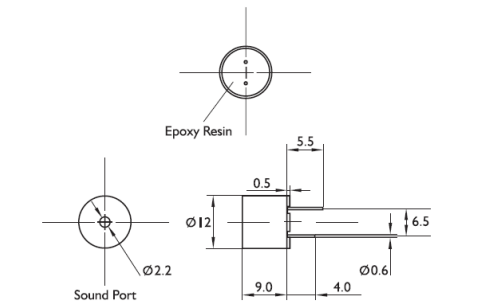
**Résumé Buzzer**

**RS 7800712**

1. La tension d’alimentation est de 3.3 à 8V. **On utilise 5V sur un driver MOS + Charge.**

**-> Courant d’alimentation < 40 mA et Résistance interne de 47 ± 5 ohms**

1. Aucune communication requise.
2. Code + Circuit



**Code général « Buzzer.ino » (fonctions principales + boucle de test)**

**Code de temps d'exécution « Buzzer-temps-exec.ino » (programme de temps d'exécution)**

**Il est possible de modifier la mélodie jouée par le buzzer lors d’une chute.**

1. **Pins**

| 1. **+POWER (pâte longue)** | **+5V** |
| --- | --- |
| 1. **GND (pâte courte)** | **GND** |

**Librairies à utiliser**

| **Tone32.h** | **pitches.h** |
| --- | --- |

**Initialisation des paramètres**

**#include “pitches.h” //Librairie des notes jouables par le buzzer  
#include “Tone32.h” //Librairie des fonctions tone pour l'ESP32  
#define buzzer 27 //Pin du driver de buzzer  
#define chan\_PWM 9 //Canal PWM de la pin "buzzer"  
#define note NOTE\_C4 //Note élémentaire du buzzer (plus de valeurs dans “pitches.h”)  
  
int son [] = { note, 2\*note, note, 2\*note } ; //Mélodie à jouer par le buzzer, basée sur la note originale  
int duree [] = { 250, 250, 250, 250 } ; //Durée de lecture de chaque note (< ou = à 250)  
int nb\_notes = 4 ; //Nombre de notes dans la mélodie**

**Initialisation LED**

**#define led\_rouge 33 //Pin de la LED rouge**

**Fonctions à utiliser**

| **Nom fonction** | **Rôle** | **Pire temps d’exécution** | **Meilleur temps d’exécution** |
| --- | --- | --- | --- |
| void Buzzer\_Setup() | Setup du buzzer | **0,00 ms** | **0,00 ms** |
| void Buzzer\_Chute() | Joue un son prédéterminé en cas de chute du robot | **3000,00 ms** | **3000,00 ms** |

**Fonction Setup - Fonction à utiliser – Fonction interne/peu probable à utiliser**