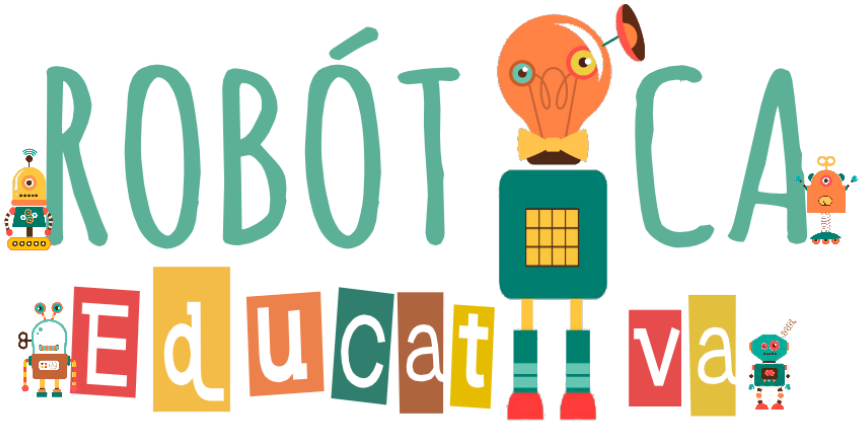
Robótica Educativa

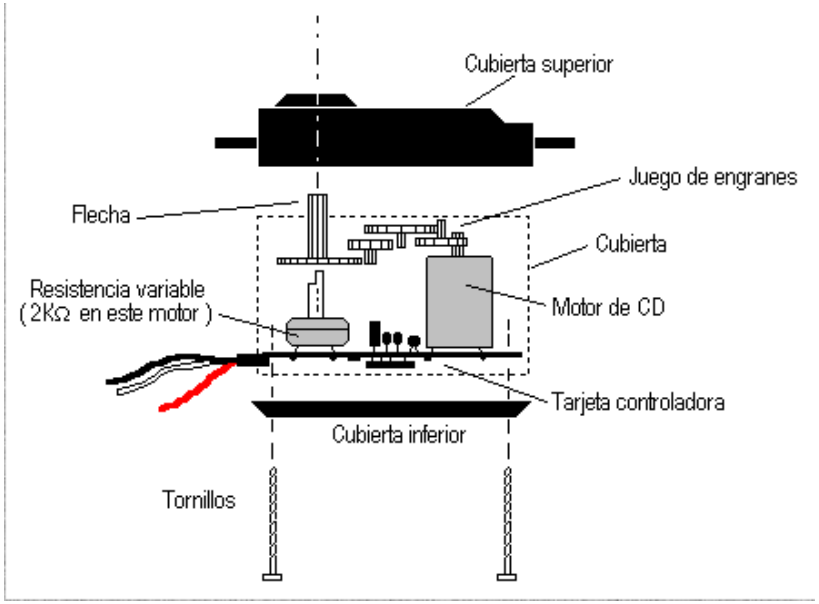


3.3 ACTUADORES - SERVOMOTOR

Servomotores - Arduino

DEFINICIÓN

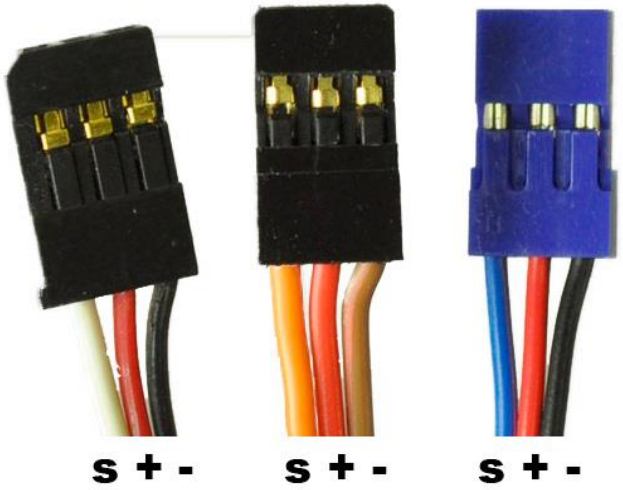
Un servomotor es un motor con control, es decir este dispositivo mecánico posee internamente un controlador que posiciona precisamente el rotor del motor en un ángulo especificado por la señal entrante de control.

