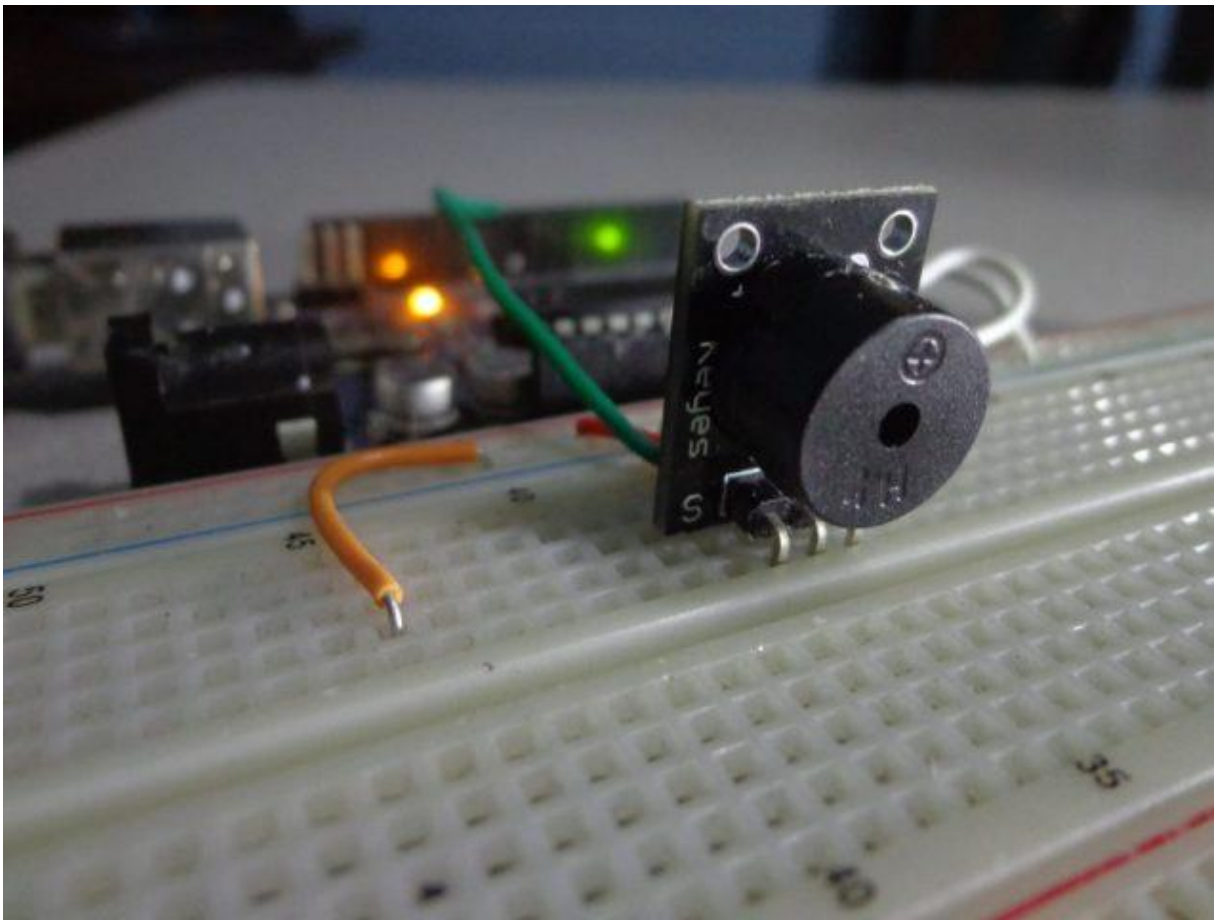
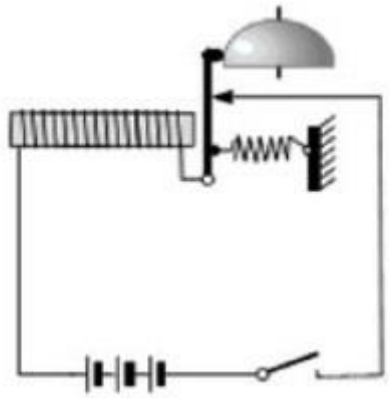
BUZZER AVEC ARDUINO



1. Intro

Le mot «buzzer» vient de l'allemand «summen» - buzz. Il s'agit essentiellement d'un dispositif émettant du son, traditionnellement utilisé comme dispositif de signalisation. A ce jour, les buzzers sont électromécaniques et piézoélectriques. Cela et d'autres trouvent une application dans différents appareils.

