Il Bastone Sensibile per Ipovedenti [Light]

Obiettivo: Il **Bastone Sensibile per Ipovedenti** rappresenta un'attività di stampo inclusivo utilizzabile in un contesto scolastico al fine di compensare la cecità e permettere agli alunni non vedenti di superare ostacoli ed impedimenti ambientali. Tale attività permette di sviluppare attraverso un **processo meta-inclusivo** l'empatia necessaria per comprendere le difficoltà affrontate da un ragazzo ipovedente all'interno di un contesto scolastico.
Tale strumento, rappresenta l'evoluzione del più comune e tradizionale bastone assistivo per compensare la cecità e permettere agli alunni non vedenti di superare ostacoli ed impedimenti ambientali. Un sensore di prossimità, montato sul telaio, conferisce la sensibilità al bastone. A seconda della distanza calcolata attraverso il sensore, vengono generati degli impulsi sonori con durata variabile. A differenza di quanto avviene con l'uso di un bastone tradizionale, con il quale gli ostacoli vengono individuati attraverso il tocco, in questo caso basterà concentrarsi sulla durata sonora per capire la posizione di eventuali impedimenti al percorso, proprio come avviene con i sensori di parcheggio delle automobili. Durante le attività, gli alunni sono chiamati in prima persona ad affrontare e cercare di superare gli ostacoli dovuti alla cecità. Nello stesso tempo si rendono conto delle diverse-abilità che un loro pari, non vedente o ipovedente, deve necessariamente sviluppare per compensare la mancanza della piena percezione visiva.

Componenti elettronici:

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

Pre-requisiti:

Il Sensore a Ultrasuoni

Buzzer Attivo

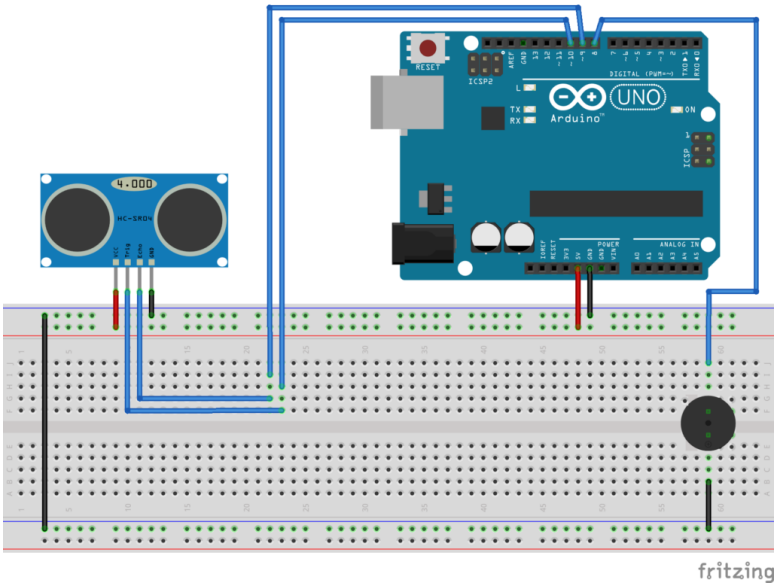
Teoria: Il **bastone sensibile per ipovedenti** rappresenta dal punto di vista tecnologico la naturale evoluzione del tradizionale sensore di parcheggio, ovvero un dispositivo ampiamente utilizzato in ambito automobilistico per favorire l'operazione del una delle moderne tecnologie che permettono al guidatore di un autoveicolo di essere a conoscenza della distanza tra la propria automobile ed un eventuale altro veicolo.
Nel caso specifico, il bastone sensibile si basa su un sensore di prossimità ad ultrasuoni (INPUT) ed un buzzer (OUTPUT) utilizzato come segnalatore acustico.
Nel dettaglio, maggiore è la distanza maggiore sarà il ritardo tra una segnalazione acustica e la successiva, analogamente, minore è la distanza minore sarà il ritardo tra una segnalazione acustica e la successiva. In particolare, la relazione ingresso uscita che lega la distanza al ritardo impiegato nella segnalazione acustica è funzione dei seguenti quattro parametri:

- DistanzaMassima: la distanza massima dopo la quale non viene più segnalato un’ostacolo.
- DistanzaMinima: la distanza minima per la quale il buzzer emette un tono continuo
- RitardoMassimo: Il ritardo tra una segnalazione acustica e la successiva nel caso di massima distanza.
- RitardoMinimo: Il ritardo tra una segnalazione acustica e la successiva nel caso di minima distanza.

Questi valori vengono utilizzati al fine di determinare l’equazione fondamentale per il calcolo del ritardo:

```
ritardo = distanza*((RitardoMax-RitardoMin)/(DistanzaMax-DistanzaMin))
```

Collegamento Circuitale:



Collegamento Circuitale

Codice:

Personalizzazioni: E’ possibile introdurre un secondo sensore ad ultrasuoni per evidenziare la presenza di un ostacolo a destra oppure a sinistra dell’utilizzatore.

Arduino Last Christmas

Obiettivo: Riprodurre la melodia Last Christmas utilizzando la piattaforma Arduino. Un progetto realizzato dall’alunno della classe 4DSA del Liceo Enrico Medi di Senigallia: Gianmarco D’Emilio

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Buzzer

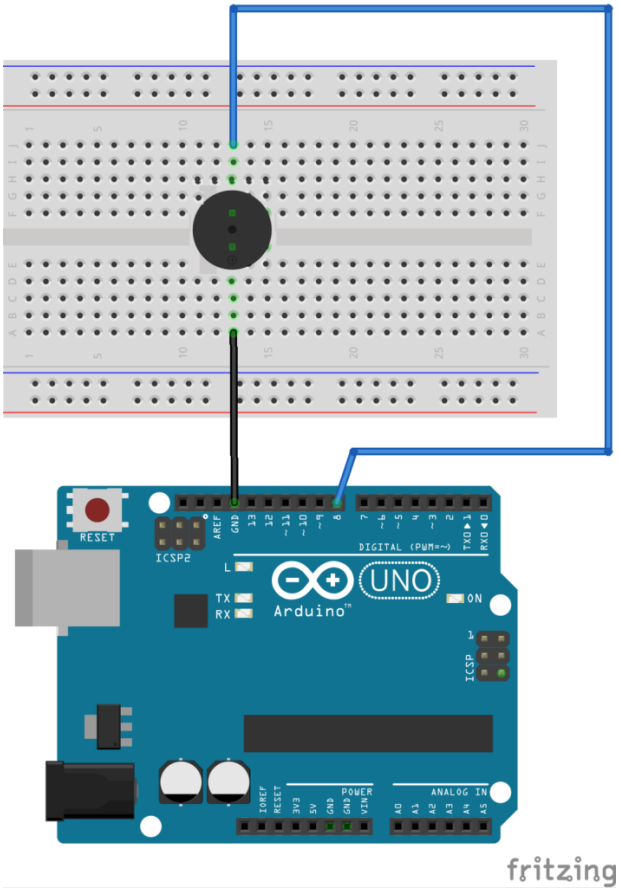
Pre-requisiti:

