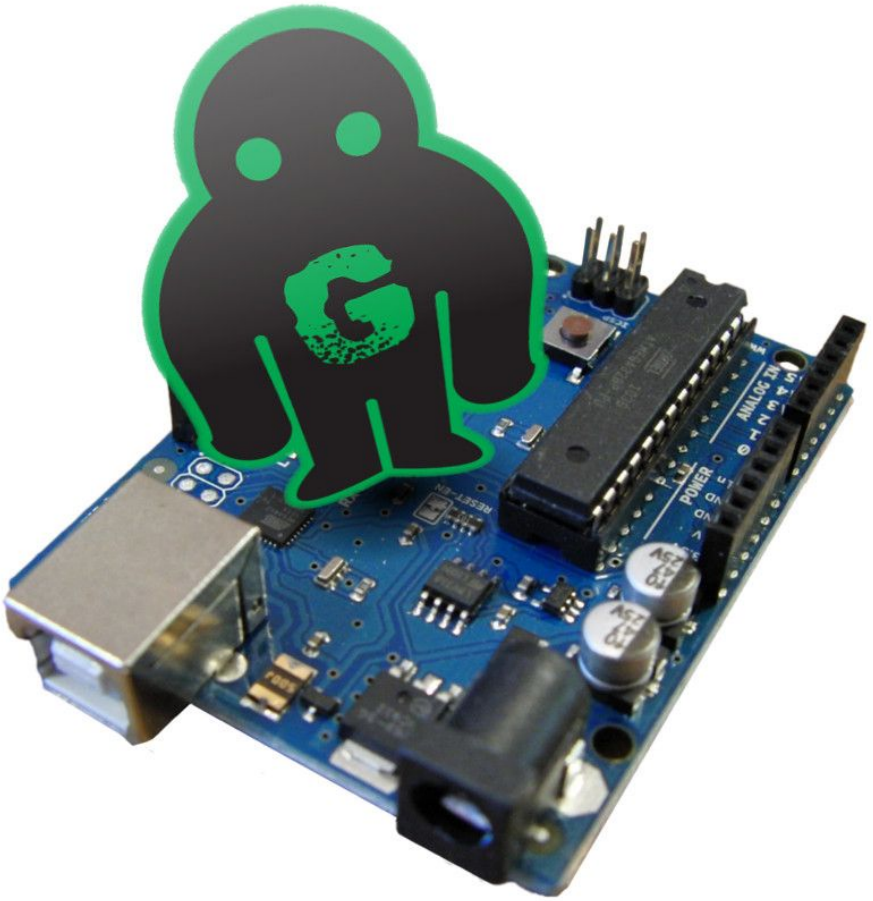
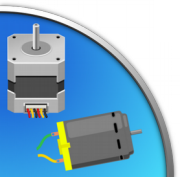
CORSO ARDUINO

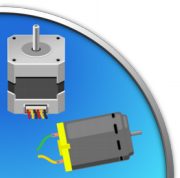


Jacopo Belli
Giulio Fieramosca
Luca Mattii
GOLEM 2016



FAQ time... domande e risposte

?



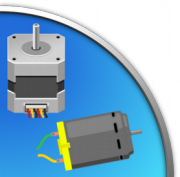
Iterazioni: while

Si usa per ripetere un blocco di codice finché una condizione è vera, ma senza tenere il conto dei giri;

```
while (condizione) {
     // Il programma esegue questo blocco
     // finché la condizione è vera
}
```

Caso particolare: non fa nulla finché la condizione è verificata

```
while (digitalRead(PIN_BOTTONE) == LOW) {
    // Non fa niente
}
```



Iterazioni: for

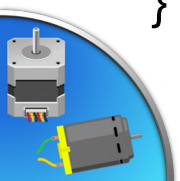
- Posso ripetere un blocco di codice tenendo il conto di quanti giri vado a fare.
- *Esempio:* esercizio “Luci di SuperCar”

Inizializzo una o più variabili da usare come contatore

Controllo ad ogni ciclo se la condizione è vera

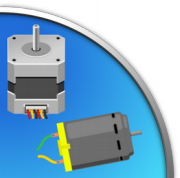
Incremento il contatore alla fine di ogni ciclo

```
void setup () {
    for (byte c = 0; c < 5; c++) {
        pinMode (9 + c, OUTPUT);
        digitalWrite (9 + c, LOW);
    }
}
```



