

Misura di Temperatura mediante TMP36 [Tinkercad]

Obiettivo: Realizzare un controllo di temperatura mediante il dispositivo TMP36. Il TMP36 è il sensore di temperatura presente sul simulatore tinkercad.

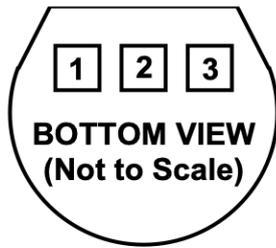
Componenti elettronici:

- Arduino UNO
- Breadboard
- TMP36

Teoria: Il componente elettronico TMP36 è un dispositivo integrato ad alta precisione utilizzato per misurare la temperatura ambientale.

Dato il basso costo e l'ampia scala di valori ammissibili (ovvero da -40°C fino a 125°C) questi dispositivi sono particolarmente diffusi. E' inoltre importante considerare che non è necessaria nessuna operazione calibrazione per ottenere valori di accuratezza pari a $\pm 1^{\circ}\text{C}$ ad una temperatura di circa $+25^{\circ}\text{C}$ e $\pm 2^{\circ}\text{C}$ nel range di temperature -40°C to $+125^{\circ}\text{C}$.

Questo dispositivo è caratterizzato da tre differenti pin ed un corpo semi-cilindrico. Guardando il lato piatto del dispositivo, il pin di sinistra è l'alimentazione (5V), il pin di destra la massa (GND), mentre sul pin centrale viene generata una tensione funzione della temperatura. La temperatura può essere pertanto misurata attraverso una lettura analogica sul pin centrale effettuata mediante il controllore Arduino.



PIN 1, $+V_S$; PIN 2, V_{OUT} ; PIN 3, GND

00337-004

Figure 4. T-3 (TO-92)

TMP36 Package

Come riportato in precedenza è possibile utilizzare un pin di input analogico per ottenere il valore di temperatura mediante l'istruzione di `analogRead`. Nel caso specifico, osservando il grafico che riporta la caratteristica tensione/temperatura (per il **TMP36** la linea è evidenziata in rosso) per una tensione di uscita di 0.5V il sensore rileva la temperatura di 0°C. Pertanto valori di tensione inferiori a 0.5V indicano una temperatura sotto lo zero, mentre valori di tensione superiori a 0.5V indicano una temperatura positiva. Inoltre, è importante considerare che "una variazione di grado corrisponde ad una variazione di tensione di 10mV". Quindi, se sul pin di input analogico sono presenti 550mV significa che il sensore sta rilevando una temperatura di 5°C (550mV - 500mV = 50 mV variazione di 5°C).

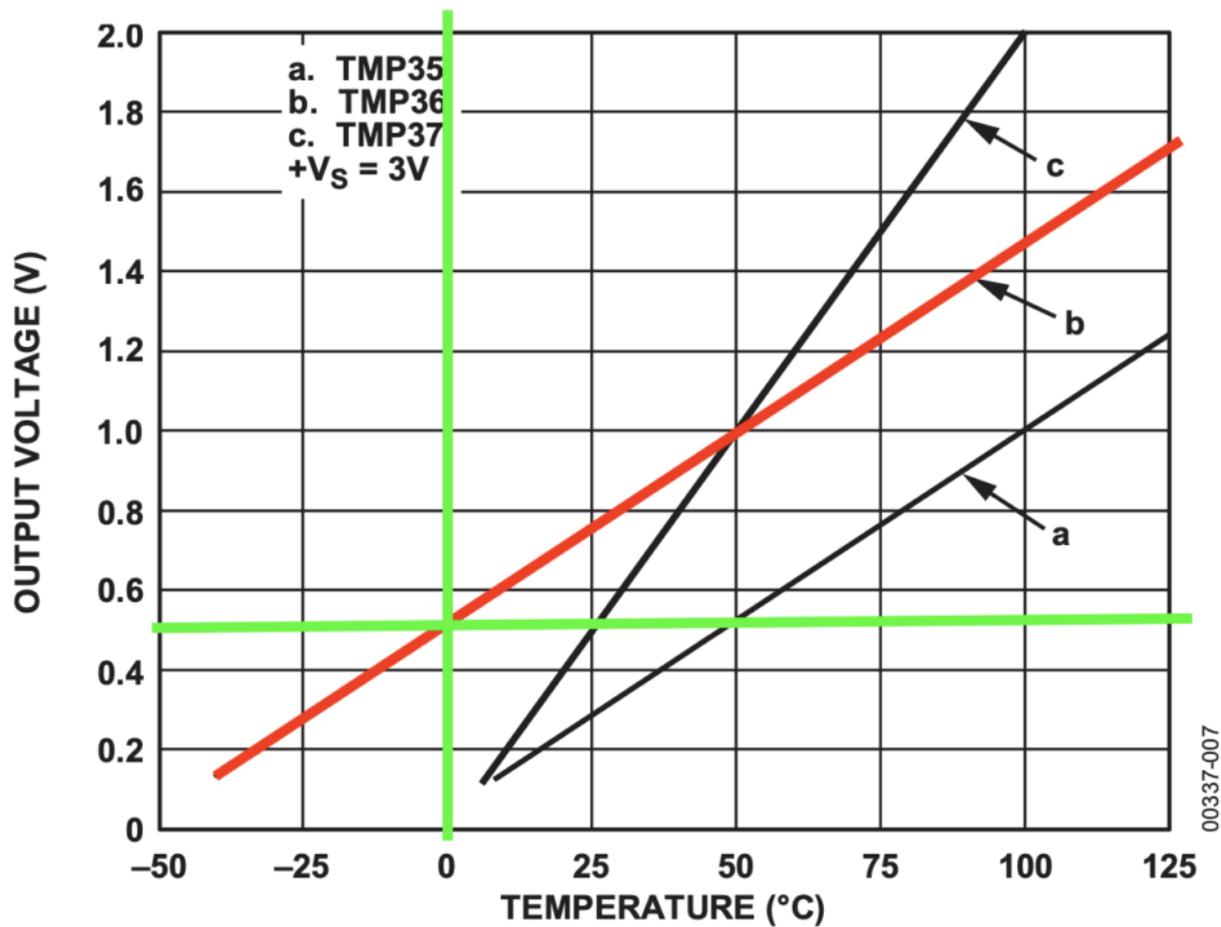


Figure 6. Output Voltage vs. Temperature

Caratteristica tensione corrente

Collegamento Circuitale:

Viene in seguito riportato lo schema elettrico utilizzato per valutare la temperatura mediante il dispositivo elettronico TMP36

variabile denominata *valTMP* la temperatura può essere ottenuta mediante la seguente formula:

$$temperatura = (((valTMP5.0)/1023.0)-0.5)*100$$

Tinkercad

Personalizzazioni:

E' possibile modificare il circuito aggiungendo una ventola che si accende in modo automatico superata una determinata temperatura.

Controllo di Temperatura mediante LM35

Obiettivo: Realizzare un controllo di temperatura mediante il dispositivo LM35.

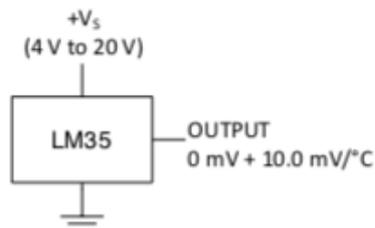
Componenti elettronici:

- Arduino UNO
- Breadboard
- LM35

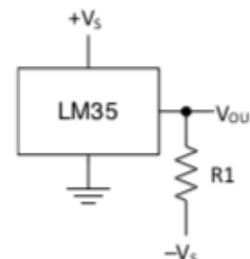
Teoria: Il componente elettronico LM35 è un dispositivo integrato ad alta precisione utilizzato per misurare la temperatura ambientale.

Dato il basso costo e l'ampia scala di valori ammissibili (ovvero da -55°C fino a 150°C) questi dispositivi sono particolarmente diffusi. E' inoltre importante considerare che e nessun tipo di calibrazione esterna è richiesta.

