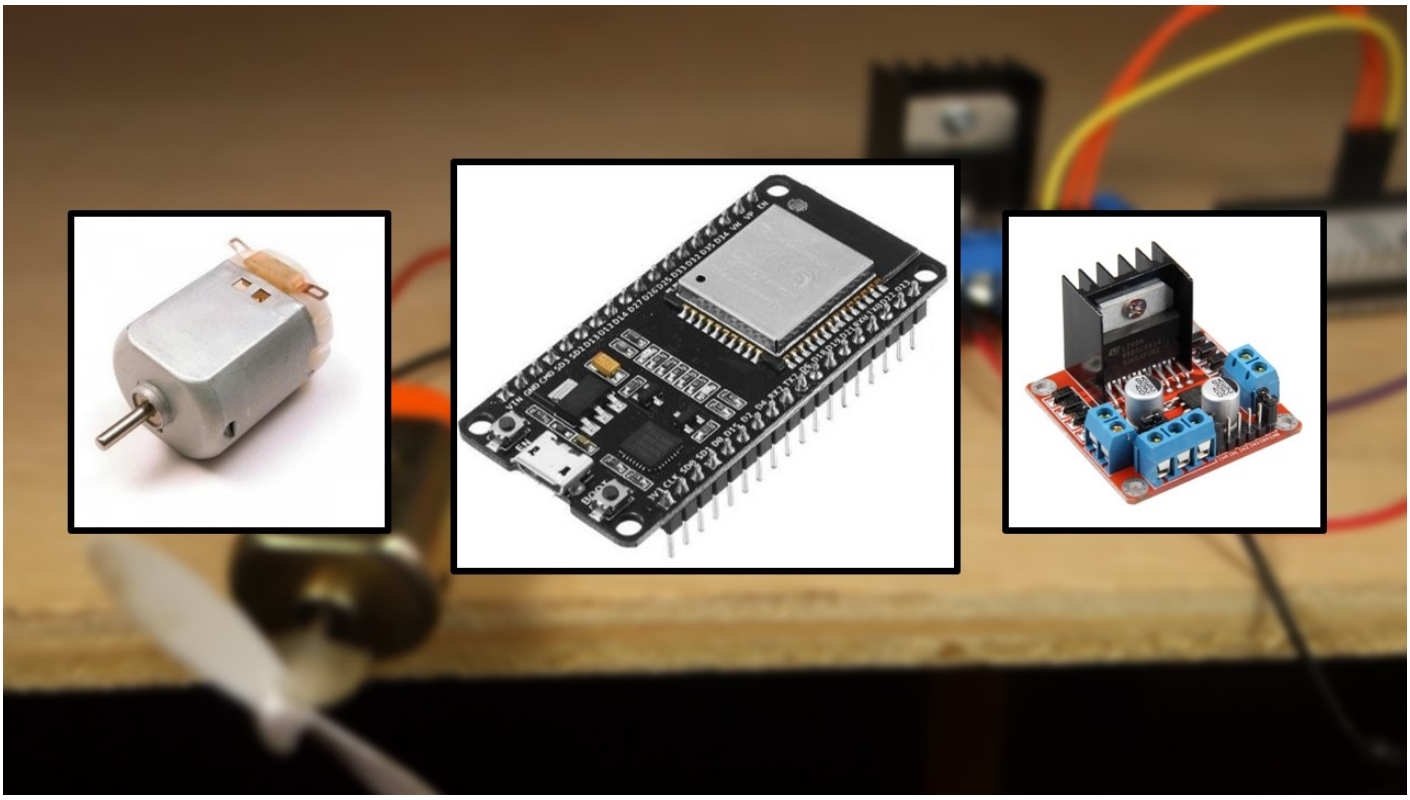
ESP32 con motor de CC y controlador de motor L298N: velocidad y dirección de control

Este tutorial muestra cómo controlar la dirección y la velocidad de un motor de CC utilizando un ESP32 y el controlador de motor L298N. Primero, veremos rápidamente cómo funciona el controlador de motor L298N. Luego, le mostraremos un ejemplo de cómo controlar la velocidad y la dirección de un motor de CC utilizando el ESP32 con Arduino IDE y el controlador de motor L298N.



