as ligações na Ponte H.



ATENÇÃO: Faça todas as ligações do tanque com a bateria desligada, caso contrário o robô pode ser danificado ! Sempre confira se as tensões estão corretas.

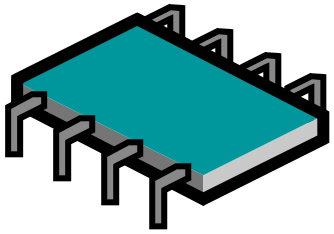
Alimentação e ligação no ESP32



Para a alimentação do robô, é necessário conectar o **positivo** da bateria na porta **12v** da ponte H, o **negativo** da bateria deve estar no **GND** da ponte e no **GND** do **ESPCAM**. A porta **5v** da ponte H deve estar no pino **Vin** do **ESP32**.

A ligação no dos pinos da ponte h no **ESPCAM** é a seguinte**:**

- **IN1:** GPIO **26**
- **IN2:** GPIO **27**
- **IN3:** GPIO **14**
- **IN4:** GPIO **12**
- **ENA:** GPIO **25**
- **ENB**: GPIO **33**



Atenção !

Por consumir muita energia, esse robô só funciona corretamente com o uso de **baterias Li-ion, com tensão de 12V**.





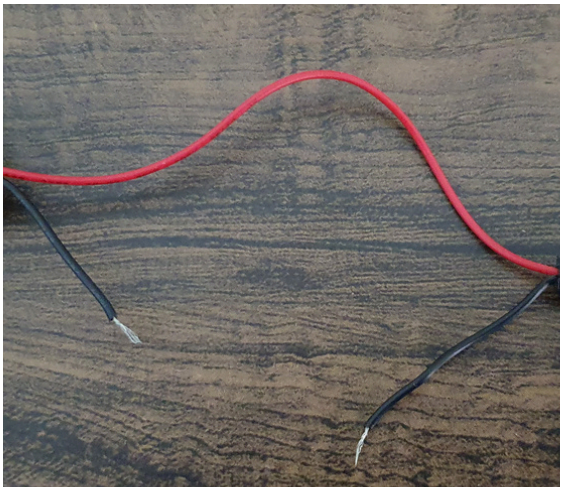
ATENÇÃO: O uso inadequado da bateria pode resultar em curto-circuitos, o que pode causar incêndios e representar sérios riscos de segurança. Por favor, manuseie com cuidado.

CARREGAMENTO

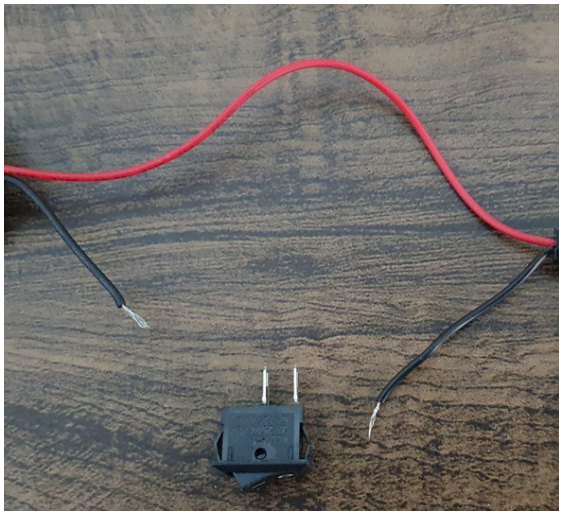
O modelo de bateria que acompanha o tanque pode ser **carregada** através de **carregadores específicos** para baterias **18650**.