**Basic Centigrade Temperature Sensor
(2°C to 150°C)**



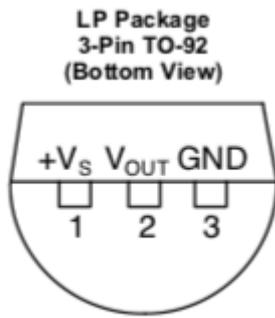
Full-Range Centigrade Temperature Sensor



Choose $R_1 = -V_S / 50\ \mu\text{A}$
 $V_{OUT} = 1500\text{ mV}$ at 150°C
 $V_{OUT} = 250\text{ mV}$ at 25°C
 $V_{OUT} = -550\text{ mV}$ at -55°C

Estratto Datasheet LM35

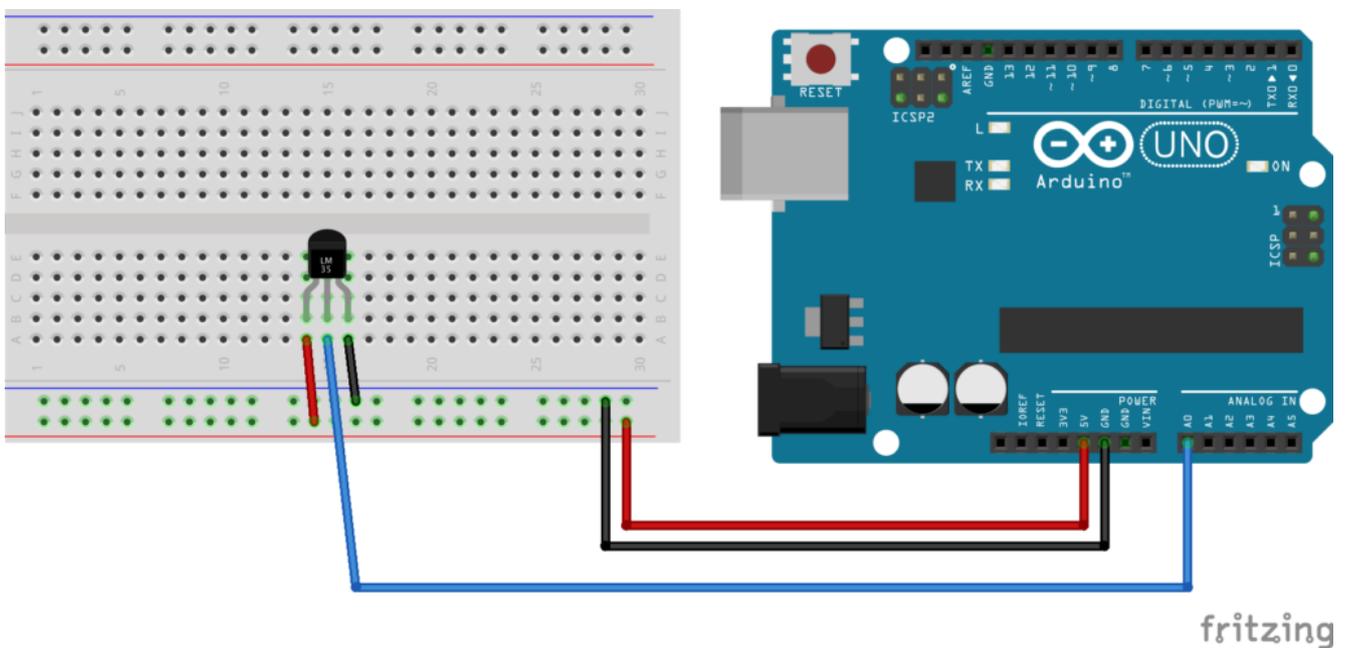
Questo dispositivo è caratterizzato da tre differenti pin ed un corpo semi-cilindrico. Guardando il lato piatto del dispositivo, il pin di sinistra è l'alimentazione (5V), il pin di destra la massa (GND), mentre sul pin centrale viene generata una tensione funzione della temperatura (10mV per ogni grado sopra lo zero). La temperatura può essere pertanto misurata attraverso una lettura analogica sul pin centrale effettuata mediante il controllore Arduino.



LM35 Package

Collegamento Circuitale:

Viene in seguito riportato lo schema elettrico utilizzato per valutare la temperatura mediante il dispositivo elettronico LM35



Collegamento Circuitale

Codice:

La temperatura viene determinata attraverso una lettura

analogica ed una opportuna scalatura.

Nel dettaglio, la tensione prodotta dal componente LM35 viene letta utilizzando il pin analogico A0 e mappata in un intervallo 0-1023. Considerando che per ogni grado percepito si ha un incremento di tensione di 10mV è opportuno riportare il valore letto attraverso la funzione analogRead in mV. Questa operazione può essere svolta dividendo il valore letto per 1023 e moltiplicando il risultato per 5000. I gradi sono infine ottenuti dividendo il risultato per 10.

Personalizzazioni:

E' possibile modificare il circuito aggiungendo una ventola che si accende in modo automatico superata una determinata temperatura.

Controllare un Servo Motore tramite Joystick

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

Pre-requisiti

[Il Servomotore](#)

Componenti elettronici:

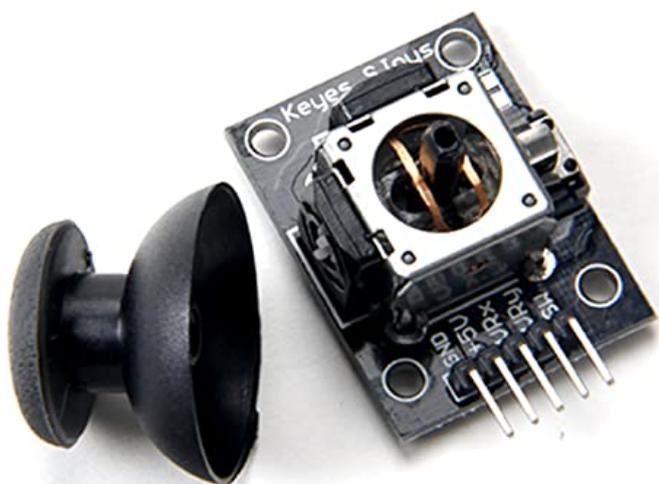
- Arduino UNO
- Breadboard
- 2 Servomotori
- 1 Joystick KY-023K

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**

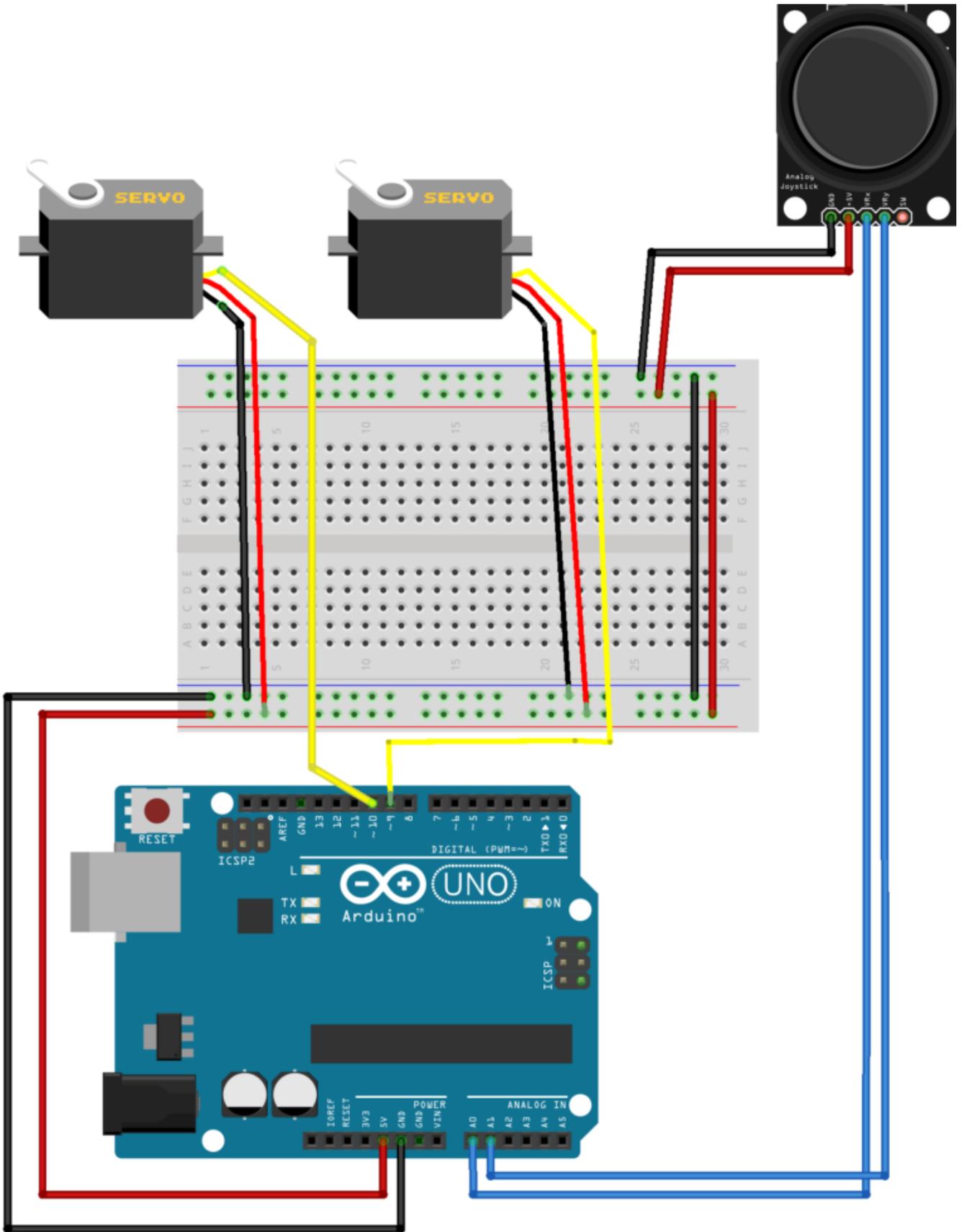
Nella tabella a seguire è riportata la piedinatura utile per collegare in modo corretto il Joystick KY-023.