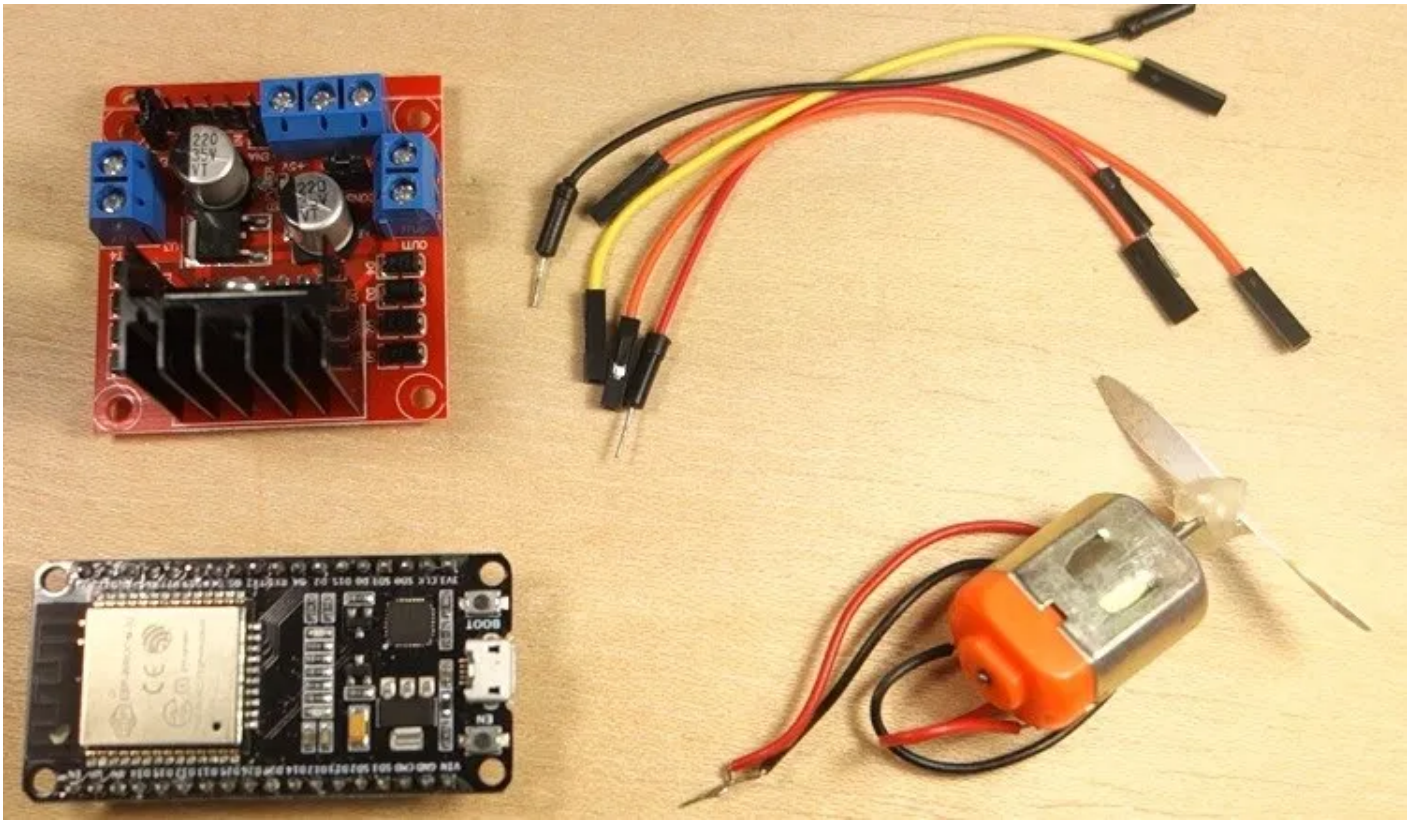
Nota : hay muchas formas de controlar un motor de CC. Usaremos el controlador de motor L298N. Este tutorial también es compatible con módulos de controlador de motor similares.

Para comprender mejor con este tutorial, es posible que desee echar un vistazo a las siguientes publicaciones:

- Primeros pasos con el módulo de desarrollo ESP32
- Instalación de la placa ESP32 en Arduino IDE (instrucciones de Windows)
- Instalación de la placa ESP32 en Arduino IDE (instrucciones para Mac y Linux)
- Servidor web ESP32 – Arduino IDE

Piezas necesarias

Para completar este tutorial necesita las siguientes partes:



- Placa ESP32 DOIT DEVKIT V1 : lea la revisión y comparación de las placas de desarrollo ESP32
- motor de corriente continua
- Controlador de motor L298N
- Fuente de alimentación: 4 pilas AA de 1,5 o fuente de alimentación de banco
- 2 condensadores cerámicos de 100 nF (opcional)
- 1 interruptor deslizante SPDT (opcional)
- Cables puente

¡Puede usar los enlaces anteriores o ir directamente a MakerAdvisor.com/tools para encontrar todas las piezas para sus proyectos al mejor precio!

