Controllo di un LED mediante un Potenziometro

Obiettivo: Controllare un LED utilizzando un potenziometro

Pre-Requisiti

Fading led

Componenti elettronici:

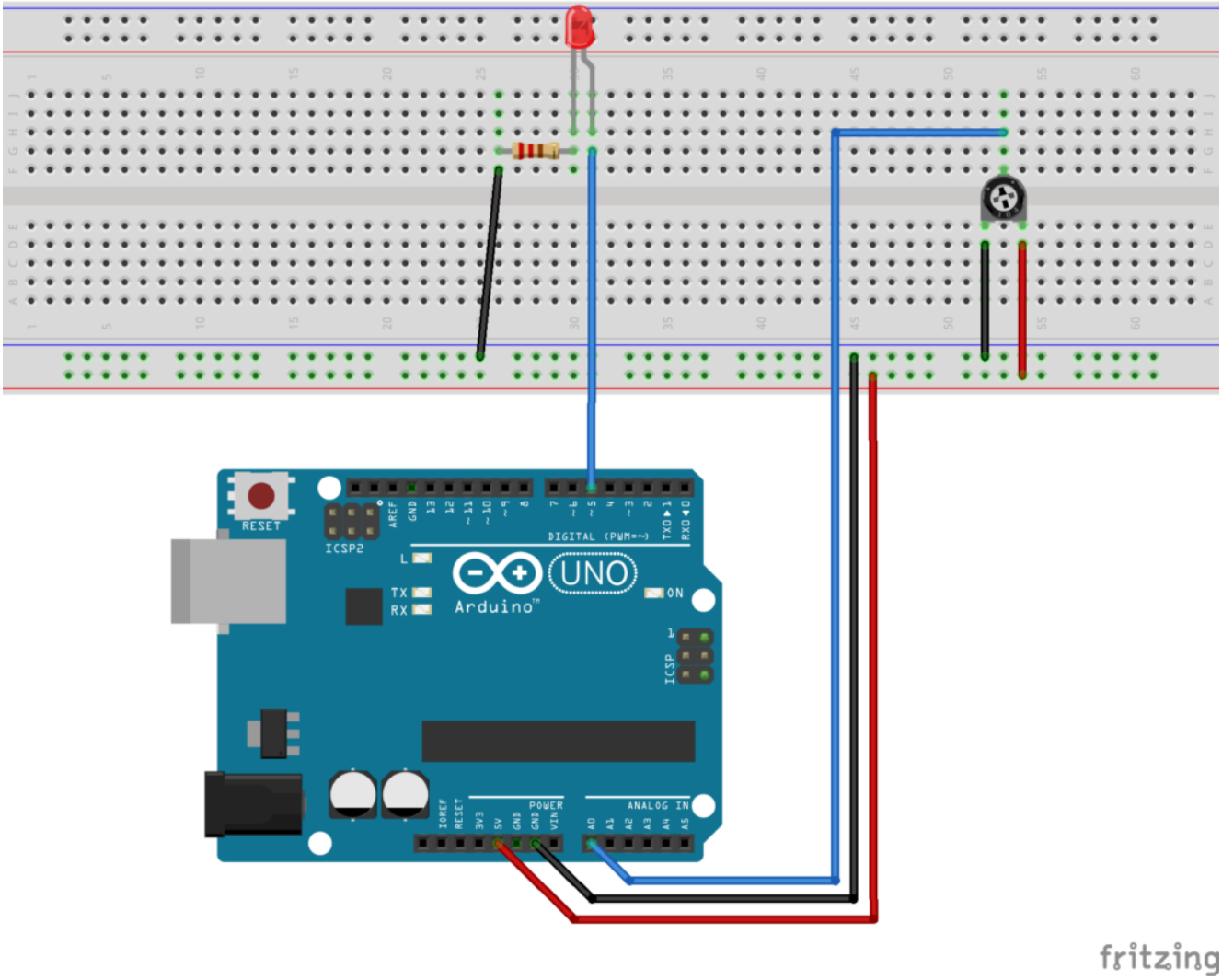
- Arduino UNO
- Breadboard
- 1 Led
- 1 Resistenza (100 Ohm)
- 1 Trimmer (4.7KOhm o similari)

Teoria: In questo articolo si propone l'utilizzo di un potenziometro per regolare in modo manuale la luminosità di un LED. E' importante considerare che (come riportato nei pre-requisiti) l'impiego della funzione **digitalWrite** non permette di modulare l'intensità luminosa del LED. Difatti, attraverso l'utilizzo di questa istruzione digitale, il LED può trovarsi solamente in due stati logici LOW (spento) o HIGH (acceso).

Per raggiungere lo scopo prefissato è pertanto necessario l'utilizzo di una differente funzione denominata: **analogWrite**. Questa funzione permette infatti di modulare l'intensità luminosa del LED fornendo 256 differenti livelli di luminosità. L'istruzione analogWrite permette infatti di emulare un finto segnale analogico attraverso l'impiego della tecnica **PWM** (Pulse Width Modulation). Solamente sei PIN (quelli contrassegnati dal simbolo tilde ~) possono essere utilizzati per fornire un segnale "analogico".

