

# 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()`.

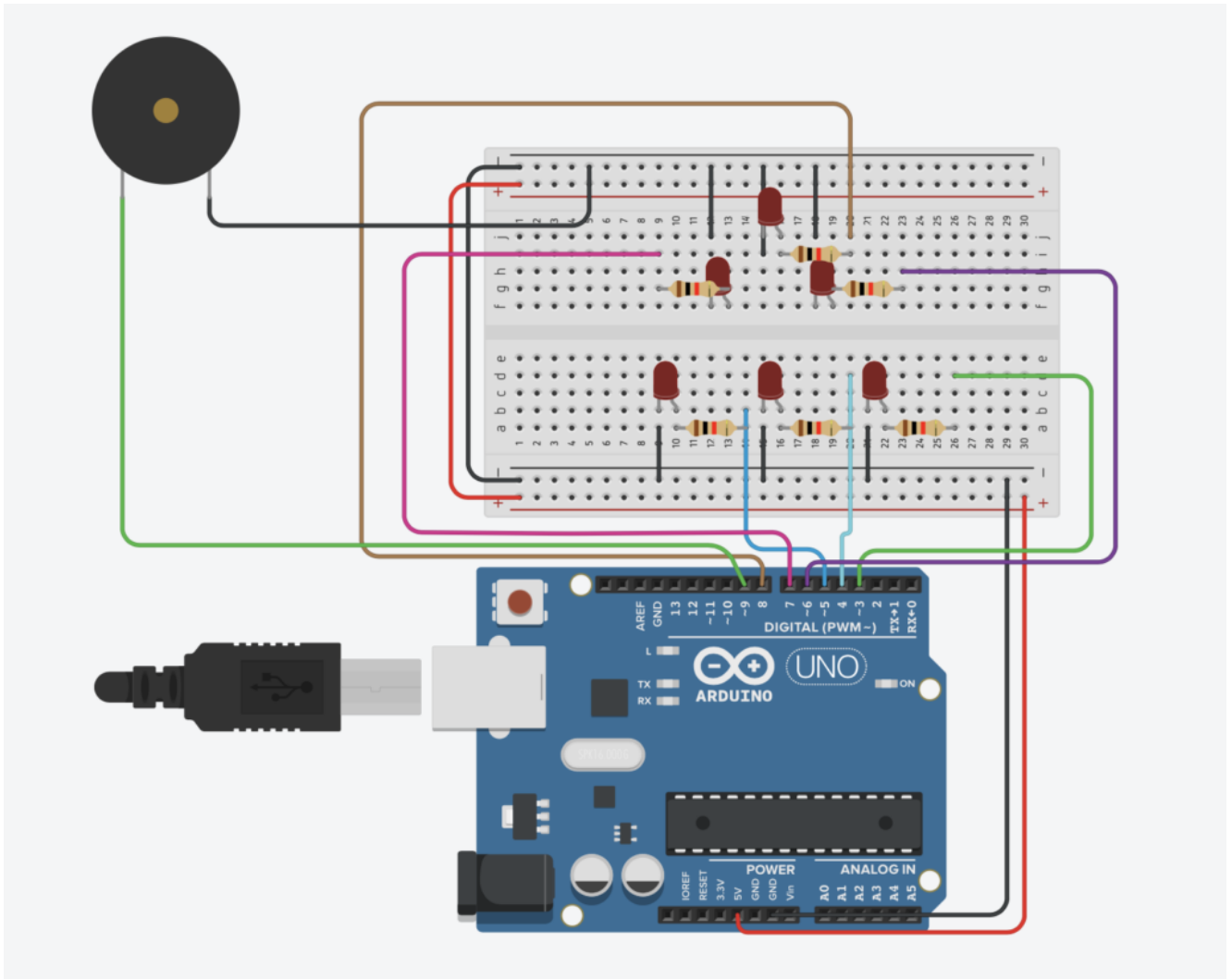
Tuttavia è importante considerare che sebbene la funzione delay risulti molto pratica e permetta una facile realizzazione di giochi luci o riproduzione di melodie, questa produce un blocco del controllore il quale può impedire il corretto funzionamento di altre operazioni.