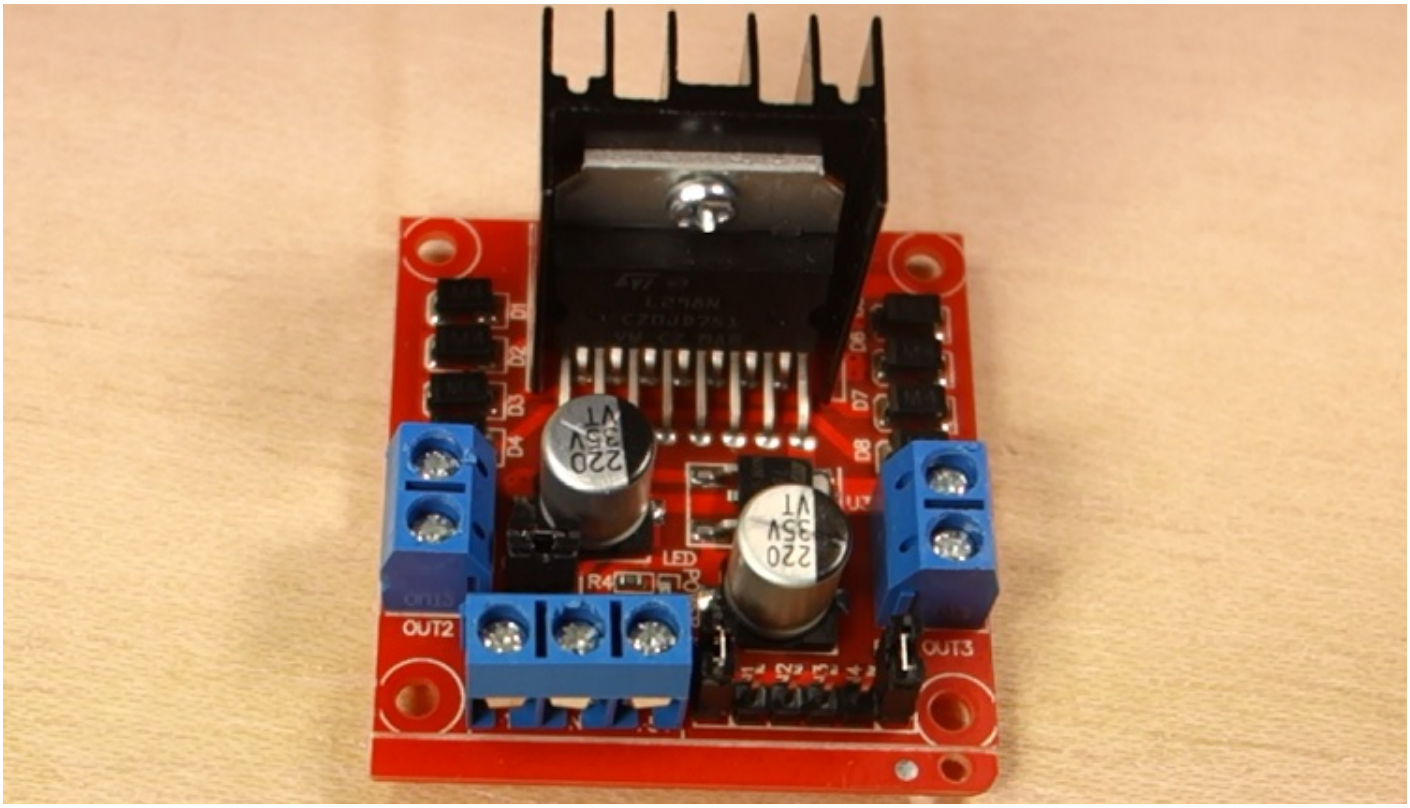
MAKER ADVISOR

Presentamos el controlador de motor L298N

Hay muchas formas de controlar un motor de CC. El método que usaremos aquí es adecuado para la mayoría de los motores de aficionados, que requieren 6 V o 12 V para funcionar.