KY-023 PIN	Descrizione
GND	Ground
+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 in motore in corrente continua ed il

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

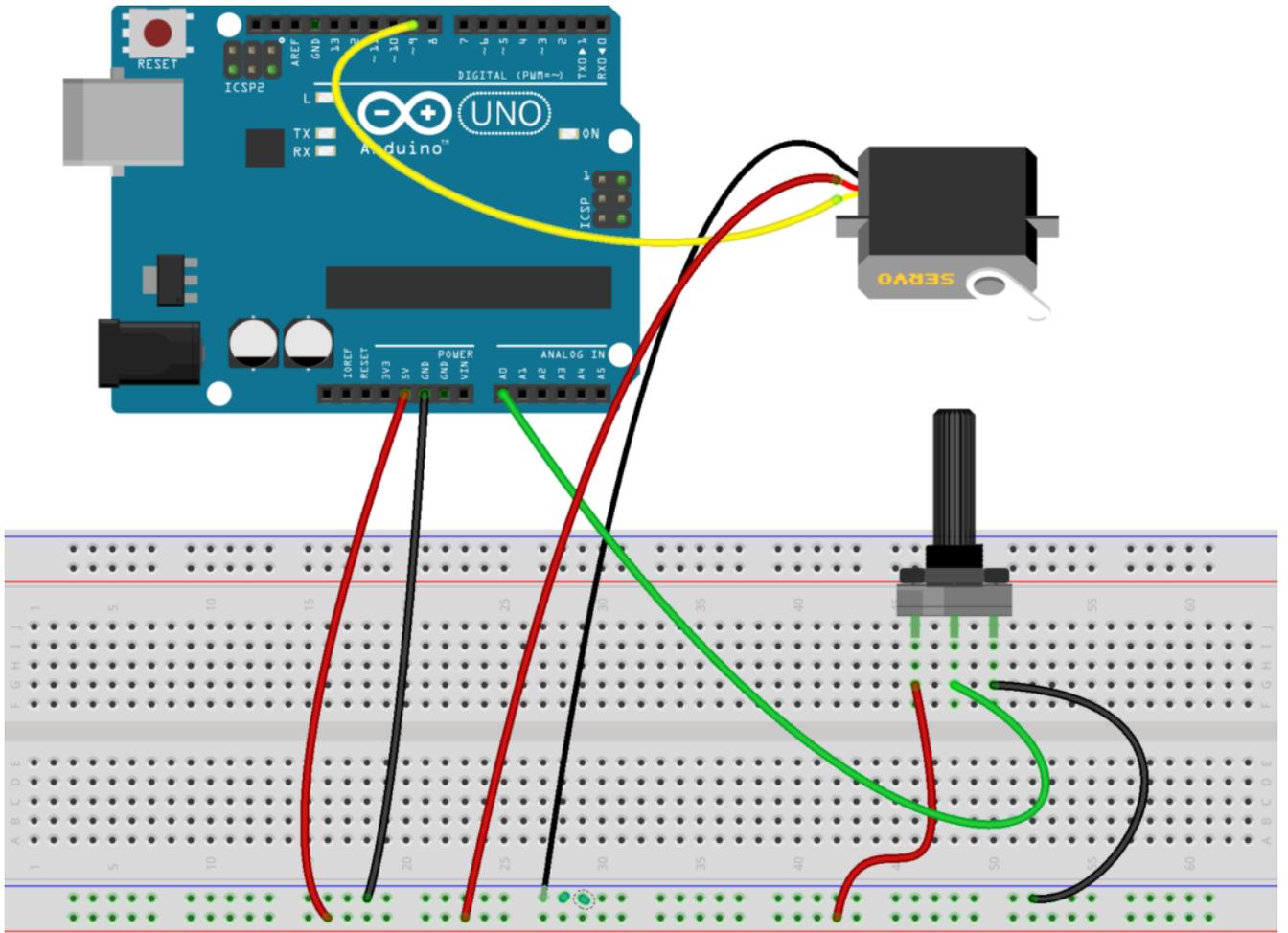
Come già accennato, i servo motori sono dispositivi molto utilizzati in svariati ambiti perché permettono la rotazione del proprio albero in base ad un angolo prestabilito.

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

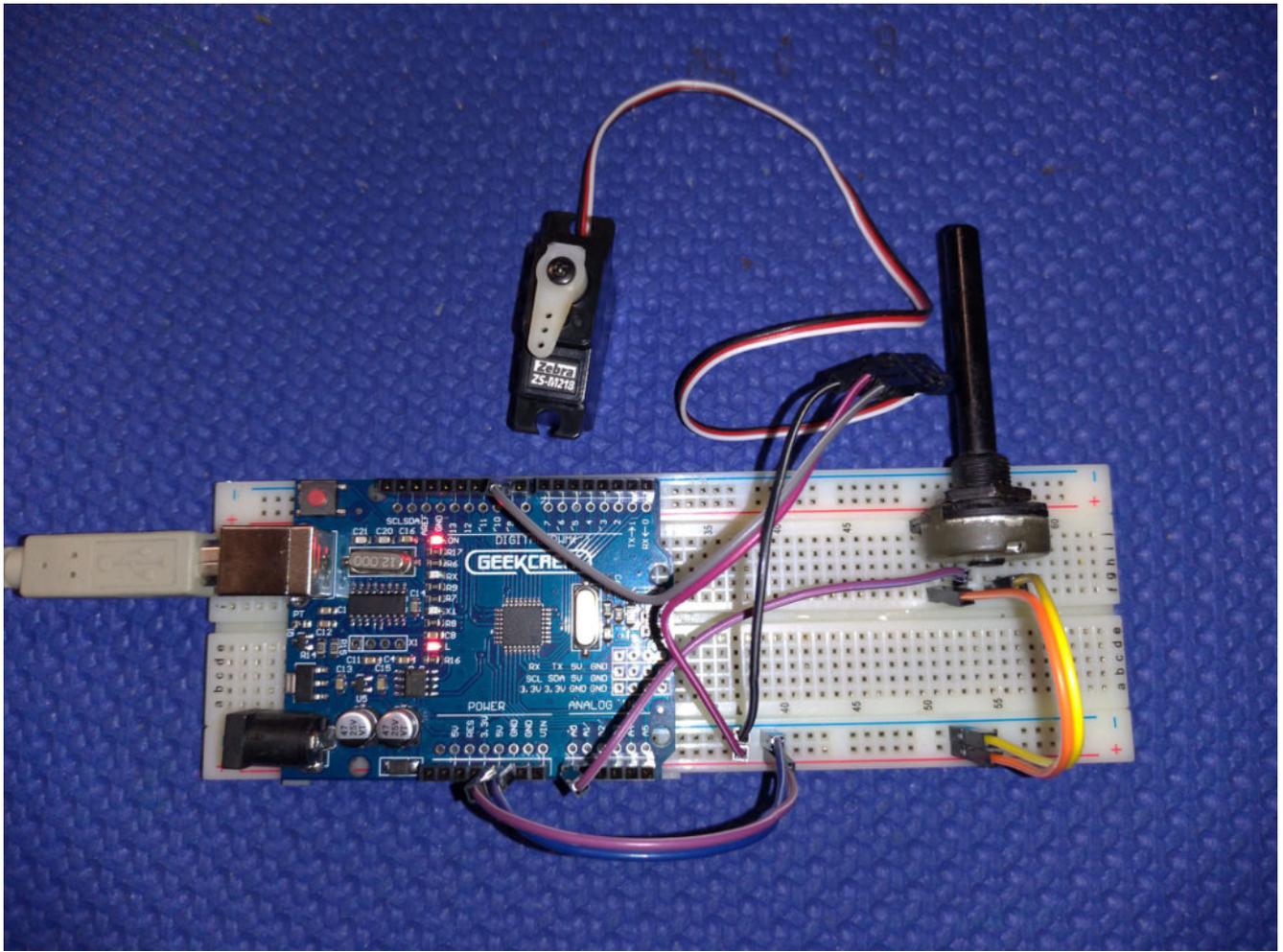
In questo tutorial vedremo come interconnettere il servo motore ad Arduino e come farlo ruotare tramite un potenziometro con pochissime istruzioni.

