REPRODUCIR SONIDOS CON ARDUINO Y UN BUZZER PASIVO O ALTAVOZ

(Versión 25-11-19)

Fuente: https://www.luisllamas.es/reproducir-sonidos-arduino-buzzer-pasivo-altavoz/

¿QUÉ ES UN BUZZER O UN ALTAVOZ?

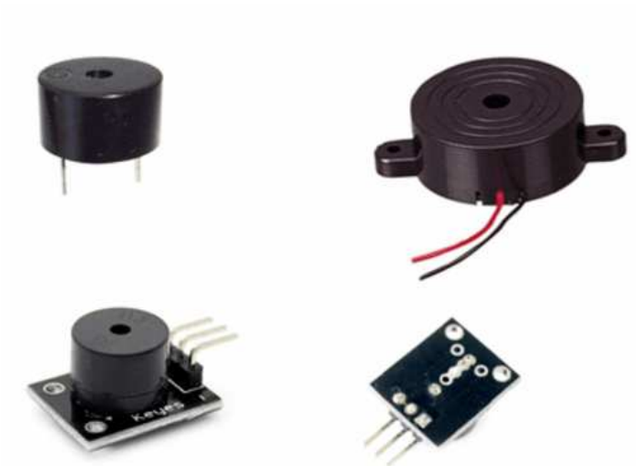
Un buzzer pasivo o un altavoz son dispositivos que permiten convertir una señal eléctrica en una onda de sonido. Estos dispositivos no disponen de electrónica interna, por lo que tenemos que proporcionar una señal eléctrica para conseguir el sonido deseado.

En oposición, los buzzer activos disponen de un oscilador interno, por lo que únicamente tenemos que alimentar el dispositivo para que se produzca el sonido.

Pese a tener la complejidad de proporcionar y controlar nosotros la señal eléctrica, los buzzer pasivos y de los altavoces tienen la ventaja de que podemos variar el tono emitido modificando la señal que aplicamos al altavoz, lo que nos permite generar melodías.

PRECIO

Podemos encontrar modelos de pequeños buzzer pasivo para usar en nuestros montajes y proyectos por muy poco dinero, desde 0,45€ en vendedores internacionales de eBay y AliExpress.



NOTA: en algunos casos diferenciamos el buzzer pasivo con respecto al activo, porque el pasivo muestra su plaquetita y el activo tiene como una brea.

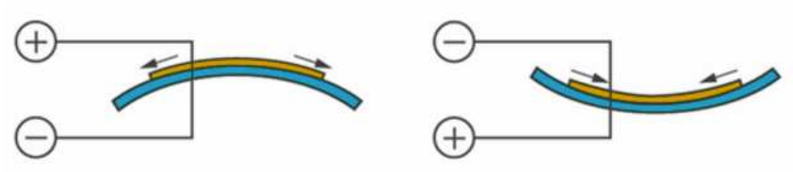
ACTIVO *– Funciona con un "1" o sea 5v* *O sea* ***digitalWrite(pin, HIGH)***	***PASIVO*** *– Es necesario generar la señal para hacerlo funcionar, es como un parlante* *Si bien se puede conectar a una salida de Arduino, se recomienda una R =220 ohms en serie.*
	

Frecuentemente, el buzzer pasivo se acompaña en una placa para facilitar su conexión, que incorpora un transistor y resitencias necesarias para hacer funcionar el buzzer pasivo o altavoz sin más que conectarlo.

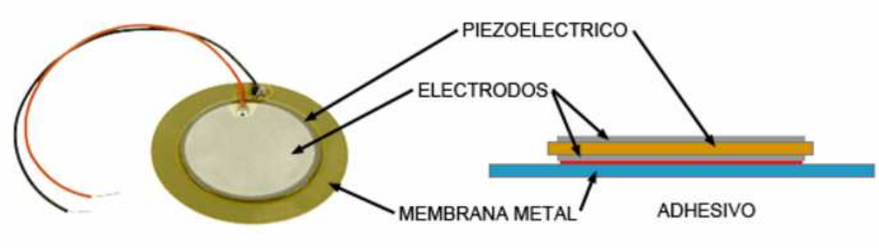
¿CÓMO FUNCIONA UN BUZZER Y UN ALTAVOZ?

