

MOTOR DE CORRIENTE CONTINUA 3 y 6V

El motor de corriente continua (denominado también motor de corriente directa, motor CC o motor DC por las iniciales en inglés "**direct current**") es una máquina que convierte energía eléctrica en mecánica, provocando un movimiento rotatorio, gracias a la acción de un campo magnético.

Algunas aplicaciones especiales de estos motores son los motores lineales, cuando ejercen tracción sobre un riel, servomotores y motores paso a paso. Además existen motores de DC sin escobillas (brushless en inglés) utilizados en el aeromodelismo por su bajo par motor y su gran velocidad.

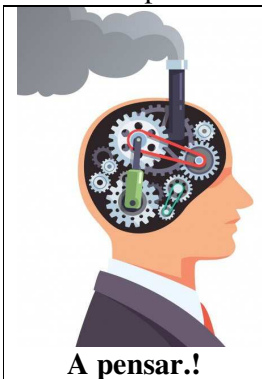
Un Motor, sin ningún tipo de aditamentos, generalmente tiene mucha velocidad, y poca fuerza. Para proveer de fuerza surgen las cajas reductoras o motorreductores.

Un **motorreductor** tiene un motor acoplado directamente, el **reductor** no tiene un motor acoplado directamente.

Un **motorreductor**, no es más que un sistema de engranajes y/o poleas. Estos dispositivos permiten Ganar fuerza a costa de la velocidad, o viceversa, ganar velocidad a costa de la fuerza, incluso desviar el movimiento en otra dirección, etc. Aunque no lo hayamos notado, prácticamente todo motor esta acompañado por algún tipo o variante de este maravilloso dispositivo llamado caja reductora (Sistema de engranajes o sistema de poleas).



Dicho de otra forma, los reductores y motorreductores son mecánicos de velocidad, se pueden encontrar entre los inventos más antiguos de la humanidad y aún en la actualidad (siglo XXI) se continúan utilizando prácticamente en cada máquina que tengamos a la vista, desde el más pequeño reductor o motorreductor capaz de cambiar y combinar velocidades de giro en un reloj de pulsera, cambiar velocidades en un automóvil, hasta enormes motorreductores capaces de dar tracción en buques de carga, molinos de cemento, grandes máquinas cavadoras de túneles o bien en molinos de caña para la fabricación de azúcar.



A pensar.!

La sencillez del principio de funcionamiento y su grado de utilidad en una gran variedad de aplicaciones es lo que ha construido la trascendencia de este invento al través de los siglos. Por ahora en nuestros primeros proyectos, usaremos motores CC con caja reductora, aunque más adelante reciclaremos motores y construiremos nuestras propias cajas reductoras.

Incluso construiremos poleas con madera y nuestros engranajes usando Poliestireno expandido, usualmente conocido como "telgopor" y moldes de Silicona.

ADVERTENCIA: Conectar un motor CC directamente con Arduino, no es una manera segura de trabajar (Podemos quemar nuestra placa), especialmente si desconocemos las especificaciones técnicas del motor. Recordar que cada pin I/O de Arduino, solo puede manejar 20 mA de corriente. Recomendando usar un puente H (Ver Apéndice A - El Puente H) y algo que permita alimentar desde otra fuente a los motores, dejando que Arduino, solo controle sus movimientos.



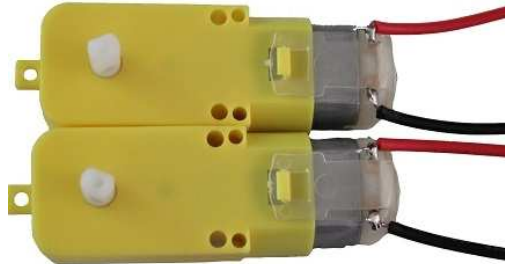
En Cuanto al avance y retroceso del motor, solo hay que invertir la polaridad de los cables de alimentación (conectados al motor), y el motor irá en un sentido u otro.

Motor Robot DC 3-6v + Caja reductora + Rueda goma

A continuación veremos uno de los motores más difundidos y utilizados para robots con ruedas. No es un simple motor, sino que tiene incorporado una caja con varios engranajes en su interior, que posibilita la reducción y estabilidad de la velocidad. Y trae adicionalmente una rueda. (Este motor es generalmente usado en pares).



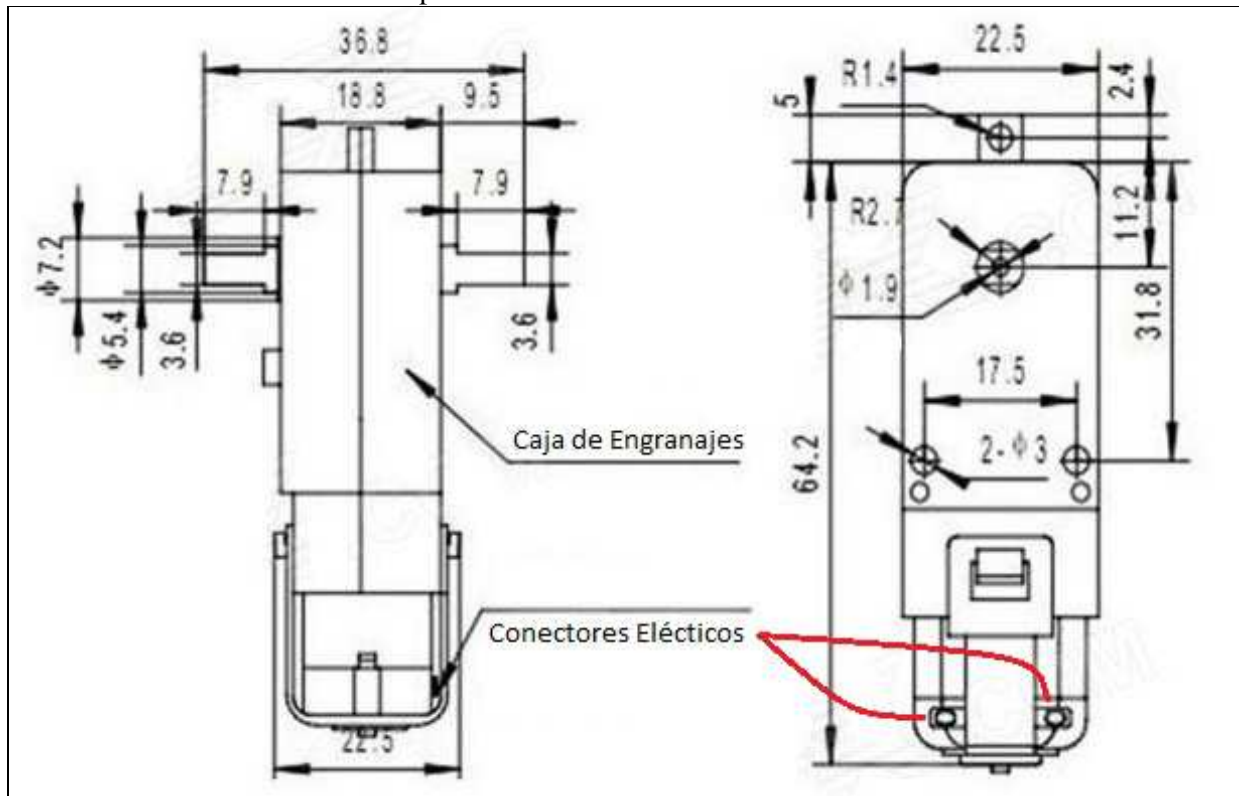
Motor Robot DC 3-6v + Caja reductora + Rueda goma



Como todo motor es muy simple de utilizar, y esta construido con medidas estándar, que facilita desarrollar dispositivos, cuyos planos se encuentran ampliamente difundidos en Internet (Chasis, y/o dispositivos par construir Autos, orugas, etc), que proveen el uso de estos motores.

A la Izquierda, vemos imagen ilustrativa, como se sueldan los cables de alimentación de cada motor y abajo el esquema con las medidas (en milímetros) de este motor y caja

reductora. A la hora de construir dispositivos les serán de de mucha utilidad.



Este será el motor elegido para nuestros trabajos, debido a que es muy fácil adquirirlo por su amplia difusión, relativamente económico, y para mantener uniformidad en la construcción de los proyectos. Más adelante, reciclaremos motores de aparatos en desuso (Equipos de Sonido, Impresoras, PCs, etc.) y aprenderemos y usarlos en forma segura (Sin quemar nuestras placas y equipos en general).

| Características Técnicas Del Motor | | | |
|------------------------------------|--------------------------------|-----------|-----------|
| Tensión | 3a 12 V - Recomendado: 5 a 8 V | | |
| | DC 3V | DC 5V | DC 6V |
| Velocidad sin carga (Aprox) | 125 RPM | 200 RPM | 230 RPM |
| Velocidad con carga (Aprox) | 95 RPM | 152 RPM | 175 RPM |
| Torque de salida (Aprox) | 0.8kg.cm | 1.0kg.cm | 1.1kg.cm |
| Corriente | 90-130mA | 120-140mA | 130-150mA |
| Diámetro máximo de llanta | 65mm | | |
| Ancho de llanta | 26 mm | | |
| Dimensiones del motor | 70mm x 22mm x 18mm | | |
| Peso del motor | 50g | | |
| Ruido | < 65dB | | |
| Reducción (engranajes) | 48:1 | | |
| Peso máximo sobre la rueda | 1.25kg | | |



Para el avance o retroceso del motor, hay que invertir la polaridad en cables de alimentación (conectados al motor), y el motor irá en un sentido u otro.

ADVERTENCIA:

Conectar un motor CC directamente con Arduino, no es una manera segura de trabajar (Podremos quemar nuestra placa), especialmente si desconocemos las especificaciones técnicas del motor.

Recordar que cada pin I/O de Arduino, solo puede manejar 20 mA de corriente.

Recomiendo usar un puente H (Ver Apéndice A - El Puente H) y algo que permita alimentar desde otra fuente a los motores, dejando que Arduino, solo controle sus movimientos.

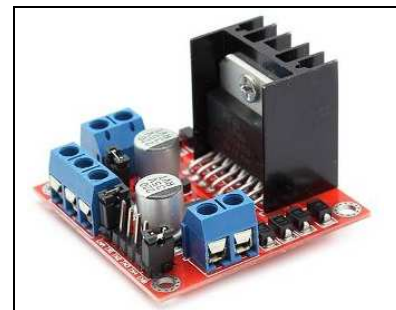
Es muy importante aprender a controlar motores de corriente continua y controladores de motores basados en puentes H. Ahora veremos un dispositivo, que nos permitirá alimentar externamente y usar, uno o dos motores de CC o incluso un motor paso a paso Bipolar, mientras los controlamos desde nuestra placa Arduino. Es decir, este dispositivo llamado "DRIVER", nos permite alimentar con alguna batería nuestros motores, mientras Arduino es alimentado desde otra fuente y a la vez controla los motores. Más Adelante construiremos nuestros propios "Drivers" o "Puentes H".

(Ver Apéndice A - El Puente H).

DRIVER L298N

(Para motores DC y paso a paso)

El módulo controlador de motores **L298N** nos permite controlar la velocidad y la dirección **de dos motores de corriente continua o un motor paso a paso bipolar, de hasta 2 amperios**, de una forma muy sencilla, gracias a los 2 los dos **H-bridge** (Puentes H) que contiene. (Ver Apéndice A - El Puente H). Además, podemos alimentarlo externamente con un amplio rango de tensiones (Acorde al motor que usemos), el rango de voltaje soportado por este Driver, oscila entre 6V y 35V.



