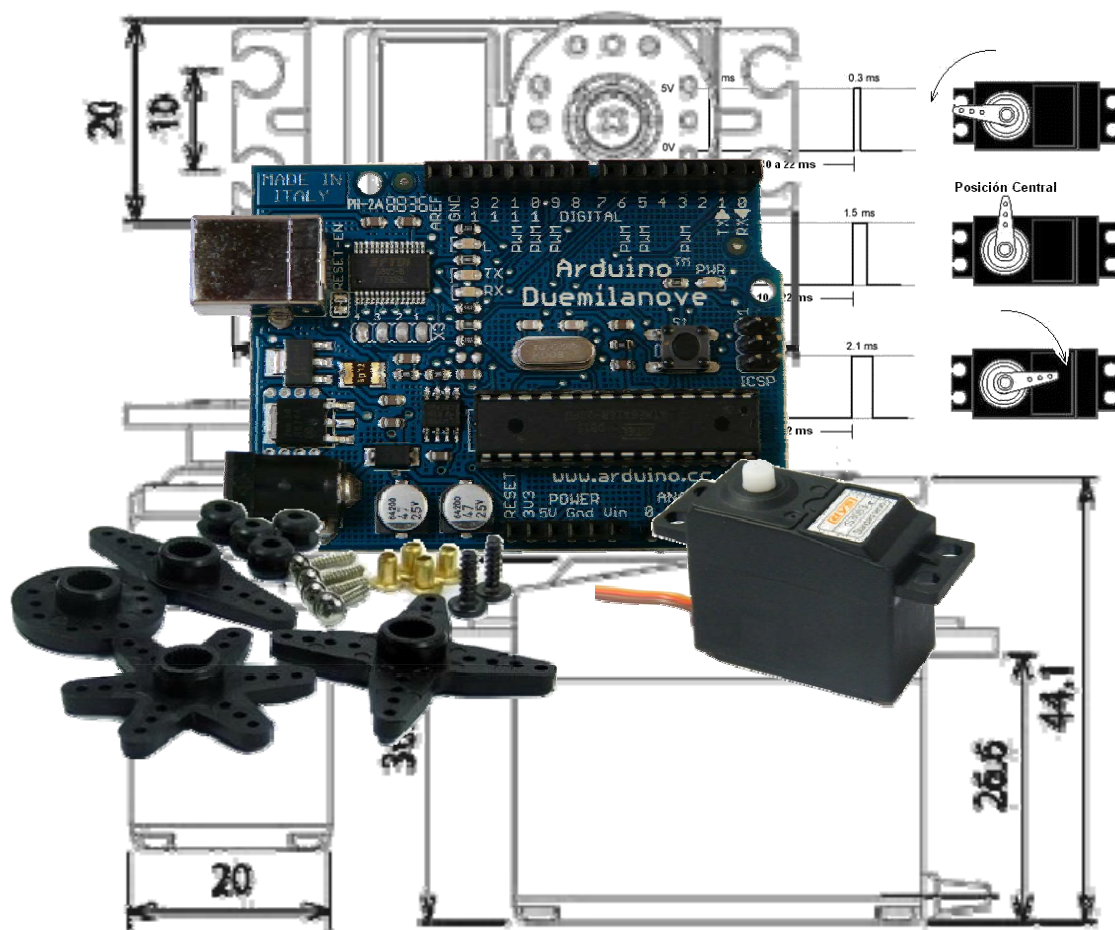


# Como utilizar un servo motor con Arduino.

Revisión Diciembre 2009



Desarrollada por:

Christopher Thompson

cthompson@olimex.cl

Revisada por:

Paul Aguayo

paguayo@olimex.cl

## 1 Introducción

Arduino es una plataforma de código abierto, basada en una sencilla placa con entradas y salidas analógicas y digitales. Posee un entorno de desarrollo basado en el lenguaje Processing/Wiring. Puede recibir señales de una variedad de sensores y afectar su entorno controlando luces, motores o actuadores, ya sea trabajando de manera autónoma o a través de un programa corriendo en un computador (por ejemplo, Macromedia Flash, Processing, Max/MSP, Pure Data, SuperCollider). Se compone de un micro controlador Atmel AVR que viene pre-programado con una secuencia de arranque (Boot Loader), por lo que no necesita un programador externo.