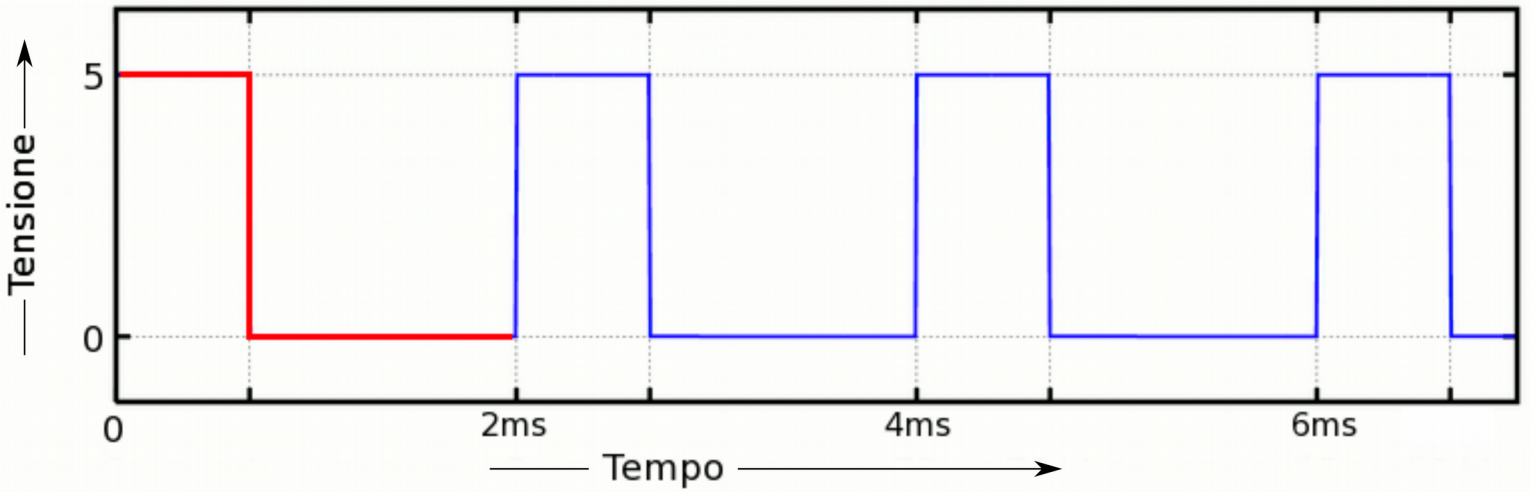
Blink... again!

```
const byte LED = 13;
void setup() {
  pinMode(LED, OUTPUT);
}
void loop() {
  digitalWrite(LED, HIGH);
  delay(2);
  digitalWrite(LED, LOW);
  delay(2);
}
```