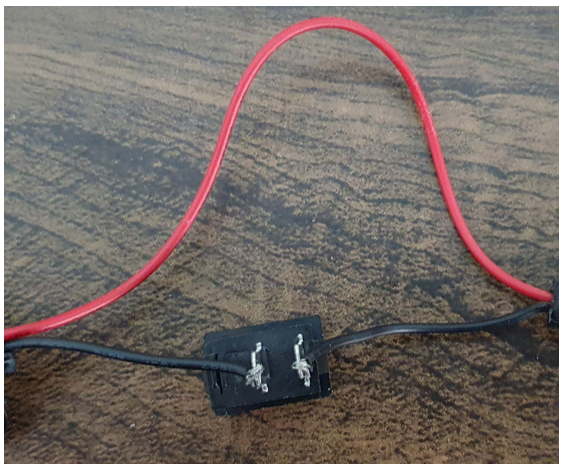


Para ligar e desligar o robô, vamos usar uma chave, para isso será necessário realizar um **corte no fio negativo** ou **positivo** dos fios da bateria e decapar as duas pontas.

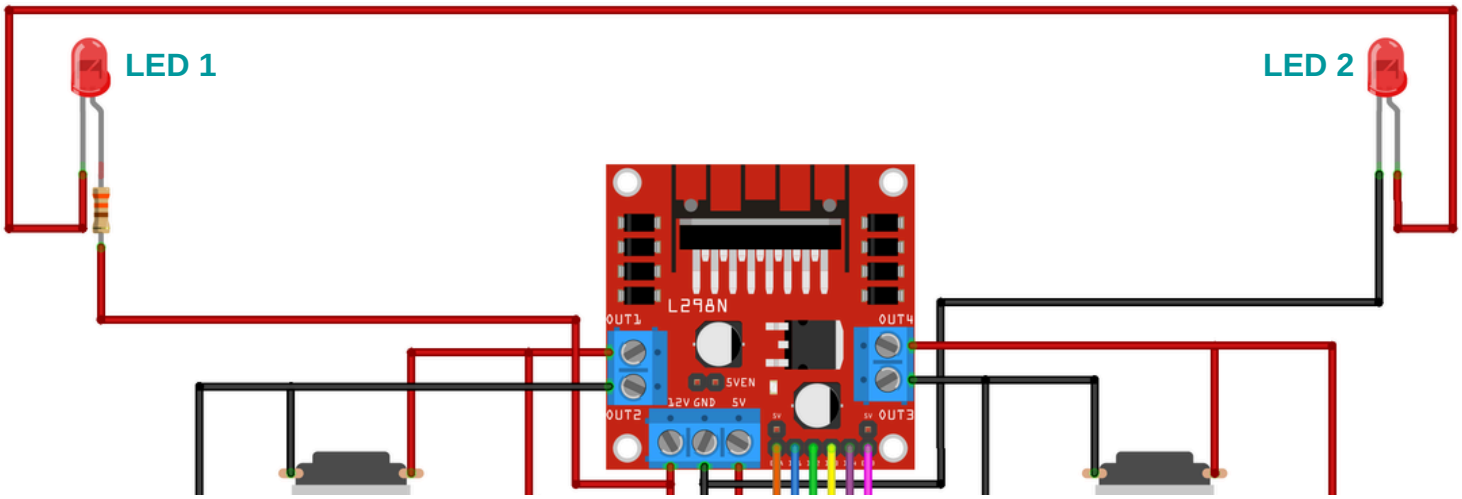


Agora, é necessário encaixar os fios decapados na **chave**.





FARÓIS



Para a montagem dos faróis, as seguintes conexões devem ser feitas em série:

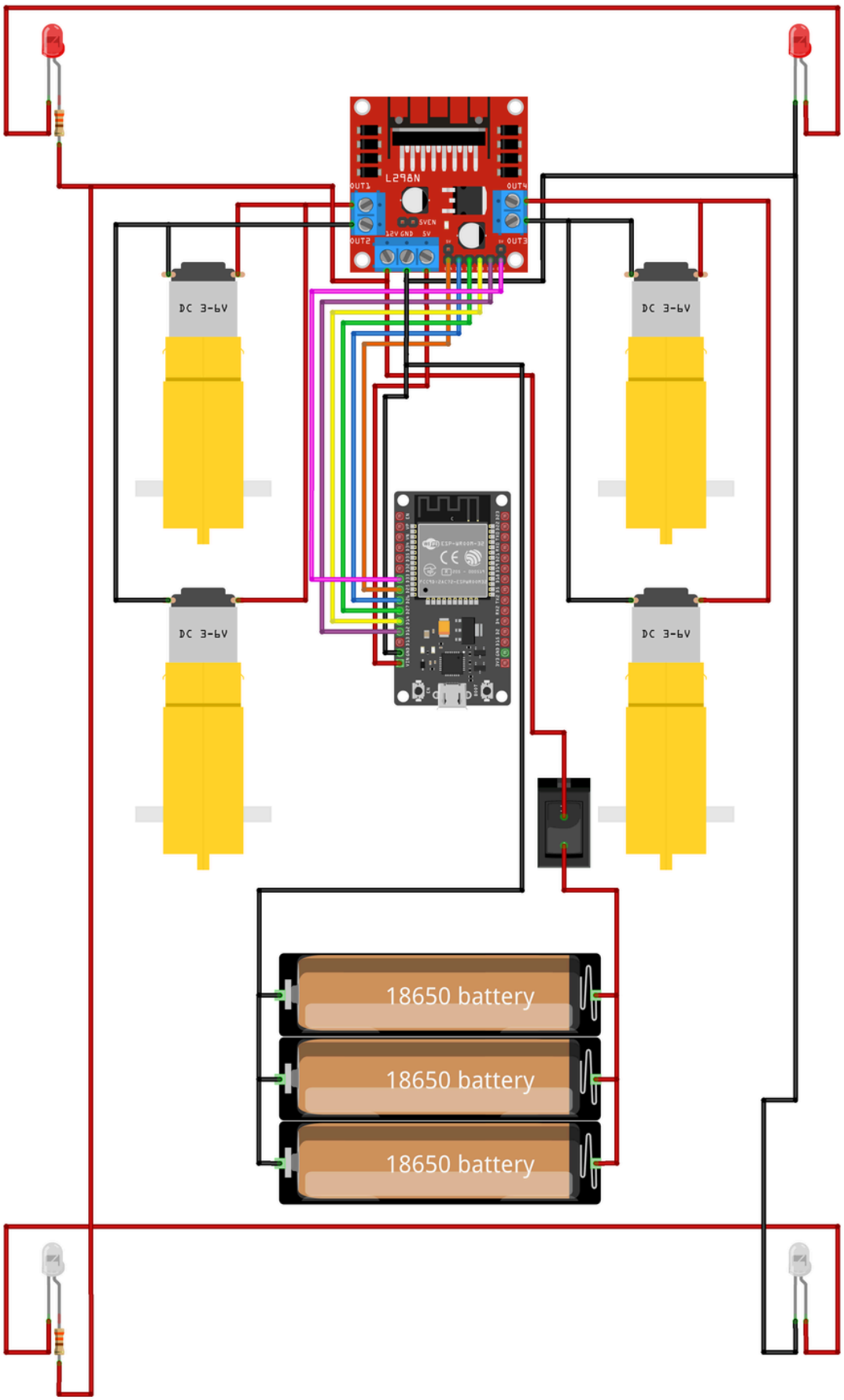
- Conecte o **terminal positivo** do **LED 1** a uma das pontas do **resistor de 330Ω**, e a outra ponta do resistor à saída de **12V** da Ponte H.
- Conecte o t**erminal negativo** do **LED 1** ao **terminal positivo** do **LED 2**.
- Conecte o **terminal negativo** do **LED 2** ao pino **GND** da Ponte H.

Dica: O terminal negativo (cátodo) é sempre a "perna" mais curta do LED.

Repita esse mesmo procedimento para os demais LEDs da parte frontal do robô.