Historiquement, le premier buzzer électromécanique est apparu, qui est un relais électromécanique à contacts normalement fermés à travers lequel la bobine de ce relais est connectée à une source de courant.
Le principe de fonctionnement du buzzer est tout simplement impossible. Lorsqu'un courant circule dans le circuit de travail du buzzer, la bobine de relais est excitée, ce qui signifie que le flux magnétique dans son noyau augmente, sous l'influence duquel les contacts à travers lesquels la bobine elle-même vient d'être alimentée s'ouvrent immédiatement.
Lorsque les contacts sont ouverts, l'enroulement du relais cesse de recevoir de l'énergie, le flux magnétique dans le noyau disparaît, ce qui signifie que le contact mobile qui vient de fermer le circuit d'alimentation du relais est libéré, et le ressort met le circuit dans son état initialement fermé.
Et maintenant, les contacts sont à nouveau fermés, la bobine reçoit à nouveau de l'énergie et le noyau attire à nouveau le contact de relais mobile, coupant à nouveau son propre circuit d'alimentation. Donc, le processus se répète encore et encore. Les vibrations du bras de relais font un bourdonnement. La bobine Rumkorff fonctionne de la même manière.



Bien entendu, le buzzer à relais au cours de son fonctionnement génère non seulement un fort bruit impulsionnel dans le circuit de puissance, mais émet également de fortes interférences dans l'air radio, par conséquent, le buzzer est utilisé, entre autres, pour tester l'immunité au bruit de divers équipements.
Le principal inconvénient du buzzer électromécanique est évident: la présence d'un élément mobile use le mécanisme, et le ressort s'affaiblit avec le temps, et donc, le temps de défaillance du buzzer ne dépasse pas 5000 heures.
Cependant, l'attention est attirée sur la première utilisation du buzzer, inventé par Johann Wagner en 1839, et développé par John Mirand, qui a ajouté une cloche au marteau vibrant. Il s'est avéré une cloche électrique, produisant un son en frappant une cloche avec un marteau. Le marteau cloche était connecté à l'armature du relais, qui fonctionnait directement en mode buzzer.

Les premières sonnettes électriques installées dans tous les appartements étaient à peu près les mêmes (voir - Dispositif de cloche électrique) Les cloches d'alarme incendie ont un dispositif similaire à celui des premières cloches suspendues dans les gares.
Une version plus moderne du buzzer est un émetteur de son piézoélectrique, lié aux appareils électro-acoustiques, et produisant un son audible ou des ultrasons en utilisant l'effet piézoélectrique inverse.

Un piézoélectrique est déposé ici sur une fine plaque métallique. De l'autre côté de la couche piézoélectrique, il y a une pulvérisation conductrice. La pulvérisation et la plaque elle-même sont les contacts auxquels l'alimentation est fournie. Pour améliorer l'amplitude de propagation des vibrations sonores, un petit embout buccal avec un trou peut être fixé à la plaque.

Le buzzer piézoélectrique est alimenté par un courant alternatif à une tension de 3 à 10 volts, et la fréquence du courant détermine la fréquence du son. La fréquence de résonance caractéristique des émetteurs de sons piézoélectriques se situe dans la plage de 1 à 4 kHz, ce qui conduit à un bourdonnement facilement reconnaissable avec une pression acoustique atteignant 75 dB à une distance de 1 mètre de l'émetteur. Ces buzzers peuvent fonctionner comme des microphones ou des capteurs.
Les avertisseurs piézoélectriques sont utilisés dans les réveils, les jouets, les appareils électroménagers et les téléphones.Les ultrasons obtenus avec leur aide sont souvent utilisés dans les répulsifs contre les rongeurs, dans les humidificateurs, dans le nettoyage par ultrasons, etc.

Il est possible d’émettre des sons à l’aide d’un microcontrôleur en branchant un buzzer sur une de ses sorties. Lorsqu’on veut créer une interface utilisateur, il est agréable d’avoir un retour selon les actions effectuées que ce soit un affichage, une lumière qui s’allume, ou change de couleur, ou encore un son. Nous allons voir dans ce tutoriel comment utilisé un buzzer (ou haut-parleur piézoélectrique).