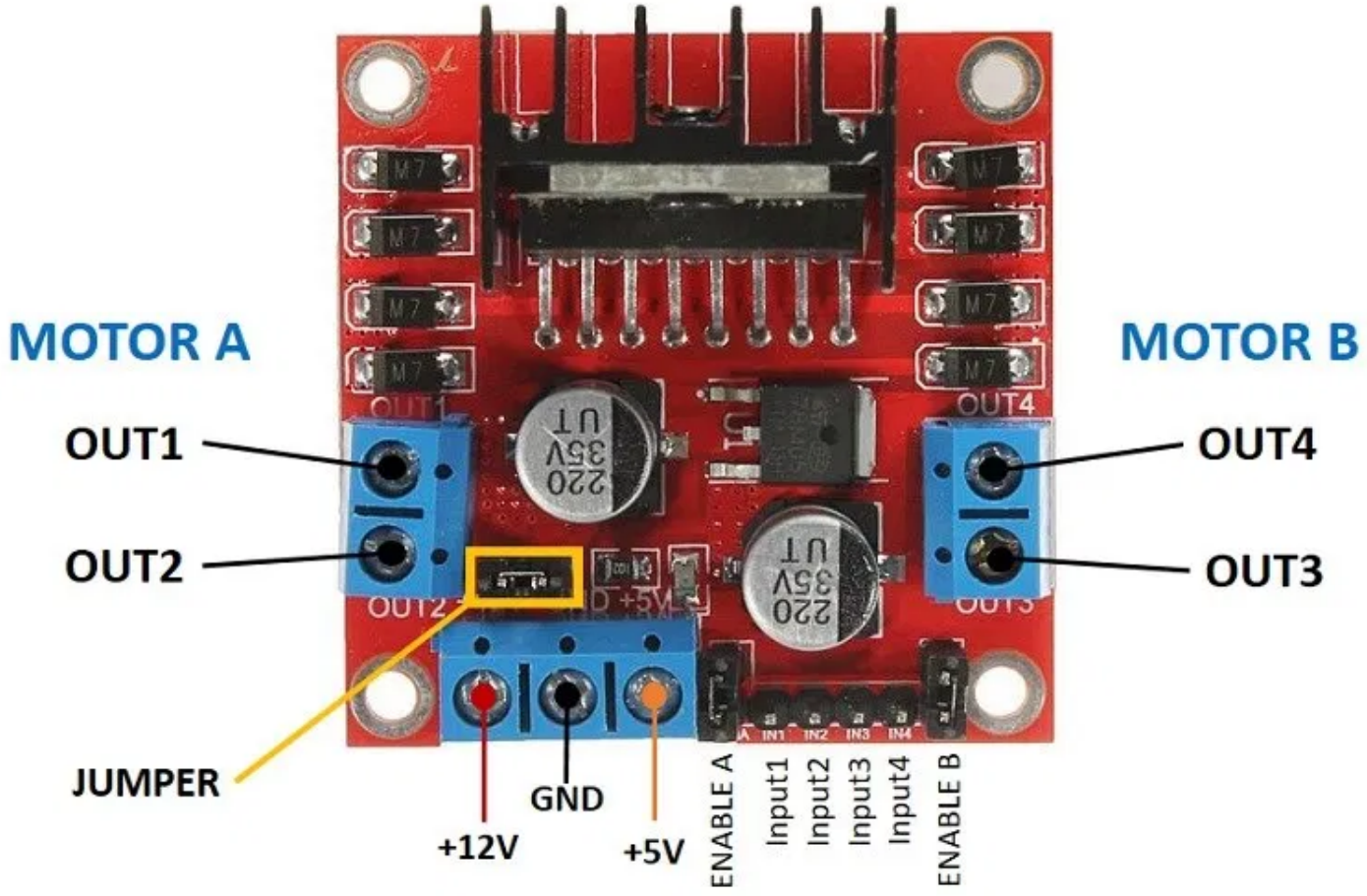
Vamos a usar el controlador de motor L298N que puede manejar hasta 3A a 35V. Además, nos permite accionar dos motores de CC simultáneamente, lo cual es perfecto para construir un robot.

El controlador de motor L298N se muestra en la siguiente figura:



Asignación de pines del controlador de motor L298N

Echemos un vistazo al pinout del controlador del motor L298N y veamos cómo funciona.



El controlador del motor tiene un bloque de dos terminales en cada lado para cada motor. OUT1 y OUT2 a la izquierda y OUT3 y OUT4 a la derecha.

