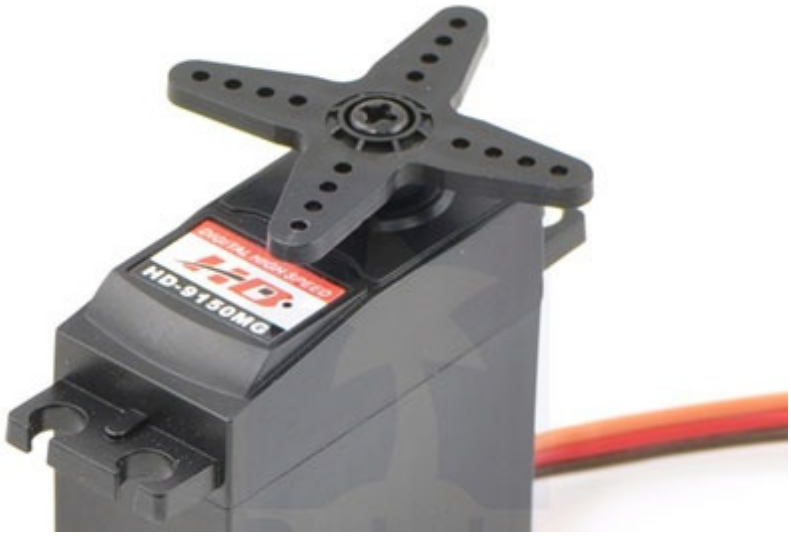
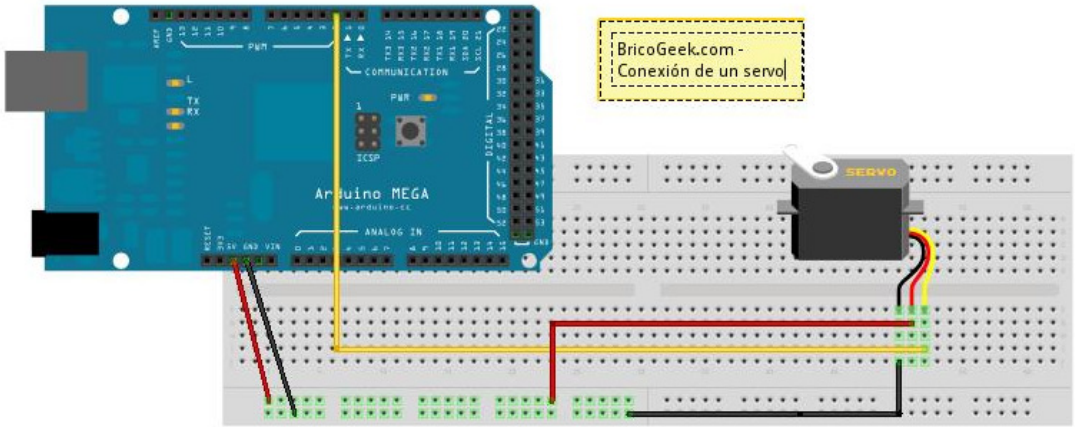
CONTROL DE UN SERVO CON ARDUINO



La interfaz de conexión del servomotor consta normalmente de 3 cables o conductores, generalmente de 3 colores distintos,a saber:

-Rojo
-Marrón o Negro
-Azul, Amarillo u otros

Donde por norma general el cable Rojo es el positivo o fase, el Marrón o Negro es el tierra o masa, y el restante de varios colores posibles es el de control, que nos permitirá controlar el servomotor con precisión.
Normalmente la señal que controla el servo es de tipo PWM, o sea pulsos de ancho variable, con los que podemos mover con precisión el servomotor a cualquier punto de su radio de acción.
¿Y como podemos provocar dichos pulsos?
Pues muy sencillo usando uno de los pines marcados como PWM en arduino.
Para conectar nuestro servo con arduino procederemos según el esquema:



Tal como se observa en el gráfico, se conecta el servo al positivo que nos da arduino, y al GND también proporcionado por nuestro arduino, y por ultimo conectamos la señal de control (amarillo) al pin numero 2, marcado como PWM.

Para el control, usaremos una librería para el control de servos que se llama Servo.h. que nos ofrece los métodos:

- attach(int):Para poner un pin en modo servo drive.
- detach() :Libera un pin del modo servo driver.
- write(int): Indica el ángulo a girar el servo, 0 a 180.
- read(): obtiene el ultimo valor enviado (posición del servo).
- attached(): devuelve 1, si el servo está conectado.
- refresh():Se debe llamar a ésta function al menos cada 50ms para asegurarse que los servos estarán en su posición.

El siguiente código que nos permita mover el servomotor. Vemos que usamos

view plaincopy to clipboardprint?

```
1. #include <Servo.h>
2.
3. Servo servo1;  // Crea un Objeto servo
4. int posicion;    // Variable de la posicion del servo
5.
6. void setup()
7. {
8.   servo1.attach(2);  // Selecionamos el pin 2 como el pin de con
   trol para els ervo
9. }
10.
11.      void loop()
12.      {
13.        val = analogRead(potpin);   // reads the value of the
   potentiometer (value between 0 and 1023)
14.        val = map(val, 0, 1023, 0, 179);     // scale it to use
   it with the servo (value between 0 and 180)
15.        myservo.write(val); // sets the servo position according
   to the scaled value
16.        delay(15);         // waits for the servo to get there
17.
18.        SoftwareServo::refresh();
19.
20.      }
```