Collegamento Circuitale:



fritzing

Risultato:



Codice:

Il Servomotore

Obiettivo: Semplice comando di un servomotore

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Servomotore

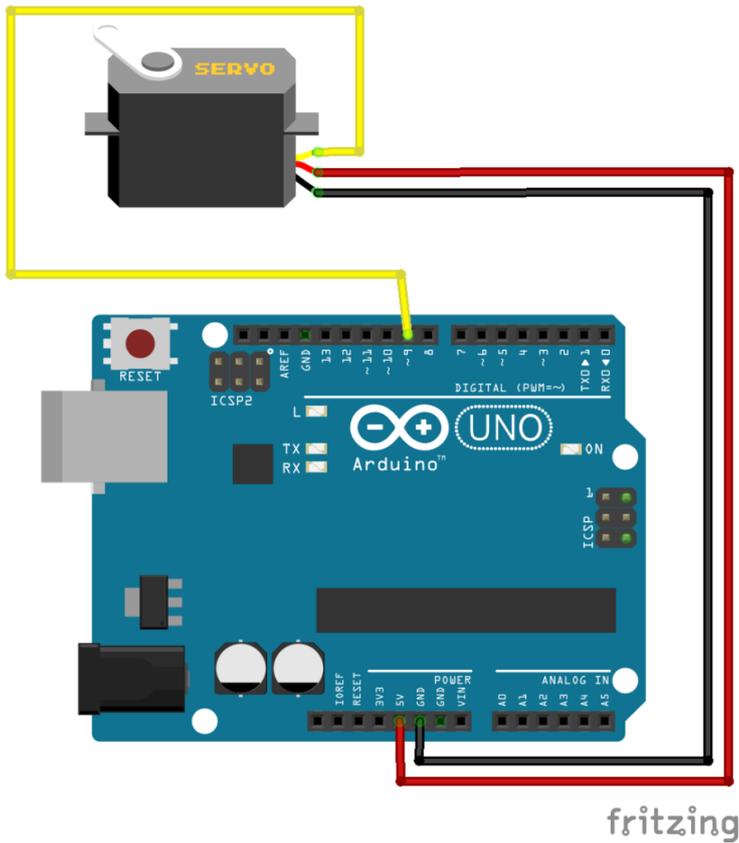
Teoria: Il Servomotore è uno particolare tipo motore ampiamente utilizzato sia in contesti industriali sia nell'ambito del modellismo. Nel dettaglio, il servomotore è impiegato in tutte le applicazioni che prevedono il controllo della posizione di un motore in corrente continua ed il raggiungimento di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

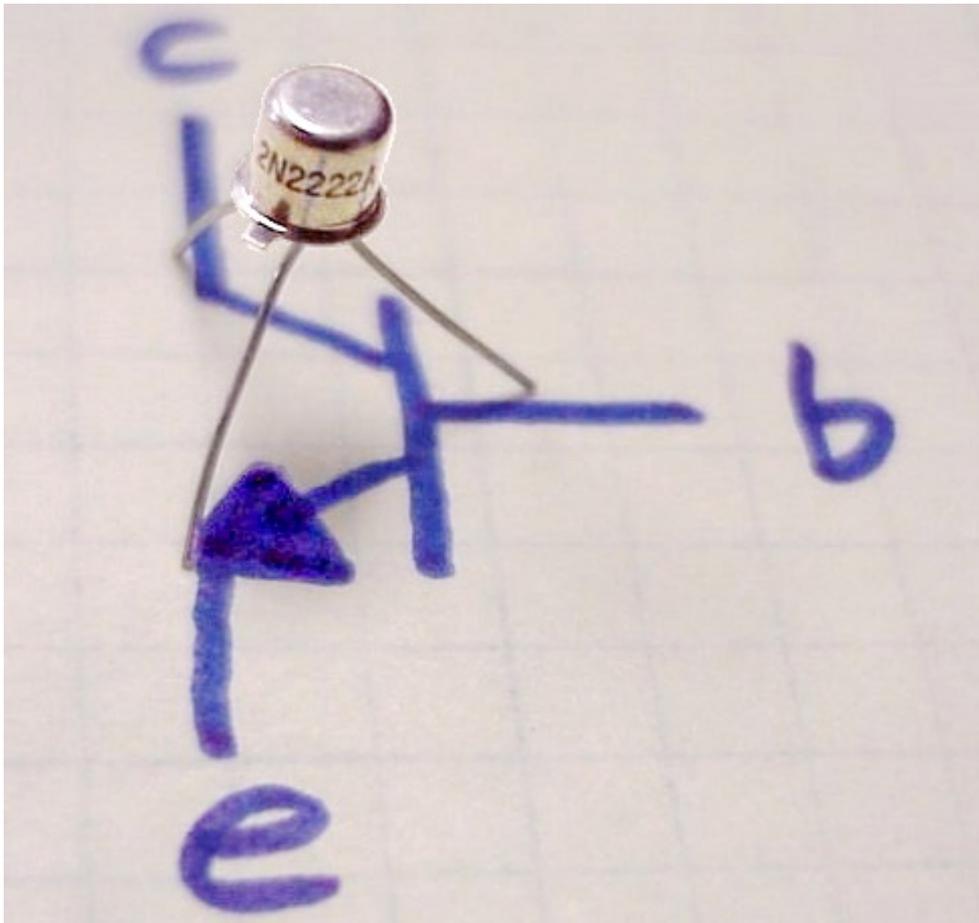
Azionare un Motore a CC con il Transistor

Obiettivo: Azionare un motore a corrente continua tramite transistor

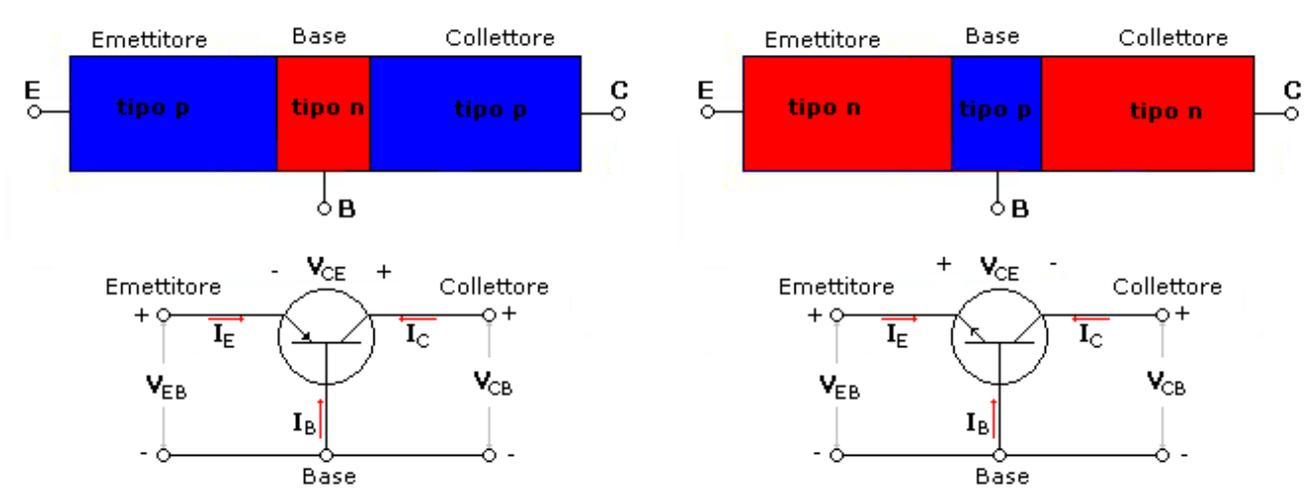
Componenti elettronici:

- 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.



