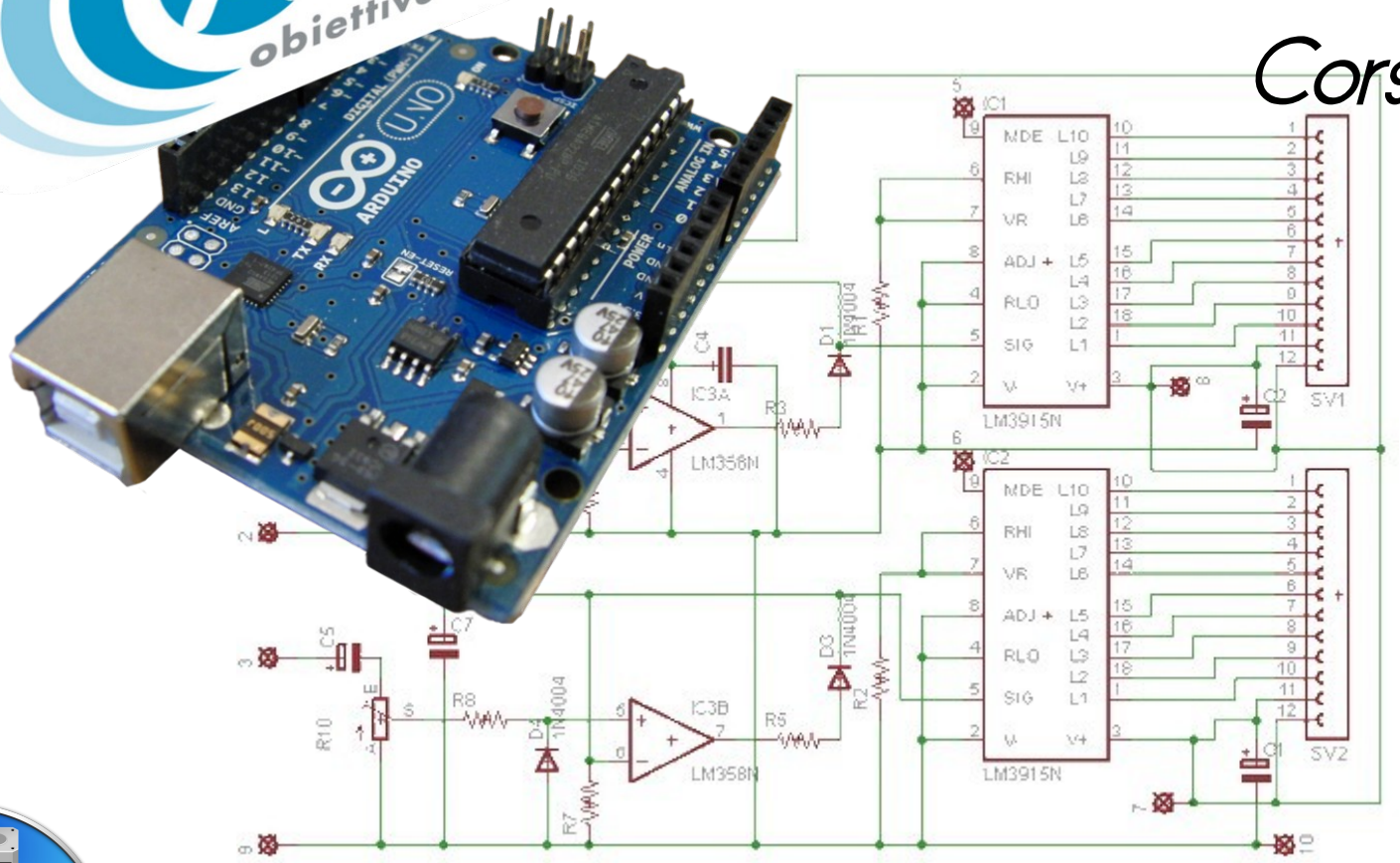
CORSO ARDUINO

Giulio Fieramosca
Stefano Panichi

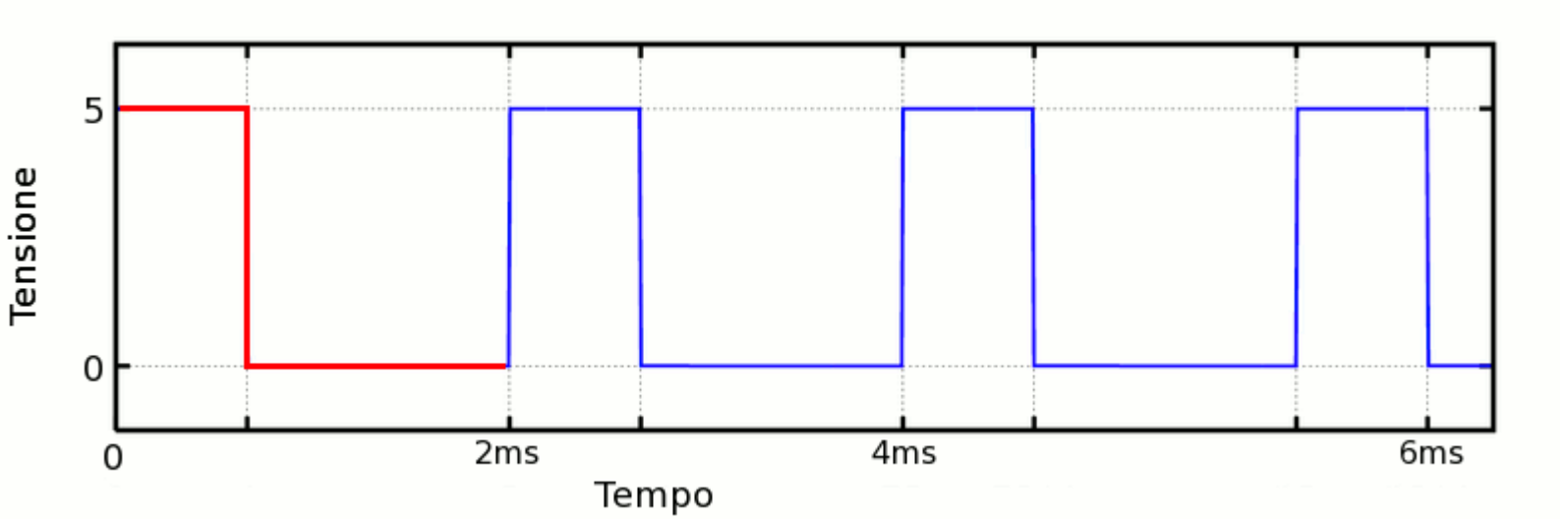
Corso ASEV 2014



Stefano Panichi e Giulio Fieramosca

PWM

modulazione a larghezza d'impulso



