Utiliser un buzzer passif (KY-006)

Dans cet article, nous passons au **buzzer passif KY-006**, beaucoup plus flexible que le buzzer actif. Il permet de **contrôler la hauteur du son**, le volume et les variations dans le temps. C'est avec lui que l'on peut produire des **alertes évoluées** et, progressivement, de **véritables mélodies**.

4GMS 5GMS 4TTR Découverte

Principe du buzzer passif

Un buzzer passif **ne produit aucun son tout seul**. Il doit être piloté par un **signal PWM**.

Avec un buzzer passif :

- la **fréquence** du PWM détermine la hauteur du son
- le **duty cycle** influence le volume
- la **durée** définit le rythme

On contrôle donc **trois paramètres**, contre un seul pour le buzzer actif.

Connexion du KY-006

- `S` (Signal) → GPIO15 (PWM)
- `-` (GND) → GND
- `+` → non connecté (c'est le `S` qui sert à générer le signal dans notre cas)

Initialisation de base

```
from machine import Pin, PWM
import utime
```

```python

buzzer = PWM(Pin(15))
buzzer.duty_u16(2000)
```

À ce stade, le buzzer est prêt, mais **silencieux** tant qu'aucune fréquence n'est définie.

Alerte simple sur deux tons (ambulance)

On alterne deux fréquences distinctes.

```python
while True:
    buzzer.freq(600)
    utime.sleep(0.5)

    buzzer.freq(900)
    utime.sleep(0.5)
```

Alerte avec montée progressive de fréquence

```python
while True:
    for freq in range(400, 1000, 20):
        buzzer.freq(freq)
        utime.sleep(0.02)
```

> La fréquence augmente progressivement, ce qui donne une sensation de montée de tension.

Alerte avec descente progressive de fréquence

```python
while True:
    for freq in range(1000, 400, -20):
```

```
        buzzer.freq(freq)
        utime.sleep(0.02)
```

Montée puis descente de fréquence

```
while True:
    for freq in range(400, 1000, 20):
        buzzer.freq(freq)
        utime.sleep(0.02)

    for freq in range(1000, 400, -20):
        buzzer.freq(freq)
        utime.sleep(0.02)
```

Alerte à deux tons avec durée qui diminue

La durée de chaque note diminue de **25 % à chaque itération**.

```
duration = 2.0

while duration > 0.3:
    buzzer.freq(600)
    utime.sleep(duration)

    buzzer.freq(900)
    utime.sleep(duration)

    duration = duration * 0.75
```

Diminuer aussi la pause

```
duration = 2.0
pause = 1.0

while duration > 0.3:
```


© 2025 Mr. Lambrechts https://v2.ttrinfo.be 3 | 7

```
    buzzer.freq(600)
    utime.sleep(duration)
    utime.sleep(pause)

    buzzer.freq(900)
    utime.sleep(duration)
    utime.sleep(pause)

    duration = duration * 0.75
    pause = pause * 0.75
```

Introduction aux notes de musique

On peut associer des noms de notes à des fréquences.

```
notes = {
    'C': 262,
    'D': 294,
    'E': 330,
    'F': 349,
    'G': 392,
    'A': 440,
    'B': 494,
    'C5': 523,
    'D5': 587,
    'E5': 659,
    'F5': 698,
    'G5': 784,
    'A5': 880
}
```

Mélodie simple, rythme monotone

```
melody = ['C', 'D', 'E', 'F', 'G', 'A', 'G', 'F']

for note in melody:
    buzzer.freq(notes[note])
    utime.sleep(0.4)
```


© 2025 Mr. Lambrechts https://v2.ttrinfo.be 4 | 7

Introduire la durée avec des tuples

Chaque élément contient :

- la note
- la durée

```python
melody = [
    ('C', 0.4),
    ('D', 0.4),
    ('E', 0.6),
    ('C', 0.6)
]
```

Lecture :

```python
for note, duration in melody:
    buzzer.freq(notes[note])
    utime.sleep(duration)
```

Frère Jacques (8 premières notes)

```python
melody = [
    ('C', 0.4), ('D', 0.4), ('E', 0.4), ('C', 0.4),
    ('C', 0.4), ('D', 0.4), ('E', 0.4), ('C', 0.4)
]

for note, duration in melody:
    buzzer.freq(notes[note])
    utime.sleep(duration)
```

Joyeux anniversaire (6 premières notes)

```python
melody = [
    ('G', 0.4), ('G', 0.2),
    ('A', 0.6),
    ('G', 0.6),
    ('C5', 0.6),
    ('B', 1.0)
```


© 2025 Mr. Lambrechts

https://v2.ttrinfo.be

5 | 7

```
]

for note, duration in melody:
    buzzer.freq(notes[note])
    utime.sleep(duration)
```

Le finale

```
melody = [
    ('A4', 0.3), ('B4', 0.3), ('D5', 0.4), ('B4', 0.4),
    ('F5', 0.6), ('F5', 0.6),
    ('E5', 0.8)
]

notes.update({
    'A4': 440,
    'B4': 494,
    'D5': 587,
    'E5': 659,
    'F5': 698
})

for note, duration in melody:
    buzzer.freq(notes[note])
    utime.sleep(duration)
```

Erreurs fréquentes

- oublier d'utiliser le PWM
- confondre fréquence et durée
- jouer trop fort dès le début
- croire qu'une mélodie est autre chose qu'une suite de fréquences et de temps

Ce qu’il faut retenir

- le buzzer passif est entièrement piloté par le code
- fréquence = hauteur du son
- durée = rythme
- les mélodies sont des données, pas de la magie
- une structure simple permet des résultats complexes

Avec ces trois articles, les élèves disposent désormais :

- d’une vision claire des buzzers
- d’une progression logique
- d’un terrain idéal pour introduire abstraction, fonctions… et surprises sonores ultérieures

© 2025 Mr. Lambrechts https://v2.ttrinfo.be 7 | 7