



En esta página

¿Qué es un A4988 o DRV8825?

Curso de Arduino en proyectos

Ir al índice

Entradas y salidas

Motores

- Servos
- Servo rotación continua
- Controlador PCA9685
- Controlador 32 servos
- Motor DC y L298N
- Motor DC y BTS7960
- Motor DC y TB6612FNG
- Motor paso a paso A4988 DRV8825
- Motor paso a paso 28BYJ-48
- Motor paso a paso y TMCxxx

Actuadores

Optico

Distancia

Inercial

Movimiento

Tactil / Gestos

Ambiental

Agua / Liquidos

Intensidad / Tension

Display

Comunicacion

Controles

Magnetico

Memoria

Tiempo

RFID / NFC

Biometrico

Inicio > Tutoriales Arduino > Curso de Arduino en proyectos

< Anterior

Siguiente >



Motores paso a paso con Arduino y driver A4988 o DRV8825

14 min | Intermedio

¿Qué es un A4988 o DRV8825?

El A4988 y el DRV8825 son controladores (drivers) que **simplifican el manejo de motores paso a paso** desde un autómata o procesador como Arduino.

Estos controladores nos permiten manejar los altos voltajes e intensidades que requieren estos motores, limitar la corriente que circula por el motor, y proporcionan las protecciones para evitar que la electrónica pueda resultar dañada.

Para su control **únicamente requieren dos salidas digitales**, una para indicar el sentido de giro y otra para comunicar que queremos que el motor avance un paso. Además permiten realizar microstepping, una técnica para conseguir precisiones superiores al paso nominal del motor.

El A4988 ha alcanzado una gran popularidad en sus últimos tiempos debido a su uso en proyectos como, por ejemplo, en impresoras 3D caseras. Por su parte **el DRV8825 es una versión mejorada del A4988** y, por tanto, tiene unas características ligeramente superiores.

En particular, el DRV8825 permite trabajar con tensiones superiores al A4988 (45V frente a 35V), e intensidades superiores (2.5A frente a 2A). Además, añade un nuevo modo de microstepping (1/32) que no está presente en el A4988.

Por lo demás, aparte de unas pequeñas diferencias, **ambos dispositivos son similares** en su montaje y uso. Incluso pueden ser compatibles entre sí, es decir, bajo ciertas condiciones podemos usar indistintamente uno u otro, e incluso una combinación de ambos de forma simultánea.

Modelo

A4988

DRV8825

- ↩ Sonido >
- ↩ Gps >

☰ Relacionado

- 🔗 Curso iniciación Arduino
- 🔗 Curso Arduino avanzado

Color	Verde o Rojo	Morado
Intensidad máxima	2A	2.5A
Tensión máxima	35V	45A
Microsteps	16	32
Rs típico	0.05, 0.1 o 0.2	0.1
Fórmulas	$I_{\text{max}} = V_{\text{ref}} / (8 * R_s)$	$I_{\text{max}} = V_{\text{ref}} / (5 * R_s)$
	$V_{\text{ref}} = I_{\text{max}} * 8 * R_s$	$V_{\text{ref}} = I_{\text{max}} * 5 * R_s$

Ambos controladores pueden alcanzar altas temperaturas durante su funcionamiento y es necesario disipar el calor para que el dispositivo no se dañe. Para intensidades superiores 1A en el A4988 y a 1.5A en el DRV8825 es necesario añadir un sistema de disipación de calor, e incluso ventilación forzada.

