

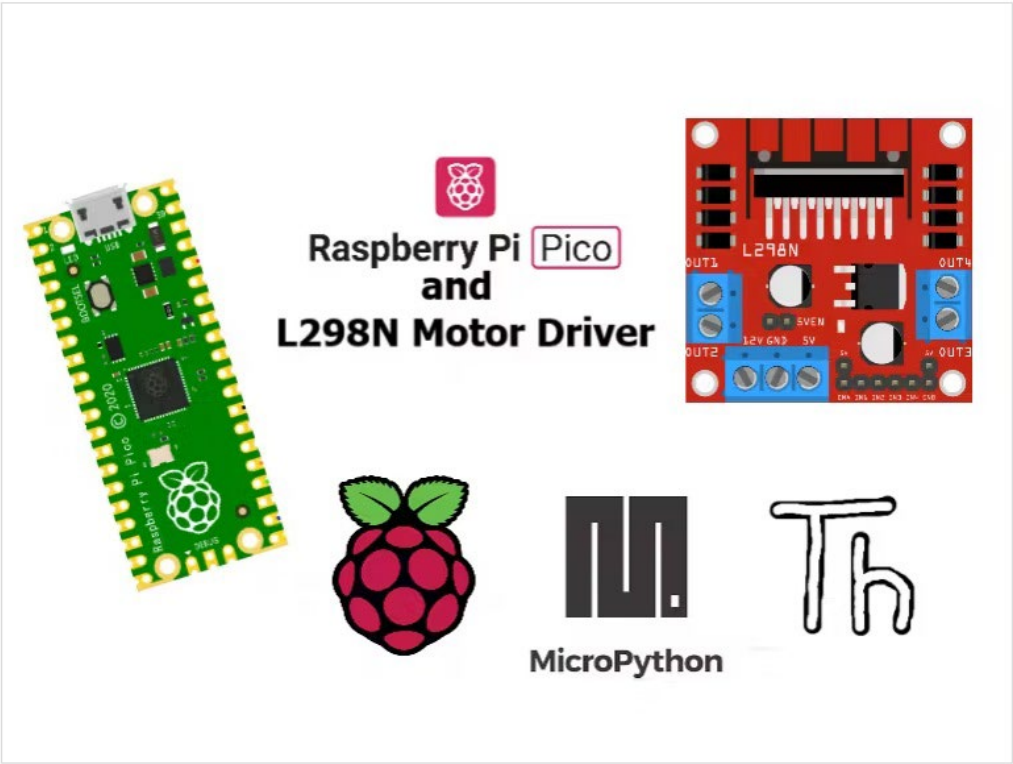
CONTROLADOR DE MOTOR RASPBERRY PI PICO Y L298N



Ramji Patel
Published June 11, 2023 © CC BY-NC

















Traducción: José Antonio de Jesús Arredondo Garza

Control de motores con Raspberry pi pico, controlador de motor L298N y MicroPython



Cosas utilizadas en este proyecto.

Componentes de hardware:

	Raspberry Pi Pico	× 1	
	Raspberry Pi Pico USB Cable	× 1	
	SparkFun Full-Bridge Motor Driver Breakout - L298N	× 1	
	Solderless Breadboard Full Size	× 1	
	Pimoroni Maker Essentials - Mini Breadboards & Jumper Jerky	× 1	
	DC Motor, 12 V	× 1	
	Rechargeable Battery, 12 V	× 1	
	Alligator Clips	× 1	

Aplicaciones de Software y Servicios en Línea:

Thonny



Herramientas manuales y otros accesorios:



Soldering Station, 110 V



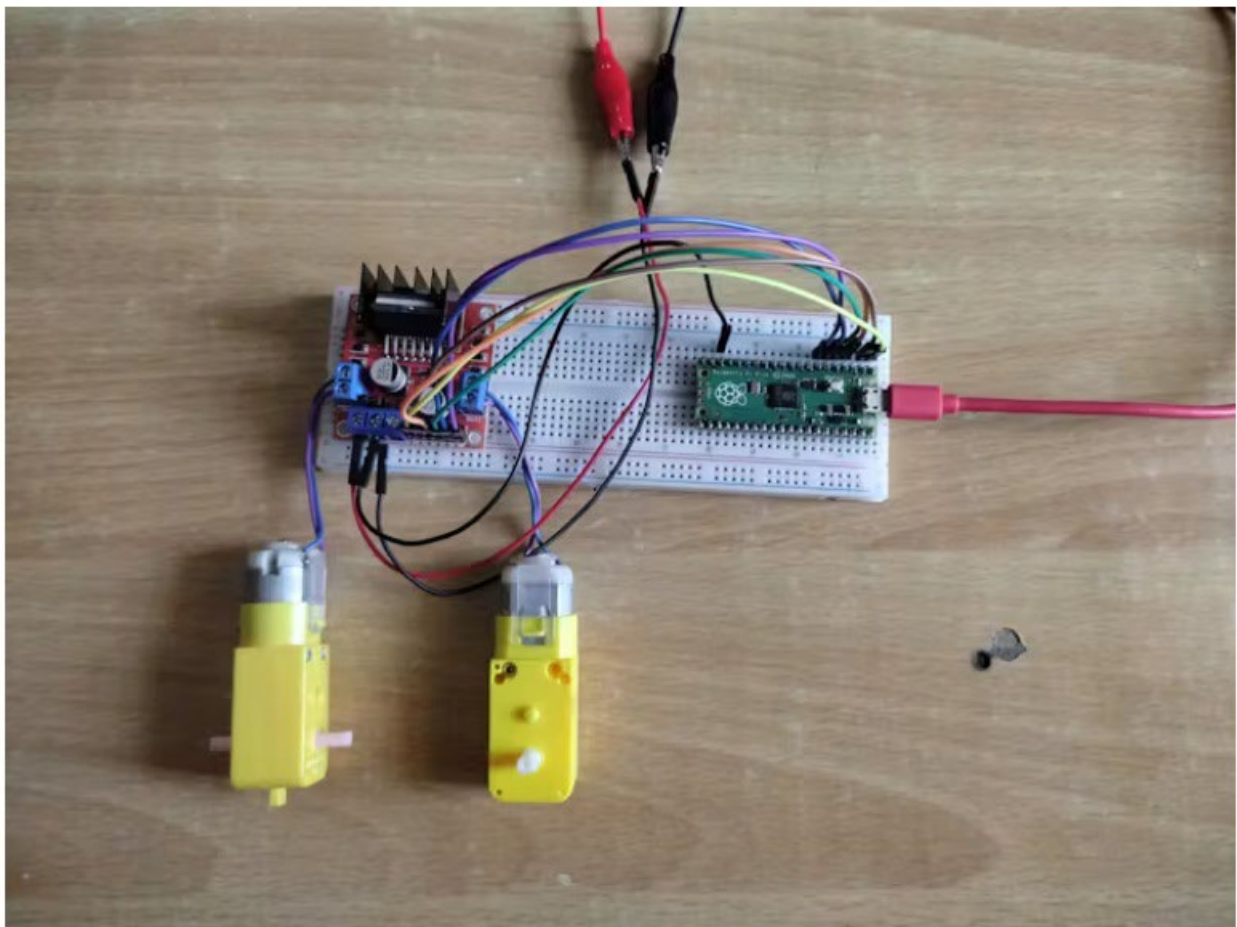
Solder Flux, Soldering



Solder Wire, Lead Free



Historia



Introducción

Si desea crear un robot (como un seguidor de línea, un evitador de obstáculos, un robot de brazo o un robot de carreras) o un automóvil usando un microcontrolador (como Arduino, ESP8266, ESP32, Raspberry pi pico, etc.), use motores para hacer su movimiento del robot. Los motores constan de muchos devanados de alambre de cobre dentro de un campo magnético. Estos motores requieren más energía de la que un microcontrolador puede suministrar, por lo que no se pueden accionar ni controlar motores utilizando un microcontrolador directamente. Esto se debe a que las placas del microcontrolador funcionan con bajo voltaje y consumen menos energía y sus pines de entrada/salida no pueden suministrar suficiente corriente para impulsar sus motores. Si intenta conectar

un motor directamente a un pin de entrada/salida de la placa del microcontrolador, es posible que la placa del microcontrolador deje de funcionar. Para deshacerse de este problema, debe utilizar un controlador de motor adecuado para controlar la dirección y la velocidad de rotación. Puede utilizar un BJT o FET como controlador de motor, pero un solo BJT o FET puede controlar solo la velocidad, no la dirección de rotación. Para controlar la velocidad y la dirección, necesita utilizar 4 BJT o FET en la configuración del puente H. Debido al uso de 4 BJT o FET en el puente H, su circuito se volverá complejo y voluminoso. Para superar este problema, utilizamos un controlador de motor IC. Dentro de un circuito integrado de controlador de motor, se fabrican uno y más puentes H en un solo chip para que pueda usarlo fácilmente en su circuito. L298N y L293D son dos circuitos integrados de controladores de motor más populares para su uso en robots. Puede usar estos circuitos integrados directamente o puede usar un circuito controlador de motor ya ensamblado que también se denomina módulo. Aquí le mostraré cómo puede controlar motores usando el módulo controlador de motor L298N y Raspberry pi pico (como placa de microcontrolador) usando Lenguaje de programación MicroPython. Después de leer y comprender este artículo detenidamente, podrá:

- Controle motores fácilmente utilizando el módulo L298N y el lenguaje de programación MicroPython.
- Cree un robot fácilmente utilizando el lenguaje de programación L298N y MicroPython.
- Controlar motores con Raspberry pi pico.

Requisitos de hardware:

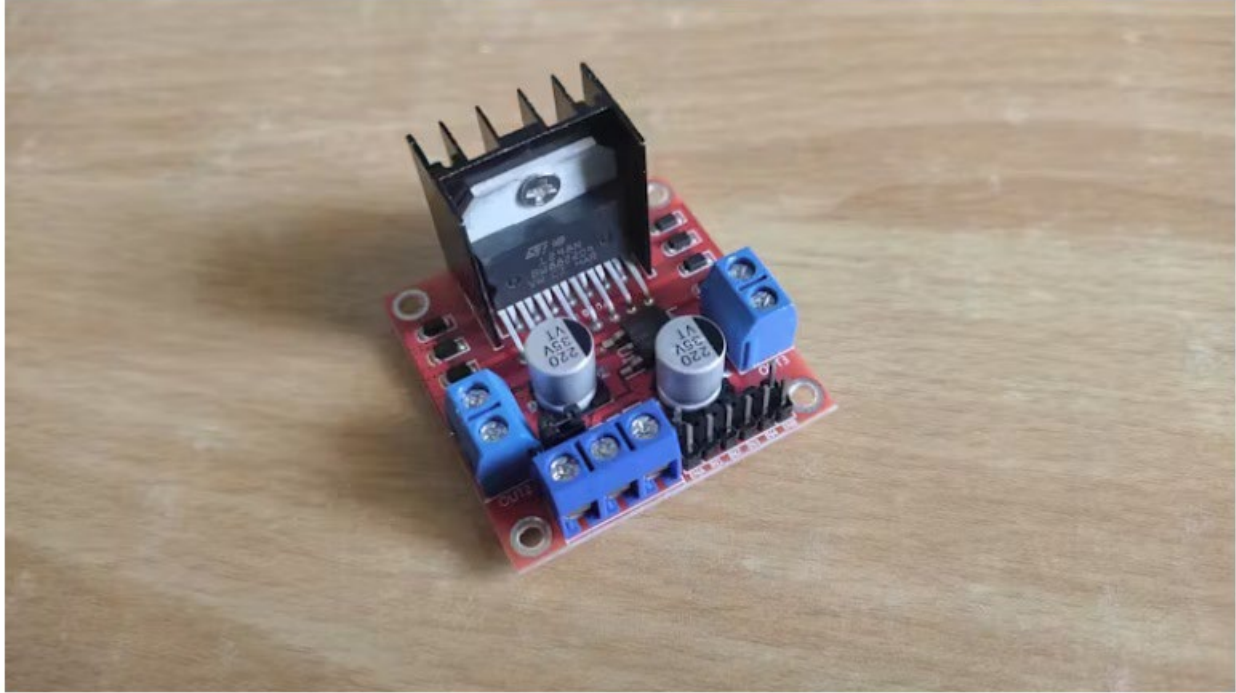
- Raspberry Pi pico
- Módulo controlador de motor L298N
- Tabla de montaje (proto board)
- Jumpers (M-t-M y M-t-F)
- Cables con pinzas caimán.
- Batería de 12V
- Cable micro USB
- Motorreductores (2)
- Destornillador
- Soldador
- Alambre y pasta para soldar
- Laptop o computadora
- Módulo Bluetooth HC-05
- potenciómetro de 10K

Requisitos de Software

- Thonny (Python IDE para principiantes).
<https://thonny.org/>
- Firmware MicroPython para Raspberry pi Pico.
<https://micropython.org/download/>
- **L298N_motor.py** Modulo MicroPython.
<https://github.com/ramjipatel041/Micropython-Driver-for-L298N-Motor-Driver.git>

Teoría del Módulo Controlador de Motor L298N

Este controlador de motor bidireccional dual se basa en el muy popular circuito integrado de controlador de motor de puente H dual L298. El circuito te permitirá controlar de forma fácil e independiente dos motores de hasta 2A cada uno en ambos sentidos. Es ideal para aplicaciones robóticas y muy adecuado para la conexión a un microcontrolador que requiere solo un par de líneas de control por motor. También se puede interconectar con interruptores manuales simples, compuertas lógicas TTL, relés, etc. Esta placa está equipada con indicadores LED de alimentación, regulador integrado de +5 V y diodos de protección. La imagen del modulo se presenta a continuación:



Especificaciones:

- Voltaje de entrada: 3,2 V ~ 40 V CC. Datos breves:
- Controlador: Controlador de motor de CC de puente H dual L298N
- Fuente de alimentación: +5 V a +35 V
- Corriente máxima: 2 amperios
- Rango de corriente de funcionamiento: 0 ~ 36 mA
- Rango de voltaje de entrada de la señal de control:
 - ✓ Bajo: $-0,3 \text{ V} \leq V_{in} \leq 1,5 \text{ V}$.
 - ✓ Alto: $2,3 \text{ V} \leq V_{in} \leq V_{ss}$.
- Rango del habilitador del voltaje de entrada de señal:
 - ✓ Bajo: $-0,3 \leq V_{in} \leq 1,5 \text{ V}$ (la señal de control no es válida).
 - ✓ Alto: $2,3 \text{ V} \leq V_{in} \leq V_{ss}$ (señal de control activa).
- Consumo máximo de energía: 20W (cuando la temperatura $T = 75 \text{ }^{\circ}\text{C}$).
- Temperatura de almacenamiento: $-25 \text{ }^{\circ}\text{C} \sim +130 \text{ }^{\circ}\text{C}$.
- Suministro de salida regulado de +5 V integrado (suministro a la placa del controlador, es decir, Arduino).
- Tamaño: 3,4 cm x 4,3 cm x 2,7 cm

Paso 1: Instalación del Firmware MicroPython en Raspberry Pi Pico

Cuando compra una nueva placa Raspberry pi pico, debe instalar la última versión del firmware MicroPython en su placa Raspberry pi pico. Para instalar un nuevo firmware en su Raspberry pi pico, mantenga presionado el botón de arranque de su Raspberry pi pico y conéctelo a la computadora con un cable micro USB. Después de conectarse, suelte el botón de arranque. Tan pronto como suelte el botón de arranque, se abrirá automáticamente una nueva unidad en la pantalla de su computadora. Seleccione el firmware descargado (archivo .UF2) y arrástrelo dentro de esta unidad. Ahora la unidad desaparecerá automáticamente de la pantalla. Abra thonny IDE y haga clic en Herramientas y luego seleccione Opciones. Ahora haga clic nuevamente en la pestaña del intérprete y seleccione el intérprete MicroPython (Raspberry pi pico). También seleccione el PUERTO COM al que está conectada su placa y haga clic en Aceptar. Ahora su placa Raspberry pi pico se conectará a Thonny IDE y verá el siguiente mensaje en el shell.

```
MicroPython v1.20.0 on 2023-04-26; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
```