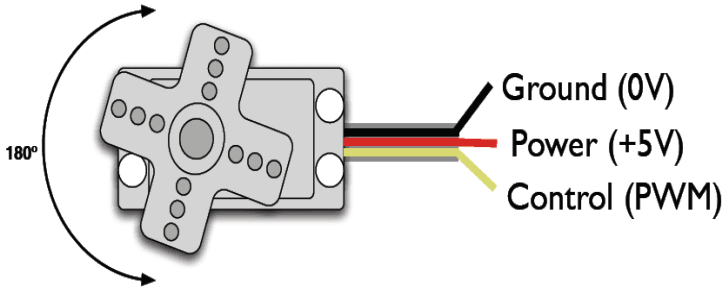


Servomotores - Arduino

Conexión

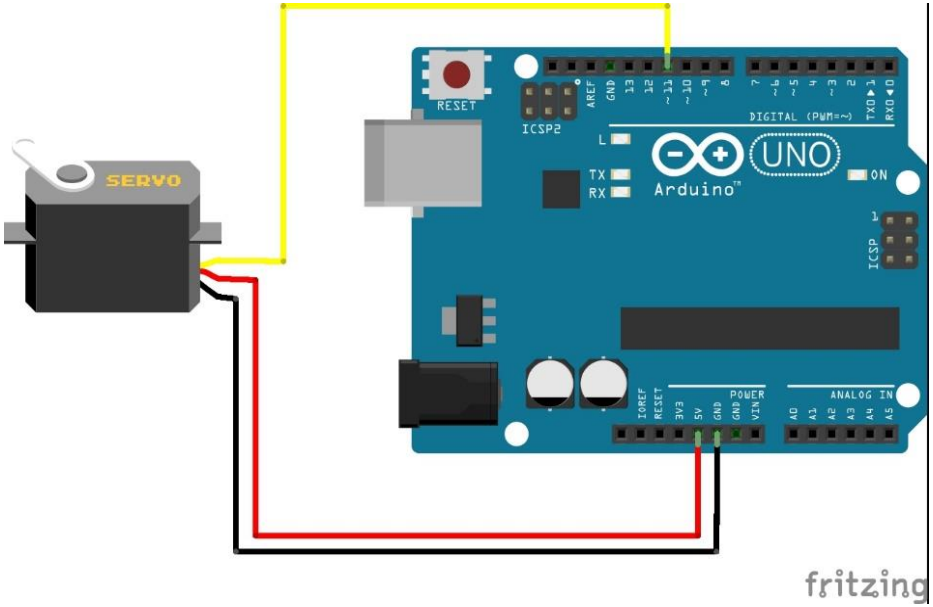
Para entender esto, veamos que físicamente el servomotor se compone de tres cables: GND, Power (5v) y Control. Es decir que los dos primeros se usan como alimentación en cuanto el cable de control es usado para enviar la señal de posicionamiento del motor con una secuencia de pulsos PWM.

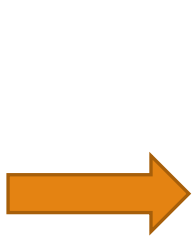




Servomotores - Arduino

Un servo con Arduino podrá tener una potencialidad tremenda, dado que la propia placa de Arduino será la encargada de enviar esa señal PWM para posicionar el motor en una posición en grados en específico.



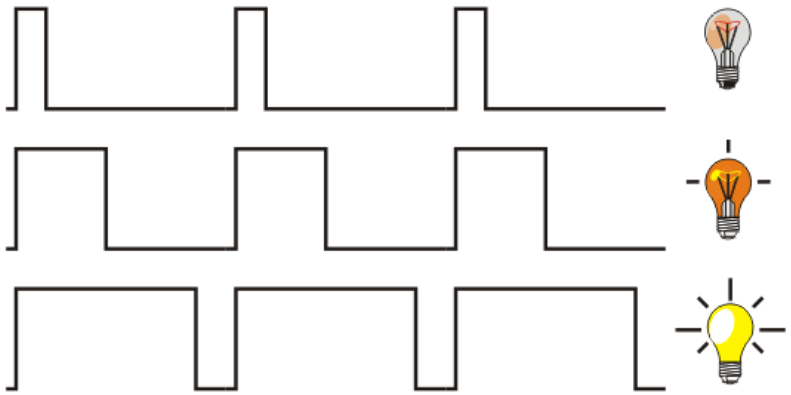


Servomotores - Arduino

Servo Arduino: Control de Posición

La señal PWM que ingresa al Servo a través de Arduino, es procesada y dependiendo del ancho de dicho pulso, el servomotor se desplazará a un determinado ángulo.

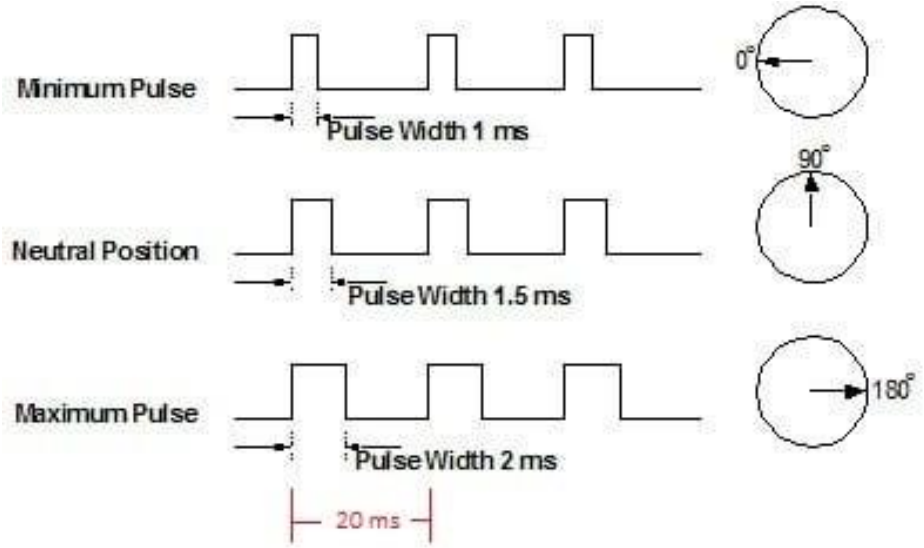
Generalmente los servomotores comerciales procesan un pulso con un periodo de 20ms para conseguir efectuar todo el movimiento del servo adecuadamente.



