

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Periféricos y Dispositivos de Interfaz Humana

Grado en Ingeniería Informática

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Practica 2 : Experimentación con un sistema de microcontrolador : Arduino . Reproductor de sonido



Félix Ramírez García

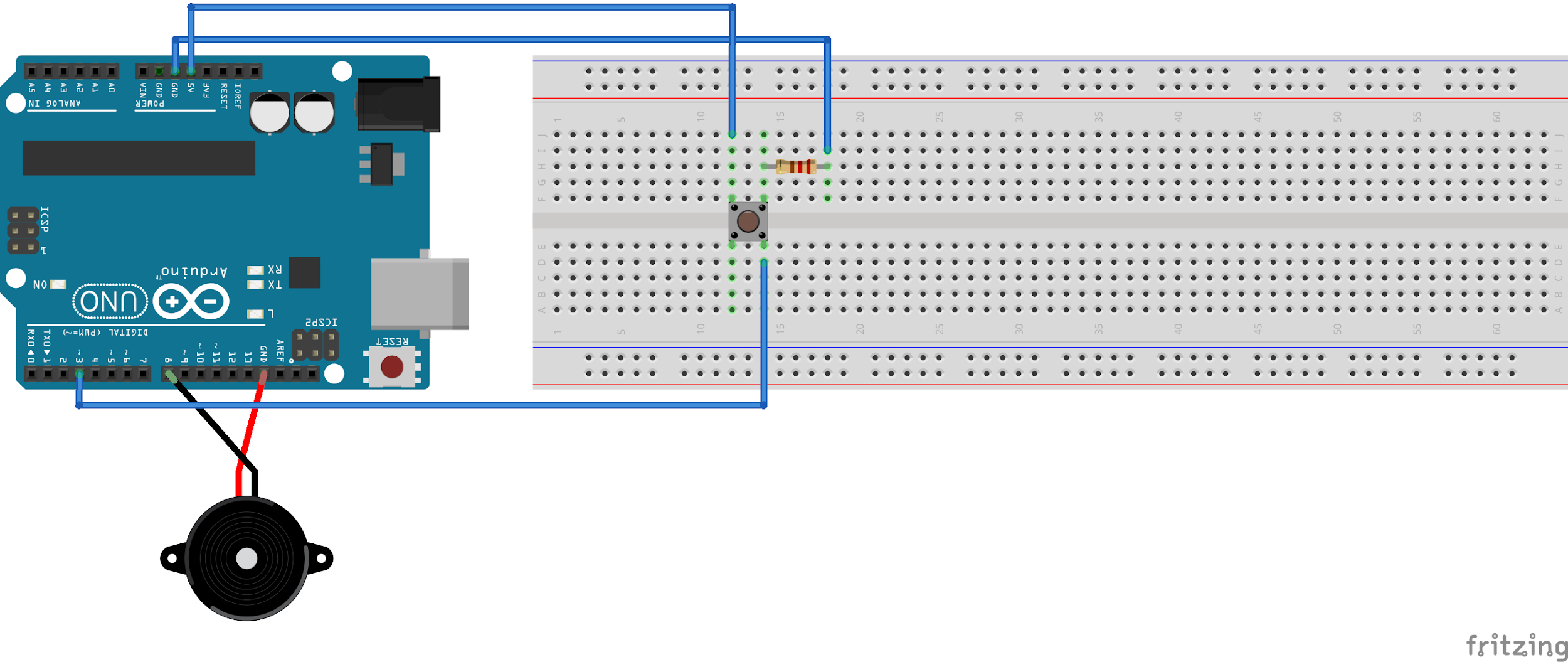
José Rubén Céspedes Heredia

Generar sonido a través del altavoz piezo-eléctrico (buzzer) incluido en el lab-kit de Cooking Hacks. Además, se controla el volumen del altavoz a través de un potenciómetro y usamos la función de "play-pause" a través de un botón.

El buzzer OBO-27225 no está polarizado, por lo que es indiferente el pin que se conecte a tierra.

El sketchde la batería de ejemplos de Arduino “toneMelody” nos sirve de referencia para implementar este proyecto. Incluimos la cabecera "pitches.h", este archivo contiene todos los valores de las frecuencias de las notas musicales en formato anglosajón. Por ejemplo, NOTE\_C4 es una C media (do de la 4ª escala). NOTE\_FS3 es F aguda (fa sostenido de la 3ª escala), y así sucesivamente. Un valor ‘0’ significa un silencio.