Formule e Dati

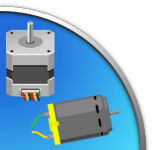
T_{on} = 1/3 del periodo

T_{off} = 2/3 del periodo

Duty Cycle = T_{on} / periodo

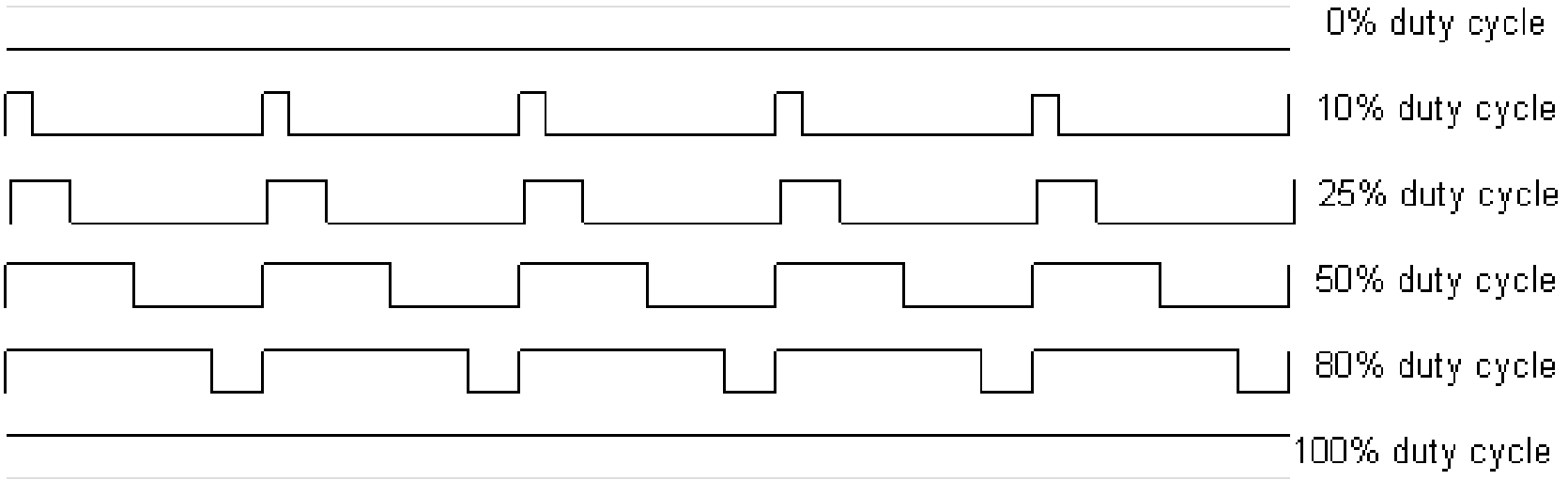
Periodo = T_{on} + T_{off} = 1/Frequenza