Buzzer Passivo

Teoria: Ogni melodia musicale è composta da note e pause. Se

le pause possono essere riprodotte utilizzando la funzione **delay()** di Arduino (già vista negli esempi precedenti), le singole note possono essere facilmente generate grazie all'utilizzo di un buzzer passivo e della funzione **tone**(). Nel dettaglio, l'impiego della funzione tone permette di selezionare la frequenza riprodotta dal buzzer e la relativa durata della nota.
Nel caso in questione, l'intera melodia (comprensiva di note, durata delle note e pause, viene salvata in due differenti array (vettori) e riprodotta sequenzialmente come un vero spartito musicale.

Collegamento Circuitale:



Collegamento Circuitale

Codice:

Le frecce dell’AUDI con Arduino

Obiettivo: Realizzare un sistema di controllo dei led che simula l’effetto delle frecce di un Audi. Le luci si accendono alla pressione di un pulsante.

Componenti elettronici:

- Arduino UNO
- Breadboard
- 8 Resistenze da 220 Ohm per i led
- 4 LED
- 2 Resistenza da 10KOhm per i pulsanti
- 2 Pulsanti

Pre-requisiti:

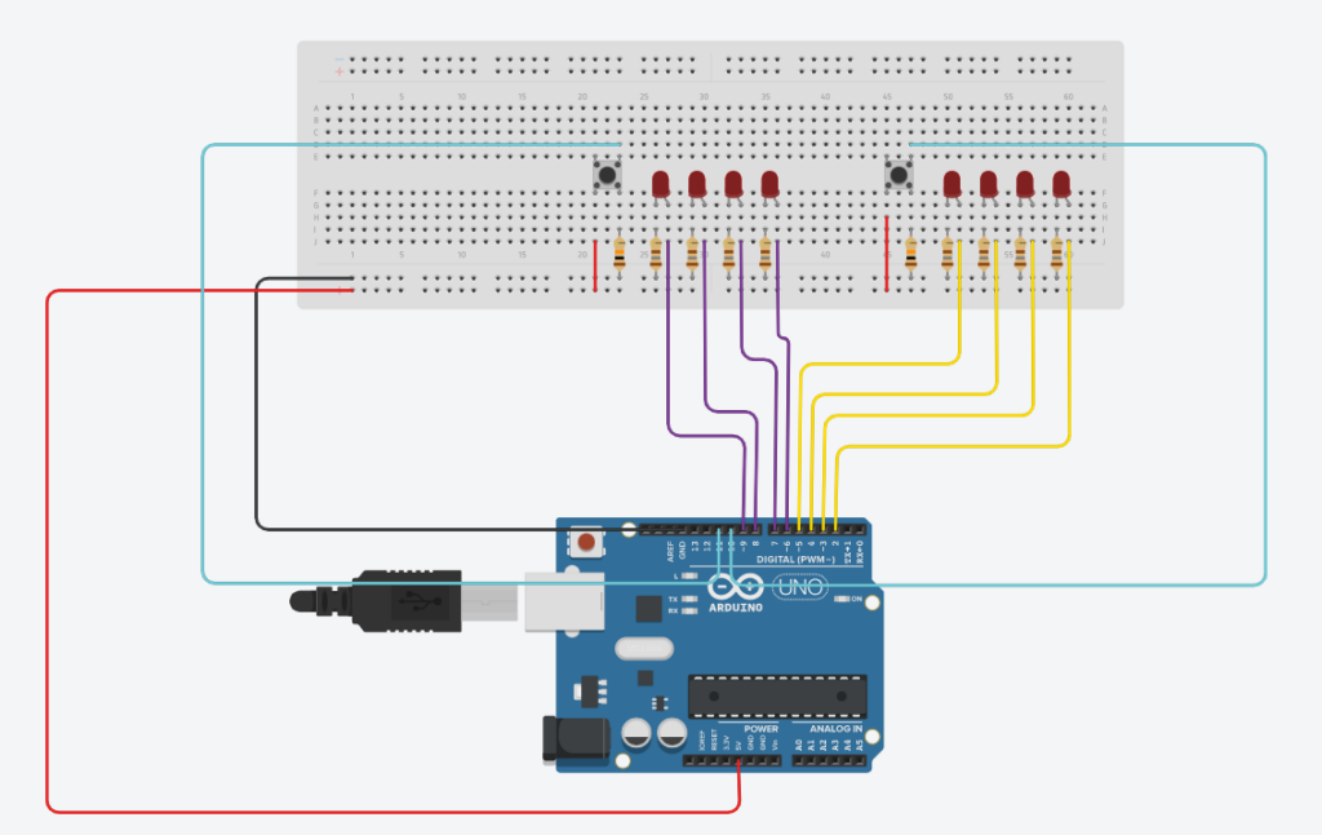
LED e Pulsante

Teoria: Come abbiamo detto, lo scopo dell’esercitazione è quello di attivare due barre LED tramite la pressione di due

pulsanti. Le barre simulano l’effetto freccia presente nelle auto Audi. La pressione del pulsante dà il comando di avvio della sequenza e, se si tiene premuto il pulsante, la sequenza rimane attiva.