**1.Imagen con el esquema de conexiones realizado en Fritzing.**

****

**2.Código fuente documentado. Describimos los pines de entrada y salida que se usan y su significado.**

**[code]  
#include "pitches.h"  
  
// Notas para la melodia  
int melody[] = {  
 NOTE\_E7, NOTE\_E7, 0, NOTE\_E7,   
 0, NOTE\_C7, NOTE\_E7, 0,  
 NOTE\_G7, 0, 0, 0,  
 NOTE\_G6, 0, 0, 0,   
  
 NOTE\_C7, 0, 0, NOTE\_G6,   
 0, 0, NOTE\_E6, 0,   
 0, NOTE\_A6, 0, NOTE\_B6,   
 0, NOTE\_AS6, NOTE\_A6, 0,   
  
 NOTE\_G6, NOTE\_E7, NOTE\_G7,   
 NOTE\_A7, 0, NOTE\_F7, NOTE\_G7,   
 0, NOTE\_E7, 0,NOTE\_C7,   
 NOTE\_D7, NOTE\_B6, 0, 0,  
  
 NOTE\_C7, 0, 0, NOTE\_G6,   
 0, 0, NOTE\_E6, 0,   
 0, NOTE\_A6, 0, NOTE\_B6,   
 0, NOTE\_AS6, NOTE\_A6, 0,   
  
 NOTE\_G6, NOTE\_E7, NOTE\_G7,   
 NOTE\_A7, 0, NOTE\_F7, NOTE\_G7,   
 0, NOTE\_E7, 0,NOTE\_C7,   
 NOTE\_D7, NOTE\_B6, 0, 0  
};  
  
// Duracion de las notas: 4 = quarter note, 8 = eighth note, etc.:  
int noteDurations[] = {  
 12, 12, 12, 12,   
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,   
  
 12, 12, 12, 12,  
 12, 12, 12, 12,   
 12, 12, 12, 12,   
 12, 12, 12, 12,   
  
 9, 9, 9,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
  
 9, 9, 9,  
 12, 12, 12, 12,  
 12, 12, 12, 12,  
 12, 12, 12, 12  
};  
  
// Tamaño de los vectores anteriores  
int tam\_vectores = 78;  
// Indica si se debe reproducir la cancion: LOw -> No se reproduce; HIGH -> Se reproduce  
volatile byte play = LOW;  
// Pin del boton  
int pinButton = 3;  
// Almacena la ultima nota reproducida, para volver a reproducir por dicha nota  
int ultima\_nota = 0;  
  
void setup() {  
 // Preparamos la interrupcion para el boton  
 pinMode(pinButton, INPUT\_PULLUP);   
 attachInterrupt(digitalPinToInterrupt(pinButton), changePlay, RISING);  
}  
  
void loop() {  
 Serial.begin(9600);  
  
 // Almacena la nota actual  
 int thisNote;  
   
 // iterate over the notes of the melody:  
 for (thisNote = ultima\_nota; thisNote < tam\_vectores && play; thisNote++) {  
 // to calculate the note duration, take one second divided by the note type.  
 //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.  
 int noteDuration = 1000 / noteDurations[thisNote];  
 tone(8, melody[thisNote], noteDuration);  
   
 // to distinguish the notes, set a minimum time between them.  
 // the note's duration + 30% seems to work well:  
 int pauseBetweenNotes = noteDuration \* 1.30;  
 delay(pauseBetweenNotes);  
 // stop the tone playing:  
 noTone(8);  
 }  
  
 // Dos opciones: o se ha terminado la cancion (play = HIGH) o hemos pulsado pause  
 if(play){  
 // Si se ha terminado la cancion, volvemos al principio de ella  
 ultima\_nota = 0;  
 } else {  
 // Si le hemos dado a pause, almacenamos la ultima nota reproducida para volver a empezar por ella  
 ultima\_nota = thisNote;  
 }  
}  
  
void changePlay(){  
 play = !play;  
  
 if(play){  
 Serial.println("Play");  
 } else {  
 Serial.println("Pause");  
 }  
}**

**3.Fotografías y/o vídeos demostrando el funcionamiento real del proyecto.**

En el siguiente link se muestran el funcionamiento del proyecto :

<https://drive.google.com/open?id=19PeQ5S98q01zeCAK29vB6uWOzGP7xANl>