V_{media} = 5V * Duty Cycle

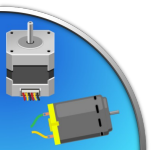


Stefano Panichi e Giulio Fieramosca

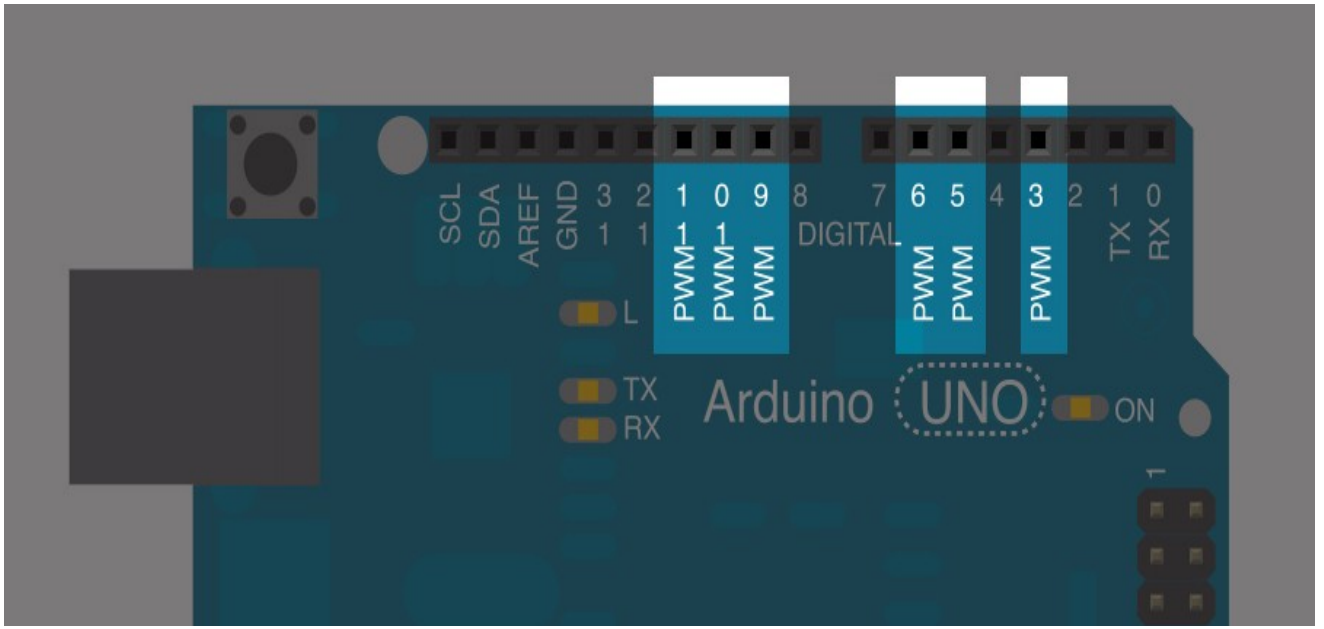
Utilizzi del PWM



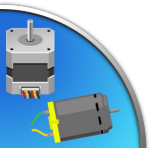
- Regolazione luminosità dei led *"Dimming"*;
- Regolazione velocità motori;
- Pilotaggio di *servomotori* analogici;
- Pilotaggio speaker;



Pin di Arduino



- Arduino dispone di 6 piedini digitali dedicati al PWM.
- Ciò consente di gestire i segnali in modo autonomo (*asincrono*) rispetto al programma.



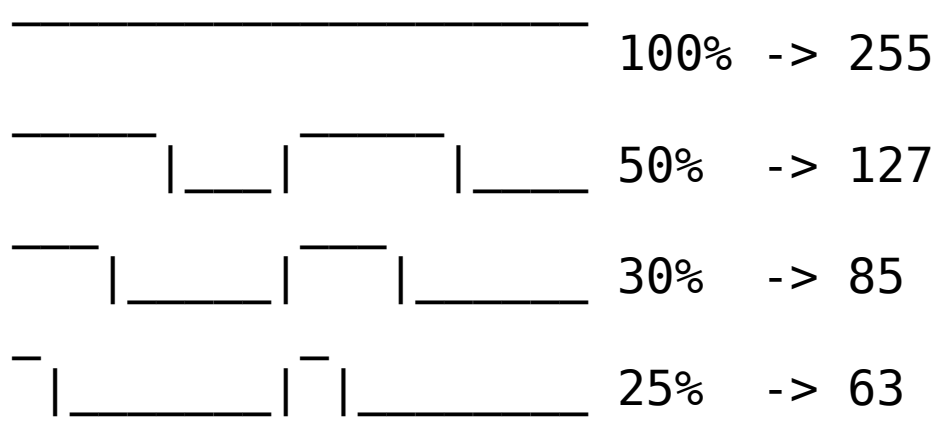
Stefano Panichi e Giulio Fieramosca

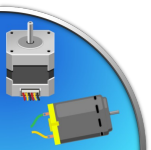
Programmazione del PWM

La funzione per mandare in uscita il PWM con il duty cycle desiderato è

```
analogWrite(pin, duty);
```

Il pwm è scalato a 8 bit, ovvero valori *duty* compresi fra 0 e 255 corrispondono ai duty cycle 0 – 100%