- **OUT1** : motor CC A + terminal
- **OUT2** : Motor CC A - terminal
- **OUT3** : motor CC B + terminal
- **OUT4** : Motor CC B - terminal

En la parte inferior tiene un bloque de tres terminales con +12V , TIERRA , y +5V . los +12V El bloque de terminales se utiliza para encender los motores. los +5V terminal se utiliza para encender el chip L298N. Sin embargo, si el puente está en su lugar, el chip se alimenta con la fuente de alimentación del motor y no necesita suministrar 5V a través del +5V Terminal.

Nota : si suministra más de 12 V, debe quitar el puente y suministrar 5 V al terminal de +5 V.

Es importante tener en cuenta que a pesar del nombre del terminal de +12 V, con la configuración que usaremos aquí (con el puente en su lugar) puede suministrar cualquier voltaje entre 6 V y 12 V. En este tutorial, usaremos 4 baterías AA de 1.5 V que combinadas producen aproximadamente 6 V, pero puede usar cualquier otra fuente de alimentación adecuada. Por ejemplo, puede usar una fuente de alimentación de banco para probar este tutorial.

En resumen:

- **+12V** : El terminal +12V es donde debe conectar su fuente de alimentación
- **GND** : fuente de alimentación GND
- **+5V** : proporciona 5V si se retira el puente. Actúa como una salida de 5 V si el puente está en su lugar
- **Puente** : puente en su lugar: utiliza la fuente de alimentación de los motores para encender el chip. Puente eliminado: debe proporcionar 5V al terminal +5V. Si suministra más de 12V, debe quitar el puente

En la parte inferior derecha tiene cuatro pines de entrada y dos terminales de habilitación. Los pines de entrada se utilizan para controlar la dirección de sus motores de CC y los pines de habilitación se utilizan para controlar la velocidad de cada motor.

- **IN1:** Entrada 1 para Motor A
- **IN2** : Entrada 2 para Motor A
- **IN3** : Entrada 1 para Motor B
- **IN4** : Entrada 2 para Motor B
- **EN1** : pin de habilitación para el motor A
- **EN2** : pin de habilitación para el motor B

Hay tapas de puente en los pines de activación de forma predeterminada. Debe quitar esas tapas de puente para controlar la velocidad de sus motores.

Controle motores de CC con el L298N

Ahora que está familiarizado con el controlador de motor L298N, veamos cómo usarlo para controlar sus motores de CC.

Habilitar pines

Los pines de habilitación son como un interruptor de ENCENDIDO y APAGADO para sus motores. Por ejemplo:

- Si envía una **señal ALTA** al pin de habilitación 1, el motor A está listo para ser controlado y en la velocidad máxima;
- Si envía una **señal BAJA** al pin de habilitación 1, el motor A se apaga;
- Si envía una **señal PWM** , puede controlar la velocidad del motor. La velocidad del motor es proporcional al ciclo de trabajo. Sin embargo, tenga en cuenta que para ciclos de trabajo pequeños, es posible que los motores no giren y emitan un zumbido continuo.

SEÑAL EN EL PIN DE HABILITACIÓN	ESTADO DEL MOTOR
ALTO	Motor habilitado
BAJO	Motor no habilitado
PWM	Motor habilitado: velocidad proporcional al ciclo de trabajo

pines de entrada

Los pines de entrada controlan la dirección en que giran los motores. La entrada 1 y la entrada 2 controlan el motor A, y las entradas 3 y 4 controlan el motor B.

- Si aplica BAJO a la entrada 1 y ALTO a la entrada 2, el motor girará hacia adelante;
- Si aplica energía al revés: ALTA a la entrada 1 y BAJA a la entrada 2, el motor girará hacia atrás. El motor B se puede controlar usando el mismo método pero aplicando ALTO o BAJO a la entrada 3 y la entrada 4.

Control de 2 motores de CC: ideal para construir un robot

Si desea construir un automóvil robot con 2 motores de CC, estos deben girar en direcciones específicas para que el robot se mueva hacia la izquierda, hacia la derecha, hacia adelante o hacia atrás.

Por ejemplo, si desea que su robot avance, ambos motores deben girar hacia adelante. Para hacerlo retroceder, ambos deben girar hacia atrás.

Para girar el robot en una dirección, debe hacer girar el motor opuesto más rápido. Por ejemplo, para hacer que el robot gire a la derecha, habilite el motor de la izquierda y deshabilite el motor de la derecha. La siguiente tabla muestra las combinaciones de estado de los pines de entrada para las direcciones del robot.