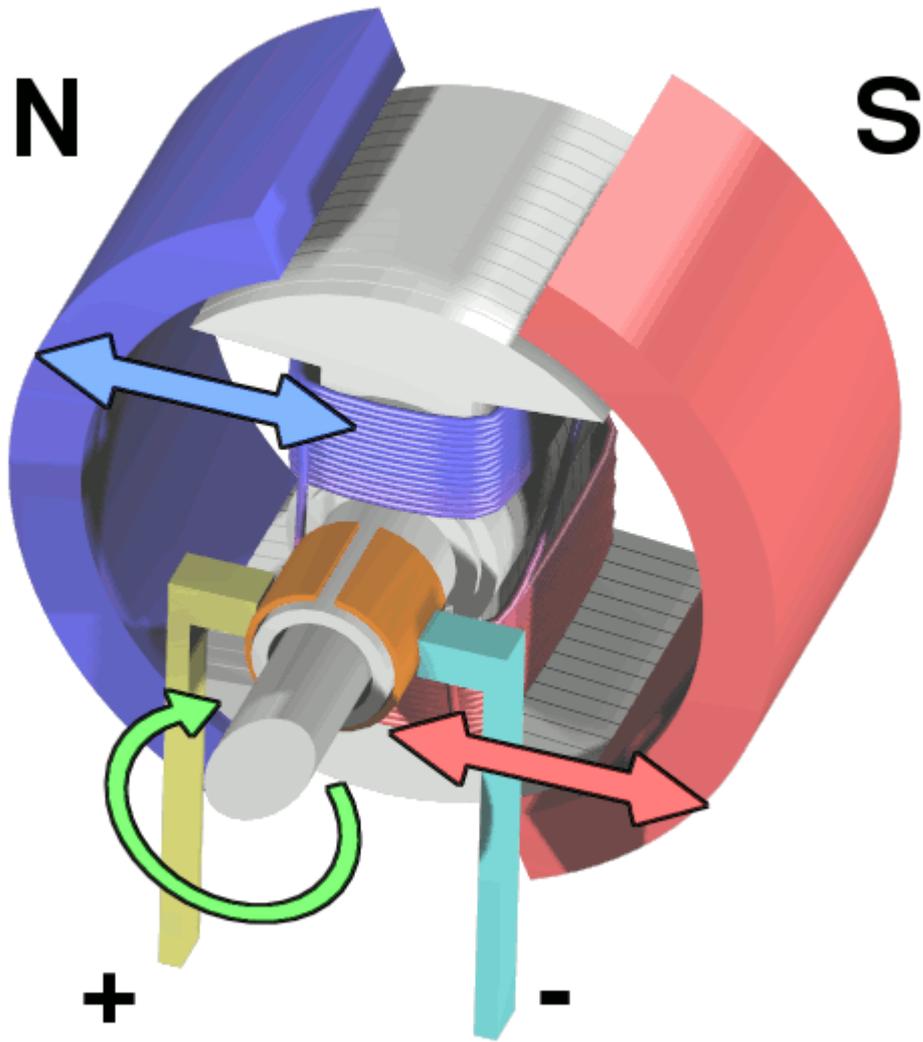
Esistono due tipologie di transistor, a seconda di come sono costruiti: Transistor PNP e NPN; l'unica differenza funzionale tra un transistor PNP e un transistor NPN è la polarità delle giunzioni durante il funzionamento



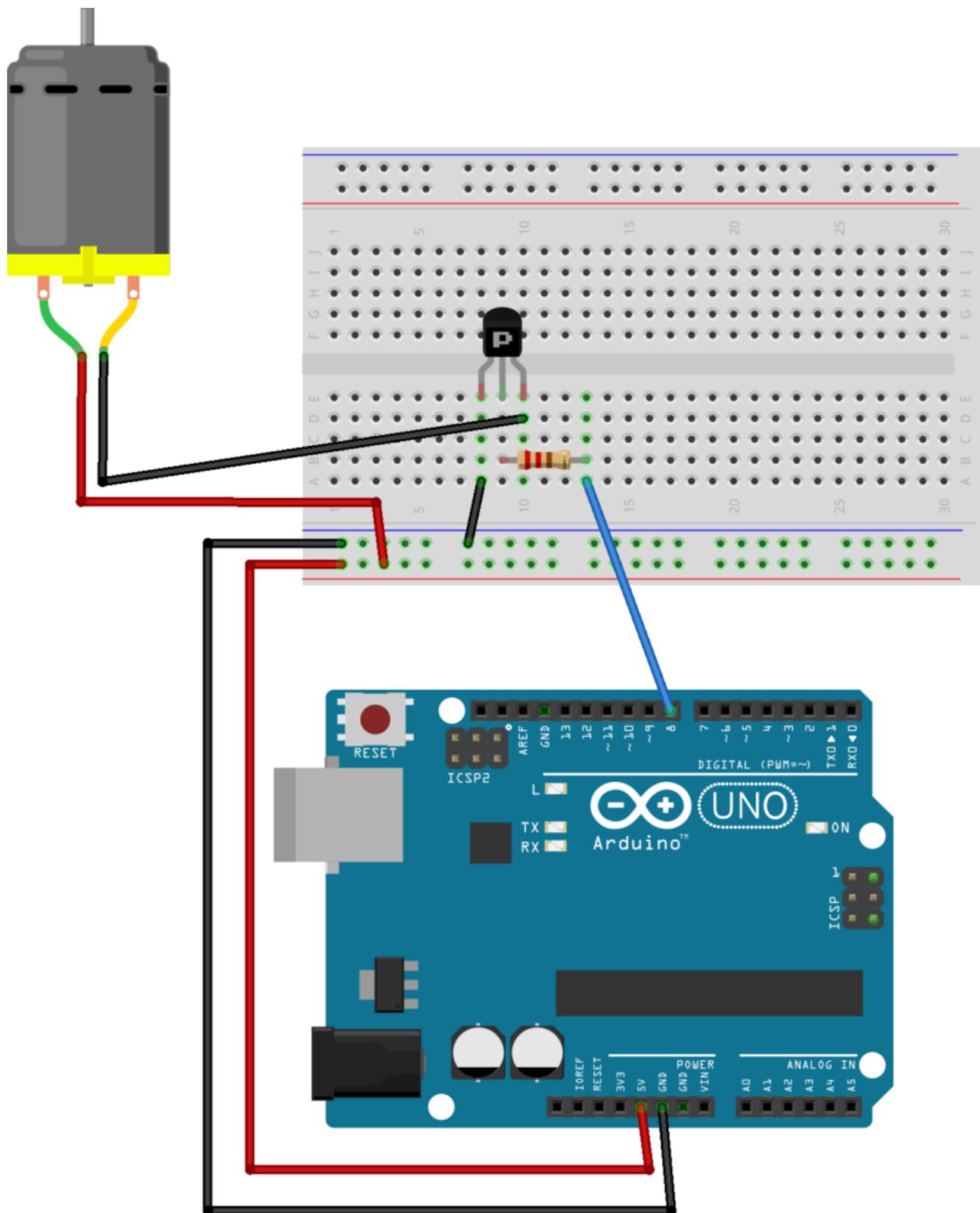
Transistor bipolare a giunzione tipo p-n-p e n-p-n con rappresentazione circuitale.

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.

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

```
// Stampa 1 sul display a 7 segmenti
void uno(){
  digitalWrite(pinA,HIGH);
  digitalWrite(pinB,LOW);
  digitalWrite(pinC,LOW);
  digitalWrite(pinD,HIGH);
  digitalWrite(pinE,HIGH);
  digitalWrite(pinF,HIGH);
  digitalWrite(pinG,HIGH);
}

// Stampa 2 sul display a 7 segmenti
void due(){
  digitalWrite(pinA,LOW);
  digitalWrite(pinB,LOW);
  digitalWrite(pinC,HIGH);
}
```

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.