PWM

modulazione a larghezza d'impulso



Formule e Dati

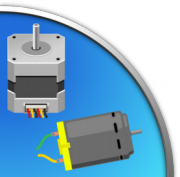
T_{on} = 1/3 del periodo

T_{off} = 2/3 del periodo

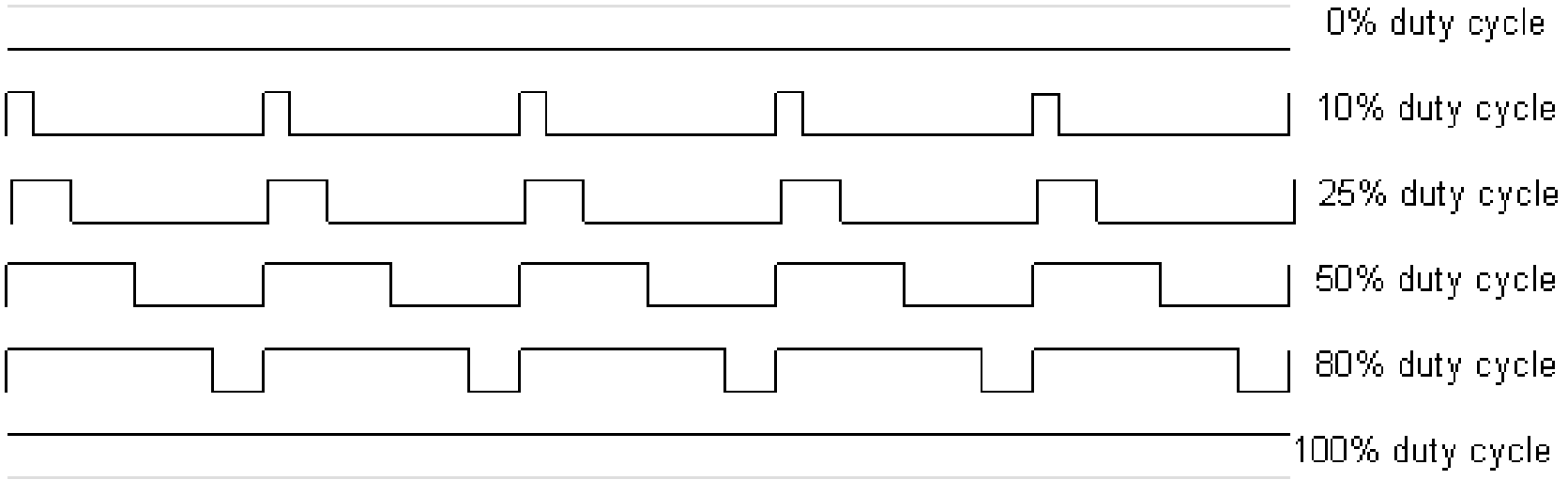
Duty Cycle = T_{on} / periodo

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

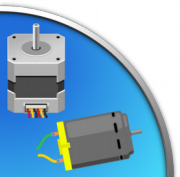
V_{media} = 5V * Duty Cycle



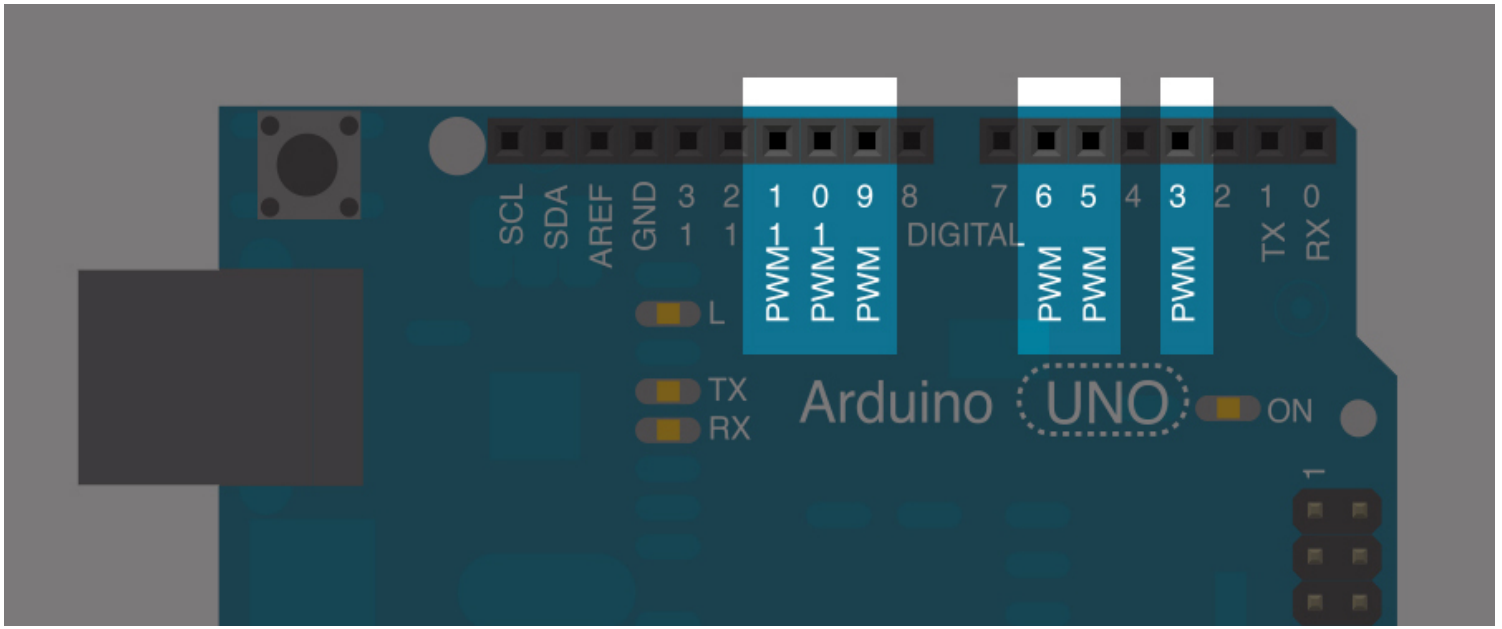
Utilizzi del PWM



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



Pin di Arduino



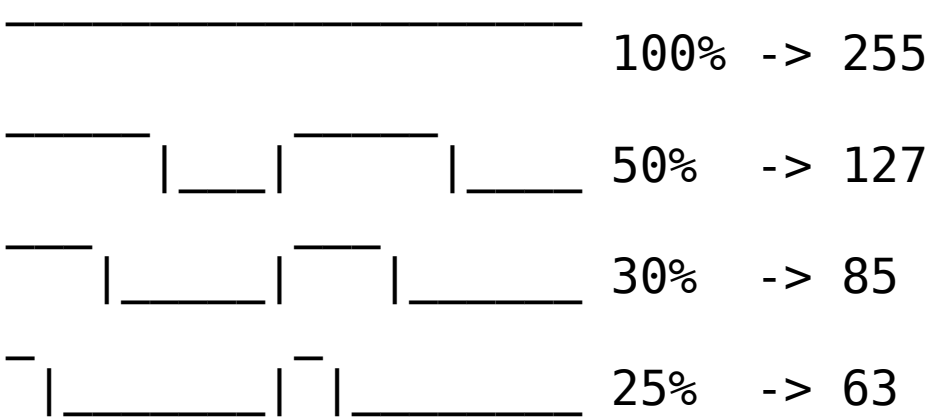
- Arduino dispone di 6 piedini digitali dedicati al PWM.
- Ciò consente di gestire i segnali in modo autonomo (*asincrono*) rispetto al programma.

