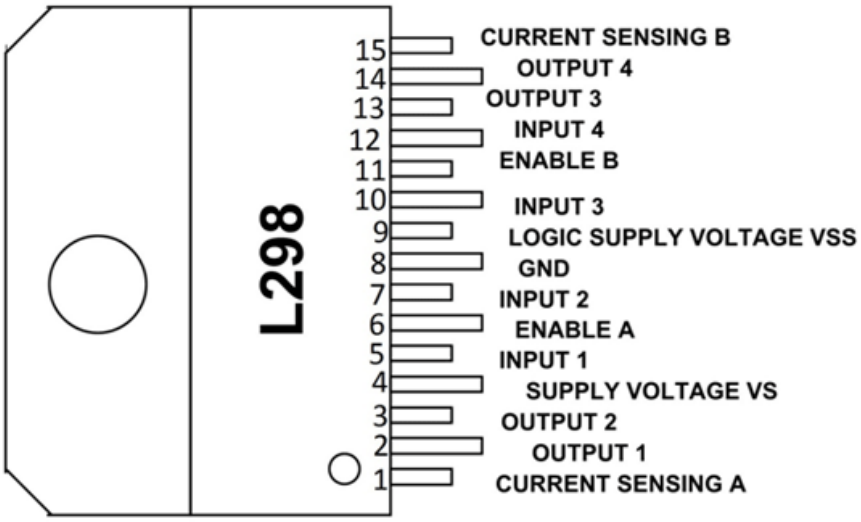
TEMA 4 . CONTROL DE MOTORES DC

Características del Driver L298

El L298 es un controlador de motores basado en un puente H que permite controlar motores de corriente continua (DC) o motores paso a paso. Es ampliamente utilizado en proyectos de robótica debido a su versatilidad y capacidad para manejar corrientes elevadas.

Especificaciones Técnicas

Característica	Detalle
Tipo	Circuito integrado de puente H dual
Voltaje de operación	5V a 46V
Corriente máxima	2A por canal (pico de 4A con disipador)
Cantidad de canales	2 (puede controlar 2 motores DC o 1 paso a paso)
Frecuencia de trabajo	Hasta 25 kHz (PWM)
Potencia máxima	25 W (dependiendo del disipador)
Protección térmica	Sí
Voltaje lógico	5V
Puentes H	2 (pueden operar de forma independiente)



Descripción de Pines

Pin	Descripción
`ENA`	Habilitación para el canal A (activa PWM)
`IN1, IN2`	Entradas de control del motor A
`ENB`	Habilitación para el canal B (activa PWM)
`IN3, IN4`	Entradas de control del motor B
`VSS`	Alimentación lógica (5V)
`VS`	Alimentación del motor (5V a 46V)
`GND`	Tierra común
`OUT1, OUT2`	Salidas del motor A
`OUT3, OUT4`	Salidas del motor B

Conexión Básica

1. Alimentación

- Conecta VS al voltaje del motor (5V-46V).
- Conecta VSS a 5V para la lógica.
- Une los pines de tierra (GND) de Arduino, el driver y la fuente de alimentación.

2. Control de un Motor DC

- Usa IN1 y IN2 para controlar la dirección del motor.
- Usa ENA con una señal PWM para ajustar la velocidad.

3. Control de un Motor Paso a Paso

- Conecta las bobinas del motor a los pines OUT1-OUT4.
- Controla el motor paso a paso utilizando señales de los pines IN1-IN4.
- Modo de control

IN1	IN2	Estado del Motor A
HIGH	LOW	Gira en una dirección
LOW	HIGH	Gira en la dirección opuesta
LOW	LOW	Motor apagado (libre)
HIGH	HIGH	Motor frenado (corto circuito)

Ventajas del L298

1. Control de dos motores: Puede manejar dos motores DC o un motor paso a paso.
2. Soporta altos voltajes: Hasta 46V para motores.
3. Corriente elevada: Hasta 2A por canal.
4. Protección térmica: Incluye un sistema que detiene el funcionamiento si se sobrecalienta.