En esta guía daremos un ejemplo fácil de cómo conectar y controlar un Motor servo estándar de 360° y de 180°. Controlándolo a través de una entrada analógica, en este caso un potenciómetro.

Un motor servo es un dispositivo actuador que tiene la capacidad de ubicarse en cualquier posición dentro de su rango de operación, y de mantenerse estable en dicha posición. Está formado por un motor de corriente continua, una caja reductora y un circuito de control, y su margen de funcionamiento generalmente es de menos de una vuelta completa.

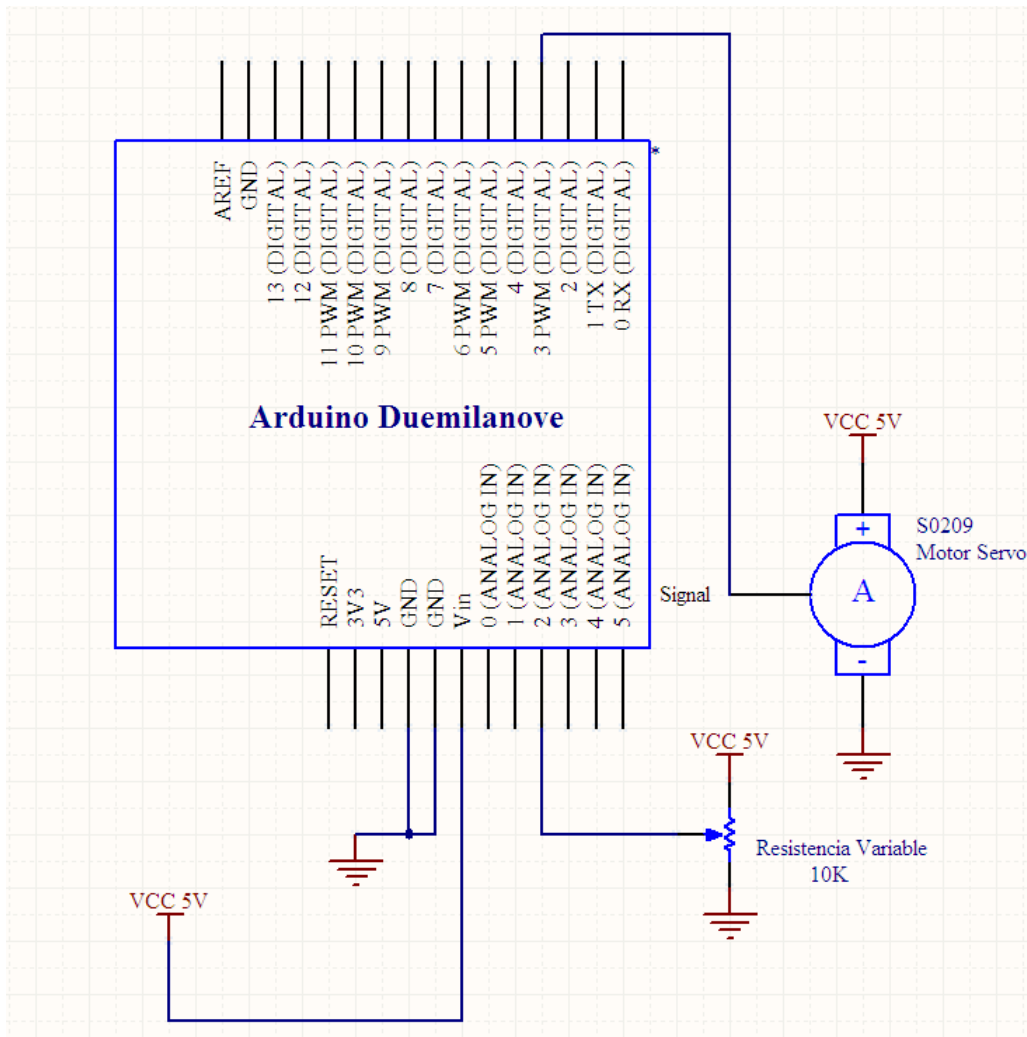
El punto de referencia o setpoint — que es el valor de posición deseada para el motor— se indica mediante una señal de control cuadrada. El ancho de pulso de la señal indica el ángulo de posición: una señal con pulsos más anchos (es decir, de mayor duración) ubicará al motor en un ángulo mayor, y viceversa. En el caso del motor servo 360° el cambio del ancho del pulso reducirá la velocidad o cambiara el sentido de dirección de este.