2. Matériel

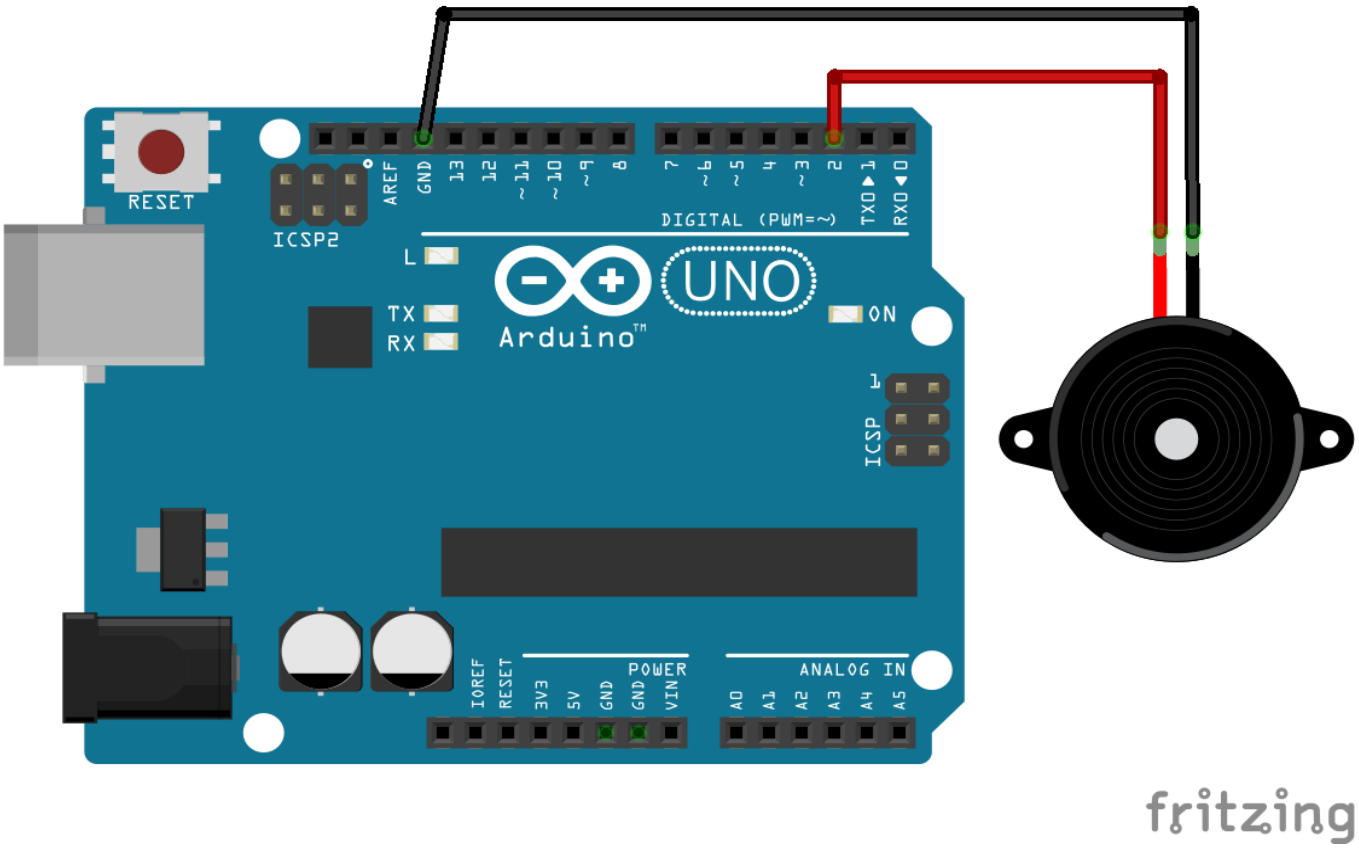
- Ordinateur
- Arduino UNO
- Câble USB A Mâle/B Mâle
- Buzzer

Principe de fonctionnement

Un buzzer est une sorte de haut-parleur mais de faible puissance qui va émettre un son en fonction de la fréquence et amplitude de vibration. Il permet de jouer des notes et de recréer des mélodies simples. Si vous souhaitez jouer des sons comme de la musique ou des voix, il vous faudra utiliser un haut-parleur avec un amplificateur audio qui viendra jouer un fichier audio stocké sur une carte SD.
Pour jouer des fichiers audio, vous pouvez suivre ce tutoriel.