Sarà necessario sistemare il codice in base a come viene montato il pulsante (se in pull-up o in pull-down)

Collegamento Circuitale:



Codice:

Il codice è basato sulla lettura del segnale del pulsante (nel nostro caso in pull-down) che, se premuto, attiva la relativa sequenza

Esperienza realizzata dalla classe 3 BMC del Dipartimento di Meccanica dell'ITIS "E.Mattei" di Urbino nell'AS 2022-23. Codice e Thinkercad realizzato da Giacomo Brancorsini

---

Realizzare un Cronometro Digitale con Arduino

Obiettivo: Realizzare un cronometro digitale per misurare lo scorrere del tempo in millisecondi utilizzando il microcontrollore Arduino, un display LCD e due pulsanti
Un progetto realizzato dagli alunni della classe 3ATLC dell'Istituto Tecnico Industriale "Enrico Mattei" di Urbino:
– Benedetti Nicolas
– Puca Edoardo

Componenti elettronici:

- Arduino UNO
- Breadboard
- 2 Resistenze da 100 Ohm per i pulsanti
- 1 Display LCD
- 2 Pulsanti

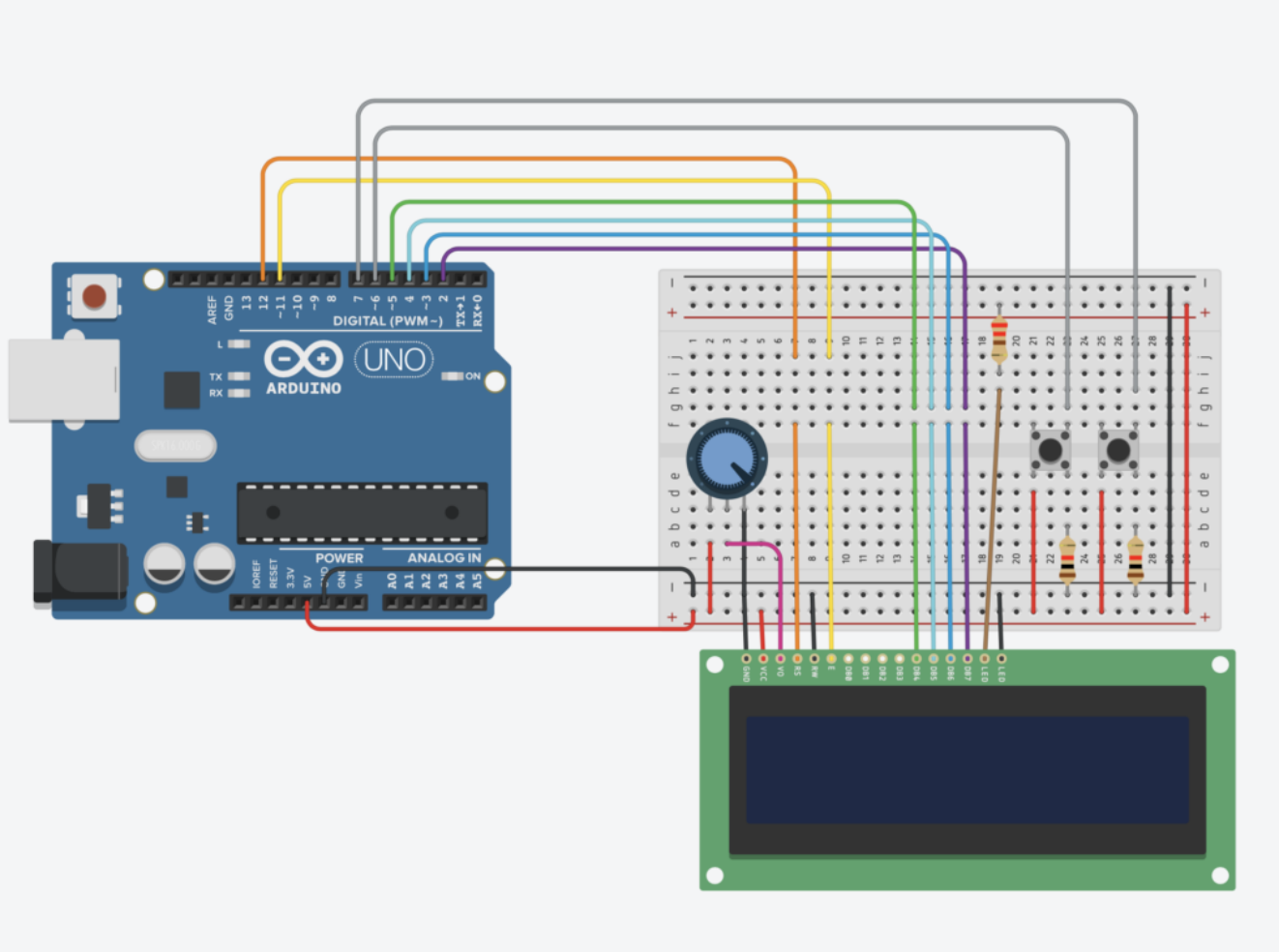
Pre-requisiti:

Blinking Led Senza Delay: MILLIS()

Teoria: Realizzare un cronometro digitale con il controllore Arduino, è relativamente semplice. Per implementare questo dispositivo, basta collegare arduino ad un display LCD ed utilizzare l'istruzione millis() nell'apposito sketch.
Nel dettaglio, la funzione millis restituisce il numero di millisecondi che sono passati da quando la board Arduino ha eseguito il programma corrente. Questo numero si riazzera dopo circa 50 giorni.
A tale fine è importante ribadire che mediante la funzione millis è possibile effettuare più misurazioni nel corso del tempo e per calcolare il tempo trascorso tra una misurazione e l'altra basta fare al differenza:

$$tempo = misura2 - misura1$$

Collegamento Circuitale:



Collegamento Circuitale

Codice:

Attraverso l'utilizzo della funzione millis è possibile realizzare un cronometro digitale con il controllore Arduino. Nel caso specifico due differenti misure vengono effettuate per determinare il tempo trascorso. La prima misura viene salvata nella variabile "*tempoI*" mentre la seconda nella variabile "*tempoF*". Il tempo trascorso tra le due differenti misure viene memorizzato nella variabile"*tempo*" ottenuta come la differenza tra "*tempoF*" e "*tempoI*".