Usaremos el controlador L298N porque es uno de los más baratos y difundidos actualmente, ya que además de controlar el sentido de **dos** motores (Adelante y atrás), nos permitirá controlar la velocidad de cada motor en forma independiente. Más adelante, podemos construir nosotros nuestros puentes H, un poco más simples y sin tantas prestaciones, pero funcionales y adaptables a las necesidades de nuestros dispositivos.

Características Técnicas del Driver L298N

Este módulo puede controlar simultáneamente hasta 2 motores de corriente directa o 1 (uno)

| |
|--|
| motor pasos a paso bifásicos (conexiones que veremos un poco más adelante). |
| Rango de voltaje soportado por este Driver, es de 6V a 35V. Para nuestros dispositivos, por ahora recomendable: 6V (Se entiende que usaremos un motor que funcione bien con 6V). |
| Posee 2 puentes H internos. Esto permite controlar dos motores, uno con cada PUENTE H. |
| Posee una gran inmunidad al ruido. |
| Tiene una protección que lo desactivará en caso de sobrecalentamiento. |
| Es capaz de soportar hasta 2 Amperios Lineales o incluso picos de 4 Amperios. |
| Tamaño: 43 x 43 x 26mm. |
| Peso Aproximado: 48Gr |

Conectando el DRIVER L298N

Se explicará la conexión de cada uno de los conectores o pin, que en la imagen, se encuentran numerados de 1 a 9. Para que sirva y cuáles son las opciones. Luego ejemplos concretos, donde podremos aplicar lo aprendido.

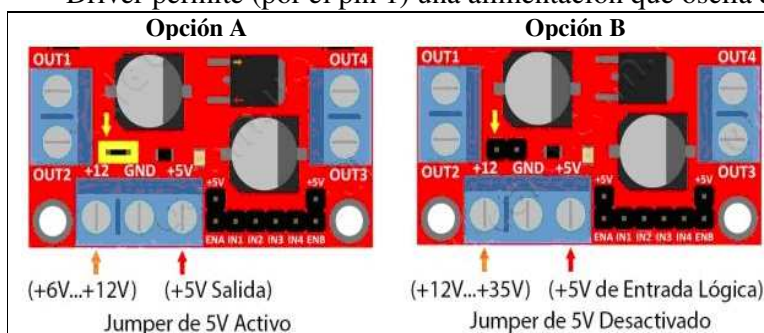
UNO A UNO CADA PIN

- 1) Para Alimentar este DRIVER y los motores que controla, hay dos maneras distintas, y estas se configuran mediante el Jumper que esta señalado en el punto 8 (Ver imagen).

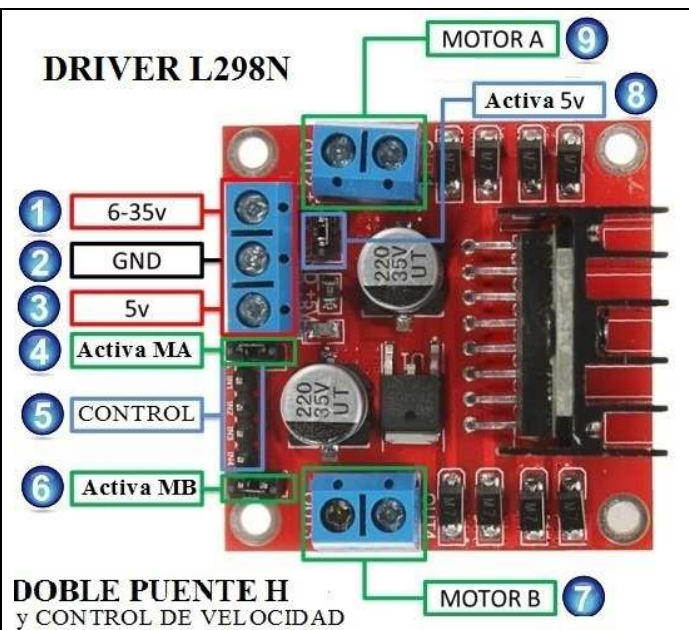
Es muy importante saber, que la tensión que ingrese por el conector marcado en el punto 1 (entre 6V y 35V), será la que se use para alimentar los motores en los puntos 7 y 9.

A) Cuando el jumper (regulador punto 8) se encuentra ACTIVO (está conectado), el Driver permite una alimentación (por el pin 1) que oscila entre 6V y 12V. En este caso, el pin marcado como 5V (punto 3) se comporta como salida y tendrá un voltaje de +5V que podemos usar para alimentar nuestra placa Arduino, pero el consumo no debe ser mayor a 500 mA. Por ahora, si no lo usamos mejor, evitaremos cometer errores fatales.

B) Cuando el jumper (regulador punto 8) se encuentra INACTIVO (No está conectado), el Driver permite (por el pin 1) una alimentación que oscila entre 12V y 35V.



externa cualquiera, asegurando así, una larga vida a nuestra Placa Arduino. :-)



Entonces, en este caso, el pin marcado como 5V (punto 3) se comporta como entrada y deberemos ingresar una tensión de +5V para alimentar la parte lógica del L298N (Nuestro Driver).

Curiosamente, ahora podrá ser Arduino el que alimente la parte lógica de nuestro Driver, aunque por ahora, para esta configuración, recomiendo una fuente



Recomendamos nunca conectar una tensión de entrada al pin de +5V (**punto 3**), cuando el jumper de selección de 5V (**regulador punto 8**) se encuentre activado (puesto). Esto provocaría un corto y podría dañar permanentemente nuestro Driver.

En nuestros primeros ejemplos usaremos el método explicado en el apartado A. Usaremos 4 pilas comunes AA (1,5V) - 6 voltios de alimentación para los motores y parte lógica del L298N (Nuestro Driver) - Por lo tanto, **el jumper (regulador punto 8) se encuentra ACTIVO (debe estar conectado)**.

- 2) En cualquier caso, GND deberá ser común a la Fuente que alimenta nuestro driver y a los dos motores y al GND de Arduino
- 3) **El uso de este PIN, quedo explicado junto en el Punto 1.**

A partir de este punto encontraremos 8 conectores (pines), mediante los cuales podremos controlar todo desde Arduino. Vista ampliada del grupo de conectores.

- 4) Hay dos conectores, el de arriba "5V" y el de abajo "ENA". ahora se nos presentan tres alternativas:

- ❖ Si dejamos desconectados ambos conectores ("5v" y "ENA"), el motor "A" (Izquierdo) quedará deshabilitado.



Vista Ampliada Conexión con Arduino



- ❖ Si conectamos el JUMPER uniendo estos dos conectores (tal como viene cuando lo compramos), estaremos dándole al motor "A" el máximo de velocidad (fijo), es decir perdemos la capacidad de controlar la velocidad.
- ❖ Finalmente, si dejamos desconectado solamente el PIN "5V" y conectamos ENA a un Pin de Arduino analógico, podremos controlar la velocidad del Motor "A" (izquierdo) con valores que oscilan entre 0 (cero) y 255. Un ejemplo de como manejar la velocidad del motor desde Arduino



| | |
|--|---|
| <code>analogWrite(PinENA, 180);</code> | Esta instrucción hará que el motor A se desplace a la velocidad de 180. Recordar que la velocidad puede oscilar entre 0 (cero) y 255. |
| <code>void Motor_A_Velocidad(int V){ analogWrite(PinENA, V); }</code> | Esta función, hará que el motor A se desplace a la velocidad que indique el parámetro "V". |

- 5) Acá analizaremos los pines IN1, IN2, IN3, IN4, que conectaremos a salidas digitales en Arduino, ellos nos permitirán controlar el sentido de giro, "**adelante**" y "**atrás**", de los Motores "A" y "B".

- ❖ Con los **IN1** e **IN2**, controlamos el sentido de giro del **motor A** y los pines **IN3** e **IN4** controlaremos el sentido de giro del **motor B**.
- ❖ Las **salidas digitales** solo pueden tomar 2 valores: **0** y **1**, **HIGH** y **LOW**, **BAJO** y **ALTO**, como prefieras llamarlos. Entonces, veamos que combinaciones podemos usar y que podemos hacer con dos motores.

| - | Motor | IN1 | IN2 | IN3 | IN4 | Acción del Motor |
|---|--|-------------|-------------|-------------|-------------|-----------------------|
| 1 | A (izquierda) (OUT1 y OUT2) | LOW | LOW | | | Detenido - (Motor A) |
| 2 | | LOW | HIGH | | | Avanza - (Motor A) |
| 3 | | HIGH | LOW | | | Retrocede - (Motor A) |
| 4 | | HIGH | HIGH | | | Detenido - (Motor A) |
| 5 | B (derecha) (OUT3 y OUT4) | | | LOW | LOW | Detenido - (Motor B) |
| 6 | | | | LOW | HIGH | Avanza - (Motor B) |
| 7 | | | | HIGH | LOW | Retrocede - (Motor B) |
| 8 | | | | HIGH | HIGH | Detenido - (Motor B) |

Por demás esta decir que:

- ✓ Si ambos motores se detienen, el dispositivo se detiene
- ✓ Si ambos motores avanzan, el dispositivo avanza
- ✓ Si ambos motores retroceden, el dispositivo retrocede
- ✓ Si motor "A" (izquierdo) avanza y el motor "B" (derecho) retrocede o se detiene, el dispositivo gira hacia la derecha
- ✓ Si motor "A" (izquierdo) retrocede o se detiene y el motor "B" (derecho) avanza, el dispositivo gira hacia la izquierdo.
- ✓ **Le dejo las otras combinaciones para que se divierta pensando.**

Un Ejemplo del código para manejar el motor A, podría ser:

| | |
|--|---|
| <pre>void Motor_A_Adelante(){ digitalWrite(PinIN1, LOW); digitalWrite(PinIN2, HIGH); }</pre> | Esta función, hace avanzar el Motor A |
| <pre>void Motor_A_Atras(){ digitalWrite(PinIN1, HIGH); digitalWrite(PinIN2, LOW); }</pre> | Esta función, hace retroceder el Motor A |
| <pre>void Motor_A_Parar(){ digitalWrite(PinIN1, LOW); digitalWrite(PinIN2, LOW); }</pre> | Esta función, detiene el Motor A |
| Las funciones homologas para el motor B, son idénticas a las anteriores, pero cambian los pines controlados. | |



Si el o los motores no giran en el sentido que esperaba, solo cambie los cables en los conectores en los puntos 7 o 9 - Es decir cambie la polaridad, y funcionara al revés.

- 6) Nuevamente encontraremos dos conectores, el de arriba "5V" y el de abajo "ENB". y repetimos lo mismo que vimos en el punto 4, pero esta vez es para el motor "B" y otra vez se nos presentan tres alternativas, que explico nuevamente:

- ❖ Si dejamos desconectados ambos conectores ("5v" y "ENB"), el motor "B" (Derecho) quedará deshabilitado.



- ❖ Si conectamos el JUMPER (tal como viene cuando lo compramos), uniendo estos dos conectores,



estaremos dándole al motor "B" el máximo de velocidad (fijo), es decir perdemos la capacidad de controlar la velocidad.