Circuito Completo



11 Código

```
#include <Bluepad32.h>

// Pinos de Direção da Ponte H
const int IN1 = 26;
const int IN2 = 27;
const int IN3 = 14;
const int IN4 = 12;

// Pinos de Velocidade (PWM)
const int ENA = 25;
const int ENB = 33;

// --- CONFIGURAÇÃO DA RAMPA DE ACELERAÇÃO ---
float velAtual = 0;
float fatorSuavizacao = 0.05; // Sensibilidade da aceleração

ControllerPtr myControllers[BP32_MAX_GAMEPADS];

void onConnectedController(ControllerPtr ctl) {
  for (int i = 0; i < BP32_MAX_GAMEPADS; i++) {
    if (myControllers[i] == nullptr) {
      myControllers[i] = ctl;
      Serial.println("CONTROLE CONECTADO!");
      break;
    }
  }
}

void onDisconnectedController(ControllerPtr ctl) {
  for (int i = 0; i < BP32_MAX_GAMEPADS; i++) {
    if (myControllers[i] == ctl) {
      myControllers[i] = nullptr;
      Serial.println("Controle Desconectado.");
      break;
    }
  }
}

void setup() {
  Serial.begin(115200);

  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(ENA, OUTPUT);
  pinMode(ENB, OUTPUT);

  BP32.setup(&onConnectedController, &onDisconnectedController);
  BP32.forgetBluetoothKeys();
}

void loop() {
  BP32.update();
  ControllerPtr myController = myControllers[0];

  if (myController && myController->isConnected()) {

    int aceleraBruto = map(myController->throttle(), 0, 1023, 0, 255);
    int reBruto = map(myController->brake(), 0, 1023, 0, 255);

    int velocidadeAlvo = aceleraBruto - reBruto;

    // --- CORREÇÃO: PARADA IMEDIATA ---
    if (velocidadeAlvo == 0) {
      velAtual = 0; // Se soltar o gatilho, corta a velocidade na hora
    } else {
      // A rampa atua apenas enquanto você estiver segurando o gatilho
      velAtual = velAtual + ((velocidadeAlvo - velAtual) * fatorSuavizacao);
    }

    int velocidade = (int)velAtual;

    int direcao = map(myController->axisX(), -511, 512, 127, -127);
    if (abs(direcao) < 15) { direcao = 0; }

    int velEsquerda = velocidade + (direcao * 2);
    int velDireita = velocidade - (direcao * 2);

    velEsquerda = constrain(velEsquerda, -255, 255);
    velDireita = constrain(velDireita, -255, 255);

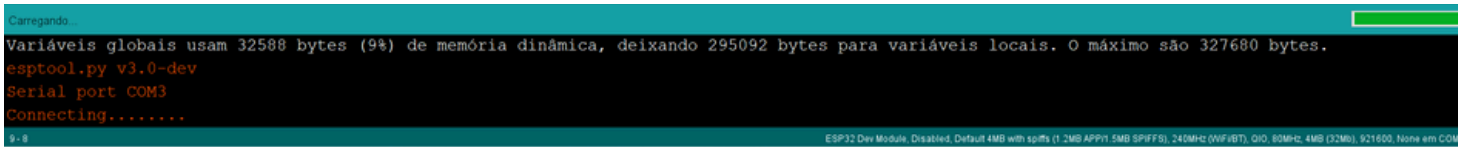
    // --- LADO ESQUERDO ---
    if (velEsquerda > 0) {
      digitalWrite(IN1, HIGH);
      digitalWrite(IN2, LOW);
```

```
    analogWrite(ENA, velEsquerda);
  } else if (velEsquerda < 0) {
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
    analogWrite(ENA, abs(velEsquerda));
  } else {
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, LOW);
    analogWrite(ENA, 0);
  }

  // --- LADO DIREITO ---
  if (velDireita > 0) {
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
    analogWrite(ENB, velDireita);
  } else if (velDireita < 0) {
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, HIGH);
    analogWrite(ENB, abs(velDireita));
  } else {
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, LOW);
    analogWrite(ENB, 0);
  }

  delay(10);
 } else {
  analogWrite(ENA, 0);
  analogWrite(ENB, 0);
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  digitalWrite(IN3, LOW);
  digitalWrite(IN4, LOW);
  velAtual = 0;
 }
}
```

Atenção !!