Tinkercad:

Realizzare un Voltmetro [0-5Volt] con Display a 7 Segmenti

Obiettivo: Realizzare un dispositivo per misurare tensioni nel range 0.5 V utilizzando il microcontrollore Arduino ed il display a sette segmenti.
Un progetto realizzato dagli alunni della classe 3AUT dell'Istituto Tecnico Industriale "Enrico Mattei" di Urbino:
– Amadori Federico
– Fucili Elia

Componenti elettronici:

- Arduino UNO
- Breadboard
- 14 Resistenze da 100 Ohm per I display a sette segmenti
- 2 Display a 7 Segmenti

Pre-requisiti:

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

Le Funzioni digitalWrite, digitalRead, analogWrite e analogRead

Teoria: Effettuare la misura di una tensione compresa nel range 0-5 V, utilizzando il controllore Arduino, è relativamente semplice. Per effettuare questa misura, basta collegare l'elemento di cui vogliamo analizzare la tensione ad un ingresso analogico (A0-A5) e utilizzare l'istruzione analogRead nell'apposito sketch.
A tale fine è importante ribadire che l'ingresso analogico compreso nel range 0-5V viene mappato utilizzato la funzione analogRead(A0) nel range 0-1023, pertanto è indispensabile utilizzare una corrispettiva funzione di conversione basta sulla seguente proporzione:

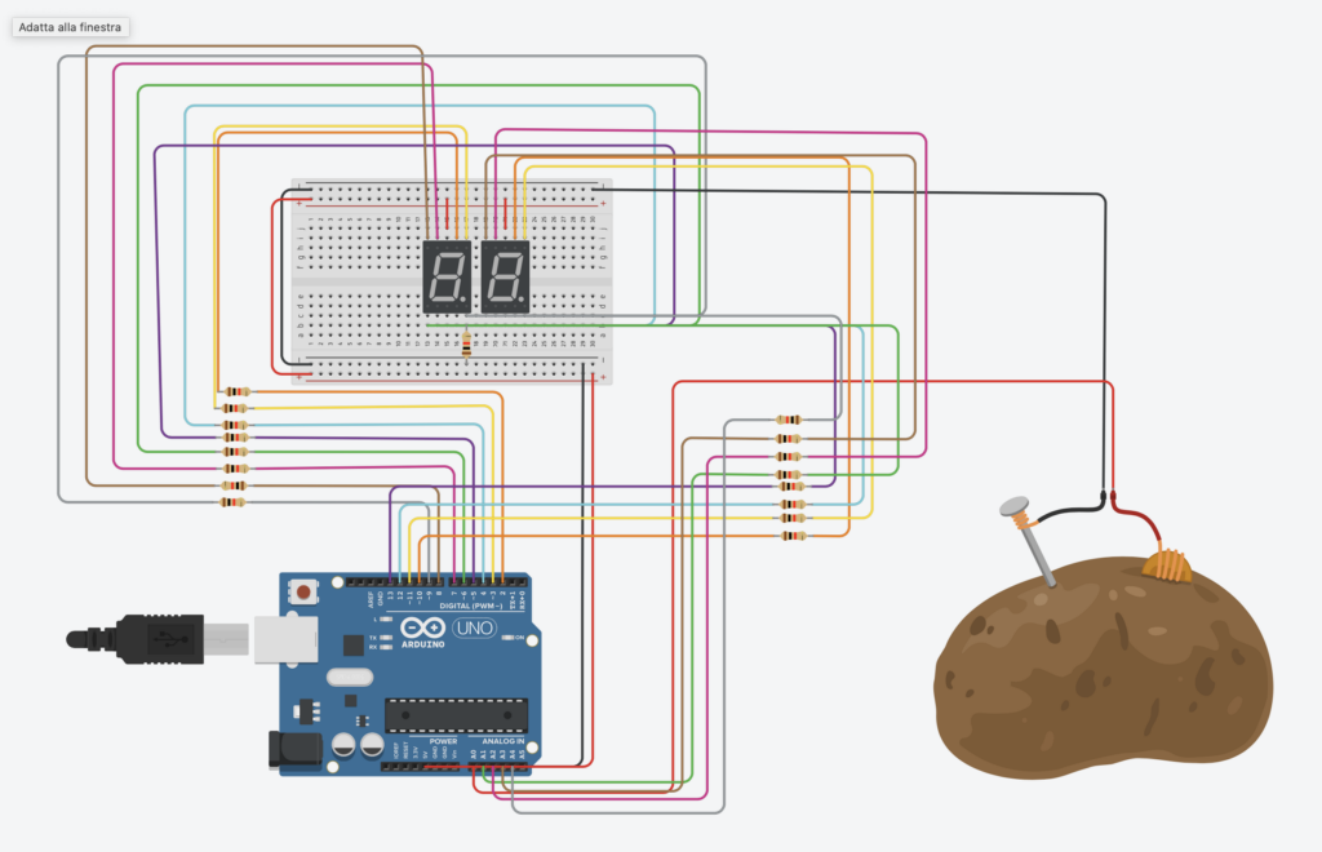
ValoreAnalogRead : 1023 = ValoreTensione : 5

In conclusione, il valore di tensione può essere semplicemente ottenuto dividendo per 1023 e moltiplicando per 5 il valore letto utilizzando la funzione analogRead(A0).

Tale valore può essere poi visualizzato su differenti tipi di display: monitor del computer, display LCD e display a 7 segmenti. Nell'esempio in questione vengono utilizzati due display a 7 segmenti (uno per la parte intera ed uno per la

parte decimale) al fine di visualizzare la tensione misurata.

Collegamento Circuitale:



Collegamento Circuitale

Codice:

Attraverso l’utilizzo di due `display a sette segmenti` è possibile visualizzare la tensione prodotta dalla “patata”. Nel caso specifico viene utilizzata una patata come generatore di tensione per verificare che il dispositivo fornisca una valutazione corretta della tensione misurata. Delle funzioni specifiche (e.g., zeroI, unoI, zeroD, unoD, etc) vengono utilizzate per visualizzare i vari numeri nei due differenti display.

Tinkercad:

Personalizzazioni: E’ possibile utilizzare altre tipologie di display per visualizzare la tensione misurata.

Realizzare un Contapunti con Display LCD

Obiettivo: Realizzare un contapunti manuale utilizzando il microcontrollore Arduino ed il display LCD

Componenti elettronici:

- Arduino UNO
- Breadboard
- 2 Resistenze da 1kOhm per i pulsanti
- 1 Resistenza da 100 Ohm per il display LCD
- 1 Trimmer (per regolare il contrasto del display LCD)
- 2 Pulsanti