Técnicamente tanto buzzers como altavoces son transductores electroacústicos, es decir, dispositivos que convierten señales eléctricas en sonido. La diferencia entre ambos es el fenómeno en el que basan su funcionamiento.

Los buzzer son transductores piezoeléctricos. Los materiales piezoeléctricos tiene la propiedad especial de variar su volumen al ser atravesados por corrientes eléctricas.

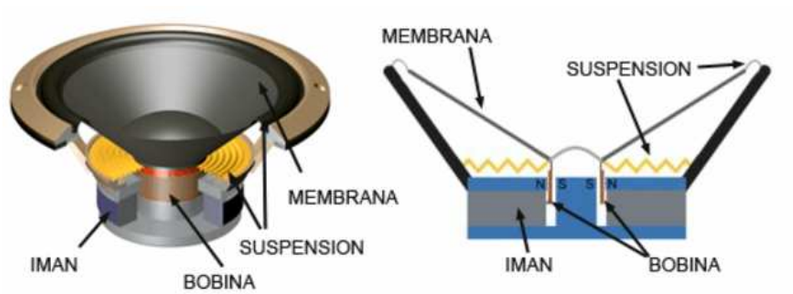


Un buzzer aprovecha este fenómeno para hacer vibrar una membrana al atravesar el material piezoeléctrico con una señal eléctrica.



Los buzzer son dispositivos pequeños y compactos, con alta durabilidad, y bajo consumo eléctrico. Por contra, la calidad de sonido es reducida.

Por su parte, un altavoz basa su funcionamiento en el magnetismo. Se dispone de un imán permanente que, normalmente, es fijo a la carcasa. Por otro lado, una bobina móvil se acopla a una membrana flexible.



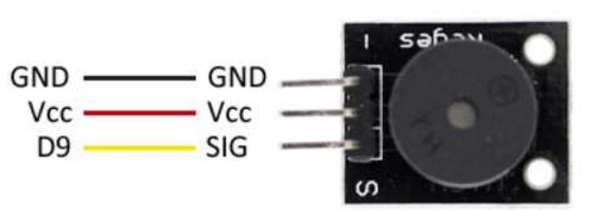
Al hacer circular una corriente por la bobina el campo magnético resultante genera una atracción con el imán, haciendo vibrar la membrana.

Los altavoces, por presentan una mejor calidad de sonido pero, en general, necesitan mayor potencia y es necesario disponer de dispositivos de amplificación para su uso.

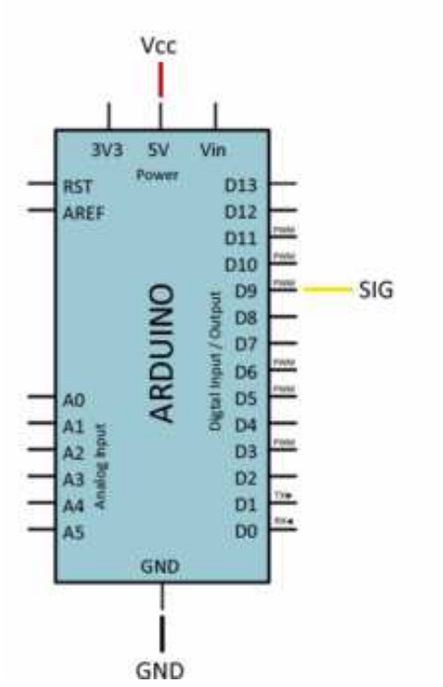
ESQUEMA DE MONTAJE

Si usamos una de las placas comerciales para pequeños proyectos y hobbies, que incorporan la electrónica y terminales necesarios, la conexión con Arduino es realmente sencilla. Simplemente alimentamos el módulo conectando Vcc y GND a Arduino, y la entrada de señal a cualquier salida digital de Arduino.