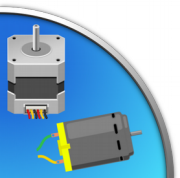
Programmazione del PWM

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%



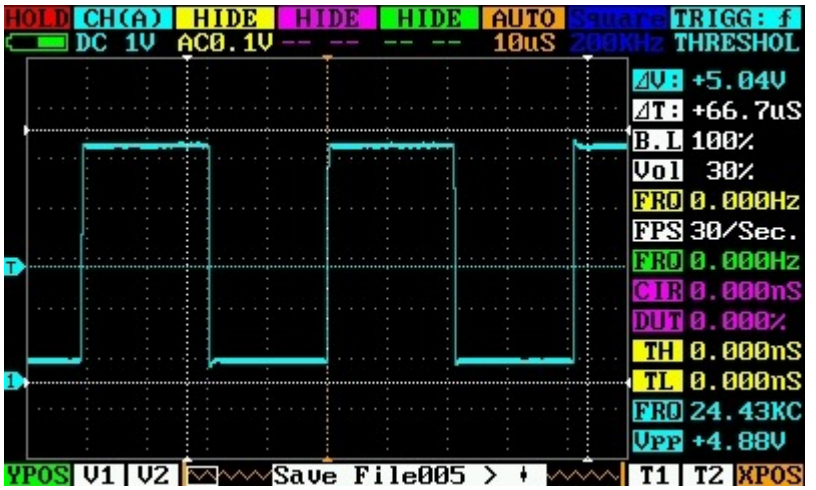


Frequenze di PWM

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

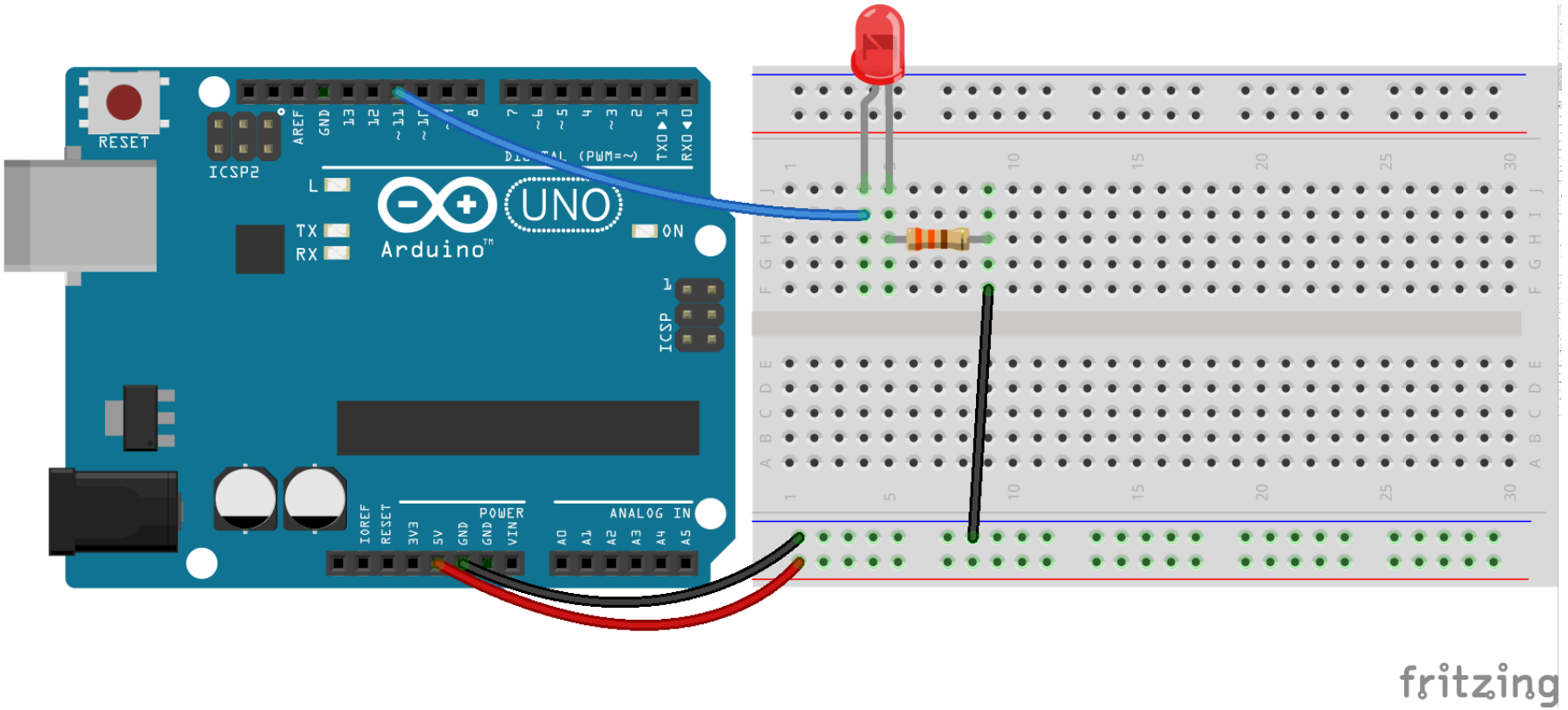
- 490 Hz per i pin 3, 9, 10, 11;
- 980 Hz per i pin 5, 6;