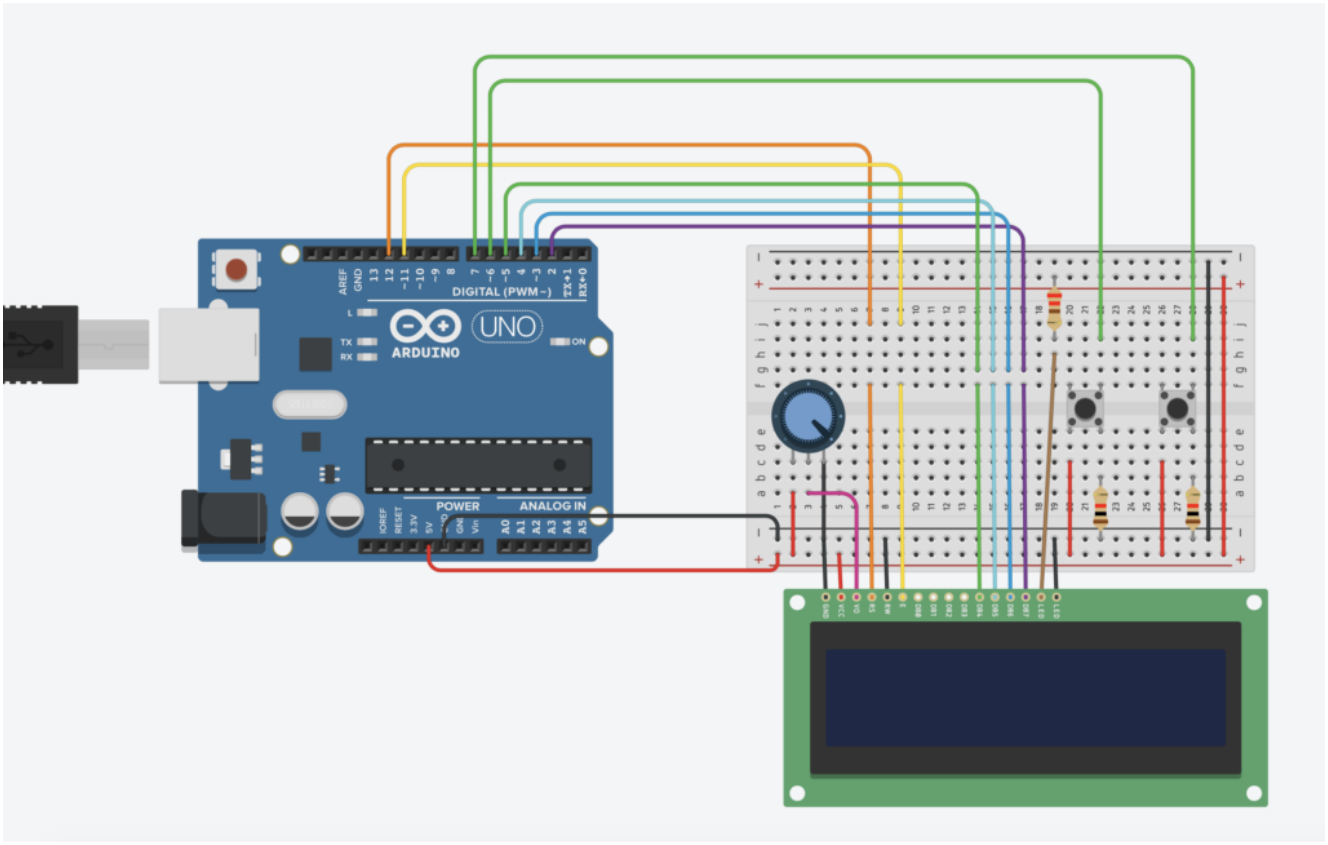
Pre-requisiti:

Come Collegare un Display LCD ad Arduino

Pulsante come Interruttore

Teoria: Attraverso l’utilizzo di due semplici pulsanti e di un display LCD è possibile realizzare un contapunti digitale per riprodurre il punteggio di una partita di biliardo, pallavolo, calcetto, etc. Ogni volta che uno dei due pulsanti è premuto viene incrementato il punteggio di una delle squadre.

Collegamento Circuitale:



Codice:

Attraverso l'utilizzo di due contatori *cnt1* e *cnt2* è possibile memorizzare il punteggio di ogni squadra. Tali valori sono riprodotti sul display LCD utilizzando l'istruzione *LCD.print()*.
Per evitare delle letture multiple che potrebbero portare ad un comportamento errato del circuito si utilizza una variabile globale che memorizza lo stato precedente del pulsante (i.e., *valButton1Old* e *valButton2Old*).

Personalizzazioni: E' possibile introdurre un ulteriore pulsante per resettare il punteggio senza necessariamente dovere riavviare il controllore Arduino. Inoltre si può

aggiungere un dispositivo di segnalazione acustica per avvisare l'utilizzatore del cambio di punteggio.

Utilizzare e Creare una Libreria per il Display a 7 Segmenti

Obiettivo: Utilizzare e creare una libreria (file header e cpp) per un Display a Sette Segmenti.

Puoi scaricare i file di libreria cliccando nel seguente link: http://www.arduinofacile.it/wp-content/uploads/2021/03/SevenSegment.zip
I file scaricati devono essere inseriti all'interno della cartella di progetto insieme al file .ino

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Display a 7 Segmenti
- 8 Resistenze

Pre-requisiti:

Creare funzioni con Arduino … per un Display a 7 Segmenti

Teoria: la realizzazione di funzioni di libreria permette di facilitare l'operazione di **riutilizzo del codice** rendendo più veloce e più rapido lo sviluppo. Nel caso specifico la funzione di libreria implementata sarà costituta da un file header (.h) e da un file sorgente (.cpp).

Un file header è un file di testo che contiene i prototipi dei metodi (funzioni) definite nel relativo file sorgente. Nel caso in questione il file header contiene anche la dichiarazione della classe "SevenSegment" utilizzata per modellare il display a sette segmenti.
Tale classe sarà caratterizzata da 10 attributi:

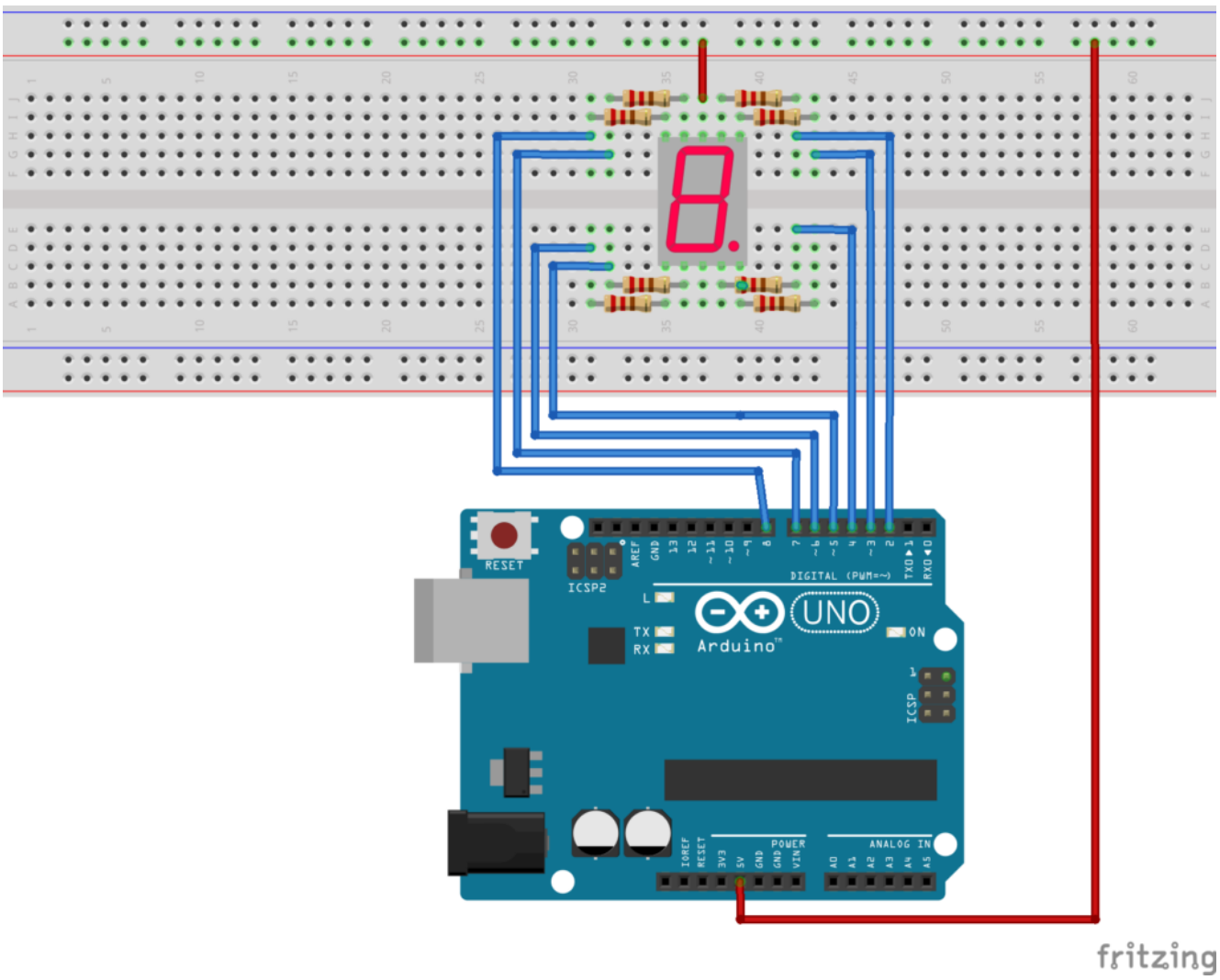
- **int pinA**: il pin A del display a sette segmenti
- **int pinB**: il pin B del display a sette segmenti
- **int pinC**: il pin C del display a sette segmenti
- **int pinD**: il pin D del display a sette segmenti
- **int pinE**: il pin E del display a sette segmenti
- **int pinF**: il pin F del display a sette segmenti
- **int pinG**: il pin G del display a sette segmenti
- **int pinDP**: il pin DP del display a sette segmenti
- **bool isCommonAnode**: indica se il display è di tipo anodo comune oppure no

e da 11 metodi

- **void Print0()**: metodo utilizzato per stampare il numero 0
- **void Print1()**: metodo utilizzato per stampare il numero 1
- **void Print2()**: metodo utilizzato per stampare il numero 2
- **void Print3()**: metodo utilizzato per stampare il numero 3
- **void Print4()**: metodo utilizzato per stampare il numero 4
- **void Print5()**: metodo utilizzato per stampare il numero 5
- **void Print6()**: metodo utilizzato per stampare il numero 6
- **void Print7()**: metodo utilizzato per stampare il numero 7
- **void Print8()**: metodo utilizzato per stampare il numero 8
- **void Print9()**: metodo utilizzato per stampare il numero 9
- **void CountDown()**: metodo utilizzato per eseguire il countdown.

Nel file sorgente viene invece riportata l'implementazione dei prototipi delle funzioni dichiarate nel file header.

Collegamento Circuitale:



Collegamento Circuitale

Codice:

Vengono in seguito riportate le tre porzioni di codice utilizzate per creare la funzione di libreria relativa al display a sette segmenti.

- File Header: contiene la definizione della classe con i propri attributi (i.e., pinA, pinB, etc) ed i prototipi dei relativi metodi.

- File Sorgente: contiene le implementazioni dei metodi riportati nel file header.

- File Arduino: Utilizzato per fornire un esempio di come utilizzare la libreria per la gestione del display a sette segmenti.

Se tutti i file sono correttamente posizionati sullo stesso livello all’interno della cartella di progetto, due nuove tab compariranno nell’ambiente di sviluppo utilizzato per programmare Arduino. Attraverso queste tab sarà possibile visionare e modificare il file sorgente (.cpp) ed il file header (.h)

IDE con l’utilizzo della libreria SevenSegment.h

Arduino Cyclone Arcade Game

Obiettivo: Realizzare un gioco Arcade, basato su LED e

pulsanti, utilizzando il microcontrollore Arduino.

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Resistenze da 1kOhm per i pulsanti
- 8 Resistenze da 100 Ohm per i LED.
- 1 Buzzer Passivo (per la realizzazione di una melodia)
- 8 LED (7 LED Rossi ed 1 Led Verde)
- 1 Pulsante

Pre-requisiti:

Pulsante come Interruttore

http://www.arduinofacile.it/2020/03/23/blinking-led-senza-delay-millis/

Buzzer Passivo

Teoria: Lo scopo di questo progetto è quello di creare un

semplice gioco per bambini interattivo e divertente, sfruttando il microcontrollore Arduino. Il progetto proposto prende spunto da un gioco realmente prodotto e commercializzato negli anni 80 denominato “Cyclon jr” (vedi figura).