Quando aparecer a mensagem "**connecting**" na IDE, pressione por alguns segundo a tecla **BOOT** do **ESP-32**.

Baixe o código do projeto **Clicando Aqui!**

Como Parear o Controle de PS4

Para conectar o controle ao **Tanque** pela primeira vez, o processo é automático e leva apenas alguns segundos:

1- Ligue a alimentação do tanque para iniciar o **ESP32**.

2- Com o controle desligado, pressione e segure simultaneamente os botões **SHARE** e **PS** (o botão redondo central).



3- Continue segurando até que a barra de luz frontal comece a piscar rapidamente com pequenos flashes.

4- Aguarde. O ESP32 fará a busca e se conectará automaticamente. Quando a luz do controle parar de piscar e ficar fixa, o pareamento foi um sucesso!

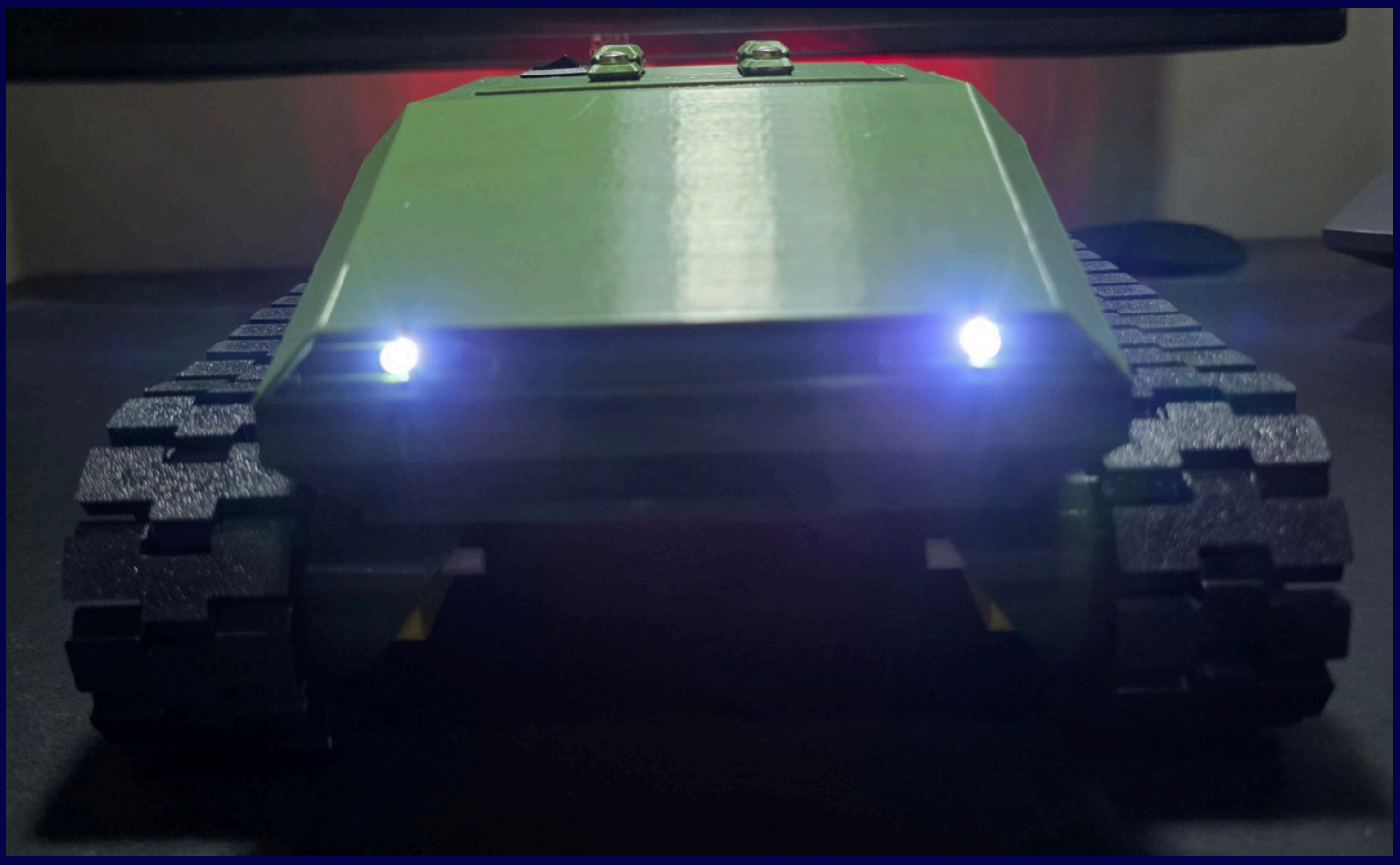
Tanque Montado













Funcionamento

Quer ver o **funcionamento** ? Acesse o vídeo abaixo !!



12 Possíveis Problemas

- **O Tanque não funciona:**

Esse problema na maioria das vezes está relacionado com a bateria, ela pode estar decarregada.