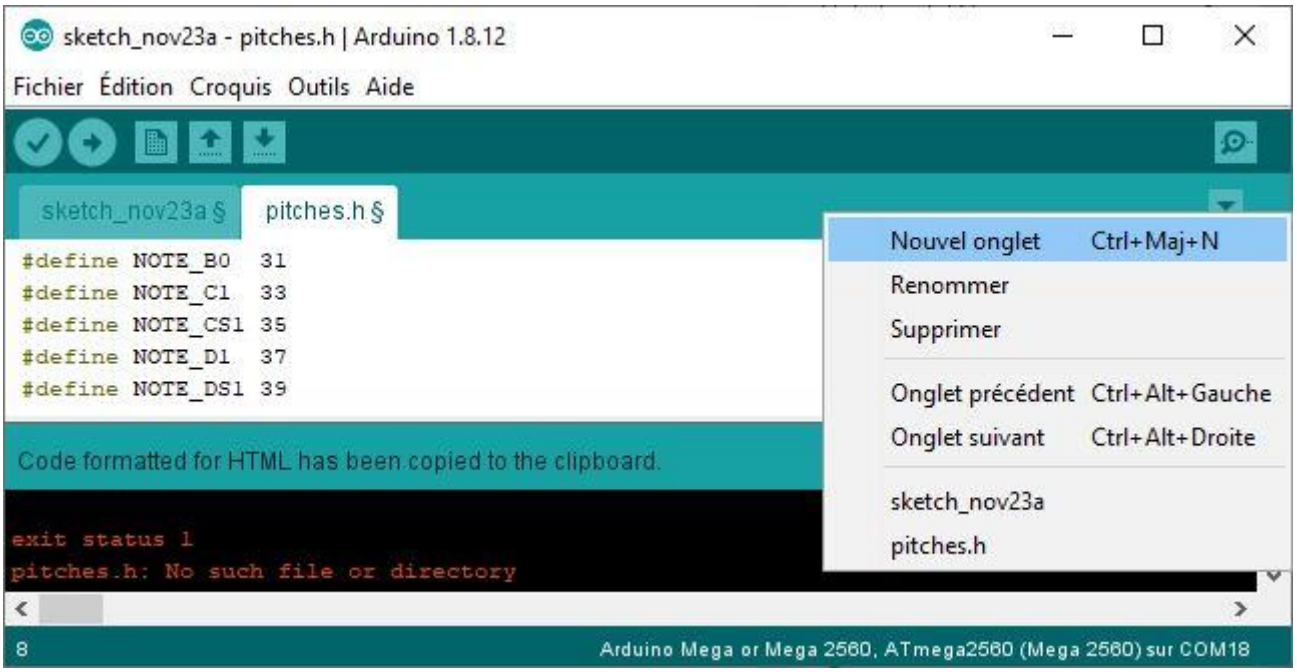
3. Schéma

Un buzzer étant de faible puissance, il peut être branché directement sur le microcontrôleur sur n'importe laquelle de ses broches de sorties. Dans ce tutoriel, nous branchons la borne – du buzzer au GND et la borne + à la sortie digitale 2.



4. Code

Pour chaque note correspond un fréquence qui est enregistrée dans le fichier pitches.h. Pour jouer une mélodie, nous allons placer les notes dans un tableau et spécifier un retard entre chaque notes. Nous utilisons ensuite la fonction tone pour appliquer cette fréquence et cette durée à la sortie du buzzer.
Pour créer le fichier pitches, cliquez sur la flèche à droite du nom du sketch, puis cliquez sur « créer un nouvel onglet ». Copier-collez ensuite le code avec la définition des notes.



Fichier .ino

```
//Libraries
#include "pitches.h" //https://www.arduino.cc/en/Tutorial/toneMelody

//Parameters
const int buzPin  = 2;

void setup() {
  //Init Serial USB
  Serial.begin(9600);
  Serial.println(F("Initialize System"));
}

void loop() {
  playPassed();
  delay(1000);
  playFailed();
  delay(2000);
}

void playPassed() { /* function playPassed */
  ////Play 'ON' Sound
  int melodyOn[] = {NOTE_C5, NOTE_C6, NOTE_D5, NOTE_A6};
  int durationOn = 200;
```

```
  for (int thisNote = 0; thisNote < 4; thisNote++) {
    tone(buzPin, melodyOn[thisNote], durationOn);
    delay(200);
  }
}

void playFailed() { /* function playFailed */
  ////Play 'OFF' Sound
  int melodyOff[] = {NOTE_C3, NOTE_D3};
  int durationOff = 200;
  for (int thisNote = 0; thisNote < 2; thisNote++) {
    tone(buzPin, melodyOff[thisNote], durationOff);
    delay(200);
  }
}
```

