SYNACORP TRADING & SERVICES
No.9, 1st Floor, Lorong 1/SS2, Bandar Tasek Mutiara, 14120 Simpang Ampat, S.Prai (S), Penang
Tel : +604.502.1726 Hunting Line : 012.403.3474 Fax : +604.502.1726
(Website) http://www.synacorp.my (Email) sales@synacorp.com.my

Electronic Piezo Buzzer 3-24VDC for Arduino

Introduction

The piezo buzzer produces sound based on reverse of the piezoelectric effect. The generation of pressure variation or strain by the application of electric potential across a piezoelectric material is the underlying principle. These buzzers can be used alert a user of an event corresponding to a switching action, counter signal or sensor input. They are also used in alarm circuits.

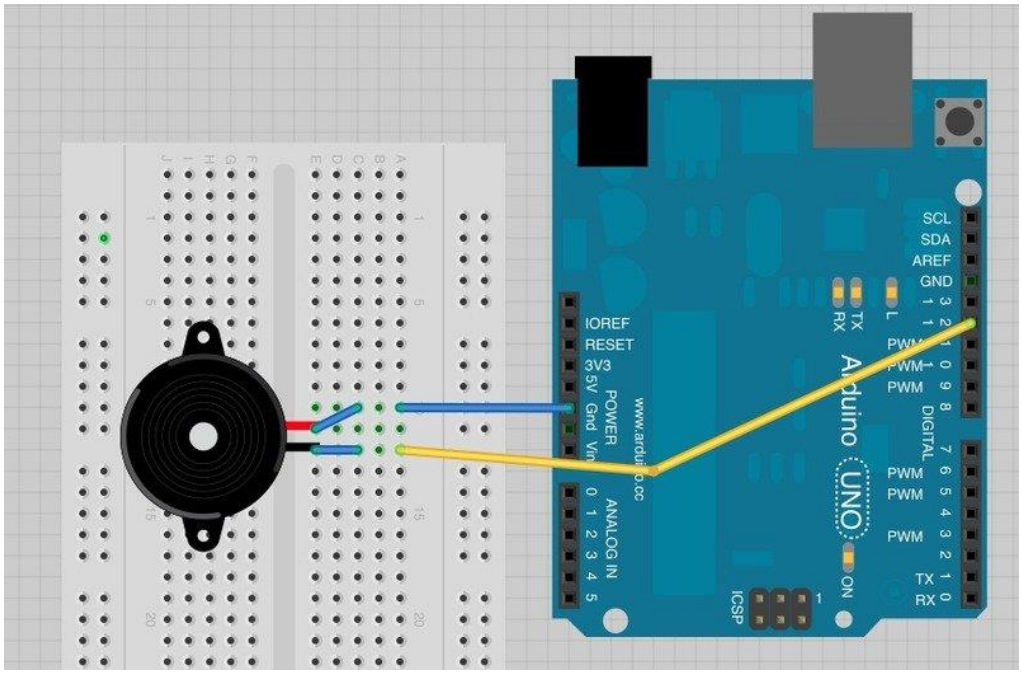
The buzzer produces a same noisy sound irrespective of the voltage variation applied to it. It consists of piezo crystals between two conductors. When a potential is applied across these crystals, they push on one conductor and pull on the other. This, push and pull action, results in a sound wave. Most buzzers produce sound in the range of 2 to 4 kHz.

In this lesson, you will learn how to make sounds with your Arduino. First you will make the Arduino play a 'musical' scale and then combine this with a photocell, to make a Theremin-like instrument that changes the pitch played as you wave your hand over the photocell.

Specification

- Operating Voltage: 3 - 24VDC
- Rated Voltage: 5VDC
- Size: 17mm (D) x 7mm (H)
- Rated Current: < 35mA

Image Set Up Diagram



Packing List

- Piezo sounder
- Half-size Breadboard
- Arduino Uno R3
- Jumper wire pack

Pin Assignment

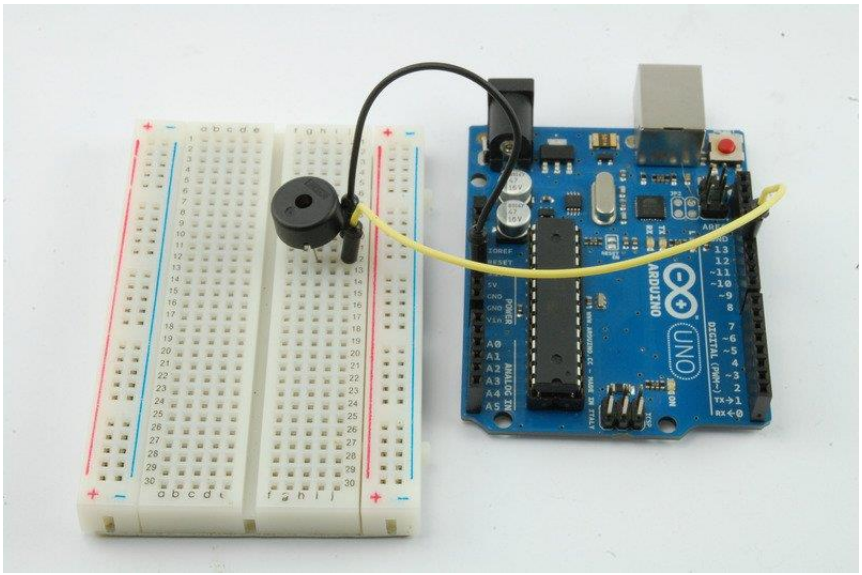
There are only 2 pins used to interface electronic piezo buzzer.