Limitaciones

1. Eficiencia baja: Genera calor, por lo que necesita disipadores.
2. Corriente limitada: No soporta motores de alta corriente sin módulos adicionales.
3. Diseño antiguo: Existen controladores más modernos y eficientes (por ejemplo, L298N o TB6612FNG).

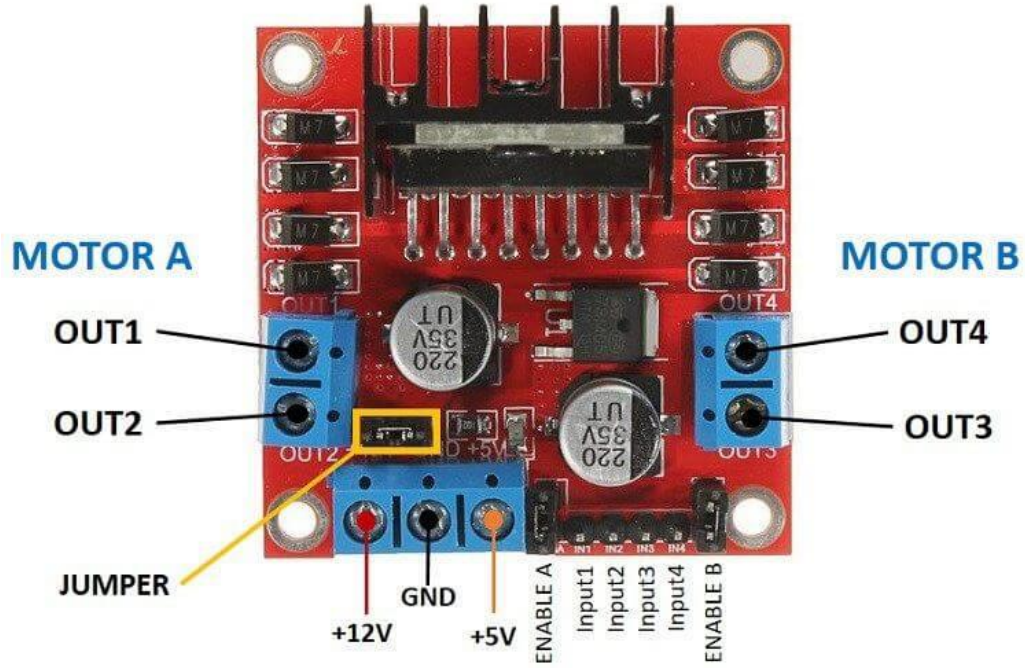
USO DE DRIVER L298 MODULO ARDUINO

Principio de funcionamiento

El módulo L298N se basa en el circuito integrado del mismo nombre. El circuito L298N es un doble puente H que permite cambiar la dirección y la intensidad de la tensión en el terminal de dos cargas eléctricas.

Las características del módulo L298N son:

- Control del motor de 5 a 35 V de tensión nominal
- 2A de corriente máxima (pico)
- Tensión de 5V aceptada en los pines de entrada