La tarjeta Arduino Duemilanove posee 6 salidas de PWM, Pulse-Width Modulation en inglés y Modulación de Ancho de Pulso en español, las cuales utilizaremos para controlar los motores.

## 2 Componentes a utilizar.

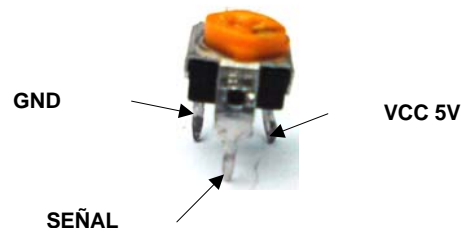
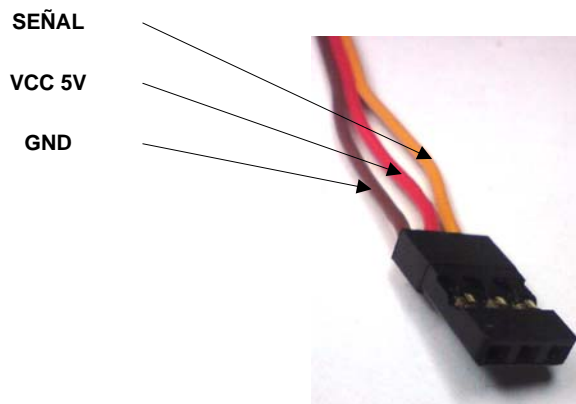
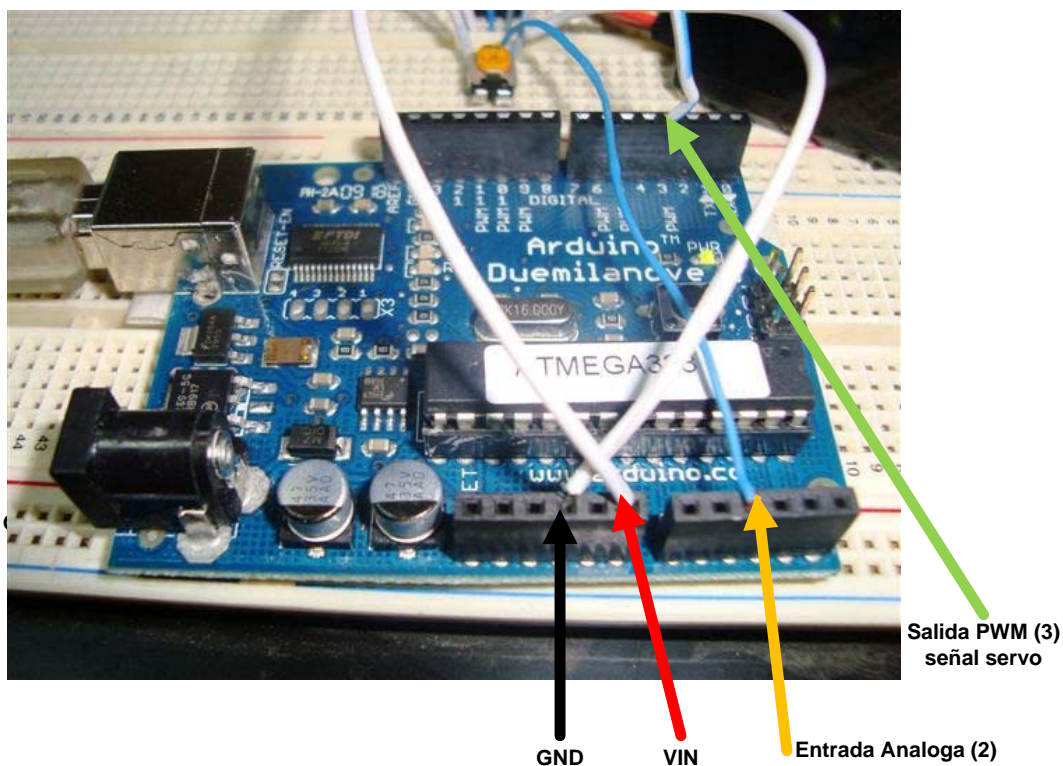
- Duemilanove ATmega328
- Servo motor estandar DYS3003 (180°)
- Servo Motor estandar DYS0209 (360°)
- Resistencia variable 10k
- Fuente de 5v DC 1000mA
- Proto Board