1. One pin of the piezo sounder goes to GND connection
2. The other to digital pin 12.

Pin Connection Of Electronic Piezo Buzzer

Playing a Scale

- For the first part of this lesson, the only thing on the breadboard is the Piezo buzzer. One pin of the piezo sounder goes to GND connection and the other to digital pin 12.



Example Code

1. This is example code Playing a Scale for electronic piezo buzzer.

```
/*
Adafruit Arduino - Lesson 10. Simple Sounds
*/

int speakerPin = 12;

int numTones = 10;
int tones[] = {261, 277, 294, 311, 330, 349, 370, 392, 415, 440};
//          mid C  C#   D    D#   E    F    F#   G    G#   A

void setup()
{
  for (int i = 0; i < numTones; i++)
  {
    tone(speakerPin, tones[i]);
    delay(500);
  }
  noTone(speakerPin);
}

void loop()
```

```
23. {
24. }
```

- To play a note of a particular pitch, you specify the frequency. See the following section on sound. The different frequencies for each note are kept in an array. An array is like a list. So, a scale can be played by playing each of the notes in the list in turn.

- The 'for' loop will count from 0 to 9 using the variable 'i'. To get the frequency of the note to play at each step, we use 'tone[i]'. This means, the value in the 'tones' array at position 'i'. So, for example, 'tones[0]' is 261, 'tones[1]' is 277 etc.

- The Arduino command 'tone' takes two parameters, the first is the pin to play the tone on and the second is the frequency of the tone to play.

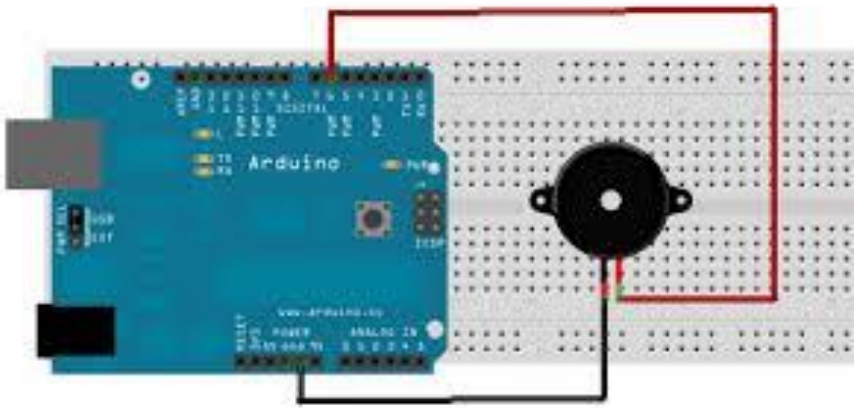
- When all the notes have been played, the 'noTone' command stops that pin playing any tone.

- We could have put the tone playing code into 'loop' rather than 'setup', but frankly the same scale being played over and over again gets a bit tiresome. So in this case, 'loop' is empty.

- To play the tune again, just press the reset button.

Piezo “Happy Birthday”

- The second song in the piezo song series.
- First plug in one side of the piezo to pin 9 and the other to ground.
- Then upload the code to your arduino.



Example Code

2. This is example code “happy birthday” for electronic piezo buzzer.

```
int speakerPin = 9;

int length = 28; // the number of notes

char notes[] = "GGAGcB GGAGdc GGxecBA yyecdc";

int beats[] = { 2, 2, 8, 8, 8, 16, 1, 2, 2, 8, 8,8, 16, 1, 2,2,8,8,8,8,16, 1,2,2,8,8,8,16 };

int tempo = 150;

void playTone(int tone, int duration) {

for (long i = 0; i < duration * 1000L; i += tone * 2) {

  digitalWrite(speakerPin, HIGH);

  delayMicroseconds(tone);

  digitalWrite(speakerPin, LOW);

  delayMicroseconds(tone);

}

}
```

```
void playNote(char note, int duration) {

char names[] = {'C', 'D', 'E', 'F', 'G', 'A', 'B',

            'c', 'd', 'e', 'f', 'g', 'a', 'b',

            'x', 'y' };

int tones[] = { 1915, 1700, 1519, 1432, 1275, 1136, 1014,

            956,  834,  765,  593,  468,  346,  224,

            655 , 715 };

int SPEE = 5;

// play the tone corresponding to the note name

for (int i = 0; i < 17; i++) {

  if (names[i] == note) {
   int newduration = duration/SPEE;
    playTone(tones[i], newduration);

  }

}

}

void setup() {

pinMode(speakerPin, OUTPUT);

}

void loop() {

for (int i = 0; i < length; i++) {

  if (notes[i] == ' ') {

    delay(beats[i] * tempo); // rest
```

```
  } else {
    playNote(notes[i], beats[i] * tempo);
  }
  // pause between notes
  delay(tempo);
}
}
```

Applications

1. Melody piano
2. Song
3. Play tune
4. Pitch played