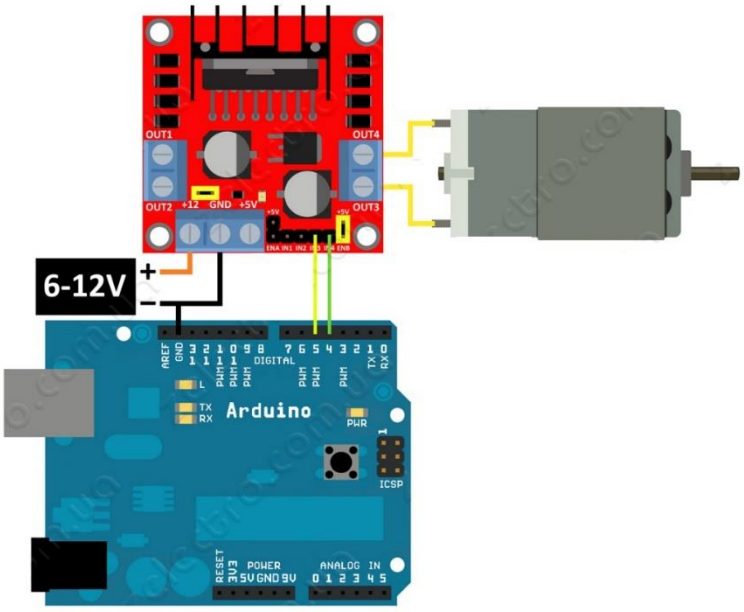


Control de un motor DC

Como demostración, vamos a controlar un motor DC a través de la salida B del módulo. El pin **ENB** se conectará con el jumper a +5V.

El ejemplo esta desarrollado en Arduino UNO, pero el código es compatible con cualquier Arduino o pinguino.

Esquema de conexión



Ejemplo de Control de Motor DC con Arduino

Circuito:

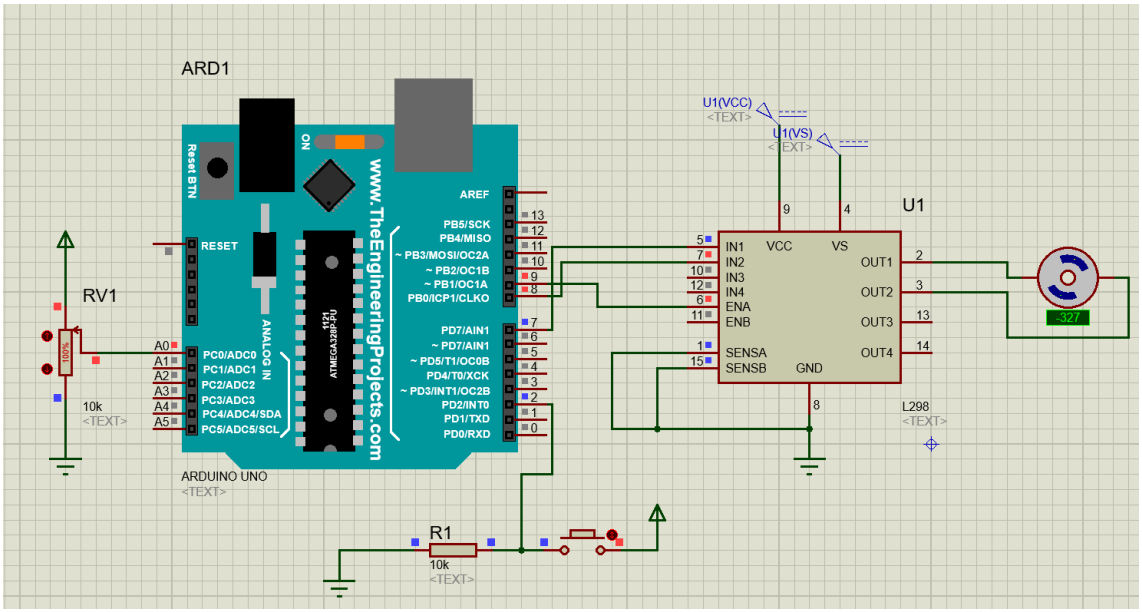
1. Conecta el motor a OUT1 y OUT2.
2. Conecta ENA al pin PWM de Arduino (por ejemplo, D9).
3. Conecta IN1 e IN2 a pines digitales de Arduino (por ejemplo, D7 y D8).
4. Alimenta VS con la fuente de alimentación del motor y VSS con 5V.