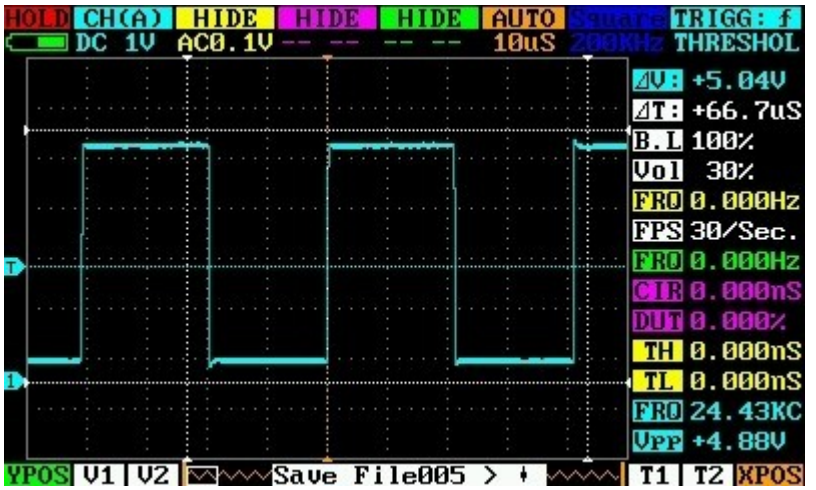


Frequenze di PWM

La frequenza (e quindi il periodo) del PWM è fissata a valori predefiniti:

- 490 Hz per i pin 3, 9, 10, 11;
- 980 Hz per i pin 5, 6;

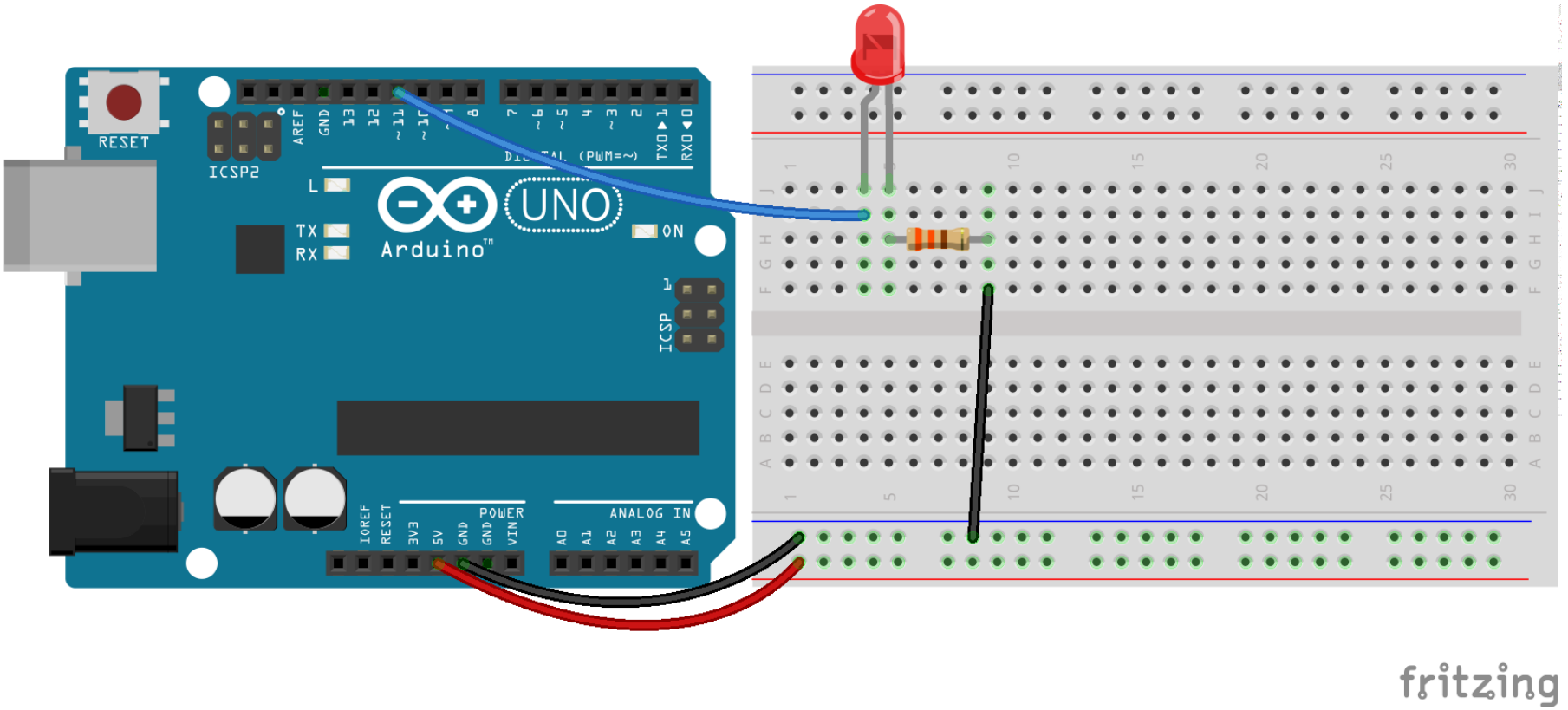
È possibile variarle, ma comporterebbe problemi sulle funzioni di temporizzazione (millis, delay, eccetera)



Vista all'oscilloscopio digitale

Stefano Panichi e Giulio Fieramosca

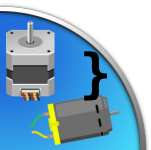
Collegamenti: Led Dimmer



Stefano Panichi e Giulio Fieramosca

Sketch: Led Dimmer

```
const byte PWMpin = 11;

void setup() {
   pinMode(PWMpin, OUTPUT);
}
void loop() {
   // Accensione
   for (byte dim = 0; dim < 255; dim++) {
     analogWrite(PWMpin, dim);
     delay(10);
   }

   // Spegnimento
   for (byte dim = 255; dim > 0; dim--) {
     analogWrite(PWMpin, dim);
     delay(10);
   }
}
```