Ahora escriba:

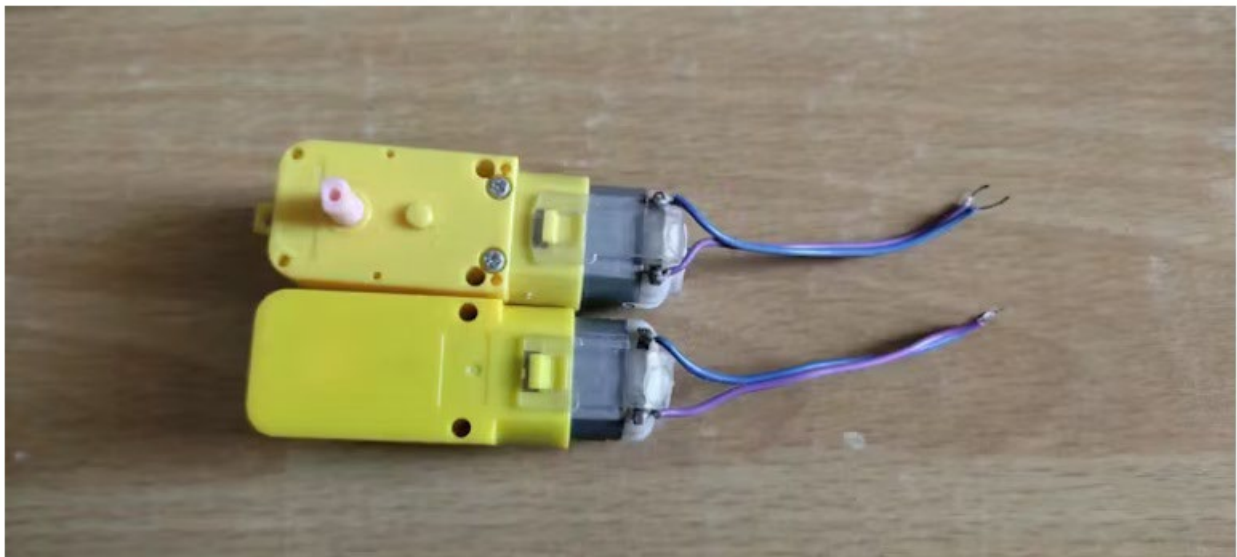
```
>>> print('hello')
```

En el shell y presione enter. Su Raspberry pi pico devolverá la siguiente respuesta:

```
>>> print('hello')
hello
>>>
```

Paso 2: Soldar Cables a los Motores

Para conectar motores al módulo L298N, necesita soldar cables a sus motores. Puede soldar fácilmente cables a sus motores utilizando herramientas de soldadura (soldador, fundente y alambre de soldadura).



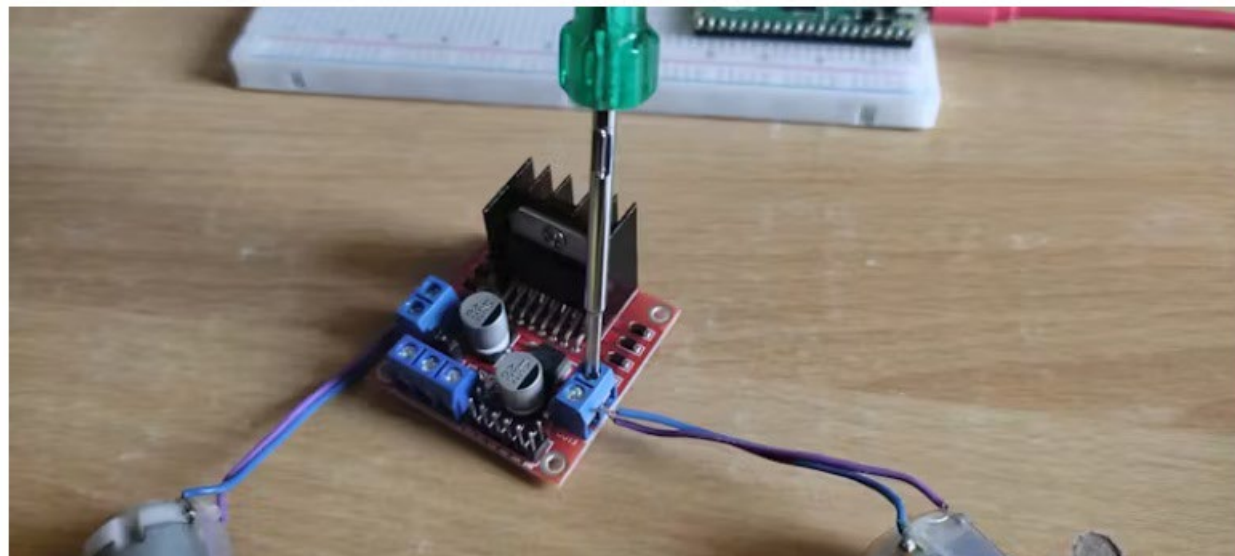
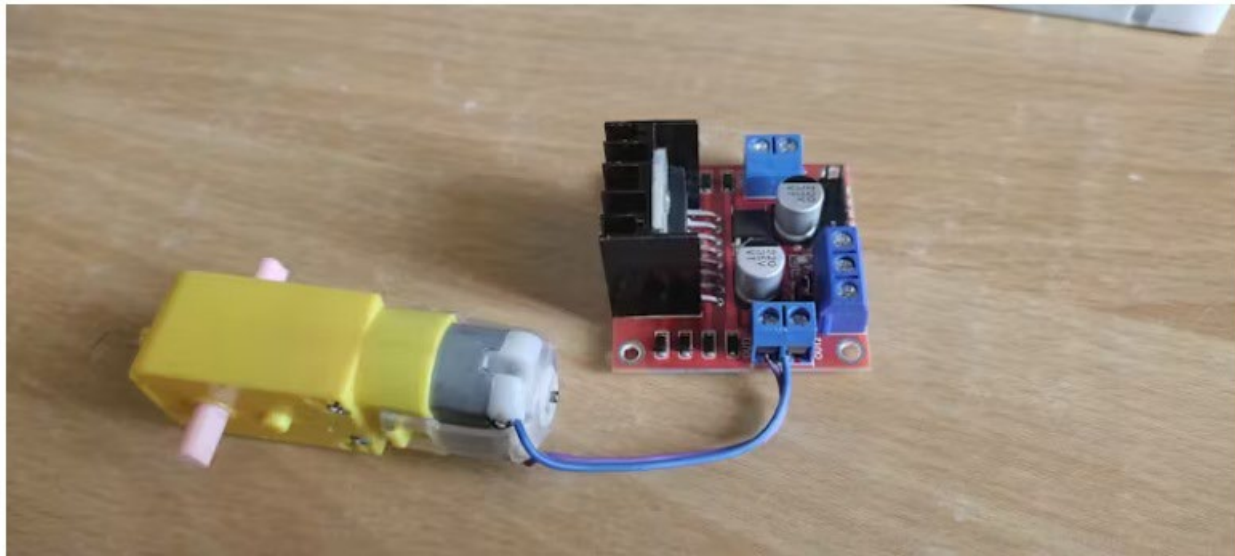
NOTA: Para el caso de motores GM 25-370, las terminales de alimentación del motor son

