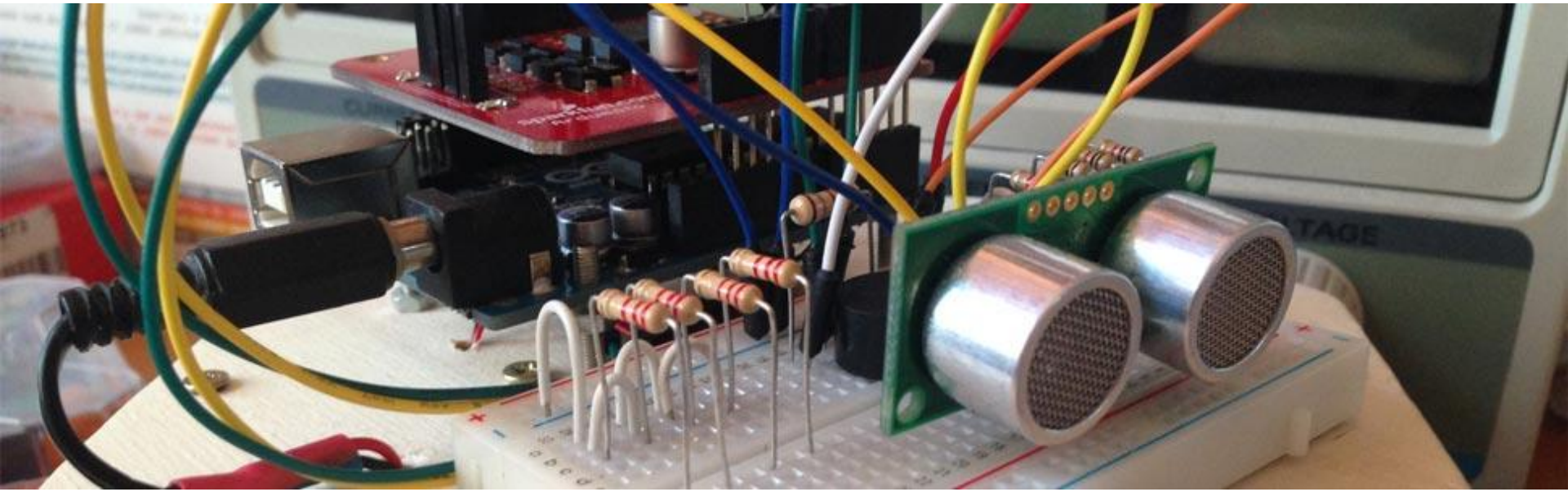
L'alfabeto di Arduino

Introduzione all'uso di Arduino
Lezione 6

Prof. Michele Maffucci



Argomenti

- Utilizzo dei servomotori
- Sperimentazioni con i Servomotori (non a rotazione continua)
  - Controllare la posizione di un servomotore
  - Controllo posizione e segnalazione
  - Controllo posizione da tastiera e segnalazione
  - Controllo posizione con un trimmer o sensore
- Sperimentazioni con i Servomotori (a rotazione continua)
  - Controllo velocità e direzione
  - Controllare la rotazione via software
- Controllare motori DC con un ponte H
  - Controllo direzione via software
  - Controllo di due motori DC via software
  - Controllo velocità e direzione di due motori via software
  - Controllo di un motore DC mediante pulsante e trimmer

Il codice e le slide utilizzate sono suscettibili di variazioni/correzioni che potranno essere fatte in ogni momento.

Introduzione

Prof. Michele Maffucci

Il seguente corso intende fornire le **competenze di base** per la realizzazione di lezioni di didattica delle robotica nella scuola secondaria di secondo grado.

Il corso ben si adatta a tutti i maker, studenti ed adulti, che per passione nell'elettronica necessitano di un'introduzione all'uso di Arduino.

Il docente che intendesse sviluppare un percorso didattico in cui si desidera realizzare dispositivi elettronici in grado di interfacciarsi col mondo fisico, potrà utilizzare queste lezioni come base per implementare moduli didattici aggiuntivi, pertanto questo corso è da intendersi come il mio personale tentativo di strutturare un percorso iniziale e modellabile a seconda del tipo di indirizzo della scuola. Chi vorrà potrà effettuare miglioramenti su quanto da me scritto.

Il percorso scelto è un estratto delle lezioni svolte durante i miei corsi di elettronica, sistemi ed impianti elettrici. Nelle slide vi sono cenni teorici di elettrotecnica che non sostituiscono in alcun modo il libro di testo, ma vogliono essere un primo passo per condurre il lettore ad un approfondimento su testi specializzati.

Il corso è basato sulla piattaforma Open Source e Open Hardware **Arduino** e fa uso dell'**Arduino starter kit**. Questa scelta non implica l'adozione di queste slide in corsi che non fanno uso di questo kit, ma è semplicemente una scelta organizzativa per lo svolgimento di questo corso di formazione. Alle proposte incluse nel kit ho aggiunto ulteriori sperimentazioni. Tutti i componenti possono essere acquistati separatamente.

Ulteriori approfondimenti e risorse a questo corso possono essere trovate sul mio sito personale al seguente link:

http://www.maffucci.it/area-studenti/arduino/

Nella sezione dedicata ad Arduino, sul mio sito personale, oltre ad ulteriori lezioni, di cui queste slide ne sono una sintesi, è possibile consultare un manuale di programmazione, in cui vengono dettagliate le istruzioni. Per rendere pratico l'utilizzo del manuale ne è stata realizzata anche una versione portable per dispositivi mobili **iOS** e **Android**, maggiori informazioni possono essere trovate seguendo il link.

Esempi utilizzati nel corso.

Tutti i programmi utilizzati nel corso possono essere prelevati al seguente link:

https://github.com/maffucci/LezioniArduino/tree/master/corso01

Gli sketch Arduino sono da scompattare nella cartella sketchbook.

E' possibile che in queste slide siano presenti delle imperfezioni, ringrazio fin d' ora chi vorrà segnalarmi correzioni e miglioramenti.

Per contatti ed ulteriori informazioni rimando alle ultime pagine di queste slide.

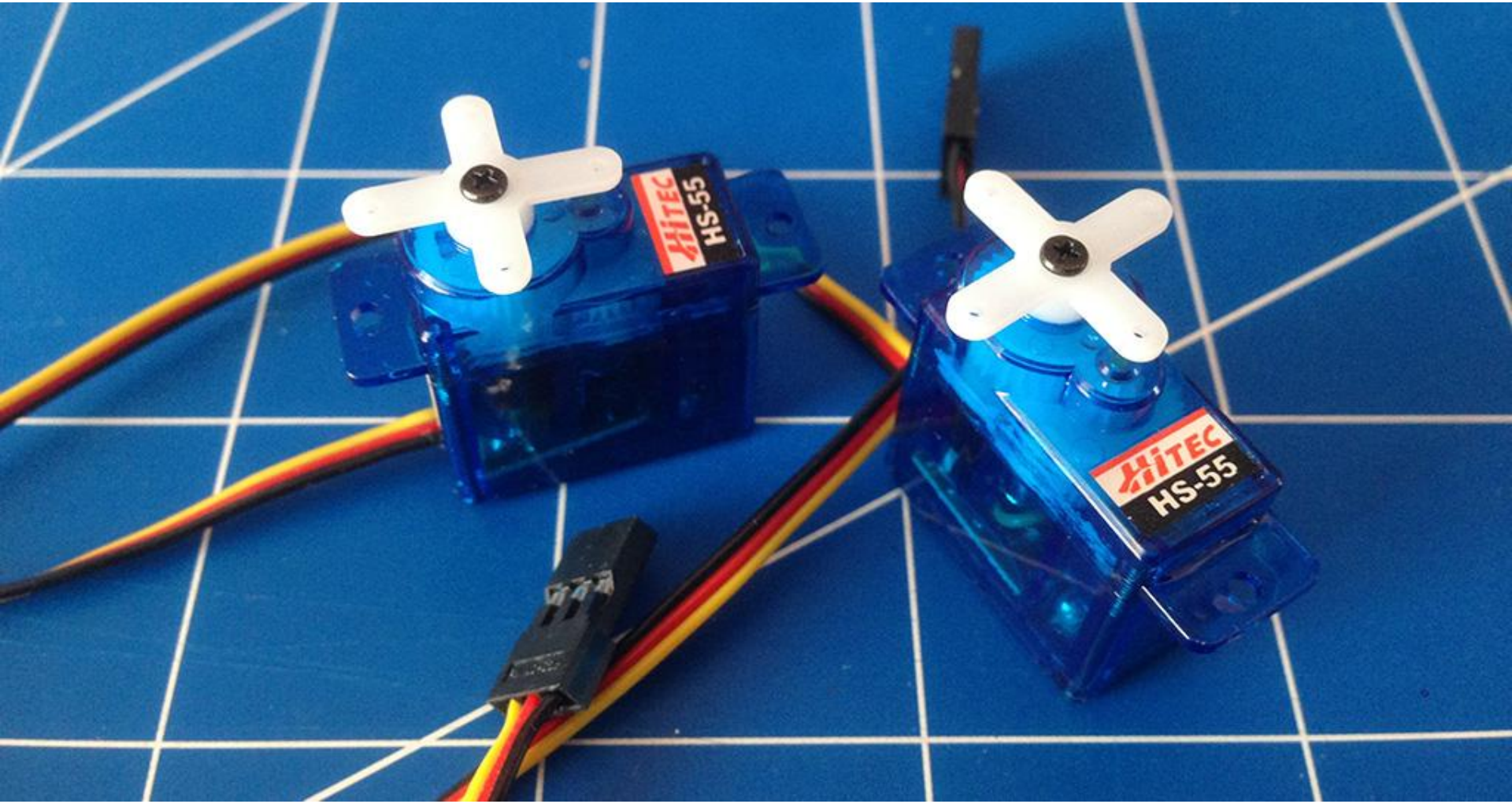
Grazie

Utilizzo dei servomotori

Prof. Michele Maffucci

servomotori

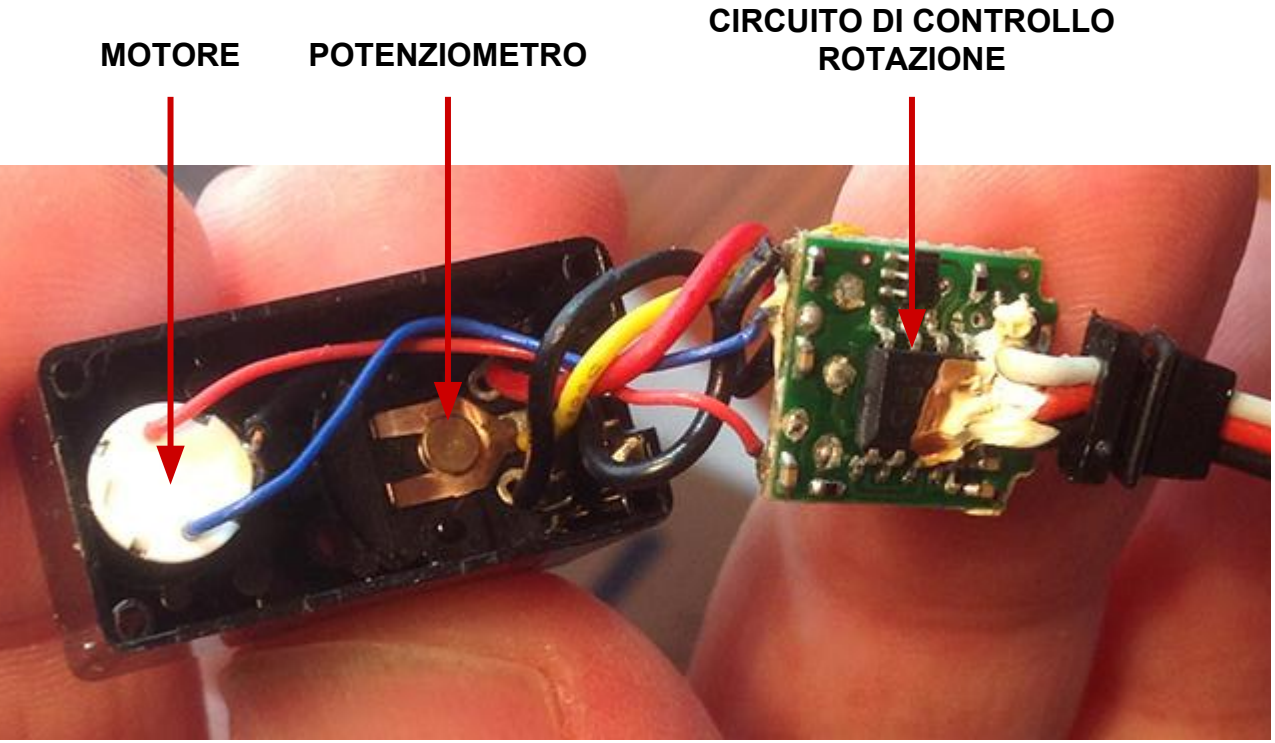
I servomotori sono degli attuatori utilizzati per far ruotare un albero di trasmissione (a cui è possibile connettere un braccio meccanico) in un intervallo fissato. Comuni sono servomotori per rotazioni da 0° a 180°, ma esistono anche servomotori a rotazione continua, da 0° a 360°, in questa lezione li analizzeremo entrambi.

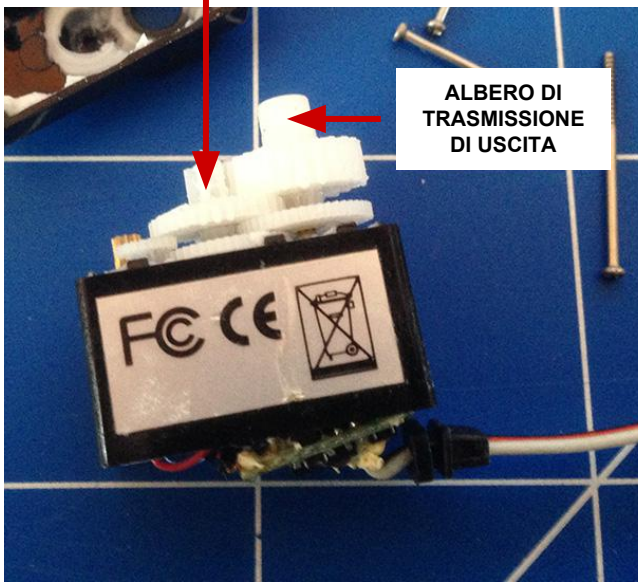


servomotori

Gli elementi che costituiscono un servomotore sono inclusi all'interno di un involucro plastico, all'interno trovano posto un motore in corrente continua connesso ad un gruppo di ingranaggi che funge da demoltiplicatore per aumentare la coppia in fase di rotazione.

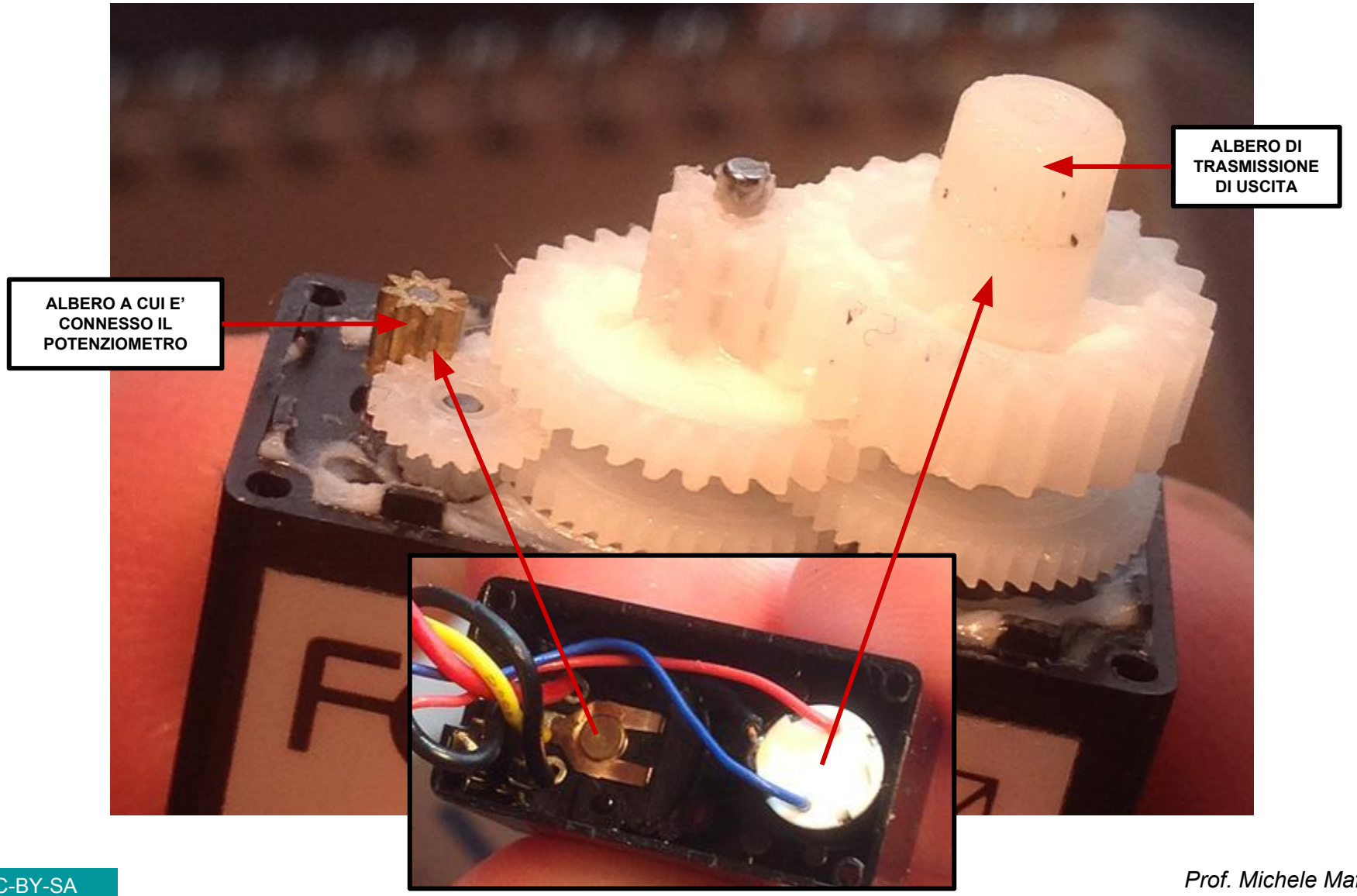
Il controllo della rotazione del motore avviene tramite un circuito a cui è connesso un potenziometro in grado di rilevare l'angolo di rotazione compiuto dal motore. Il potenziometro è connesso meccanicamente agli ingranaggi del demoltiplicatore che consente la rotazione dell'albero di trasmissione di uscita.