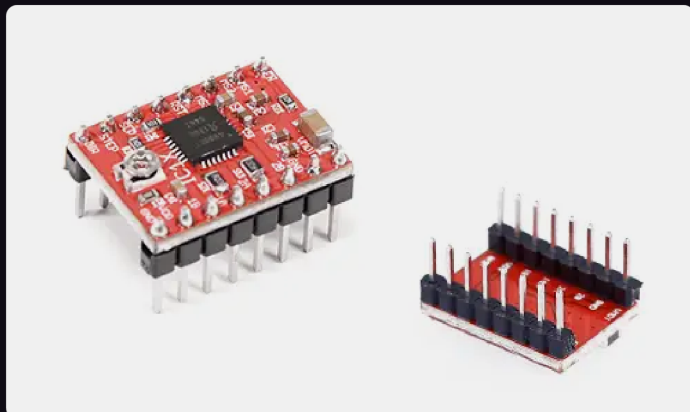
Disponen de protecciones contra sobreintensidad, cortocircuito, sobretensión y sobretemperatura. En general, **son dispositivos robustos y fiables** siempre que realicemos la conexión correctamente, e incorporemos disipación de calor si es necesario.

El A4988 y el DRV8825 son muy empleados en una gran variedad de proyectos que requieren el uso de motores paso a paso, como máquinas de CNC, plotters, robots que dibujan, impresoras 3D, y escáneres 3D.

También son un componente frecuente en proyectos para controlar robots y vehículos, especialmente en aquellos que requieren variar de forma individual la velocidad de cada rueda, como en vehículos con omniwheel o mecanum wheels.

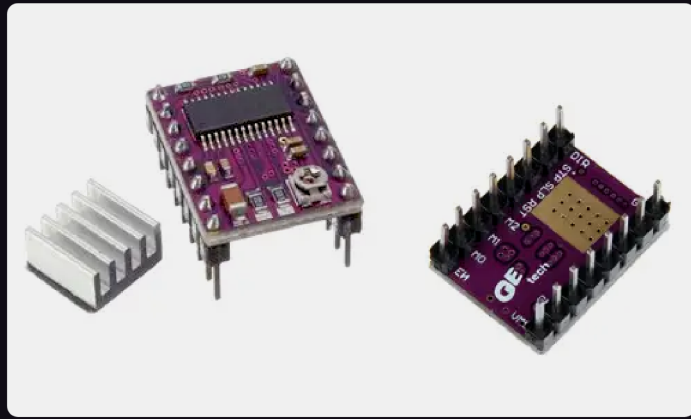
Precio

Actualmente ambos controladores son dispositivos baratos ya que han visto bajar su precio gracias al aumento de su popularidad. Es posible encontrar un driver A4988 por 0.75€, y un DRV8255 por 1.15€, en vendedores internacionales en eBay o AliExpress.



Con una diferencia de precio tan estrecha lo normal es que elijamos el DRV8255,

dado que tiene unas características técnicas superiores,

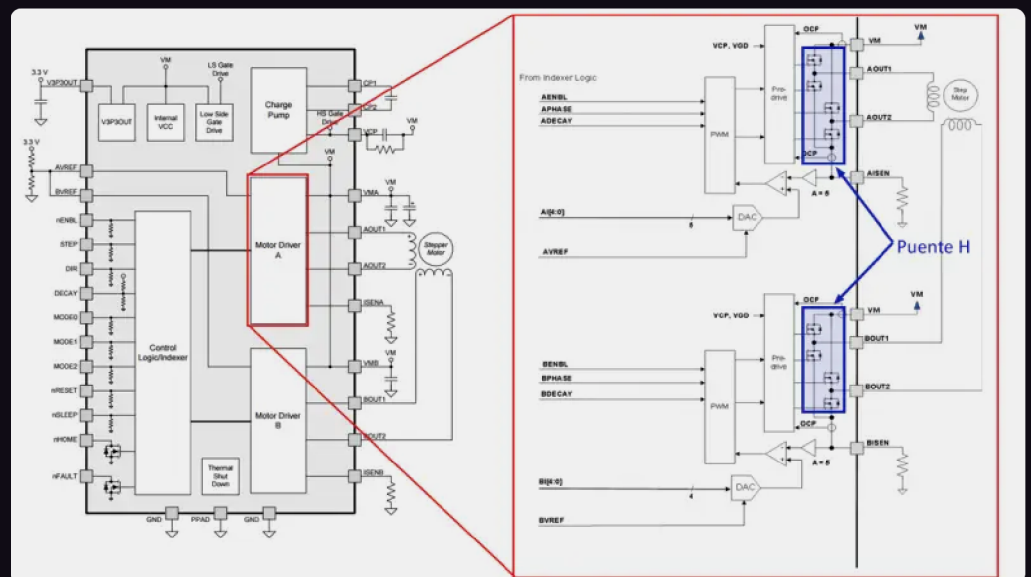


¿Cómo funciona el A4988 y el DRV8825?