- “M1 Motor –” y “M1 Motor +”

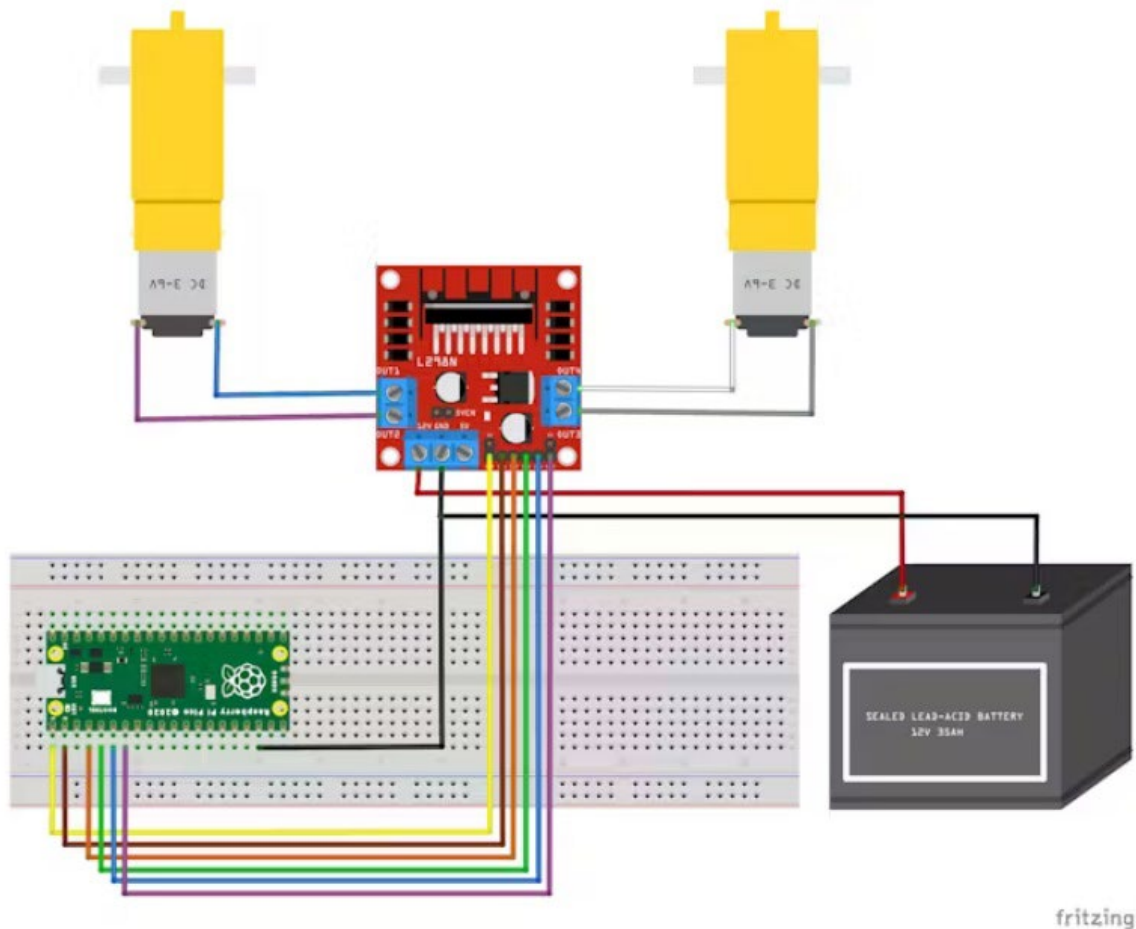


Paso 3: Conecte los Motores al Módulo Controlador L298N

Después de soldar los cables a ambos motores, conecte estos cables al módulo L298N. Hay dos terminales de tornillo de 2 pines en el módulo controlador L298N. Conecte los cables del primer motor al terminal de tornillo del lado izquierdo y los cables del segundo motor al terminal de tornillo del lado derecho y no olvide apretar las conexiones con un destornillador pequeño.