Tuttavia è importante considerare che se l'istruzione analogwrite permette di gestire la luminosità del LED, questa funzione non permette di controllare la posizione del potenziometro essendo il potenziometro un dispositivo di input (dato da leggere). Pertanto per la gestione del potenziometro sarà effettuata utilizzando una differente funzione denominata **analogRead**. Questa funzione permette infatti di leggere un livello di tensione compreso tra 0 e 5 Volt e mapparlo in un intervallo discreto composto da 1024 livelli (0-1023).

Collegamento Circuitale:



Collegamento Circuitale

Codice:

Pulsante come Interruttore

Obiettivo: Utilizzo di un pulsante come interruttore per l’accensione di un led (stato ON/OFF)

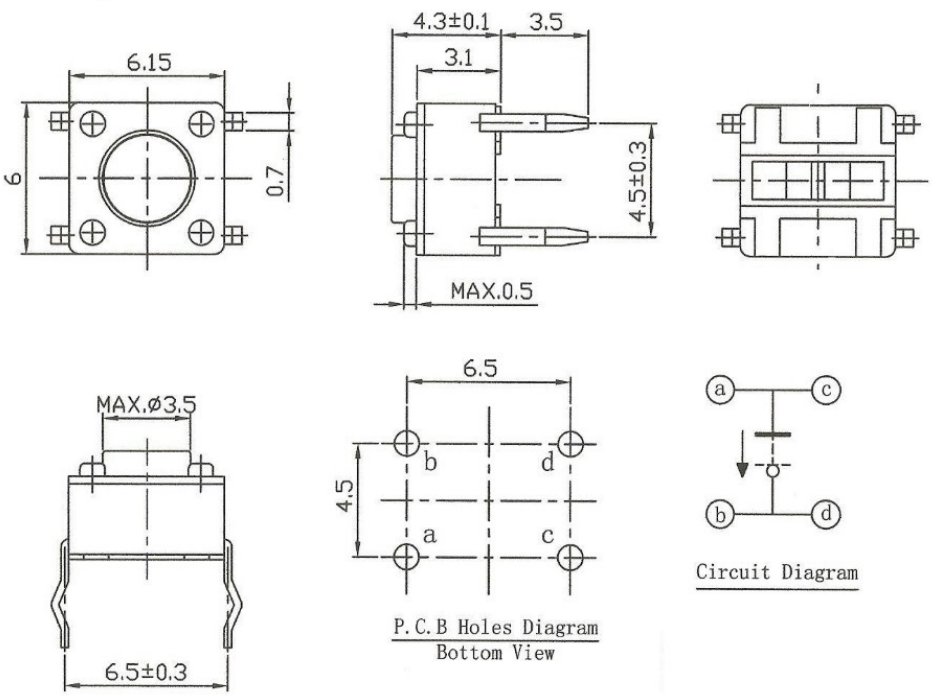
Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Led
- 1 Pulsante
- 1 Resistenza (100 Ohm)
- 1 Resistenza (1kOhm)

Pre-Requisiti

LED e Pulsante

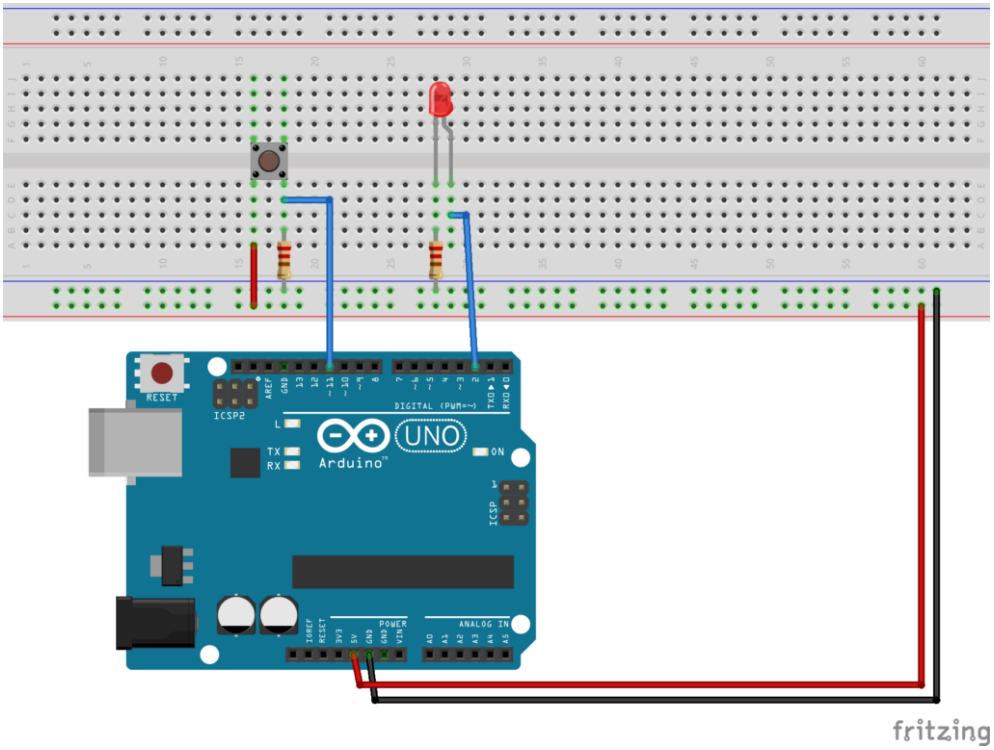
Teoria: Da un punto di vista pratico mantenere una luce accesa tenendo premuto il pulsante (vedi lezione pre-requisiti) non è proprio il massimo della vita. Se da un punto di vista elettronico il componente tipicamente utilizzato per gestire lo stato ON/OFF di una lampada è l'interruttore, attraverso una corretta gestione del software di controllo è possibile "trasformare un pulsante in interruttore". Per questo motivo, agendo direttamente sul software e lasciando invariato l'hardware proposto nell'esperienza LED e Pulsante, è possibile trasformare il sistema in questione in un controllo ON/OFF dove il pulsante è utilizzato come interruttore.