Como en la mayoría de los controladores de motores el componente fundamental es un puente-H. En el caso del A4988 y DRV8825, destinados a controlar motores paso a paso, se dispone de dos puentes-H (uno por canal) constituidos por transistores MOSFET.

Sin embargo, a diferencia de controladores más simples como el L298N o el TB6612FNG, que presenta una electrónica relativamente simple, **el A4988 y el DRV8825 tienen una electrónica considerablemente más compleja.**

A modo de ejemplo, la siguiente imagen tenemos el esquema del DRV8255 y, cómo vemos, los puentes-H (remarcados en azul) representan una parte muy pequeña del conjunto.



Uno de los motivos para esta complejidad es que **únicamente requieren dos señales digitales de control** para hacer funcionar el motor, y que incorporan las protecciones necesarias para su manejo.

El otro motivo de la complejidad de su electrónica es que **incorporan funciones**

especialmente diseñadas para el control de motores paso a paso, como son el regulador de intensidad y el Microstepping. Veremos estas funcionalidades a continuación.

► Regulación de intensidad (Chopping)

Ambos controladores disponen de **reguladores de intensidad incorporados**. El motivo es que los motores paso a paso de cierto tamaño y potencia, como por ejemplo los NEMA 17 o NEMA 23, necesitan tensiones superiores a las que podrían soportar las bobinas por su corriente nominal.

Por ejemplo, supongamos que tenemos un motor NEMA 17 con 1.2A de intensidad nominal y 1.5 Ohm de resistencia por fase. Según la ley de Ohm deberíamos aplicar 1.8V a cada bobina para que circule la intensidad nominal de 1.2A. Sin embargo, **con esa tensión el motor ni se movería**.

Para el que el motor funcione correctamente **necesitamos aplicar una tensión superior**. En este ejemplo, la tensión nominal podría ser 12V. Pero si aplicáramos 12V directamente, nuevamente por la ley de Ohm, pasarían 8A por la bobina, lo cual destruiría el motor en poco tiempo.

Por este motivo, los controladores incorporan un limitador de intensidad, que permiten alimentar el motor a tensiones nominales superiores a las que es posible por su resistencia e intensidad máxima admisible.

Por supuesto, la ley de Ohm debe cumplirse en todo momento por lo que, continuando con nuestro ejemplo, cuando alimentemos el motor a 12V por la bobina pasará inevitablemente 8A.

El limitador **interrumpe la señal proporcionando una señal pulsada PWM** de forma que el valor promedio de la intensidad que atraviesa la bobina es la intensidad nominal del motor. Terminando nuestro ejemplo, el limitador de tensión aplicaría el pulso durante el 15% del tiempo y mantendrá el motor apagado el 85% restante.



A este mecanismo de limitación de intensidad se le denomina Chopping.

Para regular la intensidad que proporcionara el limitador y ajustarlo al valor del motor que vayamos a emplear **ambas placas disponen de un potenciómetro** que regula la intensidad del limitador.

Una forma de estimar la intensidad del regulador es **medir la tensión (Vref) entre el potenciómetro y GND** y aplicar una fórmula que depende del modelo, que encontraréis en la tabla del principio de la entrada.

Estas fórmulas **dependen el valor Rs de las resistencias ubicadas en la placa** que pueden variar en función del fabricante. Los valores típicos también aparecen en la tabla, pero debéis comprobar el valor de las resistencias que monta vuestra placa.