Paso 4: Realizar una Configuración de Prueba (circuito) en una Proto Board



Aquí debe preparar una configuración de prueba en una placa para controlar los motores usando la programación Raspberry pi pico y MicroPython. He proporcionado aquí un pdf del diagrama esquemático (Raspberry Pi Pico y L298.pdf). Con la ayuda de este diagrama esquemático, puede crear fácilmente su propio circuito en una placa de pruebas. Si ha realizado algún circuito en una placa de pruebas anteriormente, puede realizar este trabajo muy fácilmente. He preparado el diagrama esquemático basándose en el siguiente esquema de cableado:

L298N module Raspberry Pi Pico

ENA -----> GPIO-0
IN1 -----> GPIO-1
IN2 -----> GPIO-2
IN3 -----> GPIO-3
IN4 -----> GPIO-4
ENB -----> GPIO-5
GND -----> GND

L298N module Battery

Vcc -----> +Ve(Red terminal)
GND -----> -Ve(Black Terminal)

También puede utilizar la ayuda de las imágenes proporcionadas para realizar las conexiones de cableado adecuadas.

Paso 5: Descargue el Módulo MicroPython “L298N_motor.py”

Para que la parte principal de programación sea más fácil para los principiantes, he desarrollado un módulo MicroPython para el controlador del motor L298N. Antes de cablear y ejecutar los programas de ejemplo, debe descargar la biblioteca L298N_motor.py de mi repositorio de GitHub. El enlace del repositorio de GitHub se proporciona a continuación:

<https://github.com/ramjipatel041/Micropython-Driver-for-L298N-Motor-Driver.git>

Después de descargar el módulo micropython anterior, ábralo en *Thonny IDE* y guárdelo en su Raspberry pi pico con el mismo nombre que **L298N_motor.py**. Si lo guarda en su pico con un nombre diferente, entonces no podrá ejecutar los programas de ejemplo que se proporcionan en el siguiente paso.

Paso 6: escribir los Programas de Ejemplo

Después de guardar el archivo L298N_motor.py en su placa pico, ahora está listo para escribir algunos programas de ejemplo. Aquí, en este paso, he escrito seis códigos de ejemplo. Puedes copiarlos y ejecutarlos en tu Raspberry pi pico.

Example-1

```
"""This micropython program makes the motor1
move in forward and backward directions."""

from machine import Pin, PWM
from L298N_motor import L298N
import time

ENA = PWM(Pin(0))
IN1 = Pin(1, Pin.OUT)
IN2 = Pin(2, Pin.OUT)

motor1 = L298N(ENA, IN1, IN2)    #create a motor1 object
motor1.setSpeed(25000)           #set the speed of motor1. Speed value varies fr
om 25000 to 65534

while True:
    motor1.forward()             #run motor1 forward
    time.sleep(5)                #wait for 5 seconds
    motor1.backward()            #run motor1 backward
    time.sleep(5)                #run motor2 backward
```


Example-2

```
"""This micropython program makes the motor1 and motor2
move in forward and backward directions."""

from machine import Pin, PWM
from L298N_motor import L298N
import time

ENA = PWM(Pin(0))
IN1 = Pin(1, Pin.OUT)
IN2 = Pin(2, Pin.OUT)
IN3 = Pin(3, Pin.OUT)
IN4 = Pin(4, Pin.OUT)
ENB = PWM(Pin(5))

motor1 = L298N(ENA, IN1, IN2)    #create a motor1 object
motor2 = L298N(ENB, IN3, IN4)    #create a motor2 object

motor1.setSpeed(30000)            #set the speed of motor1. Speed value varies fr
om 25000 to 65534
motor2.setSpeed(25000)            #set the speed of motor2. Speed value varies fr
om 25000 to 65534
```

Example-3

```
"""This micropython program makes the motor1
move in forward and backward directions with
increasing speed."""

from machine import Pin, PWM
from L298N_motor import L298N
import time

ENA = PWM(Pin(0))
IN1 = Pin(1, Pin.OUT)
IN2 = Pin(2, Pin.OUT)

motor1 = L298N(ENA, IN1, IN2)    #create a motor1 object
motor1.setSpeed(25000)            #set the speed of motor1. Speed value varies fr
om 25000 to 65534

while True:
    for speed in range(25000, 65000, 100):
        motor1.setSpeed(speed)
        motor1.forward()
        time.sleep(0.1)
```

Example-4

```
"""This micropython program makes the motor1
move in forward and backward directions with
increasing and and decreasing speed in both
directions."""

from machine import Pin, PWM
from L298N_motor import L298N
import time

ENA = PWM(Pin(0))
IN1 = Pin(1, Pin.OUT)
IN2 = Pin(2, Pin.OUT)

motor1 = L298N(ENA, IN1, IN2)    #create a motor1 object
motor1.setSpeed(25000)           #set the speed of motor1. Speed value varies fr
om 25000 to 65534

while True:
    for speed in range(25000, 65000, 100):
        motor1.setSpeed(speed)
        motor1.forward()
```