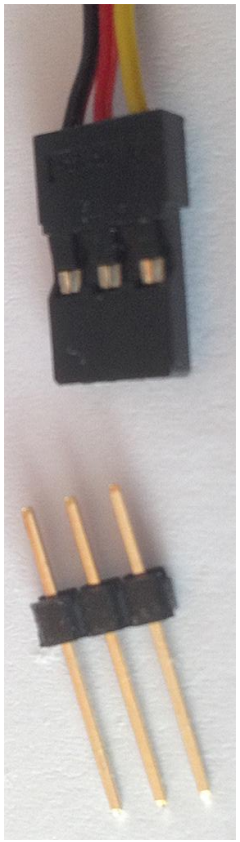
servomotori

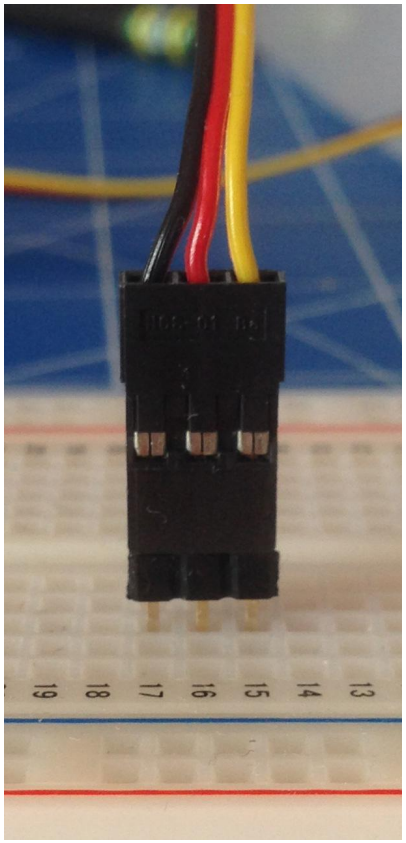
Gli ingranaggi sono in genere costituiti da perni in Nylon che possono usurarsi rapidamente. Esistono servo costituiti da materiali più resistenti o addirittura in metallo.



servomotori

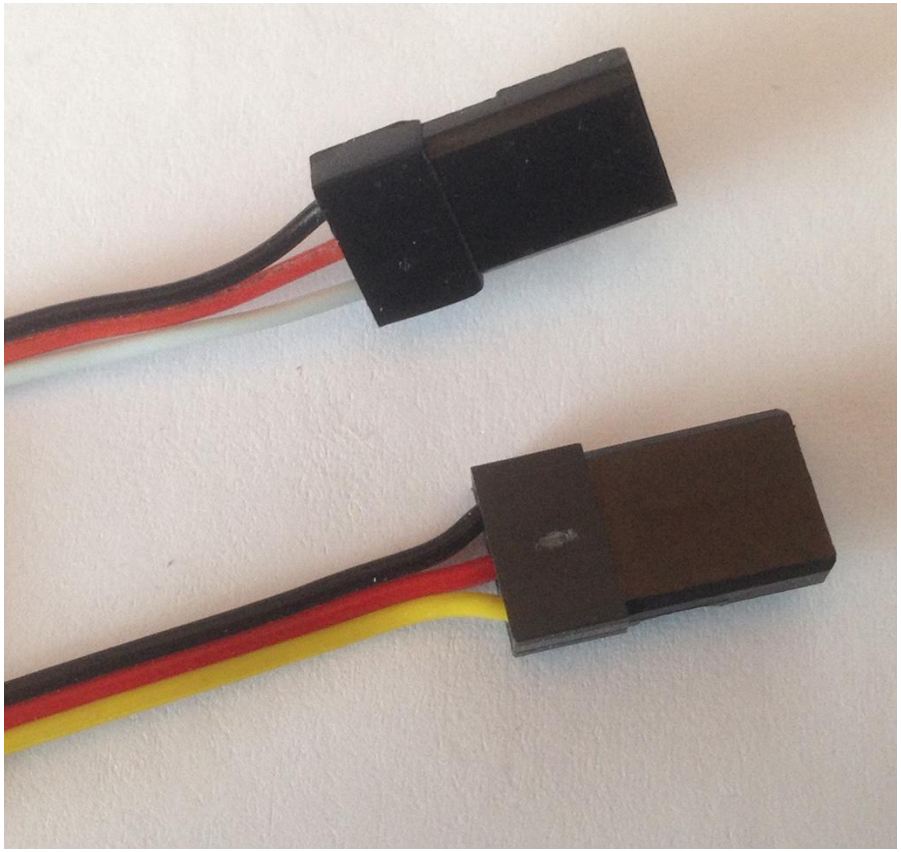
Il servomotore è inoltre costituito in genere da tre cavi connessi ad un connettore femmina con passo standard tra i fori di 2,54 mm quindi facilmente utilizzabile con qualsiasi strip che ne permette il collegamento ad esempio su una breadboard.





I fili di connessione possono assumere colori diversi in funzione della marca del servo.
Le funzionalità dei pin sono:

- Filo **ROSSO**: +V
- Filo **NERO** o **MARRONE**: GND
- Filo **BIANCO** o **ARANCIO** o **BIANCO** o **BLU**: Segnale

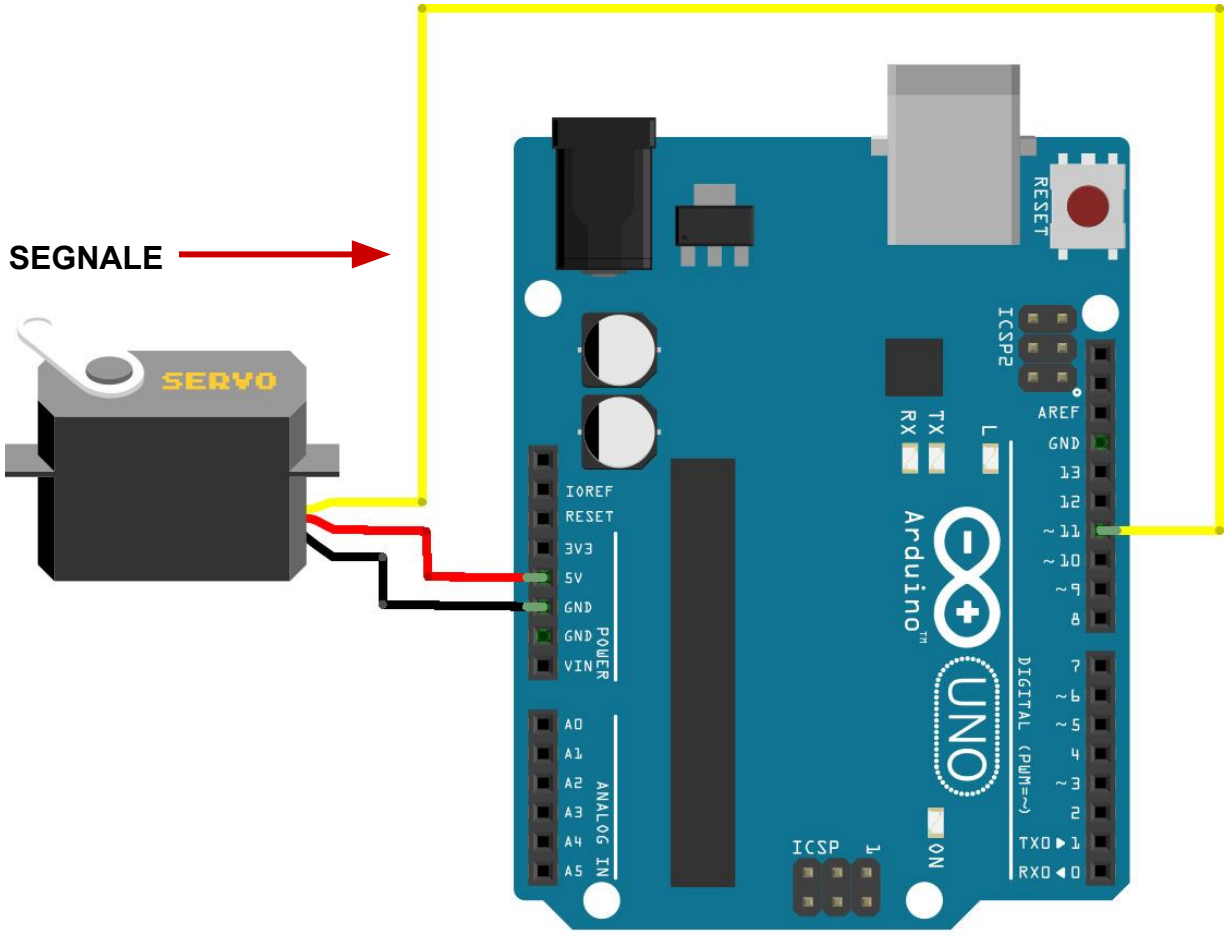


servomotori

Il segnale utilizzato per il controllo di rotazione da parte di un circuito esterno ad es. Arduino.

Le tensioni di alimentazione per i servomotori da modellismo variano tra i 4,8 V e i 6 V, tensione ottimale 5 V (fate riferimento al datasheet del servo che possedete).
Alla massima tensione corrisponde la massima potenza.

In commercio esistono servomotori alimentati anche a tensioni superiori.

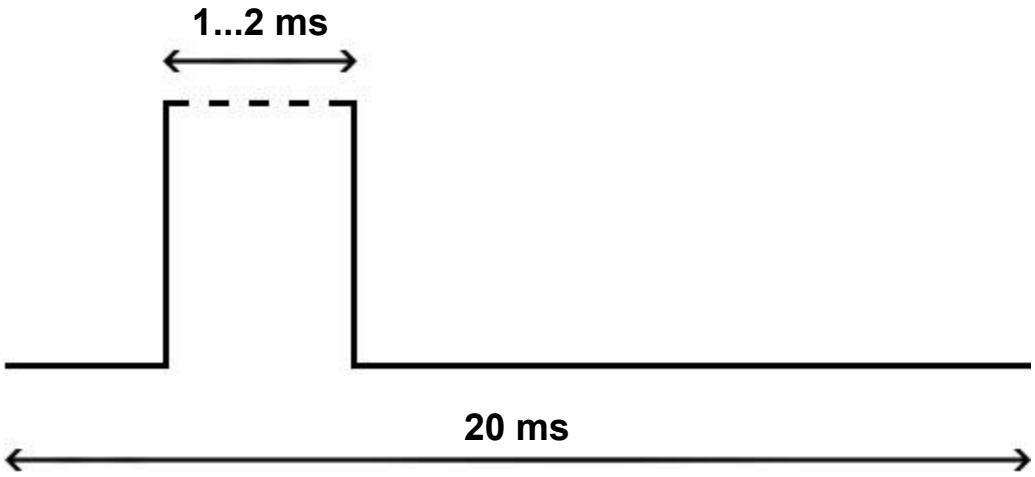


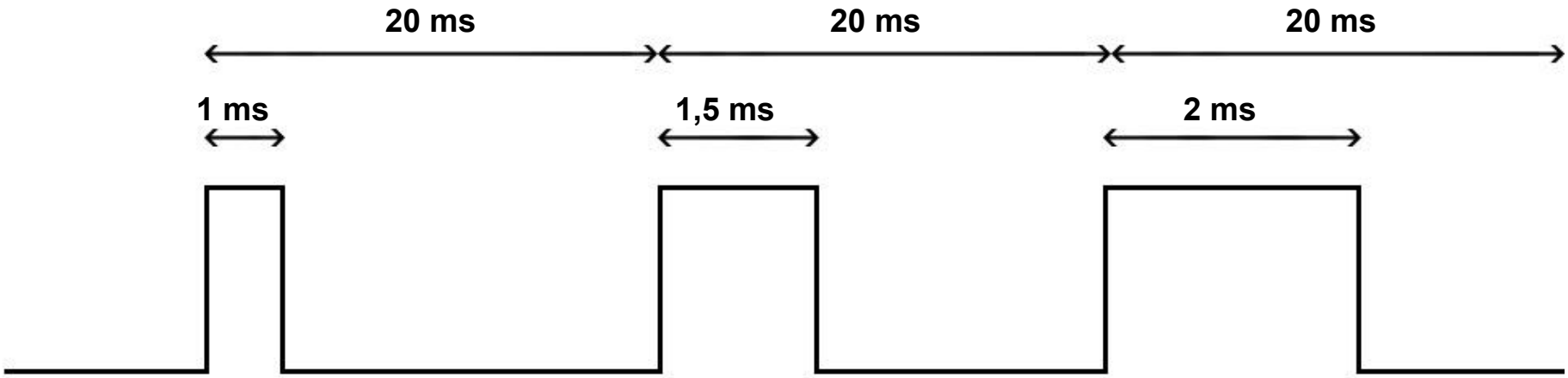
Prof. Michele Maffucci

Il segnale di controllo è costituita da un'onda quadra che viene inviata ripetutamente dal circuito esterno (Arduino) e la sua ampiezza definisce l'angolo di rotazione dell'albero di trasmissione.

Viene utilizzata la tecnica della modulazione di ampiezza d'impulso, il PWM (**P**ulse **W**ave **M**odulation) di cui si è parlato in **Alfabeto di Arduino - lezione 2** (dalla slide 82 in avanti) e sul mio sito personale lezione: **Arduino – lezione 06: modulazione di larghezza di impulso (PWM)**.

Per poter pilotare un servomotore, il circuito di controllo (ad es. Arduino) dovrà essere in grado di trasmettere al servomotore 50 impulsi al secondo, cioè un impulso ogni 20 ms (1 sec/50 impulsi = 20 ms), ovvero il controllo della rotazione dell'albero di trasmissione avviene mediante impulsi di lunghezza variabile che distano l'uno dall'altro 20 ms.





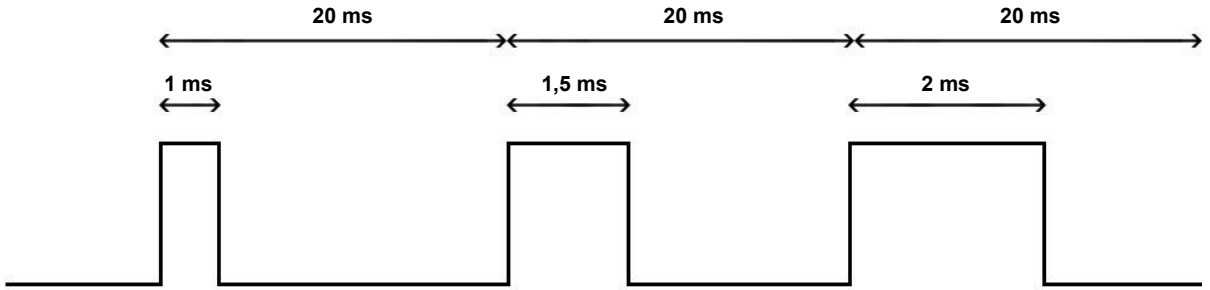
servomotori

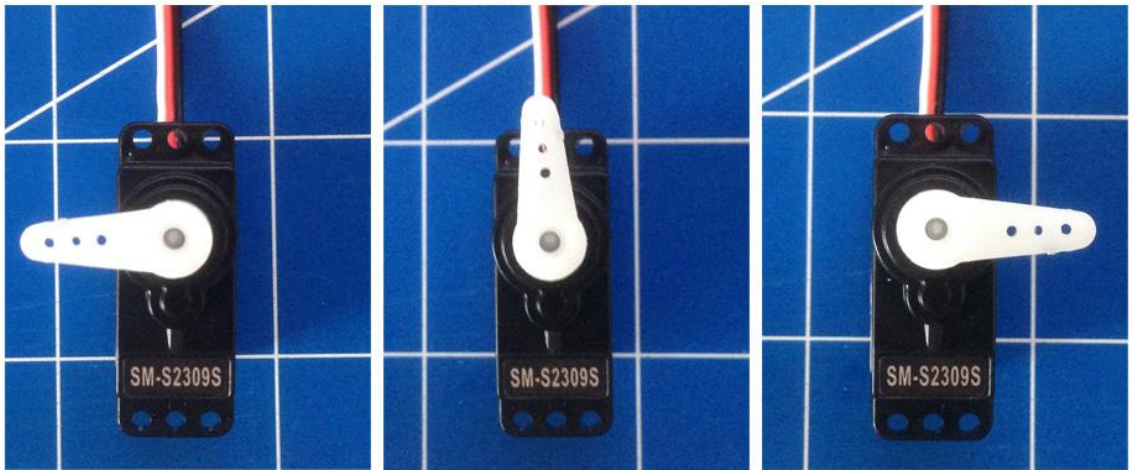
La rotazione del braccio seguirà questa regola:

- Un impulso di 1 ms (o meno) fa ruotare il servomotore ad una estremità.
- Un impulso di 2 ms fa ruotare il servomotore all'estremità opposto, cioè di 180°
- Un impulso di 1,5 ms fa ruotare il servomotore di 90°

Gli impulsi di durata compresa tra 1 ms e 2 ms fanno ruotare il servomotore di un angolo proporzionale alla durata dell'impulso.