Por ejemplo, con los valores de Rs habituales las fórmulas se reducen a:

Modelo	Rs	Fórmula reducida
A4988	50	$I_{\max} = 0,625 * V_{\text{ref}}$
A4988	100	$I_{\max} = 1,25 * V_{\text{ref}}$
A4988	200	$I_{\max} = 2,2 * V_{\text{ref}}$
DRV8825	100	$I_{\max} = 2 * V_{\text{ref}}$

No obstante, **el valor obtenido mediante esta medición es sólo una aproximación** y puede ser inexacto, por lo que lo emplearemos sólo como una calibración inicial, y terminaremos el ajuste fino midiendo la corriente real que proporciona el controlador al motor mediante un amperímetro.

Microstepping

Como hemos dicho, el microstepping es una **técnica que permite obtener pasos inferiores al paso nominal** del motor paso a paso que vamos a controlar.

Cuando vimos las secuencias típicas de encendido de un paso a paso, vimos que aplicando un control todo a nada a las bobinas teníamos varias posibles combinaciones, de las cuales vimos las tres más habituales (1-fase, 2-fases, media-fase). Pero nadie ha dicho que tengamos que encender o apagar por completo las bobinas.

El microstepping hace **variar la corriente aplicada a cada bobina** emulando un valor analógico. Si pudiéramos a ambas bobinas dos señal eléctrica senoidal perfecta desfasadas 90° conseguiríamos un campo magnético rotatorio perfecto en el interior del motor.

Por supuesto el controlador digital no genera valores analógicos perfectos, si no valores discretizados ("a saltos"), por lo que la señal eléctrica que aplica es igualmente **una función senoidal discretizada**.

El resultado es un campo magnético un campo magnético rotativo con un paso inferior al paso nominal, que depende del número de niveles de discretos que podemos emplear en las señales de excitación de la bobina.

Cuando funcionamos sin microstepping (Modo full step), los controladores aplican una secuencia de 2-fases, por lo que aplican de forma permanente el 71% de la

corriente del limitador a cada bobina. Únicamente varían el sentido en el que la corriente circula por la bobina.

Sin embargo, si aplicamos microstepping en cualquier de sus modos de funcionamiento, **el controlador llega a aplicar el 100%** de la corriente a una de las bobinas en un determinado paso. La cantidad de corriente concreta aplicada a cada bobina varía con cada paso.

Por ejemplo, con resolución 1/4 de paso, en el primer paso de la secuencia tendremos una bobina al 100% y la bobina B al 0%, en el segundo paso la bobina A al 92% y la bobina B al 38%, en el tercer paso la bobina A al 71%, y así sucesivamente.



El hecho de que sin microstepping la corriente aplicada es siempre el 71%, y si aplicamos microstepping (en cualquier resolución) la corriente aplicada llegará a ser del 100% en algún paso, es **muy importante a la hora de calibrar la intensidad que circula por las bobinas.**

Finalmente, la resolución con la que queremos que funcione el controlador **se controla aplicando tensión a los Pines M0, M1 y M2**. Estos pines están puestos a tierra mediante resistencias de Pull-Up, por lo que si no conectamos nada estarán a Low, y sólo deberemos forzar los pines en High.

Resolucion		Pines M0, M1 y M2		
A4988	DRV8825	MODE0	MODE1	MODE2
Full step	Full step	Low	Low	Low
1/2 step	1/2 step	High	Low	Low
1/4 step	1/4 step	Low	High	Low
1/8 step	1/8 step	High	High	Low
-	1/16 step	Low	Low	High
-	1/32 step	High	Low	High
-	1/32 step	Low	High	High
1/16 step	1/32 step	High	High	High