Programa ejemplo

```
const int ENA = 9;   // Pin PWM para velocidad
const int IN1 = 7;   // Control dirección
const int IN2 = 8;   // Control dirección

void setup() {
  pinMode(ENA, OUTPUT);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
}

void loop() {
  // Motor gira hacia adelante
  digitalWrite(IN1, HIGH);
  digitalWrite(IN2, LOW);
  analogWrite(ENA, 128); // Velocidad media (0-255)
  delay(2000);

  // Motor gira hacia atrás
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, HIGH);
  analogWrite(ENA, 128); // Velocidad media
  delay(2000);
```

```
  // Motor apagado
  digitalWrite(IN1, LOW);
  digitalWrite(IN2, LOW);
  analogWrite(ENA, 0);
  delay(2000);
}
```

4.1 CONTROL MOTOR DC CON PWM



Materiales necesarios

1. **Arduino Uno** (o cualquier otra placa Arduino).
2. **Driver L298N**.
3. **Motor DC**.
4. **Fuente de alimentación externa** (batería de 6-12V dependiendo de tu motor).
5. **Potenciómetro** (para controlar la velocidad).
6. **Protoboard** (opcional).
7. **Cables de conexión**.
8. **Pulsador**

Esquema de conexión

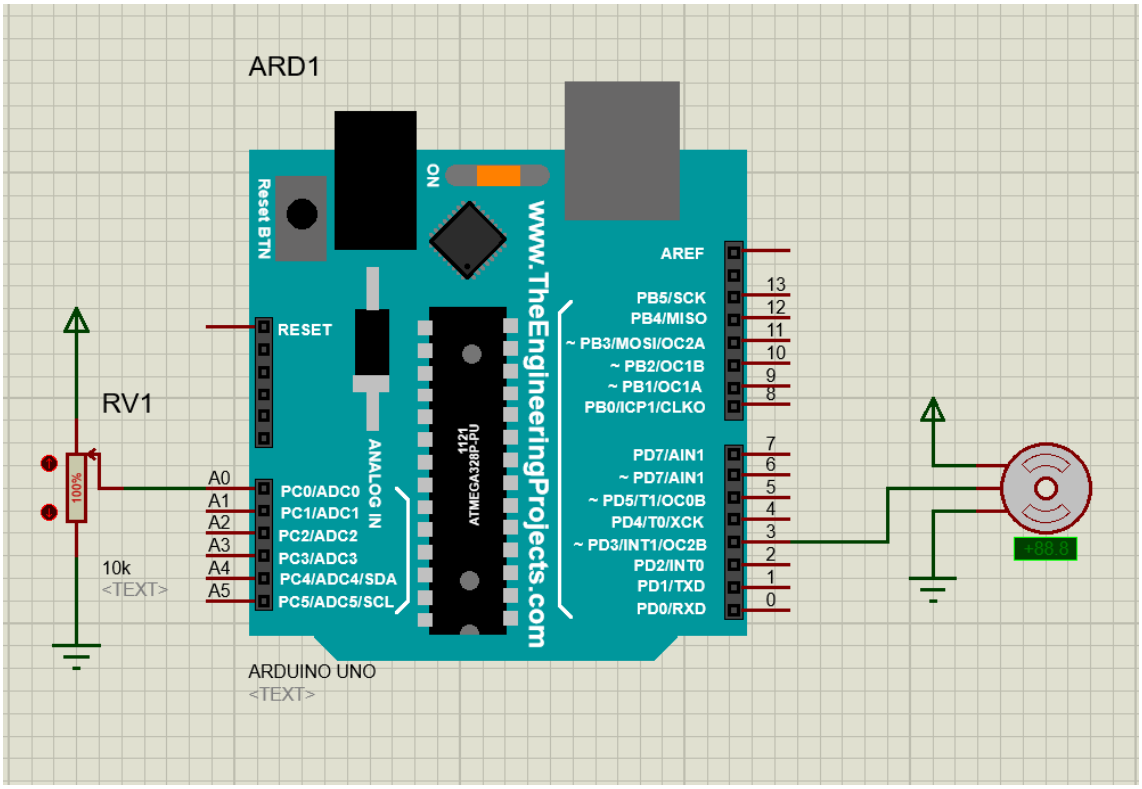
1. **Driver L298N:**
    - Conecta la **salida OUT1 y OUT2** a los terminales del motor DC.
    - Conecta **GND** del driver al GND de Arduino.
    - Conecta la **entrada IN1** al pin digital **7** de Arduino.
    - Conecta la **entrada IN2** al pin digital **8** de Arduino.
    - Conecta el pin **ENA** al pin PWM **9** de Arduino (para controlar la velocidad).
    - Alimenta el driver conectando la batería al pin **12V** y **GND** del L298N.
2. **Potenciómetro:**
    - Conecta un extremo a **5V**.
    - Conecta el otro extremo a **GND**.
    - Conecta la pata central (cursor) al pin analógico **A0** de Arduino.