### 3 Esquema conexión



#### 3.1.1 Conexiones

- Conectamos la tarjeta Arduino al USB del PC o bien a una fuente de 9V.
- Conectamos el Servo Motor a una fuente de 5V externa. No se debe conectar el servo motor a la salida de 5V de la tarjeta Arduino debido a alto consumo. Los pines de salida de Arduino pueden entregar hasta 40 mA, sin embargo, los motores pueden tener peaks de alrededor de 700mA.
- Conectamos el cable de señal del servo motor a una de las salidas PWM de la tarjeta Arduino. En este ejemplo utilizaremos la salida 3 PWM del Arduino, el cable de señal del motor servo es el naranja.
- Conectamos el potenciómetro (resistencia variable) a una fuente de 5v y una de las entradas analógicas de la tarjeta Arduino (en este caso ocupamos la número Analog IN 2)



#### Cuadro de conexión:

	LUGAR DE CONEXIÓN			
ARDUINO	VIN	GND	ANALOG IN 2	PWM 3
SERVO MOTOR	CABLE ROJO	CABLE CAFÉ	-	SEÑAL SERVO
POTENCIOMETRO	PIN 1	PIN3	PIN2	-
FUENTE 5V	5v +	GND	-	-

La conexión del servo motor puede variar según el fabricante, a continuación algunas configuraciones para marcas conocidas.

Fabricante	Voltaje positivo	Tierra	Señal de control
Futaba	Rojo	Negro	Blanco
Dong Yang	Rojo	Marrón	Naranja
Hobico	Rojo	Negro	Amarillo
Hitec	Rojo	Negro	Amarillo
JR	Rojo	Marrón	Naranja
Airtronics	Rojo	Negro	Naranja
Fleet	Rojo	Negro	Blanco
Krafr	Rojo	Negro	Naranja
E-Sky	Rojo	Negro	Blanco

El color del cable de cada terminal varía con cada fabricante, aunque el cable del terminal positivo de alimentación siempre es rojo. El cable del terminal de alimentación negativo puede ser marrón o negro, y el del terminal de entrada de señal suele ser de color blanco, naranja o amarillo.

Para realizar el ejemplo nosotros ocuparemos el motor [DYS0209](#) y el [DYS3003](#) de 360° y 180° respectivamente. Ambos modelos disponibles en [www.olimex.cl](http://www.olimex.cl)

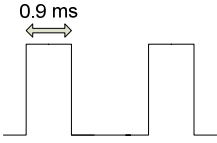

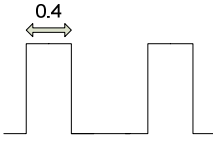

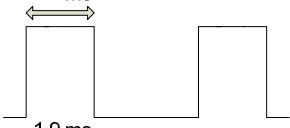

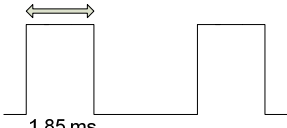

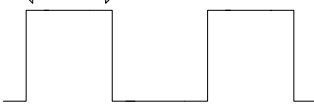

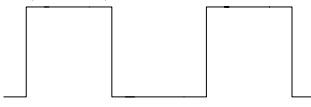

#### Especificaciones:

MODELO	VOLTAJE	VELOCIDAD	TORQUE	TAMAÑO	PESO	TIPO DE GIRO
DYS0209	4.8~6.0V	0.18sec/60°	3.5kg*cm	40.8x20.1x38.0 mm	38gr	360°
DYS3003	4.8~6.0V	0.18sec/60°	4,1kg*cm	40.8x20.1x38.0 mm	38gr	180