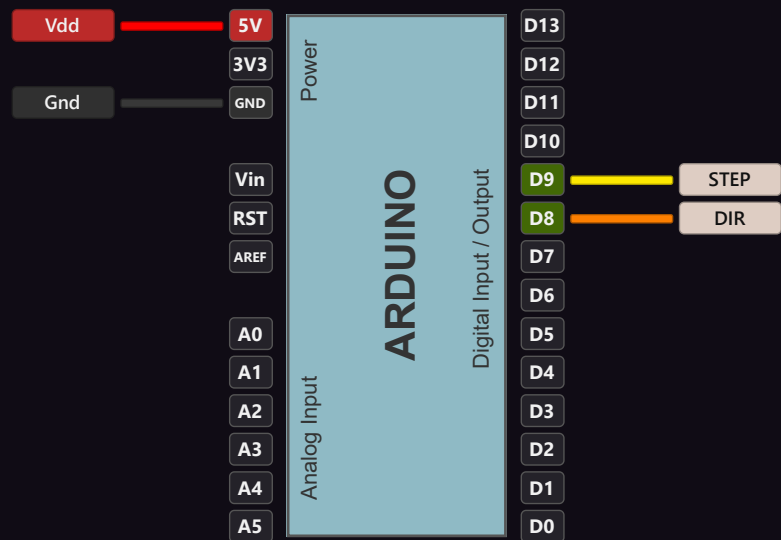
Dejar todos los pines desconectados dará lugar a usar **modo Full Step**

Esquema de montaje

El esquema de conexión de ambos controladores es muy similar. Incluso, como hemos dicho, bajo ciertas consideraciones ambos dispositivos son compatibles entre si.

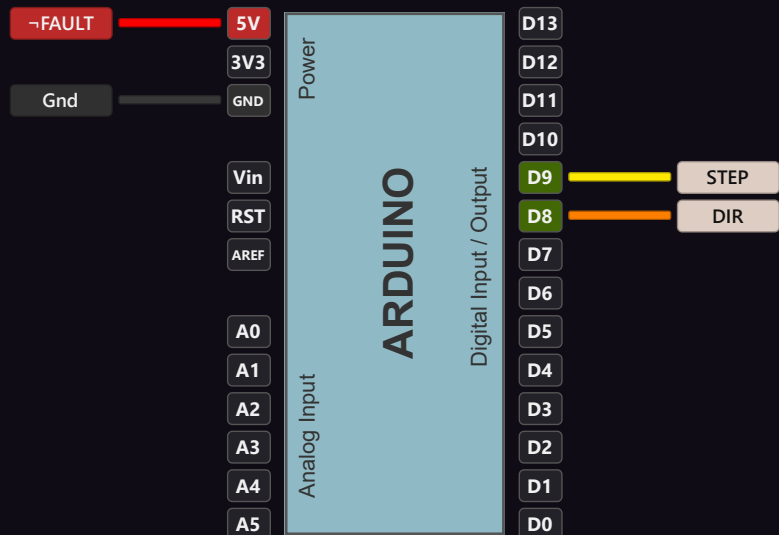
En el caso del A4988 el esquema es el siguiente

Que visto desde Arduino quedaría de la siguiente forma.



En el caso del DRV8825 el esquema es muy similar, con la excepción de que el pin Vdd se sustituye por el pin \rightarrow Fault.

El esquema desde Arduino sería idéntico al anterior.



Observar que **el potenciómetro está ubicado en extremos diferentes en ambos modelos**. Por tanto a la hora de cambiar uno por otro, porque es fácil equivocarse en la orientación, y dañar el dispositivo.



Comprobar la orientación del controlador durante el montaje y antes de dar la alimentación al sistema

Proceso de montaje

El proceso de montaje del A4988 o el DRV8825 no es demasiado complicado. La única parte que resulta un poco peliaguda es el proceso de ajuste y calibración del regulador de intensidad.

Ambos controladores son bastante **robustos y difíciles de romper cuando están trabajando**, si respetamos las necesidades de disipación de calor (cuando sea necesario).

Pero son **extremadamente sensibles a desconectar el motor cuando el controlador está alimentado**. Hacer esto causará, casi con total seguridad, que dañemos el controlador.



En general, evitar conectar o desconectar cualquier cable **mientras el controlador esté alimentado**.

Por otro lado, **el potenciómetro del regulador de intensidad no tiene tope físico**, por lo que es posible pasar de cero al máximo de valor sin danos cuenta, y podríamos dañar accidentalmente el motor.