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 riazzerà 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

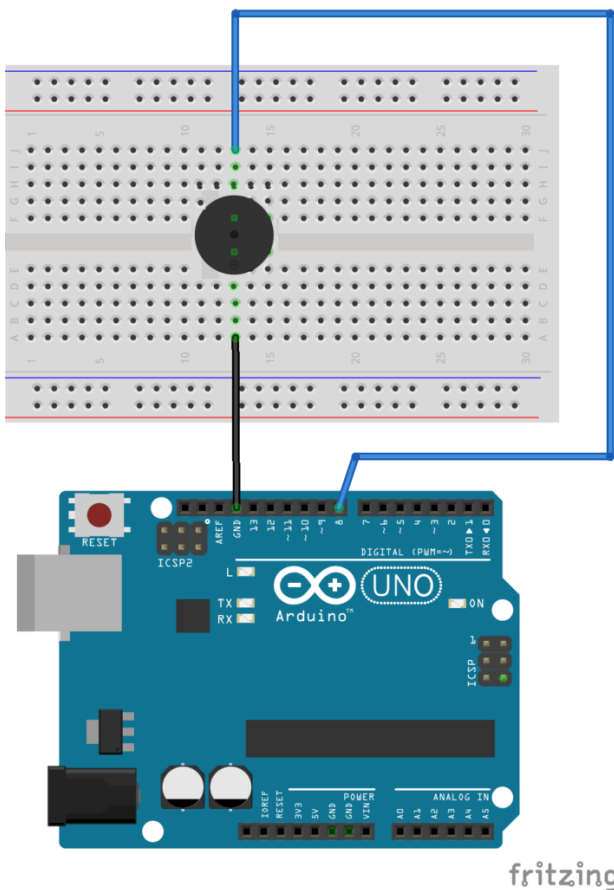
## Pre-requisiti:

### [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:**

---

# Arduino Merry Christmas

**Obiettivo:** Riprodurre la melodia Merry Christmas utilizzando la piattaforma Arduino.

## Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Buzzer

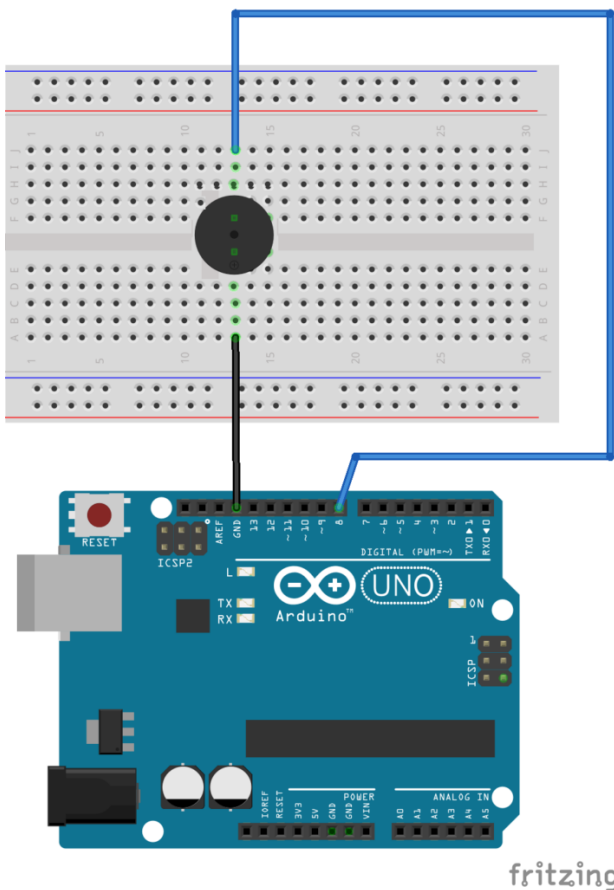
## Pre-requisiti:

### [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:**

---

# Arduino Pirati dei Caraibi

**Obiettivo:** Riprodurre la melodia del film “Pirati dei Caraibi” utilizzando la piattaforma Arduino.

## Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Buzzer

## Pre-requisiti:

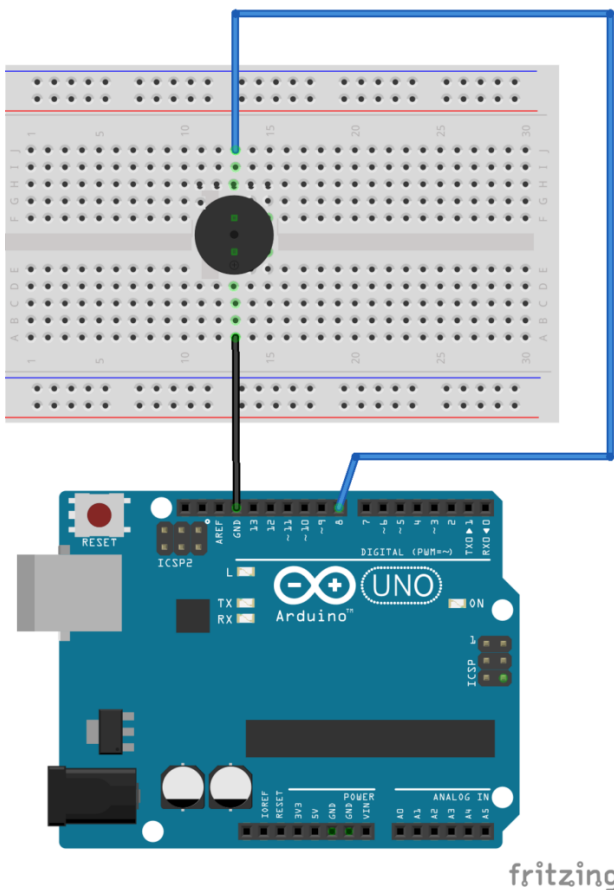
### [Buzzer Passivo](#)