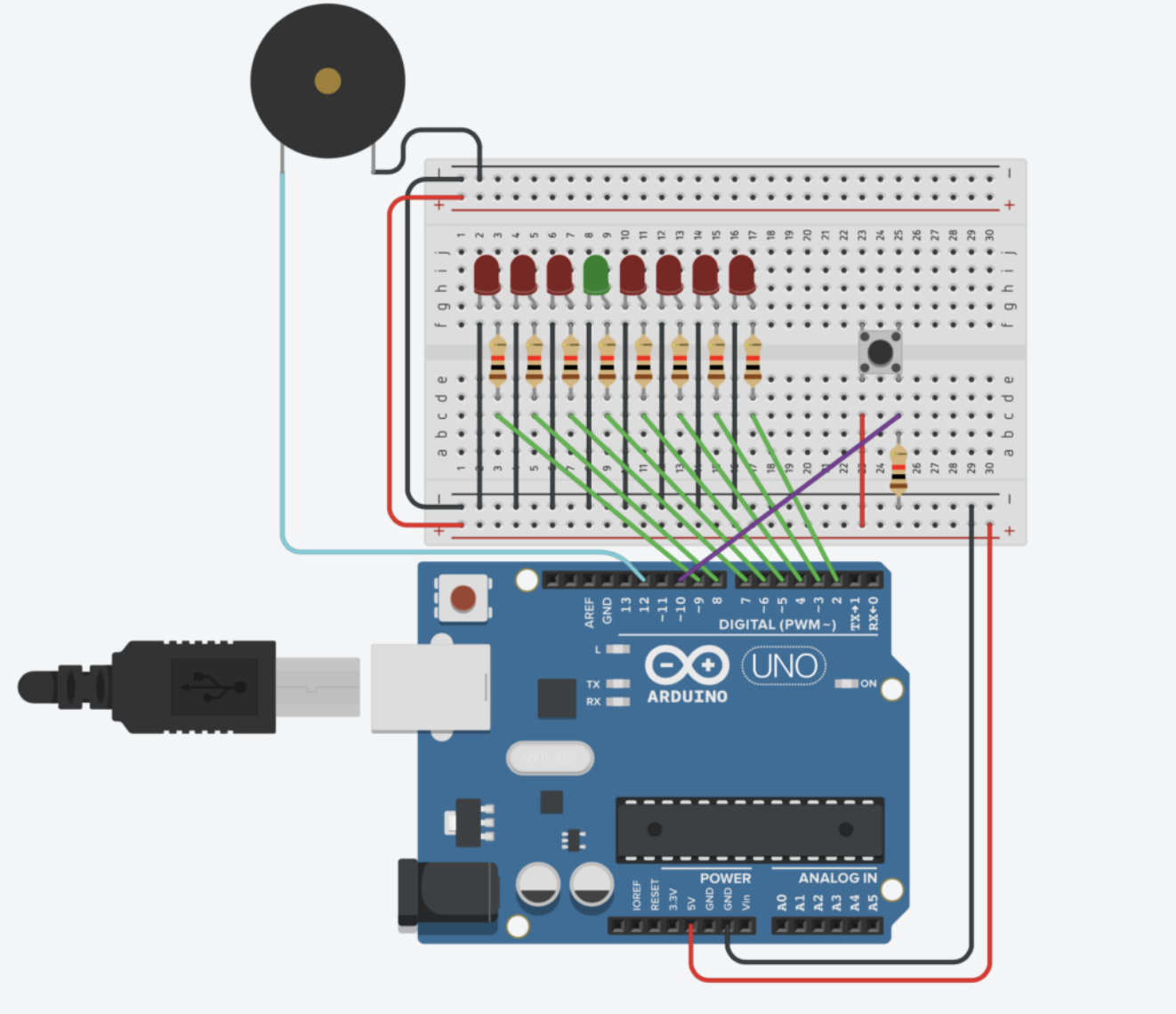
Cyclone Jr

Il progetto è di facile realizzazione e si basa sull’utilizzo

di un pulsante e di alcuni LED che si accenderanno in sequenza. L’obiettivo del gioco è premere il pulsante quando la luce del cyclone raggiunge l’unico LED verde presente nel tabellone di gioco. In caso di successo, il livello di difficoltà aumenterà ed i led si accenderanno più velocemente. Differente, in caso di sconfitta, il gioco ricomincerà da capo ed i led si accenderanno più lentamente.

L’utilizzo di un buzzer passivo permette di generare un segnale acustico in caso di vittoria o sconfitta. Analogamente una gioco di luci sarà avviato ogni volta che il giocatore preme il pulsante in modo corretto.

Collegamento Circuitale:



Collegamento Circuitale

Codice:

Sebbene le componenti hardware impiegate nel progetto in questione sono di comune utilizzo, il codice non è dei più immediati. Innanzitutto, è importante considerare che per fare lampeggiare i LED è stata utilizzata la funzione millis() a discapito della tradizionale delay. L'impiego della funzione millis() permette infatti una maggiore reazione nel rilevare la pressione di un pulsante. Inoltre l'impiego dell'operatore modulo % permette di gestire in modo facile la corretta accensione dei led in sequenza. La difficoltà è infine regolata dal parametro K che aumenta ad ogni vittoria.

Personalizzazioni: E’ possibile aggiungere un numero maggiore di led per rendere il gioco più completo.

L’Albero di Natale (Gioco Luci + Melodia)

Obiettivo: Riprodurre la melodia “Merry Christmas” e creare un gioco luci Natalizio utilizzando la piattaforma Arduino (senza utilizzare la funzione delay).

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Buzzer
- 6 Resistenze (100 Ohm)
- 6 Led (Possibilmente rossi)

Pre-requisiti:

Arduino Jingle Bells

Blinking Led Senza Delay: MILLIS()