Datasheet Pulsante

Anche se in questa esperienza il pulsante viene utilizzato come interruttore è importante ribadire che l’impiego di resistenze di Pull-up o Pull-down sono necessarie per un corretto funzionamento del circuito.Maggiori informazioni sulle resistenze di Pull-up o Pull-down possono essere trovate qui.

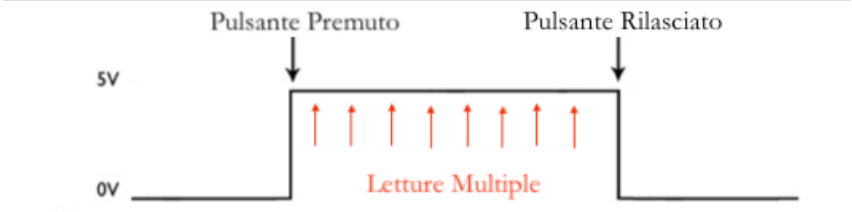
Collegamento Circuitale:



Collegamento Circuitale

Codice: Il codice utilizzato per trasformare il pulsante in interruttore si basa sull'impiego di due differenti variabili globali (create fuori dalle funzioni loop e setup, pertanto visibili in tutto il codice). Queste variabili sono:

- **valButtonOld**: Memorizza la lettura precedente effettuata sul pulsante. Attraverso questa variabile è possibile evitare letture multiple. E' importante considerare che essendo il clock di Arduino più elevato del tempo di reazione umano, premendo il pulsante anche per un istante brevissimo il controllore eseguirà più letture con stato del pulsante HIGH eseguendo in seguito il codice corrispondente. Per questo motivo si preferisce leggere lo stato di transizione da basso ad alto piuttosto che lo stato alto di un pulsante.



Esempio di lettura multipla

- **ledState**: Memorizza lo stato del led (se acceso o spento). Premuto il pulsante, nel caso in cui il led sia acceso questo viene spento viceversa nel caso in cui il led sia spento questo viene acceso. Per questo motivo è indispensabile utilizzare una variabile per controllare lo stato del pulsante.

Personalizzazioni:

L'impiego della variabile valButtonOld, utilizzata per evitare letture multiple, può essere evitato aggiungendo un delay dopo la lettura del pulsante. Si provi ad implementare il circuito ed il relativo codice.

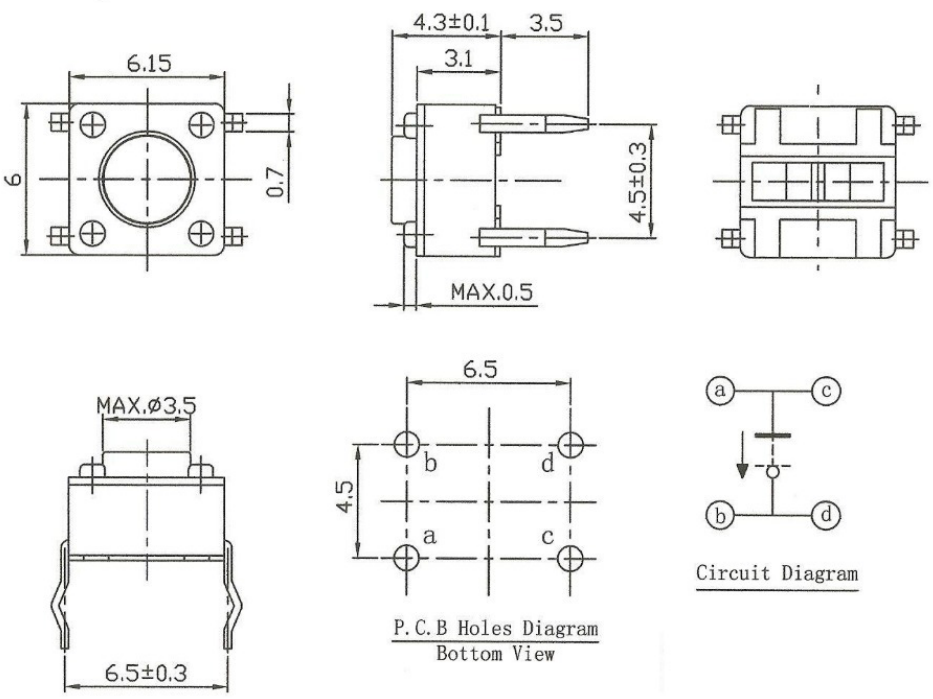
LED e Pulsante

Obiettivo: Accensione di un LED mediante un pulsante.