**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:





fritzing

Collegamento Circuitale

**Codice:**

---

# Arduino SuperMario

**Obiettivo:** Riprodurre la melodia del famoso videogioco utilizzando la piattaforma Arduino.

## Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Buzzer

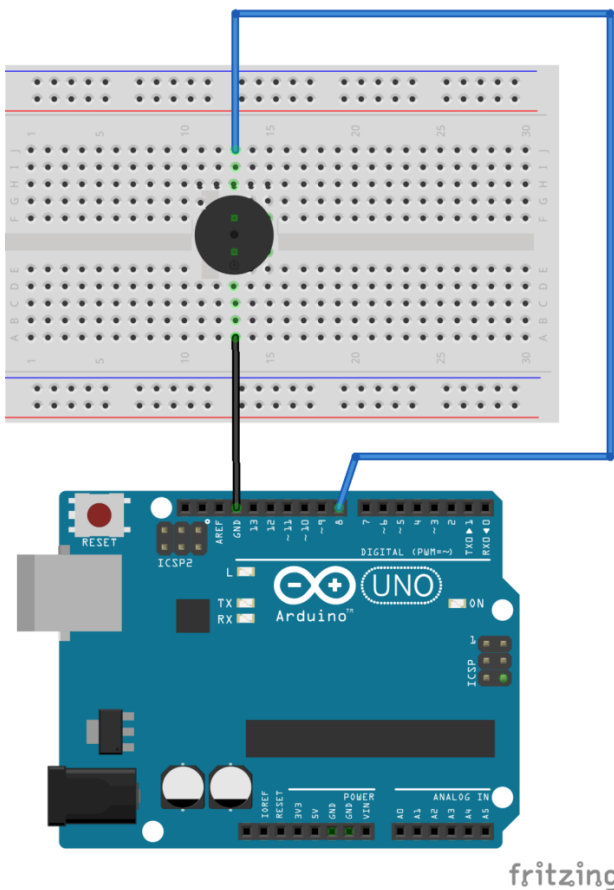
## Pre-requisiti:

### [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:



fritzing

Collegamento Circuitale

**Codice:**

---

# Arduino Happy Birthday

**Obiettivo:** Realizzare un bigliettino di auguri di buon compleanno utilizzando la piattaforma Arduino.

## Componenti elettronici:

- Arduino UNO
- Breadboard
- 1 Buzzer

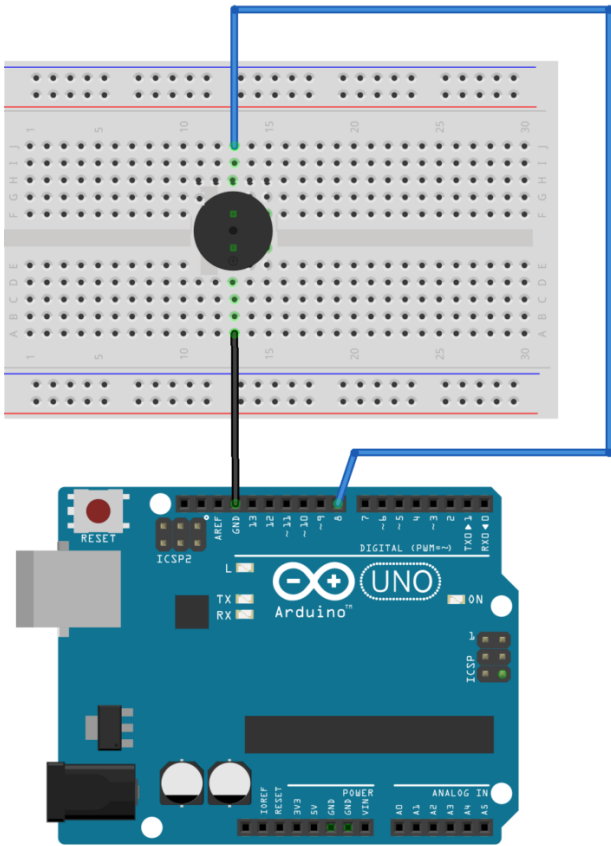
## Pre-requisiti:

### [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:



fritzing

## Collegamento Circuitale

**Codice:**