Quindi la durata esatta di un impulso definisce l'angolo di rotazione del servomotore.





Sperimentazioni con i Servomotori
(non a rotazione continua)

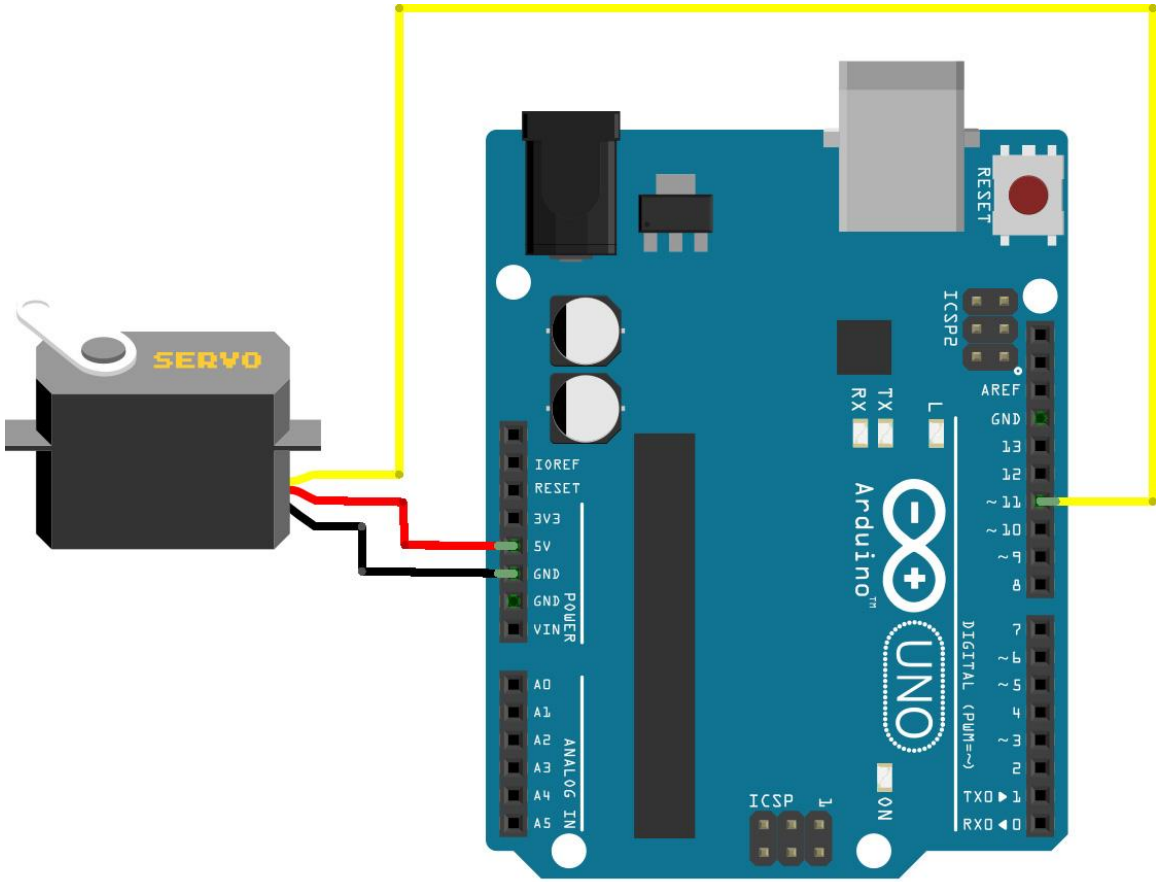
Prof. Michele Maffucci

Controllare la posizione di un servomotore 1/2 sketch39

Controlliamo la posizione del servomotore utilizzando un angolo calcolato all'interno dello sketch.

Collegare il servomotore ai piedini di alimentazione +5V e GND ed utilizzare come pin di controllo il pin 11 di Arduino.

La versione più recente della libreria consente di collegare il pin del segnale del servo a qualsiasi pin digitale di Arduino.



Nota

La libreria servo permette di gestire al massimo 12 servomotori su gran parte delle schede Arduino, sulla Mega invece possono essere gestiti fino a 48 servomotori.

Attenzione

Sulle schede Arduino, esclusa la Mega l'uso della libreria disabilita la funzionalità analogWrite() (quindi l'uso del PWM) sui pin 9 e 10 anche se su questi pin non sono collegati dei servo.

Sul mega possono essere utilizzati 12 servo senza interferire con la funzionalità del PWM.
L'utilizzo di servomotori da 12 a 23 disabilita la funzione PWM sui pin 11 e 12.

Controllare la posizione di un servomotore 2/2 sketch39

```
/* Prof. Michele Maffucci
   18.04.2014

   Utilizzo di un servomotore
   Modello: SM-S2309S da 9.9g
   Angolo di rotazione: +/- 60 gradi
   Momento torcente a 4,8 V: 1.1 kg-cm
   Momento torcente a 6 V: 1.3 kg-cm
*/

#include <Servo.h>

Servo myservo;  // crea un oggetto Servo per controllare un servomotore

int angolo = 0;      // variabile dove conservare la posizione del dervo
int angoloMax = 120; // angolo massimo di rotazione del servo
                     // il servo utilizzato pu ruotare da -60 a +60 gradi
                     // Variare angoloMax secondo il servo posseduto

void setup()
{
  myservo.attach(11);  // collega il servo al pin 11 all'oggetto servo
}


void loop()
{
  for(angolo = 0; angolo < angoloMax; angolo += 1)  // va da 0 a 120 gradi
  {                                                 // in passi di 1 grado
                                                    // viene detto al servo di andare
    myservo.write(angolo);                          // alla posizione indicata da "angolo"
    delay(20);                                      // aspetta 20ms tra i comandi impartiti al servo
  }
  for(angolo = angoloMax; angolo >= 1; angolo -= 1) // va da 120 a 0 gradi
  {
                                                    // viene detto al servo di andare
                                                    // alla posizione indicata da "angolo"
    myservo.write(angolo);
    delay(20);                                      // aspetta 20ms tra i comandi impartiti al servo
  }
}
```

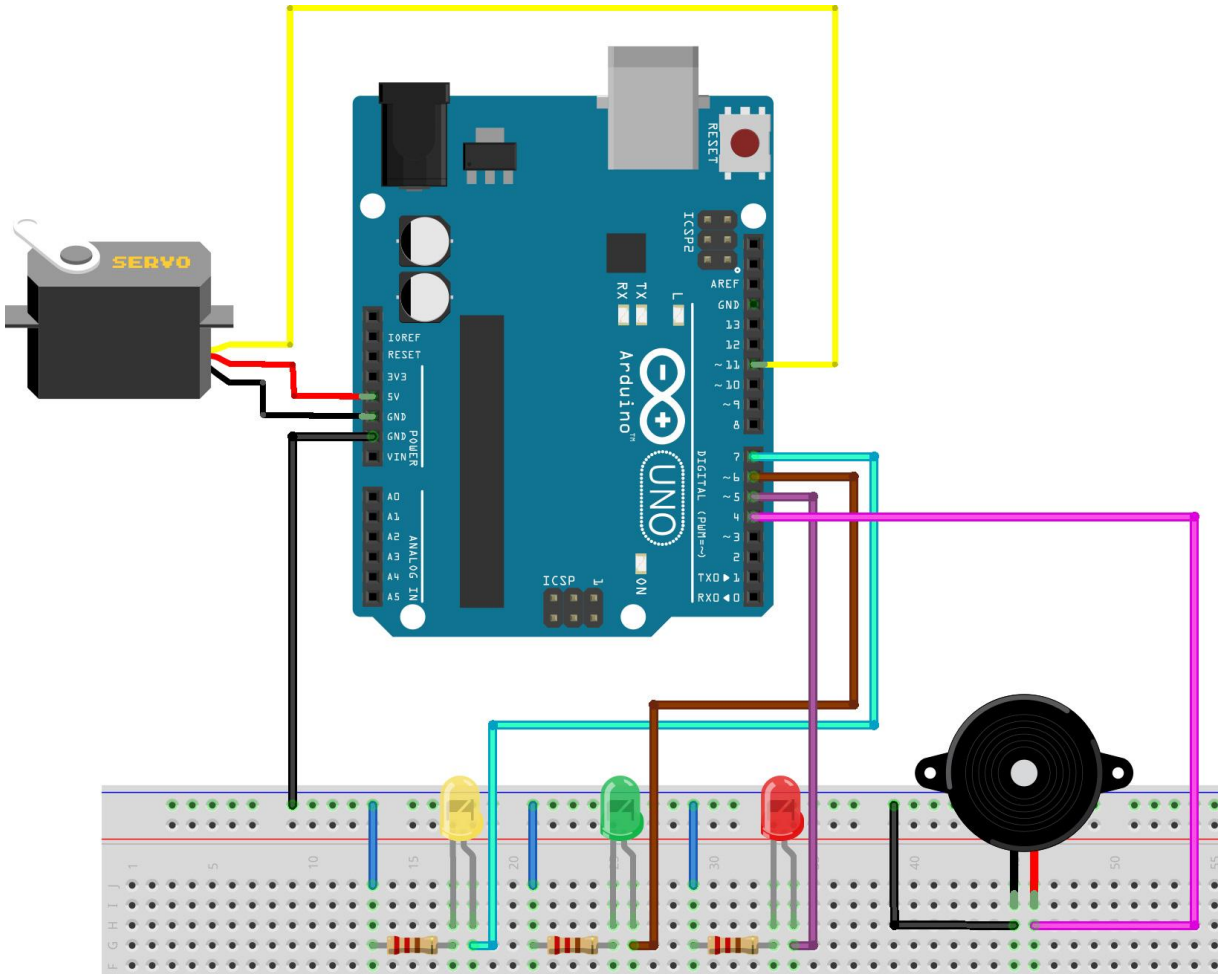
Non tutti i servo si muovono lungo l' intervallo 0 - 180 gradi, nel caso del servo utilizzato in questa esercitazione il movimento avviene in un intervallo compreso tra -60 e +60 gradi.

Controllo posizione e segnalazione

Controlliamo la posizione del servomotore utilizzando un angolo calcolato all'interno dello sketch, rilevare la posizione raggiunta mediante segnalazione visiva e sonora.

Collegamenti:

- pin segnale servo: 11
- pin LED giallo: 7
- pin LED verde: 6
- pin LED rosso: 5
- pin buzzer: 4



Si utilizzi 3 LED per la segnalazione:

- posizione 0° - LED giallo
- posizione 60° - LED verde
- posizione 120° - LED rosso

Al raggiungimento delle tre posizioni deve essere emessa una nota diversa:

- posizione 0° - LA
- posizione 60° - DO
- posizione 120° - RE

Il controllo dell'angolo raggiunto e della nota da emettere deve essere realizzato con due funzioni esterne richiamate all'interno del loop.

La nota emessa e la sua durata devono essere variabili globali.

Controllo posizione e segnalazione 2/4 sketch40

```
/* Prof. Michele Maffucci
   18.04.2014

   Utilizzo di un servomotore
   Modello: SM-S2309S da 9.9g
   Angolo di rotazione: +/- 60 gradi
   Momento torcente a 4,8 V: 1.1 kg-cm
   Momento torcente a 6 V: 1.3 kg-cm

   La segnalazione del raggiungimento dell'angolo:
   minimo, medio e massimo avvenire mediante segnalazione
   visiva (con tre LED di colore diverso)
   e sonora (con nota divers)

*/

#include <Servo.h>

Servo myservo;          // crea un oggetto Servo per controllare un servomotore

int angolo = 0;         // variabile dove conservare la posizione del dervo
int angoloMax = 120;    // angolo massimo di rotazione del servo
                        // il servo utilizzato pu ruotare da -60 a +60 gradi
                        // Variare angoloMax secondo il servo posseduto

int ledMassimo = 5;     // pin a cui collegato il led che segnala l'angolo massimo
int ledMedio = 6;       // pin a cui collegato il led che segnala l'angolo medio
int ledMinimo = 7;      // pin a cui collegato il led che segnala l'angolo minimo

int pinBuzzer = 4;      // pin a cui collegato il buzzer
int notaLA = 440;       // frequenza della nota LA
int durataNotaLA = 200; // durata (in millisecondi) della durata della nota

int notaDO = 262;       // frequenza della nota DO
int durataNotaDO = 200; // durata (in millisecondi) della durata della nota

int notaRE = 294;       // frequenza della nota RE
int durataNotaRE = 200; // durata (in millisecondi) della durata della nota
```

continua...

Controllo posizione e segnalazione


```
void setup()
{
  myservo.attach(11);           // collega il servo al pin 11 all'oggetto servo
  pinMode(ledMassimo, OUTPUT);  // ledMassimo definito come OUTPUT
  pinMode(ledMedio, OUTPUT);    // ledMedio definito come OUTPUT
  pinMode(ledMinimo, OUTPUT);   // ledMinimo definito come OUTPUT
  pinMode(pinBuzzer, OUTPUT);   // pinBuzzer definito come OUTPUT
}
void loop()
{
  for (angolo = 0; angolo < angoloMax; angolo += 1) // va da 0 a 120 gradi
  {                                                 // in passi di 1 grado
                                                    // viene detto al servo di andare
    myservo.write(angolo);                          // alla posizione indicata da "angolo"
    verificaAngolo(angolo);
    delay(20);                                      // aspetta 20ms tra i comandi impartiti al servo
  }

  for (angolo = angoloMax; angolo >= 1; angolo -= 1)  // va da 120 a 0 gradi
  {                                                   // viene detto al servo di andare
                                                      // alla posizione indicata da "angolo"
    myservo.write(angolo);
    verificaAngolo(angolo);
    delay(20);                                        // aspetta 20ms tra i comandi impartiti al servo
  }
}
```

continua...

Controllo posizione e segnalazione

sketch40

```
// la funzione verificaAngolo riceve come input un "angoloRaggiunto" e in funzione di questo
// accende il LED ed emette la nota corrispondente al valore che gli  stato passato

void verificaAngolo(int angoloRaggiunto) {
  switch (angoloRaggiunto) {
    case 0:
      digitalWrite(ledMinimo, HIGH);
      digitalWrite(ledMedio, LOW);
      digitalWrite(ledMassimo, LOW);
      suona(notaLA, durataNotaLA);
      break;
    case 60:
      digitalWrite(ledMinimo, LOW);
      digitalWrite(ledMedio, HIGH);
      digitalWrite(ledMassimo, LOW);
      suona(notaDO, durataNotaDO);
      break;
    case 120:
      digitalWrite(ledMinimo, LOW);
      digitalWrite(ledMedio, LOW);
      digitalWrite(ledMassimo, HIGH);
      suona(notaRE, durataNotaRE);
      break;
  }
}

// La funzione suona emette la nota "notaEmessa" per la durata "durataNota"

void suona(int notaEmessa, int durataNota) {
  tone(pinBuzzer, notaEmessa, durataNota);
}
```

Controllo posizione da tastiera e segnalazione 1/4 sketch41

Mantenendo le specifiche dell'esercizio precedente realizzzare uno sketch che permetta di controllare da tastiera mediante i tasti: "1" e "2" l'avanzamento ed il decremento di 20° di rotazione del servo.

- pressione tasto "1": incremento di 20 gradi;
- pressione tasto "2": decremento di 20 gradi.