Para no dañar ni el controlador ni el componente, debemos seguir siempre el proceso **rigurosamente sin saltarnos ningún paso**.

- ▶ Conectar el driver a tensión, sin el motor y sin microstepping
- ▶ Medir con un voltímetro la tensión entre GND y el potenciómetro
- ▶ Ajustar el potenciómetro hasta que la tensión sea el valor proporcionado por la fórmula
- ▶ Apagar el montaje
- ▶ Conectar el motor, interponiendo en medio un amperímetro
- ▶ Realizar con cuidado el ajuste fino del potenciómetro, hasta que la intensidad sea la nominal del motor
- ▶ Apagar el montaje
- ▶ Retirar el amperímetro, y conectar el motor definitivamente
- ▶ Conectar Arduino al montaje

Como hemos dicho, el hecho de que vayamos a usar posteriormente microstepping influye en el valor límite que debemos fijar en el regulador de intensidad.

Si no vamos a usar Microstepping, podemos aplicar aumentar el límite del regulador de intensidad hasta el **100% de la intensidad nominal del motor**.

Si vamos a usar Microstepping, el valor que hemos medido es el 71% del que luego circulará realmente por la bobina. Por tanto, en el amperímetro debemos ajustar el **71% de la intensidad nominal del motor**.

Ejemplo de código

Aunque entender los pormenores del A4988 y el DRV8825 puede haber tenido su dificultad, las ventajas de su uso claras. El código necesario para su control **es extremadamente sencillo**, lo cual hace de ellos unos componentes muy prácticos y útiles de usar.

Simplemente tenemos que indicar mediante dos salidas digitales el instante en el que queremos que el motor avance un paso, y el sentido de giro. La velocidad de giro se controla por el tiempo que dejamos transcurrir entre paso y paso.

El siguiente ejemplo, hace girar el motor paso a paso una revolución en un sentido, y dos en sentido contrario a una velocidad ligeramente superior.

```
1  const int dirPin = 8;
2  const int stepPin = 9;
3
4  const int steps = 200;
5  int stepDelay;
6
7  void setup() {
8    // Marcar los pines como salida
9    pinMode(dirPin, OUTPUT);
10   pinMode(stepPin, OUTPUT);
11 }
```



```
12
13 void loop() {
14     //Activar una direccion y fijar la velocidad con stepDelay
15     digitalWrite(dirPin, HIGH);
16     stepDelay = 250;
17     // Giramos 200 pulsos para hacer una vuelta completa
18     for (int x = 0; x < steps * 1; x++) {
19         digitalWrite(stepPin, HIGH);
20         delayMicroseconds(stepDelay);
21         digitalWrite(stepPin, LOW);
22         delayMicroseconds(stepDelay);
23     }
24     delay(1000);
25
26     //Cambiamos la direccion y aumentamos la velocidad
27     digitalWrite(dirPin, LOW);
28     stepDelay = 150;
29     // Giramos 400 pulsos para hacer dos vueltas completas
30     for (int x = 0; x < steps * 2; x++) {
31         digitalWrite(stepPin, HIGH);
32         delayMicroseconds(stepDelay);
33         digitalWrite(stepPin, LOW);
34         delayMicroseconds(stepDelay);
35     }
36     delay(1000);
37 }
```

Descarga el código

Todo el código de esta entrada está disponible para su descarga en Github.

Temas:

Actuadores

Motores



Curso de Arduino en proyectos

[Ir al índice](#)

< Anterior

Controlar dos motores DC con
Arduino y driver TB6612FNG

Siguiente >

Motor paso a paso 28BYJ-48 con
Arduino y driver ULN2003



Luis Llamas
Ingeniería, informática y
diseño

Excepto notación expresa, los
contenidos de este sitio se ofrecen
bajo licencia Creative Commons
License BY-NC-SA.



Secciones

Ingeniería
Informática
Arduino
ESP32
Raspberry
Programación
Diseño 3D

Destacados

Cursos
Desde cero
Roadmaps
Formularios
CheatSheet
Quizzes
Calculadoras



DISCORD