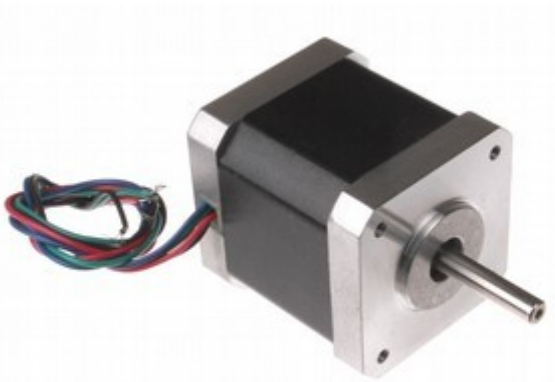
Stefano Panichi e Giulio Fieramosca

Motori

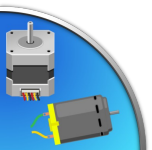
Troviamo sul mercato tre categorie di motori

Motori DC:
2 fili

Motori stepper:
4~6 fili

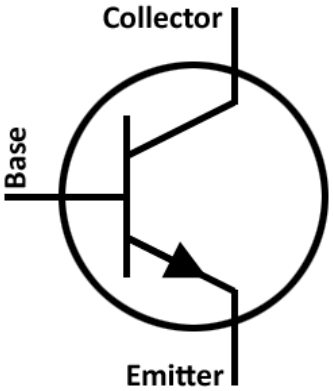


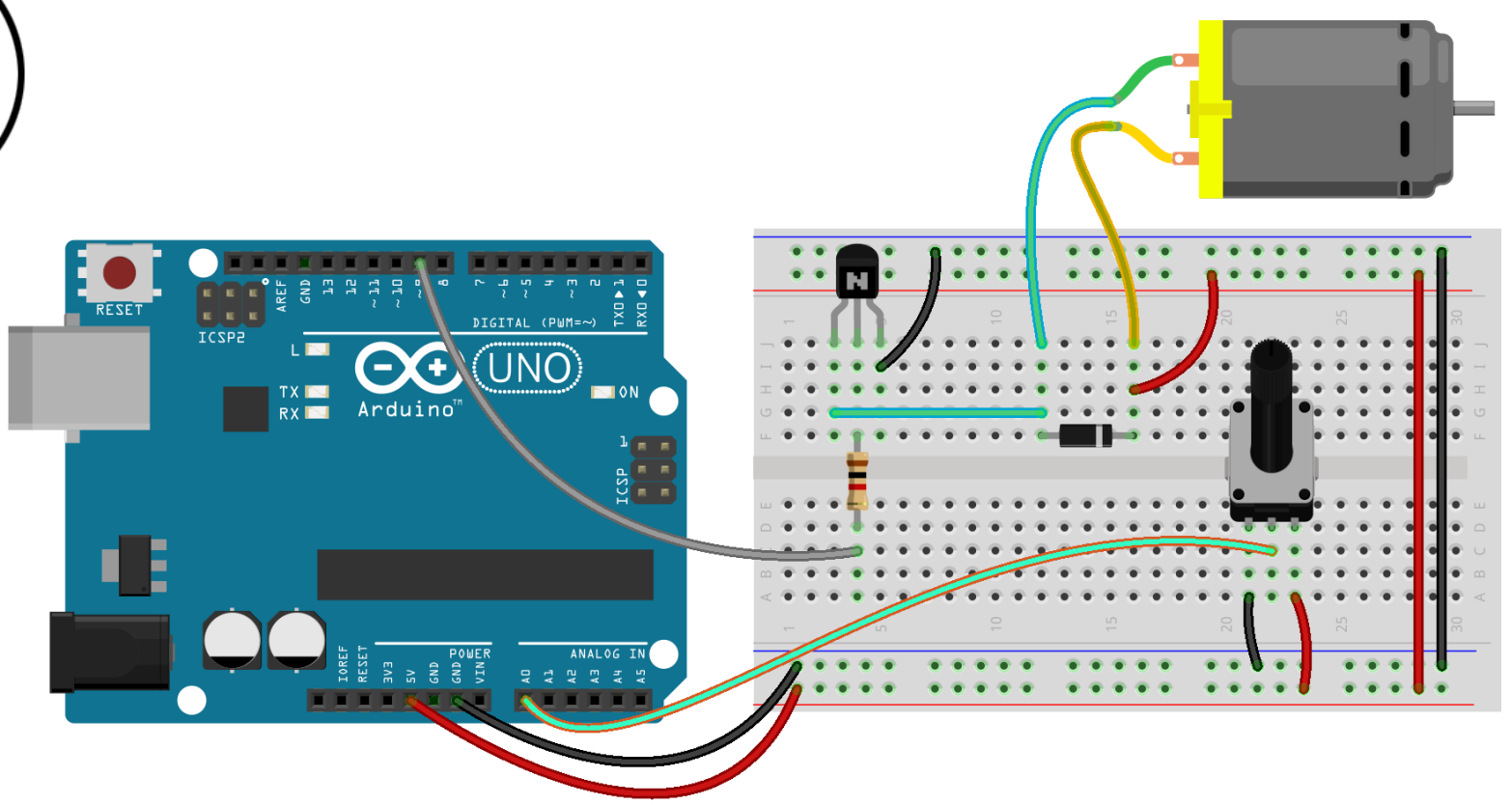
Servomotori:
3 fili



Stefano Panichi e Giulio Fieramosca

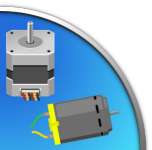
Motore DC: metodo "transistor"