Servomotores - Arduino

Servo Arduino: Control de Posición

Tomemos como ejemplo un servomotor 180 grados con Arduino. En este caso, para posicionar el motor en 0 grados, nuestro pulso debe ser de 1ms encendido y los 19ms restantes debe estar apagado. Para 90 grados, nuestro pulso debe ser de 1.5ms encendido y los 18.5ms restantes debe estar apagado. Y por último, para 180 grados, nuestro pulso debe ser de 2ms encendido y los 18ms restantes debe estar apagado.



Servomotores - Arduino

Servo Arduino: Características

En el mercado es posible encontrar varios tipos de servos, y es importante entender sus características para adaptarlos adecuadamente a nuestros proyectos, por lo tanto, es de extrema importancia de que leas la hoja de datos del servo que deseas comprar para ver si se adapta en fuerza y grados de movimiento a tu proyecto.

Existen 2 servos básicos que son:

1. Servomotor de -90 grados a 90 grados (total de 180 grados)
2. Servomotor de 360 grados

Servomotores - Arduino

Servo Arduino: Características

Con el servomotor de 180 grados con Arduino podremos controlar la posición donde deseamos ubicar el motor, por otro lado cuando conectamos un servomotor 360 grados en Arduino ya NO tendremos control de posición, porque el servo gira constantemente. Este servomotor de 360 grados es utilizado para realizar control de velocidad, bastante empleado en pequeños o medianos robots para el control de velocidad de las ruedas.

Un factor a tener en cuenta es observar la fuerza del servo, que sea adecuada para mover la carga que necesitamos en nuestro proyecto.

Servomotores - Arduino

Servo Arduino: Calibrar Servomotor Arduino

Vimos que el Arduino necesita de un determinado pulso para poder desplazarse a los diferentes ángulos, sin embargo, cada servomotor tendrá un comportamiento diferente, y antes de usarlo en nuestro proyecto, conviene calibrarlo.

No siempre 1ms va a colocar el Arduino en 0 grados, muchas veces este valor es mucho menor, debemos entonces ir moviendo este valor con la instrucción servo.attach(pin, min, max) que veremos mas adelante para establecer los valores en milisegundos mínimos y máximos en los que el motor hace el barrido completo.

Servomotores - Arduino

Servo Arduino: Programar Servomotor en Arduino

Vamos a emplear la librería del Arduino llamada Servo Library con la cual controlaremos estos dispositivos de una manera adecuada que nos permite controlar hasta 12 servos en Arduino Uno/Nano, y hasta 48 en Arduino Mega. Claro también puedes controlar el servomotor con Arduino sin Librería y para eso te invito a ver la entrada que hicimos con PIC donde hicimos algo parecido, donde básicamente debemos generar la señal PWM.

El servomotor Puede ser conectado en cualquier PIN DIGITAL de la PLACA DE ARDUINO.

Servomotores - Arduino

Servo Arduino: Programar Servomotor en Arduino

Primero incluimos la librería

```
#include<Servo.h>
```

Definimos el PIN de control donde conectaremos el SERVO

Sintaxis:

```
servo.attach(pin)
servo.attach(pin, min, max)
```

Parámetros:

servo: una variable de tipo Servo

pin: el número del pin que el servo está conectado

min (opcional): el ancho de pulso, en microsegundos, correspondiente al ángulo mínimo (0 grados) del servo (por defecto 544)

max (opcional): el ancho de pulso, en microsegundos, correspondiente al ángulo máximo (180 grados) del servo (por defecto 2400)

Servomotores - Arduino

Servo Arduino: Programar Servomotor en Arduino

Para escribir en el servo una posición especifica usamos el comando:

servo.write(angle)

Parámetros:

servo: una variable de tipo Servo

angle: el valor a escribir en el servo, de 0 a 180

Para leer la posición actual en la que se encuentra nuestro servo con Arduino podemos usar:

servo.read()

Parámetros:

servo: una variable de tipo Servo

Retornos:

El ángulo del servo, de 0 a 180 grados

Servomotores - Arduino

Servo Arduino: SERVOMOTOR TRUNCADO A 360

Los servo motores de 360 grados son servos RC estándar que se han modificado para ofrecer un control de velocidad en lazo abierto en lugar de su control habitual de posición de lazo cerrado.