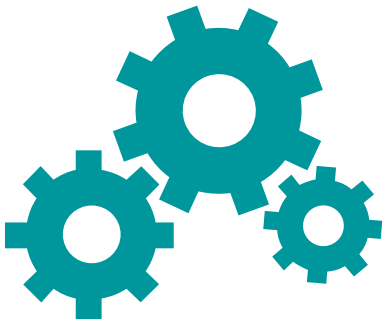
- **Movimentos errados**

Caso o robô responda com os movimentos contrários, isso significa que os fios dos motores foram conectados de forma invertida nos bornes da Ponte H. Basta inverter a posição desses fios e realizar um novo teste de direção

- **Outros**

Problemas como: **Desconexão da placa**, e **não obedecer comandos**, está quase sempre relacionado a bateria. Como apresentado anteriormente, não recomendamos o uso de baterias convencionais, visto que o robô exige uma capacidade de carga maior.



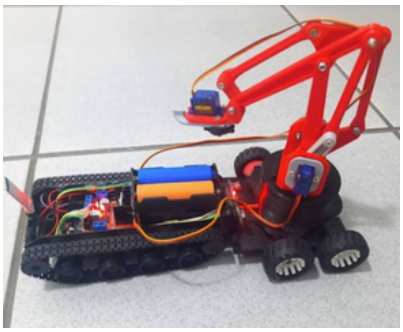


13 Finalização

Neste tutorial, ensinamos como construir um **Tanque** com câmera utilizando o **ESPCAM**. Desejamos que tenha sido uma experiência interessante e que tenha despertado sua curiosidade pelo mundo da **robótica**. Com a crescente demanda por soluções robóticas em nossa sociedade, é cada vez mais importante estudar essa área. Esperamos que você continue explorando e descobrindo tudo o que a robótica tem a oferecer.