- ❖ Finalmente, si dejamos desconectado solamente el conector 5V y conectamos el pin ENB a un Pin de Arduino analógico, podremos controlar la velocidad del Motor "B" (Derecho) con valores que

oscilan entre 0 (cero) y 255. Un ejemplo de como manejar la velocidad del motor B desde Arduino.

| | |
|--|---|
| <code>analogWrite(PinENB, 180);</code> | Esta instrucción hará que el motor B se desplace a la velocidad de 180. Recordar que la velocidad puede oscilar entre 0 (cero) y 255. |
| <code>void Motor_B_Velocidad(int V){ analogWrite(PinENB, V); }</code> | Esta función, hará que el motor B se desplace a la velocidad que indique el parámetro "V". |

Resumiendo los Puntos 4, 5 y 6:

- Los pines ENA, IN1, IN2 corresponden a las entradas para controlar el MOTOR A (OUT1 y OUT2).

- De igual manera ENB, IN3, IN4 permiten controlar el MOTOR B (OUT3 y OUT4).

- Los conectores (pines del driver) IN1, IN2, IN3, IN4, los usaremos para controlar las direcciones de giro para cada uno de los motores (Adelante-Atras). Los pines **IN1** e **IN2**, para el **motor A**; y los pines **IN3** e **IN4** para el **motor B**. Cada pin estará conectado a una **salida digital** de Arduino.
- Las **salidas digitales** solo pueden tomar 2 valores: **0** y **1**, es decir "**HIGH**" y "**LOW**", **BAJO** y **ALTO**, como prefieran decir. Ejemplo: `digitalWrite(PinIN1, HIGH)`

- ENA y ENB, sirven para habilitar o deshabilitar sus respectivos motores, aunque generalmente se utilizan para controlar la velocidad, ingresando una señal de PWM por estos pines. Si no se usan se deben conectar los Jumper para que siempre estén habilitados.

- Las **salidas analógicas** o **PWM** pueden tomar valores entre **0** y **255**.
- Si le damos un valor demasiado bajo, no será suficiente para mover los motores.
- Si por el contrario le damos uno demasiado alto no vamos a tener suficiente potencia para mover las ruedas y hacer funcionar el receptor de control remoto y/o Bluetooth que usaremos un poco más adelante.
- Los componentes que sirven para comunicarse a distancia, como el Bluetooth, consumen mucha energía.

7) **Acá conectamos los dos cables del motor B.** Si el motor gira en el sentido inverso del esperado, simplemente invierta los cables (solo acá).

8) **El uso de este Jumper, quedo explicado junto en el Punto 1.**

9) **Acá conectamos los dos cables del motor A.** Si el motor gira en el sentido inverso del esperado, simplemente invierta los cables (solo acá).



Nota: Como siempre, si usamos alimentaciones diferentes para el controlador de los motores (el DRIVER) y Arduino, hay que unir los GND de ambas placas.

Donde ENCUENTRO MÁS JUMPER.?

Debido a su diminuto tamaño es común perderlos. no te preocupes, es muy fácil encontrar nuevos. Si tienes en desuso: discos duros (Hard Disk), lector o grabadores de CD o DVD, busca, ahí seguro encontraras uno o dos. En una Placa Madre de PC seguro encuentras varios, en viejos equipos de Sonido e impresoras nunca faltan.





Y si no dispones de nada de lo anterior, en cualquier casa de computación te pueden regalar algunos, o si los venden, son sumamente baratos. También puedes recurrir a cualquier casa donde venden circuitos electrónicos.

Sugerencia: Siempre deberías tener a mano una pinza para depilar, son muy practicas a la hora de poner y sacar los Jumpers, cuando quieres configurar un dispositivo. Casi te diría que deben ir de la mano en cualquier caja de

herramientas de un estudiante de robótica o electrónica.

COMENCEMOS CON LA PRÁCTICA

Arduino nos permite controlar motores de Corriente Continua con una facilidad extrema, sin embargo también existen librerías externas, que nos facilitan aun más la tarea de controlarlos desde nuestros programas. Me parece muy interesante incluir ejemplos del uso de estas librerías, y será por este motivo que algunos proyectos, tendrán dos programas, uno sin librería y otro que si usa librería externa.

La librería externa que usaremos es "[LEANTEC ControlMotor](#)", la que pueden descargar desde la sección de "**Librerías Adicionales**" de nuestra página.

1) Primeros Movimientos de un Motor DC de 6v – Avance, Retroceso y Parar – Usamos controlador L298N.

Este programa nos permite controlar los movimientos básicos de un motor DC de 6V. Adelante, Atrás y Parar. Tenemos activa la Capacidad de Controlar la velocidad, pero la dejamos fija en 150 (recordar que la velocidad puede oscilar entre 0 y 255) - En este caso usamos el Motor Izquierdo.



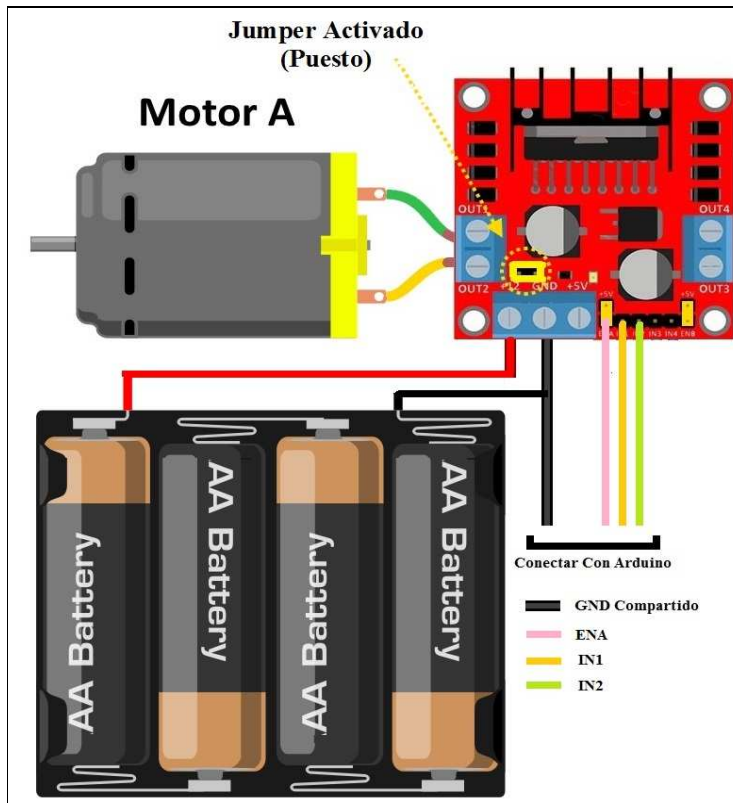
(Programa "010_Motores_DC_Driver_L298N_01_Un_Motor_Sin_Libreria")

| | |
|----|--|
| 1 | int PinIN1 = 3, //El pin 3 de Arduino se conecta con el pin In1 del L298N |
| 2 | PinIN2 = 5, //El pin 5 de Arduino se conecta con el pin In2 del L298N |
| 3 | PWM_Izquierdo_ENA = A0; //Pin Analógico se conecta con pin EnA del L298N (Velocidad) |
| - | |
| 4 | int V = 150; //Declaramos una variable para la velocidad |
| - | |
| 5 | void setup(){ |
| 6 | pinMode(PinIN1, OUTPUT); |
| 7 | pinMode(PinIN2, OUTPUT); |
| 8 | pinMode(PWM_Izquierdo_ENA, OUTPUT); |
| 9 | } |
| - | |
| 10 | void Adelante_MA(int Veloc){ // Motor Izquierdo |
| 11 | digitalWrite(PinIN1,LOW); |
| 12 | digitalWrite(PinIN2,HIGH); |
| 13 | analogWrite(PWM_Izquierdo_ENA, Veloc); |
| 14 | } |
| - | |
| 15 | void Atras_MA(int Veloc){ // Motor Izquierdo |
| 16 | digitalWrite(PinIN1, HIGH); |
| 17 | digitalWrite(PinIN2, LOW); |
| 18 | analogWrite(PWM_Izquierdo_ENA, Veloc); |
| 19 | } |
| 20 | void Parar_MA(){ // Motor Izquierdo |
| 21 | digitalWrite(PinIN1,LOW); |
| 22 | digitalWrite(PinIN2,LOW); |
| 23 | analogWrite(PWM_Izquierdo_ENA, 0); |

| | |
|----|---|
| 24 | } |
| 25 | void loop() { |
| 26 | Adelante_MA(V); |
| 27 | delay(3000); //Durante 3 segundos ejecutamos función anterior |
| 28 | Parar_MA(); |
| 29 | delay(3000); //Durante 3 segundos ejecutamos función anterior |
| 30 | Atras_MA(V); |
| 31 | delay(3000); //Durante 3 segundos ejecutamos función anterior |
| 32 | Parar_MA(); |
| 33 | delay(3000); //Durante 3 segundos ejecutamos función anterior |
| 34 | } |

CIRCUITO PARA NUESTRO PROYECTO

Lista de Materiales: 1 Porta Pilas para 4 pilas comunes AA (1,5V) - 1 **Motor DC 6v con Caja reductora y Rueda goma** - 1 Driver **L298N** - Cables de conexión (dependiendo del modelo de Arduino serán Hembra/Hembra o Macho/Hembra) - Placa Arduino y 1 Cable USB.



En nuestros primeros ejemplos, para alimentar el o los motores, conectados a nuestro Drives, aplicaremos lo explicado en el punto 1- A, referente a la forma de conectar el **DRIVER L298N**. Usaremos 4 pilas comunes AA (1,5V c/u), es decir 6 voltios como fuente de alimentación para los motores y parte lógica del L298N (Nuestro Driver) - Por lo tanto, el **jumper (regulador punto 8)** se encuentra **ACTIVO** (debe estar conectado).

Los cables deberán ser conectados: De Izquierda a Derecha, Cable Rojo, al Positivo del Porta Pilas. **Cable Negro** al Negativo del Porta Pilas y GND común de Arduino. **Cable Rosa:** Habiendo quitado el Jumper, deberemos conectarlo al Pin Analógico de Arduino que se configure en el Programa (en este ejemplo A0), recordar que con este pin controlamos la velocidad con que gira el motor. **Cable Amarillo,** al Pin Digital que se haya configurado en el programa a IN1 (en este caso 3).

Cable Verde, al Pin Digital que se haya configurado en el programa a IN2 (en este caso 5).



Si el motor no gira en el sentido que esperaba, solo invierta los cables en el conector del motor A - Es decir cambie la polaridad, y funcionará al revés.

2) Manejar dos Motores DC de 6v simultáneamente, controlando Avance, Retroceso, Parar, Giro a la derecha y giro a la Izquierda – Usamos controlador L298N.

Este programa nos permite controlar los movimientos de un motor DC de 6V. Tenemos activa la Capacidad de Controlar la velocidad, pero la dejamos fija en 150 (recordar que la velocidad puede oscilar entre 0 y 255).



```

1  int PinIN1 = 3,    //El pin 3 de Arduino se conecta con el pin In1 del L298N
2      PinIN2 = 5,    //El pin 5 de Arduino se conecta con el pin In2 del L298N
3      PWM_Izquierdo_ENA = A0; //Pin Analógico Arduino (Control de Velocidad)
4  int PinIN3 = 7,    //Pin Digital que conecta Arduino con el pin In3 del L298N
5      PinIN4 = 9,    //Pin Digital que conecta Arduino con el pin In4 del L298N
6      PWM_Derecho_ENB = A1; //Pin Analógico Arduino (Control de Velocidad)
7
8  int V = 150; //Declaramos una variable para guardar la velocidad
9
10 void setup( ){
11     pinMode(PinIN1, OUTPUT);
12     pinMode(PinIN2, OUTPUT);
13     pinMode(PWM_Izquierdo_ENA, OUTPUT);
14     pinMode(PinIN3, OUTPUT);
15     pinMode(PinIN4, OUTPUT);
16     pinMode(PWM_Derecho_ENB, OUTPUT);
17 }
18
19 void Adelante_MA(int Veloc){ // Motor Izquierdo
20     digitalWrite(PinIN1,LOW);
21     digitalWrite(PinIN2,HIGH);
22     analogWrite(PWM_Izquierdo_ENA, Veloc);
23 }
24
25 void Adelante_MB(int Veloc){ // Motor Derecho
26     digitalWrite(PinIN3,LOW);
27     digitalWrite(PinIN4,HIGH);
28     analogWrite(PWM_Derecho_ENB, Veloc);
29 }
30
31 void Atras_MA(int Veloc){ // Motor Izquierdo
32     digitalWrite(PinIN1,HIGH);
33     digitalWrite(PinIN2,LOW);
34     analogWrite(PWM_Izquierdo_ENA, Veloc);
35 }
36
37 void Atras_MB(int Veloc){ // Motor Derecho
38     digitalWrite(PinIN3,HIGH);
39     digitalWrite(PinIN4,LOW);
40     analogWrite(PWM_Derecho_ENB, Veloc);
41 }
42
43 void Parar_MA( ){ // Motor Izquierdo
44     digitalWrite(PinIN1,LOW);
45     digitalWrite(PinIN2,LOW);
46     analogWrite(PWM_Izquierdo_ENA, 0);
47 }
48
49 void Parar_MB( ){ // Motor Derecho
50     digitalWrite(PinIN3,LOW);
51     digitalWrite(PinIN4,LOW);
52     analogWrite(PWM_Derecho_ENB, 0);
53 }
54

```

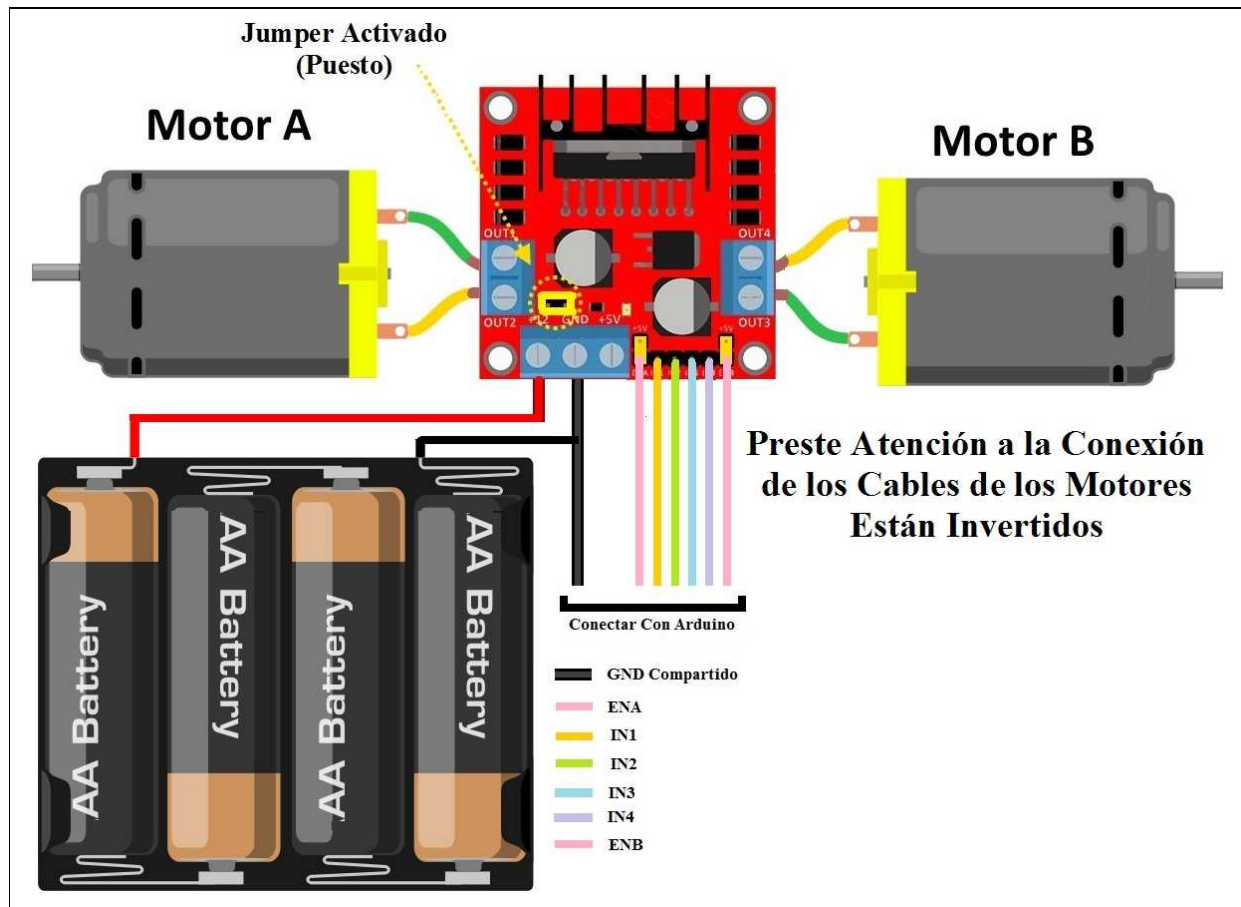
```

46 void Avanzar(int Velocidad){
47     Adelante_MA(Velocidad);
48     Adelante_MB(Velocidad);
49 }
-
50 void Retroceder(int Velocidad){
51     Atras_MA(Velocidad);
52     Atras_MB(Velocidad);
53 }
-
54 void Alto( ){
55     Parar_MA( );
56     Parar_MB( );
57 }
-
58 void Derecha(int Velocidad){
59     Adelante_MA(Velocidad);
60     Parar_MB( );
61     // Atras_MB(Velocidad);
62 }
-
63 void Izquierda(int Velocidad){
64     Parar_MA( );
65     // Atras_MA(Velocidad);
66     Adelante_MB(Velocidad);
67 }
-
68 void loop( ){
69     Avanzar(V);
70     delay(3000); //Durante 3 segundos ejecutamos función anterior
71     Alto( );
72     delay(3000); //Durante 3 segundos ejecutamos función anterior
73     Retroceder(V);
74     delay(3000); //Durante 3 segundos ejecutamos función anterior
75     Alto( );
76     delay(3000); //Durante 3 segundos ejecutamos función anterior
77     //////////////////////////////////////
78     Derecha(V);
79     delay(3000); //Durante 3 segundos ejecutamos función anterior
80     Avanzar(V);
81     delay(3000); //Durante 3 segundos ejecutamos función anterior
82     Alto( );
83     delay(3000); //Durante 3 segundos ejecutamos función anterior
84     //////////////////////////////////////
85     Izquierda(V);
86     delay(3000); //Durante 3 segundos ejecutamos función anterior
87     Retroceder(V);
88     delay(3000); //Durante 3 segundos ejecutamos función anterior
89     Alto( );
90     delay(3000); //Durante 3 segundos ejecutamos función anterior
91 }

```

CIRCUITO PARA NUESTRO PROYECTO

Lista de Materiales: 1 Porta Pilas para 4 pilas comunes AA (1,5V) - 2 Motores DC 6v con Caja reductora y Rueda goma - 1 Driver L298N - Cables de conexión (dependiendo del modelo de Arduino serán Hembra/Hembra o Macho/Hembra) - Placa Arduino y 1 Cable USB.



Si el o los motores no giran en el sentido que esperaba, solo invierta los cables en el conector del motor A o del motor B según sea el caso - Es decir cambie la polaridad, y funcionará al revés.

Los cables deberán ser conectados: De Izquierda a Derecha, Cable Rojo, al Positivo del Porta Pilas. Cable Negro al Negativo del Porta Pilas y GND común de Arduino. Cable Rosado Al Pin Analógico que se configure en el Programa ENA (en este ejemplo A0). Cable Amarillo, al Pin Digital que se haya configurado en el programa a IN1 (en este caso 3). Cable Verde, al Pin Digital que se haya configurado en el programa a IN2 (en este caso 5). Cable Celeste, al Pin Digital que se haya configurado en el programa a IN3 (en este caso 7). Cable Morado, al Pin Digital que se haya configurado en el programa a IN4 (en este caso 9). Cable Rosado (Ultimo del grupo) Al Pin Analógico que se configure en el Programa ENB (en este ejemplo A1).

En nuestros primeros ejemplos, para alimentar el o los motores, conectados a nuestro Drives, aplicaremos lo explicado en el punto 1- A, referente a la forma de conectar el DRIVER L298N. Usaremos 4 pilas comunes AA (1,5V c/u), es decir 6 voltios como fuente de alimentación para los motores y parte lógica del L298N (Nuestro Driver) - Por lo tanto, el jumper (regulador punto 8) se encuentra ACTIVO (debe estar conectado).

3) Manejar dos Motores DC de 6v simultáneamente, controlando Avance, Retroceso, Parar, Giros a la derecha e Izquierda – Usamos controlador L298N y Librería "LEANTEC_ControlMotor".



Este programa nos permite controlar los movimientos de un motor DC de 6V. Tenemos activa la Capacidad de Controlar la velocidad (recordar que la velocidad puede oscilar entre 0 y 255). En este proyecto ganamos la capacidad de controlar cuanto queremos girar.

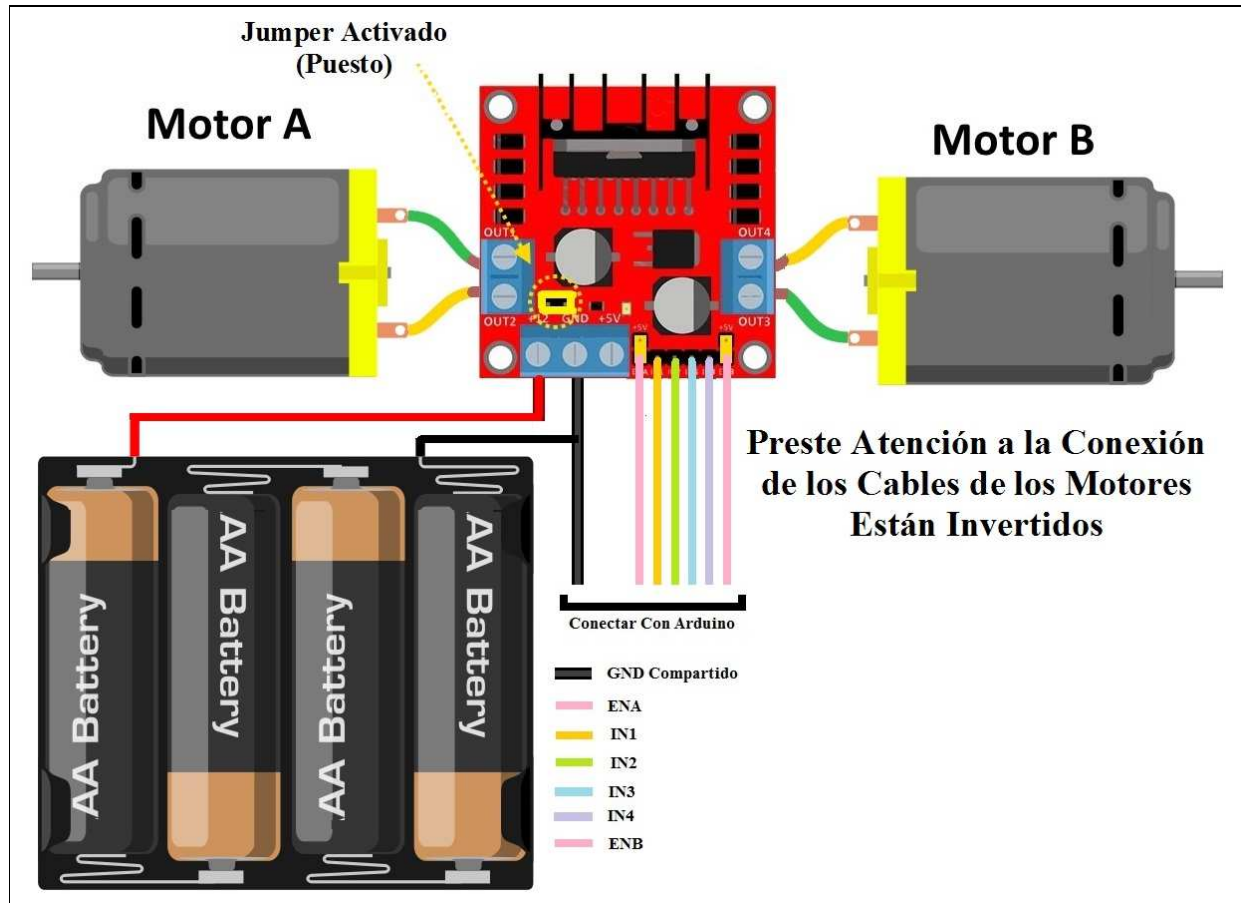
(Programa "010_Motores_DC_Driver_L298N_03_Dos_Motores_Con_Libreria")

| | |
|----|---|
| 1 | #include <LEANTEC_ControlMotor.h> //Incluimos la librería control de motores |
| - | |
| 2 | int PinIN1 = 3, //El pin 3 de Arduino se conecta con el pin In1 del L298N |
| 3 | PinIN2 = 5, //El pin 5 de Arduino se conecta con el pin In2 del L298N |
| 4 | PWM_Izquierdo_ENA = A0; //Pin Analógico Arduino para ENB del L298N (Velocidad) |
| - | |
| 5 | int PinIN3 = 7, //Pin Digital que conecta Arduino con el pin In3 del L298N |
| 6 | PinIN4 = 9, //Pin Digital que conecta Arduino con el pin In4 del L298N |
| 7 | PWM_Derecho_ENB = A1; //Pin Analógico Arduino para ENB del L298N (Velocidad) |
| - | |
| 8 | // Declaramos variable, Inicializamos y Configuramos los pines que vamos a usar |
| 9 | ControlMotor control(PinIN1, PinIN2, PinIN3, PinIN4, PWM_Izquierdo_ENA, PWM_Derecho_ENB); |
| - | |
| 10 | int velocidad; |
| - | |
| 11 | void setup(){ |
| 12 | Serial.begin(9600); |
| 13 | while (!Serial) { |
| 14 | ; // Esperamos que el puerto serie este abierto. |
| 15 | } |
| 16 | // No necesitamos configurar pines, la librería lo hace por nosotros. |
| 17 | } |
| - | |
| 18 | void loop(){ |
| 19 | velocidad = 10; // Es la velocidad en la que empezará a moverse el Robot. |
| - | |
| 20 | Serial.println("Adelante y Acelerando (desde 10 a 255)"); |
| 21 | while(velocidad<254){ //El robot avanza y acelera. |
| 22 | velocidad++; |
| 23 | control.Motor(velocidad,1); |
| 24 | delay (50); |
| 25 | } |
| - | |
| 26 | Serial.println("Adelante a Velocidad Fija = 180"); |
| 27 | control.Motor(180,1); // El robot Avanza recto, a velocidad de 180. |
| 28 | delay(3000); |
| - | |
| 29 | Serial.println("Atras a Velocidad Fija = 180"); |
| 30 | control.Motor(-180,1); // El robot retrocede recto, a velocidad de 180. |
| 31 | delay(3000); |
| - | |
| 32 | Serial.println("Gira a la Izquierda con Velocidad 180"); |
| 33 | control.Motor(180,100); // El robot gira 100% a la Izquierda. |
| 34 | delay(3000); |
| - | |
| 35 | Serial.println("Gira a la Derecha con Velocidad 180"); |
| 36 | control.Motor(180,-100); // El Robot gira 100% a la Derecha. |

| | |
|----|--|
| 37 | delay(3000); |
| - | |
| 38 | Serial.println("Se detiene"); |
| 39 | control.Motor(0,1); //Los motores paran. |
| 40 | delay(5000); |
| 41 | } |

CIRCUITO PARA NUESTRO PROYECTO

Lista de Materiales: 1 Porta Pilas para 4 pilas comunes AA (1,5V) - 2 Motores DC 6v con Caja reductora y Rueda de Goma - 1 Driver L298N - Cables de conexión (dependiendo del modelo de Arduino serán Hembra/Hembra o Macho/Hembra) - Placa Arduino y 1 Cable USB.



Si el o los motores no giran en el sentido que esperaba, solo invierta los cables en el conector del motor A o del motor B según sea el caso - Es decir cambie la polaridad, y funcionará al revés.

Los cables deberán ser conectados: De Izquierda a Derecha, Cable Rojo, al Positivo del Porta Pilas. Cable Negro al Negativo del Porta Pilas y GND común de Arduino. Cable Rosado Al Pin Analógico que se configure en el Programa ENA (en este ejemplo A0). Cable Amarillo, al Pin Digital que se haya configurado en el programa a IN1 (en este caso 3). Cable Verde, al Pin Digital que se haya configurado en el programa a IN2 (en este caso 5). Cable Celeste, al Pin Digital que se haya configurado en el programa a IN3 (en este caso 7). Cable Morado, al Pin Digital que se haya configurado en el programa a IN4 (en este caso 9). Cable Rosado (Ultimo del grupo) Al Pin Analógico que se configure en el Programa ENB (en este ejemplo A1).

En nuestros primeros ejemplos, para alimentar el o los motores, conectados a nuestro Drives, aplicaremos lo explicado en el punto 1- A, referente a la forma de conectar el **DRIVER L298N**.
Usaremos 4 pilas comunes AA (1,5V c/u), es decir 6 voltios como fuente de alimentación para los motores y parte lógica del L298N (Nuestro Driver) - Por lo tanto, **el jumper (regulador punto 8) se encuentra ACTIVO (debe estar conectado)**.

4) Controlar el Avance y Retroceso de dos Motores DC de 6v simultáneamente por medio de un Potenciómetro – Usamos controlador L298N y Librería "LEANTEC_ControlMotor".



Este programa nos permite controlar los movimientos adelante y atrás de dos motores DC de 6V. Tenemos activa la Capacidad de Controlar la velocidad (recordar que la velocidad puede oscilar entre 0 y 255).

(Programa "010_Motores_DC_Driver_L298N_04_Dos_Motores_Un_Poten_Y_Libreria")

| | |
|----|---|
| 1 | #include <LEANTEC_ControlMotor.h> <i>//Incluimos la librería control de motores</i> |
| - | |
| 2 | int PinIN1 = 3, <i>//El pin 3 de Arduino se conecta con el pin In1 del L298N</i> |
| 3 | PinIN2 = 5, <i>//El pin 5 de Arduino se conecta con el pin In2 del L298N</i> |
| 4 | PWM_Izquierdo_ENA = A0; <i>//Pin Analógico Arduino, pin EnB del L298N (Velocidad)</i> |
| - | |
| 5 | int PinIN3 = 7, <i>//Pin Digital que conecta Arduino con el pin In3 del L298N</i> |
| 6 | PinIN4 = 9, <i>//Pin Digital que conecta Arduino con el pin In4 del L298N</i> |
| 7 | PWM_Derecho_ENB = A1; <i>//Pin Analógico Arduino, pin EnA del L298N (Velocidad)</i> |
| - | |
| 8 | int PinPotenciometro = A2, |
| 9 | velocidad; |
| - | |
| 10 | double ValorOriginal; |
| - | |
| 11 | <i>// Declaramos variable y Configuramos los pines que vamos a usar</i> |
| 12 | ControlMotor control(PinIN1, PinIN2, PinIN3, PinIN4, PWM_Izquierdo_ENA, PWM_Derecho_ENB); |
| - | |
| 13 | void setup() { |
| 14 | Serial.begin(9600); |
| 15 | while (!Serial) { |
| 16 | <i>; // Esperamos que el puerto serie este abierto.</i> |
| 17 | } |
| 18 | <i>// No necesitamos configurar pines, la librería lo hace por nosotros.</i> |
| 19 | } |
| - | |
| 20 | void loop() { |
| 21 | ValorOriginal = analogRead(PinPotenciometro); <i>// realizar la lectura analógica</i> |
| 22 | ValorOriginal = ValorOriginal - 512; <i>//Resto 512, para que el valor Medio represente 0 (Cero).</i> |
| 23 | <i>// velocidad = (ValorOriginal / 512) *255;</i> |
| 24 | velocidad = map(ValorOriginal, -512, 512, -255, 255); |
| 25 | Serial.print("Velocidad = "); |
| 26 | Serial.println(velocidad); |
| 27 | delay(50); |
| 28 | control.Motor(velocidad,1); |
| 29 | delay(50); |
| 30 | } |

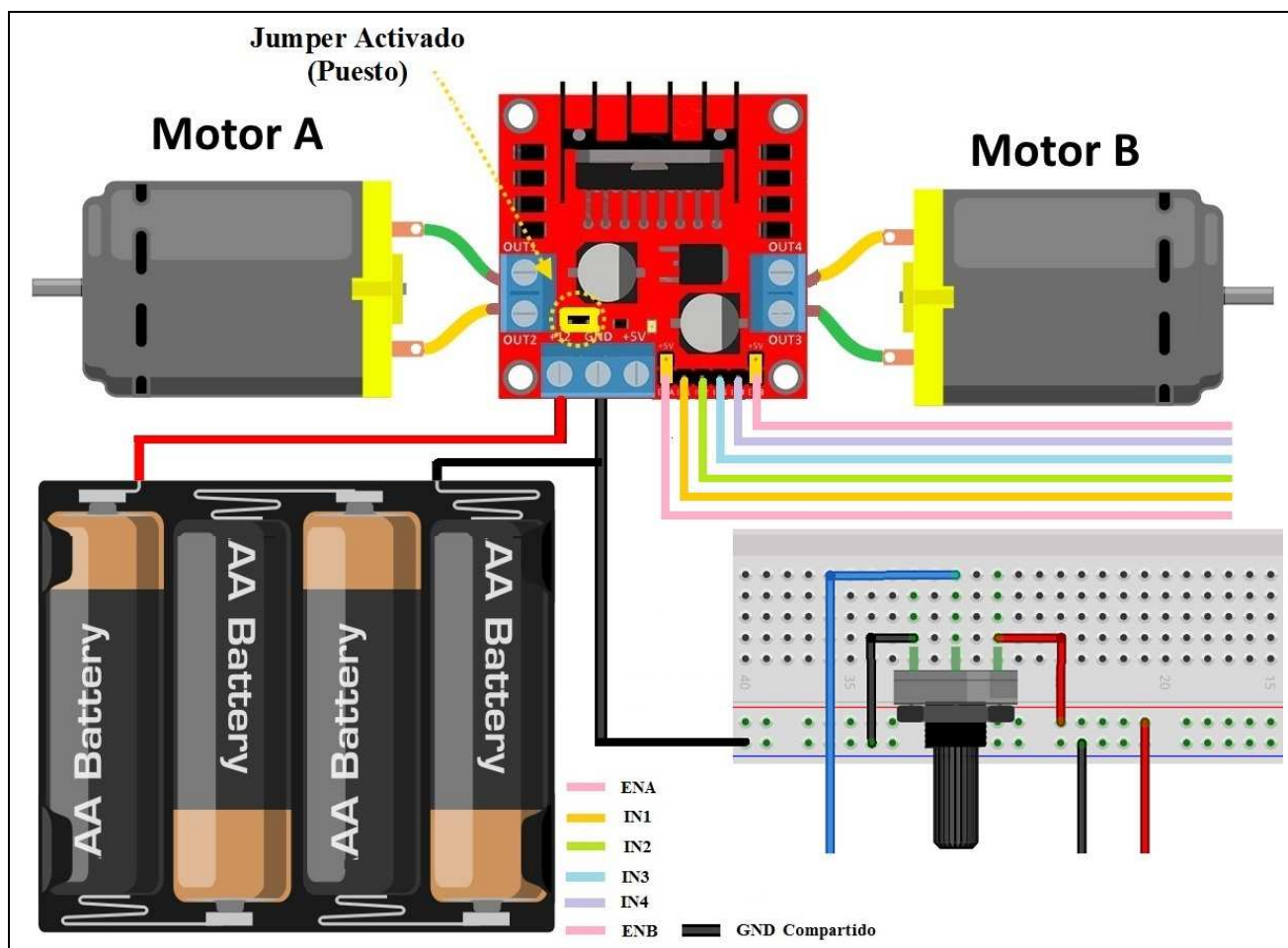
Explicación Líneas Importantes

| | |
|----|--|
| 21 | Leo valor entregado por potenciómetro |
| 22 | En esta línea, resto 512 (es la mitad del máximo: 1024 / 2), al valor entregado por el |

| | |
|---------|---|
| | potenciómetro. De esta forma cuando el potenciómetro entrega 512, el resultado será cero, por lo tanto el motor estará detenido. Si el potenciómetro entrega el máximo, el resultado será 511 (1023-512) por lo tanto la velocidad será la máxima hacia adelante.. Y finalmente si el potenciómetro entregara un cero, el resultado será -512 (0-512) por lo tanto la velocidad será la máxima hacia atrás. |
| 23 y 24 | Estas dos líneas hacen exactamente lo mismo, transforman el valor del potenciómetro a valores admisibles para la velocidad. Una esta comentariada, pero me parece interesante que sepan que hay dos formas de hacer lo mismo |
| 28 | Damos la Orden a los motores que se desplacen en la dirección y velocidad indicada |

CIRCUITO PARA NUESTRO PROYECTO

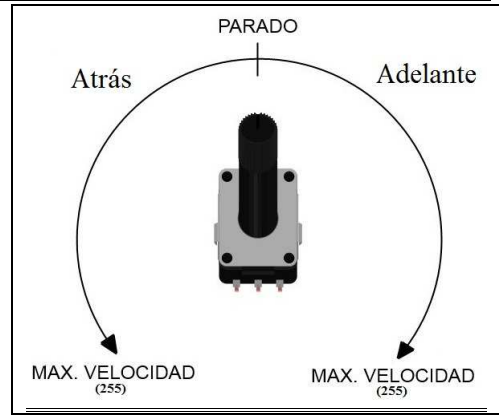
Lista de Materiales: 1 Porta Pilas para 4 pilas comunes AA (1,5V) - 2 Motores DC 6v con Caja reductora y Rueda de Goma - 1 Driver L298N - 1 Potenciómetro - Cables de conexión (dependiendo del modelo de Arduino serán Hembra/Hembra o Macho/Hembra) - 1 Protoboard - Placa Arduino y 1 Cable USB.



Si el o los motores no giran en el sentido que esperaba, solo invierta los cables en el conector del motor A o del motor B según sea el caso - Es decir cambie la polaridad, y funcionará al revés.

Los cables deberán ser conectados: Primero los cables de alimentación que salen del DRIVER L298N, entonces el Cable Rojo, directo al Positivo del Porta Pilas. Cable Negro, debe conectarse con el negativo del Porta Pilas y el Negativo o GND de la Protoboard, donde estará el Potenciómetro. Ahora veamos el grupo de cables de están a la derecha. Veremos como debemos conectarlos desde arriba hacia abajo.

Comenzamos por el Cable Rosado, que conectaremos al Pin Analógico que se configure en el Programa ENB (en este ejemplo A1). Cable Morado, al Pin Digital que se haya configurado en el programa a IN4 (en este caso 9). Cable Celeste, al Pin Digital que se haya configurado en el programa a IN3 (en este caso 7). Cable Verde, al Pin Digital que se haya configurado en el programa a IN2 (en este caso 5). Cable Amarillo, al Pin Digital que se haya configurado en el programa a IN1 (en este caso 3). Cable Rosado (Ultimo del grupo, cerda de la Protoboard) deberemos conectarlo al Pin Analógico que se configure en el Programa ENA (en este ejemplo A0). Y finalmente los tres cables que están abajo de la imagen. Cable Azul, lo conectaremos al Pin Analógico que destinamos para tomar datos del Potenciómetro (En este ejercicio el A2). Cable Negro, al GND de Arduino (este resulta ser el GND común a todo el dispositivo), Cable Rojo debe ser conectado a 5V de Arduino.



En nuestros primeros ejemplos, para alimentar el o los motores, conectados a nuestro Drives, aplicaremos lo explicado en el punto 1- A, referente a la forma de conectar el DRIVER L298N.

Usaremos 4 pilas comunes AA (1,5V c/u), es decir 6 voltios como fuente de alimentación para los motores y parte lógica del L298N (Nuestro Driver) - Por lo tanto, **el jumper (regulador punto 8) se encuentra ACTIVO (debe estar conectado)**.

5) Controlar el Avance, Retroceso, Izquierda y Derecha de dos Motores DC de 6v simultáneamente, por medio del Monitor del Puerto Serie – Usamos controlador L298N y Librería "LEANTEC_ControlMotor".



Este programa nos permite controlar los movimientos adelante y atrás de dos motores DC de 6V.

Tenemos activa la Capacidad de Controlar la velocidad sin embargo mantenemos una velocidad fija de 150 (recordar que la velocidad puede oscilar entre 0 y 255).

Los comandos (Mayúscula o Minúscula) que reconocerá el programa, y se deben enviar desde el monitor del puerto serie, son los siguientes:

| | |
|---|-----------------------|
| A | Adelante |
| R | Retroceder |
| D | Doblar a la Derecha |
| I | Doblar a la Izquierda |
| S | Alto |

(Programa "010_Motores_DC_Driver_L298N_05_Dos_Motores_Desde_la_PC_Libreria")

| | |
|----|---|
| 1 | #include <LEANTEC_ControlMotor.h> //Incluimos la librería control de motores |
| - | |
| 2 | int PinIN1 = 3, //El pin 3 de Arduino se conecta con el pin In1 del L298N |
| 3 | PinIN2 = 5, //El pin 5 de Arduino se conecta con el pin In2 del L298N |
| 4 | PWM_Izquierdo_ENA = A0; //Pin Analógico conecta con ENB del L298N (Velocidad) |
| 5 | int PinIN3 = 7, //Pin Digital que conecta Arduino con el pin In3 del L298N |
| 6 | PinIN4 = 9, //Pin Digital que conecta Arduino con el pin In4 del L298N |
| 7 | PWM_Derecho_ENB = A1; //Pin Analógico conecta con ENA del L298N (Velocidad) |
| - | |
| 8 | int Sentido = 150, // Valores Positivos Adelantes - Negativos Atrás - 0 (Cero) detiene |
| 9 | Se_A = 0, |
| 10 | La_A = 0, |
| 11 | Lateral = 1; |
| - | |
| 12 | char dato; |
| - | |
| 13 | // Declaramos variable y Configuramos los pines que vamos a usar |
| 14 | ControlMotor control(PinIN1, PinIN2, PinIN3, PinIN4, PWM_Izquierdo_ENA, PWM_Derecho_ENB); |
| - | |

```

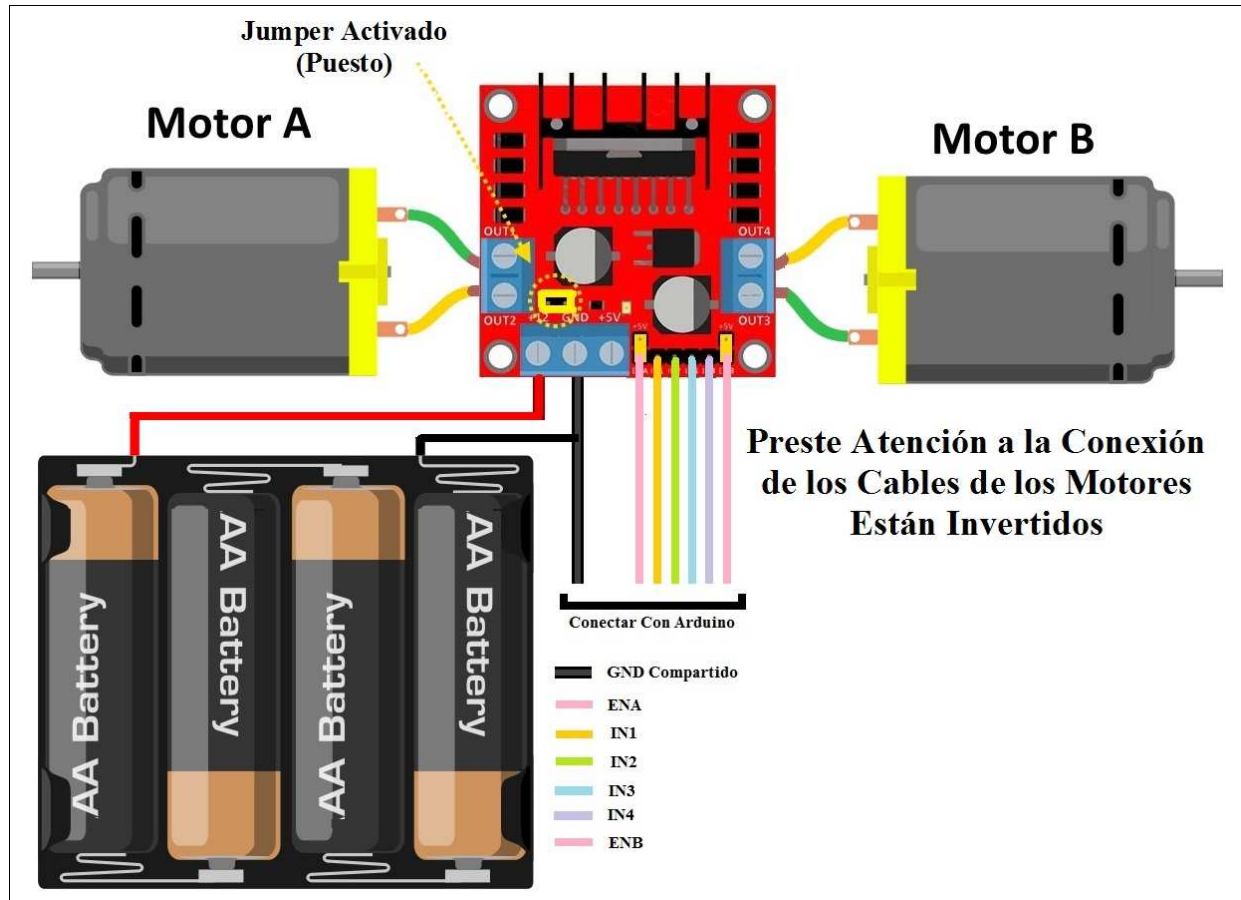
15 void setup() {
16   Serial.begin(9600);
17   while (!Serial) {
18     ; // Esperamos que el puerto serie este abierto.
19   }
20   // No necesitamos configurar pines, la librería lo hace por nosotros.
21 }
-
22 void loop() {
23   if (Serial.available() > 0) {
24     dato=Serial.read(); //Se lee la variable enviada por la APP.
25     switch (dato){ //Seleccionamos el caso, dependiendo del carácter recibido.
26       case 'A':
27         case 'a': { // Adelante
28           Sentido = 150; // Positivos Adelantes - Negativos Atrás - 0 (Cero) detiene
29           Lateral = 1; // 1 Derecho - Mas de uno dobla a la derecha - Negativos dobla a la Izquierda
30         } break;
31       case 'R':
32         case 'r': { // Atrás
33           Sentido = -150; // Positivos Adelantes - Negativos Atrás - 0 (Cero) detiene
34           Lateral = 1; // 1 Derecho - Mas de uno dobla a la derecha - Negativos dobla a la Izquierda
35         } break;
36       case 'd':
37         case 'D': { // Doblar a la Derecha
38           Sentido = 150; // Positivos Adelantes - Negativos Atrás - 0 (Cero) detiene
39           Lateral = 100; // 1 Derecho - Mas de uno dobla a la derecha - Negativos dobla a la Izquierda
40         } break;
41       case 'i':
42         case 'I': { // Doblar a la Izquierda
43           Sentido = 150; // Positivos Adelantes - Negativos Atrás - 0 (Cero) detiene
44           Lateral = -100; // 1 Derecho - Mas de uno dobla a la derecha - Negativos dobla a la Izquierda
45         } break;
46       case 's':
47         case 'S': { // Detener Motores
48           Sentido = 0; // Positivos Adelantes - Negativos Atrás - 0 (Cero) detiene
49           Lateral = 1; // 1 Derecho - Mas de uno dobla a la derecha - Negativos dobla a la Izquierda
50         } break;
51       default: {
52         int d = dato;
53         Serial.print("Comando Incorrecto: ");
54         Serial.print(d);
55         Serial.print(" * ");
56         Serial.println(dato);
57       } break;
58     } // Finaliza switch
59   } // Finaliza if - Donde se controla si se ha leído algo
60
61   if( Se_A != Sentido || La_A != Lateral){ // Entra y ejecuta Acción, si la Acción es diferente de la Anterior
62     control.Motor( Sentido, Lateral );
63     Se_A = Sentido;
64     La_A = Lateral,
65     Serial.print("Dirección: ");
66     Serial.print(Sentido);
67     Serial.print(" * ");
68     Serial.println(Lateral);

```

| | |
|----|--------------------------------|
| 69 | delay(50); |
| 70 | } |
| 71 | } //Finaliza la Función loop() |

CIRCUITO PARA NUESTRO PROYECTO

Lista de Materiales: 1 Porta Pilas para 4 pilas comunes AA (1,5V) - 2 Motores DC 6v con Caja reductora y Rueda de Goma - 1 Driver L298N - Cables de conexión (dependiendo del modelo de Arduino serán Hembra/Hembra o Macho/Hembra) - Placa Arduino y 1 Cable USB.



Si el o los motores no giran en el sentido que esperaba, solo invierta los cables en el conector del motor A o del motor B según sea el caso - Es decir cambie la polaridad, y funcionará al revés.

Los cables deberán ser conectados: De Izquierda a Derecha, Cable Negro al GND común de Arduino. Cable Rosado Al Pin Analógico que se configure en el Programa ENA (en este ejemplo A0). Cable Amarillo, al Pin Digital que se haya configurado en el programa a IN1 (en este caso 3). Cable Verde, al Pin Digital que se haya configurado en el programa a IN2 (en este caso 5). Cable Celeste, al Pin Digital que se haya configurado en el programa a IN3 (en este caso 7). Cable Morado, al Pin Digital que se haya configurado en el programa a IN4 (en este caso 9). Cable Rosado (Ultimo del grupo) Al Pin Analógico que se configure en el Programa ENB (en este ejemplo A1).

6) Controlar el Avance, Retroceso, Izquierda y Derecha de dos Motores DC de 6v simultáneamente, con instrucciones Pre Grabadas en un Archivo – Usamos controlador L298N y Librería "LEANTEC_ControlMotor".



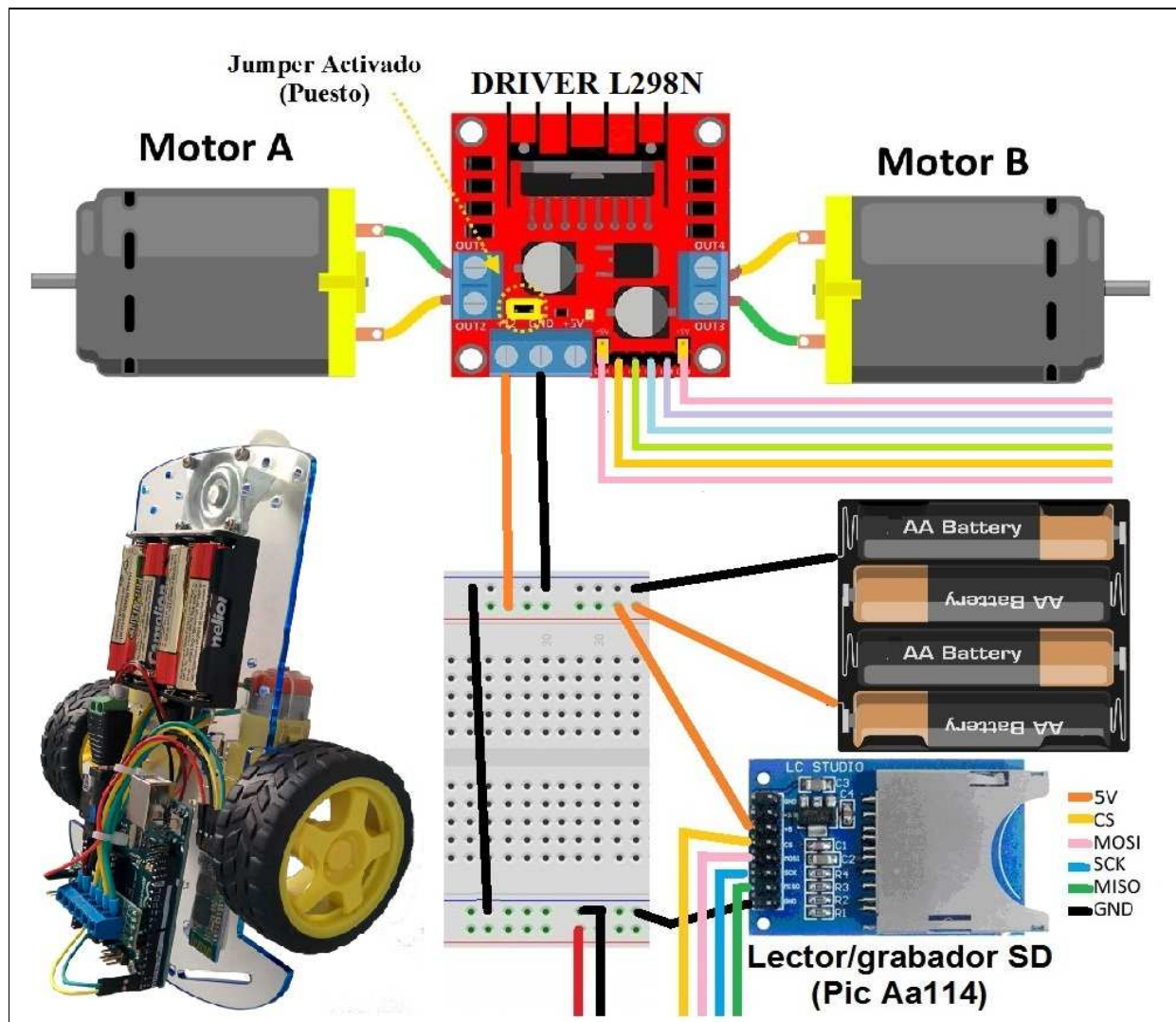
Este programa nos permite controlar los movimientos adelante y atrás de dos motores DC de 6V. Tenemos activa la Capacidad de Controlar la velocidad (recordar que la velocidad puede oscilar entre 0 y 255).

(Programa “”)

| | |
|---|--------------------------------|
| 1 | PRÓXIMAMENTE el Código. |
|---|--------------------------------|

CIRCUITO PARA NUESTRO PROYECTO

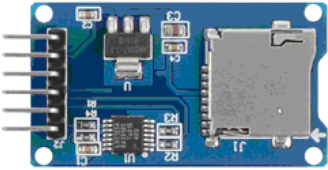
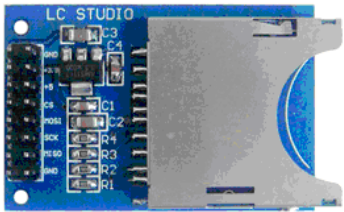
Lista de Materiales: 1 Porta Pilas para 4 pilas comunes AA (1,5V) - 2 Motores DC 6v con Caja reductora y Rueda de Goma - 1 Driver **L298N** - Módulo Lector/grabador Sd Card - Pic Aa114 - Cables de conexión (dependiendo del modelo de Arduino serán Hembra/Hembra o Macho/Hembra) - 1 Protoboard - Placa Arduino y 1 Cable USB.



Si el o los motores no giran en el sentido que esperaba, solo invierta los cables en el conector del motor A o del motor B según sea el caso - Es decir cambie la polaridad, y funcionará al revés.

Los cables deberán ser conectados: Primero veremos los cables de la derecha y los analizaremos en ordenadamente desde arriba hacia abajo. Comenzamos por el Cable Rosado, que conectaremos al Pin Analógico que se configure en el Programa ENB (en este ejemplo A1). Cable Morado, al Pin Digital que se haya configurado en el programa a IN4 (en este caso 9). Cable Celeste, al Pin Digital que se haya configurado en el programa a IN3 (en este caso 7). Cable Verde, al Pin Digital que se haya configurado en el programa a IN2 (en este caso 5). Cable Amarillo, al Pin Digital que se haya configurado en el programa a IN1 (en este caso 3). Cable Rosado (Ultimo del grupo, cerca de la Protoboard) deberemos conectarlo al Pin Analógico que se configure en el Programa ENA (en este ejemplo A0).

A continuación los cables que visualizamos en la parte inferior de la imagen, de izquierda a derecha, Cable Rojo a 5V de Arduino (En este Trabajo no lo usamos) y Cable Negro a GND de Arduino, luego los siguientes cuatro cables los usaremos para controlar el dispositivo Lector/grabador y también deberán ser conectado a la placa Arduino, según corresponda al Modelo de placa Arduino que usted Tenga. Por lo tanto, en estos casos, deberás releer un poco, donde se explica la conexión (del Lector-Grabador) con cada modelo de placa Arduino. O directamente repasar el PinOut correspondiente en nuestro Sitio. Ver Imagen de conexión.

| | |
|--|---|
| <p> GND — GND 5V — Vcc MISO — MISO MOSI — MOSI SCK — SCK D9 — CS </p>  <p>Conexión Micro SD con Modelos Arduino</p> | <p> 5V — 5V D9 — CS MOSI — MOSI SCK — SCK MISO — MISO GND — GND </p>  <p>Conexión Tarjeta SD con Modelos Arduino</p> |
| <p>IMPORTANTE: “CS” representa a cualquier PIN digital que podamos usar, en la imagen hace referencia al PIN Digital 9, aunque en el programa usemos otro PIN digital para trabajar.</p> | |

Apéndice A - Puente H

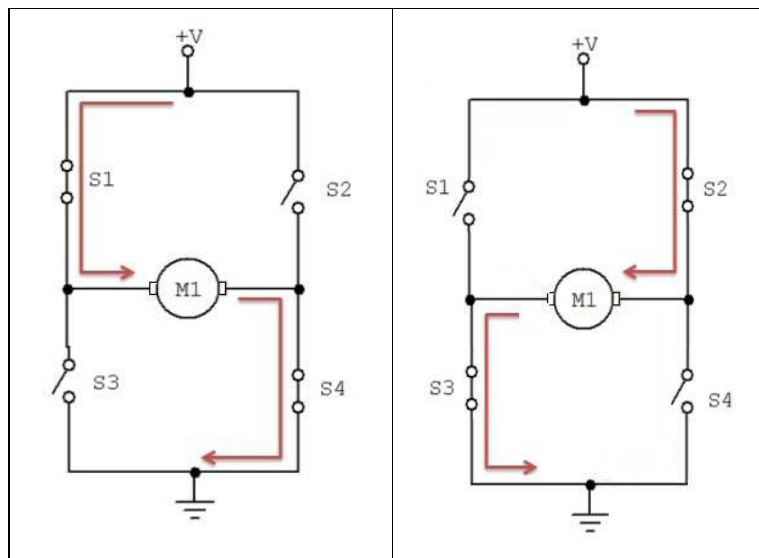
Un Puente en H es un circuito electrónico que generalmente se usa para controlar y permitir a un motor eléctrico DC girar en ambos sentidos (avance y retroceso).

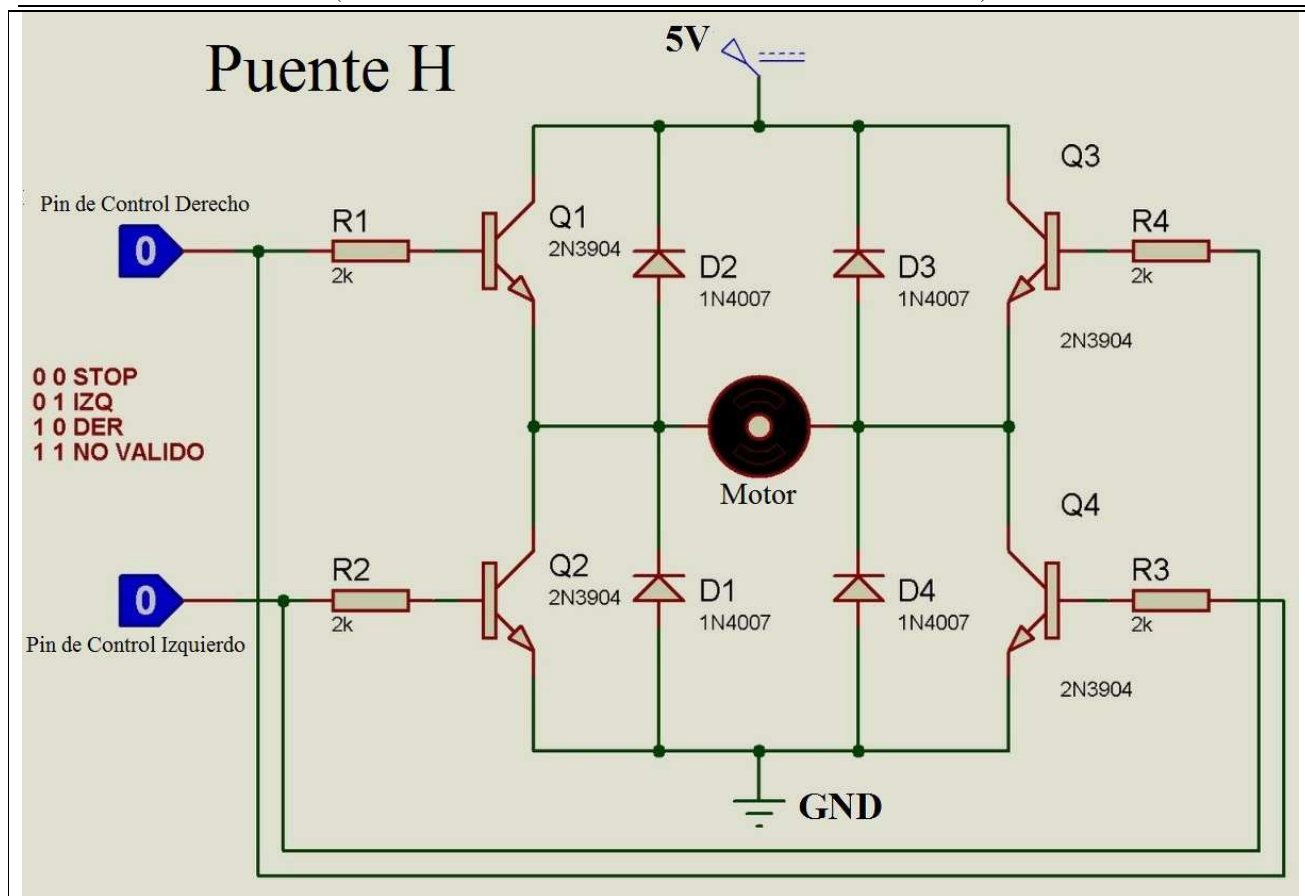
Los puentes H están disponibles como circuitos integrados, pero también pueden construirse a partir de componentes discretos, y esta compuesto principalmente por 4 interruptores, los cuales se accionan de dos en dos.

En la primera imagen activamos los interruptores 1 y 4, así el positivo le llega al motor por el borne izquierdo y el motor gira en sentido de las agujas del reloj.

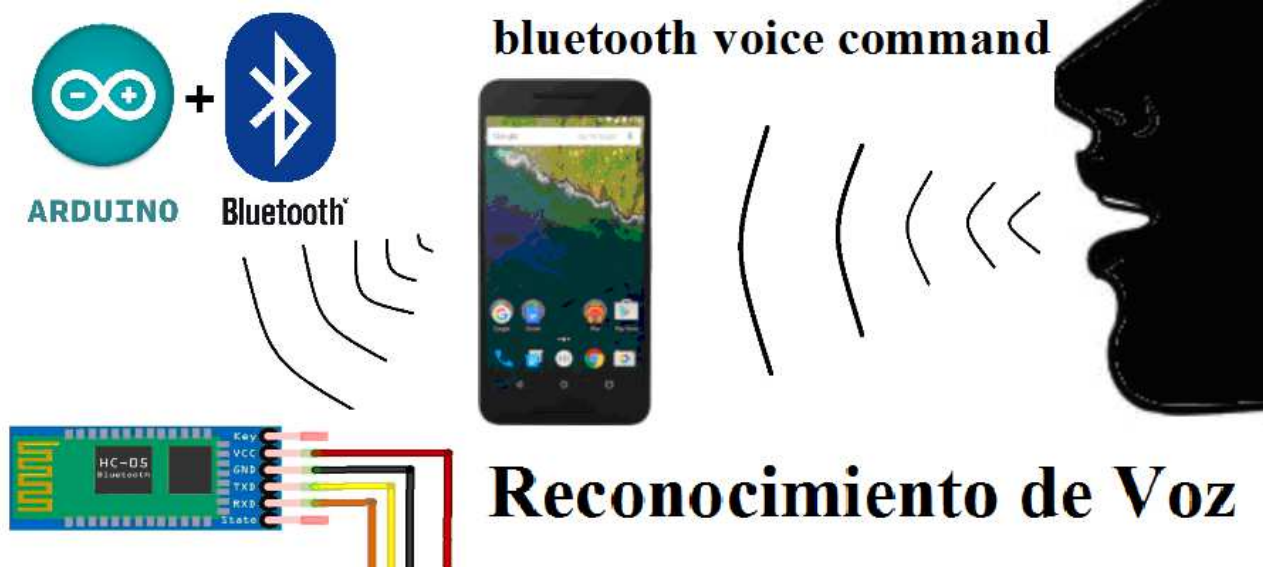
En la segunda imagen, desactivamos los interruptores 1 y 4 y activamos los interruptores 2 y 3, de esta forma el positivo le llega al motor por el borne derecho y el motor gira en sentido antihorario.

Estos interruptores se pueden sustituir por componentes electrónicos realicen la misma tarea, como por ejemplo, transistores y relés. Y para los que realmente les gusta, a continuación dejo el circuito de un Puente H. Es barato y muy simple de construir. Tienen el código (nombre) de cada componente. A divertirse!





Imagina ..!



FIN



| |
|--|
| 01001010 01010101 01010011 01010100 01001001 01000110 01001001 01000011 01000001 00100000 01010100 01010101 01010011 00100000 01001100 01001001 01001101 01001001 01010100 01000001 01000011 01001001 01001111 01001110 01000101 01010011 00101100 00100000 01011001 00100000 01000011 01001001 01000101 01010010 01010100 01000001 01001101 01000101 01001110 01010100 01000101 00100000 01001100 01000001 01010011 00100000 01010100 01000101 01001110 01000100 01010010 11000011 10000001 01010011 00101110 00100000 01000110 01110010 01100001 01100111 01101101 01100101 01101110 01110100 01101111 00100000 01110100 01101111 01101101 01100001 01100100 01101111 00100000 01100100 01100101 01101100 00100000 01101100 01101001 01100010 01110010 01101111 00100000 11100010 10000000 10011100 01001001 01101100 01110101 01110011 01101001 01101111 01101110 01100101 01110011 11100010 10000000 10011101 00100000 01100100 01100101 00100000 01010010 01101001 01100011 01101000 01100001 01110010 01100100 00100000 01000010 01100001 01100011 01101000 00101110 |
|--|

Si tienes algunas Correcciones y/o Sugerencias, por favor contáctame.