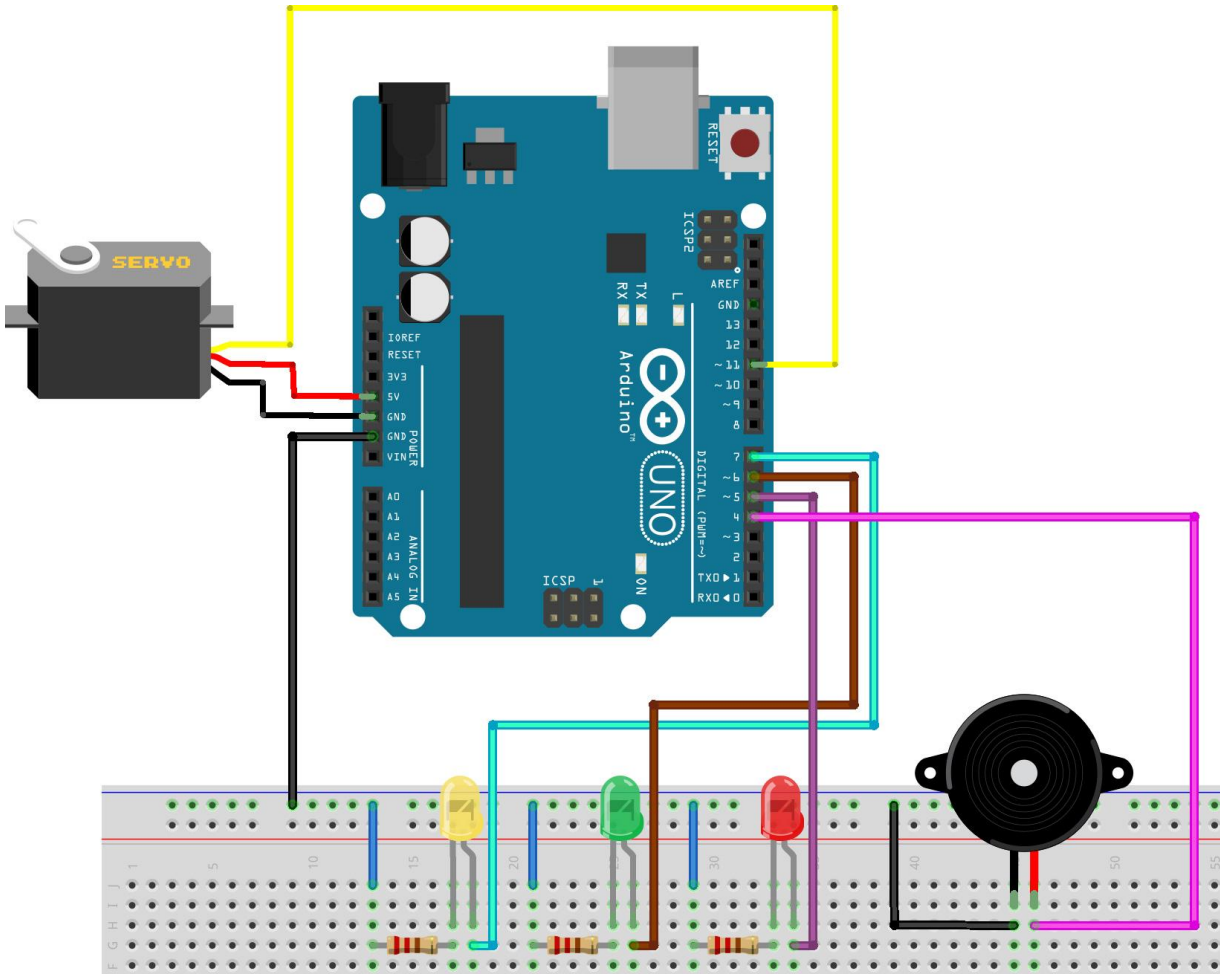
La posizione di partenza del servo è a 0 gradi.

Raggiunto uno dei 3 limiti (0°, 60°, 120°) dell' intervallo di rotazione deve accendersi il LED corrispondente e deve essere emessa la nota corrispondente alla posizione.

Superata una delle 3 posizioni limite il LED corrispondente si deve spegnere ed il suono non deve essere più emesso.

Superato il limite di 120°, le successive pressioni del tasto "2" non provocano nessun movimento in avanti.

Superato il limite inferiore di 0 gradi, successioni pressioni del tasto "1" non provocano nessun movimento all'indietro.



Controllo posizione da tastiera e segnalazione 2/4 sketch41

```
/* Prof. Michele Maffucci
   18.04.2014

   Utilizzo di un servomotore
   Modello: SM-S2309S da 9.9g
   Angolo di rotazione: +/- 60 gradi
   Momento torcente a 4,8 V: 1.1 kg-cm
   Momento torcente a 6 V: 1.3 kg-cm

   La segnalazione del raggiungimento dell'angolo:
   minimo, medio e massimo avvenire mediante segnalazione
   visiva (con tre LED di colore diverso)
   e sonora (con nota divers)

*/

#include <Servo.h>

Servo myservo;          // crea un oggetto Servo per controllare un servomotore

int angolo = 0;         // variabile dove conservare la posizione del dervo
int angoloMax = 120;    // angolo massimo di rotazione del servo
                        // il servo utilizzato pu ruotare da -60 a +60 gradi
                        // Variare angoloMax secondo il servo posseduto

int ledMassimo = 5;     // pin a cui collegato il led che segnala l'angolo massimo
int ledMedio = 6;       // pin a cui collegato il led che segnala l'angolo medio
int ledMinimo = 7;      // pin a cui collegato il led che segnala l'angolo minimo

int pinBuzzer = 4;      // pin a cui collegato il buzzer
int notaLA = 440;       // frequenza della nota LA
int durataNotaLA = 200; // durata (in millisecondi) della durata della nota

int notaDO = 262;       // frequenza della nota DO
int durataNotaDO = 200; // durata (in millisecondi) della durata della nota

int notaRE = 294;       // frequenza della nota RE
int durataNotaRE = 200; // durata (in millisecondi) della durata della nota

byte posizione = 0;         // variabile per posizione corrente del servo
byte controlloLimite = 0;   // variabile per il controllo del raggiungimento di uno dei limiti (0, 60, 120)
```

continua...

Controllo posizione da tastiera e segnalazione 3/4 sketch41

```
void setup()
{
  myservo.attach(11);           // collega il servo al pin 11 all'oggetto servo
  Serial.begin(9600);           // inizializzazione della porta seriale
  pinMode(ledMassimo, OUTPUT);  // ledMassimo definito come OUTPUT
  pinMode(ledMedio, OUTPUT);    // ledMedio definito come OUTPUT
  pinMode(ledMinimo, OUTPUT);   // ledMinimo definito come OUTPUT
  pinMode(pinBuzzer, OUTPUT);   // pinBuzzer definito come OUTPUT
}

void loop()
{
  if (Serial.available())
  {
    byte passo = Serial.read();

    // fino a quando non si raggiunge l'angolo minimo, premendo "1" viene decrementata l'angolo
    if (passo == '1') // premendo '1' decremento la variabile fino al valore minimo 20
    {
      if (posizione >= 20) // fino a quando la posizione maggiore di 20 gradi puo' diminuire
      {
        posizione = posizione - 20;
        Serial.println("Mi sono spostato di 20 gradi indietro");
        controllaPosizione(posizione);
      }
    }

    // fino a quando non si raggiunge l'angolo massimo, premendo "2" viene incrementato l'angolo
    else if (passo == '2') // premendo '2' aumento la variabile fino al valore massimo 120
    {
      if (posizione < 120) // fino a quando la posizione minore di 120 gradi puo' diminuire
      {
        posizione = posizione + 20;
        Serial.println("Mi sono spostato di 20 gradi in avanti");
        controlloLimite = posizione;
        controllaPosizione(controlloLimite);
      }
    }
  }
  myservo.write (posizione);
}
```

continua...

Controllo posizione da tastiera e segnalazione 4/4 sketch41

```
// la funzione verificaAngolo riceve come input un "angoloRaggiunto" e in funzione di questo
// accende il LED ed emette la nota corrispondente al valore che gli  stato passato

void controllaPosizione(int angoloRaggiunto) {
  switch (angoloRaggiunto) {
    case 0:                                 // controllo raggiungimento 0 gradi
      digitalWrite(ledMinimo, HIGH);
      digitalWrite(ledMedio, LOW);
      digitalWrite(ledMassimo, LOW);
      suona(notaLA, durataNotaLA);
      controlloLimite=0;
      break;
    case 60:                                // controllo raggiungimento 60 gradi
      digitalWrite(ledMinimo, LOW);
      digitalWrite(ledMedio, HIGH);
      digitalWrite(ledMassimo, LOW);
      suona(notaDO, durataNotaDO);
      controlloLimite=0;
      break;
    case 120:                               // controllo raggiungimento 120 gradi
      digitalWrite(ledMinimo, LOW);
      digitalWrite(ledMedio, LOW);
      digitalWrite(ledMassimo, HIGH);
      suona(notaRE, durataNotaRE);
      controlloLimite=0;
      break;
      default:
      digitalWrite(ledMinimo, LOW);
      digitalWrite(ledMedio, LOW);
      digitalWrite(ledMassimo, LOW);
  }
}

// La funzione suona emette la nota "notaEmessa" per la durata "durataNota"

void suona(int notaEmessa, int durataNota) {
  tone(pinBuzzer, notaEmessa, durataNota);
}
```

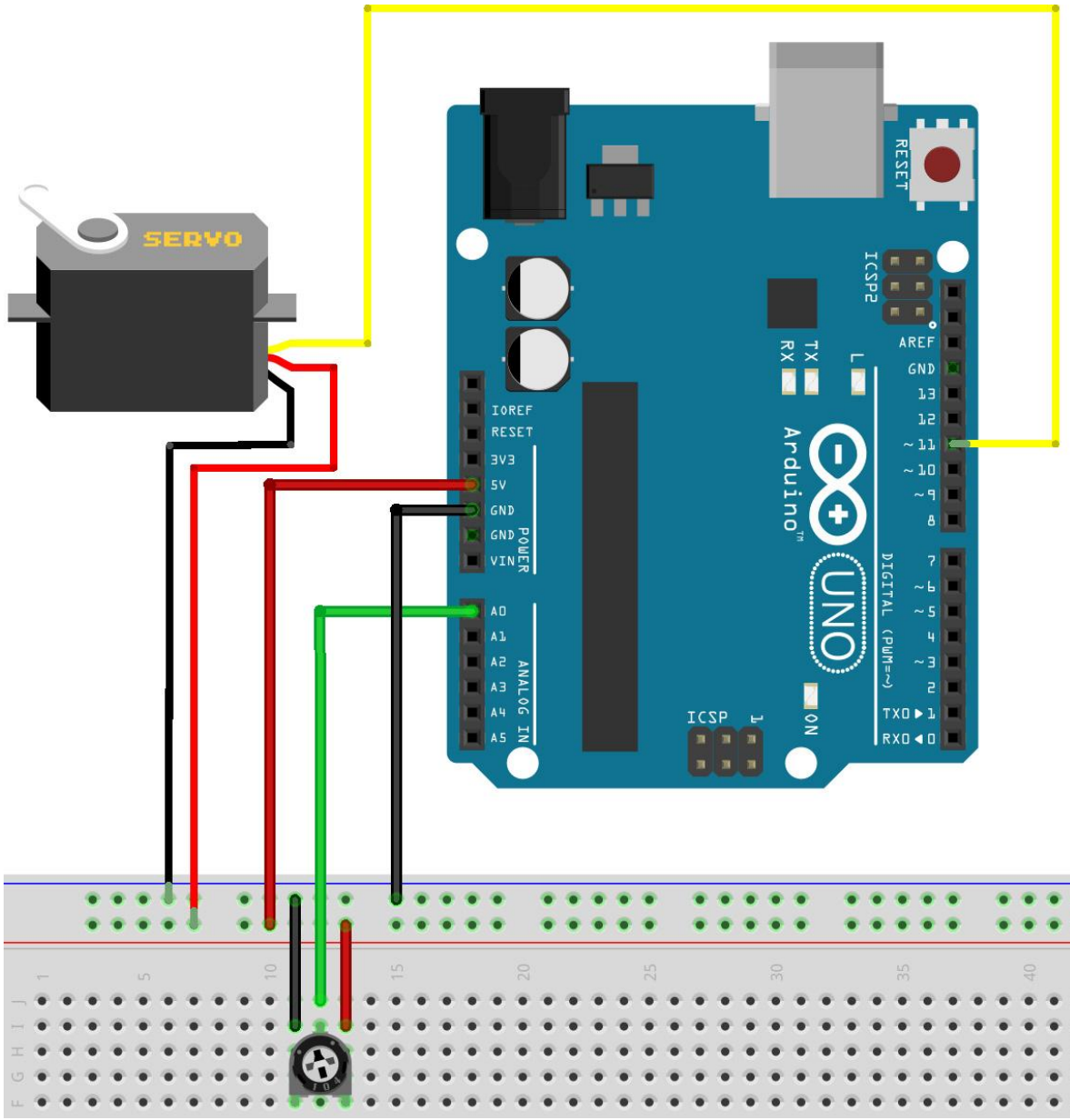
Controllo posizione con un trimmer o sensore

sketch42

Si vuole controllare la direzione e la velocità di rotazione di un servomotore mediante un trimmer.

Si utilizzerà un codice simile allo sketch39, ma bisognerà aggiungere il codice che permette di leggere la tensione in ingresso sul pin A0 impostata da un trimmer.

Il valore analogico in ingresso dovrà essere rimappato da un intervallo (0, 1023) su un valore compreso tra 0 e 120 gradi, corrispondente all'escursione del servo utilizzato in queste sperimentazioni.



Approfondimento

Si provi ad utilizzare al posto del trimmer un qualsiasi sensore la cui grandezza misurata consenta la rotazione del servomotore.

Prof. Michele Maffucci

Controllo posizione con un trimmer o sensore 2/2 sketch42

```
/* Prof. Michele Maffucci
   18.04.2014

   Controllo rotazione servo con trimmer

   Utilizzo di un servomotore
   Modello: SM-S2309S da 9.9g
   Angolo di rotazione: +/- 60 gradi
   Momento torcente a 4,8 V: 1.1 kg-cm
   Momento torcente a 6 V: 1.3 kg-cm

*/

#include <Servo.h>

Servo myservo;  // crea un oggetto Servo

int pinTrimmer = 0;  // pin analogico a cui viene connesso il trimmer
int valore;          // variabile usata per memorizzare il valore di ingresso

void setup()
{
  myservo.attach(11);  // collegamento del servo sul pin 11
}

void loop()
{
  valore = analogRead(pinTrimmer);          // legge il valore dl trimmer
  valore = map(valore, 0, 1023, 0, 120);    // cambiamento di coordinate per poter essere usato con il servo
  myservo.write(valore);                    // la posizione viene impostata sul valore scalato
  delay(20);                                // attesa per permettere al server di raggoingere la posizione
}
```

Sperimentazioni con i Servomotori
(a rotazione continua)

Prof. Michele Maffucci

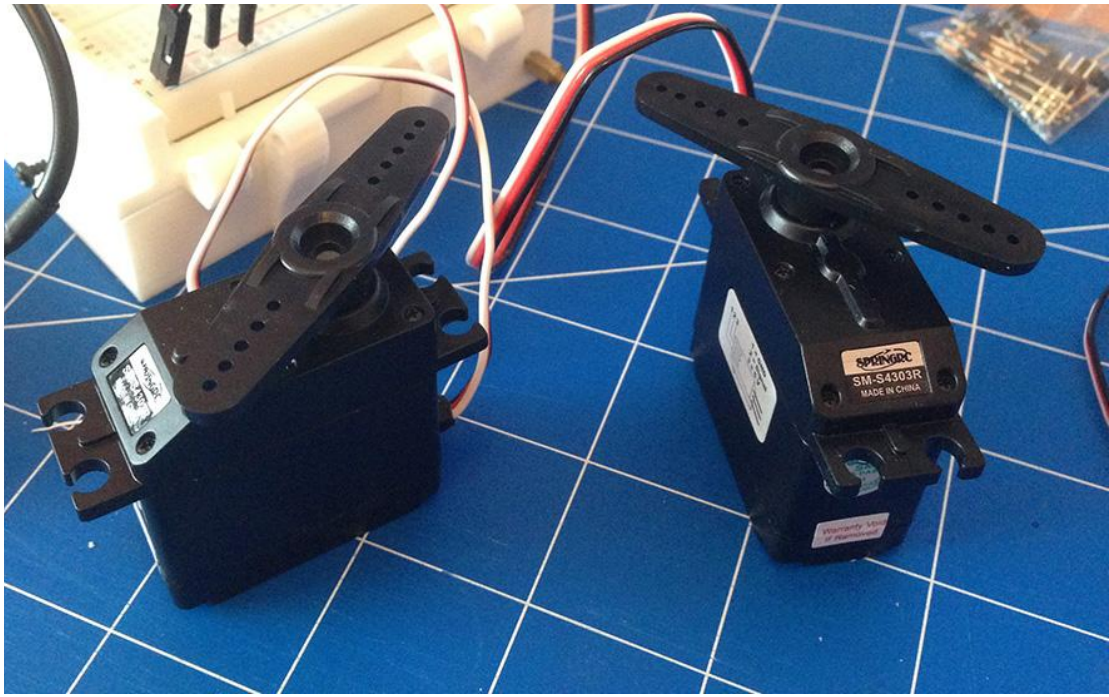
controllare velocità e direzione

sketch43

Esistono servomotori a rotazione continua, la cui rotazione può avvenire in senso orario e antiorario.

E' possibile modificare servomotori con rotazione compreso tra 0° e 180° per renderli a rotazione continua scollegando il sistema di controllo della posizione scollegato.
In questa lezione utilizziamo servomotori non modificati a 360°.

Il funzionamento di un servomotore a rotazione continua è simile a quella di un motore in corrente continua con la differenza che non necessitano di appositi shield per poter funzionare.
Rispetto ai motori esterni offrono scelte limitate per il controllo della velocità e limitazioni di alimentazione.



Utilizzo:

- passando il valore **0** gradi alla funzione write() il servo ruota alla massima velocità in una direzione.
- passando il valore **90** gradi alla funzione write() poniamo il servo in stop (posizione "neutra")
- passando il valore **180** gradi alla funzione write() diciamo al servo di ruotare in senso opposto alla massima velocità.

L'alimentazione necessaria può variare a seconda del servo di cui si dispone ed è probabile che si renda necessario alimentare i servo con una alimentazione esterna da 5 o 6 V. Si ricordi che la messa a terra delle batterie deve essere collegata al GND di Arduino.

L'alimentazione potrà avvenire direttamente Attraverso Arduino o mediante alimentazione esterna. Per questa esercitazione può essere scelta una a piacere.

controllare velocità e direzione

sketch43

versione 1

Si immagini di realizzare il sistema di movimentazione su ruote di un robot utilizzando due servomotori.

Si vuole sperimentare solamente il movimento in avanti e indietro.

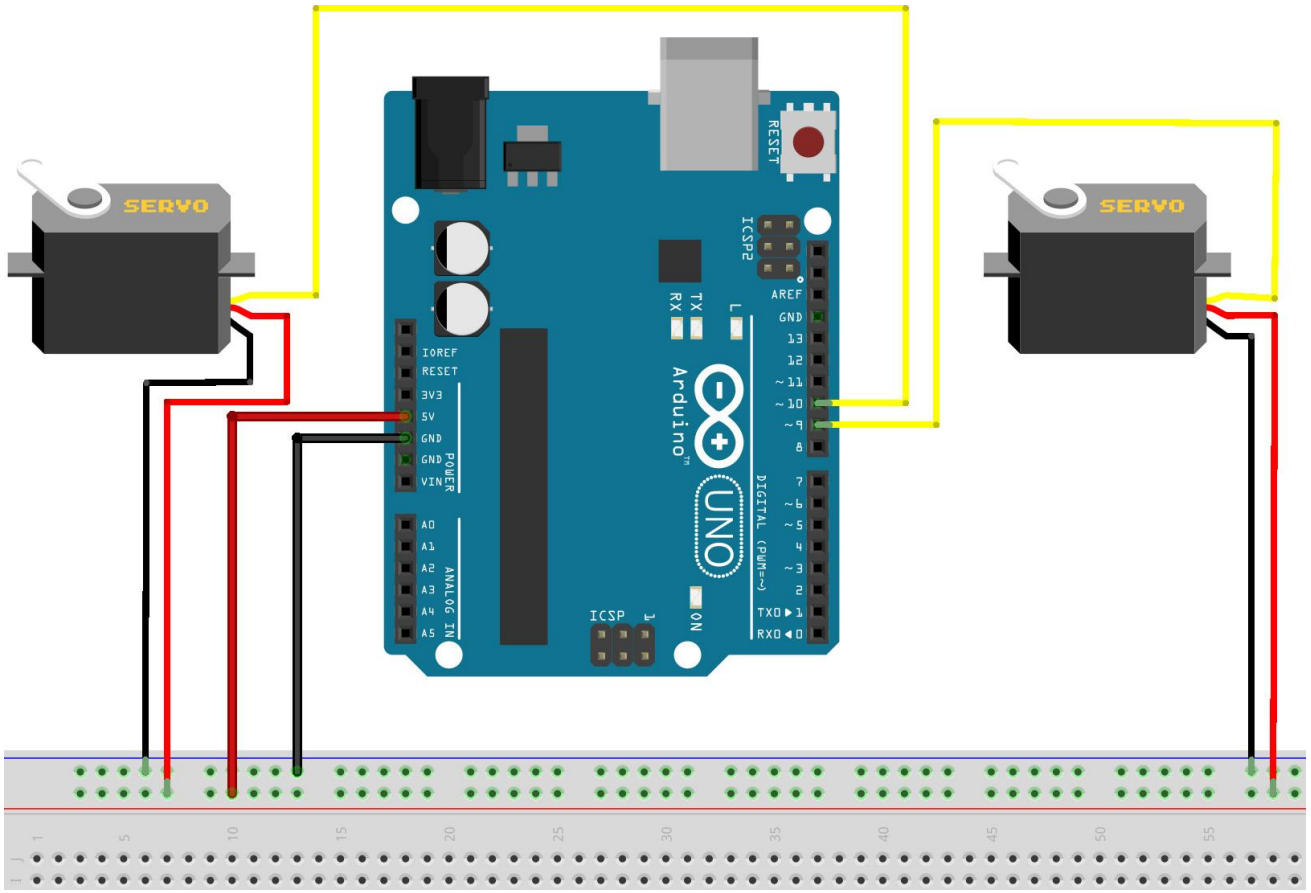
Si vuole controllare la direzione e la velocità di rotazione di due servomotori a rotazione continua.

Suggerimento

Si realizzino due cicli:

- nel primo ciclo l'angolo di rotazione varia tra 0 e 180 gradi;
- nel secondo ciclo l'angolo di rotazione varia tra 180 e 0 gradi.

Il verso di rotazione dei servo sarà quindi opposto per ogni ciclo.

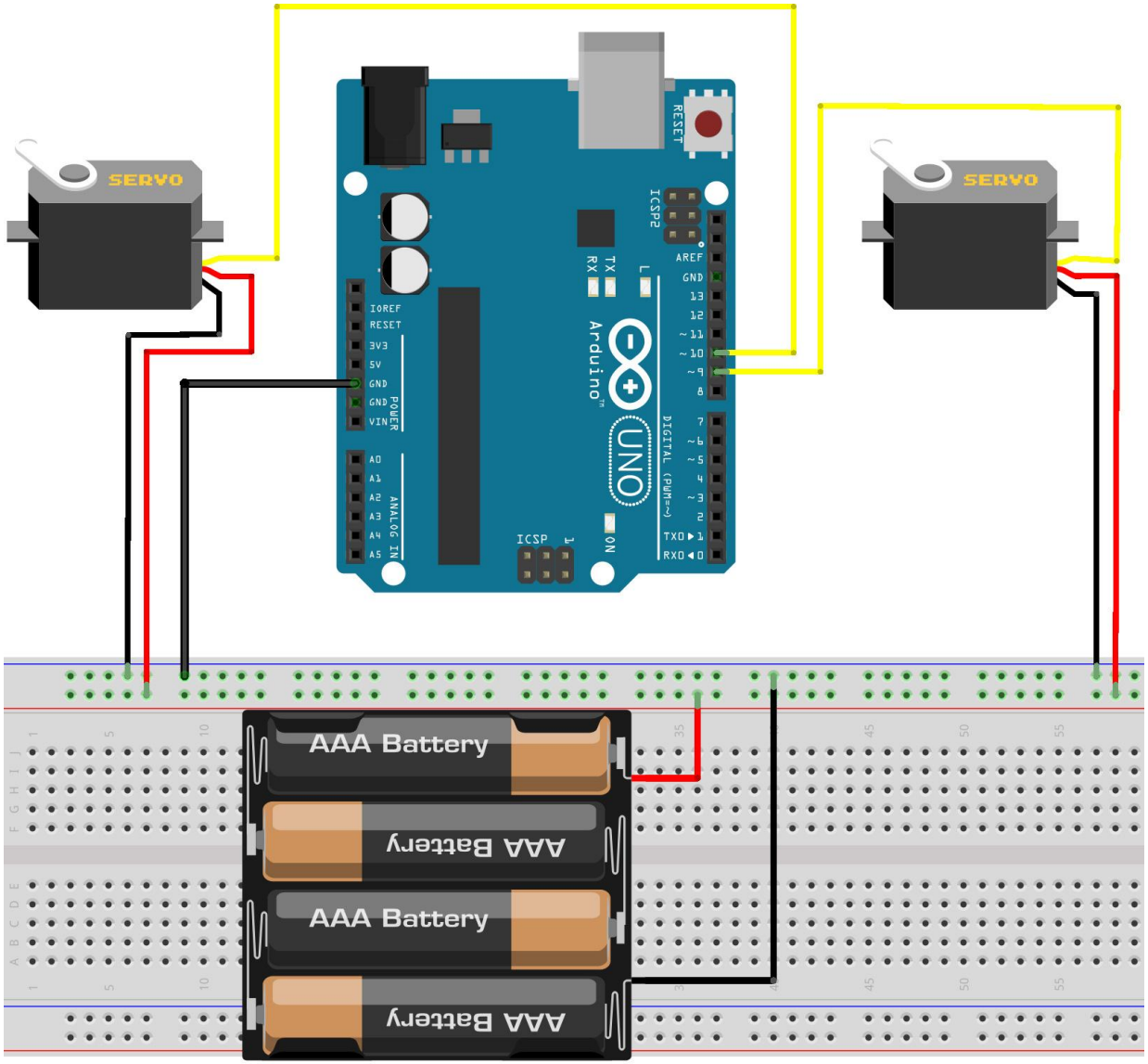


controllare velocità e direzione

3/5

sketch43

versione 2



controllare velocità e direzione | 4/5 | sketch43

```
/* Prof. Michele Maffucci
   19.04.2014

   Utilizzo di un servomotore
   Modello: SM-S4303R da 41g
   Angolo di rotazione: 360 gradi
   Momento torcente a 4,8 V: 3.3 kg-cm
   Momento torcente a 6 V: 5.1 kg-cm

   Accelerazione e decelerazione in senso orario
   e antiorario di un servo a 360 gradi

*/

#include <Servo.h>  // libreria servo

Servo servoSx;  // crea un oggetto servo per controllare il servomotore sinistro
Servo servoDx;  // crea un oggetto servo per controllare il servomotore destro

int angolo = 0;    // variable in cui memorizzare la posizione del servo

void setup()
{
  Serial.begin(9600); // inizializzazione della porta seriale
  servoSx.attach(9);  // collegamento del servo di sinistra al pin 9
  servoDx.attach(10); // collegamento del servo di sinistra al pin 10
}
```

continua...

```
void loop()
{
  for(angolo = 0; angolo < 180; angolo += 1)    // conteggio da 90 a 180 gradi
  {                                             // in passi da 1 grado
                                                // a 90 gradi il servo si ferma
    servoSx.write(angolo);                      // ruota il servo alla velocita' "angolo"
    servoDx.write(180-angolo);                  // va nella direzione opposta
    delay(100);                                 // aspetta 100ms tra i comandi impartiti al servo
                                                // e per percepire l'accelerazione
  }

  for(angolo = 180; angolo >= 0; angolo -= 1) // conteggio da 180 a 90 gradi
  {                                             // in passi da 1 grado
                                                // a 90 gradi il servo si ferma
    servoSx.write(angolo);                      // ruota il servo alla velocita' "angolo"
    servoDx.write(180-angolo);                  // va nella direzione opposta
    delay(100);                                 // aspetta 100ms tra i comandi impartiti al servo
                                                // e per percepire la decelerazionze
 }
}
```

continua...

controllare la rotazione via software

sketch44

Si vuole controllare mediante input da tastiera sulle Serial Monitor il verso di rotazione di due servomotori.

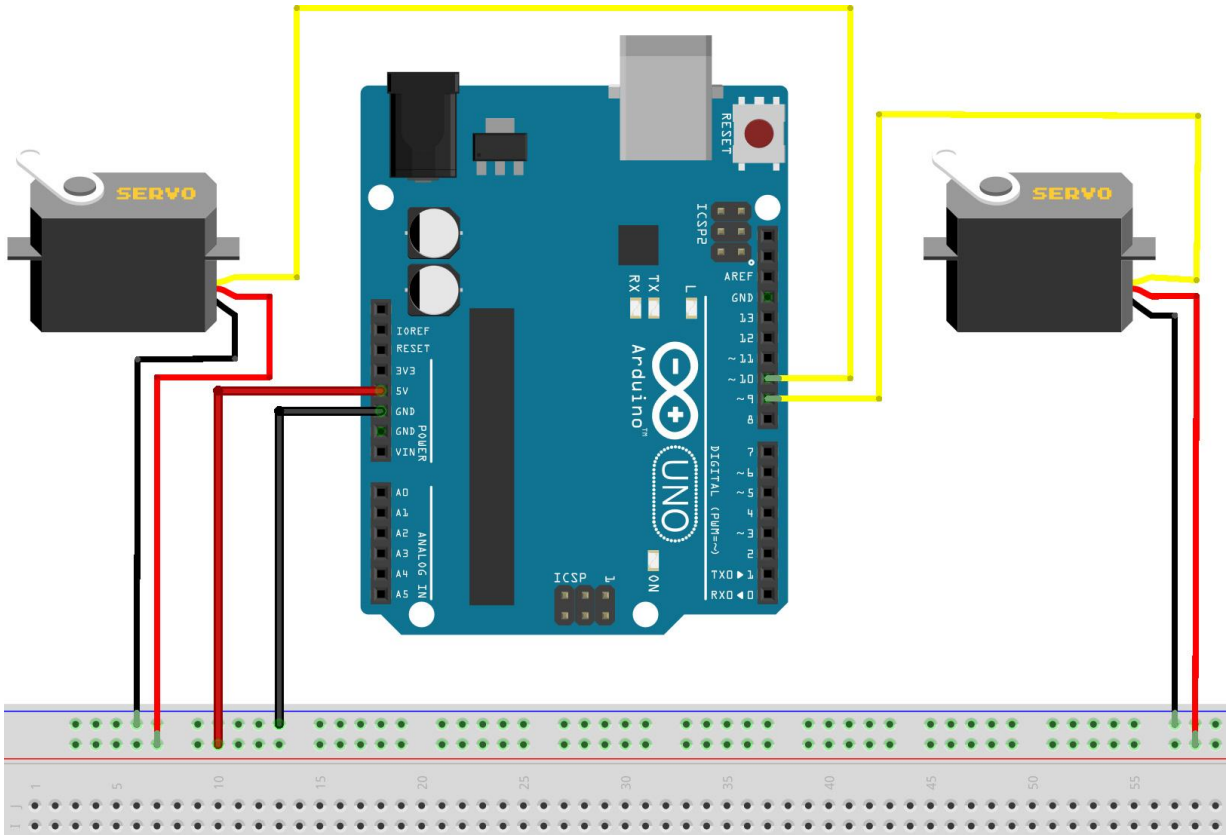
Specifiche:

- premendo il tasto 1 i sevo girano in senso orario;
- premendo il tasto 2 i sevo girano in senso orario;
- premendo il tasto 3 i servo si fermano;

Segnalare sulla Serial Monitor lo stato in cui si trovano i due servomotori mediante messaggi.

All'avvio dello sketch far comparire un menù in cui vengono segnalati le funzionalità dei tasti di controllo.

Anche per questo esercizio è possibile utilizzare uno dei due circuiti dell' esercizio precedente (alimentazione servo tramite Arduino oppure alimentazione servo tramite batterie).



controllare la rotazione via software

2/3

sketch44

```
/* Prof. Michele Maffucci
   19.04.2014

   Utilizzo di un servomotore
   Modello: SM-S4303R da 41g
   Angolo di rotazione: 360 gradi
   Momento torcente a 4,8 V: 3.3 kg-cm
   Momento torcente a 6 V: 5.1 kg-cm

   Controllo rotazione servomotore 360 gradi
   mediante input da tastiera.

*/

#include <Servo.h>  // libreria servo

Servo servoSx;  // crea un oggetto servo per controllare il servomotore sinistro
Servo servoDx;  // crea un oggetto servo per controllare il servomotore destro

void setup()
{
  Serial.begin(9600); // inizializzazione della porta seriale
  servoSx.attach(9);  // collegamento del servo di sinistra al pin 9
  servoDx.attach(10); // collegamento del servo di sinistra al pin 10

// menu' di utilizzo

  Serial.println("---------- menu' ----------");
  Serial.println("pulsante 1 - vai avanti");
  Serial.println("pulsante 2 - vai indietro");
  Serial.println("pulsante 3 - fermati");
  Serial.println("---------------------------");
}
```

continua...

controllare la rotazione via software

3/3

sketch44

```
void loop()
{
  if (Serial.available())          // verifica disponibilia' di un carattere sulla serial
  {
    byte rotazione = Serial.read(); // legge il valore inserito

 // se 1 viene chiamata la funzione avanti e inviato messaggio sulla serial
    if (rotazione == '1')
    {
      avanti();
      Serial.println("Sto andando avanti");
    }
 // se 2 viene chiamata la funzione indietro e inviato messaggio sulla serial
    if (rotazione == '2')
    {
      indietro();
      Serial.println("Sto andando indietro");
    }
 // se 3 viene chiamata la funzione ferma e inviato messaggio sulla serial
    if (rotazione == '3')
    {
      ferma();
      Serial.println("Sono fermo");
    }
  }
}

void avanti() {
  servoSx.write(0);
  servoDx.write(180);
}

void indietro() {
  servoSx.write(180);
  servoDx.write(0);
}

void ferma() {
  servoSx.write(90);
  servoDx.write(90);
}
```

Controllare motori DC con un ponte H

Prof. Michele Maffucci

Controllo direzione via software

sketch45

Tipicamente Arduino non è in grado di fornire la corrente sufficiente per permettere la rotazione di un motorino elettrico in corrente continua, pertando risulta necessario utilizzare circuiti esterni che consentano di risolvere il problema.
Arduino verrà utilizzato quindi per comandare i circuiti di "potenza" e per impostare velocità e senso di rotazione dei motori.

L'interfaccia di pilotaggio motori è costituita da appositi circuiti a transitor e a diodi, che per la modalità di collegamento vengono detti a ponte H.

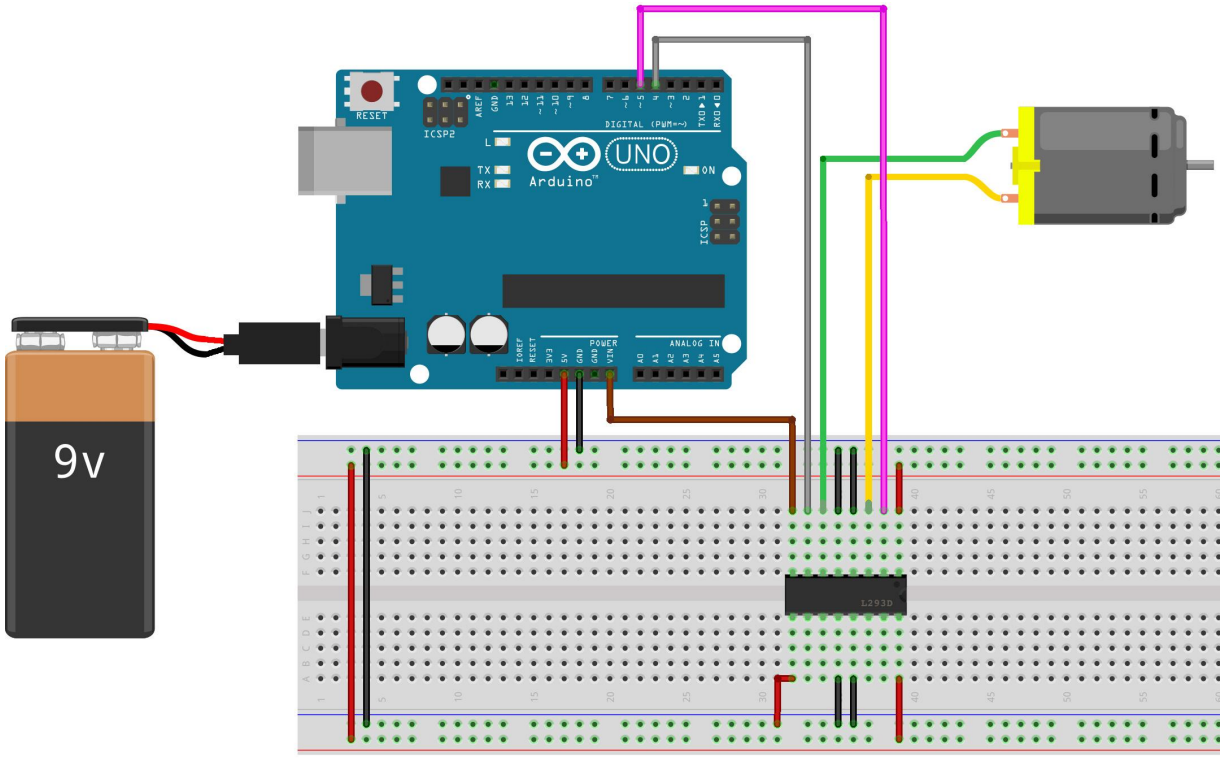
Più praticamente si può sostituire un circuito a transistor con appositi integrati che semplificano la realizzazione dei circuiti di potenza.

Tra gli integrati più comuni a questo scopo si utilizza L293D.

In questo esercizio vedremo come pilotare un motore in CC.

Realizzare uno sketch che permetta di controllare la rotazione oraria e antioraria di un motorino in CC mediante comandi inseriti attraverso la porta seriale.
All'avvio dello sketch visualizzare un menù che mostri le funzionalità dei pulsanti.

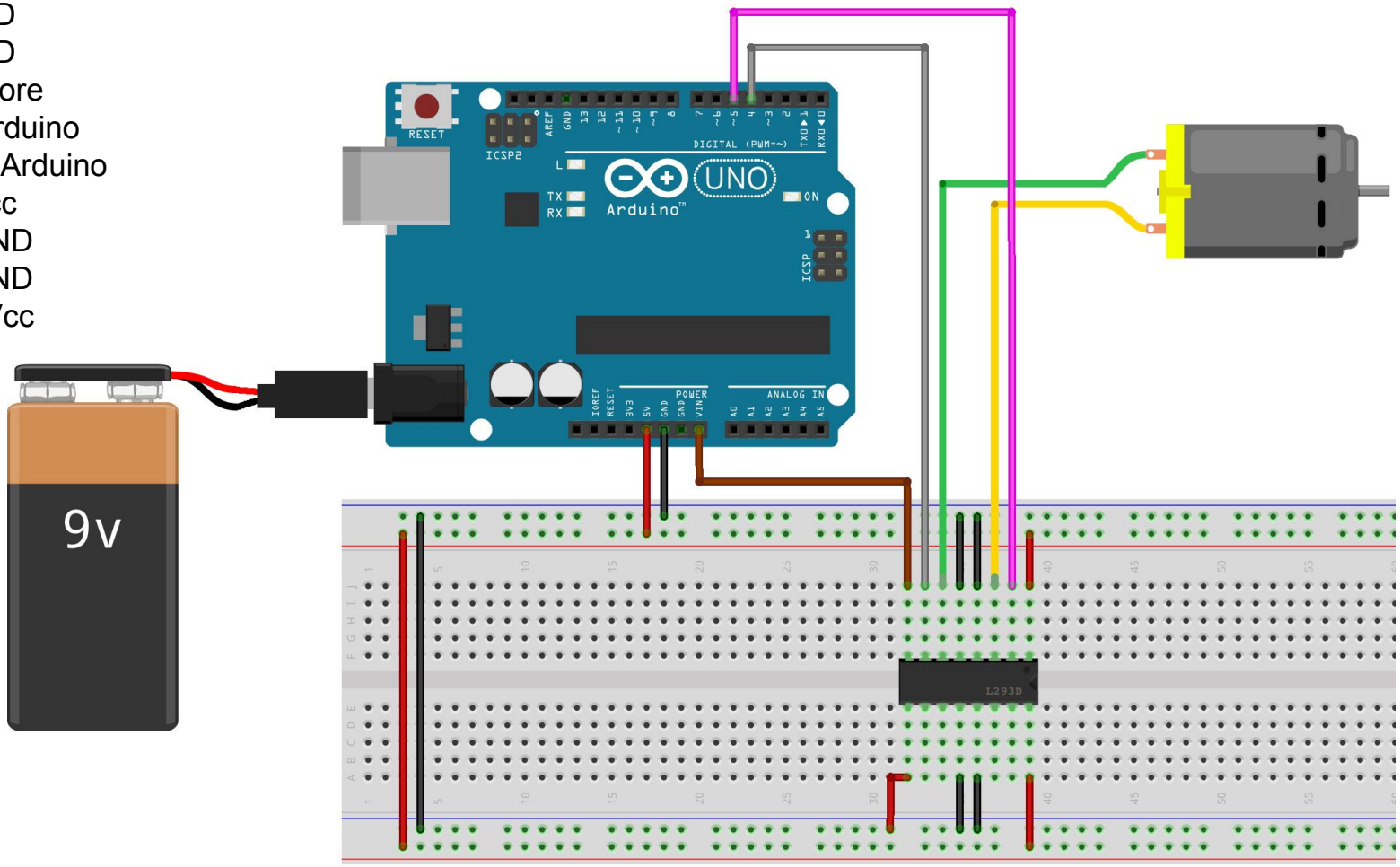
- 1 - gira in senso orario
- 2 - gira in senso antiorario
- qualsiasi tasto - ferma il motore



Controllo direzione via software

Collegamenti
L293D > Arduino

pin 1: +Vcc
pin 2: pin 4 Arduino
pin 3: motore
pin 4: GND
pin 5: GND
pin 6: motore
pin 7: 4 Arduino
pin 8: Vin Arduino
pin 9: +Vcc
pin 12: GND
pin 13: GND
pin 16: +Vcc



Controllo direzione via software

sketch45

```
/* Prof. Michele Maffucci
   19.04.2014

   Utilizzo del ponte H L293D
   per pilotare da tastiers il senso
   di rotazione di un motore in CC

   Collegamenti
   L293D > Arduino

   pin 1 - pin 9 - pin 16: +Vcc
   pin 2: pin 4 Arduino
   pin 3: motore
   pin 4 - pin 5 - pin 12 - pin 13: GND
   pin 6: motore
   pin 7: 4 Arduino
   pin 8: Vin Arduino
*/

// Pin di input del ponte H
const int pinIngresso1 = 5; // collegato al pin 3 dell'L293D
const int pinIngresso2 = 4; // collegato al pin 6 dell'L293D

void setup()
{
  Serial.begin(9600);              // inizializzazione della porta seriale
  pinMode(pinIngresso1, OUTPUT);   // pin di controllo senso di rotazione
  pinMode(pinIngresso2, OUTPUT);   // pin di controllo senso di rotazione
  Serial.println("1 - 2 impostano la direzione, qualsiasi altro pulsante ferma il motore");
  Serial.println("-----------------------------------------------------------------");
}
```

continua...

Controllo direzione via software

sketch45

```
void loop()
{
  if ( Serial.available()) {        // verifica disponibilia' di un carattere sulla serial
    char ch = Serial.read();        // legge il carattere inserito
    if (ch == '1')                  // rotazione oraria
    {
      Serial.println("Rotazione oraria");
      digitalWrite(pinIngresso1,LOW);
      digitalWrite(pinIngresso2,HIGH);
    }
    else if (ch == '2')              // rotazione antioraria
    {
      Serial.println("Rotazione antioraria");
      digitalWrite(pinIngresso1,HIGH);
      digitalWrite(pinIngresso2,LOW);
    }
    else
    {
      Serial.println("Motore fermo");   // ferma il motore
      digitalWrite(pinIngresso1,LOW);
      digitalWrite(pinIngresso2,LOW);
    }
  }
}
```

continua...

Controllo direzione via software

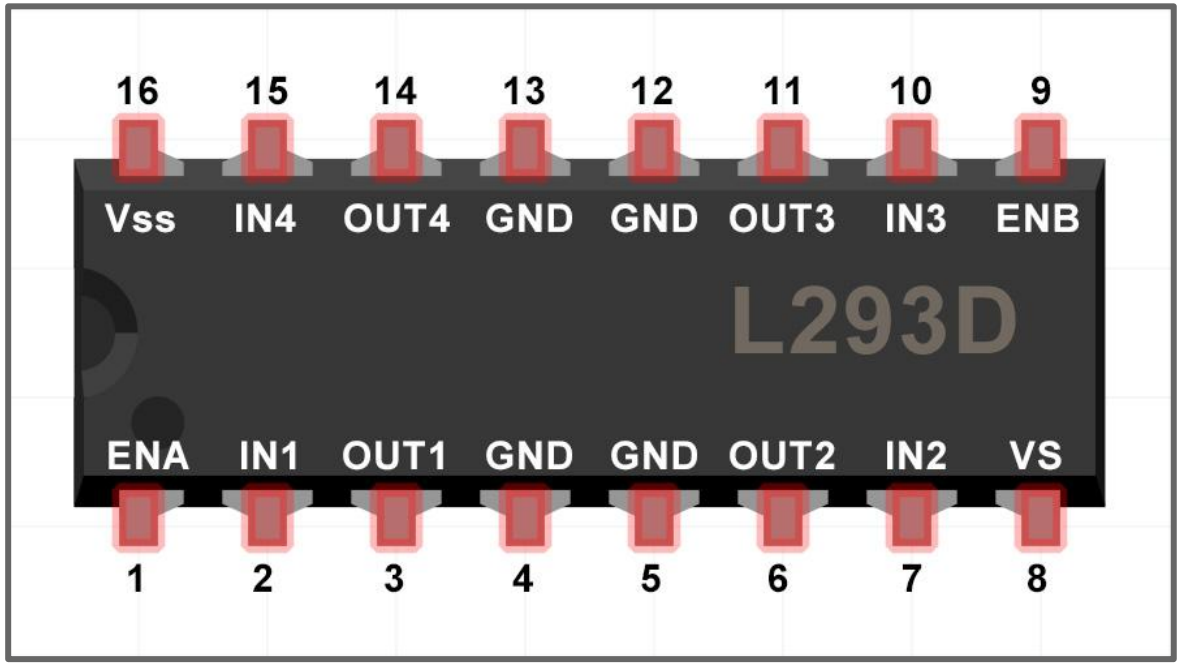
5/5 sketch45

Logica di funzionamento del ponte H per movimentare un motore in CC.

Nell'immagine che segue può essere visualizzata la relazione tra la parte di codice che pilota il motore e la tabella con la logica di funzionamento.

```
void loop()
{
  if ( Serial.available()) {          // verifi
    char ch = Serial.read();          // legge
    if (ch == '1')                    // rotazi
    {
      Serial.println("Rotazione oraria");
      digitalWrite(pinIngresso1,LOW);
      digitalWrite(pinIngresso2,HIGH);
    }
    else if (ch == '2')               // rotaz
    {
      Serial.println("Rotazione antioraria");
      digitalWrite(pinIngresso1,HIGH);
      digitalWrite(pinIngresso2,LOW);
    }
    else
    {
      Serial.println("Motore fermo");   // fe
      digitalWrite(pinIngresso1,LOW);
      digitalWrite(pinIngresso2,LOW);
    }
  }
}
```

EN	IN1	IN2	FUNZIONE
HIGH	LOW	HIGH	Gira in senso orario
HIGH	HIGH	LOW	Gira in senso antiorario
HIGH	LOW	LOW	Ferma il motore
HIGH	HIGH	HIGH	Ferma il motore
LOW	IGNORATO	IGNORATO	Ferma il motore



Controllo di due motori DC via software

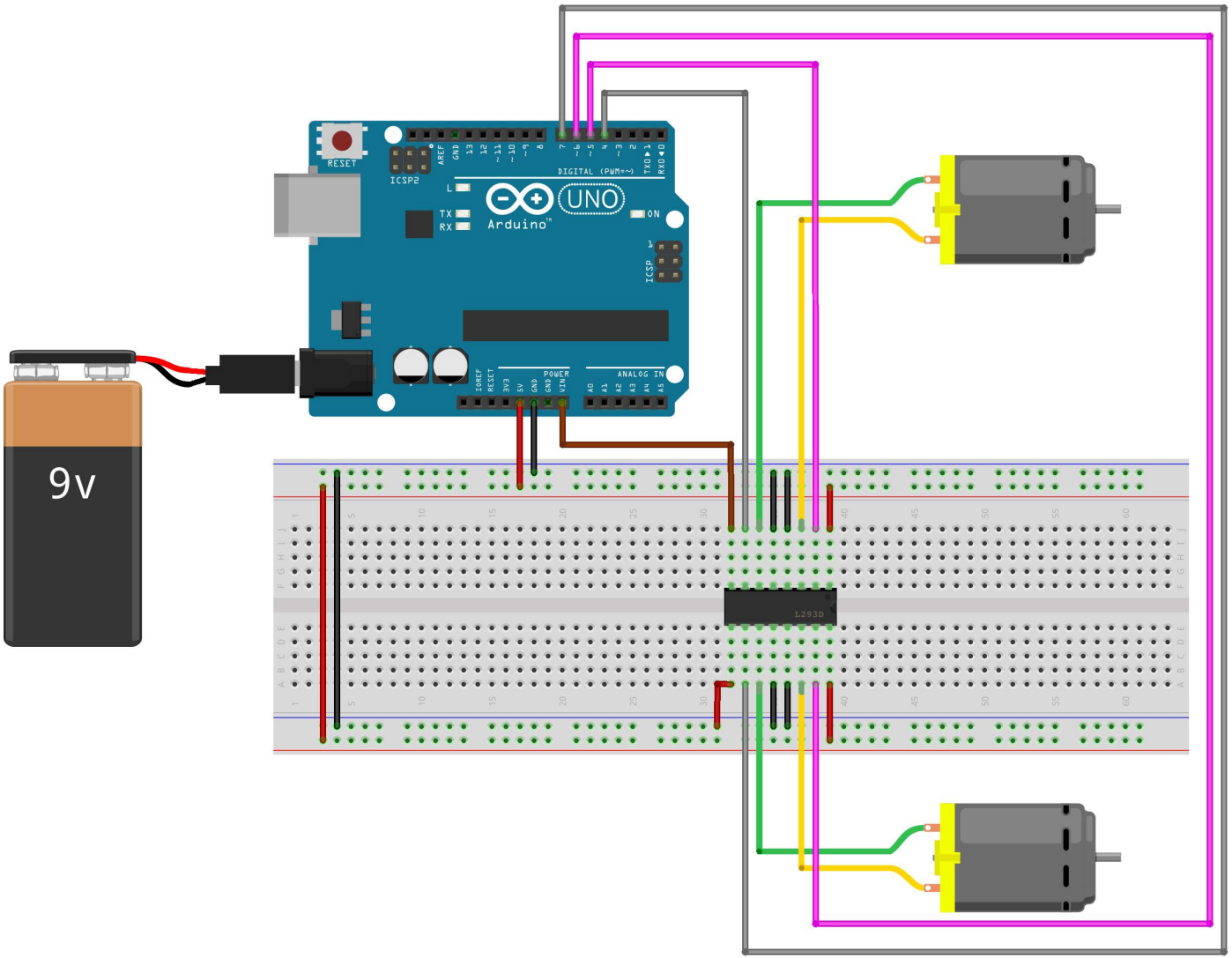
1/5

sketch46

Realizzare uno sketch che permetta di controllare la rotazione oraria e antioraria di due motorini in CC mediante comandi inseriti attraverso la porta seriale.

All'avvio dello sketch visualizzare un menù che mostri le funzionalità dei pulsanti.

- 1 - girano in senso orario;
- 2 - girano in senso antiorario;
- qualsiasi tasto - motori fermi.

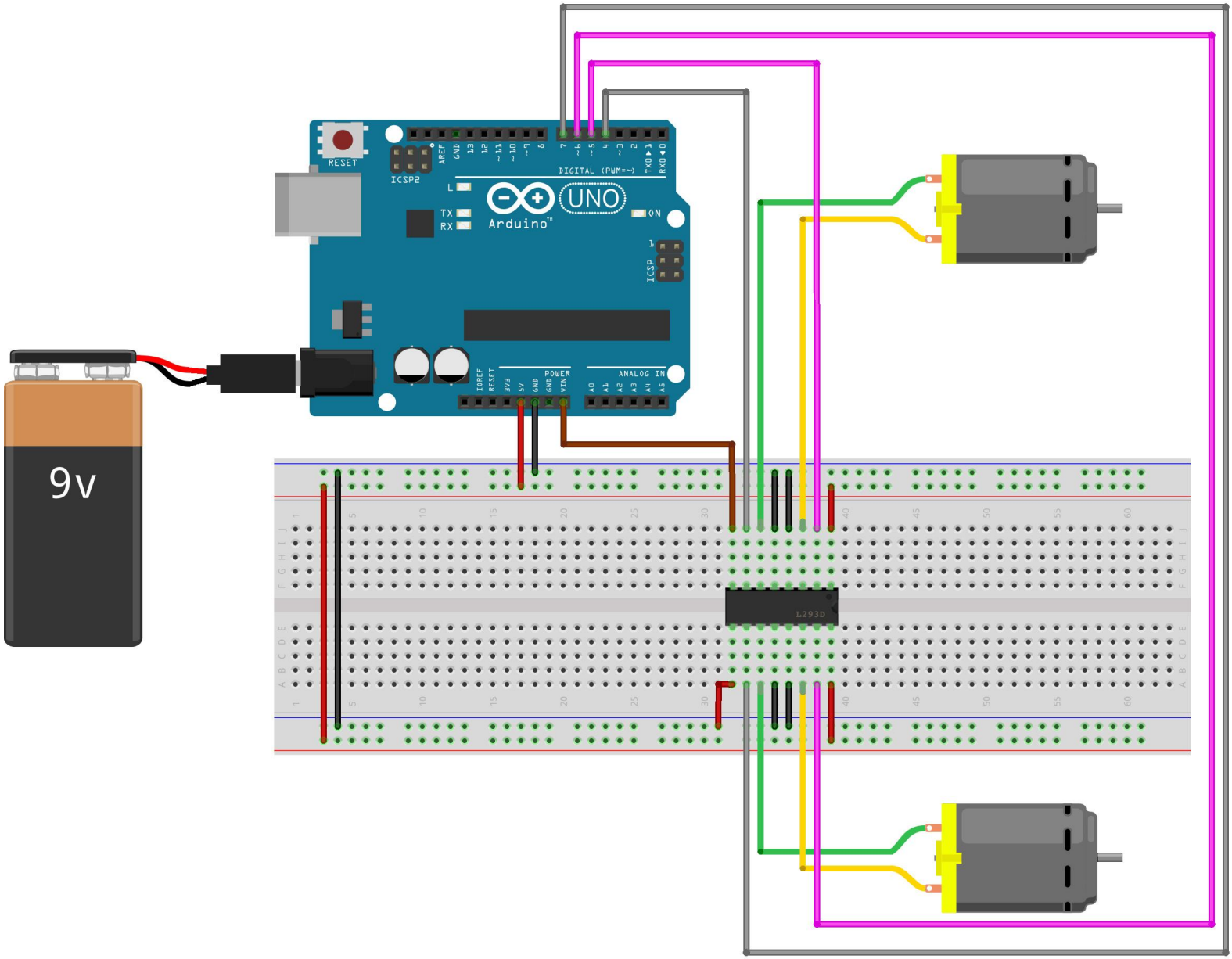


Prof. Michele Maffucci

Controllo di due motori DC con ponte H

Collegamenti
L293D > Arduino

pin 1: +Vcc
pin 2: pin 4 Arduino
pin 3: motore 1
pin 4: GND
pin 5: GND
pin 6: motore 1
pin 7: 4 Arduino
pin 8: Vin Arduino
pin 9: +Vcc
pin 10: pin 7
pin 11: motore 2
pin 12: GND
pin 13: GND
pin 14: motore 2
pin 15: pin 6
pin 16: +Vcc



Controllo di due motori DC con ponte H


```
/* Prof. Michele Maffucci
   19.04.2014

   Utilizzo del ponte H L293D
   per pilotare da tastiers il senso
   di rotazione di due motore in CC

   Collegamenti
   L293D > Arduino

   pin 1 - pin 9 - pin 16: +Vcc
   pin 2: pin 4 Arduino
   pin 3: motore 1
   pin 4 - pin 5 - pin 12 - pin 13: GND
   pin 6: motore 1
   pin 7: 4 Arduino
   pin 8: Vin Arduino
   pin 10: pin 7
   pin 11: motore 2
   pin 14: motore 2
   pin 15: pin 6
*/

// Pin di input del ponte H
const int pinIngresso1 = 5; // collegato al pin 3 dell'L293D
const int pinIngresso2 = 4; // collegato al pin 6 dell'L293D
const int pinIngresso3 = 7; // collegato al pin 11 dell'L293D
const int pinIngresso4 = 6; // collegato al pin 14 dell'L293D

void setup()
{
  Serial.begin(9600);             // inizializzazione della porta seriale
  pinMode(pinIngresso1, OUTPUT);  // pin di controllo senso di rotazione - motore 1
  pinMode(pinIngresso2, OUTPUT);  // pin di controllo senso di rotazione - motore 1
  pinMode(pinIngresso3, OUTPUT);  // pin di controllo senso di rotazione - motore 2
  pinMode(pinIngresso4, OUTPUT);  // pin di controllo senso di rotazione - motore 2
  Serial.println("1 - 2 impostano la direzione, qualsiasi altro pulsante ferma il motore");
  Serial.println("--------------------------------------------------------------------");
}
```

continua...

Controllo di due motori DC con ponte H

4/5

sketch46

```
void loop()
{
  if ( Serial.available()) {        // verifica disponibilia' di un carattere sulla serial
    char ch = Serial.read();        // legge il carattere inserito
    if (ch == '1')                  // rotazione oraria
    {
      Serial.println("Rotazione oraria");
      // motore 1
      digitalWrite(pinIngresso1,LOW);
      digitalWrite(pinIngresso2,HIGH);
      // motore 2
      digitalWrite(pinIngresso3,LOW);
      digitalWrite(pinIngresso4,HIGH);
    }
    else if (ch == '2')              // rotazione antioraria
    {
      Serial.println("Rotazione antioraria");
      // motore 1
      digitalWrite(pinIngresso1,HIGH);
      digitalWrite(pinIngresso2,LOW);
      // motore 2
      digitalWrite(pinIngresso3,HIGH);
      digitalWrite(pinIngresso4,LOW);
    }
    else
    {
      Serial.println("Motori fermi");   // ferma i motori
      // motore 1
      digitalWrite(pinIngresso1,LOW);
      digitalWrite(pinIngresso2,LOW);
      // motore 2
      digitalWrite(pinIngresso3,LOW);
      digitalWrite(pinIngresso4,LOW);
    }
  }
}
```

Controllo di due motori DC con ponte H

5/5

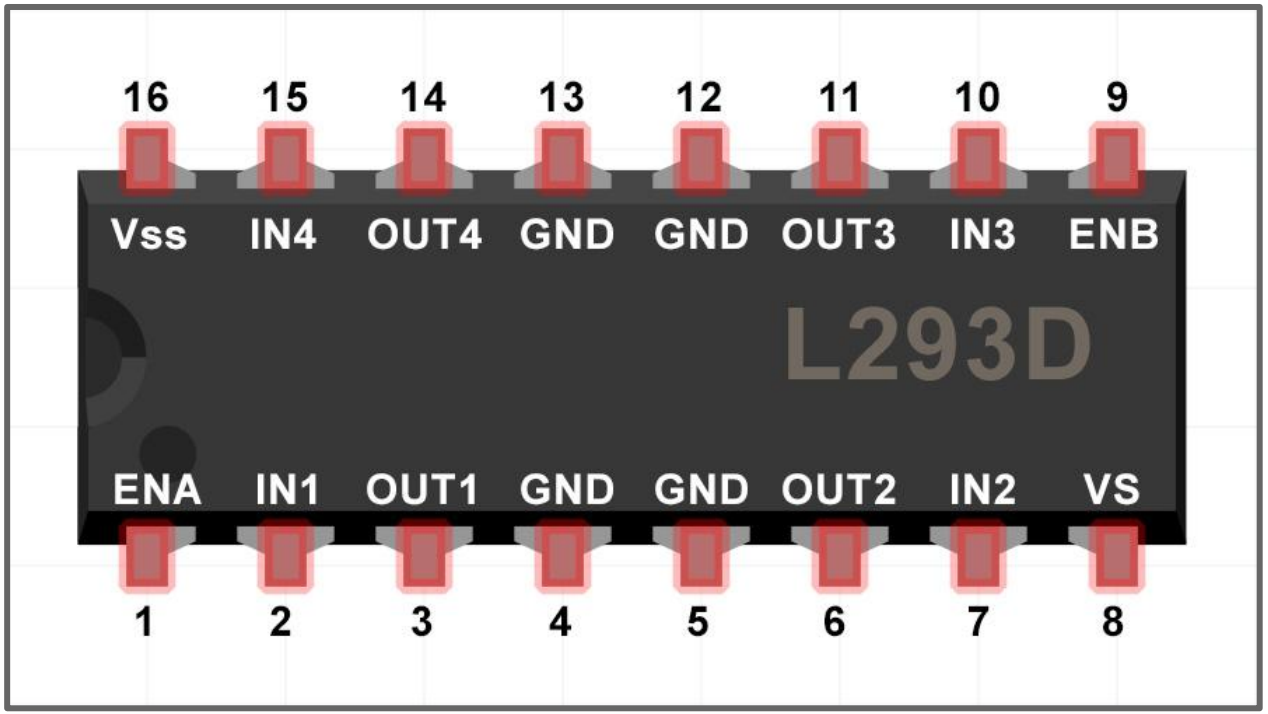
sketch46

Logica di funzionamento del ponte H per movimentare un motore in CC.

Nell'immagine che segue può essere visualizzata la relazione tra la parte di codice che pilota il motore e la tabella con la logica di funzionamento.

```
void loop()
{
  if ( Serial.available()) {        // verifica disponi
    char ch = Serial.read();        // legge il caratte
    if (ch == '1')                  // rotazione oraria
    {
      Serial.println("Rotazione oraria");
      // motore 1
      digitalWrite(pinIngresso1,LOW);
      digitalWrite(pinIngresso2,HIGH);
      // motore 2
      digitalWrite(pinIngresso3,LOW);
      digitalWrite(pinIngresso4,HIGH);
    }
    else if (ch == '2')             // rotazione antio
    {
      Serial.println("Rotazione antioraria");
      // motore 1
      digitalWrite(pinIngresso1,HIGH);
      digitalWrite(pinIngresso2,LOW);
      // motore 2
      digitalWrite(pinIngresso3,HIGH);
      digitalWrite(pinIngresso4,LOW);
    }
    else
    {
      Serial.println("Motori fermi");   // ferma i moto
      // motore 1
      digitalWrite(pinIngresso1,LOW);
      digitalWrite(pinIngresso2,LOW);
      // motore 2
      digitalWrite(pinIngresso3,LOW);
      digitalWrite(pinIngresso4,LOW);
    }
  }
}
```

EN	IN1	IN2	IN3	IN4	FUNZIONE
HIGH	LOW	HIGH	LOW	HIGH	Gira in senso orario
HIGH	HIGH	LOW	HIGH	LOW	Gira in senso antiorario
HIGH	LOW	LOW	LOW	LOW	Ferma il motore
HIGH	HIGH	HIGH	HIGH	HIGH	Ferma il motore
LOW	IGNORATO	IGNORATO	IGNORATO	IGNORATO	Ferma il motore



Controllo velocità e direzione di due motori DC con ponte H

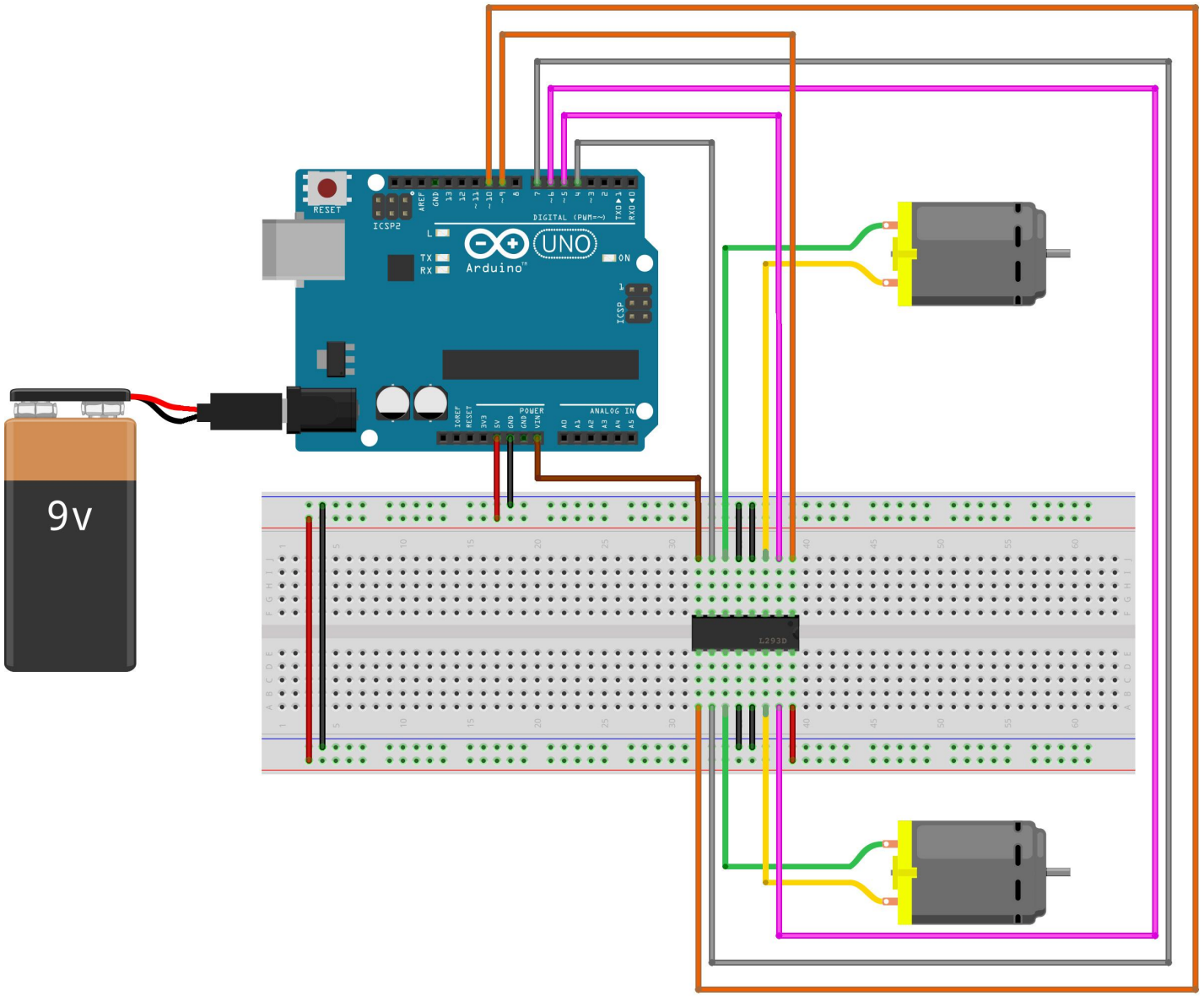
sketch47

Realizzare uno sketch che permetta di controllare attraverso la serial monitor due motori in CC che abbia le seguenti funzionalità:

- rotazione oraria e antioraria di due motorini
- velocità di rotazione

All'avvio dello sketch visualizzare un menù che mostri le funzionalità dei pulsanti.

- O - girano in senso orario;
- A - girano in senso antiorario;
- 0 (zero) - motori fermi;
- da 1 a 9 si imposta la velocità di rotazione.



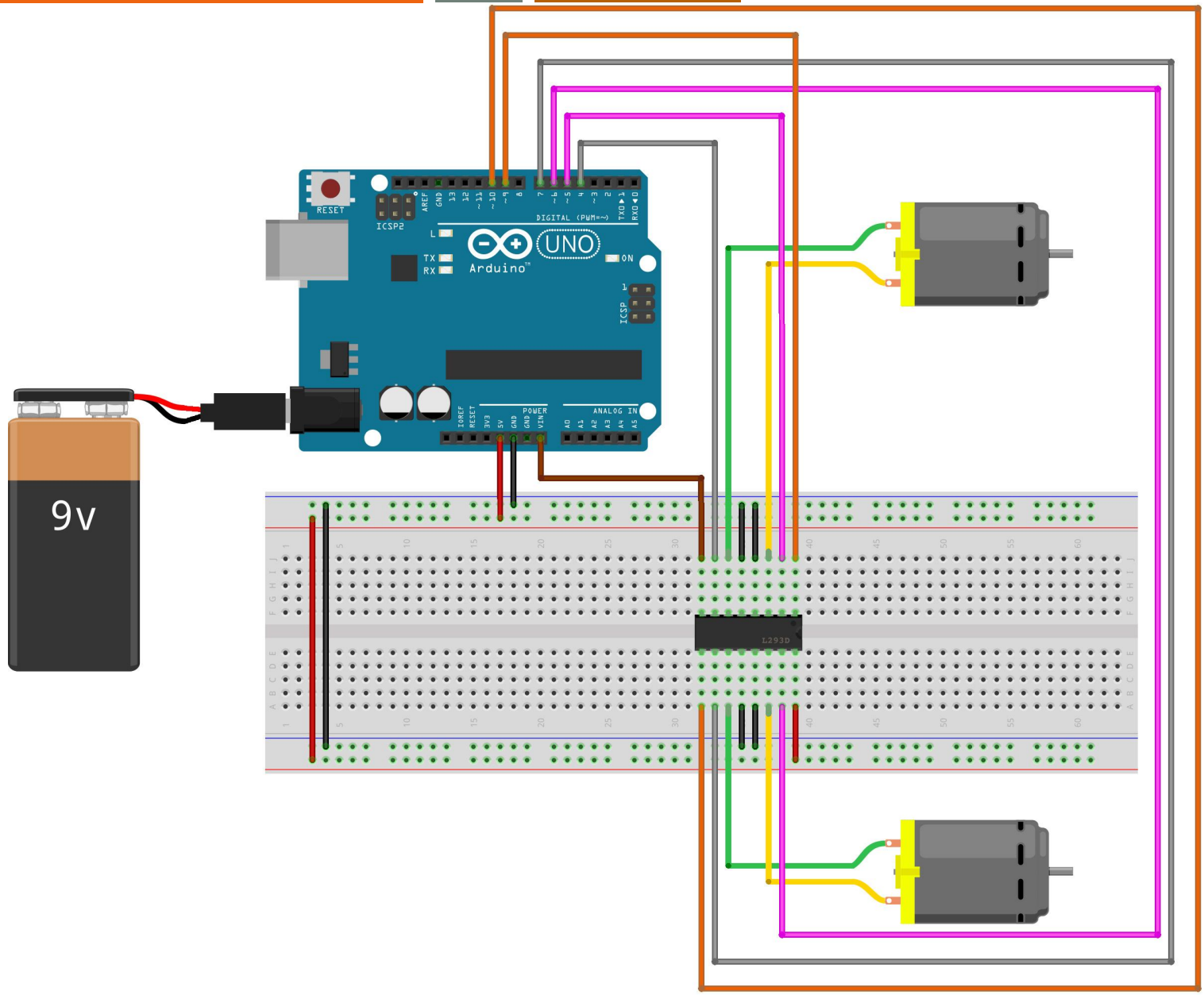
Prof. Michele Maffucci

Controllo velocità e direzione di due motori DC con ponte H

sketch47

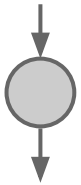
Collegamenti
L293D > Arduino

pin 1: pin 9
pin 2: pin 4 Arduino
pin 3: motore 1
pin 4: GND
pin 5: GND
pin 6: motore 1
pin 7: 4 Arduino
pin 8: Vin Arduino
pin 9: pin 10
pin 10: pin 7
pin 11: motore 2
pin 12: GND
pin 13: GND
pin 14: motore 2
pin 15: pin 6
pin 16: +Vcc



Controllo velocità e direzione di due motori DC con ponte H

sketch47



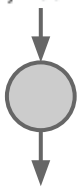
```
/* Prof. Michele Maffucci
   19.04.2014

   Utilizzo del ponte H L293D
   per pilotare da tastiera il senso
   di rotazione e la velocita' di due motori in CC

   Collegamenti
   L293D > Arduino

   pin 1 - pin 9
   pin 16: +Vcc
   pin 2: pin 4 Arduino
   pin 3: motore 1
   pin 4 - pin 5 - pin 12 - pin 13: GND
   pin 6: motore 1
   pin 7: 4 Arduino
   pin 8: Vin Arduino
   pin 9 - pin 10
   pin 10: pin 7
   pin 11: motore 2
   pin 14: motore 2
   pin 15: pin 6
*/

// Pin di input del ponte H
const int pinIngresso1 = 5; // collegato al pin 3 dell'L293D
const int pinIngresso2 = 4; // collegato al pin 6 dell'L293D
const int pinIngresso3 = 7; // collegato al pin 11 dell'L293D
const int pinIngresso4 = 6; // collegato al pin 14 dell'L293D
const int pinEnableA = 9;   // collegato al pin 1 dell'L293D
const int pinEnableB = 10;  // collegato al pin 9 dell'L293D
```



```
void setup()
{
  Serial.begin(9600);             // inizializzazione della porta seriale
  pinMode(pinIngresso1, OUTPUT);  // pin di controllo senso di rotazione - motore 1
  pinMode(pinIngresso2, OUTPUT);  // pin di controllo senso di rotazione - motore 1
  pinMode(pinIngresso3, OUTPUT);  // pin di controllo senso di rotazione - motore 2
  pinMode(pinIngresso4, OUTPUT);  // pin di controllo senso di rotazione - motore 2

  // i pin pinEnableA (corrispondente al 9) e pinEnableB (corrispondente al 10) sono di tipo PWM
  pinMode(pinEnableA, OUTPUT);  // pin enable - motore 1
  pinMode(pinEnableB, OUTPUT);  // pin enable - motore 2

  Serial.println("0 - A - senso di rotazione dei motori");
  Serial.println("-------------------------------------");
  Serial.println("da 0 a 9 - velocita' motori");
  Serial.println("-------------------------------------");
}
```

continua...

Controllo velocità e direzione di due motori DC con ponte H

sketch47

```
void loop()
{
  if ( Serial.available()) {              // verifica disponibilia' di un carattere sulla serial

    char carattere = Serial.read();       // legge il carattere inserito

    if (isDigit(carattere)) {
    int velocita = map(carattere, '0', '9', 0, 255);  // rimappa i valori tra 0 e 9, nell'intervallo 0, 255
                                                      // per impostare il duty cyicle
    analogWrite(pinEnableA, velocita);                // imposta la velocita' per il motore 1
    analogWrite(pinEnableB, velocita);                // imposta la velocita' per il motore 2
    Serial.println(velocita);                          // stampa sulla serial monitor la velocita' dei motori
    }
    else if (carattere == 'O')                 // rotazione oraria
    {
      Serial.println("Rotazione oraria");
      // motore 1
      digitalWrite(pinIngresso1,LOW);
      digitalWrite(pinIngresso2,HIGH);
      // motore 2
      digitalWrite(pinIngresso3,LOW);
      digitalWrite(pinIngresso4,HIGH);
    }
    else if (carattere == 'A')                // rotazione antioraria
    {
      Serial.println("Rotazione antioraria");
      // motore 1
      digitalWrite(pinIngresso1,HIGH);
      digitalWrite(pinIngresso2,LOW);
      // motore 2
      digitalWrite(pinIngresso3,HIGH);
      digitalWrite(pinIngresso4,LOW);
    }
    else
    {
      Serial.println("Carattere non consentito");   // altri caratteri non sono consentiti
      Serial.print(carattere);                      // stampa il carattere inserito
    }
  }
}
```

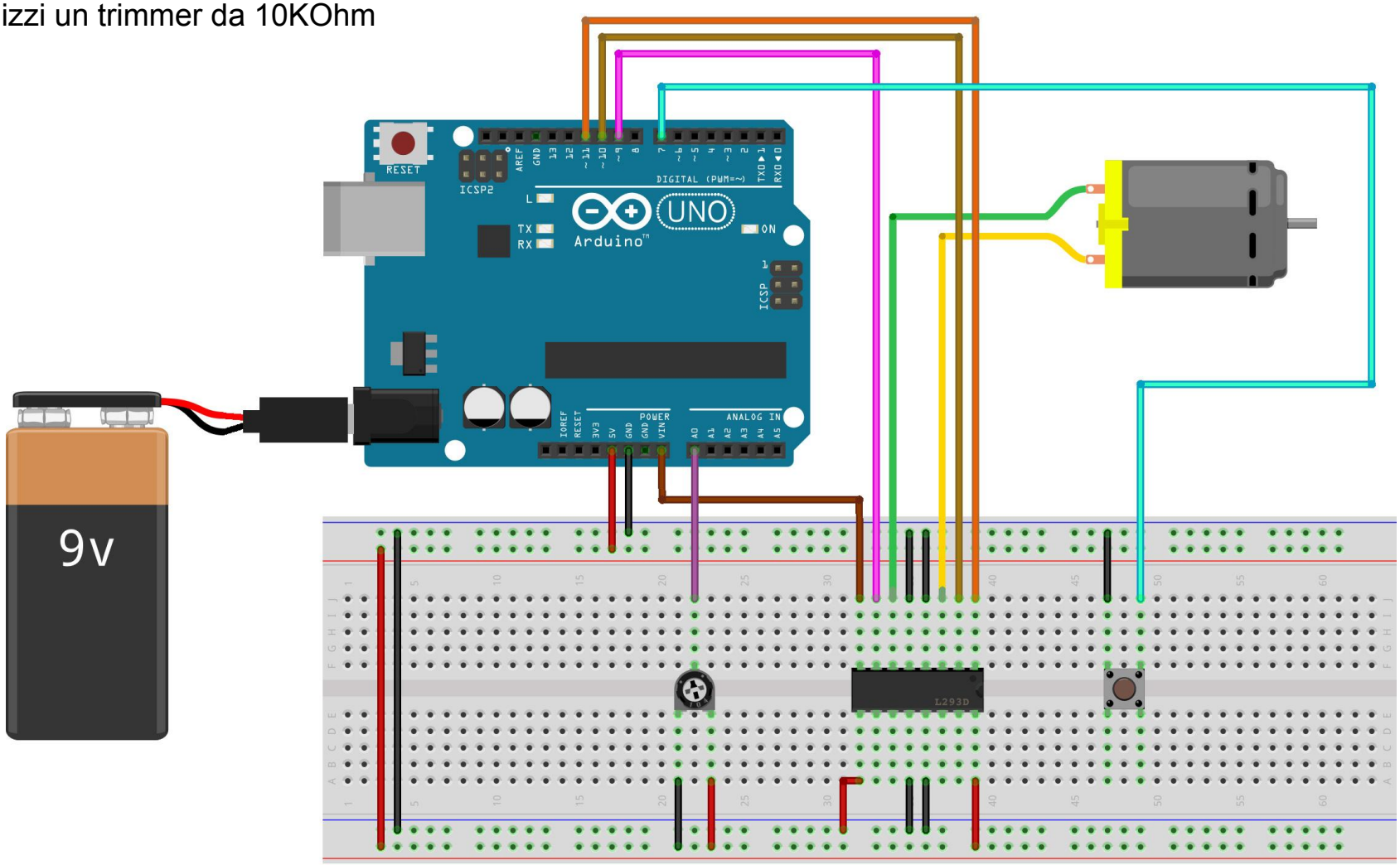
Controllo di un motore DC mediante pulsante e trimmer

1/4

sketch48

Diversamente dalle esercitazioni precedenti, in cui il controllo di velocità e rotazione veniva effettuato via software, ora si vuole realizzare un circuito in cui il verso di rotazione è impostato mediante la pressione di un pulsante e la velocità di rotazione viene stabilita con un trimmer.

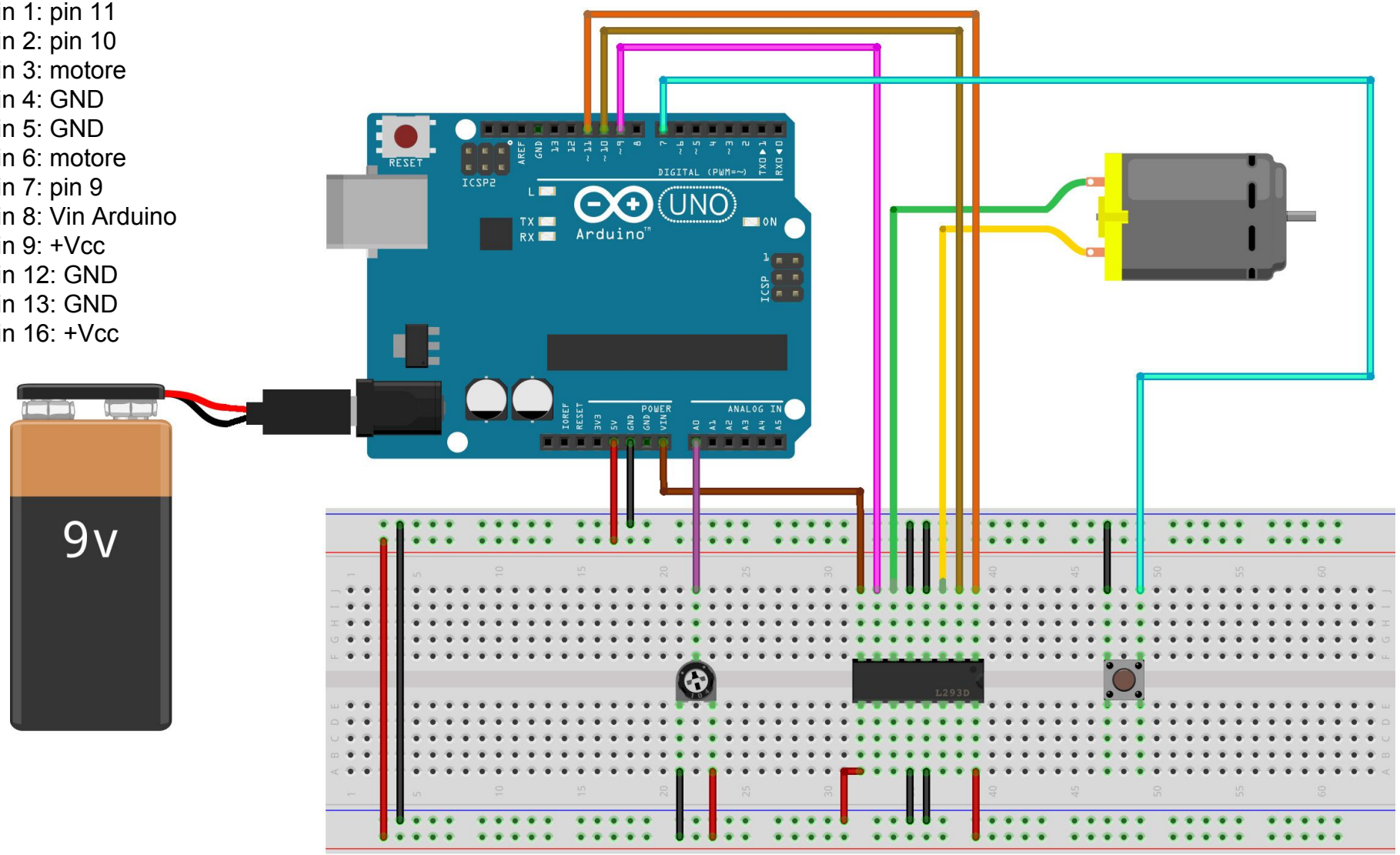
Si utilizzi un trimmer da 10KOhm



Controllo di un motore DC mediante pulsante e trimmer | 2/4 | sketch48

Collegamenti
L293D > Arduino

pin 1: pin 11
pin 2: pin 10
pin 3: motore
pin 4: GND
pin 5: GND
pin 6: motore
pin 7: pin 9
pin 8: Vin Arduino
pin 9: +Vcc
pin 12: GND
pin 13: GND
pin 16: +Vcc



Prof. Michele Maffucci

Controllo di un motore DC mediante pulsante e trimmer

3/4

sketch48

```
/* Prof. Michele Maffucci
   19.04.2014

   Utilizzo del ponte H L293D
   per pilotare mediante pulsante e trimmer
   direzione e velocita' di rotazione
   di un motorino in CC

   Collegamenti
   L293D > Arduino

   pin 1: pin 11
   pin 2: pin 10
   pin 3: motore
   pin 4 - pin 5 - pin 12 - pin 13: GND
   pin 6: motore
   pin 7: pin 9
   pin 8: Vin
   pin 9 - pin 16: +Vcc
*/

const int pinEnableA = 11;       // collegato al pin 1 dell'L293D
const int pinIngresso1 = 10;     // collegato al pin 2 dell'L293D
const int pinIngresso2 = 9;      // collegato al pin 7 dell'L293D
const int cambioDirezione = 7;   // collegato al pulsante
const int pinTrimmer = 0;        // collegato al trimmer

void setup()
{
  pinMode(pinIngresso1, OUTPUT);  // pin di controllo senso di rotazione - motore 1
  pinMode(pinIngresso2, OUTPUT);  // pin di controllo senso di rotazione - motore 1
  pinMode(pinEnableA, OUTPUT);    // pin enable - motore 1
  pinMode(cambioDirezione, INPUT_PULLUP); // abilitazione della resistenza di pull-up
}
```

ATTENZIONE!
Ricordarsi di abilitare la resistenza di pull-up interna.

continua...

Controllo di un motore DC mediante pulsante e trimmer

sketch48

```
void loop()
{
  int velocita = analogRead(pinTrimmer);                // lettura dell'impostazione dal trimmer
  int velocitaMotore = map(velocita, 0, 1023, 0, 255);  // rimappiamo i valori in un intervallo idoneo per il PWM
  boolean retromarcia = digitalRead(cambioDirezione);   // viene conservato il valore imposto dal pulsante
  impostaMotore(velocitaMotore, retromarcia);           // funzione che imposta velocita' e direzione del motore
}

void impostaMotore(int velocita, boolean retromarcia)
{
  analogWrite(pinEnableA, velocita);                    // viene passato il valore del PWM
  digitalWrite(pinIngresso1, ! retromarcia);            // inversione di marcia
  digitalWrite(pinIngresso2, retromarcia);              // inversione di marcia
}
```

Grazie

Prof. Michele Maffucci

www.maffucci.it
michele@maffucci.it

www.twitter.com/maffucci/
www.facebook.com/maffucci.it/
plus.google.com/+MicheleMaffucci/

it.linkedin.com/in/maffucci