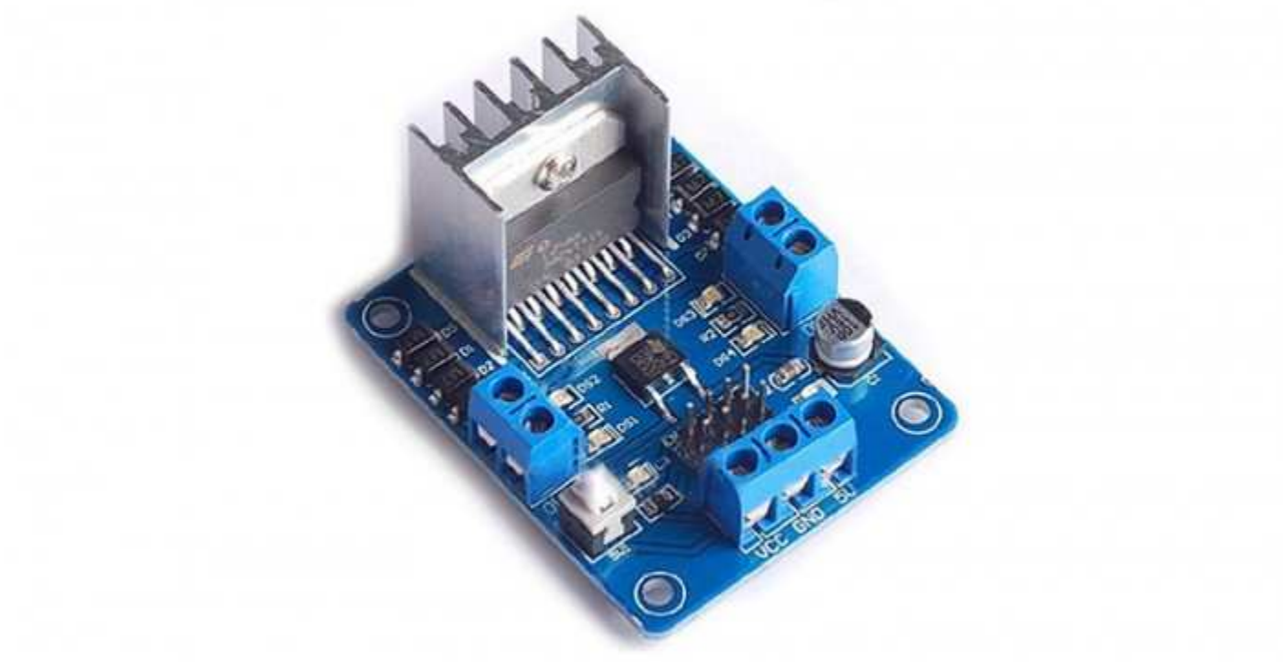
Puente H L298 Modulo

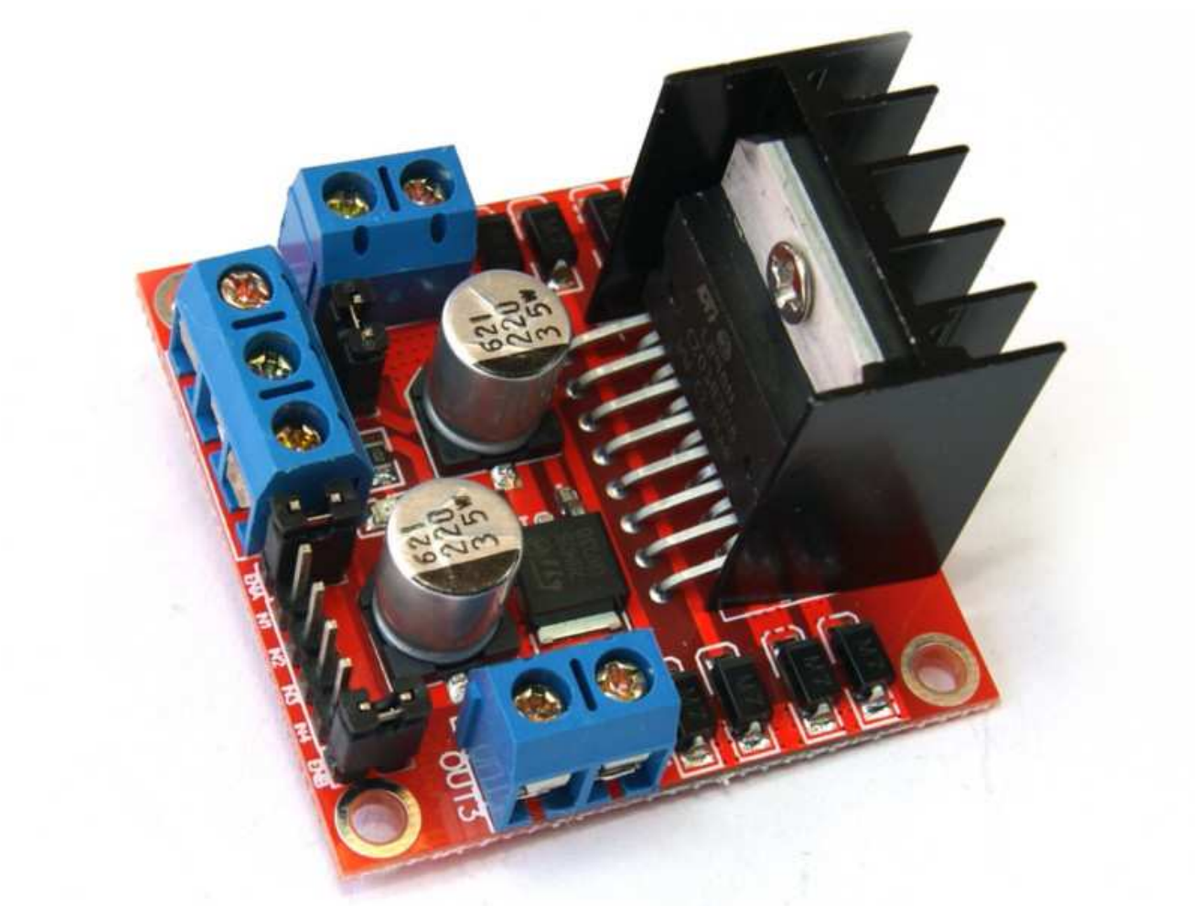
by Martin | Modulos, Motores, Rovers | 0 Comentarios



Especificaciones

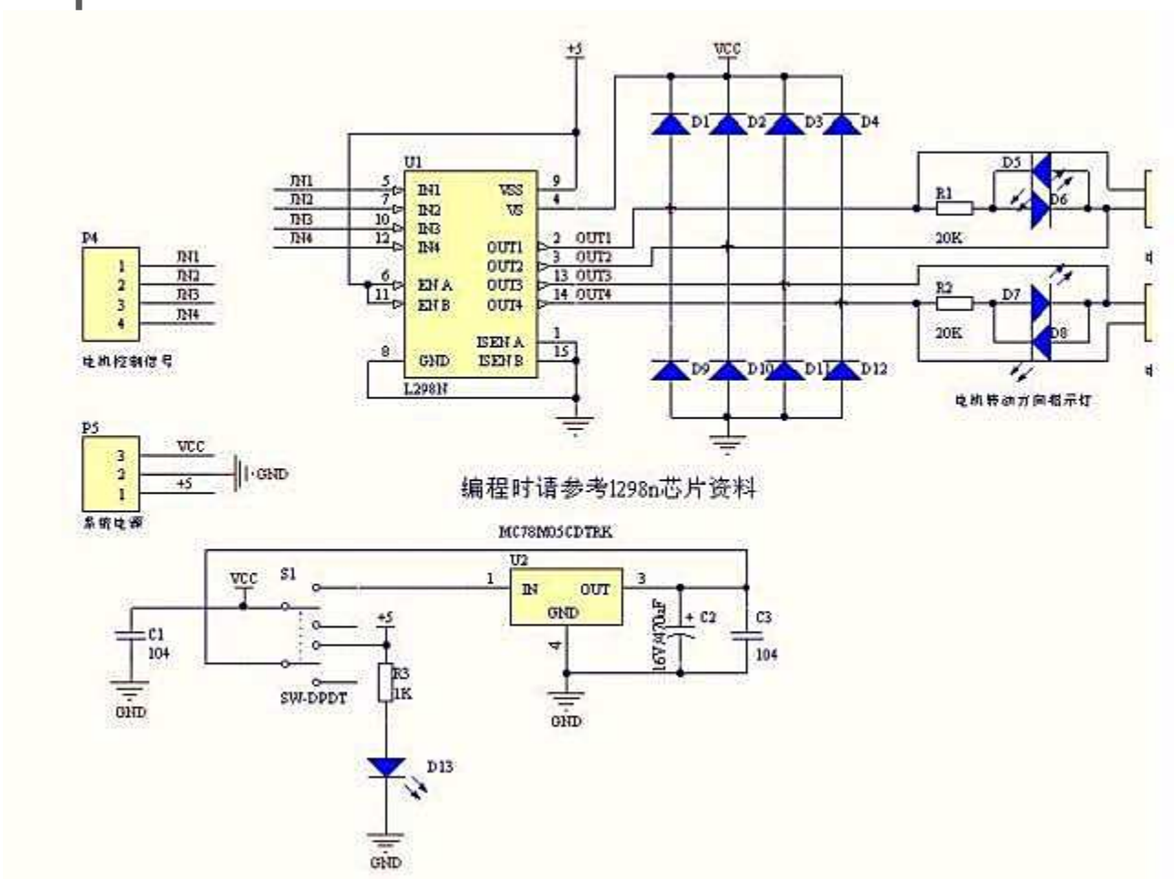
- Controlador: L298N Dual H Bridge DC Motor Driver IC
- Fuente de alimentación del motor Vs: +5 V a +35 V
- Corriente media máxima 2A
- pico de corriente Io: 3A
- Nivel de potencia lógico Vss: +5 V ~ +7 V (el regulador de 5V incorporado se puede usar si la potencia del motor es> 7.0V)
- Nivel lógico de potencia: 0 ~ 36mA
- Rango de entrada de la señal lógica: [Bajo: -0.3 V ≤ Vin ≤ 1.5V] [Alto: 2.3V ≤ Vin ≤ Vss]
- Disipación máxima de potencia: 20W (cuando la temperatura T = 75)