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

```
// Stampa 2 sul display a 7 segmenti
void due(){
  digitalWrite(pinA,LOW);
  digitalWrite(pinB,LOW);
  digitalWrite(pinC,HIGH);
  digitalWrite(pinD,LOW);
  digitalWrite(pinE,LOW);
  digitalWrite(pinF,HIGH);
  digitalWrite(pinG,LOW);
}

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

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.

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

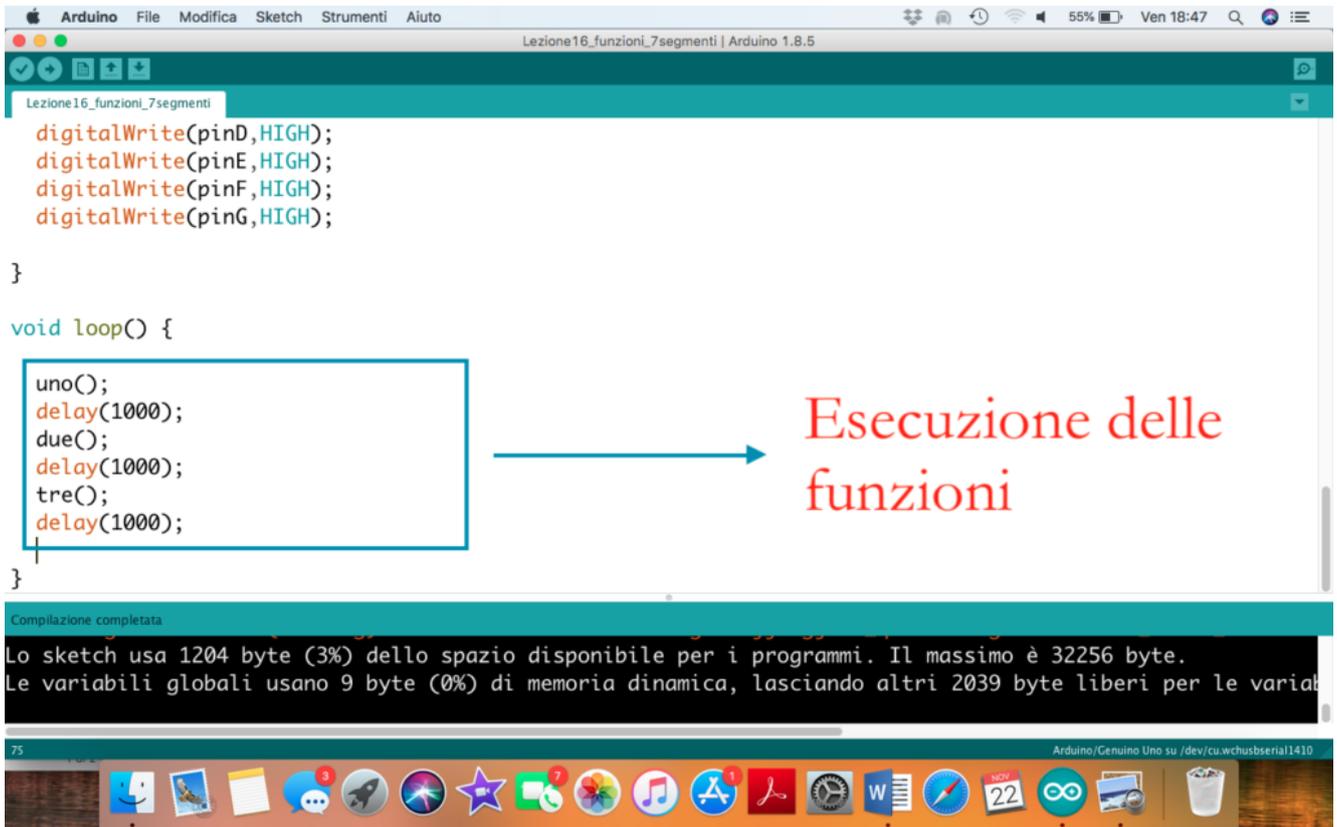
```
// Stampa 3 sul display a 7 segmenti
void tre(){
  digitalWrite(pinA,LOW);
  digitalWrite(pinB,LOW);
  digitalWrite(pinC,LOW);
  digitalWrite(pinD,LOW);
  digitalWrite(pinE,HIGH);
  digitalWrite(pinF,HIGH);
  digitalWrite(pinG,LOW);
}

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

Compilazione completata

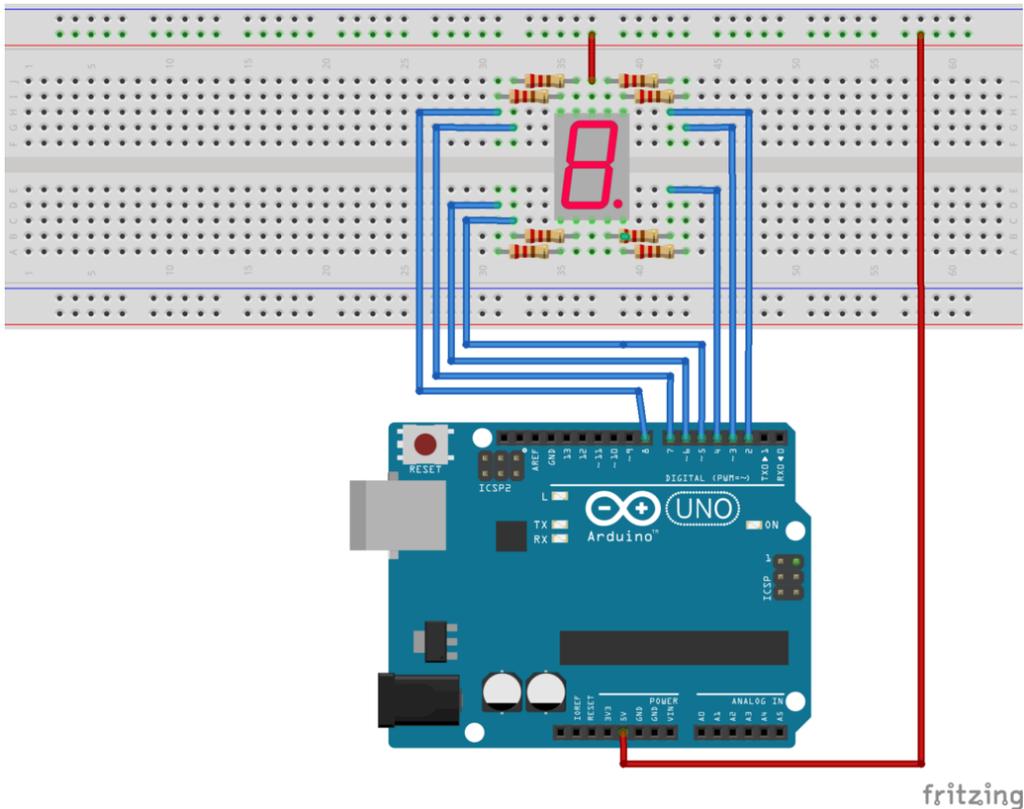
Lo sketch usa 1204 byte (3%) dello spazio disponibile per i programmi. Il massimo è 32256 byte.
Le variabili globali usano 9 byte (0%) di memoria dinamica, lasciando altri 2039 byte liberi per le variabili locali.

- **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 .

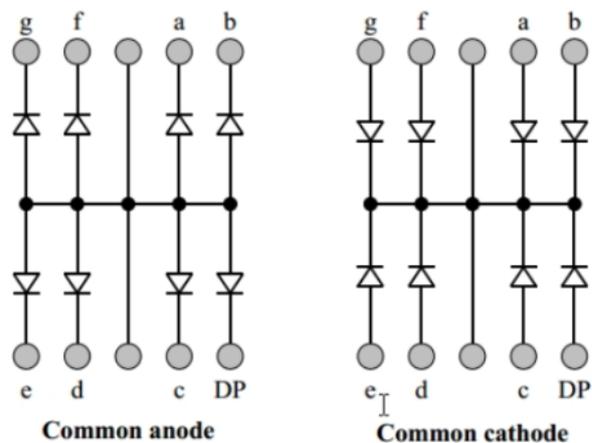
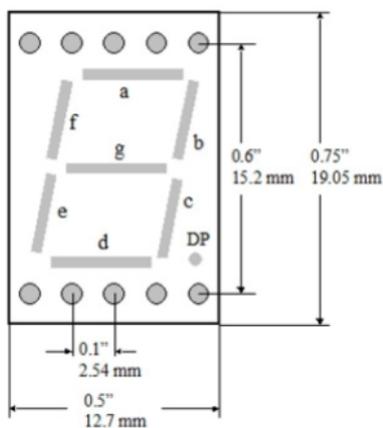
Componenti elettronici:

- 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.