La modificación los convierte en motores con un sistema de motoreducción que le permite aumentar su fuerza y disminuir la velocidad, donde adicionalmente se puede controlar el sentido de giro, todo eso integrado en un paquete compacto y económico.

Se puede usar en ruedas de robots y controlarse mediante una señal RC o una simple conexión directa a una sola línea de E / S del microcontrolador (Arduino).

Servomotores - Arduino

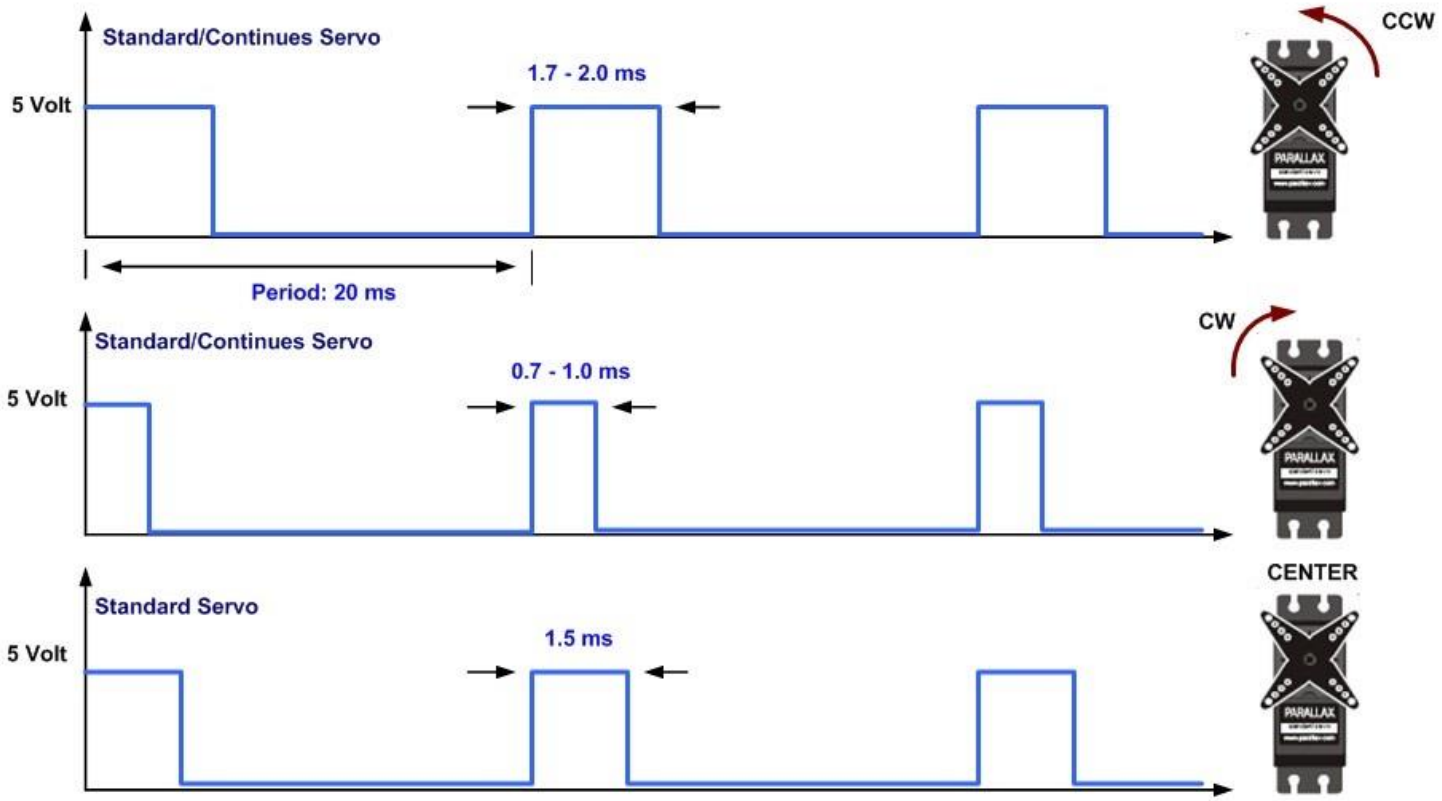
Servo Arduino: CONTROL SERVOMOTOR TRUNCADO A 360

Para eso vamos a valernos de la misma librería del Arduino solo que en este caso usaremos las siguientes configuraciones para manipular un servo continuo o de 360 grados.

Primero veamos que las señales PWM que deben ser enviadas son exactamente las mismas, con la diferencia de que ya no tendremos el control de posición (CCW: sentido antihorario; CW: Centido Horario)

Servomotores - Arduino

Servo Arduino: CONTROL SERVOMOTOR TRUNCADO A 360



Servo Motor PWM Timing Diagram

Servomotores - Arduino

Servo Arduino: DETENER SERVOMOTOR TRUNCADO A 360

Es decir que si deseas saber como detener un servomotor con arduino de 360 usamos la siguiente instrucción:

```
servo.write(90);
```

Detener un servomotor de 360 grados es equivalente a programar un servomotor común en 90 grados.