Por ejemplo, 5 y 6, necesita agregar dos dispositivos más al circuito creado anteriormente. Estos dos dispositivos son HC-05 Bluetooth y potenciómetro de 10K. Para conectar estos dispositivos, puede utilizar el diagrama esquemático (Raspberry pi pico_HC05 _L298N.pdf). Usando HC-05 Bluetooth puedes controlar tus motores con tu teléfono inteligente. El código de ejemplo para esto se proporciona a continuación:

Example-5

```
''' This is a micropython program to control the speed
and direction of the motor using serial communication'''

from machine import Pin, PWM, UART
from L298N_motor import L298N
import time

uart = UART(1, 9600)
uart.init(9600, bits = 8, parity = None, stop = 1, rx = Pin(9), tx = Pin(8))

ENA = PWM(Pin(0))
IN1 = Pin(1, Pin.OUT)
IN2 = Pin(2, Pin.OUT)

motor1 = L298N(ENA, IN1, IN2)
motor1.setSpeed(30000)

while True:
    if uart.any() > 0:
        data = uart.read()
        print(data)
```

El potenciómetro se utiliza aquí para controlar la velocidad de los motores. El código de ejemplo para esto es el siguiente:

Example-6

```
''' This is a micropython program to control the
speed of motor1 using a potentiometer'''

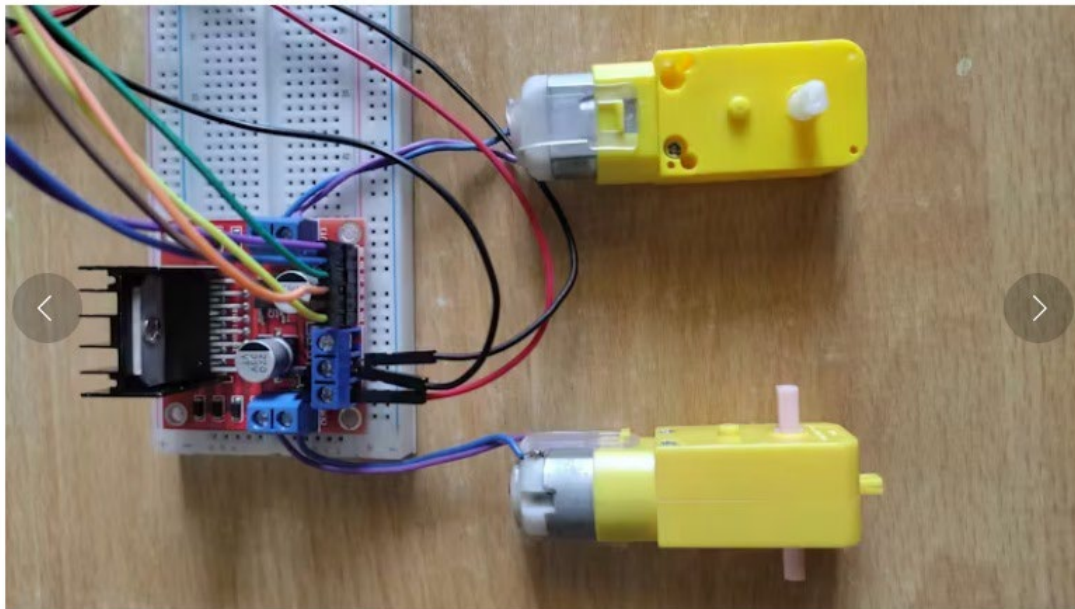
from machine import Pin, PWM, ADC
from L298N_motor import L298N
import time

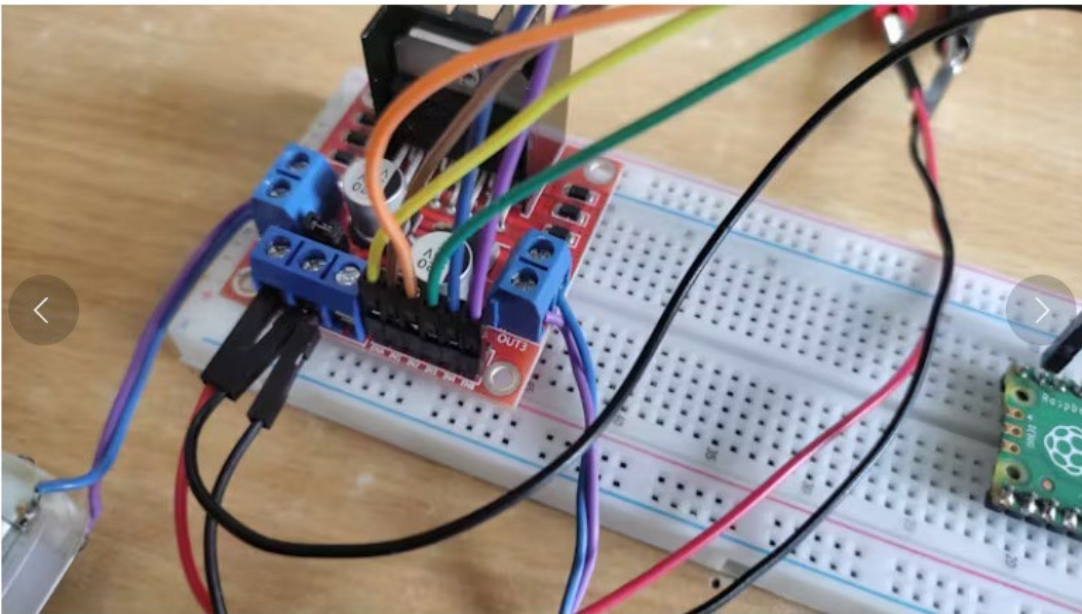
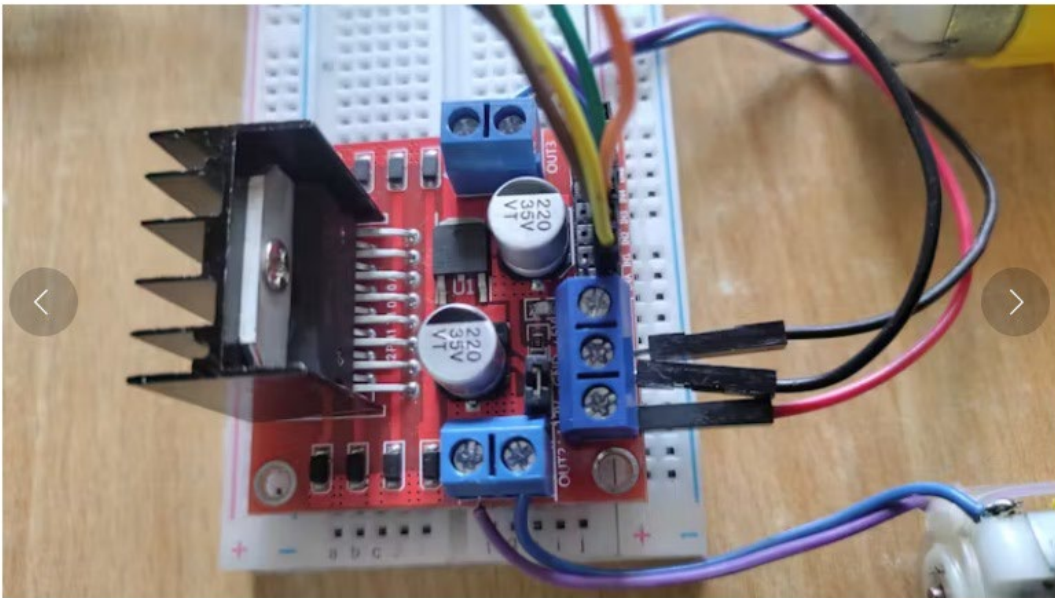
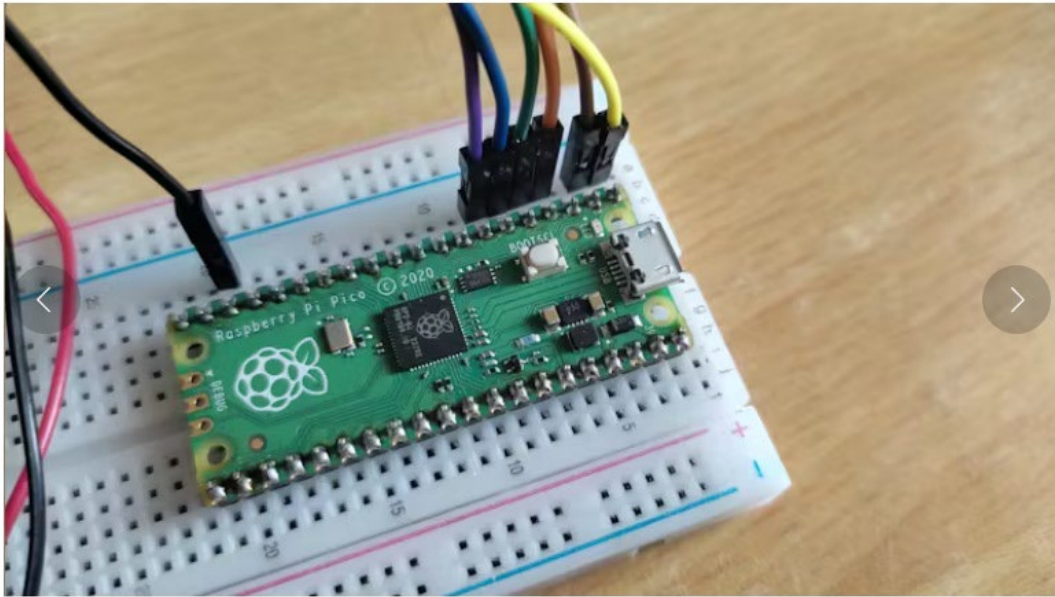
potentiometer = ADC(26)
ENA = PWM(Pin(0))
IN1 = Pin(1, Pin.OUT)
IN2 = Pin(2, Pin.OUT)

motor1 = L298N(ENA, IN1, IN2)

while True:
    reading = potentiometer.read_u16()
    motor1.setSpeed(int(reading))
    motor1.forward()
    time.sleep(0.1)
```

Imágenes de mi trabajo





Créditos



Ramji Patel

27 projects • 14 followers

Yo mismo, Ramji Patel. Soy estudiante de ingeniería y estoy cursando mi B-Tech en el Instituto de ingeniería y tecnología rural de Prayagraj, India.