DIRECCIÓN	ENTRADA 1	ENTRADA 2	ENTRADA 3	ENTRADA 4
Delantero	0	1	0	1
Hacia atrás	1	0	1	0
Derecha	0	1	0	0
Izquierda	0	0	0	1
Deténgase	0	0	0	0

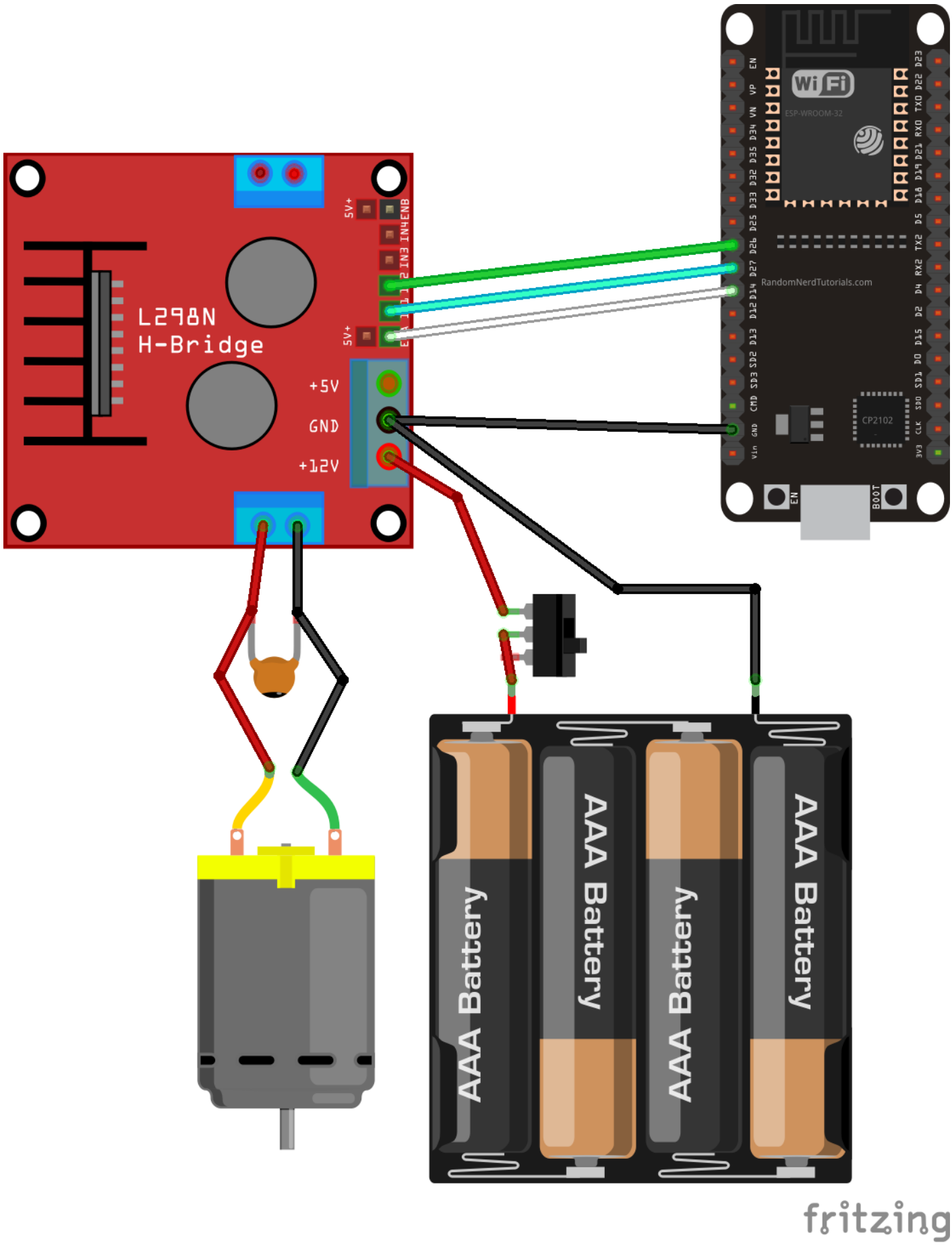
Lectura recomendada: Build Robot Car Chassis Kit para ESP32, ESP8266, Arduino, etc...

Controle el motor de CC con ESP32: velocidad y dirección

Ahora que sabe cómo controlar un motor de CC con el controlador de motor L298N, construyamos un ejemplo simple para controlar la velocidad y la dirección de un motor de CC.

Esquemático

El motor que controlaremos está conectado a los pines de salida del motor A, por lo que debemos conectar los pines ENABLEA, INPUT1 e INPUT2 del controlador del motor al ESP32. Siga el siguiente diagrama esquemático para conectar el motor de CC y el controlador del motor L298N al ESP32.