4.1. PROGRAMA : CONTROL DE MOTOR DRIVER L298

```
const int IN1 = 7;       // Pin para controlar la dirección (sentido 1)
const int IN2 = 8;       // Pin para controlar la dirección (sentido 2)
const int ENA = 9;       // Pin PWM para controlar la velocidad
const int potPin = A0;   // Pin del potenciómetro (entrada analógica)
const int BOTON=2;
int val;
void setup() {
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(ENA, OUTPUT);

  pinMode(BOTON,INPUT);
```

```
}

void loop() {
  // Leer el valor del potenciómetro (0 a 1023)
  int potValue = analogRead(potPin);

  // Mapear el valor del potenciómetro a rango PWM (0 a 255)
  int speed = map(potValue, 0, 1023, 0, 255);
  val=digitalRead(BOTON);
  // Control de dirección
  if (val == HIGH)
     { // Gira en un sentido
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
  } else { // Gira en el otro sentido
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
  }
  // Controlar la velocidad del motor
  analogWrite(ENA, speed);
  delay(10); // Pequeño retraso para estabilizar
}
```

4.2.CONTROL MOTOR SERVO



Materiales Necesarios

1. **Componentes Electrónicos:**
   - 1 Arduino Uno o similar.
   - 1 Servomotor (como el SG90 o MG996R).
   - 1 Potenciómetro (10 kΩ recomendado).
   - 1 Protoboard.
   - Cables de conexión (macho-macho y macho-hembra).
2. **Herramientas:**
   - Computadora con el IDE de Arduino instalado.
   - Cable USB para conectar el Arduino.

Esquema de Conexiones

Aquí se detallan las conexiones necesarias para el proyecto:

Conexión del Servomotor:

- **Cable rojo (VCC):** Conectar al pin de 5V del Arduino.
- **Cable negro o marrón (GND):** Conectar a GND del Arduino.
- **Cable amarillo o naranja (Señal):** Conectar al pin digital 3 del Arduino.

Conexión del Potenciómetro:

- **Terminal izquierdo:** Conectar al pin de 5V del Arduino.
- **Terminal central:** Conectar al pin analógico A0 del Arduino.
- **Terminal derecho:** Conectar a GND del Arduino.

PROGRAMA

```
#include <Servo.h>

Servo myservo;  // Crear objeto para el servomotor

void setup() {
  myservo.attach(3);  // Conectar el servomotor al pin 3
  Serial.begin(9600); // Iniciar monitor serial
}

void loop() {
  int adc = analogRead(A0);              // Leer valor del potenciómetro
  int angulo = map(adc, 0, 1023, 0, 180); // Mapear valor a rango 0-180 grados
  myservo.write(angulo);                 // Enviar ángulo al servomotor
  Serial.print("Ángulo: ");              // Mostrar ángulo en serial
  Serial.println(angulo);
  delay(10);                             // Pequeño retraso
}
```