Servomotores - Arduino

Servo Arduino: Sentido Anti-Horario SERVOMOTOR TRUNCADO A 360

.

```
servo.write(180);
```

Mover un servomotor de 360 grados en sentido anti-horario es equivalente a programar un servomotor común en 180 grados.

Servomotores - Arduino

Servo Arduino: Sentido Horario SERVOMOTOR TRUNCADO A 360

---

servo.write(0);

Mover un servomotor de 360 grados en sentido horario es equivalente a programar un servomotor común en 0 grados.

.

Servomotores - Arduino

Servo Arduino: Controlar la Velocidad SERVOMOTOR TRUNCADO A 360

Primero que todo cabe aclarar que el control de velocidad de un servomotor de 360 grados no es muy precisa, por lo tanto si tu objetivo es lograr una velocidad precisa deberas optar por agregar un sensor adicional.

Como ya lo notaste, en un servo de giro completo en lugar de enviar un ángulo entre 0 y 180 grados lo que estaremos enviando realmente es su **velocidad** y no solo **el sentido** siendo, el control de velocidad del servo 360 se logra entonces como:

.

Servomotores - Arduino

Servo Arduino: Controlar la Velocidad SERVOMOTOR TRUNCADO A 360

Como ya lo notaste, en un servo de giro completo en lugar de enviar un ángulo entre 0 y 180 grados lo que estaremos enviando realmente es su **velocidad** y no solo **el sentido** siendo, el control de velocidad del servo 360 se logra entonces como:

0: máxima velocidad de giro en sentido horario.

Entre 0 y 90: se conserva el mismo sentido horario pero la velocidad se reduce conforme nos acercamos a 90

90: el servomotor 360 se detiene.

Entre 90 y 180: la velocidad aumenta conforme nos acercamos al valor de 180, girando en sentido antihorario.

180: máxima velocidad de giro en el sentido antihorario.

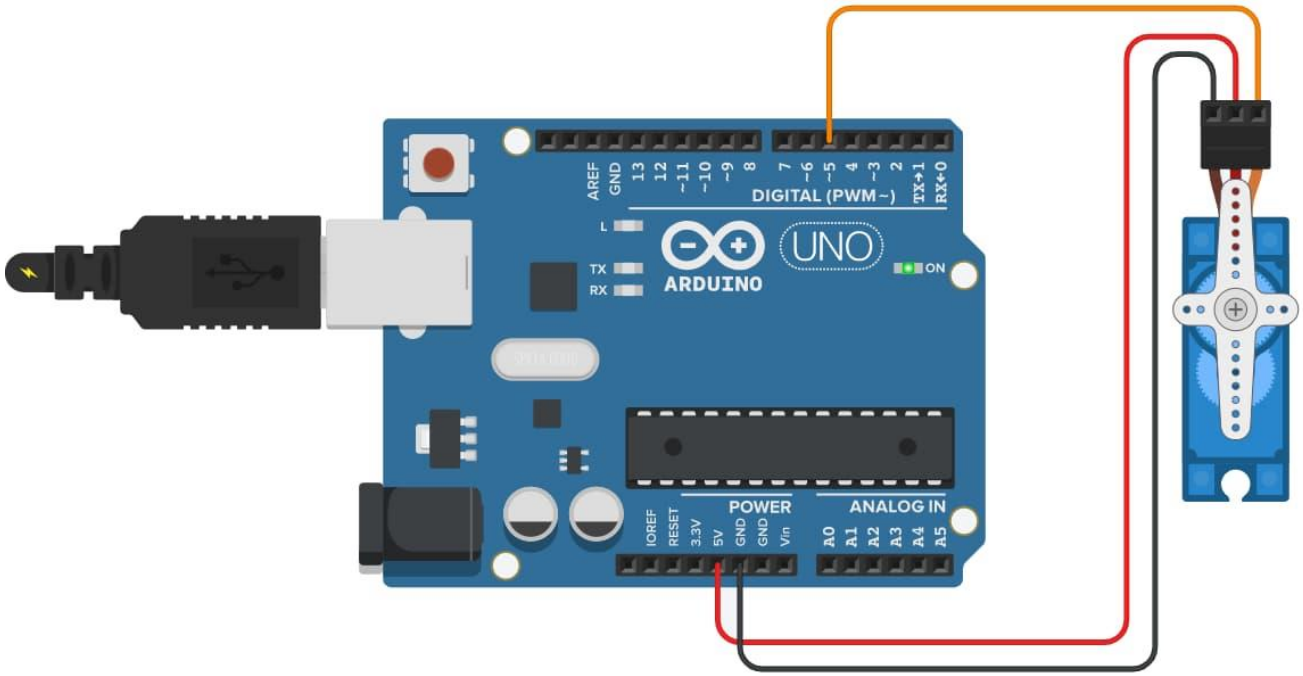
Servomotores - Arduino

ANTES DE EMPEZAR CON LA PRÁCTICA

Servomotores – Arduino - Ejemplos

1. Servo motor de 180 controlado.

Implementar el circuito



Servomotores – Arduino - Ejemplos

Servo inicial

```
#include <Servo.h>
Servo myservo;
int pos = 0;
void setup() {
  myservo.attach(5);
}
void loop() {
myservo.write(45);
delay(1000);
myservo.write(0);
delay(1000);
}
```

Servomotores – Arduino - Ejemplos

Servo motor de 0 a 180 y de 180 a 0 grados controlado.