El motor de CC requiere un gran salto en la corriente para moverse, por lo que los motores deben alimentarse con una fuente de alimentación externa del ESP32. Como ejemplo, estamos usando baterías 4AA, pero puede usar cualquier otra fuente de

alimentación adecuada. En esta configuración, puede utilizar una fuente de alimentación con 6V a 12V.

El interruptor entre el soporte de la batería y el controlador del motor es opcional, pero es muy útil para cortar y aplicar energía. De esta manera, no necesita conectar y desconectar constantemente los cables para ahorrar energía.

Recomendamos soldar un condensador cerámico de 0,1 uF a los terminales positivo y negativo del motor de CC, como se muestra en el diagrama, para ayudar a suavizar los picos de voltaje. (Nota: los motores también funcionan sin el condensador).

Preparando el IDE de Arduino

Hay un complemento para el IDE de Arduino que le permite programar el ESP32 usando el IDE de Arduino y su lenguaje de programación. Siga uno de los siguientes tutoriales para preparar su Arduino IDE para trabajar con el ESP32, si aún no lo ha hecho.

- **Instrucciones de Windows** – Placa ESP32 en Arduino IDE
- **Instrucciones Mac y Linux** – Placa ESP32 en Arduino IDE

Después de asegurarse de que tiene instalado el complemento ESP32, puede continuar con este tutorial.

Cargando código

El siguiente código controla la velocidad y la dirección del motor de CC. Este código no es útil en el mundo real, este es solo un ejemplo simple para comprender mejor cómo controlar la velocidad y la dirección de un motor de CC con el ESP32.

```
/*********
  Rui Santos
  Complete project details at https://randomnerdtutorials.com
*********/

// Motor A
int motor1Pin1 = 27;
int motor1Pin2 = 26;
int enable1Pin = 14;

// Setting PWM properties
const int freq = 30000;
const int pwmChannel = 0;
const int resolution = 8;
```

```
const int resolution = 8;
int dutyCycle = 200;

void setup() {
  // sets the pins as outputs:
  pinMode(motor1Pin1, OUTPUT);
  pinMode(motor1Pin2, OUTPUT);
  pinMode(enable1Pin, OUTPUT);

  // configure LED PWM functionalitites
  ledcSetup(pwmChannel, freq, resolution);

  // attach the channel to the GPIO to be controlled
  ledcAttachPin(enable1Pin, pwmChannel);
```

Ver código sin procesar

Sube el código a tu ESP32. Asegúrese de tener la placa y el puerto COM correctos seleccionados. Echemos un vistazo a cómo funciona el código.

Declaración de pines de motor

Primero, define los GPIO a los que están conectados los pines del motor. En este caso, la Entrada 1 para el motor A está conectada a GPIO 27, la Entrada 2 a GPIO 26 y el pin Habilitar a GPIO 14.

```
int motor1Pin1 = 27;
int motor1Pin2 = 26;
int enable1Pin = 14;
```

Configuración de las propiedades de PWM para controlar la velocidad

Como vimos anteriormente, puede controlar la velocidad del motor de CC aplicando una señal PWM al pin de habilitación del controlador del motor L298N. La velocidad será proporcional al ciclo de trabajo. Para usar PWM con el ESP32, primero debe configurar las propiedades de la señal PWM.

```
const int freq = 30000;
const int pwmChannel = 0;
```

```
const int resolution = 8;
int dutyCycle = 200;
```

En este caso, estamos generando una señal de 30000 Hz en el canal 0 con una resolución de 8 bits. Comenzamos con un ciclo de trabajo de 200 (puede establecer un valor de ciclo de trabajo de 0 a 255).

Para la frecuencia que estamos usando, cuando aplica ciclos de trabajo menores a 200, el motor no se moverá y emitirá un zumbido extraño. Entonces, es por eso que establecimos un ciclo de trabajo de 200 al principio.