- Tamaño de la placa del conductor: 55 mm * 49 mm * 33 mm (con pilar de cobre fijo y la altura del disipador de calor)
- Peso de la placa del conductor: 33 g
- Indicadores de dirección LED
- Interruptor de fuente de alimentación lógica. (ABAJO para el regulador a bordo, ARRIBA para 5V Vcc en ambos motores y lógica)



Se pueden conectar 2 motores de CC a las salidas de la derecha. Arduino controla las entradas para que los terminales del motor estén conectados a tierra o Vcc (fuente de alimentación del motor). Por lo tanto, se puede aplicar voltaje en cualquier dirección a través de los motores, o ambos terminales pueden conectarse a tierra para detenerse.
Las entradas de “Habilitación” EN A y EN B se pueden conectar a un pin “PWM” de Arduino para controlar la velocidad del motor.
También es posible usar las 4 salidas a la vez para operar un solo motor paso a paso.
Tenga en cuenta las medidas tomadas aquí para controlar "Back EMF" y "fuente de alimentación" como se menciona en el tutorial (motores DC). Hay 8 diodos que están conectados desde las salidas del motor a tierra y la fuente de alimentación del motor. Estos son para limitar o "rechazar" los voltajes de EMF posteriores que de lo contrario dañarían el chip. Los diodos en el lado derecho son LED que se encienden cuando los motores están encendidos e indican en qué dirección está fluyendo la corriente. También hay un chip regulador de +5 voltios (MC78M05) que suministra +5 voltios al chip
L298N. Viendo el esquema del circuito es muy sencillo, lo gestionamos con las entradas PWD, tenemos las entradas IN1, IN2, IN3, IN4 y la manera de controlar es por parejas activando dos entradas y dejando dos a 0, por lo que gestionaremos la dirección de los motores.

Esquematico



Consejos

Los diodos D1 a D12, son diodos que están pensados para proteger al chip L298 de las “patadas” que tiran las bobinas cuando se desconectan, o lo que es igual, cuando pasan de UNO a CERO.

Los diodos Led D5 a D8, son leds que indican que canal de salida esta energizado. Básicamente indica el sentido de movimiento del motor, o el estado de la tabla de movimientos de un motor paso a paso.