Fichier pitches.h

```
/*************************************************

 * Public Constants

 *************************************************/
#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
```

```
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
#define NOTE_A2  110
#define NOTE_AS2 117
#define NOTE_B2  123
#define NOTE_C3  131
#define NOTE_CS3 139
#define NOTE_D3  147
#define NOTE_DS3 156
#define NOTE_E3  165
#define NOTE_F3  175
#define NOTE_FS3 185
#define NOTE_G3  196
#define NOTE_GS3 208
#define NOTE_A3  220
#define NOTE_AS3 233
#define NOTE_B3  247
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440
#define NOTE_AS4 466
#define NOTE_B4  494
#define NOTE_C5  523
#define NOTE_CS5 554
#define NOTE_D5  587
#define NOTE_DS5 622
#define NOTE_E5  659
#define NOTE_F5  698
#define NOTE_FS5 740
#define NOTE_G5  784
```

```
#define NOTE_GS5 831
#define NOTE_A5  880
#define NOTE_AS5 932
#define NOTE_B5  988
#define NOTE_C6  1047
#define NOTE_CS6 1109
#define NOTE_D6  1175
#define NOTE_DS6 1245
#define NOTE_E6  1319
#define NOTE_F6  1397
#define NOTE_FS6 1480
#define NOTE_G6  1568
#define NOTE_GS6 1661
#define NOTE_A6  1760
#define NOTE_AS6 1865
#define NOTE_B6  1976
#define NOTE_C7  2093
#define NOTE_CS7 2217
#define NOTE_D7  2349
#define NOTE_DS7 2489
#define NOTE_E7  2637
#define NOTE_F7  2794
#define NOTE_FS7 2960
#define NOTE_G7  3136
#define NOTE_GS7 3322
#define NOTE_A7  3520
#define NOTE_AS7 3729
#define NOTE_B7  3951
#define NOTE_C8  4186
#define NOTE_CS8 4435
#define NOTE_D8  4699
#define NOTE_DS8 4978
```

Résultat

Modifier les notes, la taille des tableaux et les délais pour recréer les mélodies que vous voulez.

Applications

- Créer un clavier interactif avec retour sonore
- Créer un robot qui émet des sons comme R2-D2

Code 1 - Allumer le Buzzer

Ce programme est très simple. Il est basé sur le même principe que le Blink de la LED. Il s'agit d'éteindre et d'allumer le buzzer en faisant passer du courant pendant un temps déterminé.

```
int buzzer = 3;// le buzzer est connecté à la broche 3
void setup() {
  pinMode(buzzer,OUTPUT);
  Serial.begin(9600);  // ouvre le port série à 9600 bauds

}

void loop() {
  digitalWrite(buzzer, HIGH);// le courant passe
  delay(500);
  digitalWrite(buzzer, LOW);// le courant ne passe pas
  delay(500);
}
```

Exercice

Modifiez les temps (delay) afin de constater les changements. Le temps entre les changements d'état HIGH et l'état LOW n'ont pas besoin d'être identiques.

Code 2 – la tonalité

Ce programme utilise l'instruction tone qui permet de paramétrer la tonalité du buzzer. Celui-ci joue un « son » en fonction d'un niveau donné de fréquence. Il ne s'agit plus d'une simple impulsion électromagnétique.

```
int buzzer = 3; // le buzzer est connecté à la broche 3
void setup() {
  pinMode(buzzer,OUTPUT);
  Serial.begin(9600);  // ouvre le port série à 9600 bauds
  }

void loop() {
  tone (buzzer, 500);// définit le signal du son à 500 Hz
  delay(1000);
  noTone(buzzer);// le son est coupé
  delay(1000);// pendant 1 seconde
}
```

Syntaxe

tone(pin ; fréquence);

noTone(pin);

Exercice

Modifiez la valeur de la tonalité. Une valeur basse donne une tonalité grave, et une valeur élevée, une tonalité aiguë.

Les sons audibles s'étendent de 20Hz à 20 000 Hz. Mais ce type de buzzer ne permet de telles fréquences audibles.

Une fiche synthétique est disponible sur mon Padlet.

#defiArduino1

Sur le principe du Code 1, allumez et éteignez le buzzer en modifiant à chaque fois la valeur de 500. Faites un code simple, long mais simple.

Les valeurs à tester sont : 100 – 500 – 1500 – 2000 – 2500 – 3000 – 3500 – 4000 - 4500 - 5000

Solution

#defiArduino2

Sur le modèle du Code 1 présenté dans l'article sur l'intensité d'une LED, augmentez progressivement la fréquence du buzzer jusqu'à une valeur de 80.

Solution