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Led
- 1 Pulsante
- 1 Resistenza (100 Ohm)
- 1 Resistenza (1kOhm)

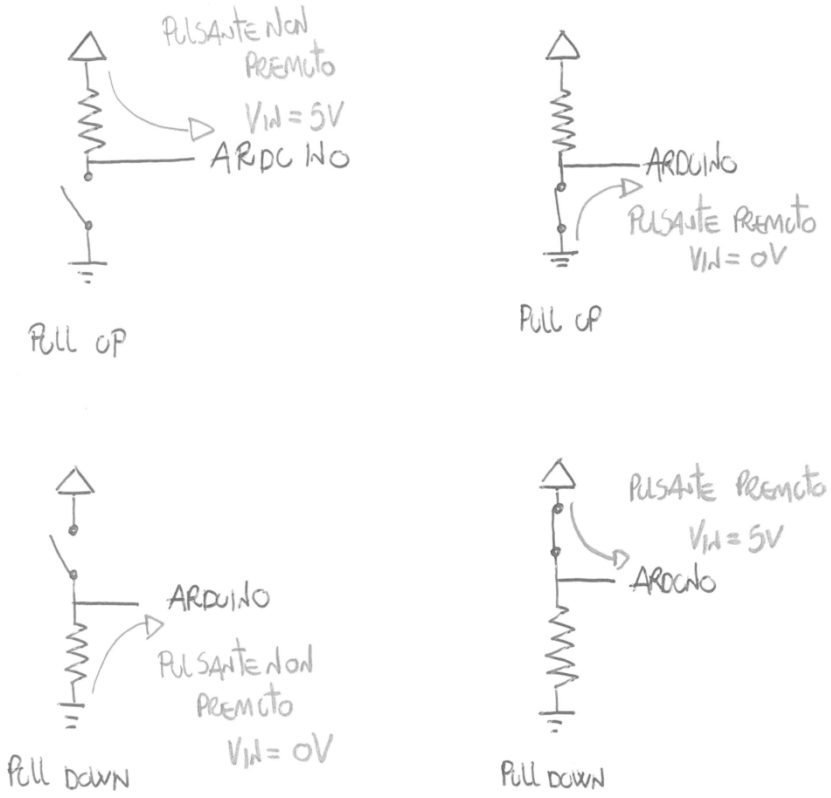
Teoria: Il pulsante è un dispositivo elettronico a due stati (ON, OFF) con una sola posizione monostabile. Nel caso specifico i pulsanti permettono di aprire o chiudere un circuito e pertanto collegare a GND (0V) o a VCC (5V) una specifica uscita. A seguire, viene riportato lo schema circuitale di un pulsante tipicamente impiegato in applicazioni realizzate mediante Arduino.



Datasheet Pulsante

L’utilizzo di una resistenza è indispensabile al fine di collegare correttamente il pulsante ad Arduino evitando cortocircuiti. A seconda del collegamento realizzato, la resistenza prende il nome di:

- Resistenza di **Pull Up**: la resistenza viene collegata direttamente all’alimentazione.
- Resistenza di **Pull Down**: la resistenza viene collegata a massa.



Resistenze di Pull Up e di Pull Down

Il comportamento del circuito e la tensione letta dal microcontrollore dipende dalla tipologia di collegamento circuitale utilizzato. In particolare:

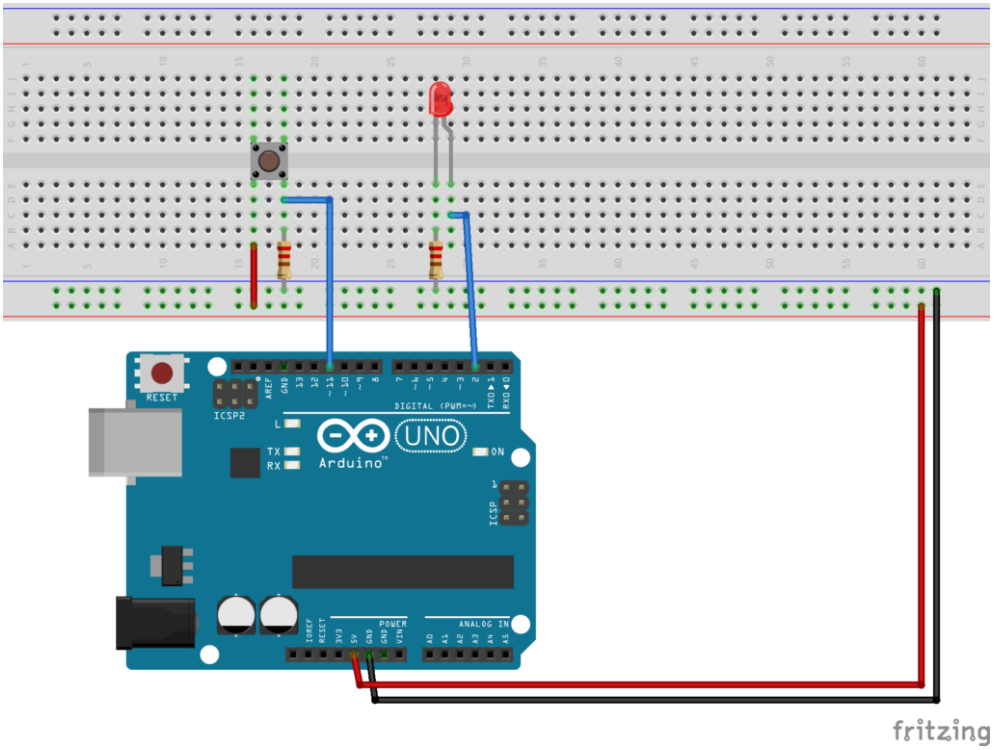
- Resistenza di **Pull Up**:

        - 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 Vcc (5V)
    - Resistenza di **Pull Down**:
        - se il pulsante viene premuto la tensione in ingresso ad Arduino è pari a Vcc (5V).
        - se il pulsante non viene premuto la tensione in ingresso ad Arduino è pari a 0.

Attraverso l'utilizzo del comando **digitalRead** è possibile leggere la tensione su uno specifico pin digitale (0-13) di Arduino. La funzione digitalRead restituisce un valore (i.e., LOW o HIGH) a seconda della tensione letta dal microcontrollore. Tale valore può essere facilmente utilizzato per controllare un led mediante l'**istruzione condizionale** IF.

A seguire viene riportato lo schema elettrico ed il codice utilizzato per l'accensione di un pulsante sfruttando una resistenza di Pull Down.

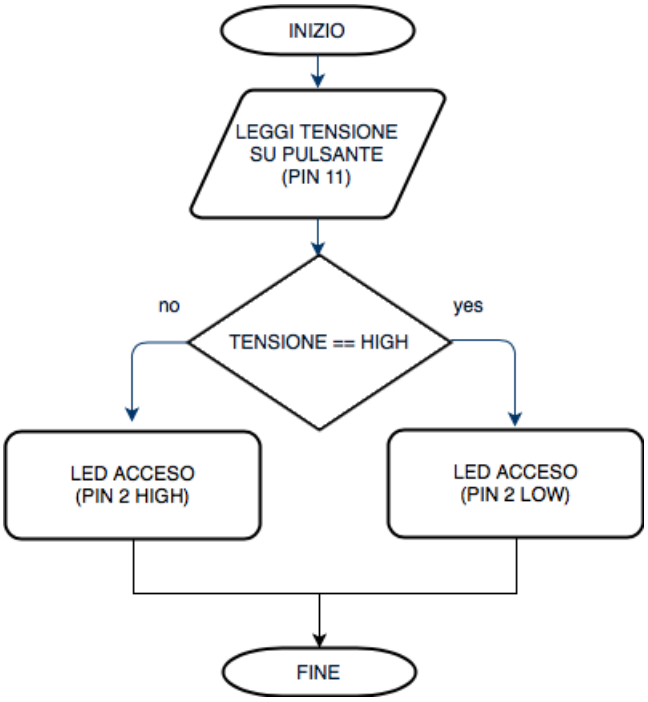
Collegamento Circuitale:



Collegamento Circuitale

Codice:

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



Flowchart

Personalizzazioni:

E’ possibile modificare il circuito aggiungendo pulsanti e led. E’ inoltre possibile modificare il codice al fine di realizzare una lampada che rimanga acceso fino a quando il pulsante non venga premuto una seconda volta (Attenzione possibili problematiche di rimbalzo).

Il LED RGB

Obiettivo: Accensione di un LED RGB (Red Green Blue) .

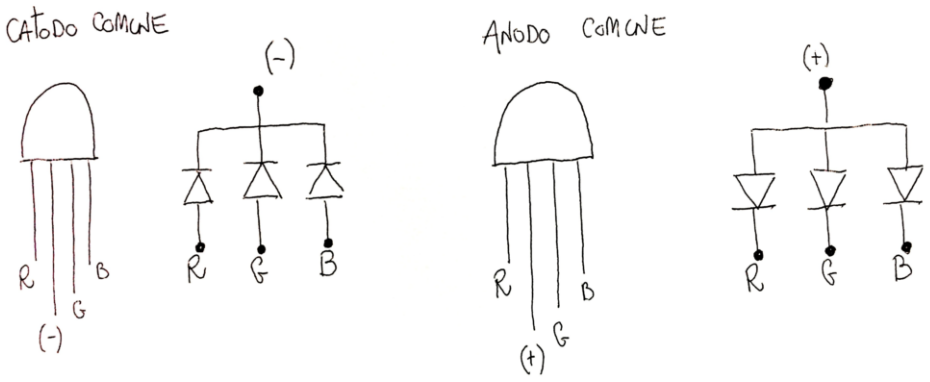
Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Led RGB
- 3 Resistenza (100 Ohm)

Teoria: Gli RGB rappresentano una particolare sottocategoria di LED capaci di riprodurre differenti colori attraverso la combinazione dei colori fondamentali Rosso Verde e Blu. Nel dettaglio, questi dispositivi sono costituiti da tre differenti led azionati in funzione del colore da riprodurre.

