

# Controllare un Servo Motore tramite Joystick

**Obiettivo:** Come controllare due servomotori utilizzando un Joystick per Arduino.

## **Pre-requisiti**

[\*Il Servomotore\*](#)

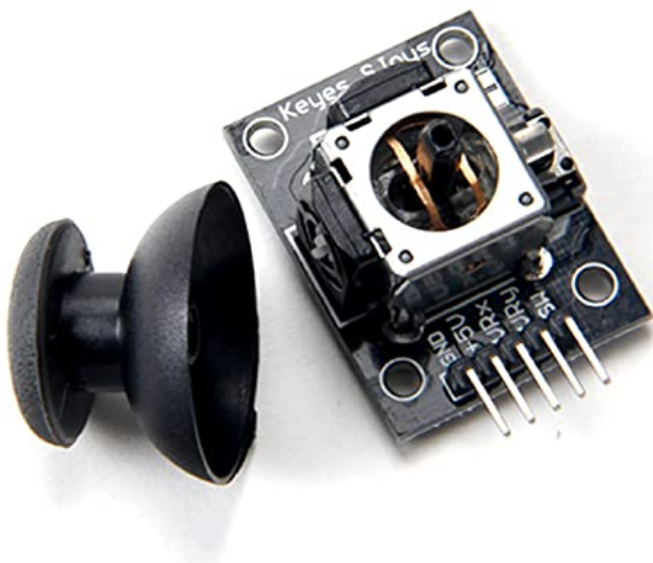
## **Componenti elettronici:**

- Arduino UNO
- Breadboard
- 2 Servomotori
- 1 Joystick KY-023K

**Teoria:** Il **Joystick** è un dispositivo elettronico che trasforma i movimenti di una leva manovrata dall'utente in una serie di segnali elettrici, questi segnali possono essere in seguito utilizzati per controllare un software (e.g., videogame), un'apparecchiatura o un attuatore meccanico. Possono esistere due differenti tipologie di Joystick:

- Joystick Digitale: Rileva solamente la direzione dell'inclinazione della leva.
- Joystick Analogico: Rileva anche l'ampiezza dell'inclinazione.

Nello specifico in questo articolo viene illustrato il funzionamento di uno dei controller più tipicamente utilizzati e presenti nei vari kit Arduino: il "Dual Axis Joystick Module **KY-023**", Questo dispositivo, basato sul controller della PlayStation2, utilizza due potenziometri bi-assiali per controllare l'asse X e l'asse Y. Inoltre è possibile premere il controller per attivare uno switch. Nello specifico, la tensione di funzionamento del dispositivo è compresa nel range 3.3 – 5 V. Mentre le dimensioni sono pari a 2.6 x 3.4 cm.



### Dual Axis Joystick Module **KY-023**

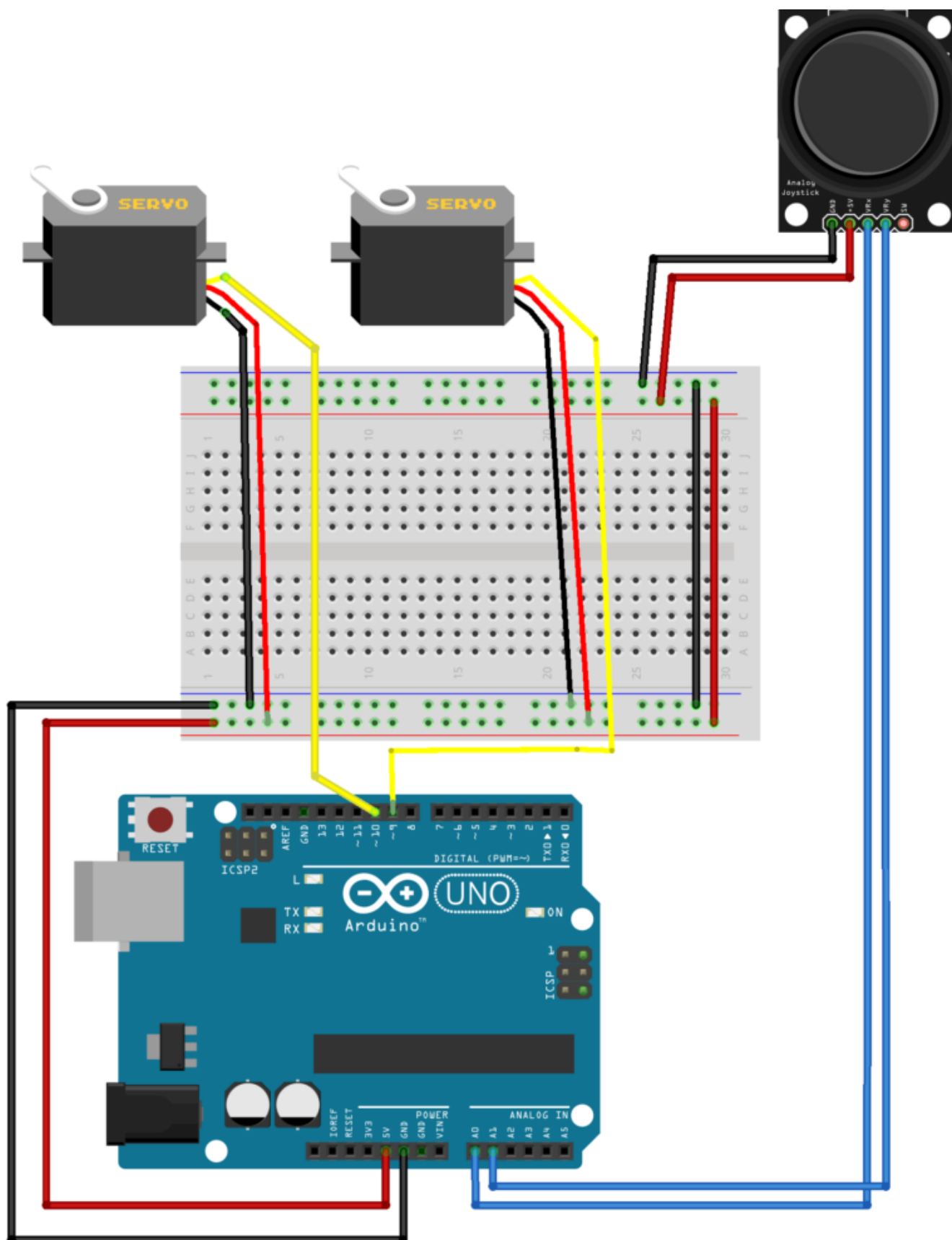
Nella tabella a seguire è riportata la piedinatura utile per collegare in modo corretto il Joystick KY-023.

<b>KY-023 PIN</b>	<b>Descrizione</b>
GND	Ground

KY-023 PIN	Descrizione
+5V	Alimentazione Vcc
VRx	Uscita Analogica (Asse X)
VRy	Uscita Analogica (Asse Y)
SW	Switch

In questo articolo il Joystick viene utilizzato per comandare due differenti servomotori. Un servo è associato all'asse X ed un altro è invece associato all'asse Y. La posizione di riposo dei due servomotori è per entrambi 90 gradi. Spostando il Joystick lungo l'asse X si può modificare la posizione del servo associato all'asse X di un angolo variabile da 0 a 180 gradi. Lo stesso accade modificando la posizione del joystick lungo l'asse y.

### **Collegamento Circuitale:**



fritzing

Codice:

---

# Pilotare un Servo Motore tramite Potenzziometro

**Obiettivo:** Ruotare un Servo Motore tramite un potenziometro

**Pre-requisiti**

[\*Il Servomotore\*](#)

**Componenti elettronici:**

- Arduino UNO
- Breadboard
- 1 Servomotore
- 1 Potenzziometro

**Teoria:** Il Servomotore è uno particolare tipo motore ampiamente utilizzato sia in contesti industriali sia nell'ambito del modellismo. Nel dettaglio, il servomotore è impiegato in tutte le applicazioni che prevedono il controllo della posizione di un motore in corrente continua ed il

raggiungimento di un determinato angolo in modo preciso indipendentemente dalla posizione iniziale. Le caratteristiche principali del servomotore sono:

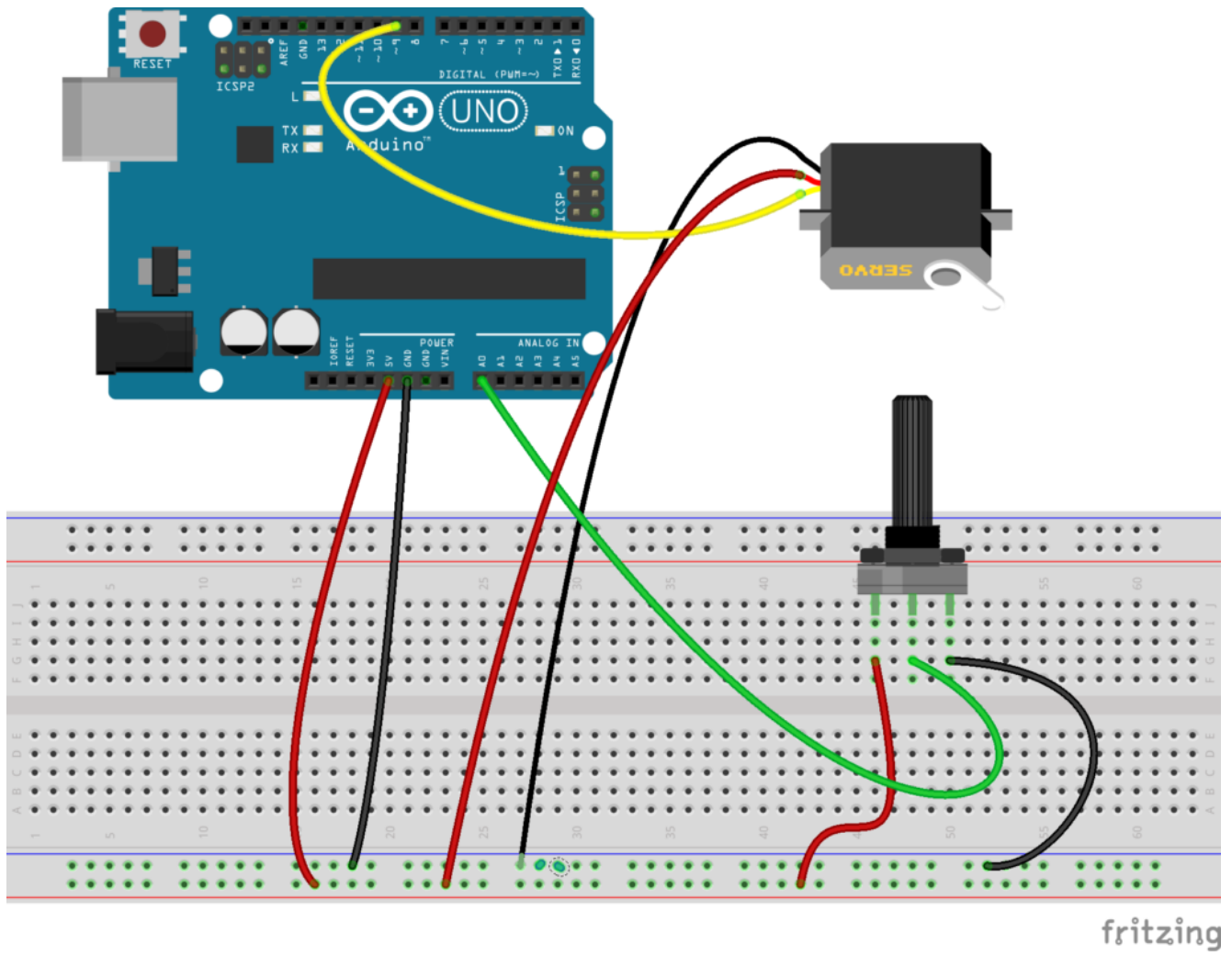
Come già accennato, i servo motori sono dispositivi molto utilizzati in svariati ambiti perché permettono la rotazione del proprio albero in base ad un angolo prestabilito.

I servomotori sono stati utilizzati per la prima volta nel mondo del modellismo RC, generalmente per controllare lo sterzo delle auto RC o i flap su un aereo RC o per aprire botole su un drone. Con il tempo, hanno trovato il loro uso anche nella robotica, nell'automazione e in svariati progetti Arduino.

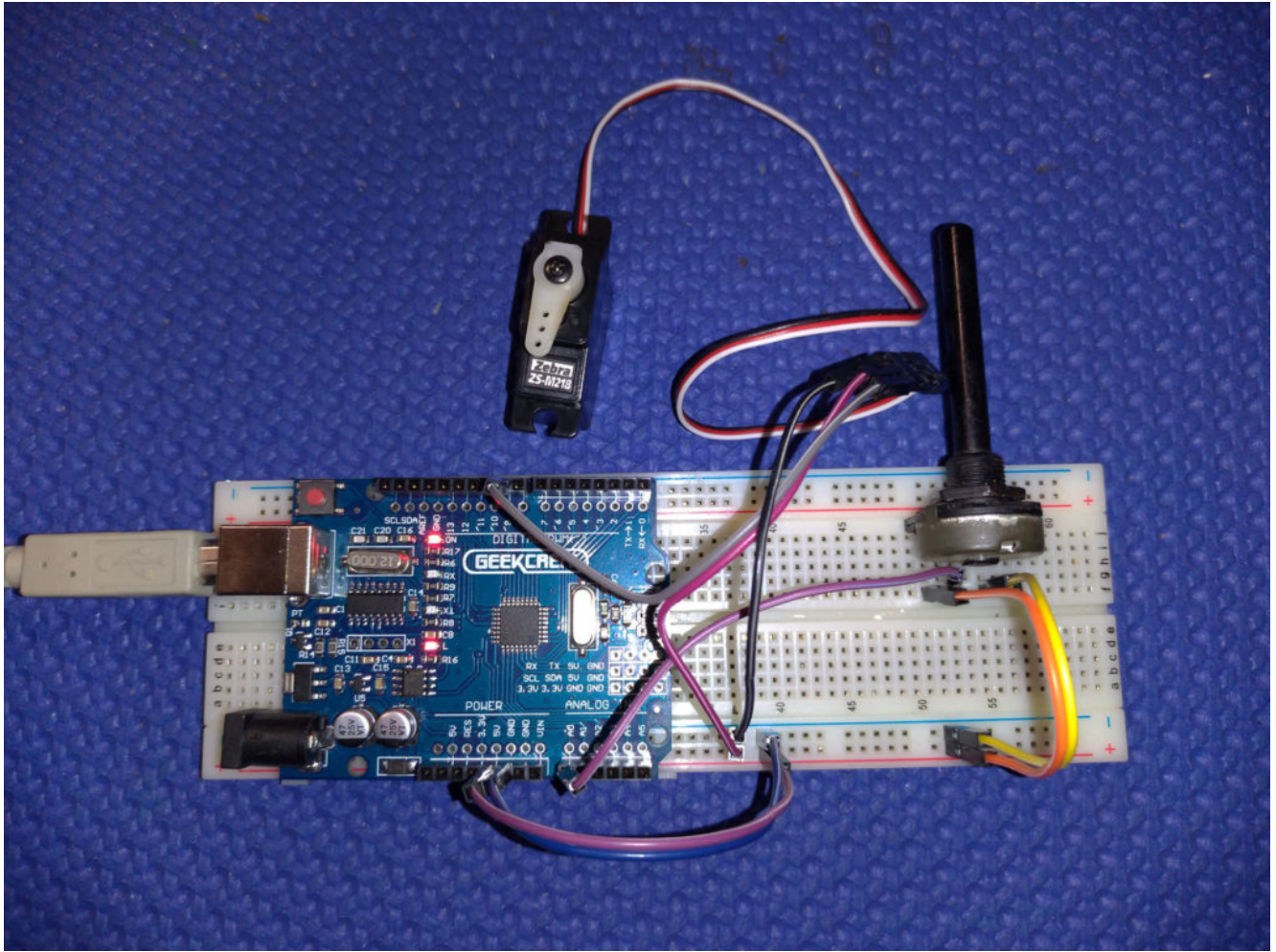
Solitamente l'albero può ruotare da 0 a 180 gradi e usando Arduino, possiamo dire a un servo di andare in una posizione specificata

In questo tutorial vedremo come interconnettere il servo motore ad Arduino e come farlo ruotare tramite un potenziometro con pochissime istruzioni.

**Collegamento Circuitale:**



**Risultato:**



**Codice:**

---

# Il Servomotore

**Obiettivo:** Semplice comando di un servomotore



## **Componenti elettronici:**

- Arduino UNO
- Breadboard
- 1 Servomotore

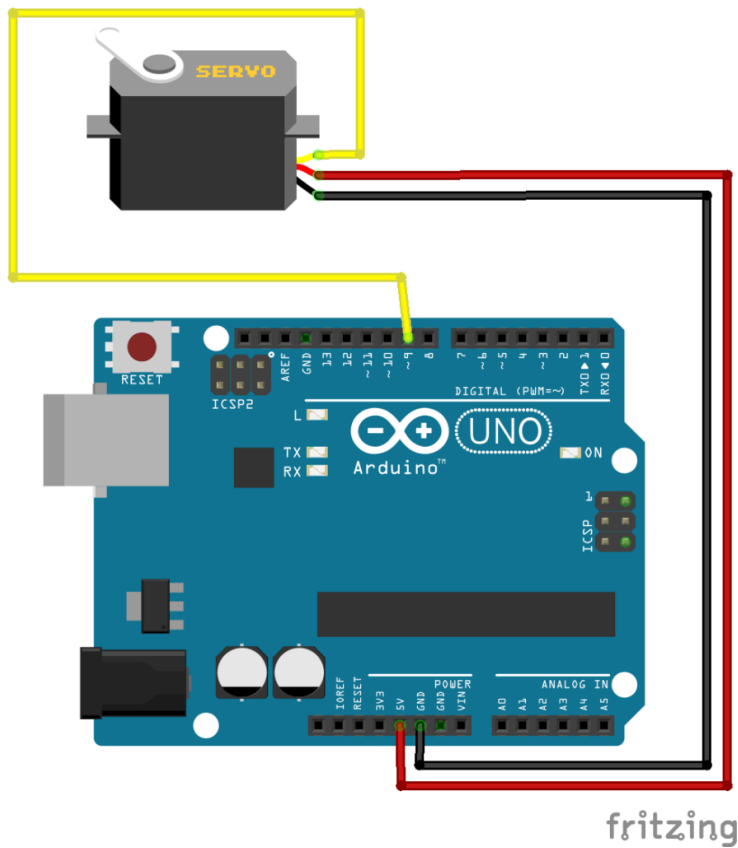
**Teoria:** Il Servomotore è uno particolare tipo motore ampiamente utilizzato sia in contesti industriali sia nell'ambito del modellismo. Nel dettaglio, il servomotore è impiegato in tutte le applicazioni che prevedono il controllo della posizione di un motore in corrente continua ed il raggiungimento di un determinato angolo in modo preciso indipendentemente dalla posizione iniziale. Le caratteristiche principali del servomotore sono:

- Tensione di alimentazione
- Coppia Massima (espressa in Kg)
- Angolo di rotazione

I servomotori sono caratterizzati da tre cavi che devono essere opportunamente collegati ad Arduino:

- Il cavo di alimentazione positiva (+)
- Il ground (-)
- Il controllo

## **Collegamento Circuitale:**



Collegamento Circuitale

**Codice:**

**Personalizzazioni:** E' possibile modificare la velocità e l'angolo di rotazione del servomotore intervenendo direttamente sulle variabili in gioco.

---

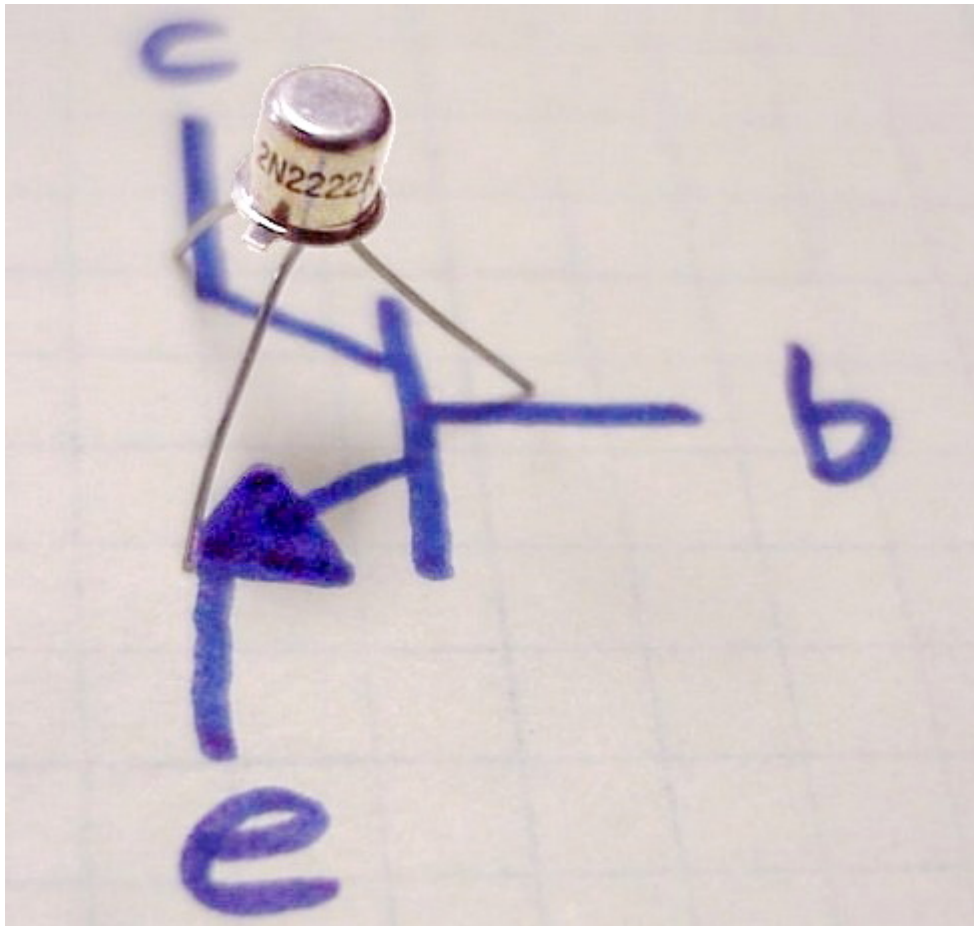
# Azionare un Motore a CC con il Transistor

**Obiettivo:** Azionare un motore a corrente continua tramite transistor

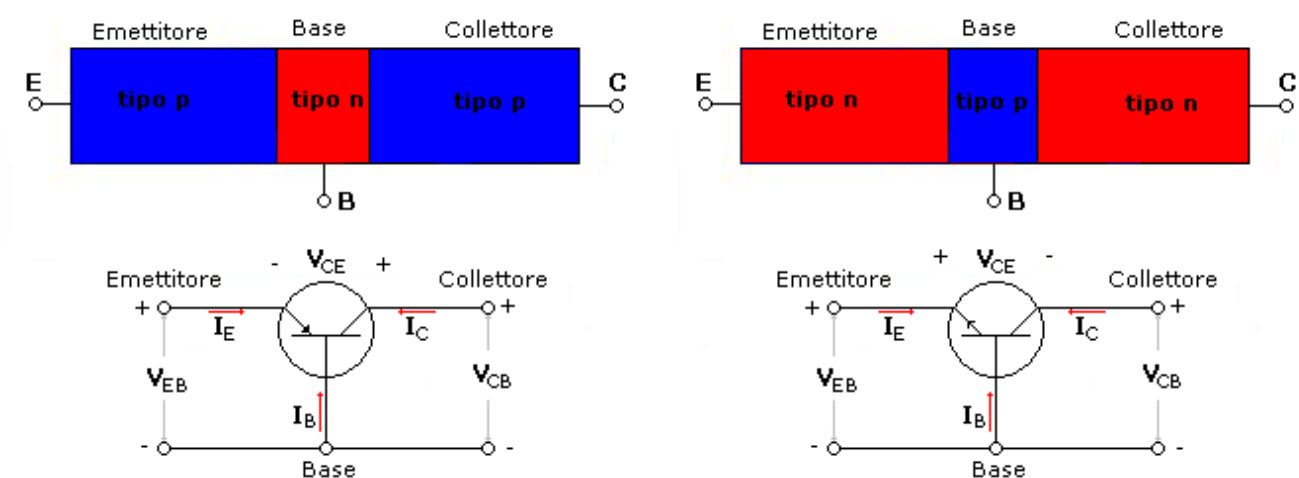
## **Componenti elettronici:**

- Arduino UNO
- Breadboard
- 1 [Transistor p2n2222a](#)
- 1 Motore a DC 3V – 6V 17000 RPM
- 1 Resistenza da 1K0hm

**Teoria:** Il **Transistor** è un elemento elettronico utilizzato come amplificatore di corrente o interruttore. E' costruito da tre strati di materiale semiconduttori uniti con una doppia giunzione p-n, tipica dei diodi. Ad ogni strato è collegato un terminale: quello centrale si chiama *Base*, e quelli esterni *Elettore* e *Collettore*. Il principio di funzionamento è basato sulla possibilità di controllare il passaggio di corrente tra collettore ed elettore, tramite un impulso elettrico fornito alla base.



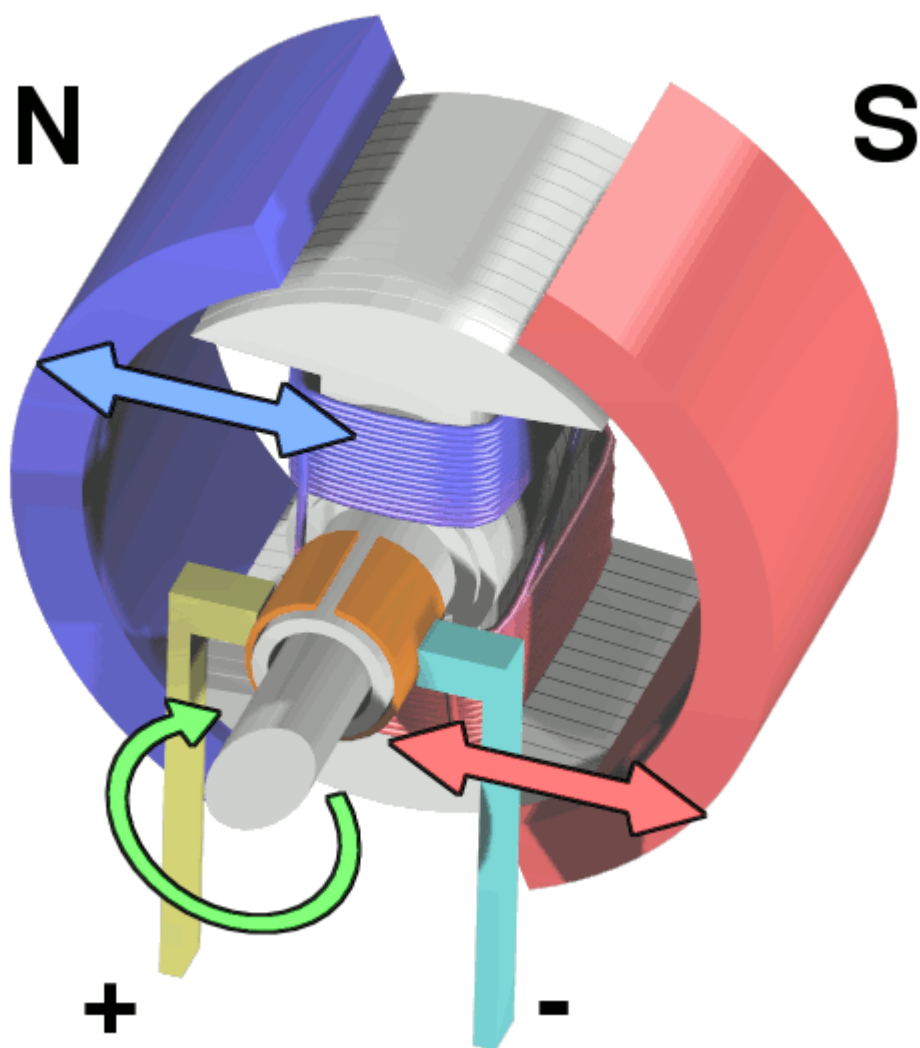
Esistono due tipologie di transistor, a seconda di come sono costruiti: Transistor PNP e NPN; l'unica differenza funzionale tra un transistor PNP e un transistor NPN è la polarità delle giunzioni durante il funzionamento



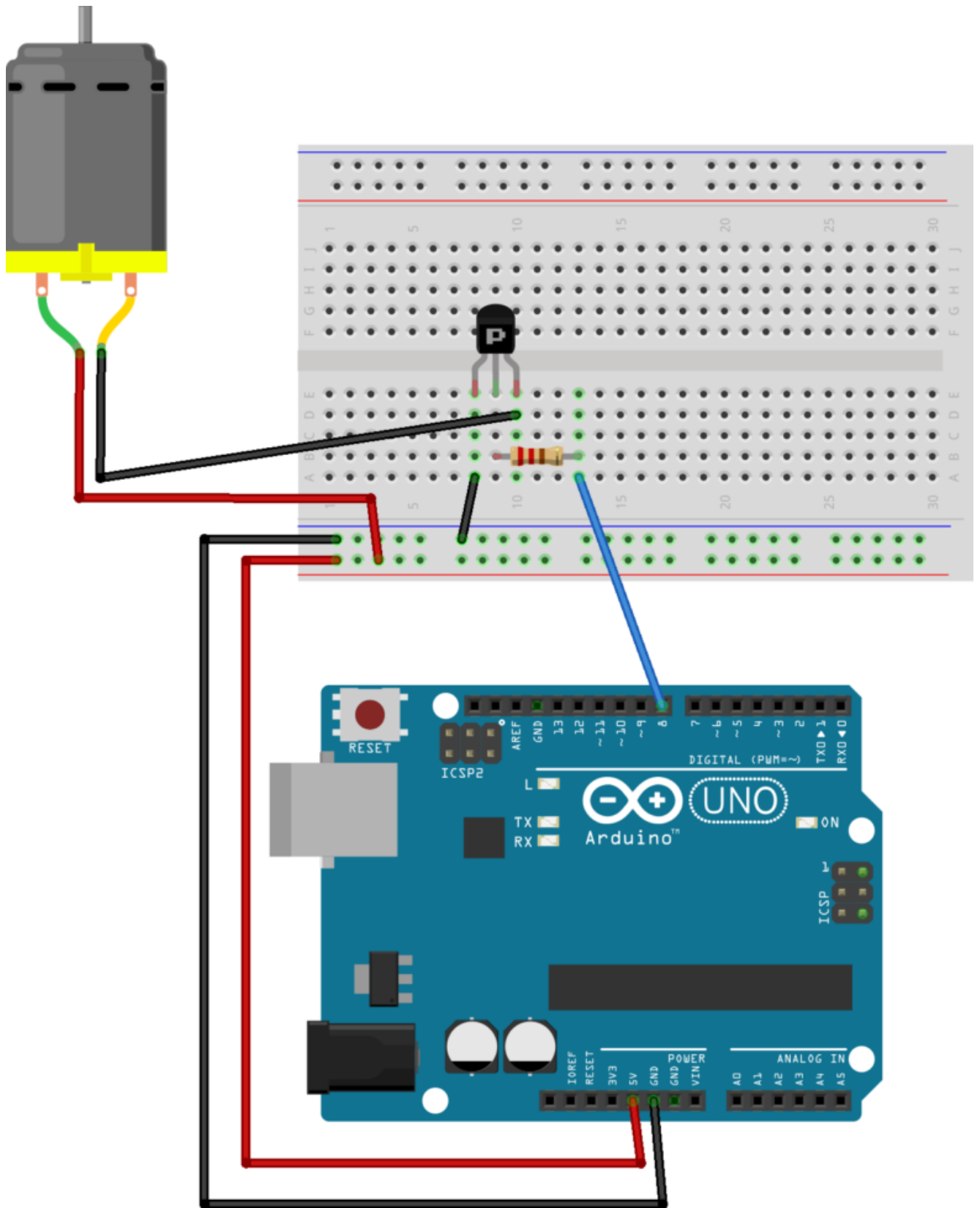
Transistor bipolare a giunzione tipo p-n-p e n-p-n con rappresentazione circuitale.

Tramite i transistor è possibile controllare attuatori che necessitano di grandi correnti, utilizzando una piccola tensione sulla base. In pratica basta collegare un ramo della corrente (ad esempio la terra) sul collettore e sull'emettitore del transistor e gestire il collegamento tramite la base.

**Motori a Corrente Continua (DC):** sono costituiti, al loro interno, da un magnete permanente e da un'elettrocalamita che viene alimentata da dei contatti striscianti, che, per come sono montanti, invertono la tonalità dell'elettrocalamita ad ogni mezzo giro, mantenendo in rotazione l'asse.



Collegamento Circuitale:



fritzing

**Codice:**

**Personalizzazioni:** E' possibile collegare un sensore di temperatura per far attivare il motore quando l'ambiente si surriscalda.

---

# Come Collegare un Display LCD ad Arduino

**Obiettivo:** Utilizzare un Display LCD 16x2 (basato su un Driver Hitachi HD44780).

## **Componenti elettronici:**

- Arduino UNO
- Breadboard
- 1 Display LCD (e.g., 1602A) compatibile con Driver Hitachi HD44780
- 1 Potenzziometro da 10 kOhm
- 1 Resistenza da 220 Ohm

**Teoria:** I display basati su Driver Hitachi HD44780 sono tra i più diffusi in ambito embedded. Caratterizzati da differenti formati (i.e., 8x1, 8x2, 16x2, 20x2, 16x3, etc), questi display permettono di visualizzare solo caratteri. Nella seguente tabella si riporta la descrizione dei **PIN** del



dispositivo:

<b>PIN</b>	<b>Descrizione</b>
1	Vss (Massa)
2	Vcc (Genericamente 5 V)
3	Vee (Controllo contrasto, collegato in genere ad un potenziometro con tensione che varia da 0 a 5v)
4	R/S (0 per selezionare l'invio di un comando, 1 per i dati)
5	R/W (0 per selezionare la scrittura di dati o comandi, 1 per la lettura dei dati o dello stato)
6	E (inizia il ciclo di scrittura o lettura, secondo R/S e R/W)
7	D0 (Bus dati)
8	D1 (Bus dati)
9	D2 (Bus dati)
10	D3 (Bus dati)
11	D4 (Bus dati)
12	D5 (Bus dati)
13	D6 (Bus dati)
14	D7 (Bus dati)
15	A (Vcc retroilluminazione, se presente)
16	K (Vss retroilluminazione, se presente)

Il Driver HD44780 si basa su una modalità di trasferimento dati di tipo parallelo. Nel dettaglio è supportato sia il trasferimento di 8 bit (l'intero comando D0-D7) sia il trasferimento di 4 bit (D4-D7). Nel secondo caso, per trasmettere un byte vengono effettuati due trasferimenti.

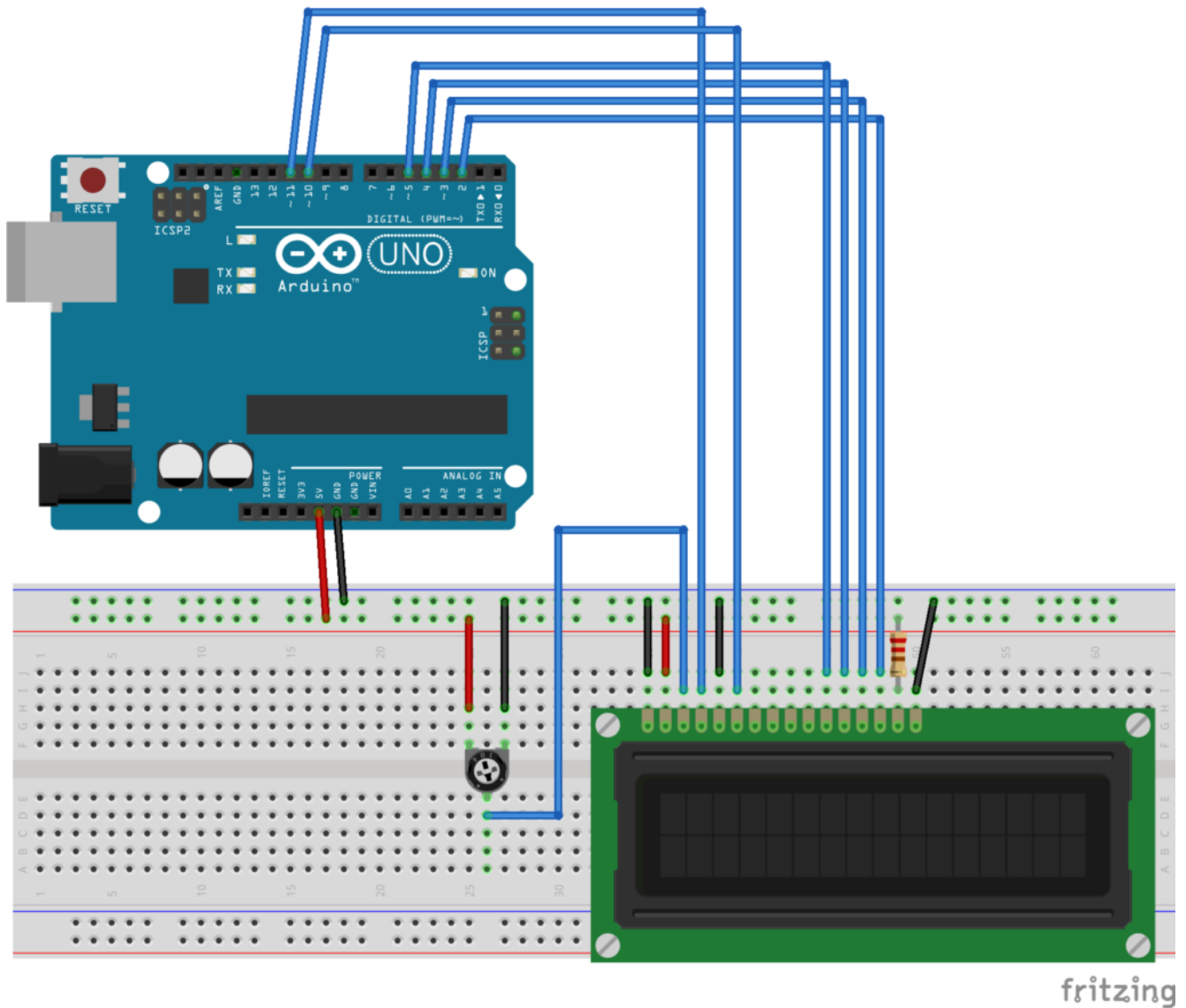
Per la scrittura di un singolo carattere è necessario rispettare il seguente **“protocollo di trasmissione”**:

- Posizionare il cursore nella posizione desiderata
- Impostare a 1 il pin R/S e a 0 il pin R/W
- Inviare il codice ASCII del carattere
- Impostare ad 1 il pin E per un minimo di 450 nanosecondi ed, in seguito, riportarlo a 0

Da un punto di vista pratico, grazie all'impiego della specifica **libreria [LiquidCrystal](#)** inclusa nel pacchetto software di Arduino è possibile pilotare il display semplificando notevolmente la parte di gestione dei pin, dei comandi e delle relative temporizzazioni. La libreria mette infatti a disposizione una classe **LiquidCrystal** all'interno della quale sono definite le principali funzioni necessarie per utilizzare i display basati su Driver HD44780. Quali:

- **Begin**: Inizializza l'interfaccia del display LCD specificandone le dimensioni (larghezza ed altezza)
- **SetCursor**: Posiziona il cursore LCD ovvero la posizione nella quale verrà visualizzato il testo scritto
- **Print**: Scrive il testo sul display LCD.

**Collegamento Circuitale:**



## Codice:

**Personalizzazioni:** E' possibile modificare il contrasto del display intervenendo sul potenziometro.

---

# Creare funzioni con Arduino ... per un Display a 7 Segmenti

**Obiettivo:** Creare una serie di funzioni con Arduino per utilizzare un display a 7 segmenti .

## **Componenti elettronici:**

- Arduino UNO
- Breadboard
- 1 Display a 7 Segmenti
- 7 Resistenza (100 Ohm)

## **Pre-Requisiti:**

[1..2..3.. Il Display a 7 Segmenti](#)

**Teoria:** La possibilità di riutilizzare il codice precedentemente scritto (**code re-use**) rappresenta una delle pratiche più comuni nella programmazione ovvero richiamare/invocare parti di codice precedentemente già

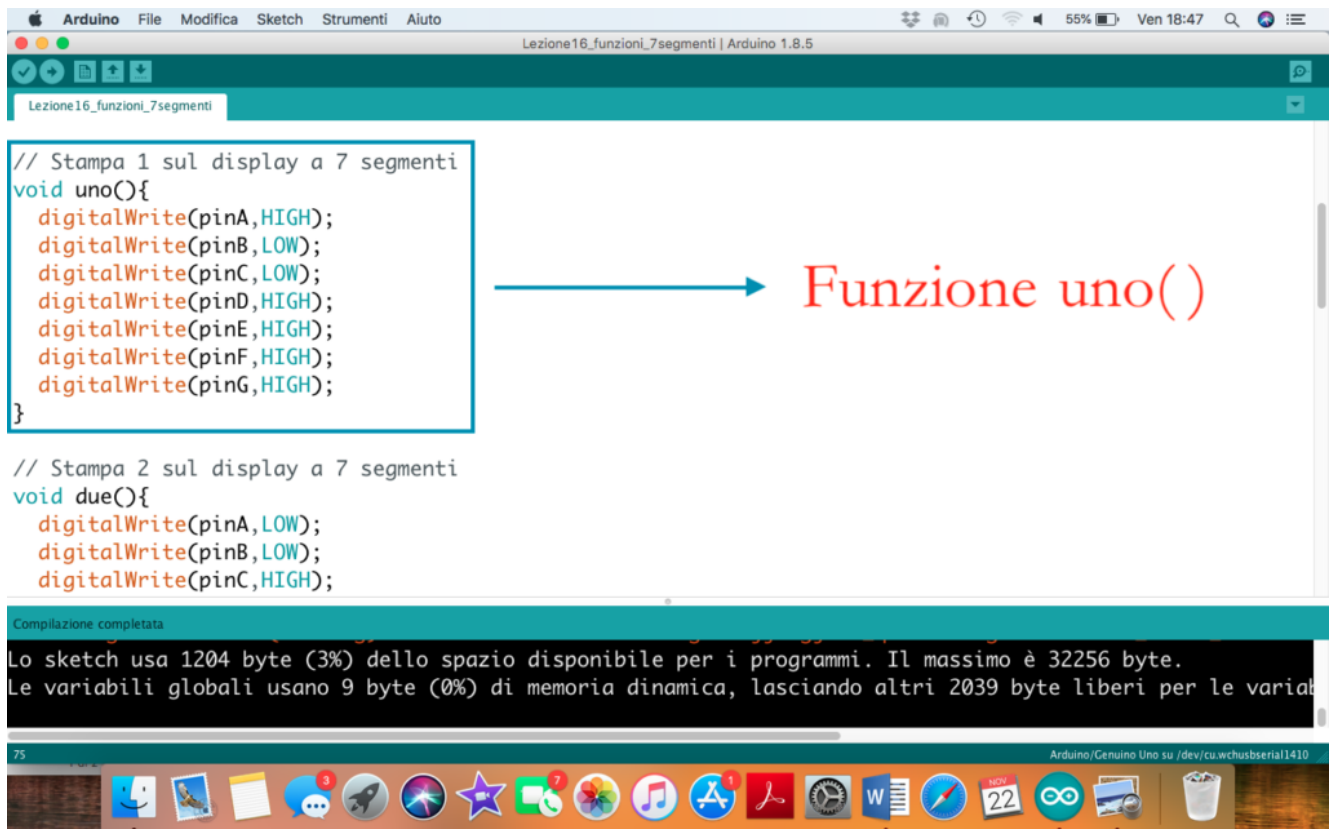
sviluppate, ogni qualvolta risulta necessario, senza doverle riscrivere daccapo.

Nello specifico, questa possibilità si concretizza attraverso la scrittura di funzioni che possono essere richiamate/invocate all'interno dello stesso programma.

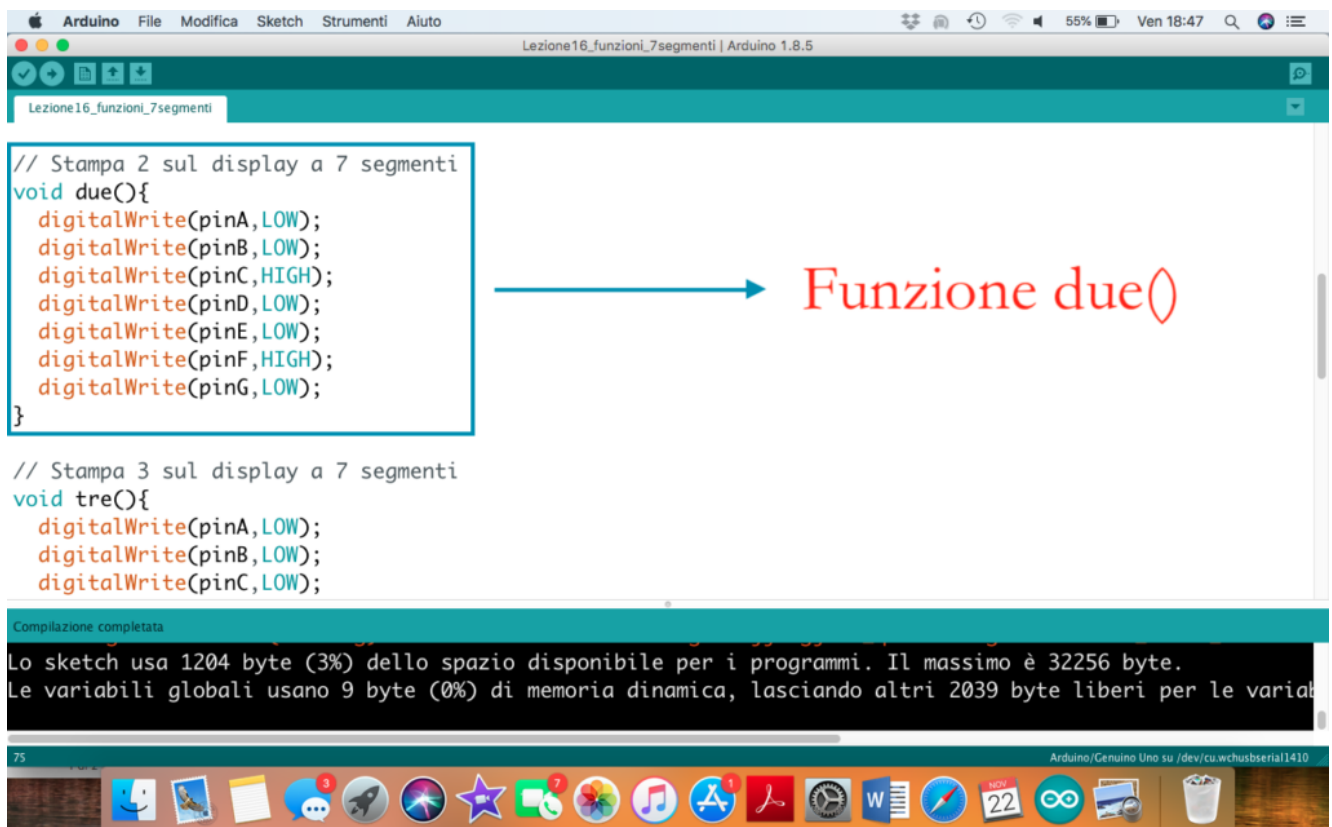
Queste funzioni possono svolgere diverse operazioni, che permettono di manipolare degli input dati ed eventualmente restituire degli output desiderati.

Nel caso in questione le tre funzioni implementate (denominate, uno, due e tre) non prevedono né il passaggio specifico di parametri di input né la restituzione di un output dato; queste tre funzioni svolgono solamente una serie di azioni sequenziali volte ad accendere alcuni elementi di un display a 7 segmenti. E' importante considerare che le tre funzioni sono tutte definite prima del loro utilizzo (ovvero prima delle due funzioni principali di setup e loop).

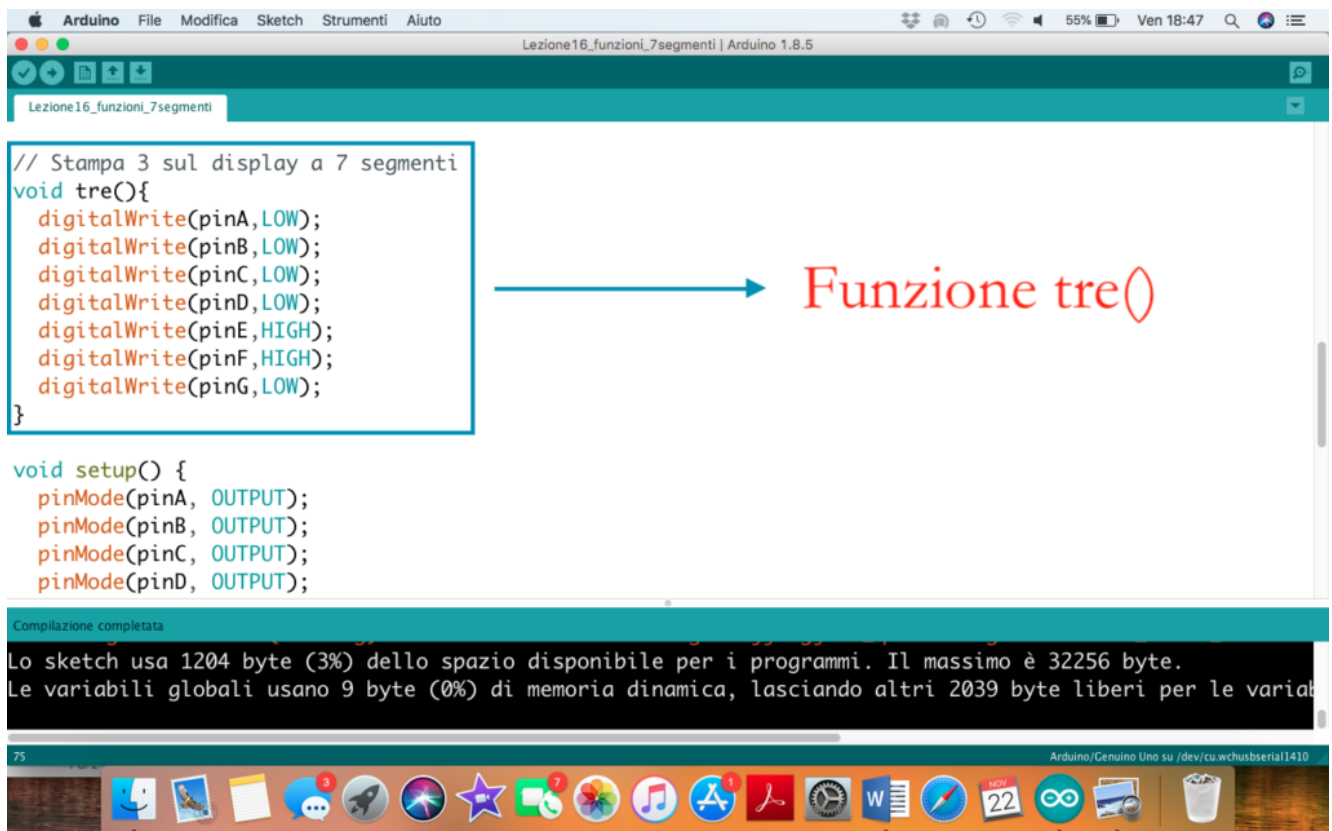
- **Funzione Uno():** Stampa il carattere 1 sul display a 7 segmenti



- **Funzione Due():** Stampa il carattere 2 sul display a 7 segmenti



- **Funzione Tre():** Stampa il carattere 3 sul display a 7 segmenti



```
// Stampa 3 sul display a 7 segmenti
void tre(){
  digitalWrite(pinA,LOW);
  digitalWrite(pinB,LOW);
  digitalWrite(pinC,LOW);
  digitalWrite(pinD,LOW);
  digitalWrite(pinE,HIGH);
  digitalWrite(pinF,HIGH);
  digitalWrite(pinG,LOW);
}

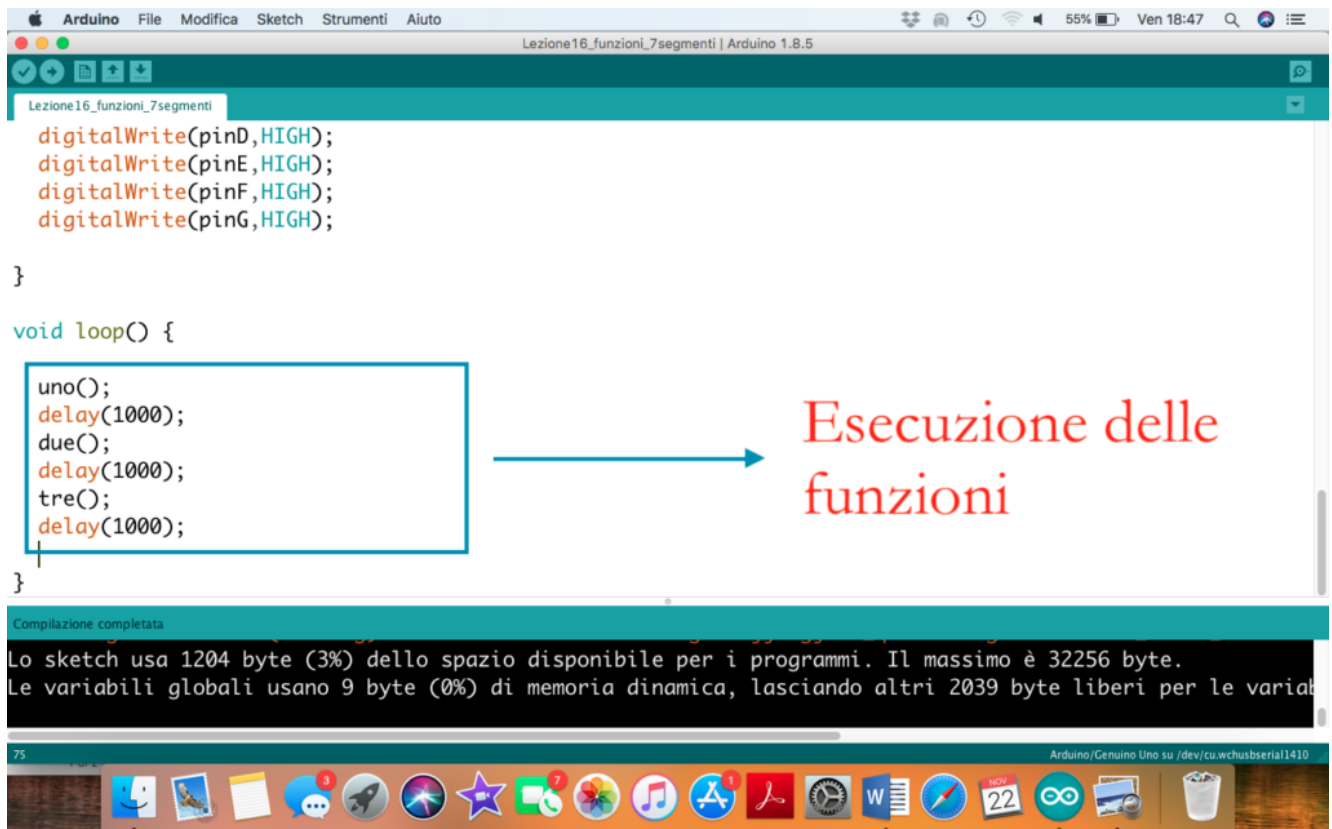
void setup() {
  pinMode(pinA, OUTPUT);
  pinMode(pinB, OUTPUT);
  pinMode(pinC, OUTPUT);
  pinMode(pinD, OUTPUT);
}
```

Funzione tre()

Compilazione completata

Lo sketch usa 1204 byte (3%) dello spazio disponibile per i programmi. Il massimo è 32256 byte.  
Le variabili globali usano 9 byte (0%) di memoria dinamica, lasciando altri 2039 byte liberi per le variabili locali.

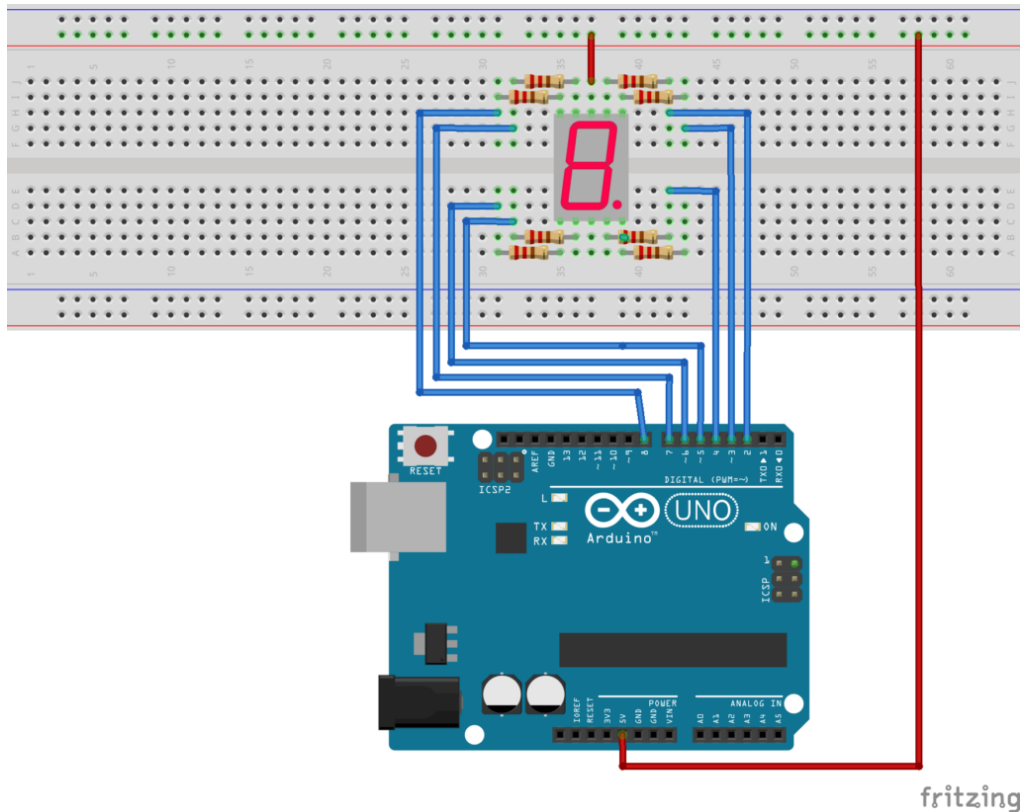
- **Invocazione delle Funzioni:** Le funzioni possono essere invocate semplicemente richiamandole nel punto in cui devono essere eseguite. Nel caso in questione le funzioni sono richiamate all'interno del blocco loop().



## Collegamento Circuitale:

Viene in seguito riportato lo schema elettrico utilizzato per comandare un display a 7 segmenti mediante funzioni. Nel dettaglio il display impiegato è della modalità anodo comune.





## Collegamento Circuitale

### Codice:

Considerando l'impiego di un display ad anodo comune i singoli LED possono essere accesi mediante l'istruzione `digitalWrite(pinLed,LOW);`

Tre differenti funzioni sono state create per gestire il display. Le funzioni sono, rispettivamente denominate, `uno()`, `due()` e `tre()`. Queste funzioni devono essere definite prima dei blocchi `setup` e `loop`, non prevedono l'impiego di parametri in ingresso né la restituzione di output in uscita (funzioni di tipo `void`).

## **Personalizzazioni:**

E' possibile modificare il software aggiungendo altre funzioni per la rappresentazione dei vari numeri sul display a 7 segmenti.

---

# **1..2..3.. Il Display a 7 Segmenti**

**Obiettivo:** Utilizzo di un display a 7 segmenti .

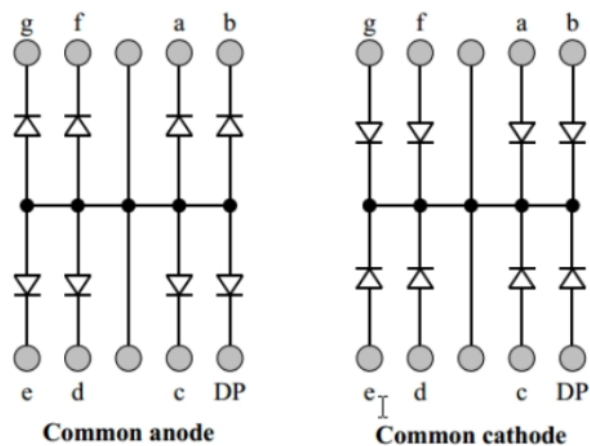
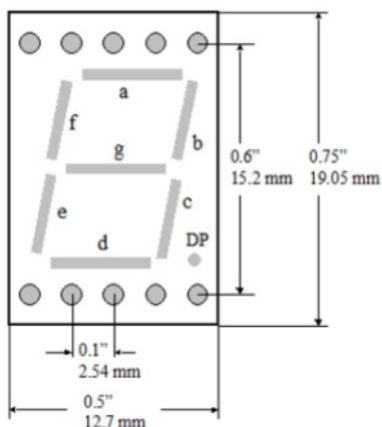
## **Componenti elettronici:**

- Arduino UNO
- Breadboard
- 1 Display a 7 Segmenti
- 7 Resistenza (100 Ohm)

**Teoria:** Il display a 7 segmenti è un dispositivo elettronico in grado di visualizzare, attraverso l'accensione di combinazioni di sette differenti segmenti luminosi, le 10 cifre numeriche, alcune lettere alfabetiche e simboli grafici.

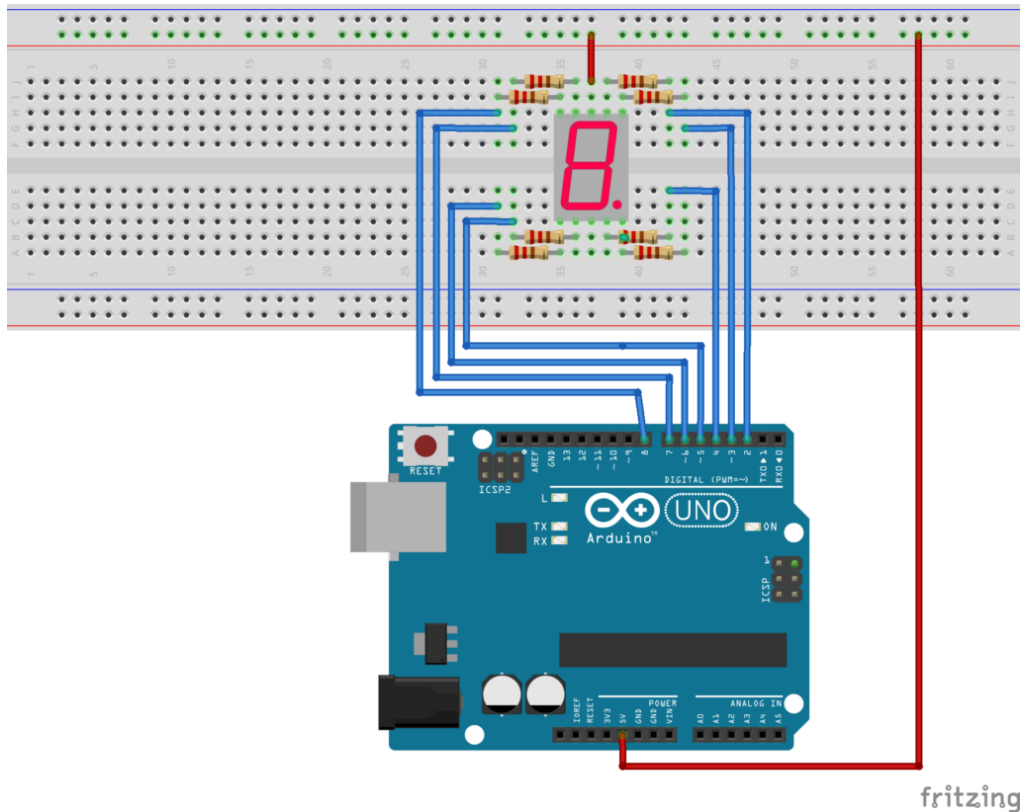
I display a 7 segmenti sono dotati di 10 differenti pin e sono così classificati:

- **Display 7 segmenti ad Catodo Comune:** il pin centrale del LED deve essere collegato a massa (GND) mentre gli altri 8 differenti PIN sono utilizzati per comandare i 7 segmenti ed il punto DP. Nel caso in questione i singoli LED sono accesi attraverso l'istruzione Arduino `digitalWrite(pinLed, HIGH);`
- **Display 7 segmenti ad Anodo Comune:** il pin centrale del LED deve essere collegato a 5V (VCC) mentre gli altri 8 differenti PIN sono utilizzati per comandare i 7 segmenti ed il punto DP. Nel caso in questione i singoli LED sono accesi attraverso l'istruzione Arduino `digitalWrite(pinLed, LOW);`



### Collegamento Circuitale:

Viene in seguito riportato lo schema elettrico utilizzato per comandare un display a 7 segmenti. Nel dettaglio il display impiegato è della modalità anodo comune.



Collegamento Circuitale

### **Codice:**

Considerando l'impiego di un display ad anodo comune i singoli LED possono essere accesi mediante l'istruzione `digitalWrite(pinLed,LOW);`

### **Personalizzazioni:**

E' possibile modificare il codice aggiungendo la possibilità di visualizzare altri numeri e non solo 1, 2 e 3.

---

# Il Sensore di Presenza

**Obiettivo:** Utilizzare un Sensore di presenza PIR.

## **Componenti elettronici:**

- Arduino UNO
- Breadboard
- 1 Led
- 1 Sensore PIR

**Teoria:** Il **sensore ad infrarossi passivo** è un dispositivo elettronico che misura i raggi infrarossi irradiati dai vari oggetti nel suo campo di vista. Per questo motivo, questo dispositivo è ampiamente utilizzato come rilevatore di movimento partendo dal presupposto che ogni persona irradia energia nello spettro dell'infrarosso. Partendo da questo presupposto è importante considerare che un sensore PIR, a differenza di un sensore ad ultrasuoni non emette onde (rileva soltanto informazioni nello spettro infrarosso), inoltre è capace di misurare una variazione dell'energia associata al movimento di un oggetto/persona.

Nel caso specifico il sensore PIR proposto nell'attività è un **HC-SR501**.

In seguito sono riportate le principali caratteristiche

tecniche di questo sensore:

- Tensione di alimentazione 5-20V
- Corrente assorbita 65mA
- Tensione in uscita 0-3,3V
- Range di sensibilità: meno di 120 gradi per 7 metri

Sono inoltre presenti due differenti trimmer per personalizzare le caratteristiche del dispositivo:

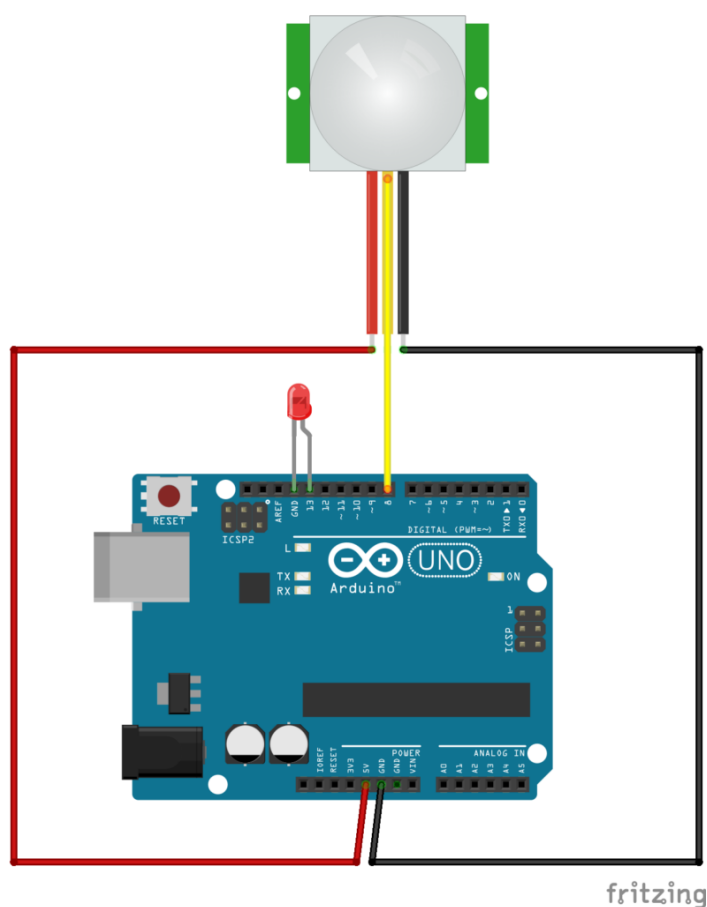
- Modifica la sensibilità legata alla distanza: ruotando in senso orario la distanza aumenta fino ad un massimo di 7 metri, differentemente, ruotando in senso antiorario il potenziometro la distanza diminuisce fino ad un minimo di 3 metri.
- Modifica la sensibilità legata al tempo per il quale il segnale di uscita rimane alto: ruotando in senso orario il tempo aumenta (fino ad un massimo di 5 minuti), differentemente, ruotando in senso antiorario il tempo diminuisce (fino ad un minimo di tre secondi).

Nel sensore è inoltre presente un Jumper che permette di impostare due differenti modalità di funzionamento:

- **H (Hold/Repeat/Retriggering)**: In questa posizione il sensore continuerà a mantenere il livello del segnale in uscita HIGH fintanto che il movimento continuerà ad essere percepito.
- **L (Intermittent or No-Repeat/Non-Retriggering)**: In questa posizione il sensore continuerà a mantenere il livello del segnale in uscita HIGH per il tempo definito attraverso il potenziometro.

Il sensore di presenza **HC-SR501** è costituito da 3 pin. Un pin di alimentazione (5v), un pin di ground, ed il pin che riporta l'eventuale presenza di un oggetto (da collegare ad un pin di input digitale di Arduino).

### Collegamento Circuitale:



Collegamento Circuitale

### Codice:

A seguire viene riportato il codice necessario per l'utilizzo del sensore PIR utilizzato per realizzare il programma.

## Personalizzazioni:

E' possibile modificare il circuito introducendo un relè indispensabile per comandare una "lampada reale" e non un semplice led.

---

# Collegare più Sensori Ultrasuoni ad Arduino

**Obiettivo:** Utilizzare simultaneamente due sensori a ultrasuoni (HC-SR04). Nel dettaglio, al fine di dimostrare il corretto funzionamento del circuito, ad ogni sensore è associato un led. Quando la distanza misurata dal sensore è inferiore ad una data soglia, il led associato si accende, mentre quando la distanza è superiore il led si spegne.

## Componenti elettronici:

- Arduino UNO
- Breadboard
- 2 Sensori Ultrasuoni (e.g., HC-SR04)
- 2 Led (utilizzati per dimostrare il funzionamento del circuito)
- 2 Resistenze (100 Ohm) (indispensabili per il corretto funzionamento dei led)



## Pre-Requisiti:

### [Il Sensore a Ultrasuoni](#)

**Teoria:** Il sensore di prossimità è un dispositivo che permette di rilevare la presenza di oggetti nelle immediate vicinanze, senza che vi sia un effettivo contatto.

Nel caso specifico, il sensore di prossimità ad ultrasuoni sfrutta il principio del Sonar. Degli impulsi sonori (ultrasonici) vengono emessi dal dispositivo il quale attraverso l'eventuale eco di ritorno permette di rilevare la presenza di un oggetto all'interno della portata nominale. Esempi pratici di sensori ad ultrasuoni sono i sensori di retromarcia e di parcheggio utilizzati nelle moderne automobili.

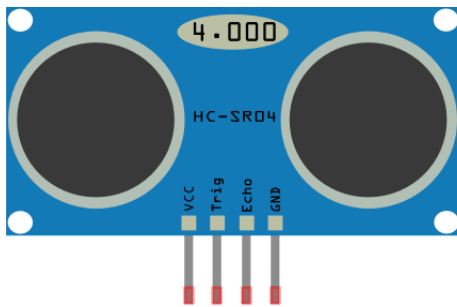
Nel dettaglio, le principali caratteristiche tecniche del sensore ad ultrasuoni HC-SR04 sono:

- Alimentazione: +5V DC
- Angolo di misura:  $< 30^\circ$
- Distanza di rilevamento: da 2cm a 400cm
- Risoluzione: 1cm
- Frequenza: 40kHz

Il sensore, dispone di 4 pin: Vcc (+5V), Trigger, Echo, GND.

- VCC: +5V DC
- Trigger: Genera l'impulso ultrasonico
- Echo: Rileva il segnale ultrasonico di ritorno

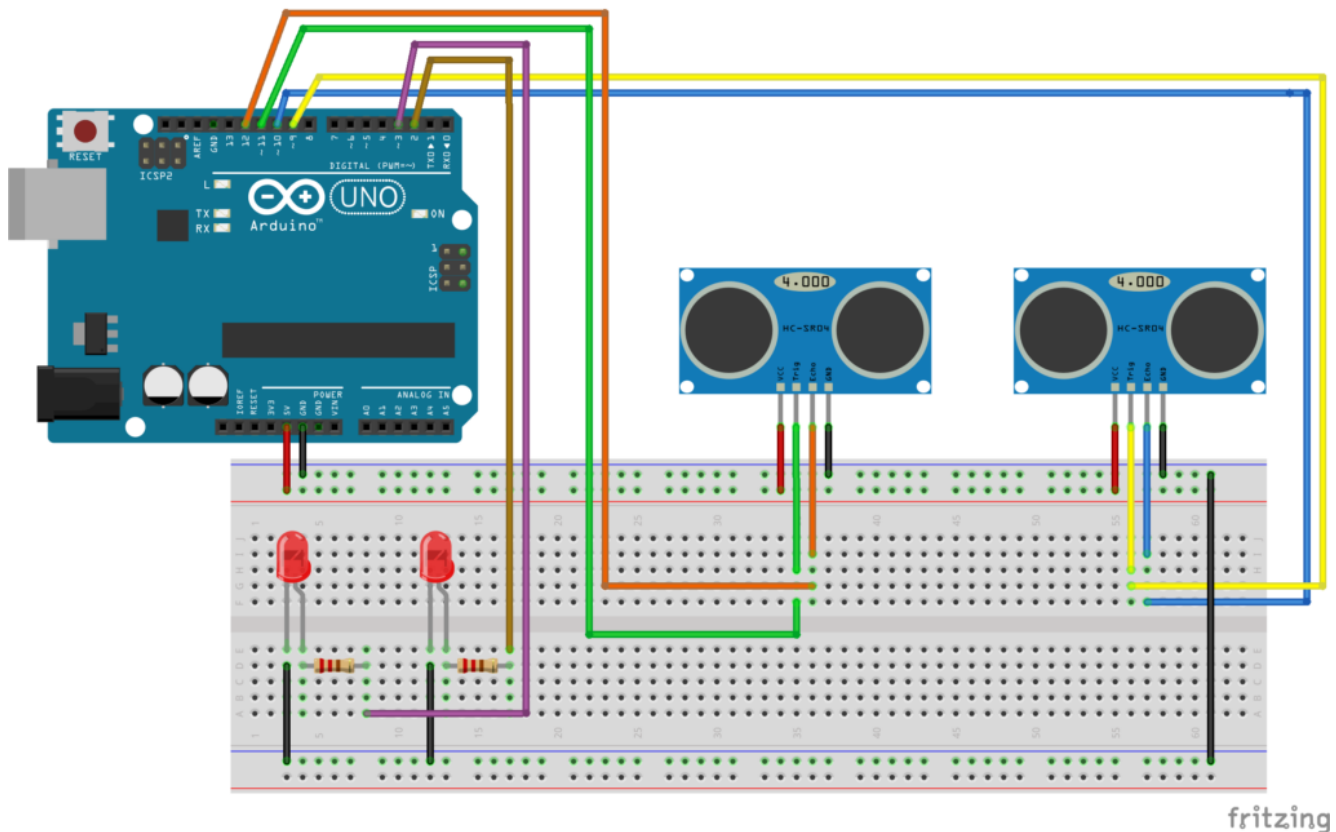
- GND: 0V Ground



Modello Sensore HC-SR04

### Collegamento Circuitale:

La realizzazione hardware di circuiti che utilizzano più sensori ad ultrasuoni è relativamente facile. Ognuno dei pin di Echo e di Trig del sensore deve essere semplicemente collegato ai pin di input/output di Arduino.



### **Codice:**

Nel caso specifico, in cui si vogliano utilizzare più sensori ad ultrasuoni per misurare simultaneamente la distanza in più punti è opportuno gestire i sensori in modo corretto attraverso il relativo codice. Nel dettaglio, è importante considerare che inizialmente entrambi i sensori generano l'impulso utilizzato per determinare la distanza. In seguito le letture di echo (effettuate mediante l'istruzione pulseIn) sono eseguite in modo sequenziale (una dopo l'altra).

---

# Il Sensore a Ultrasuoni

**Obiettivo:** Utilizzare un Sensore a Ultrasuoni (HC-SR04) per misurare la distanza.

### **Componenti elettronici:**

- Arduino UNO
- Breadboard
- 1 Sensore Ultrasuoni (e.g., HC-SR04)

**Teoria:** Il sensore di prossimità è un dispositivo che permette di rilevare la presenza di oggetti nelle immediate vicinanze, senza che vi sia un effettivo contatto.

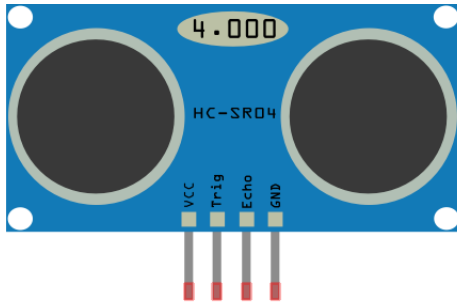
Nel caso specifico, il sensore di prossimità ad ultrasuoni sfrutta il principio del Sonar. Degli impulsi sonori (ultrasonici) vengono emessi dal dispositivo il quale attraverso l'eventuale eco di ritorno permette di rilevare la presenza di un oggetto all'interno della portata nominale. Esempi pratici di sensori ad ultrasuoni sono i sensori di retromarcia e di parcheggio utilizzati nelle moderne automobili.

Nel dettaglio, le principali caratteristiche tecniche del sensore ad ultrasuoni HC-SR04 sono:

- Alimentazione: +5V DC
- Angolo di misura:  $< 30^\circ$
- Distanza di rilevamento: da 2cm a 400cm
- Risoluzione: 1cm
- Frequenza: 40kHz

Il sensore, dispone di 4 pin: Vcc (+5V), Trigger, Echo, GND.

- VCC: +5V DC
- Trigger: Genera l'impulso ultrasonico
- Echo: Rileva il segnale ultrasonico di ritorno
- GND: 0V Ground



Modello Sensore HC-SR04

Pertanto considerando la formula che lega velocità, spazio e tempo:

$$s = v * t$$

e la velocità del suono pari a 343 m/s (che espressa in microsecondi diventa 0,0343 m/uS), si ottiene:

$$s = 0,0343 * t$$

Considerando inoltre che il suono percorrerà due volte la distanza da misurare (dal sensore all'oggetto e dall'oggetto al sensore); il tempo  $t$  ottenuto deve essere diviso per due, ottenendo:

$$s = 0,0343 * (t/2)$$

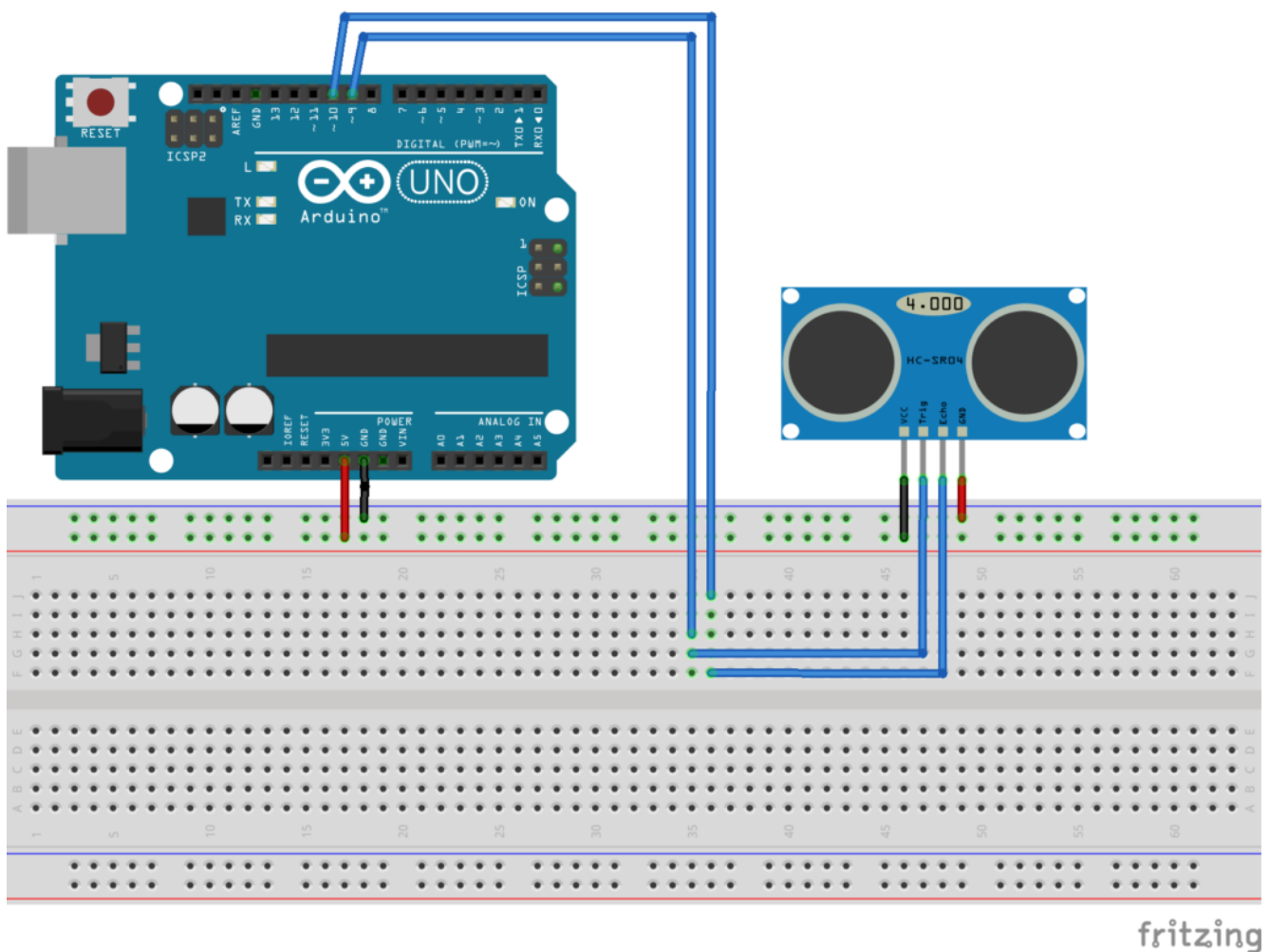
$$s = 0,01715 * t$$

$$s = t/58,31$$

Per valutare la presenza di un oggetto è necessario rispettare il seguente “**protocollo**”:

- Il PIN Trigger deve essere dettato alto (HIGH value) per almeno 10microsecondi.
- In automatico il modulo HC-SR04 invierà 8 impulsi ultrasonici ad una frequenza pari a 40kHz.
- Il PIN Echo viene posto in ascolto. Calcolando il tempo di arrivo/ritorno dell’impulso ultrasonico.

### Collegamento Circuitale:



Collegamento Circuitale

Codice:

[crayon-664d494579cf3731605984/]

**Personalizzazioni:** E' possibile modificare il contrasto del display intervenendo sul potenziometro.