2.-Codigo

```
#include <Servo.h>
Servo myservo;
int pos = 0;
void setup() {
  myservo.attach(9);
}
void loop() {
  for (pos = 0; pos <= 180; pos += 1) {
  myservo.write(pos);
  delay(15);
  }
  for (pos = 180; pos >= 0; pos -= 1) {
    myservo.write(pos);
    delay(15);
  }
}
```

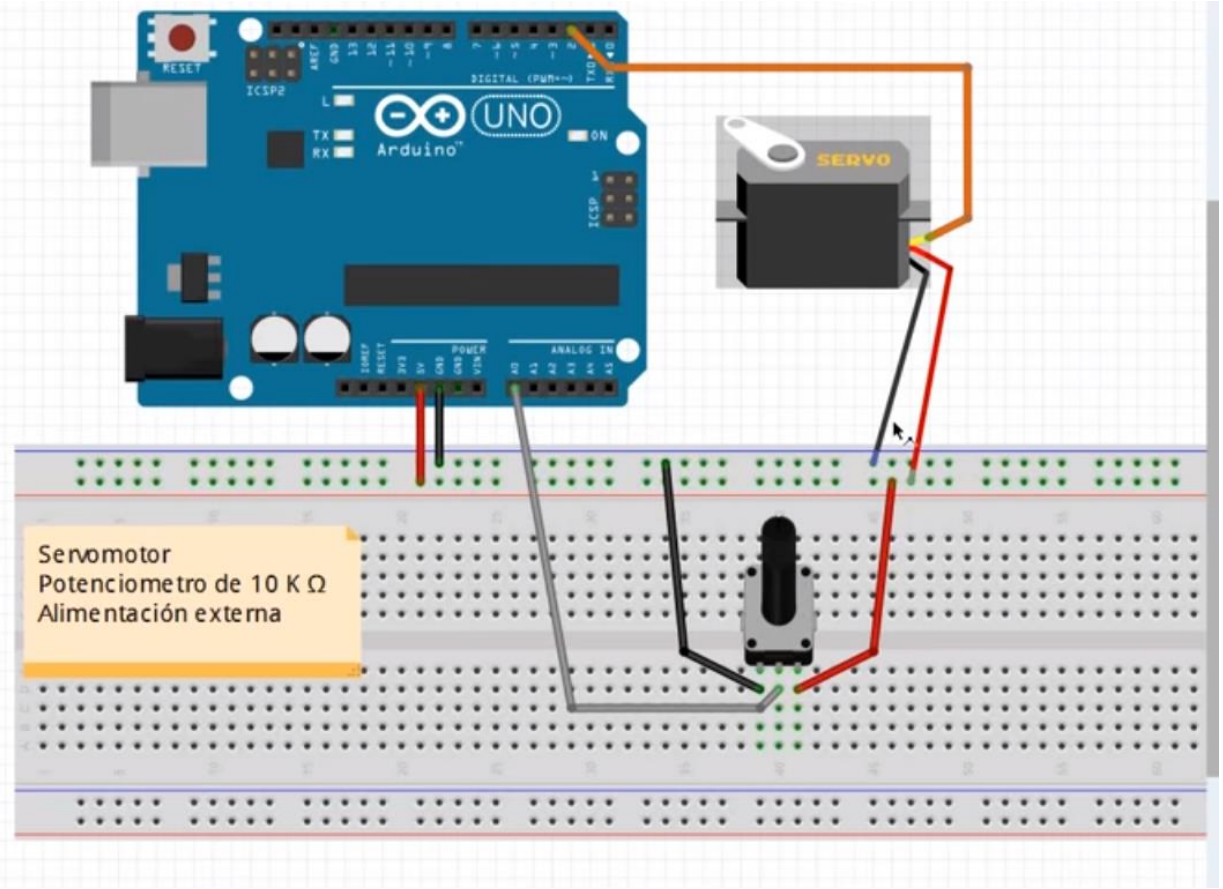
```
for (valor_inicial_contador;condicion_final;incremento) {
//Instrucciones que se repetirán un número determinado de veces
}
```

