



ECOLE CENTRALE DE PÉKIN
LYCÉE LA MARTINIÈRE MONPLAISIR LYON

SCIENCES INDUSTRIELLES POUR L'INGÉNIEUR

ANNÉE 2025 - 2026



北航中法工程师学院
Centrale Pékin



ECHANGES PÉDAGOGIQUES AVEC L'ÉCOLE CENTRALE DE
PÉKIN

20 OCTOBRE 2025

TD 1 - Résolution de problèmes

Compétences

- **Analyser**

- Traduire un besoin fonctionnel en exigences.
- Justifier le choix des constituants dédiés aux fonctions d'un système.
- Identifier la structure d'un système asservi.

- **Modéliser**

- Choisir les grandeurs physiques et les caractériser
- Identifier les performances à prévoir ou à évaluer.
- Identifier les grandeurs d'entrée et de sortie d'un modèle.
- Identifier les paramètres d'un modèle.
- Identifier et justifier les hypothèses nécessaires à la modélisation.
- Modéliser un système par schéma-blocs.

- **Résoudre**

- Proposer une démarche permettant de répondre à un cahier des charges.

- **Expérimenter**

- Repérer les constituants réalisant les principales fonctions des chaînes fonctionnelles.
- Faire des mesures.
- Identifier les erreurs de méthode.

Exercice 1 : Problème ouverts sur le Robot Maqueen

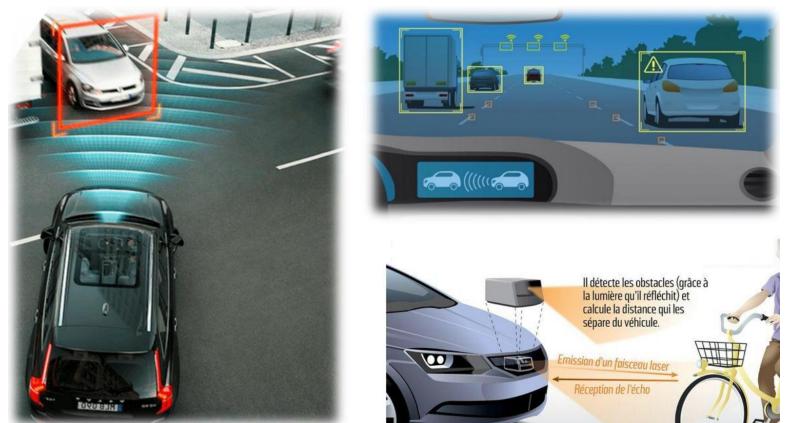
Source : Emilien DURIF

1 Présentation

a) Objectifs

Objectif 1 :

L'objectif est de mettre en place un système permettant à un véhicule de s'arrêter à une distance donnée d'un obstacle, ou de suivre un autre véhicule en conservant un espace avec ce dernier.



b) Matériel à disposition

On propose de réaliser une maquette à échelle réduite.

- 1 robot macqueen vplusV2.1 avec Capteur ultra-son
- 1 robot macqueen vplus V3 avec Capteur Lidar
- 2 accumulateurs 3,7 V
- 2 shield pour carte micro : bit
- 2 câbles usb
- 2 ordinateurs avec avec un navigateur web pour aller sur makecode



2 Mise en oeuvre du robot

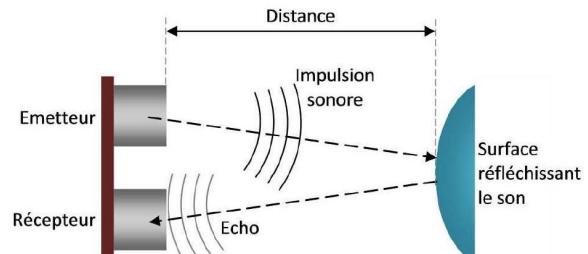
a) Mesure de distance

Le principe de fonctionnement du capteur est basé sur une mesure du temps que met le son pour faire l'aller-retour entre le capteur et l'obstacle.

Voilà comment se déroule une prise de mesure :

1. On envoie une impulsion HIGH de $10\mu\text{s}$ sur la broche TRIGGER du capteur.
2. Le capteur envoie alors une série de 8 impulsions ultrasoniques à 40 kHz (inaudible pour l'être humain, c'est quand plus agréable qu'un biiiiiiip).
3. Les ultrasons se propagent dans l'air jusqu'à toucher un obstacle et retournent dans l'autre sens vers le capteur.
4. Le capteur détecte l'écho et clôture la prise de mesure.
5. Le signal sur la broche ECHO 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.

N.B. Il y a toujours un silence de durée fixe après l'émission des ultrasons pour éviter de recevoir prématurément un écho en provenance directement du capteur.



fonctionnement du capteur

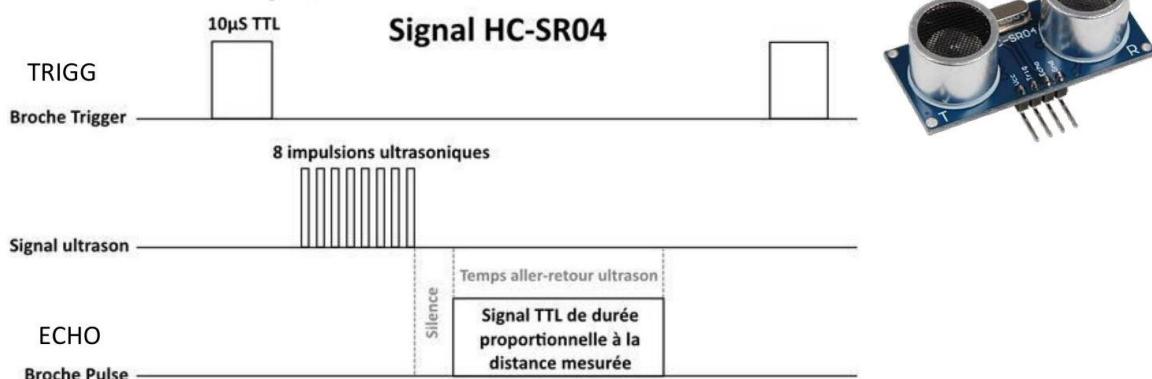


FIGURE 1 – Principe de fonctionnement du capteur ultra-son

Algorithme 1 :

L'algorithme se trouve sur le site de la classe et se nomme **lire_us.py**. Vous pouvez l'ouvrir dans **mu editor**.

```
from microbit import *
import utime

def distance():
    # Programme de lecture du capteur ultra son
    # Envoi impulsion TRIG sur P1
    pin1.write_digital(0)
    utime.sleep_us(2)
    pin1.write_digital(1)
    utime.sleep_us(10)
    pin1.write_digital(0)

    # Attente du front montant sur ECHO (P2)
    timeout = utime.ticks_add(utime.ticks_us(), 30000) # 30 ms timeout
    while pin2.read_digital() == 0:
        if utime.ticks_diff(timeout, utime.ticks_us()) <= 0:
            return None
    start = utime.ticks_us()

    # Attente du front descendant
    timeout = utime.ticks_add(utime.ticks_us(), 30000)
    while pin2.read_digital() == 1:
        if utime.ticks_diff(timeout, utime.ticks_us()) <= 0:
            return None
    end = utime.ticks_us()

    # Durée de l'écho
    duration = utime.ticks_diff(end, start)

    # Conversion en cm
    dist_cm = (duration / 2) / 29.1
    return dist_cm

while True:
    d = distance()
    if d:
        # affichage des mesures de disance en cm
        print((int(d),))
    else:
        # on affiche 0 si pas d'echo
        print((0,))
    sleep(50)
```

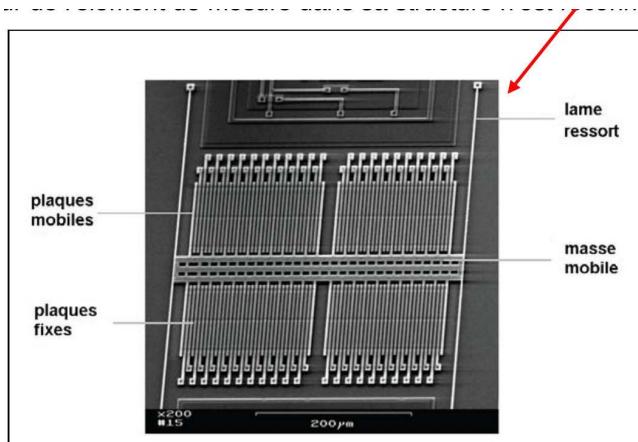
Activité 1 :