I LED RGB sono dotati di 4 differenti morsetti e sono così classificati:

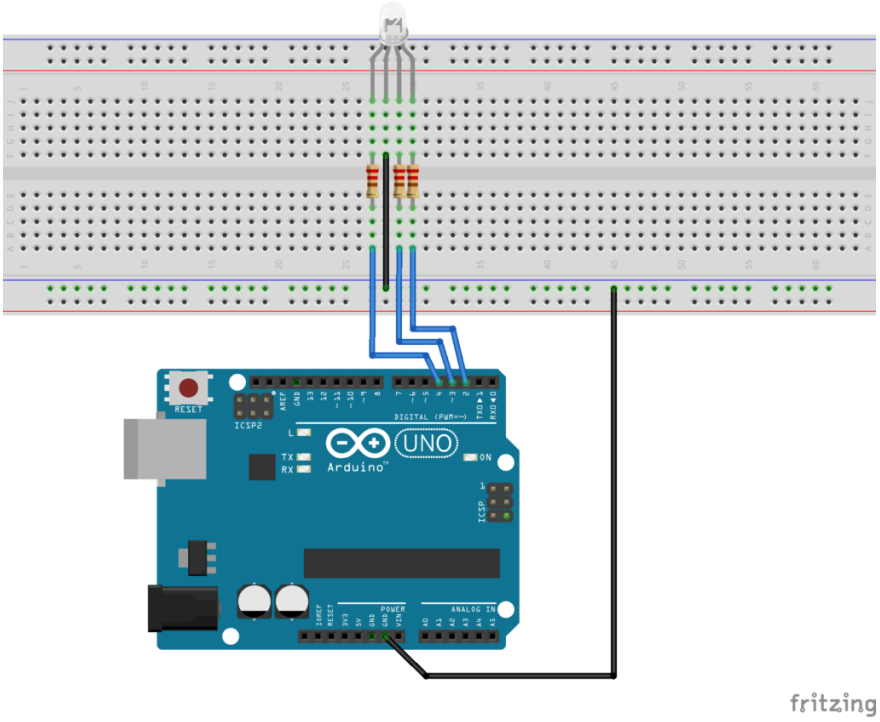
- **RGB ad Anodo Comune:** il pin più lungo del LED deve essere collegato a massa (GND) mentre gli altri 3 differenti PIN sono utilizzati per comandare i 3 LED (rosso, verde, blue). Nel caso in questione i singoli LED sono accesi attraverso l'istruzione Arduino digitalWrite(pinLed, HIGH);
- **RGB a Catodo Comune:** il pin più lungo del LED deve essere collegato a VCC (5V) mentre gli altri 3 differenti PIN sono utilizzati per comandare i 3 LED (rosso, verde, blue). Nel caso in questione i singoli LED sono accesi attraverso l'istruzione Arduino digitalWrite(pinLed, LOW);



LED RGB (Anodo e Catodo comune)

Collegamento Circuitale:

Viene in seguito riportato lo schema elettrico utilizzato per comandare il LED RGB. Nel dettaglio il LED impiegato è della modalità catodo comune.



Collegamento Circuitale

Codice:

Considerando l’impiego di un LED a catodo comune i singoli LED (R, G, B) possono essere accesi mediante l’istruzione digitalWrite(pinLed,HIGH);

Personalizzazioni:

E’ possibile modificare il circuito aggiungendo pulsanti

Fading led

Obiettivo: Realizzazione di un LED con dissolvenza (fading).

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Led
- 1 Resistenza (100 Ohm)

Teoria: L’utilizzo dell’istruzione **digitalWrite** non permette di modulare/regolare la luminosità di un LED. Attraverso l’utilizzo di questa istruzione digitale infatti il LED può

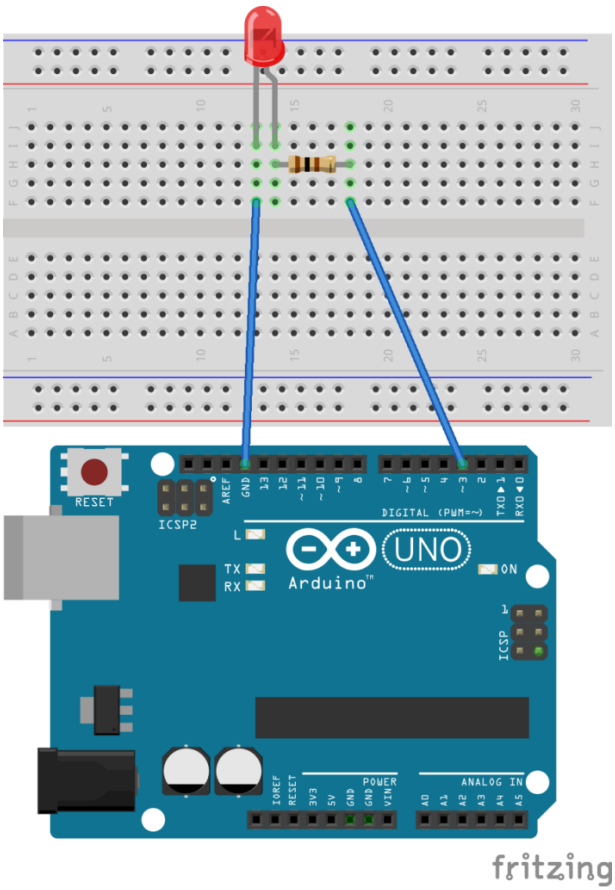
trovarsi solamente in due stati logici LOW (spento) o HIGH (acceso).
Per raggiungere lo scopo prefissato è pertanto necessario l'utilizzo di una differente funzione denominata: **analogWrite**. Questa funzione permette infatti di modulare l'intensità luminosa del LED fornendo 256 differenti livelli di luminosità.
L'istruzione analogWrite permette infatti di emulare un finto segnale analogico attraverso l'impiego della tecnica **PWM** (Pulse Width Modulation). Solamente sei PIN (quelli contrassegnati dal simbolo tilde ~) possono essere utilizzati per fornire un segnale "analogico".
A titolo di esempio, volendo alimentare un dispositivo con una tensione analogica pari a 3V il valore da utilizzare come parametro della funzione analogWrite può essere così calcolato:

valore = 3/5*255 = 153

dove:

- Il valore analogico che si vuole produrre è pari a 3V
- La tensione massima in uscita ad Arduino è pari a 5V
- Il valore massimo utilizzabile dalla funzione analogWrite è pari a 255

Collegamento Circuitale:



Collegamento Circuitale

Codice:

[crayon-6633aae302cc1935940071/]

Blinking led [Avanzato]

Obiettivo: Realizzazione, mediante breadboard, di un led che lampeggi ad una frequenza specifica (e.g., 1Hz) su un PIN differente dal 13.

Componenti elettronici:

- Arduino UNO
- Breadboard
- Led
- Resistenza (100 Ohm)

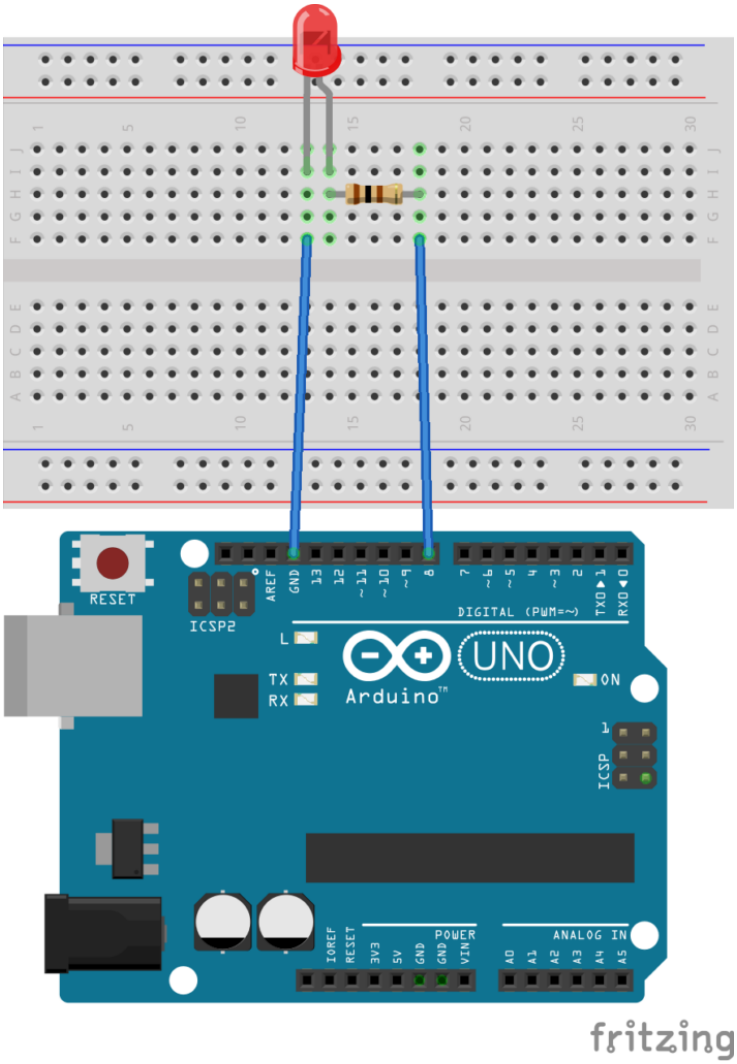
Teoria: Al fine di garantire il corretto funzionamento del LED su un pin differente dal 13, è indispensabile l'utilizzo di una resistenza in serie al dispositivo. La resistenza permette di regolare fissare i corretti valori di tensione e corrente necessari ad alimentare il LED.
Ad esempio, considerando una tensione sul pin di Arduino pari a 5V ed i seguenti parametri caratteristici del LED:

- ILed = 20 mA
- VLed = 1,5 V

Data la **legge di Ohm** V=RI, la resistenza necessaria per garantire un corretto funzionamento del diodo emettitore di luce può essere così calcolata:

$$R = (5-2)/20*10^{(-3)}$$

Collegamento Circuitale:



Collegamento Circuitale

Codice:

[crayon-6633aae3034fe172600879/]

Tinkercad:

https://www.tinkercad.com/embed/ckyY3Pamusd

Personalizzazioni: E’ possibile modificare il comportamento del circuito in questione intervenendo sul valore della variabile *ledTime*. Modificando il suo valore infatti cambia la frequenza di lampeggiamento del LED.
E’ inoltre possibile modificare il pin digitale utilizzato per pilotare il LED cambiando rispettivamente hardware e software.