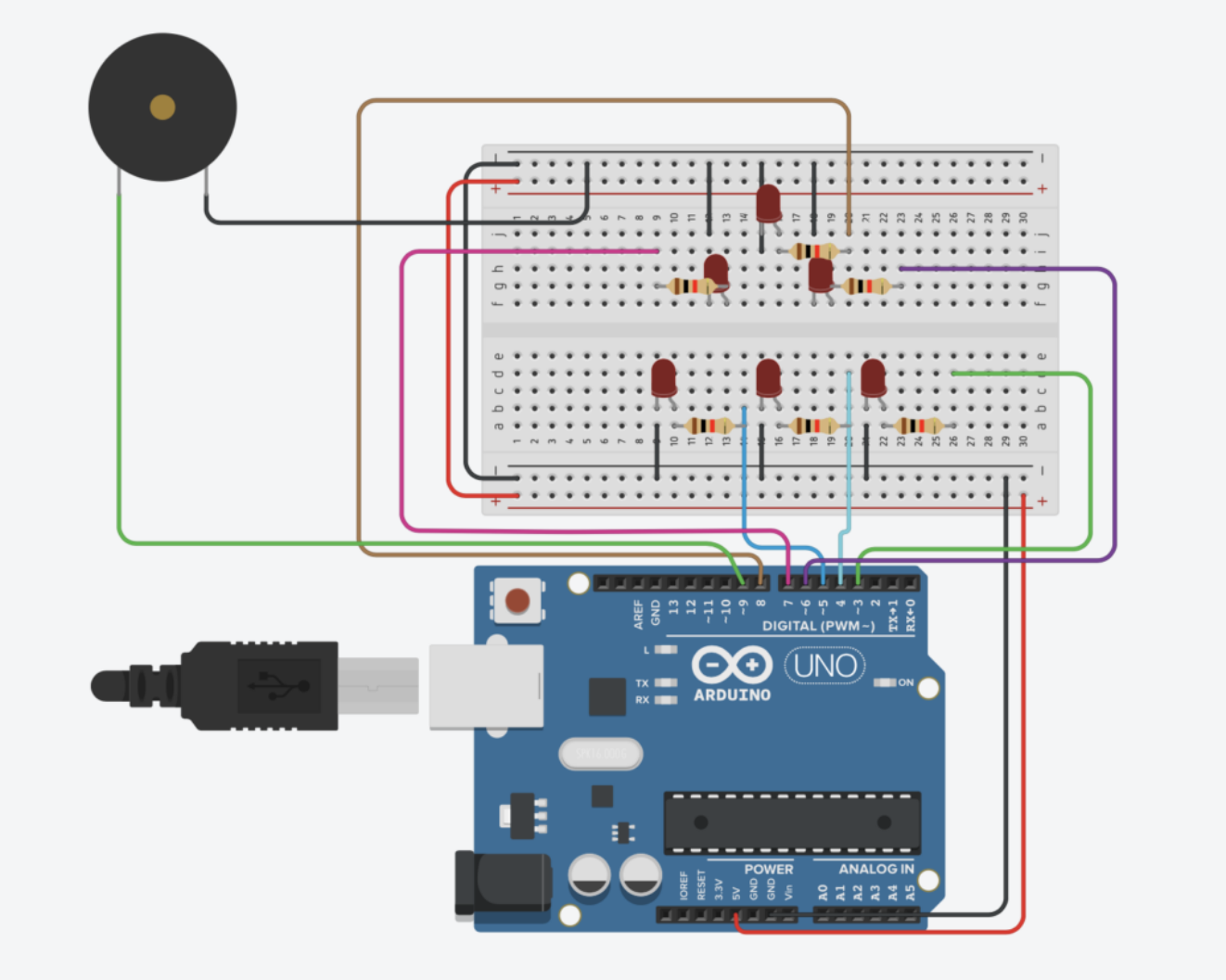
Teoria: Nelle lezioni precedenti è stato illustrato come ogni melodia musicale è composta da note e pause. Se le singole note possono essere facilmente generate grazie all'utilizzo di un buzzer passivo e della funzione **tone**() le pause possono essere riprodotte utilizzando la funzione **delay()**.
Tuttavia è importante considerare che sebbene la funzione delay risulti molto pratica e permetta una facile realizzazione di giochi luci o riproduzione di melodie, questa produce un blocco del controllore il quale può impedire il corretto funzionamento di altre operazioni.
Nel caso specifico, se utilizzassimo la funzione delay sia per gestire il gioco luci sia per riprodurre la melodia, la funzione delay utilizzata in entrambi i task andrebbe a danneggiare la corretta esecuzione di una delle due attività. Ad esempio, si avrebbero delle pause troppo lunghe nella melodia rendendola incomprensibile.

Per risolvere questo problema si è deciso di utilizzare la funzione millis sia per realizzare il gioco luci sia per implementare la melodia.

La funzione millis restituisce il numero di millisecondi che sono passati da quando la board Arduino ha eseguito il programam corrente. Questo numero si riazzera dopo circa 50

giorni.

Collegamento Circuitale:



Circuito Elettrico

Codice:

Arduino Jingle Bells

Obiettivo: Riprodurre la melodia “Jingle Bells” utilizzando la piattaforma Arduino.

Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Buzzer

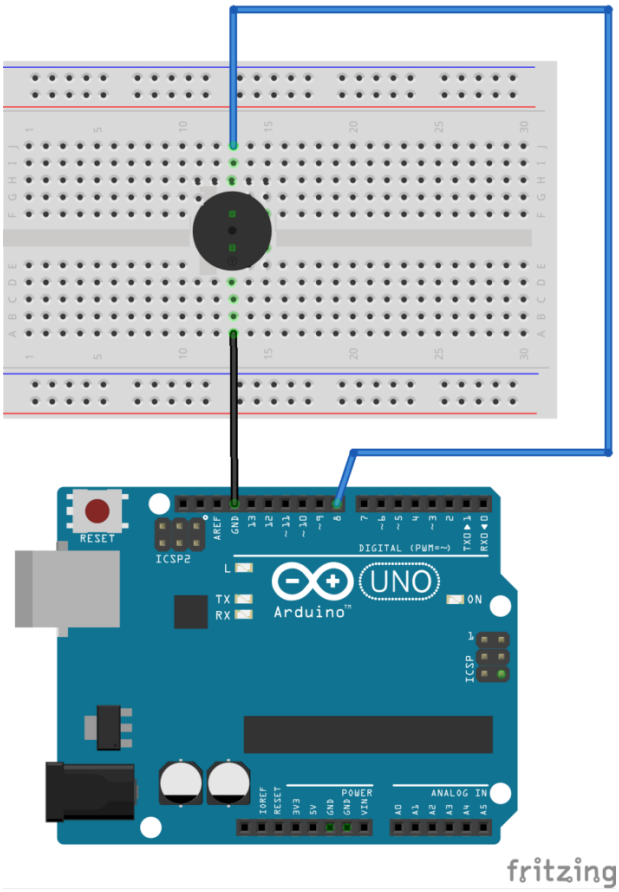
Pre-requisiti:

Buzzer Passivo

Teoria: Ogni melodia musicale è composta da note e pause. Se le pause possono essere riprodotte utilizzando la funzione **delay()** di Arduino (già vista negli esempi precedenti), le singole note possono essere facilmente generate grazie all’utilizzo di un buzzer passivo e della funzione **tone**(). Nel dettaglio, l’impiego della funzione tone permette di selezionare la frequenza riprodotta dal buzzer e la relativa durata della nota.
Nel caso in questione, l’intera melodia (comprensiva di note, durata delle note e pause, viene salvata in due differenti array (vettori) e riprodotta sequenzialmente come un vero

spartito musicale.

Collegamento Circuitale:



Collegamento Circuitale

Codice: