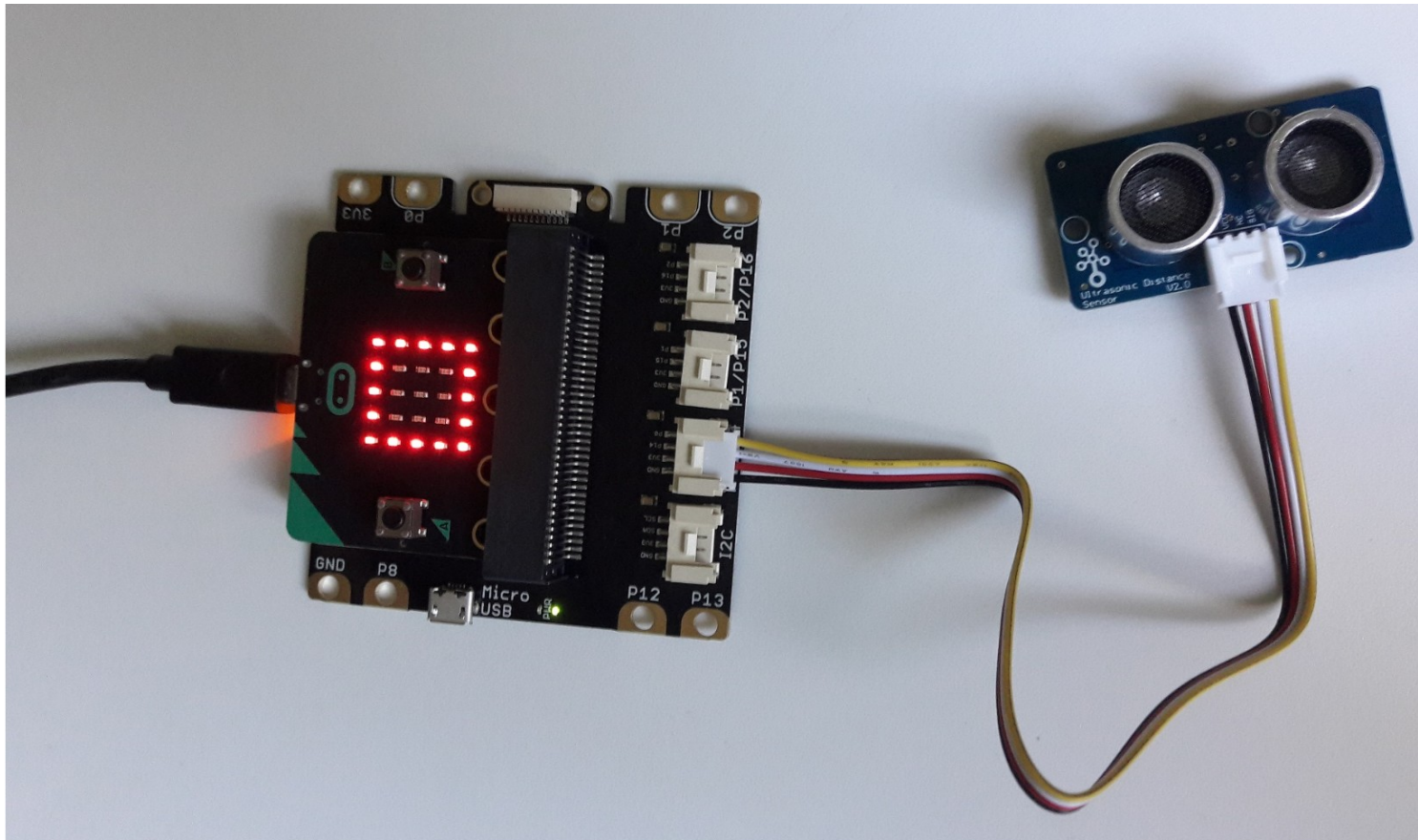
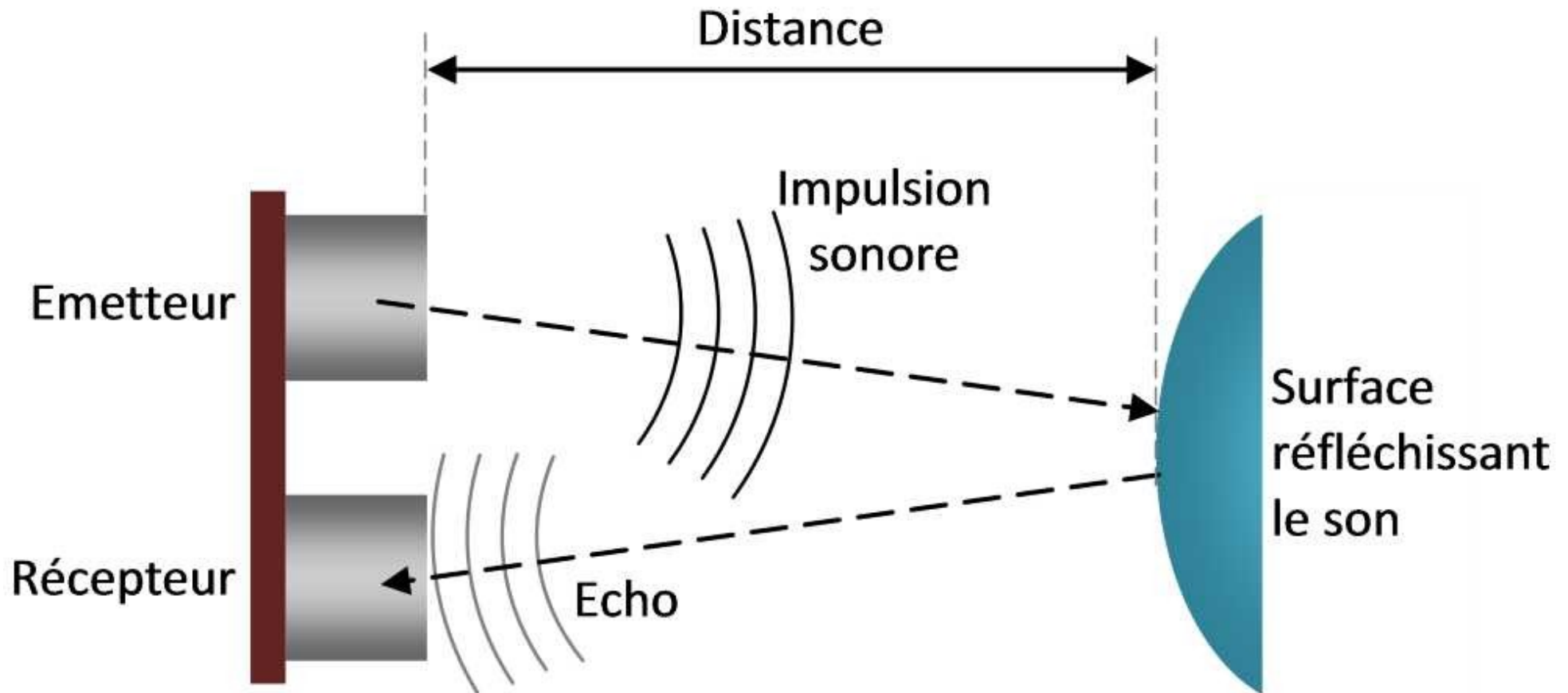


Télémètre à Ultrasons



Télémètre à Ultrasons

- Principe du télémètre

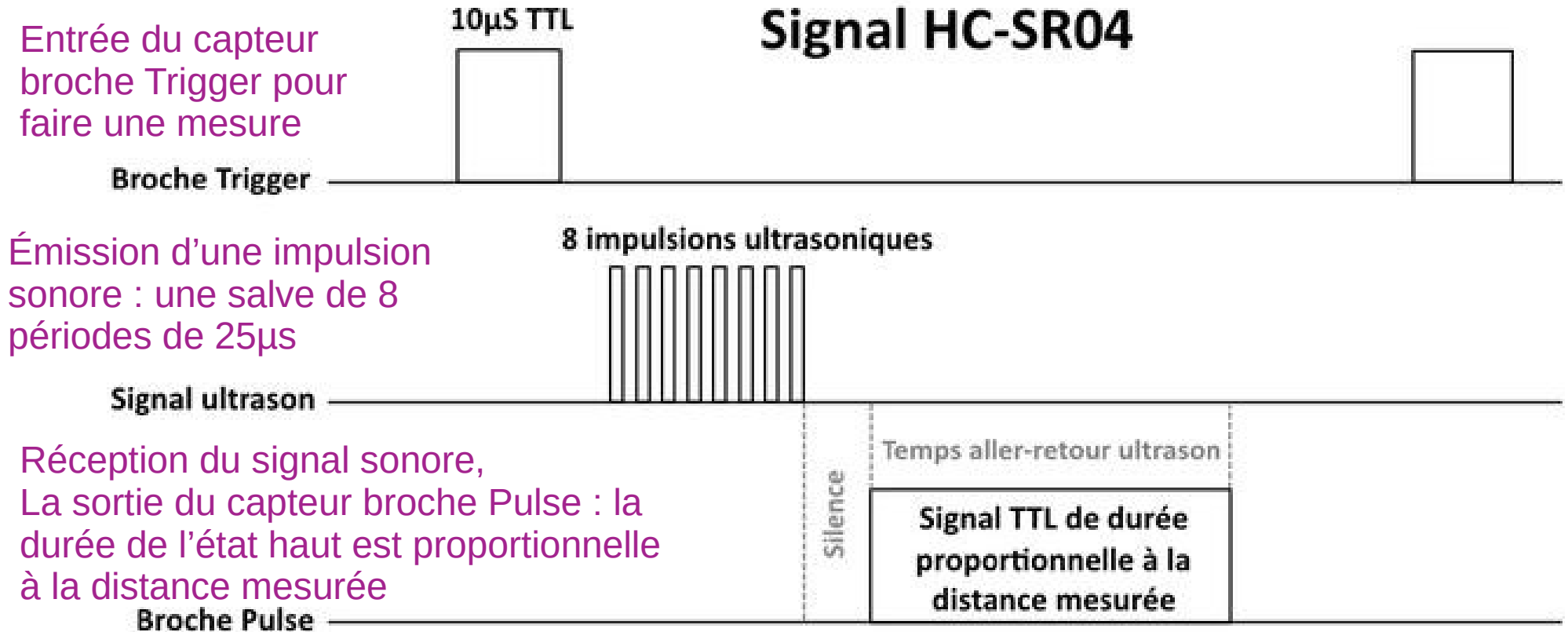


L'émetteur envoie un signal à un instant t_0 et le récepteur le reçoit à l'instant t_1 .

On connaît la vitesse du son 340m/s, à partir de $t_1 - t_0$ on en déduit la distance.

Télémètre à Ultrasons

Principe du Télémètre : Les signaux du capteurs à Ultrasons



Voilà comment se déroule une prise de mesure :

1. On envoie une impulsion HIGH de 10µs sur la broche TRIGGER du capteur.
 2. Le capteur envoie alors une série de 8 impulsions ultrasoniques à 40KHz
 3. Les ultrasons se propagent dans l'air jusqu'à toucher un obstacle et retourne dans l'autre sens vers le capteur à la vitesse de 340 m/s
 4. Le capteur détecte l'écho et clôture la prise de mesure. Broche Pulse
- Le signal sur la broche ECHO (Pulse sur le schéma) du capteur reste à HIGH durant les étapes 3 et 4, ce qui permet de mesurer la durée de l'aller-retour des ultrasons et donc de déterminer la distance.

Branchement

Carte shield

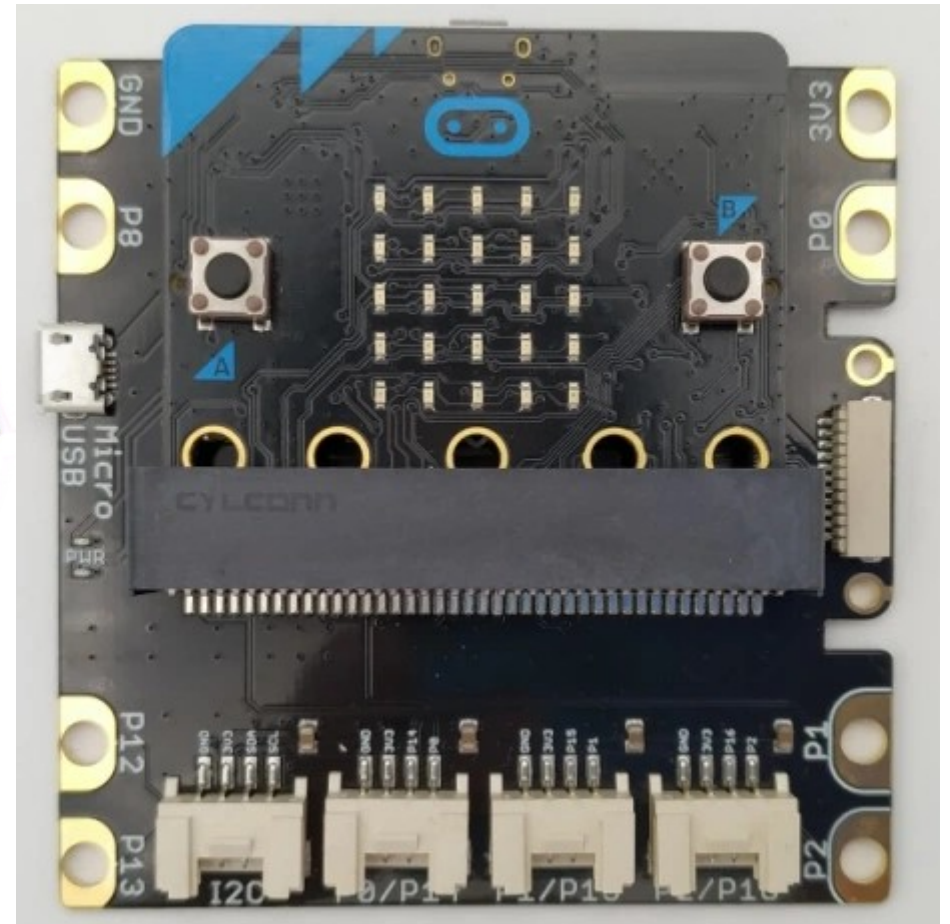
Le Micro:Bit se glisse très facilement dans sa carte d'extension. Elle ajoute quelques anneaux, permettant de connecter les ports P8, P12 et P13 (en plus des ports P0, P1, P2, GND et 3V3).

Attention au sens de la carte Micro:bit, les LEDs et boutons sont visibles comme sur l'image

Le détecteur à ultrasons se branche sur le connecteur P0/P14 voir la première diapo.

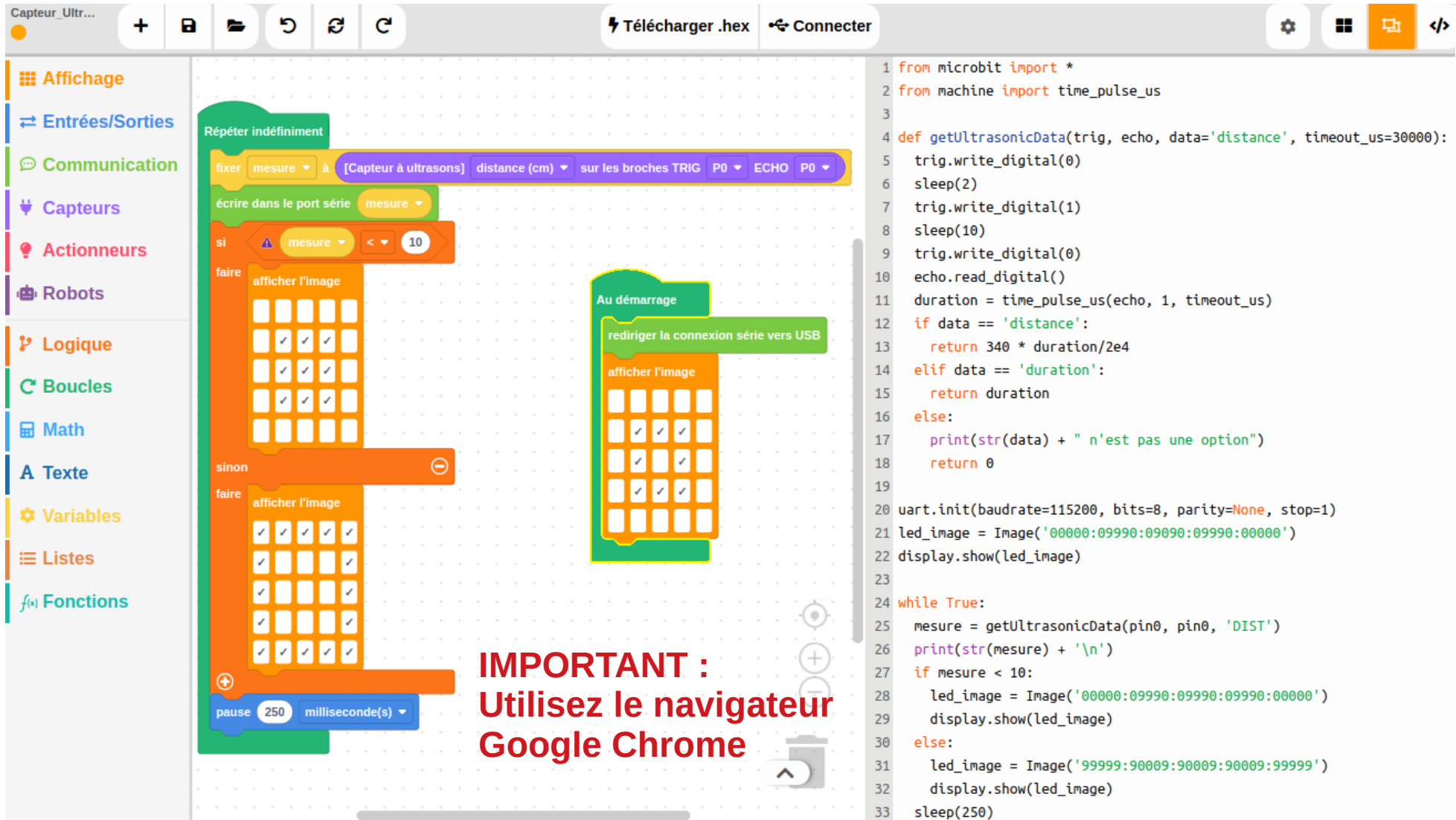


N'oubliez pas le câble USB entre la carte **Micro:bit** et le **PC** ! N'utilisez pas le Micro USB de la carte Shield



AP : Télémètre à ultrasons Grove

Programme : détection d'un obstacle à moins de 10 cm



The image shows the MicroPython IDE interface. On the left is a sidebar with categories: Affichage, Entrées/Sorties, Communication, Capteurs, Actionneurs, Robots, Logique, Boucles, Math, Texte, Variables, Listes, and Fonctions. The main workspace contains a block-based program and a Python code editor.

Block-based Program:

- Répéter indéfiniment** (Loop):
 - fixer mesure à [Capteur à ultrasons] distance (cm) sur les broches TRIG P0 ECHO P0**
 - écrire dans le port série mesure**
 - si mesure < 10** (If):
 - faire** (Do):
 - afficher l'image** (Display image block)
 - sinon** (Else):
 - faire** (Do):
 - afficher l'image** (Display image block)
 - pause 250 milliseconde(s)**

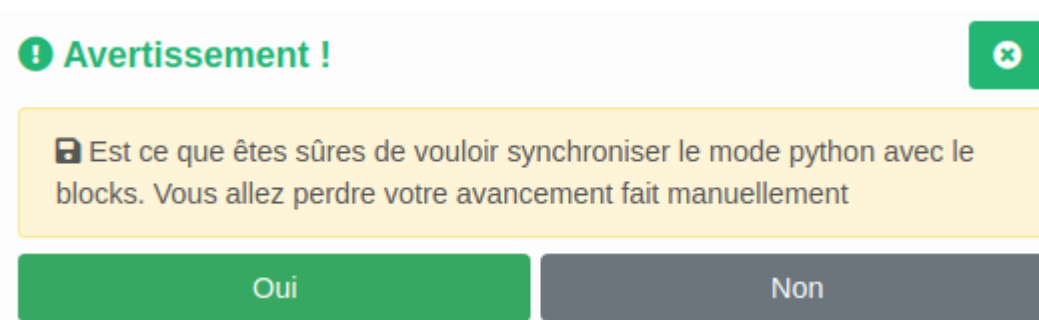
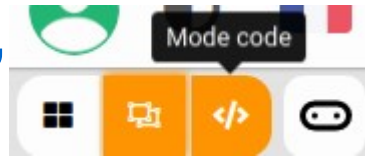
Python Code:

```
1 from microbit import *
2 from machine import time_pulse_us
3
4 def getUltrasonicData(trig, echo, data='distance', timeout_us=30000):
5     trig.write_digital(0)
6     sleep(2)
7     trig.write_digital(1)
8     sleep(10)
9     trig.write_digital(0)
10    echo.read_digital()
11    duration = time_pulse_us(echo, 1, timeout_us)
12    if data == 'distance':
13        return 340 * duration/2e4
14    elif data == 'duration':
15        return duration
16    else:
17        print(str(data) + " n'est pas une option")
18        return 0
19
20 uart.init(baudrate=115200, bits=8, parity=None, stop=1)
21 led_image = Image('00000:09990:09090:09990:00000')
22 display.show(led_image)
23
24 while True:
25     mesure = getUltrasonicData(pin0, pin0, 'DIST')
26     print(str(mesure) + '\n')
27     if mesure < 10:
28         led_image = Image('00000:09990:09990:09990:00000')
29         display.show(led_image)
30     else:
31         led_image = Image('99999:90009:90009:90009:99999')
32         display.show(led_image)
33     sleep(250)
```

IMPORTANT :
Utilisez le navigateur Google Chrome

AP : Télémètre à ultrasons Grove

Important : il faut terminer les blocs avant de passer en mode code en python, pour vous assurer que le code est synchronisé avec les blocs, cliquez sur Synchroniser Python avec les blocs puis



Une fois en mode code, ne revenez plus en mode mixte ou blocs !

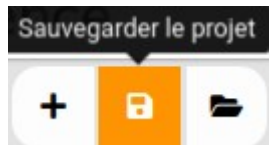
Important : Modification du code à effectuer. Remplacer la ligne :

`mesure = getUltrasonicData(pin0, pin0, 'DIST')`

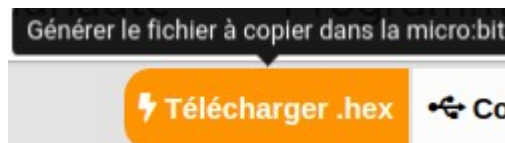
par

`mesure = getUltrasonicData(pin0, pin0, 'distance')`

On ne change que DIST par distance en fait.



Sauvegarder votre projet et enregistrer le fichier Python



Télécharger le fichier hex à mettre dans la carte Micro:bit

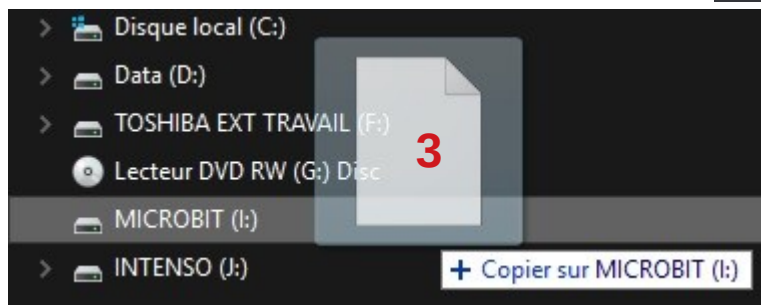
AP : Télémètre à ultrasons Grove

Programmation de la carte Micro:bit

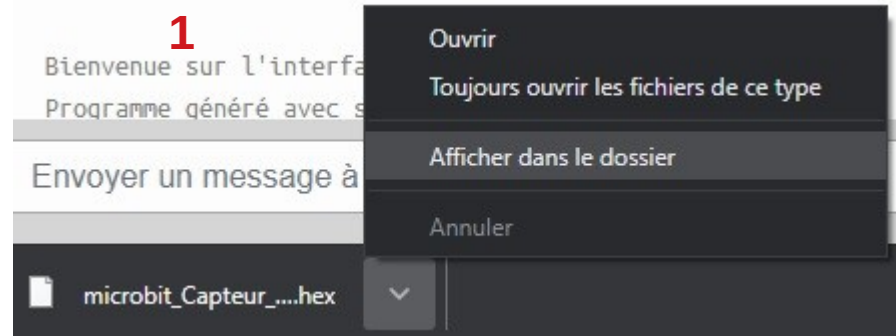
1 Nous venons de générer le fichier .hex
Nous allons en bas du navigateur Chrome
Pour ouvrir le dossier qui contient ce fichier



2 On sélectionne le fichier pour le copier dans MICROBIT **3**



4 On attend que le fichier soit transféré dans la carte Microbit



5 Dans Chrome on se connecte à la carte Micro:bit



AP : Télémètre à ultrasons Grove

Affichage des mesures de distance dans la console

En bas de Chrome : Ouvrez la fenêtre du bas et cliquez sur le Mode console



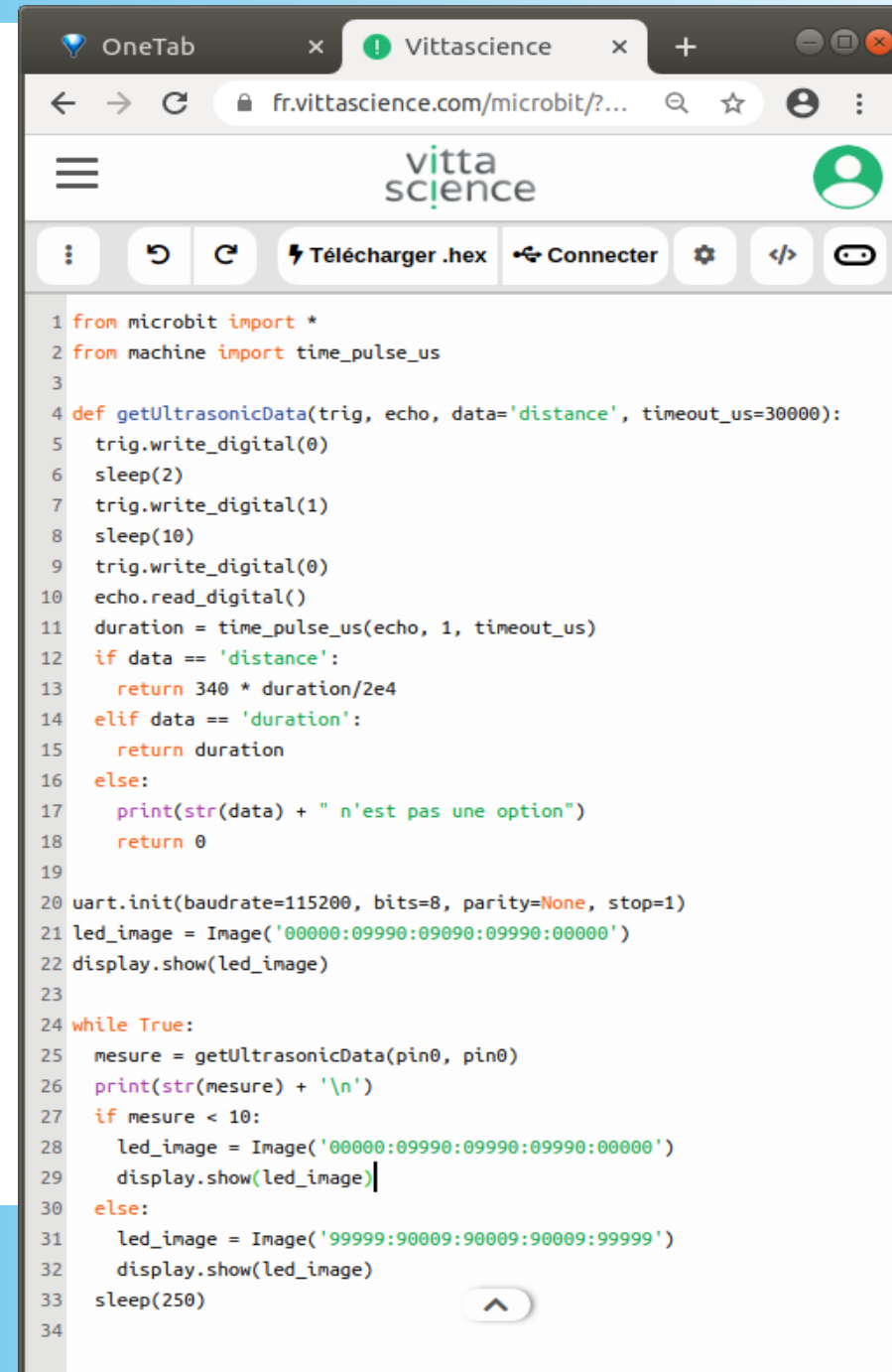
8.993

8.976

8.993

Les valeurs de la distance entre le capteur et un obstacle doivent défiler
Comme on le voit sur l'image à droite ou le zoom à gauche

Question : avec ces valeurs affichées, quelle est l'image sur les LED de la carte Micro:bit ?

A screenshot of a web browser window. The address bar shows 'fr.vittascience.com/microbit/?...'. The page header includes the 'vitta science' logo and a user profile icon. Below the header is a navigation bar with buttons: 'Télécharger .hex', 'Connecter', and others. The main content area displays a Python code editor for a Micro:bit project. The code defines a function 'getUltrasonicData' and uses it in a loop to read distance data and update an LED display. The LED display is represented as a 5x5 grid of LEDs, each with a hexadecimal value. The current state of the display is '00000:09990:09090:09990:00000'.

```
1 from microbit import *
2 from machine import time_pulse_us
3
4 def getUltrasonicData(trig, echo, data='distance', timeout_us=30000):
5     trig.write_digital(0)
6     sleep(2)
7     trig.write_digital(1)
8     sleep(10)
9     trig.write_digital(0)
10    echo.read_digital()
11    duration = time_pulse_us(echo, 1, timeout_us)
12    if data == 'distance':
13        return 340 * duration/2e4
14    elif data == 'duration':
15        return duration
16    else:
17        print(str(data) + " n'est pas une option")
18    return 0
19
20 uart.init(baudrate=115200, bits=8, parity=None, stop=1)
21 led_image = Image('00000:09990:09090:09990:00000')
22 display.show(led_image)
23
24 while True:
25     mesure = getUltrasonicData(pin0, pin0)
26     print(str(mesure) + '\n')
27     if mesure < 10:
28         led_image = Image('00000:09990:09990:09990:00000')
29         display.show(led_image)
30     else:
31         led_image = Image('99999:90009:90009:90009:99999')
32         display.show(led_image)
33     sleep(250)
34
```


Analyse du programme

Dans un premier temps, expliquez le programme général, la fonction `getUltrasonicData` sera étudié ensuite.

Commentez chaque ligne pour expliquer le programme

```
from microbit import *  
from machine import time_pulse_us
```

```
uart.init(baudrate=115200, bits=8, parity=None, stop=1)  
led_image = Image('00000:09990:09090:09990:00000')  
display.show(led_image)
```

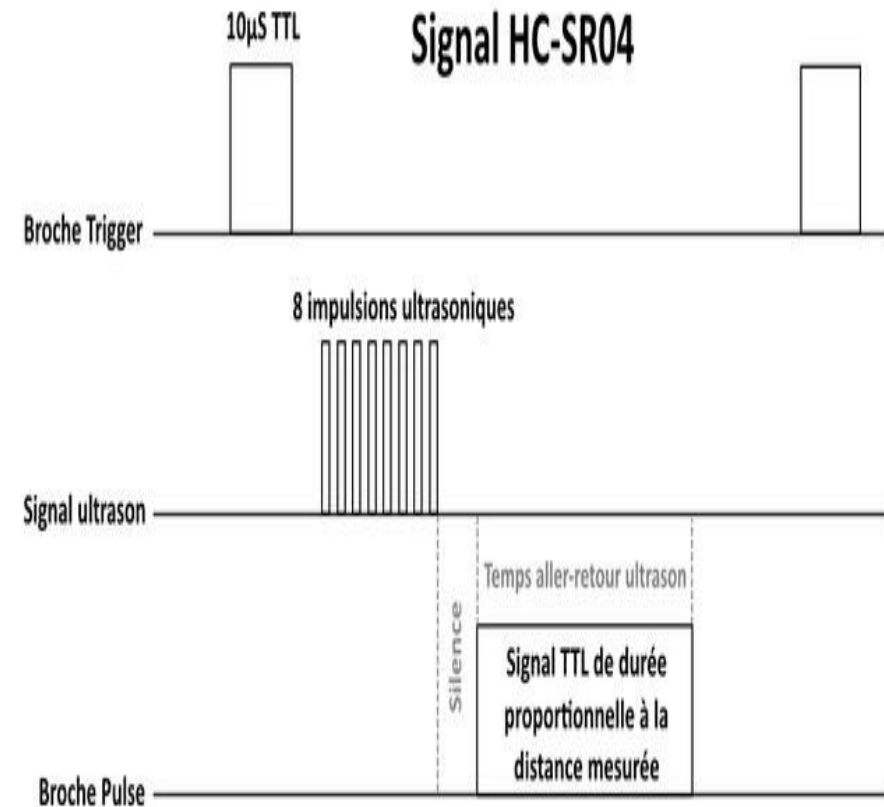
```
while True:  
    mesure = getUltrasonicData(pin0, pin0, 'distance')  
    print(str(mesure) + '\n')  
    if mesure < 10:  
        led_image = Image('00000:09990:09990:09990:00000')  
        display.show(led_image)  
    else:  
        led_image = Image('99999:90009:90009:90009:99999')  
        display.show(led_image)  
    sleep(250)
```

Analyse du programme

Faites un lien entre la fonction python getUltrasonicData et les signaux de « Signal HC-SR04 »

```
from microbit import *
from machine import time_pulse_us

def getUltrasonicData(trig, echo, data='distance', timeout_us=30000):
    trig.write_digital(0)
    sleep(2)
    trig.write_digital(1)
    sleep(10)
    trig.write_digital(0)
    echo.read_digital()
    duration = time_pulse_us(echo, 1, timeout_us)
    if data == 'distance':
        return 340 * duration/2e4
    elif data == 'duration':
        return duration
    else:
        print(str(data) + " n'est pas une option")
        return 0
```



Structure if elif else voir Boussole magnétique Exo14

Analyse du programme

- ✓ Expliquer la formule « $340 \cdot \text{duration} / 2e4$ » dans la ligne `return 340*duration/2e4`

Aidez-vous des lignes précédentes et du dessin ainsi que de la vitesse du son. Les unités son très importantes.

Si je remplace la ligne :

```
print(str(getUltrasonicData(pin0, pin0, 'distance')) + '\n') par
```

```
print(str(getUltrasonicData(pin0, pin0, 'duration')) + '\n')
```

- ✓ que se passe-t-il ?

Le timeout correspond au temps auquel s'arrêtera la mesure, même si le signal n'est pas revenu à 0, pas d'obstacle détecté.

- ✓ En le fixant à 30000 μs par défaut, quelle est la distance maximale mesurable ?

Modification du programme

Détection d'obstacle pour un robot

On souhaite faire 2 seuils de distance un premier à 30 cm avec une image1 et un second seuil à 10 cm avec une image2 et sinon une image3. Il faudra créer une nouvelle image.

Pour vous aider, regardez les exemples dans Bloc_Python_Microbit2.odp

Vous pouvez le faire en reprenant les blocs mais n'oubliez pas dans ce cas de synchroniser et de modifier la ligne « mesure = ... » (diapo6) ou bien directement en python.

Testez votre programme, quand celui-ci est satisfaisant sauvegardez votre programme python (.py) avec un nouveau nom ainsi que le fichier (.hex) (voir diapo 6 et 7)

Vous devrez rendre 2 fichiers python, 2 fichiers .hex et un fichier texte `telemetre_nom_prenom.odt` avec vos réponses aux questions