Caso você tenha interesse em conhecer mais projetos de eletrônica usando o Microcontroladores, a **Eletrônica Ômega** disponibiliza vários outros tutoriais de forma gratuita em **nosso blog**, temos vários outros tutorais para você mergulhar no mundo da programação. Veja abaixo exemplos de alguns tutoriais que você encontra no blog:

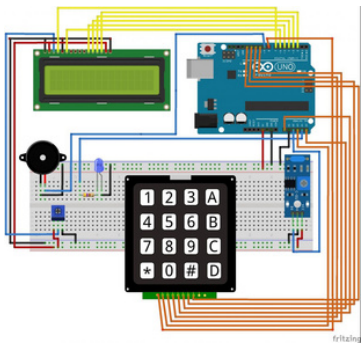
ELETRÔNICA ÔMEGA BLOG



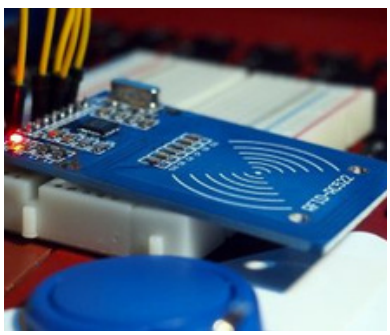
Tanque Reboque Bluetooth



Braço Robótico Controlado por voz



Sistema de Alarme Codificado



Sistema de Acesso com RF ID

E-book Robótica

Venha aprender robótica com a gente! Com o **Kit Arduino Robôs** e o nosso **E-book Robótica para Iniciantes** você irá aprender sobre **Arduino**, eletrônica, física, programação e robótica! Bora lá?

Elaboramos o **E-book** e o **Kit Arduino Robôs**, para serem totalmente amigáveis aos iniciantes, todo o material descrito no **E-book** está contido neste Kit, todos os projetos descritos no **E-book** são possíveis de montar com os componentes deste kit!





Em nosso **E-book** há o detalhamento de **4 projetos de robôs**: o **Veículo Autônomo**, o **Veículo Controlado por App**, **Braço Robótico Controlado por Joystick** e **Braço Robótico Controlado por App**! Você irá encontrar o código fonte completo, com explicação de todos os pontos do código, diagrama elétrico mostrando como fazer a ligação de cada componentes, detalhamentos dos principais problemas que podem ocorrer na montagem, entre várias dicas.

E-book IoT

Venha aprender sobre **Internet das Coisas** com a gente! Com o **Kit Internet das Coisas** e o nosso **E-book** você irá aprender sobre **NodeMCU**, eletrônica, física, programação e projetos mais avançados! Bora lá?

Elaboramos o **E-book** e o **Kit Internet das Coisas** para serem totalmente amigáveis aos iniciantes, todo o material descrito no **E-book** está contido neste Kit, todos os projetos descritos no **E-book** são possíveis de montar com os componentes deste kit!





Quer receber uma notificação no seu celular quando começar a chover? Quer irrigar o seu jardim de forma automática quando o solo estiver seco? Quer acender lâmpadas ou ligar eletrodomésticos com o seu celular? Então adquira o Kit e baixe o **E-Book** e tenha sua casa dentro do seu celular!

Quem escreveu o E-book ?



Rangel Gabriel

Formado em **Eletroeletrônica** pelo **SENAI** Shunji Nishimura e atualmente graduando em **Engenharia Elétrica** pela **UNESP**, o autor tem experiência em eletrônica e microcontroladores.

E-mail: rangelarena@gmail.com

Eletrônica Ômega

A **Eletrônica Ômega** é uma loja virtual sediada em **BH/MG**, especializada em **Arduino** e **componentes eletrônicos**.

Temos a missão de **promover transformação cultural e social através da educação**, inserindo o maior número possível de pessoas no movimento maker, levando acesso a tecnologia para todos os interessados, e **mostrando que é possível desenvolver suas ideais através da tecnologia**.

Caso queria falar com a gente não deixe de mandar seu email para **contato@arduinoomega.com**.

Redes Sociais:

Referências

Motor DC:

https://www.mundodaeletrica.com.br/motor-de-corrente-continua-caracteristicas-e-aplicacoes/ **- acesso em 23/06/2021**.

Ponte H:

https://www.arduinoportugal.pt/modulo-ponte-h-l298n-primeiro-passo-montar-robo-arduino/ **- acesso em 23/06/2021**.