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



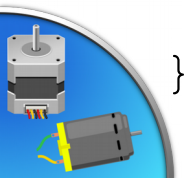
Vista all'oscilloscopio digitale

Collegamenti: Led Dimmer



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);
   }
}
```



Motori

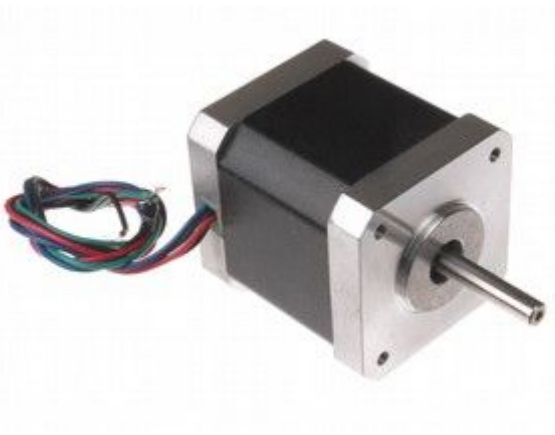
Troviamo sul mercato tre categorie di motori

Motori DC:

2 fili

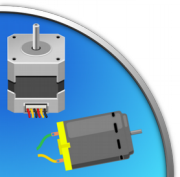
Motori stepper:

4~6 fili

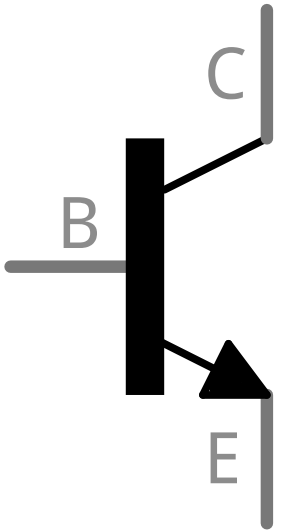


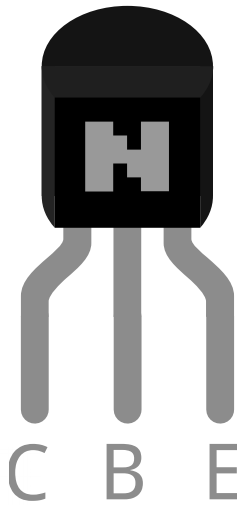
Servomotori:

3 fili



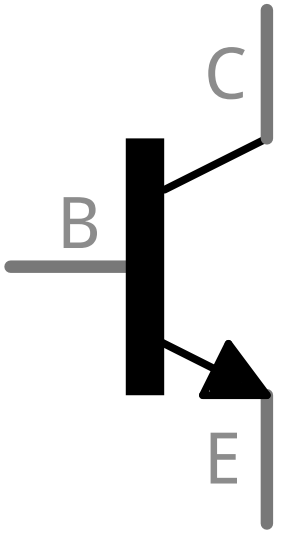
Motore DC: metodo “transistor”

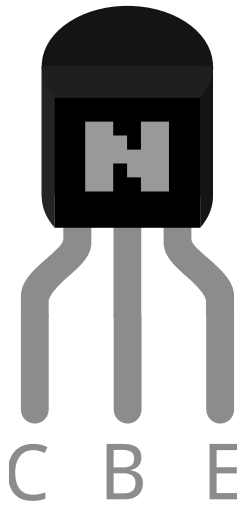


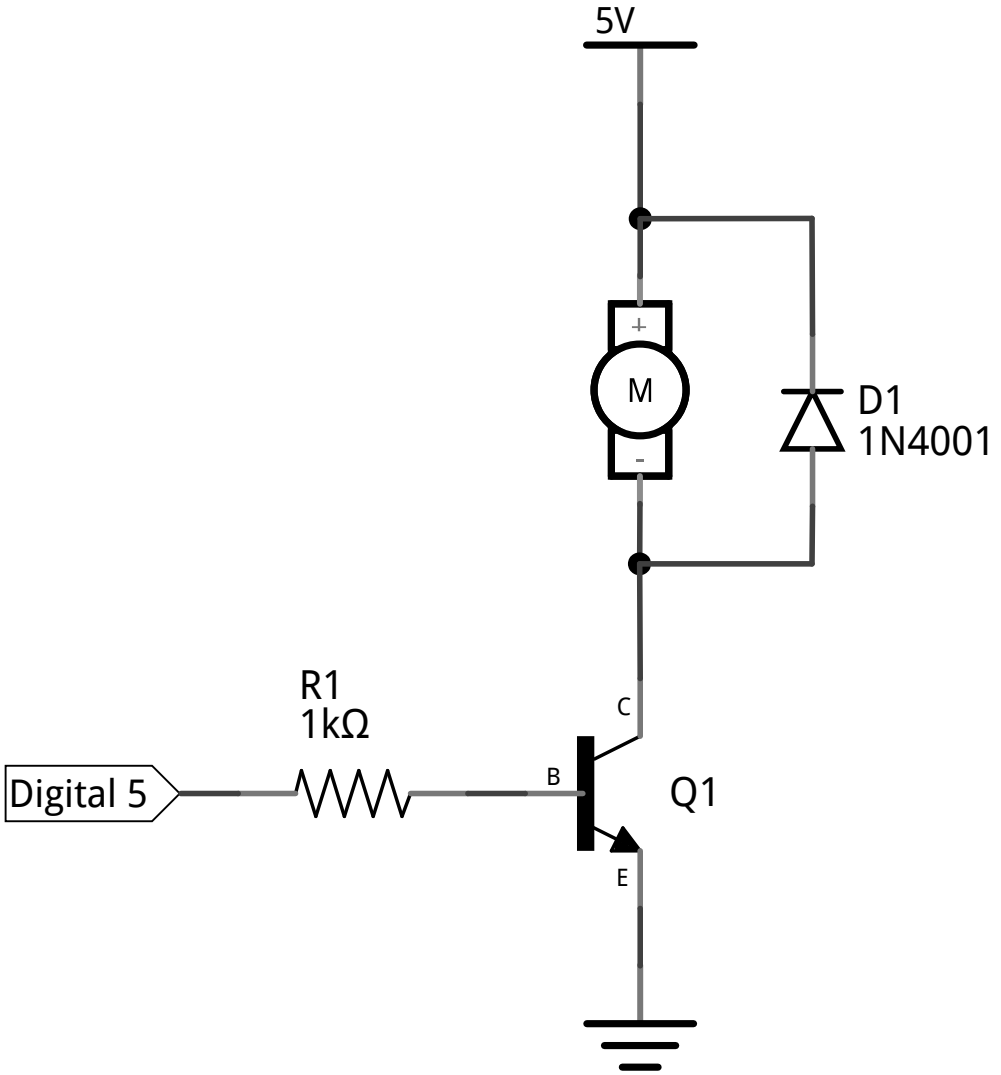


C: Collettore
B: Base
E: Emettitore