Nota : las propiedades de PWM que estamos definiendo aquí son solo un ejemplo. El motor funciona bien con otras frecuencias.

configuración()

En el `configuración()` , comienza configurando los pines del motor como salidas.

```
pinMode(motor1Pin1, OUTPUT);
pinMode(motor1Pin2, OUTPUT);
pinMode(enable1Pin, OUTPUT);
```

Debe configurar una señal PWM con las propiedades que ha definido anteriormente utilizando el `ledcConfiguración()` función que acepta como argumentos, el `pwmChannel` , la `frecuencia` , y el `resolución` , como sigue:

```
ledcSetup(pwmChannel, freq, resolution);
```

A continuación, debe elegir el GPIO del que obtendrá la señal. Para eso utiliza el `ledcAdjuntarPin()` función que acepta como argumentos el GPIO de donde se quiere sacar la señal, y el canal que está generando la señal. En este ejemplo, obtendremos la señal en el `habilitar1Pin` GPIO, que corresponde a GPIO 14. El canal que genera la señal es el `pwmChannel` , que corresponde al canal 0.

```
ledcAttachPin(enable1Pin, pwmChannel);
```

Mover el motor de CC hacia adelante

En el `círculo()` es donde se mueve el motor. El código está bien comentado sobre lo que hace cada parte del código. Para mover el motor hacia adelante, configure la entrada 1 pin en BAJO y la entrada 2 pin en ALTO. En este ejemplo, el motor acelera hacia adelante durante 2 segundos (2000 milisegundos).

```
// Move the DC motor forward at maximum speed
Serial.println("Moving Forward");
digitalWrite(motor1Pin1, LOW);
digitalWrite(motor1Pin2, HIGH);
delay(2000);
```

Mover el motor de CC hacia atrás

Para mover el motor de CC hacia atrás, aplique energía a los pines de entrada del motor al revés. ALTO a la entrada 1 y BAJO a la entrada 2.

```
// Move DC motor backwards at maximum speed
Serial.println("Moving Backwards");
digitalWrite(motor1Pin1, HIGH);
digitalWrite(motor1Pin2, LOW);
delay(2000);
```

Detenga el motor de CC

Para hacer que el motor de CC se detenga, puede configurar el pin de habilitación en BAJO o configurar los pines de entrada 1 y 2 en BAJO. En este ejemplo, estamos configurando ambos pines de entrada en BAJO.

```
// Stop the DC motor
Serial.println("Motor stopped");
digitalWrite(motor1Pin1, LOW);
digitalWrite(motor1Pin2, LOW);
delay(1000);
```

Control de la velocidad del motor de CC

Para controlar la velocidad del motor de CC, necesitamos cambiar el ciclo de trabajo de la señal PWM. Para eso usas el `ledcWrite()` función que acepta como argumentos el

canal PWM que está generando la señal (no el GPIO de salida) y el ciclo de trabajo, como sigue.

```
ledcWrite(pwmChannel, dutyCycle);
```

En nuestro ejemplo, tenemos un bucle while que aumenta el ciclo de trabajo en 5 en cada bucle.

```
// Move DC motor forward with increasing speed
digitalWrite(motor1Pin1, HIGH);
digitalWrite(motor1Pin2, LOW);
while (dutyCycle <= 255){
  ledcWrite(pwmChannel, dutyCycle);
  Serial.print("Forward with duty cycle: ");
  Serial.println(dutyCycle);
  dutyCycle = dutyCycle + 5;
  delay(500);
}
```

Cuando la condición while ya no es cierta, establecemos el ciclo de trabajo en 200 nuevamente.

```
dutyCycle = 200;
```

Vea la demostración en video

Mire el siguiente video para ver el proyecto en acción:

Terminando

En este tutorial, le mostramos cómo controlar la dirección y la velocidad de un motor de CC utilizando un controlador de motor ESP32 y L298N. En resumen:

- Para controlar la dirección en que gira el motor de CC, use los pines de entrada 1 y entrada 2;
- Aplique BAJO a la entrada 1 y ALTO a la entrada 2 para hacer girar el motor hacia adelante. Aplique potencia al revés para que gire hacia atrás;

- Para controlar la velocidad del motor de CC, utiliza una señal PWM en el pin de habilitación. La velocidad del motor de CC es proporcional al ciclo de trabajo.

Esperamos que hayas encontrado útil este tutorial.

Este es un extracto de nuestro curso: Aprende ESP32 con Arduino IDE . Si te gusta ESP32 y quieres aprender más sobre él, te recomendamos inscribirte en el curso Learn ESP32 with Arduino IDE .

También te puede gustar leer:

- Aprende ESP32 con Arduino IDE
- Servidor web de servomotor ESP32 con Arduino IDE
- Alexa (Echo) con ESP32 y ESP8266 - Relé controlado por voz
- Cree un escudo de estación meteorológica ESP32 todo en uno
- ESP32 Publicar lecturas de sensores en hojas de cálculo de Google