- ouvrir le logiciel mu editor et ouvrir le fichier lire_us.py (on peut faire un glisser depuis le répertoire)
- Téléverser votre programme sur la carte (Flasher)
- Cliquer sur Graphique
- Puis cliquer sur REPL
- mettre sous tension le robot maqueen
- Observer les valeurs sur le prompt et le graphique

Activité 2 : Mettre en place un protocole permettant de valider la mesure. Conclure sur la linéarité, la justesse, le domaine de validité de la mesure de distance.

b) Mesure des accélérations

Les accéléromètres sont composés de deux puces de silicium : l'élément de mesure et le circuit d'interprétation. Il permet de mesurer des accélération en $m \cdot s^{-2}$. Le coeur de l'élément de mesure dans sa structure n'est reconnaissable qu'au microscope. On peut retrouver plus d'informations sur son fonctionnement en annexe.

**Algorithme 2 :**

L'algorithme se trouve sur le site de la classe et se nomme **lire_accelero.py**. Vous pouvez l'ouvrir dans **mu editor**.

```
from microbit import *

while True:
    # Lire des données de l'accelerometer
    acc = accelerometer.get_values()
    # Affichage des données
    print(acc)
    sleep(50) # Pause entre chaque mesure
```

Activité 3 :

- ouvrir le logiciel mu editor et ouvrir le fichier lire_us.py (on peut faire un glisser depuis le répertoire)
- Téléverser votre programme sur la carte (Flasher)
- Cliquer sur Graphique
- Puis cliquer sur REPL
- Observer les valeurs sur le prompt et le graphique

Activité 4 : Vérifier qualitativement les mesures. Comment pourrait-on trouver les bonnes unités ?

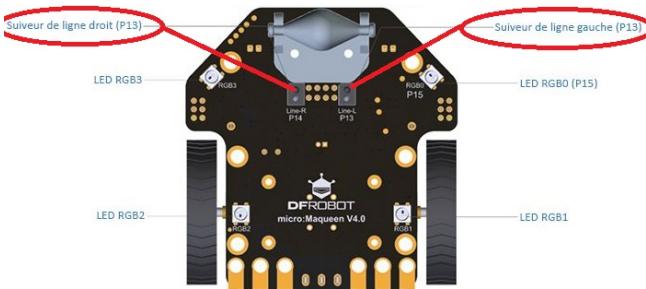
c) Mesure de présence de ligne

Le Maqueen est équipé de **deux capteurs infrarouges** placés à l'avant du robot :

- Capteur gauche connecté à la broche P13.
- Capteur droit connecté à la broche P14.

Chaque capteur émet un rayon infrarouge et mesure la lumière réfléchie par le sol :

Le capteur de ligne à infrarouge renvoie une information logique en fonction de la présence ou de l'absence de ligne noire. Le capteur renvoie 1 (ou VRAI) s'il est au-dessus du noir et 0 (ou FAUX) s'il est au-dessus d'une surface claire.



- Si la surface est **claire** (blanche), la lumière est réfléchie ⇒ le capteur renvoie la valeur **1**.
- Si la surface est **foncée** (ligne noire), la lumière est absorbée ⇒ le capteur renvoie la valeur **0**.



Algorithme 3 :

En MicroPython, on lit ces capteurs avec `read_digital()` :

```
from microbit import *

while True:
    gauche = pin13.read_digital()
    droite = pin14.read_digital()

    # 0 = ligne noire détectée
    print("G:", gauche, " D:", droite)
    sleep(200)
```

Valeur lue	Surface détectée
0	Ligne noire (absorbe l'IR)
1	Surface claire (réfléchit l'IR)

Activité 5 :

- ouvrir le logiciel mu editor et ouvrir le fichier `lire_capteur_ligne.py` (on peut faire un glisser depuis le répertoire)
- Téléverser votre programme sur la carte (Flasher)
- Cliquer sur Graphique
- Puis cliquer sur REPL
- mettre sous tension le robot maqueen
- Observer les valeurs sur le prompt et le graphique

Activité 6 : Vérifier qualitativement les mesures à l'aide des circuit noir et blanc imprimés sur des feuilles A3.

d) Mise en mouvement

Le principe de pilotage du motoréducteur est basé sur une communication en I2C entre la carte microbit et le driver (hacheur) implanté dans la carte électronique du châssis.

Le constructeur annonce les caractéristiques suivantes pour le moto-réducteur sur le figure 2

Le constructeur annonce les caractéristiques suivantes pour le moto-reducteur :



Puissance nominale : 0.5W
Rapport de réduction : 1 :150
Rendement : 0.9
Vitesse de rotation sous 5V : 133 tr/min
 (en sortie de réducteur)
Mode de pilotage en PWM

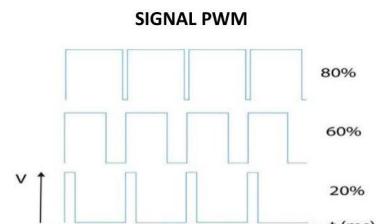


FIGURE 2 – Principe de fonctionnement du moteur

Le pilotage en PWM consiste à envoyer au moteur une tension entre 0 et 5 V hachée sur une période très courte. La période des créneaux étant si faible que le moteur reçoit la tension moyenne du signal, ce qui permet ainsi d'obtenir

une variation de la tension à ses bornes, ainsi une variation de vitesse du moteur.

Le pilotage de chaque moteur est indépendant.

Fonctions python permettant le pilotage du moteur en fonction des deux variables sens et vitesse.

Algorithme 4 :

```
from microbit import *

def moteurDroit(sens, vitesse):
    i2c.write(0x10, bytearray([2, sens, vitesse]))

def moteurGauche(sens, vitesse):
    i2c.write(0x10, bytearray([0, sens, vitesse]))

moteurDroit(0, 255)
moteurGauche(1, 255)
```

La variable sens prend pour valeur **0** pour la marche avant ou **2** pour la marche arrière.

La variable vitesse est une valeur numérique codée sur 8 bits, elle varie donc entre 0 et 255.

✓ A l'aide des instructions en annexe 1, implémenter le programme permettant la mise en route des moteurs pour une vitesse de 125.

✓ A l'aide du tachymètre, mesurer la vitesse obtenue de la roue droite, la vitesse expérimentale correspond-elle à la vitesse théorique ?

✓ Reprendre pour un PWM de 80, 180, 255.

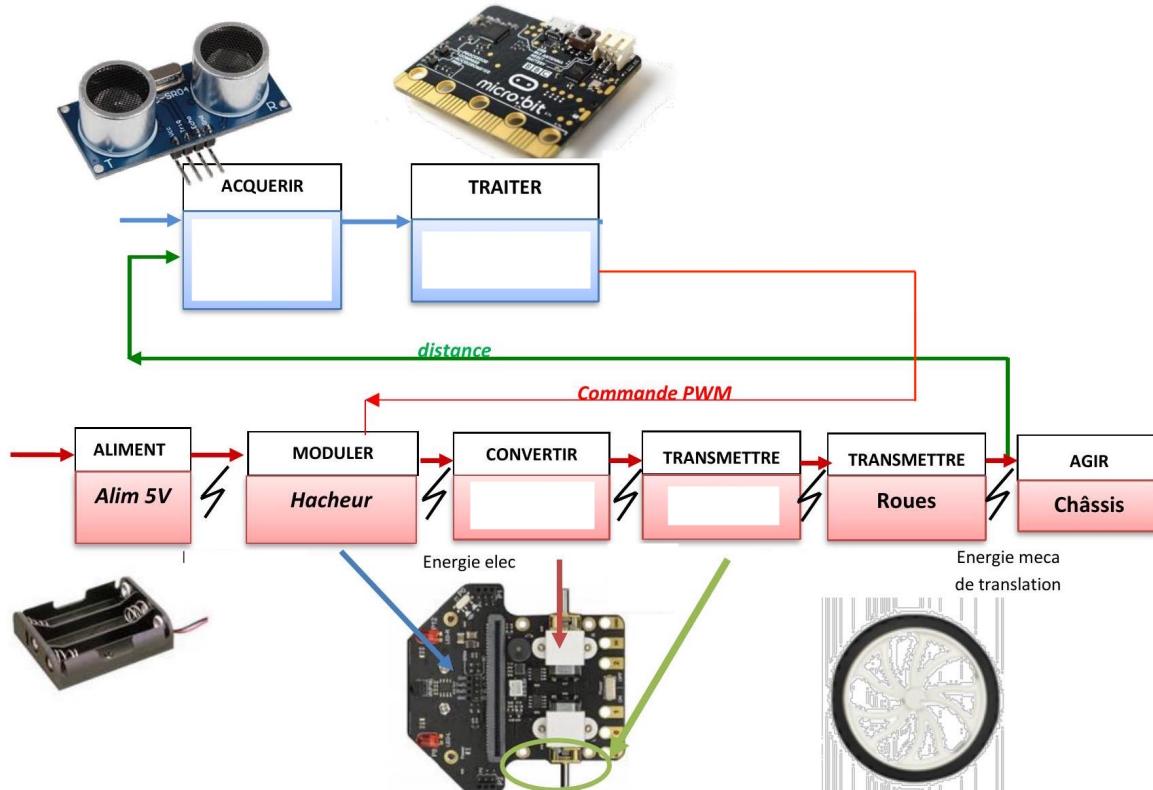
Activité 7 : Mettre en oeuvre le programme pour faire tourner le robot sur place, le faire aller en ligne droite ou lui faire prendre un virage.

Activité 8 :

- ouvrir le logiciel mu editor et ouvrir le fichier test_moteur.py (on peut faire un glisser depuis le répertoire)
- téléverser votre programme sur la carte (Flasher)
- débrancher le câble USB
- mettre sous tension le robot maqueen

3 Analyse structurelle du robot

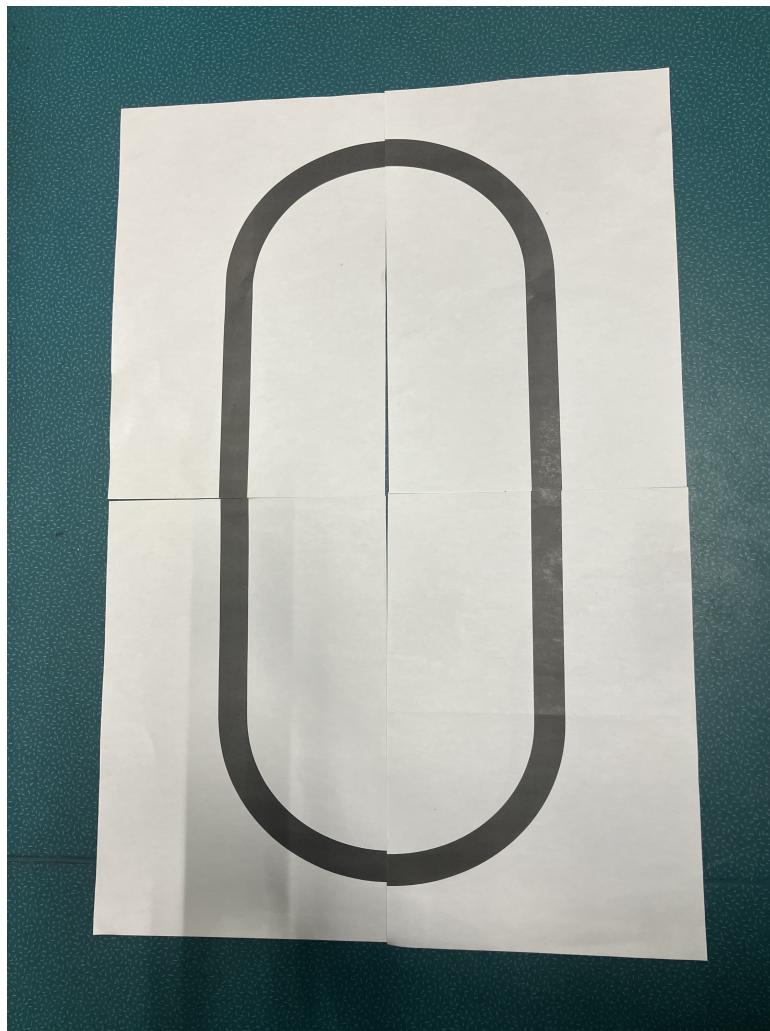
Q 1 : A l'aide des activités proposées précédemment, proposer une analyse structurelle sous la forme d'une chaîne fonctionnelle du robot maqueen.



4 Pour aller plus loin

a) Suivi de circuit en boucle ouverte

On souhaite que le robot suive le circuit ci-contre.



Q 2 : Sous la forme d'un diagramme de séquence proposer une évolution séquentielle du robot.

Q 3 : En utilisant les parties précédentes écrire un programme permettant de réaliser cette action et le mettre en oeuvre.

b) Suivi de circuit en utilisant les capteurs

Q 4 : Sous la forme d'un diagramme de séquence proposer une évolution séquentielle du robot.

Q 5 : En utilisant les parties précédentes écrire un programme permettant de réaliser cette action et le mettre en oeuvre.

5 Annexes :

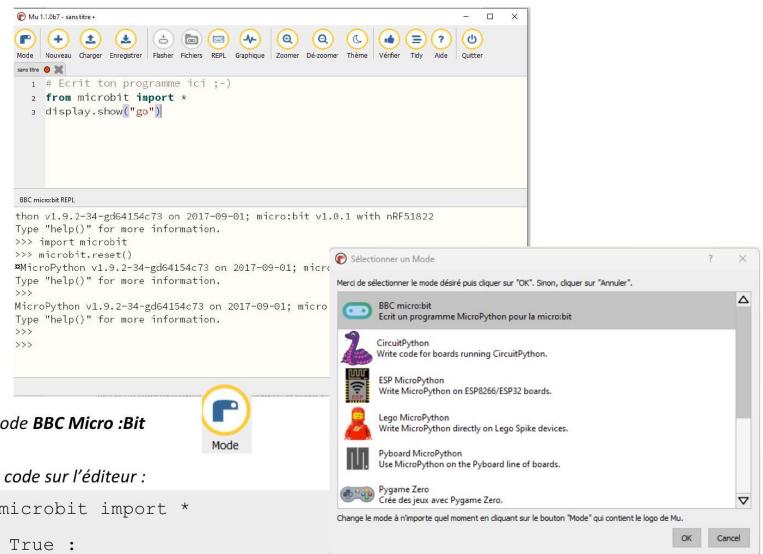
a) Prise en main de la carte

L'éditeur Python utilisé sera l'éditeur Mu¹, permettant d'utiliser facilement les cartes microbit.

1. Lien pour installer le logiciel mu : <https://codewith.mu/en/download>

1. Brancher la carte microbit via le port USB au PC.
2. Ouvrir l'éditeur Mu :
3. Choisir le mode BBC Micro :Bit
4. Recopiez ce code sur l'éditeur :

```
from microbit import *
while True :
    display.show(Image.HAPPY)
    sleep(1000)
    display.show(Image.DUCK)
    sleep(1000)
    display.show(Image.GHOST)
    sleep(1000)
```



5. Cliquez sur Flasher pour téléverser le programme.

b) Prise en main de MU editor avec le robot Maqueen

- Lancer le logiciel Mu Editor
- Mettre hors tension le robot avec l'interrupteur ON/OFF à l'arrière [1]. La DEL «Power» indique que le robot est sous tension [2].
- Connecter le Maqueen au PC avec le câble USB, l'indication en bas à droite permet de vérifier que le robot est bien connecté [3]. Vous pouvez fermer la fenêtre Windows confirmant la présence de la carte si elle s'ouvre [4].
- Le programme est écrit dans la fenêtre principale [5].
- Il est possible de vérifier les erreurs de syntaxe avec l'outil «Vérifier» [6].
- L'outil «Flasher» [7] permet de copier le programme sur la carte micro bit. Attendre que la DEL à l'arrière de la carte arrête de clignoter [8].
- Si un message s'affiche sur la matrice de DEL, le programme ne fonctionnera peut-être pas à cause d'une ou plusieurs erreurs [9].
- Débrancher le câble USB et mettre le robot sous tension. Le programme s'exécute immédiatement.
- Observer le fonctionnement et mettre le robot hors tension.
- Il est possible d'appuyer sur le bouton «reset» à l'arrière de la carte [10] pour relancer le programme et arrêter le robot.

Si vous devez attraper le robot en mouvement soyez très précautionneux. Il faut le soulever par le châssis sans toucher aux roues ni à l'électronique.

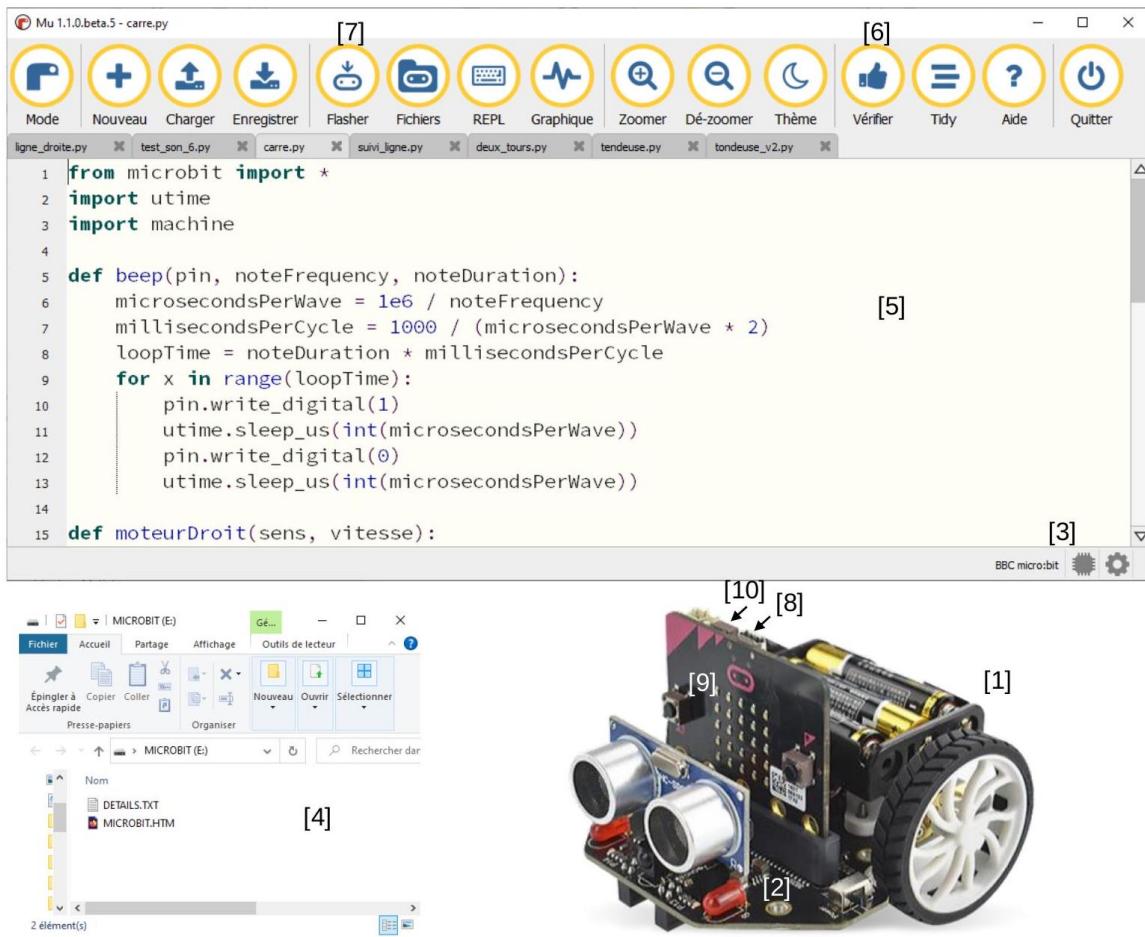


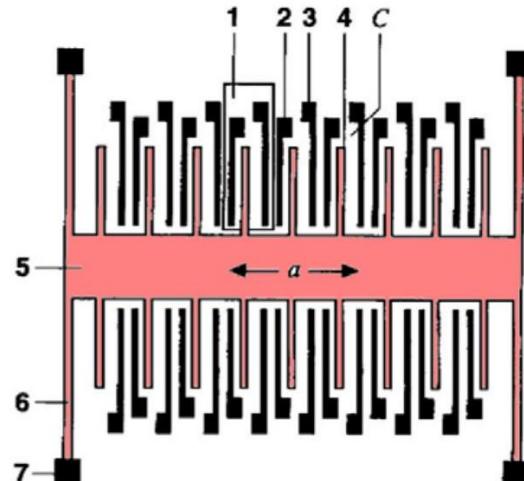
FIGURE 3 – Interface MU editor et présentation du robot

c) Accéléromètre

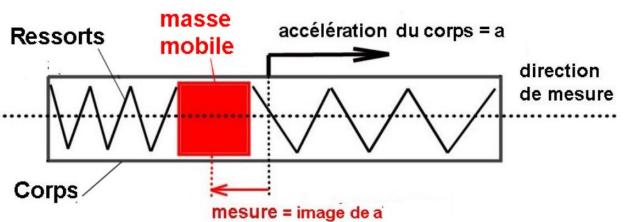
Dans le détail, la masse mobile (5) avec ses électrodes en forme de peigne (1,2,3,4) est suspendue de manière souple sur de fines barrettes (6). Des électrodes fixes (2,3) placées sur la puce se trouvent de chaque côté des électrodes mobiles (4).

A l'accélération du capteur, la distance entre les électrodes mobiles et celles fixes se modifie, entraînant une modification du signal électrique dans la puce d'interprétation.

**1 cellule élémentaire, 2,3 plaques fixes,
4 plaques mobiles, 5 masse mobile,
6 lame-ressort, 7 ancrage.
 a accélération, C capacités de mesure.**



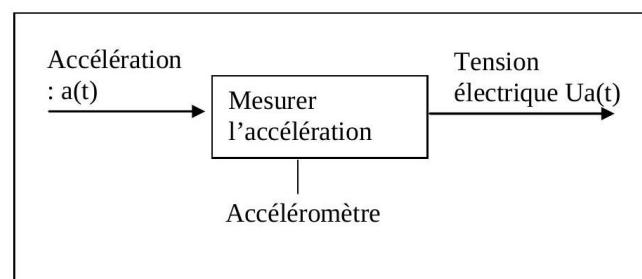
Un schéma équivalent qui représente bien le fonctionnement mécanique de l'accéléromètre à un axe est le «système masseressort» : fonction pour laquelle l'accéléromètre a été conçu : « mesurer l'accélération selon la direction de mesure «dm» »



C'est le principe fondamental de la dynamique (ou deuxième loi de Newton) qui nous permet d'écrire que La masse m est soumise à une force $F = m.a$;

Et donc ce sont les ressorts qui exercent cette force $F = m.a$. (en réaction à l'accélération)

La mesure fournie par le capteur est : une image de cette force, et est donc une image de « a ». si l'accéléromètre est placé verticalement, et fixé au sol, la masse mobile est soumise à la force de pesanteur $P = m.g$; les ressorts exercent sur la masse la force de réaction $F = m.g$; donc l'accéléromètre donne une mesure qui est l'image de l'accélération de la pesanteur : « g »



On peut se servir d'un accéléromètre comme d'un inclinomètre à condition que le mouvement soit plutôt lent « mesurer l'inclinaison « i » » (voir l'angle « i » sur la figure ci-contre). Si l'accéléromètre est placé incliné d'un angle « i » par rapport à l'horizontale, la composante du poids qui intervient dans la direction de mesure est $P_y = m.g.sin(i)$; en conséquence les ressorts exercent la force de réaction $F = m.g.sin(i)$; l'accéléromètre fournit donc une image de $g.sin(i)$ et permet donc de mesurer l'angle « i ».