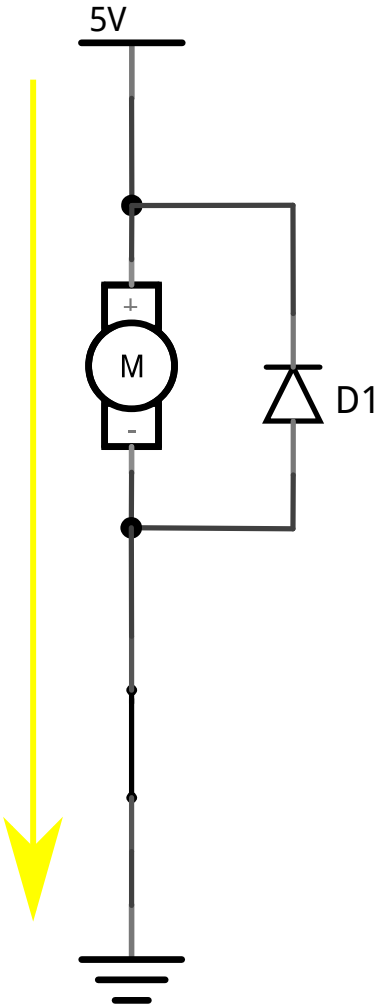
Motore DC: metodo “transistor”

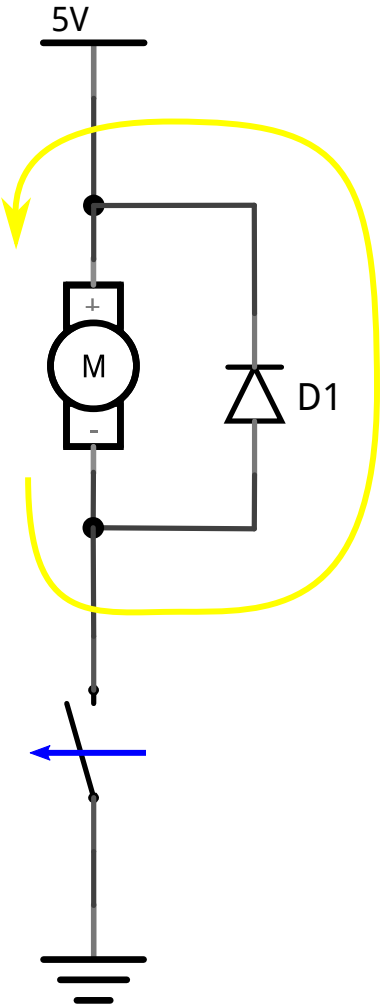


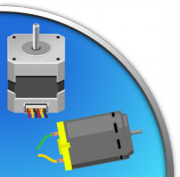




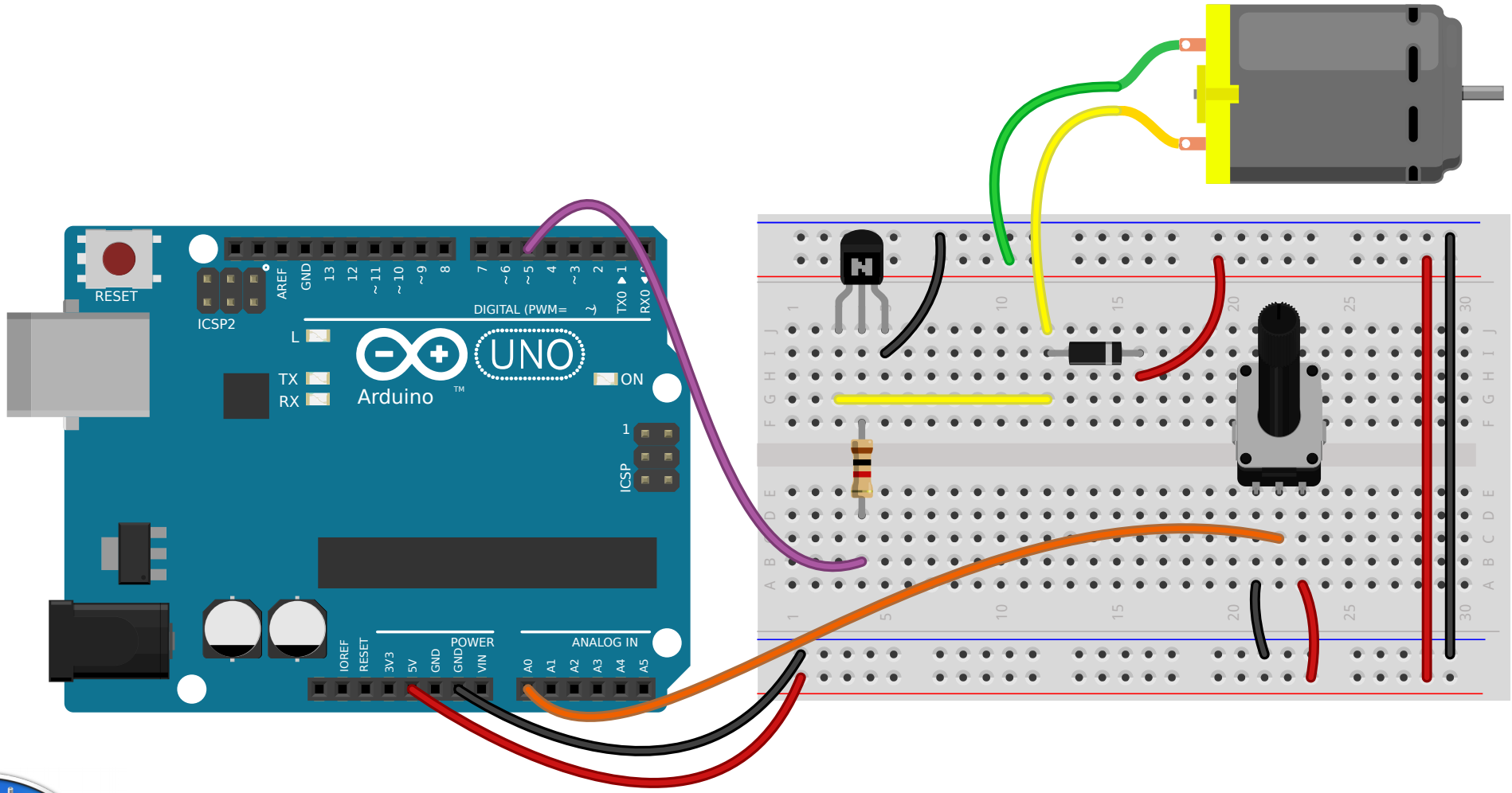
Motore DC: metodo “transistor”







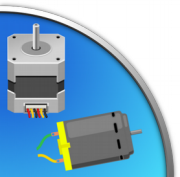
Motore DC: metodo “transistor”



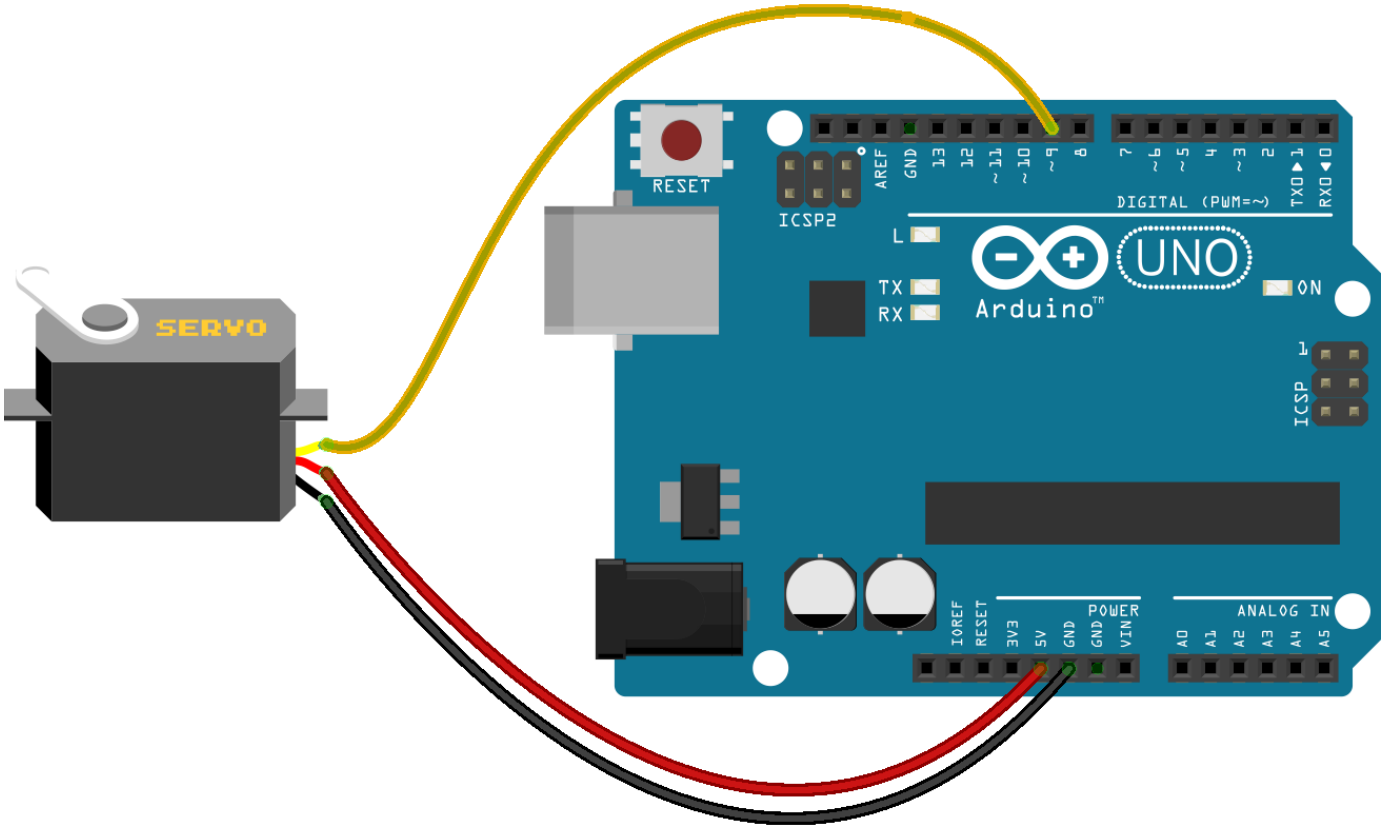
Motore DC: metodo “transistor”