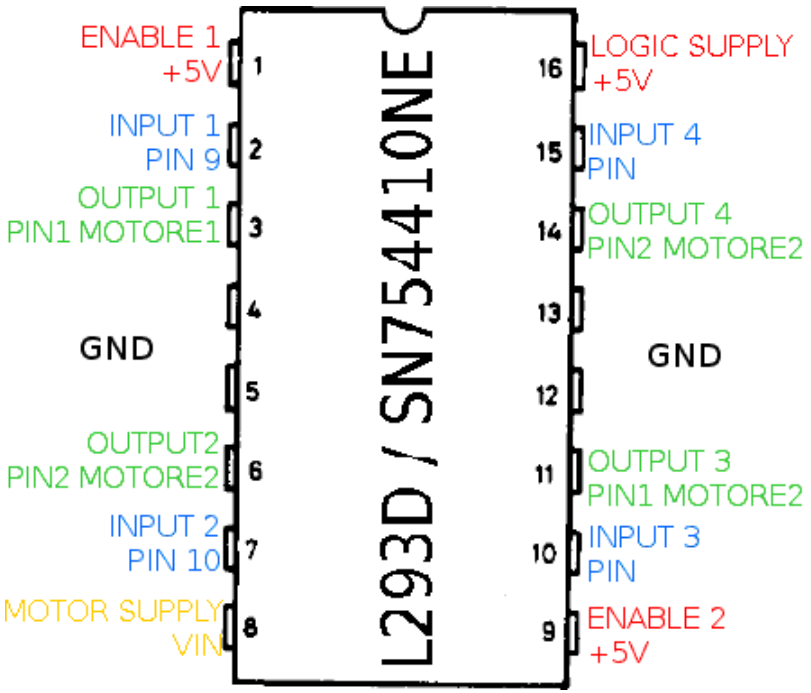
Listato Motore DC Transistor

```
const byte POTENZ = A0; // potenziometro
const byte MOTORE = 9;  // motore

void setup() {
   // inizializza il motore come output
   pinMode(MOTORE, OUTPUT);
}

void loop(){
   byte valore = map(analogRead(POTENZ),0,1023,0,255);
   // il motore gira con velocità proporzionale alla
   // rotazione del potenziometro
   analogWrite(MOTORE,valore);
   delay(10);
}
```