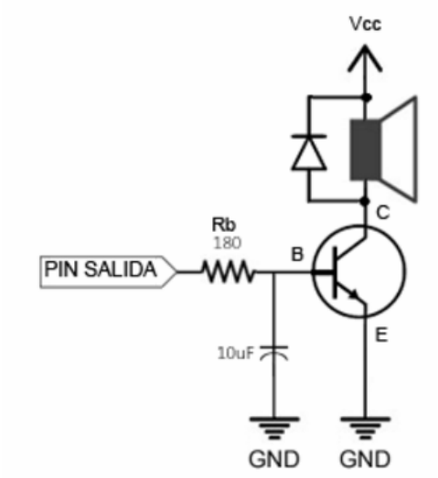
El esquema de conexión visto desde el componente sería el siguiente



Mientras que el esquema de conexión visto desde Arduino quedaría así



Si queremos usar un altavoz, que consumen mayor corriente de la que puede proporcionar Arduino, tendremos que proporcionar una etapa de amplificación, como vimos en la entrada de transistores BJT (TBJ).



NOTA: Debería calcula el valor de la R de base de acuerdo a lo visto en transistores

Aunque en general, en estos casos lo normal es que empleemos un amplificador específico diseñado para audio, en lugar de hacerlo nosotros mismos. Conseguiremos mejores niveles de calidad e incluso un menor coste.

EJEMPLOS DE CÓDIGO

Arduino dispone de dos funciones que nos permiten generar fácilmente señales eléctricas para convertir en sonido, usando cualquiera de las salidas digitales disponibles.

Estas funciones son tone() y noTone() y, como su nombre indican, permiten generar o detener la señal del tono en un pin.

```
tone(pin, frecuencia); //activa un tono de frecuencia determinada en un pin dado
noTone(pin);           //detiene el tono en el pin
```

La función tone() también permite especificar la duración del sonido generado.

```
tone(pin, frequencia, duracion);  //activa un tono de frecuencia y duracion determinados en un pin dado
```

Pese a su sencillez, al usar las funciones para la generación de tone tenemos que asumir importantes limitaciones.

- **La función Tone emplea el Timer 2, por lo que mientras este funcionando no podremos usar las salidas PWM en los pines 3 y 11 en Arduino Nano y Uno (pines 9 y 10 en Arduino Mega).**
- **No podemos usar la función tone() en dos pines de forma simultánea. Deberemos apagar el tono con la funcion noTone() antes de poder usarlo en otro pin.**
- **Los rangos de la función tone son de 31 Hz a 65535 Hz.**

El siguiente código muestra el uso de estas funciones estas funciones en un ejemplo simple, en el que empleamos el buzzer o altavoz conectado en el Pin3 para generar una función de 440Hz durante un segundo, pararlo durante 500ms, y finalmente un tono de 523Hz durante 300ms, para repetir el programa tras una pausa de 500ms.

```
const int pinBuzzer = 3;

void setup()
{
}

void loop()
{
  //generar tono de 440Hz durante 1000 ms
  tone(pinBuzzer, 440);
  delay(1000);

  //detener tono durante 500ms
  noTone(pinBuzzer);
  delay(500);

  //generar tono de 523Hz durante 500ms, y detenerlo durante 500ms.
  tone(pinBuzzer, 523, 300);
  delay(500);
}
```

El siguiente, también muy básico, emplea un array con frecuencias que recorremos secuencialmente para realizar un barrido que aproxima las distintas notas musicales.

```
const int pinBuzzer = 3;

const int tonos[] = {261, 277, 294, 311, 330, 349, 370, 392, 415, 440, 466, 494};
const int countTonos = 10;

void setup()
{
}

void loop()
{
  for (int iTono = 0; iTono < countTonos; iTono++)
  {
   tone(pinBuzzer, tonos[iTono]);
   delay(1000);
  }
  noTone(pinBuzzer);
}
```