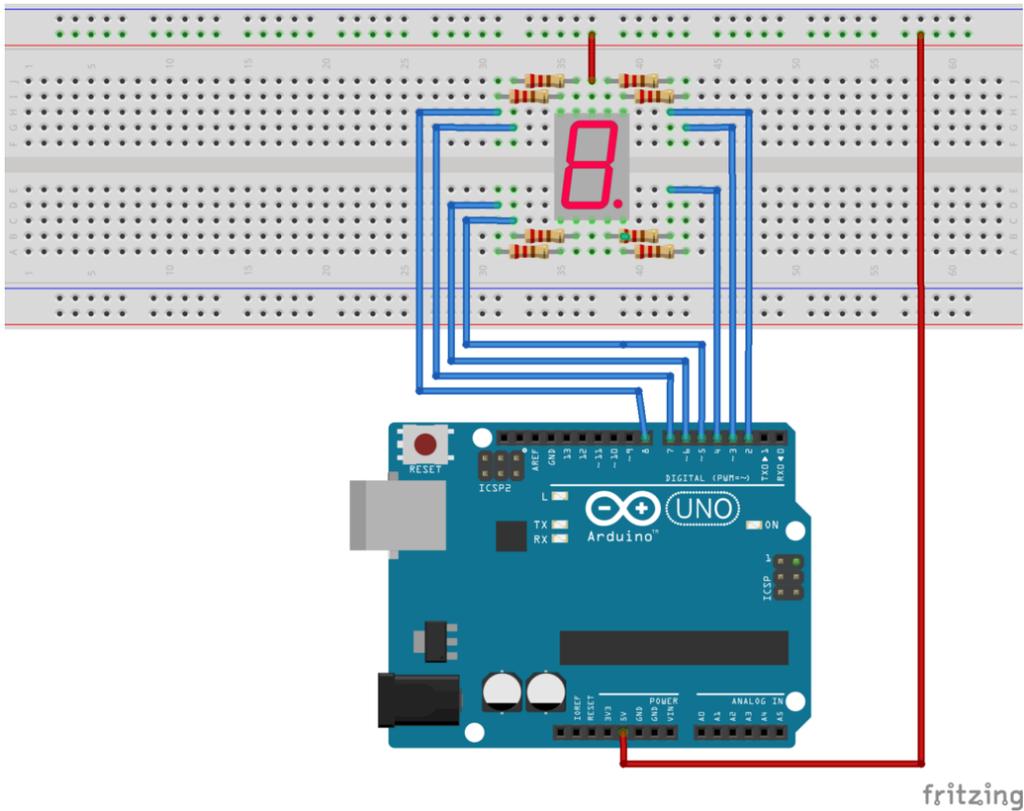
I display a 7 segmenti sono dotati di 10 differenti pin e sono così classificati:

- **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:

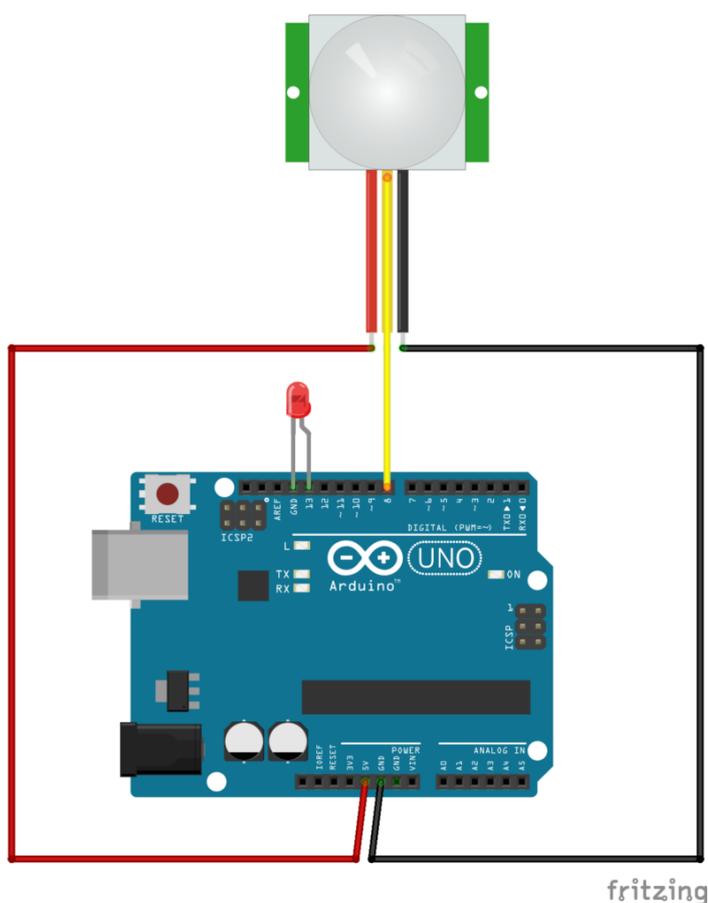
- Modifica la sensibilità legata alla distanza: ruotando in senso orario la distanza aumenta fino ad un massimo di 7 metri, diversamente, ruotando in senso antiorario il potenziometro la distanza diminuisce fino ad un minimo di 3 metri.
- Modifica la sensibilità legata al tempo per il quale il segnale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

Crepuscolare [Avanzato] (Smart Lamp)

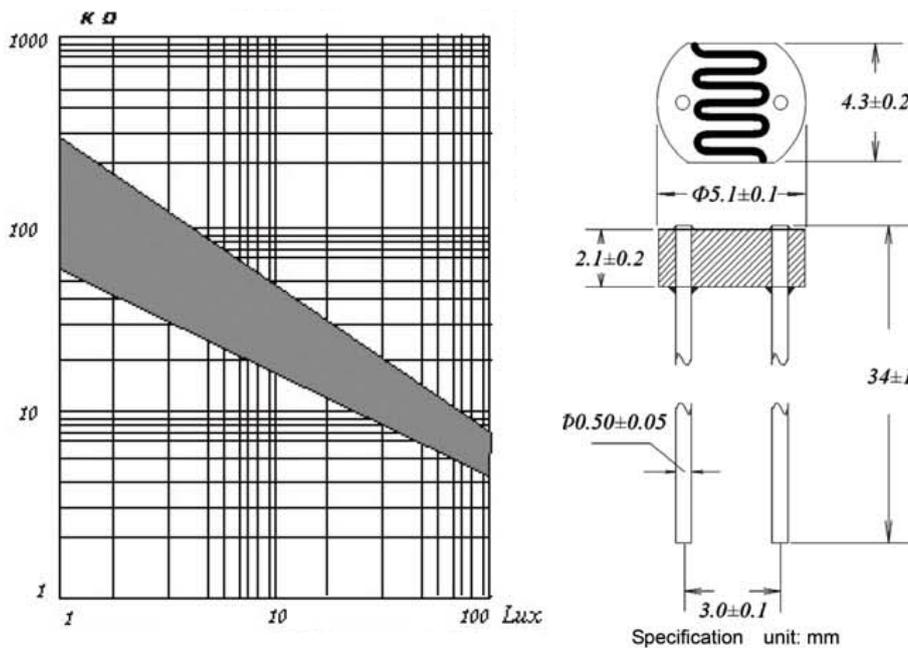
Obiettivo: Accensione automatica di quattro LED al diminuire dell'intensità di luce rilevata.

Componenti elettronici:

- Arduino UNO
- Breadboard
- 4 Led
- 1 Foto-resistenza
- 4 Resistenze (1000hm) per Led
- 1 Resistenza (2.2k0hm) per Foto-resistenza

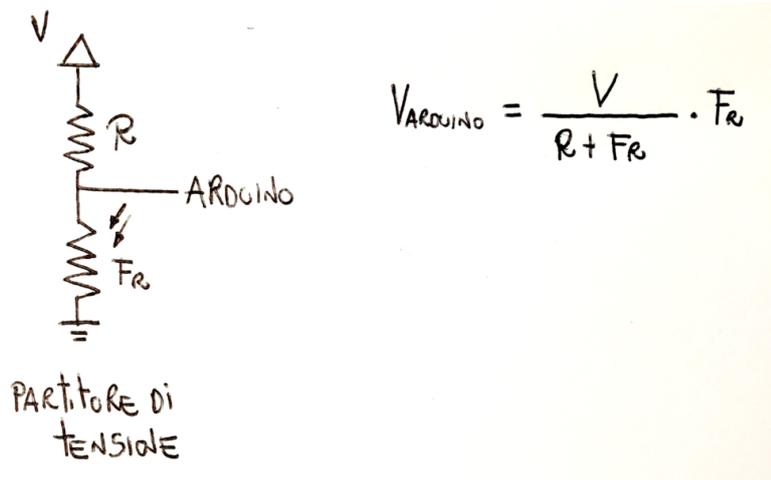
Teoria: La foto resistenza è un componente elettronico la cui resistenza è inversamente proporzionale alla quantità di intensità luminosa che lo colpisce. Questo significa che la

corrente (inversamente proporzionale alla resistenza) aumenta all'aumentare dell'intensità luminosa.
 A seguire, viene riportata la caratteristica Ohm/Lux di una fotoresistenza tipicamente impiegata in applicazioni realizzate mediante Arduino.



Datasheet Fotoresistenza

Nel caso specifico, è importante ricordare che Arduino non può rilevare né variazioni di resistenza né variazioni di corrente. Il microcontrollore può infatti analizzare solamente valori di tensione. Per questo motivo, l'utilizzo di una resistenza è indispensabile al fine di collegare correttamente la fotoresistenza ad Arduino. Nel dettaglio, il circuito previsto per trasformare la variazione di resistenza in una variazione di tensione è il partitore di tensione.



Partitore di Tensione e Fotoresistenza

Nel circuito presentato, la variazione di luminosità produce una variazione del valore della fotoresistenza. Di conseguenza anche il valore della corrente risulta funzione dell'intensità luminosa e di conseguenza anche il valore di tensione in ingresso ad Arduino. Nel dettaglio:

- Un **incremento** della luminosità porta ad un decremento della tensione.
- Un **decremento** della luminosità porta ad un incremento della tensione.
 - se il pulsante viene premuto la tensione in ingresso ad Arduino è pari a 0.
 - se il pulsante non viene premuto la tensione in ingresso ad Arduino è pari a V_{CC} (5V)
- Resistenza di **Pull Down**:
 - se il pulsante viene premuto la tensione in ingresso ad Arduino è pari a V_{CC} (5V).
 - se il pulsante non viene premuto la tensione in ingresso ad Arduino è pari a 0.

Attraverso l'utilizzo del comando **analogRead** è possibile leggere la tensione su uno specifico pin analogico (A0-A5) di Arduino. La funzione analogRead restituisce un valore compreso

tra 0 e 1023 a seconda della tensione letta dal microcontrollore.

A titolo di esempio, se il valore di tensione letto utilizzando la funzione `analogRead` sul pin A0 di Arduino risulta pari a 613. Il valore di tensione può essere facilmente calcolato:

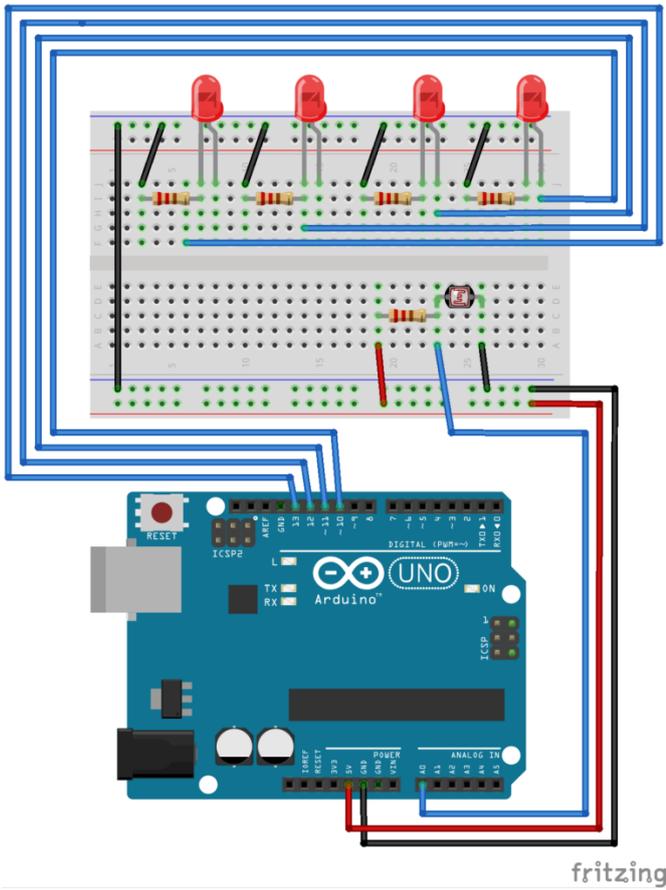
$$\text{valoreTensioneAnalogico} = 613/1023*5 = 3V$$

dove:

- Il valore analogico di tensione letto utilizzando l'istruzione `analogRead` è pari a 613
- Il valore di tensione massimo che può essere letto dalla funzione `analogRead` è pari a 1023
- La tensione massima in uscita ad Arduino è pari a 5V

Tale valore può essere facilmente utilizzato per controllare uno o più led mediante l'**istruzione condizionale IF**.

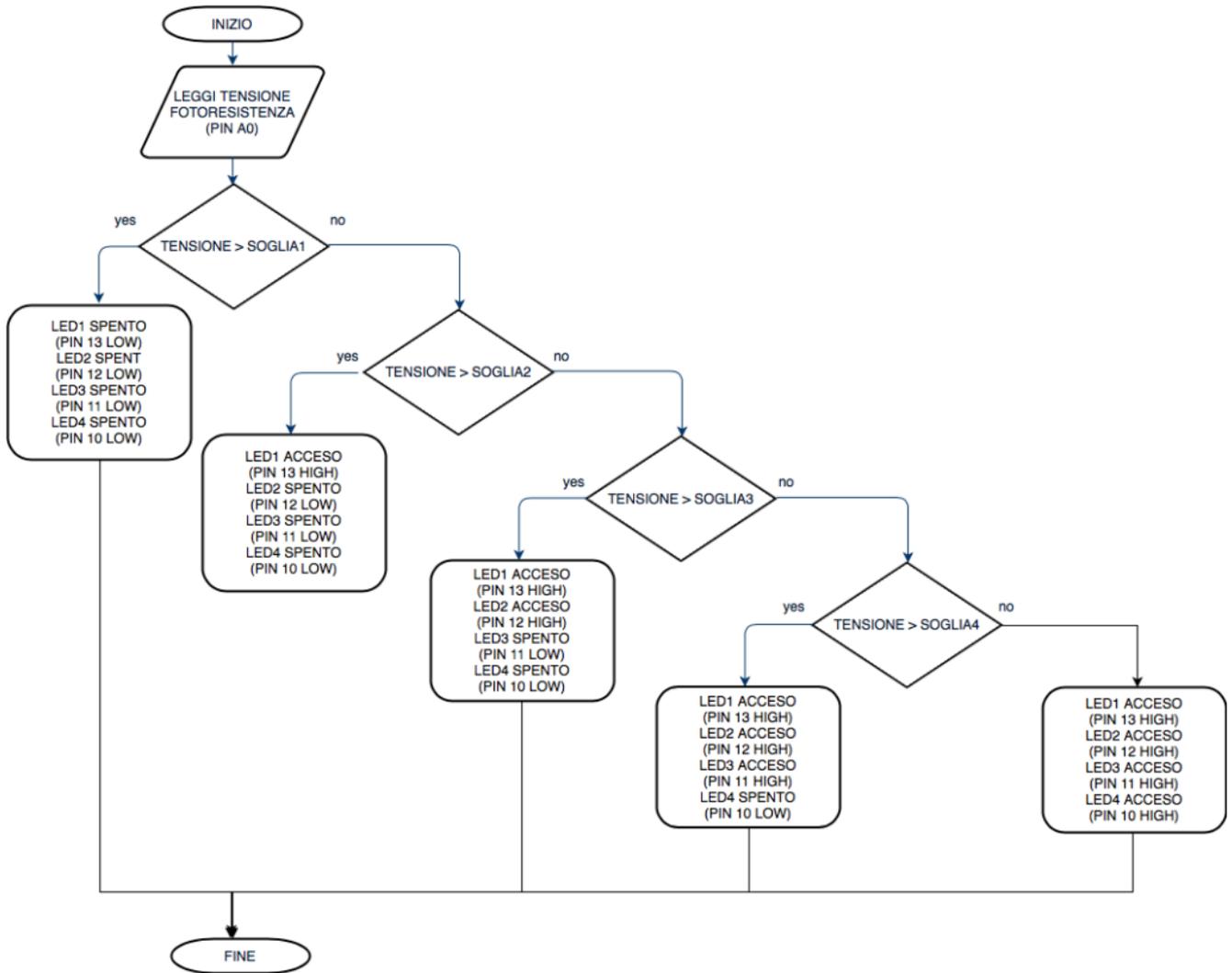
Collegamento Circuitale:



Collegamento Circuitale

Codice:

A seguire viene riportata la schematizzazione mediante flowchart dell'algoritmo utilizzato per realizzare il programma.



Flowchart

Codice:

[crayon-663bd9ae2beb9412048892/]

P