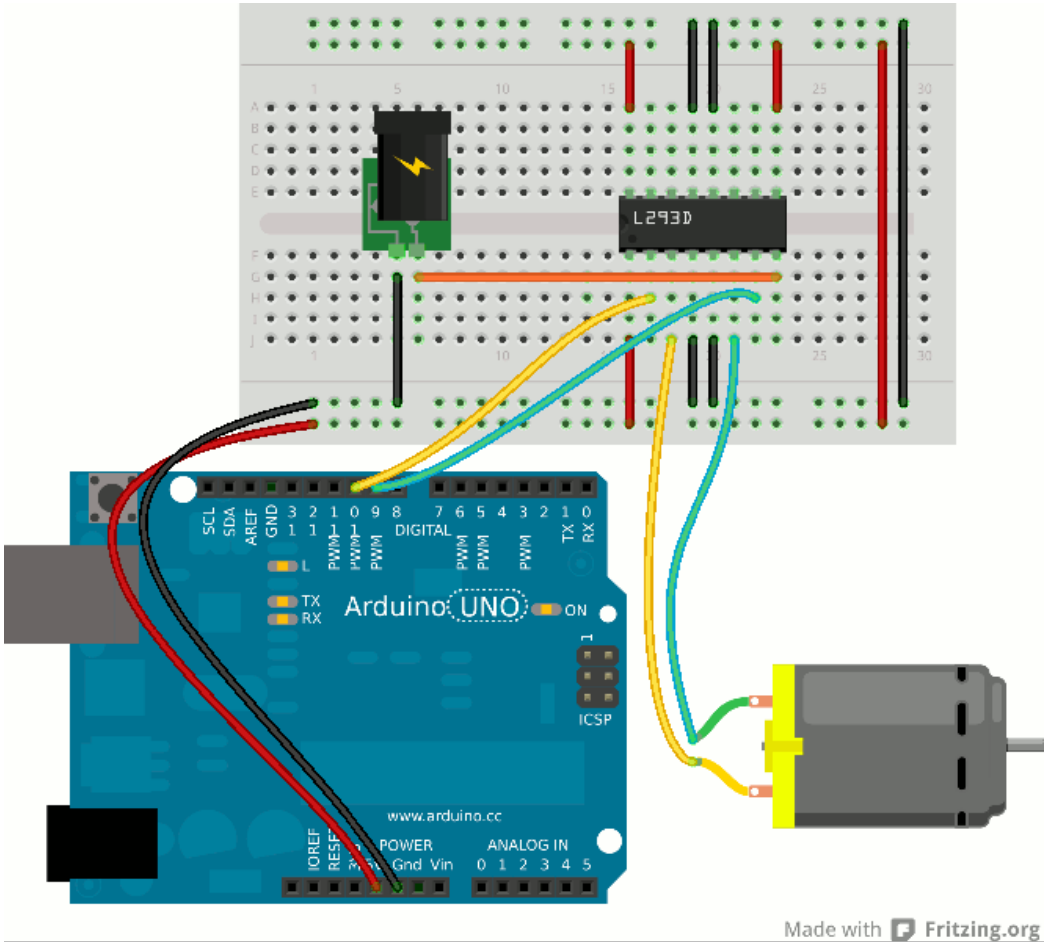


Stefano Panichi e Giulio Fieramosca

Motore DC: metodo "ponte H"

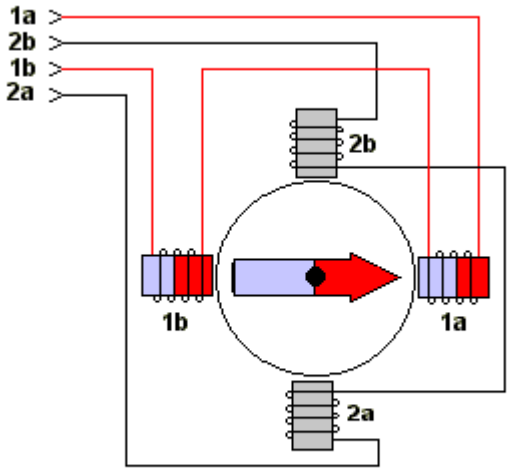


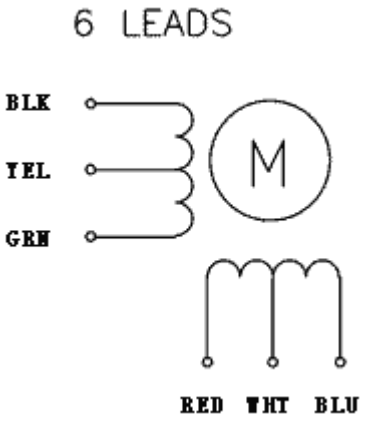
DRIVER



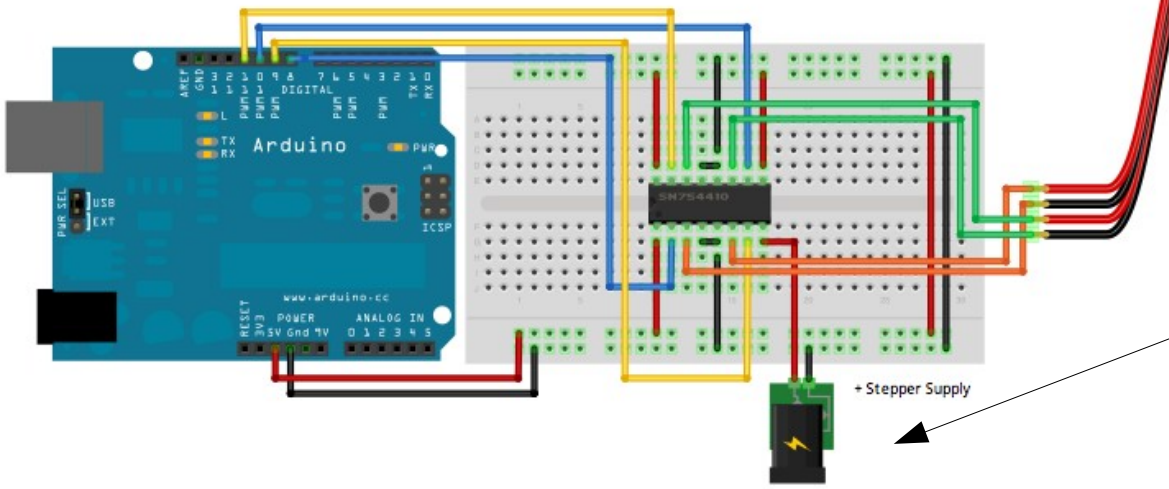
Stefano Panichi e Giulio Fieramosca

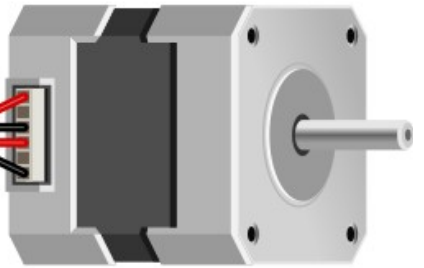
Pilotare uno Stepper bipolare





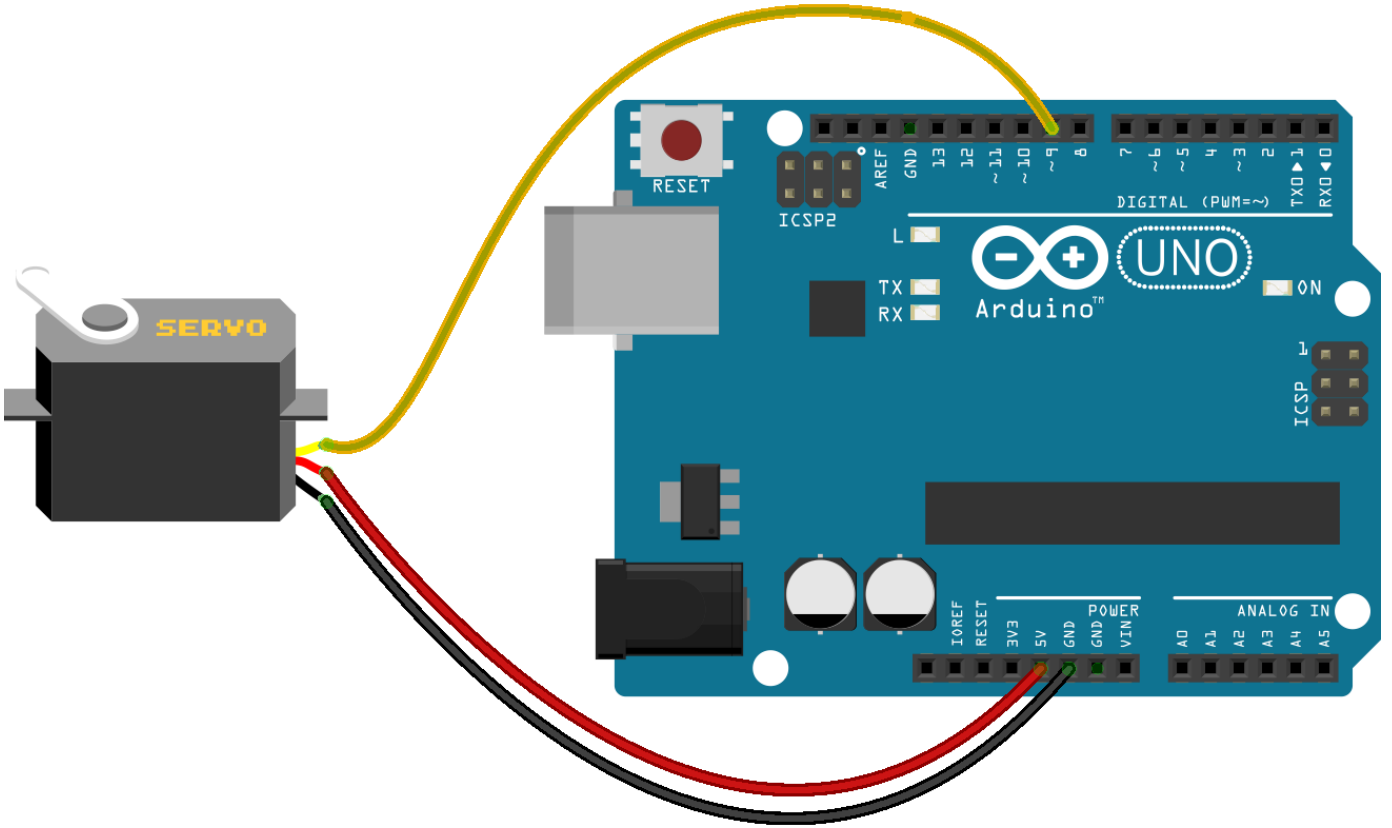
Conceptual Model of Bipolar Stepper Motor

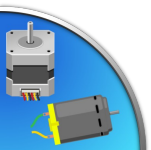




Alimentazione esterna:
Gli stepper funzionano perlopiù a 12~24 V

Motore Servo

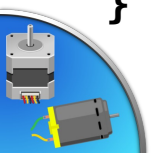




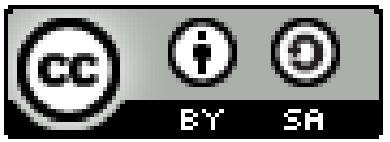
Stefano Panichi e Giulio Fieramosca

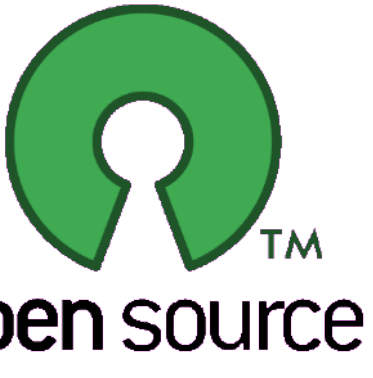
Motore Servo: listato

```
#include <Servo.h>    // Libreria per i servo
Servo myservo;       // crea un oggetto Servo (myservo)

void setup()
{
  myservo.attach(9);  // setta il pin 9 al servo
}

void loop()
{
  for (byte passi = 0; passi < 180; passi++) {
     myservo.write(passi);                  // Muove il servo a zero
     delay(10);
   }
  delay(1000);  // Pausa
   for (byte passi = 178; passi > 0; passi--) {
     myservo.write(passi);                  // Muove il servo a zero
     delay(10);
   }
  delay(1000);  // Pausa
}
```



Stefano Panichi e Giulio Fieramosca





Presentazione realizzata con software open source
(LibreOffice Impress, Gimp, Arduino, Fritzing)