Approfondimento Teorico:

Legge di Ohm: La legge di Ohm mette in relazione le tre grandezze elettriche fondamentali Tensione (Volt), Intensità di Corrente (Ampere) e Resistenza (Ohm) secondo la seguente relazione.

$$V = RI$$

L’utilizzo di una resistenza, in serie al LED, serve appunto per limitare la quantità di corrente presente sul diodo emettitore di luce.

Blinking led

Obiettivo: Realizzazione di un led che lampeggi ad una frequenza specifica (e.g., 1Hz)

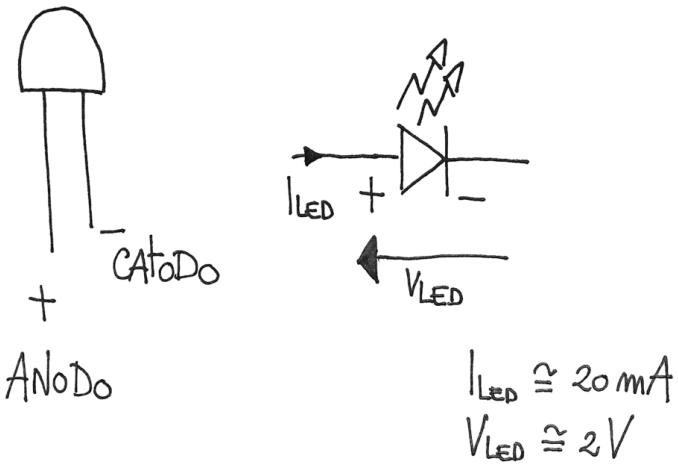
Componenti elettronici:

- Arduino UNO
- Led

Teoria: Il LED (Light Emitting Diode) o Diodo Emettitore di Luce è un dispositivo elettronico che sfrutta le proprietà di alcuni materiali semiconduttori di emettere fotoni (produrre luce).
Questo dispositivo è ampiamente utilizzato in molti applicativi realizzati con Arduino ed è caratterizzato da una propria tensione e corrente di funzionamento. Valori tipici sono:

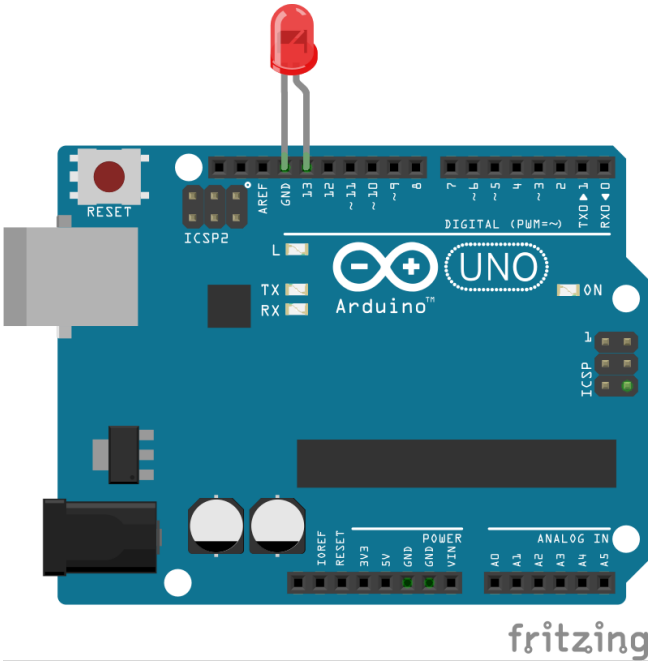
- ILed = 20 mA
- VLed = 2 V

Valori superiori o inferiori possono danneggiare il dispositivo. Per questo motivo si utilizza solitamente una resistenza in serie al LED al fine di limitarne corrente e tensione.
Per un adeguato funzionamento il LED deve anche essere correttamente polarizzato. Nel dettaglio il terminale più lungo di un led rappresenta l'anodo (+) mentre quello più corto il catodo (-).



Rappresentazione grafica di un LED e simbolo circuitale

Collegamento Circuitale:



Collegamento Circuitale

Codice:

L’istruzione **digitalWrite** permette di impostare lo stato logico di un PIN digitale al valore HIGH (5 Volt) o LOW (0 Volt).
Mentre l’istruzione **delay** permette di bloccare Arduino nello stato in considerazione per un certo numero di millisecondi.

Personalizzazioni: E’ possibile modificare il comportamento del circuito in questione intervenendo sul valore della variabile *ledTime*. Modificando il suo valore infatti cambia la frequenza di lampeggiamento del LED.

Approfondimento Teorico:

- Pin13: Nelle prime versioni delle schede Arduino, come riportato nella relativa documentazione: “*there is, however, about 1000 ohms of resistance on pin 13, so you can connect an LED without external resistor*” *il PIN13 presenta un’uscita limitata in correte a causa di una resistenza integrata. Grazie alla presenza di questo elemento non è necessario introdurre una resistenza esterna al fine di limitare la corrente sul LED.*