Operadores Aritméticos

Servomotores – Arduino - Ejemplos

2. Servo motor de 180 controlado por potenciómetro.

Implementar el circuito



Servomotores – Arduino - Ejemplos

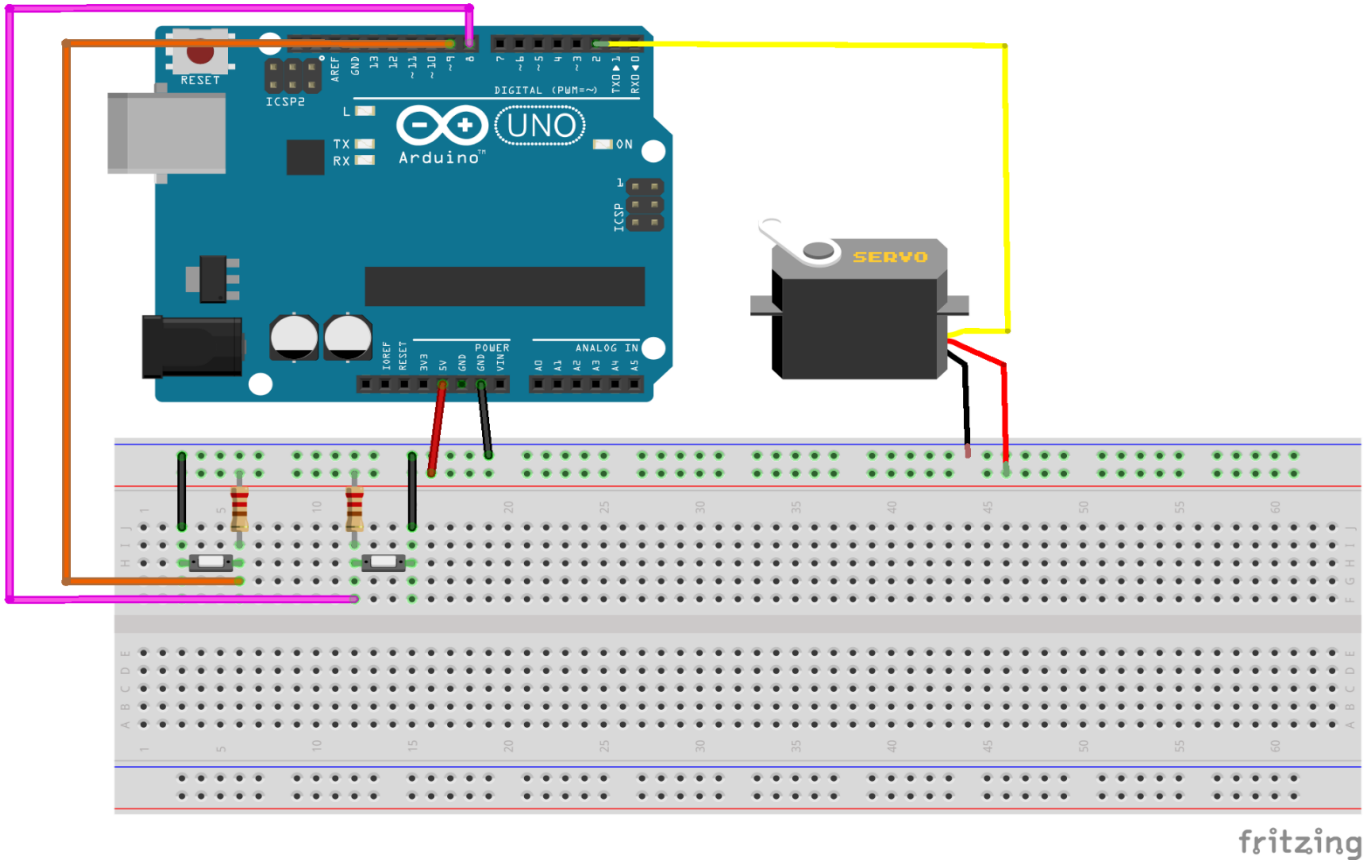
Servo motor controlado con potenciometro.

```
#include <Servo.h>                          // incluye libreria de Servo
Servo servo1;                               // crea objeto
int PINSERVO = 2;                  // pin 2 conectado a señal del servo
-------------------------------------------------------------------------------------------------------
int PULSOMIN = 1000;                        // pulso minimo en microsegundos
int PULSOMAX = 2000;                        // pulso maximo en microsegundos
int VALORPOT;                               // variable para almacenar valor leido en entrada A0
int ANGULO;                                 // valor de angulo a cargar en el servo
int POT = 0;                                // potenciometro en entrada analogica A0
void setup()
{
 servo1.attach(PINSERVO, PULSOMIN, PULSOMAX);   // inicializacion de servo
// las entradas analogicas no requieren inicializacion
}
void loop()
{
 VALORPOT = analogRead(POT);                          // lee valor de entrada A0
 ANGULO = map(VALORPOT, 0, 1023, 0, 180);   // con funcion map convierte rango de 0 a 1023
                                                                  // a rango de angulo de 0 a 180
 servo1.write(ANGULO);                                  // envia al servo el valor del angulo
 delay(20);                                             // demora para que el servo llegue a posicion
}
```

