Étape 7 : De la lumière au son

Objectifs:

- ❑ Générer un son d'une certaine fréquence
- ❑ Réaliser un capteur de voiture avec du son !

Générer un son d'une certaine fréquence

Figure 2

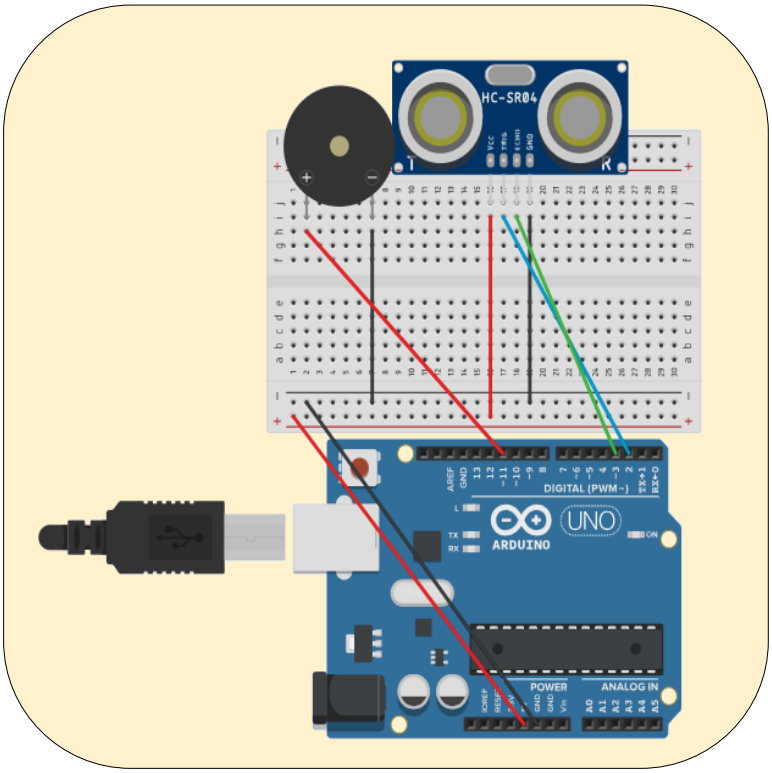

Figure 3

Le buzzer du kit Arduino ressemble à celui de la Figure 2. Pour utiliser le buzzer correctement, on va connecter la patte (-) à la masse, et la patte (+) à la patte de la carte qui commandera le buzzer (11 sur Figure 3).

Il y a plusieurs fonctions importantes à retenir pour l'utilisation du buzzer.

D'abord, dans le void setup, déclare le port (PIN) utilisé et on définit le mode (OUTPUT) :

```
pinMode(PIN,OUTPUT);
```

Puis, pour générer un son :

```
tone(PIN,f);
```

Où `PIN` est la patte utilisée (ici 11) et `f` la fréquence du son généré.

Enfin, pour que le buzzer n'émette plus de son :

```
noTone(PIN)
```

? Exercice pratique : Compléter le code pour générer un son à une fréquence de 440 Hz pendant 500ms peu importe le résultat de la distance.

```
1.  #define port_trigger 2   //on déclare le port de l'émetteur
2.  #define port_echo 3      //et du capteur
3.  float vitesse_son = 343.0 / 10000;
4.  void setup() {
5.    pinMode(11, OUTPUT);    //on declare port et mode du buzzer
6.    pinMode(port_trigger, OUTPUT);  //l'émetteur est une sortie
7.    digitalWrite(port_trigger,LOW);   qui n'émet rien au début
8.    pinMode(port_echo,INPUT);        //le capteur est une entrée
9.    Serial.begin(9600);
10. }
11. void loop() {
12.   digitalWrite(port_trigger,HIGH);  //on allume l'émetteur
13.   delayMicroseconds(10);            //pendant 10 microsecondes
14.   digitalWrite(port_trigger,LOW);   //puis on l'éteint
15.   long measure = pulseIn(port_echo,HIGH); //on récupère le temps
16.   float distance_cm = measure / 2.0 * vitesse_son;
17.   Serial.println(distance_cm);      //on affiche la distance
18.
19.   tone(11,……);                  //on émet un son
20.   delay(……);                    //on patiente
21.   noTone(11);                    //on arrête
22.   delay(500);                    //on patiente 500ms
23.  }
```

Réaliser un capteur de voiture avec du son !

Passons maintenant au capteur de voiture avec avertisseur sonore, comme sur les vraies voitures !

Avec ce que vous venez d'apprendre sur le son, on adapte le code de la LED pour le buzzer

Le début du code ne change pas :

```
1. #define port_trigger 2
2. #define port_echo 3
3. /* Vitesse du son en cm/us */
4. float vitesse_son = 343.0 / 10000;
5.
6. void setup() {
7.   pinMode(11, OUTPUT);
8.   pinMode(port_trigger, OUTPUT);
9.   digitalWrite(port_trigger,LOW);
10.  pinMode(port_echo,INPUT);
11.  Serial.begin(9600);
12. }
13.
14. void loop() {
15.  digitalWrite(port_trigger,HIGH);
16.  delayMicroseconds(10);
17.  digitalWrite(port_trigger,LOW);
18.  long measure = pulseIn(port_echo,HIGH);
19.  float distance_cm = measure / 2.0 * vitesse_son;
20.  Serial.println(distance_cm);
```

Suite et fin du code : Les choses changent

```
21.   if (... > 50) {
22.      noTone(11);
23.      delay(500);
24.    }
25.
26.    else if (distance_cm<4) {
27.      tone(11,440);
28.      delay(400);
29.    }
30.    else {
31.      tone(11,440);     //le délai est proportionnel
32.      delay(...*20);      à la distance
33.      noTone(11);
34.      delay(...*20);
35.      tone(11,440);
36.      delay(...*20);
37.      noTone(11);
38.      delay(distance_cm*20);
39.    }
40. }
```

? Exercice pratique : Compléter le code. Faire appel à un encadrant. Admirer votre capteur de voiture fonctionnel.

Étape 8 : Bonus

? Exercice pratique : En pratique, en plus de sonner de plus en plus rapidement, le son émis est de plus en plus aigu lorsque la distance est faible. Complétez le code suivant (la première partie reste identique) pour que le son soit plus aigu à mesure que la distance diminue. On commencera par à un son à 400 Hz lorsque la distance vaut 50cm et on finira par un son à 5000 Hz lorsque la distance est inférieure à 4 cm.

```
21.   if (distance_cm > 50) {
22.      noTone(11);
23.      delay(500);
24.    }
25.
26.    else if (distance_cm<4) {
27.      tone(11,……);
28.      delay(400);
29.    }
30.    else {
31.      tone(11,……);
32.      delay(distance_cm*20);
33.      noTone(11);
34.      delay(distance_cm*20);
35.      tone(11,……);
36.      delay(distance_cm*20);
37.      noTone(11);
38.      delay(distance_cm*20);
39.    }
40. }
```