El único botón, sirve para poder energizar la SALIDA de 5v del modulo. Esta salida existe porque los motores muchas veces se alimentan con tensiones superiores a 5v, con lo cual uno compra una fuente de tensión y potencia necesarias para mover el motor y ademas tendríamos que comprar otra de 5v para el arduino, el modulo L298 posee un regulador de 5v para alimentar el arduino casualmente, sin la necesidad de otra fuente.

Obviamente el diodo led D13 nos indica si los 5v estan encendidos y el arduino alimentado.

Otro detalle, Si bien la tensión máxima es de 35v y la corriente máxima nominal es de 2A, la potencia máxima de este chip es de 20w a 25w. En un caso hipotético que conectemos un motor de 35v y 2A, nos da una potencia de 70w; El chip SE VA A QUEMAR. esos son valores máximos absolutos de tensión o corriente, para saber si un motor es compatible con este chip, es usar la ecuación:

$$V[volt] x I[amp] <= P[watt]$$

si la tensión de nuestro motor multiplicada por la corriente del mismo es menor a 20w, no se va a quemar.

Codigos

```
/ ********************************************** ********************************************** /
#define MOTOR1_CTL1 8 // I1
#define MOTOR1_CTL2 9 // I2
#define MOTOR1_PWM 11 // EA
#define MOTOR2_CTL1 6 // I3
#define MOTOR2_CTL2 7 // I4
#define MOTOR2_PWM 10 // EB
#define MOTOR_DIR_FORWARD 0
#define MOTOR_P.P
. )
{
// Configurar pines para motor 1
pinMode (MOTOR1_CTL1, SALIDA);
pinMode (MOTOR1_CTL2, SALIDA);
pinMode (MOTOR1_PWM, SALIDA);
//
Pernos de configuración para el motor 2 pinMode (MOTOR2_CTL1, SALIDA);
pinMode (MOTOR2_CTL2, SALIDA);
pinMode (MOTOR2_PWM, SALIDA);
}
void setSpeed (char motor_num, char motor_speed)
{
if (motor_num == 1)
{
analogWrite (MOTOR1_PWM, motor_speed);
}
else
{
analogWrite (MOTOR2_PWM, motor_speed);
}
}
void motorStart (char motor_num, dirección del byte)
{
char pin_ctl1;
char pin_ctl2;
if (motor_num == 1)
{
pin_ctl1 = MOTOR1_CTL1;
pin_ctl2 = MOTOR1_CTL2;
}
else
{
pin_ctl1 = MOTOR2_CTL1;
pin_ctl2 = MOTOR2_CTL2;
}
interruptor (dirección)
{
caso MOTOR_DIR_FORWARD:
{
digitalWrite (pin_ctl1, LOW);
digitalWrite (pin_ctl2, HIGH);
}
Romper;
caso MOTOR_DIR_BACKWARD:
{
digitalWrite (pin_ctl1, HIGH);
digitalWrite (pin_ctl2, LOW);
}
Romper;
}
}
void motorStop (char motor_num)
{
setSpeed (motor_num, 0);
if (motor_num == 1)
{
```

```
digitalWrite (MOTOR1_CTL1, HIGH);
digitalWrite (MOTOR1_CTL2, ALTO);
}
else
{
digitalWrite (MOTOR2_CTL1, HIGH);
digitalWrite (MOTOR2_CTL2, ALTO);
}
}
void loop ()
{
// Start motors!
motorStart (1, MOTOR_DIR_FORWARD);
setSpeed (1, 200);
motorStart (2, MOTOR_DIR_FORWARD);
setSpeed (2, 200);
retraso (2000);
motorStart (1, MOTOR_DIR_BACKWARD);
setSpeed (1, 200);
motorStart (2, MOTOR_DIR_BACKWARD);
setSpeed (2, 200);
retraso (2000);
motorStart (1, MOTOR_DIR_FORWARD);
setSpeed (1, 140);
motorStart (2, MOTOR_DIR_BACKWARD);
setSpeed (2, 140);
retraso (2000);
motorStart (1, MOTOR_DIR_BACKWARD);
setSpeed (1, 140);
motorStart (2, MOTOR_DIR_FORWARD);
setSpeed (2, 140);
retraso (2000);
}
```

Referencia

http://arduino-info.wikispaces.com/MotorDrivers