#### Accesorios:

DYS0209	DYS3003
	

## Ancho de pulso según pruebas en Laboratorio MCI

DYS0209	DYS3003
 	  0°
 	  90°
 	  180°

En el ejemplo que detallaremos, haremos funcionar estos dos tipos de motores usando la misma conexión detallada anteriormente.

Para hacer funcionar los motores ocuparemos una salida PWM (Modulación de Ancho de Pulso), y a través de una señal análoga (potenciometro), regularemos el ancho de pulso de la señal y con esto el giro del motor servo.

En el caso del motor de 360°, podemos regular su velocidad y sentido de giro, para ello utilizamos el siguiente código.

```

servo_360 $
/*Arduino con un motor servo estándar de 360°
Ingeniería MCI Ltda. - Luis Thayer Ojeda 0115 of 402, Providencia, Santiago, Chile
www.olimex.cl Fono: + 56 2 3339579 Fax: +56 2 3350589
*/
float potPin = 2; // ENTRADA DE SEÑAL DEL POTENCIOMETRO
int pwmPin = 3; // SALIDA SEÑAL SERVO MOTOR
int sensorValue;
float calibrate;

void setup () {
  pinMode (pwmPin, OUTPUT);
}

void loop() {
  sensorValue = analogRead(potPin);
  calibrate = map (sensorValue, 0 , 1024, 120, 240); /*ajustes de ancho
                                                    de pulso según
                                                    potenciometro*/

  analogWrite (pwmPin, calibrate); /*Ancho de pulso
                                    según la conversión
                                    de la entrada analoga
                                    a digital*/
}

```

Después de definir las variables, y configurar el PWM como salida en el bloque *setup*, leemos el valor de la entrada analógica y luego ocupamos la función "map" que en esencia nos permite convertir un rango de variación en otro.

En este caso conectamos el potenciómetro a una entrada analógica. Las entradas analógicas en Arduino son de 10 bits, por lo que entregan valores entre 0 y 1023 ( $2^{10} - 1 = 1023$ ). El rango de voltaje que está ingresando a la entrada analógica está dado por el potenciómetro y puede variar entre 0 y 5 volts. En consecuencia cuando tengamos 5V en la entrada analógica la función *analogRead* va a entregar un valor de 1023 y cuando tengamos 0V en la entrada la función entregará un valor de 0

Tomamos la señal obtenida con la función *analogRead* y con ella variamos el ancho de pulso de la salida PWM utilizando la función *analogWrite*. Esta función recibe como parámetro un número ente entre 0 y 255,

En este ejemplo, vamos a configurar la salida del PWM para que trabaje en el rango 120 a 240, para obtener solo el ancho de pulso que necesitamos según las especificaciones del motor de 360°, de esta forma acotamos el potenciómetro a los valores que necesitamos que se mueva el motor, para ello utilizamos la función *map*

```
void loop() {  
  sensorValue = analogRead(potPin);  
  calibrate = map (sensorValue, 0 , 1024, 120, 240);
```

Para el motor de 180° ocupamos el mismo código, pero variamos la salida del PWM de la siguiente forma.

```
void loop() {  
  sensorValue = analogRead(potPin);  
  calibrate = map (sensorValue, 0 , 1024, 50, 240);
```

De esta forma se obtienen los pulsos para cada ejemplo.

Los videos que muestran el funcionamiento de ambos servos se pueden ver en los siguientes links:

<http://www.youtube.com/watch?v=ZZbwNUL5MK0>

[http://www.youtube.com/watch?v=byq\\_-puHhAE](http://www.youtube.com/watch?v=byq_-puHhAE)