Servomotores – Arduino - Ejemplos

3. Servo motor de 180 controlado por pulsos.

Implementar el circuito



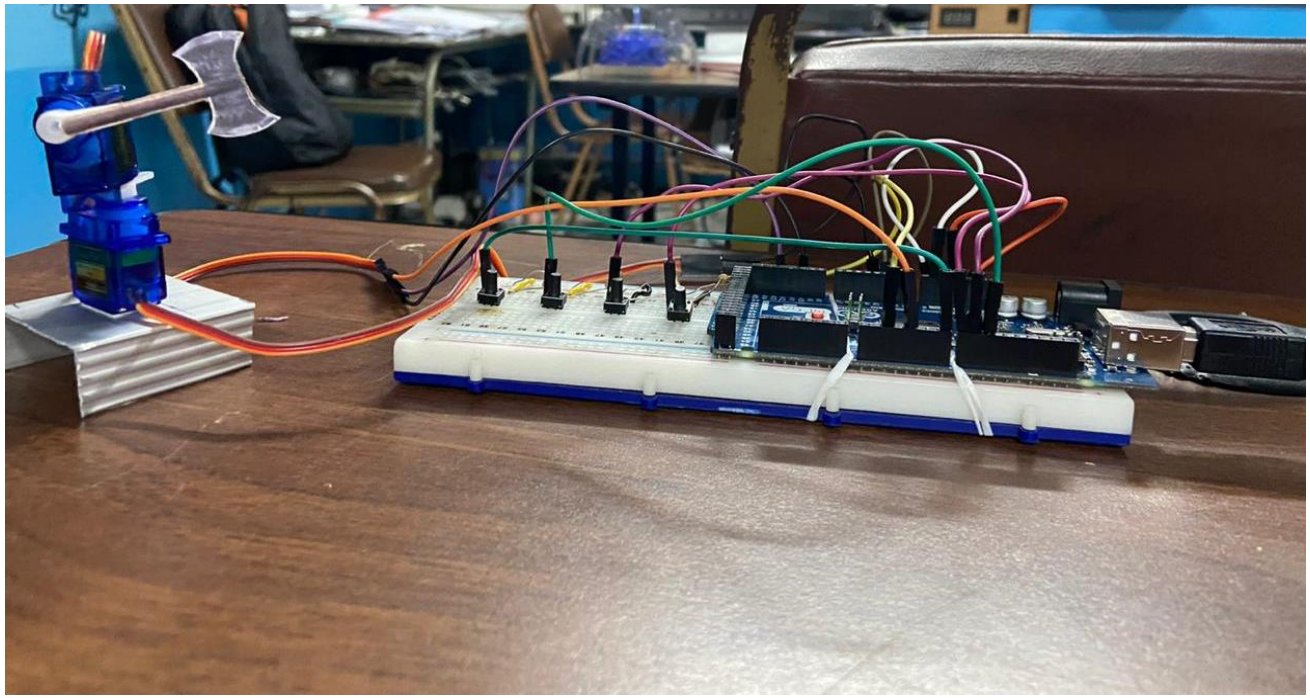
Servomotores – Arduino - Ejemplos

```
//Servo motor controlado con pulsos

#include <Servo.h>

Servo mi_servo;

int grados = 90;   // Inicializa en la posición 90°

int suma=8;  // Boton para sumar angulo

int resta=9; // Boton para restar angulo

void setup() {

 mi_servo.attach(2,750,1800);  // Configura el Servo, recuerden que en mi caso 750ms = 0° y 1800 ms = 180°

 pinMode(suma, INPUT);  // Configuros pines de entrada

 pinMode(resta, INPUT);

 mi_servo.write(grados);  // Muevo el motor hasta 90°


 //digitalWrite(suma,HIGH);

// digitalWrite(resta,HIGH);

}

void loop() {


 if (digitalRead(suma) == LOW) // Pregunto por el boton suma = 12 Presionado, se activa con estado Bajo

 {

  grados++;   // Suma grados


  if (grados >= 180) // Proteje el motor, para que no exceda los 180°, se puede dañar el motor

  {
```

Servomotores – Arduino - Ejemplos

4. Servos motores de 180 controlado por pulsos.

Implementar el circuito



Servomotores – Arduino

Conversión de escala

La conversión de una variable de entrada análoga a un valor de medida real, se lo hace mediante una conversión de escala.

Toda entrada análoga tiene un valor de lectura mínimo en bits y un lectura máxima en bits.

Lectura mínima : 0 bits

Lectura máxima : 1023 bits

Servomotores – Arduino

Conversión de escala - **Función map**

La función map() de Arduino permite transformar un valor entero de un rango de entrada al valor correspondiente a otro rango de salida.

Los 5 parámetros de entrada son valores enteros:

valor de entrada

inicio rango de entrada

final rango de entrada

inicio rango de salida

final rango de salida

Y la función devuelve el valor entero de salida una vez realizado el «mapeo».

Servomotores – Arduino

Conversión de escala - **Función map**

```
valor_led = map(valor_ldr, 0,320, 0,255);
```

El «**valor-ldr**» corresponde al valor leído por un sensor de luz, cuyo rango puede variar de **0** a **320**, valores que ponemos en el segundo y tercer lugar; los valores de salida deben oscilar entre **0** y **255** (valores cuarto y quinto) porque servirán para iluminar un led, este valor se almacena en la variable «**valor_led**» como valor de salida de la función map().