```
const byte POTENZ = A0; // potenziometro
byte       MOTORE = 5;  // 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);
}
```

$$valore = \frac{lettura}{1023} \cdot 255$$

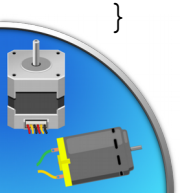


Motore Servo



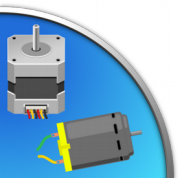
Motore Servo: listato

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



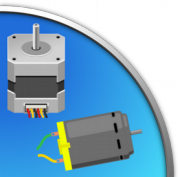
Compiti per casa - 1

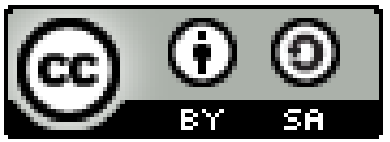
- **Crepuscolare proporzionale:** riprendi l'esercizio del crepuscolare della lezione precedente, e accendi il led con una luminosità inversamente proporzionale alla luce letta;
- **Arcobaleno:** realizza un programma che replichi i colori della scala cromatica con un LED RGB, in modo continuo;
- **Lampada colorata regolabile:** crea una piccola lampada, usando un led RGB, un pulsante e un potenziometro. Con il pulsante selezioni uno dei colori fondamentali (rosso, verde o blu), mentre con il potenziometro regoli l'intensità di quello selezionato;

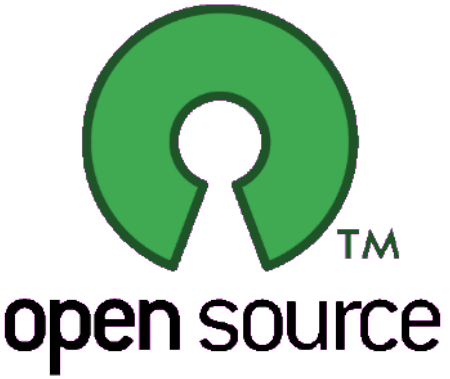


Compiti per casa - 2

- **Servo comando:** muovi il servo nella posizione letta da un potenziometro, stabilizzando le letture per evitare oscillazioni;
- **Termometro a lancetta:** utilizza il servomotore e una termoresistenza per simulare un vecchio termometro a lancetta. Puoi usare del cartoncino per realizzare una freccia ed una scala graduata;







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