STRUCTURES DE BASES – APPLICATIONS SUR CARTE PROGRAMMABLE

I. PRÉSENTATION : CARTE BBC MICRO:BIT

A. HISTORIQUE

La carte BBC Micro:bit est un « micro ordinateur de poche » (ou carte microcontrôleur) réalisé par la BBC en 2015. Initialement conçue pour permettre aux élèves du Royaume-Uni de s'initier dès l'âge de sept ans à l'algorithmique et à la programmation, elle est désormais accessible à tous. Ce projet prévoit d'offrir gratuitement un exemplaire à chaque écolier britannique de douze ans.

Cette carte peut être programmée à partir d'un ordinateur, d'un smartphone ou d'une tablette. Elle permet de s'initier à l'informatique embarquée, disposant nativement de nombreux capteurs et broches d'entrée-sortie, et possède la dernière technologie qui équipe les appareils modernes : téléphones mobiles, réfrigérateurs, montres intelligentes, alarmes antivol, robots, etc...

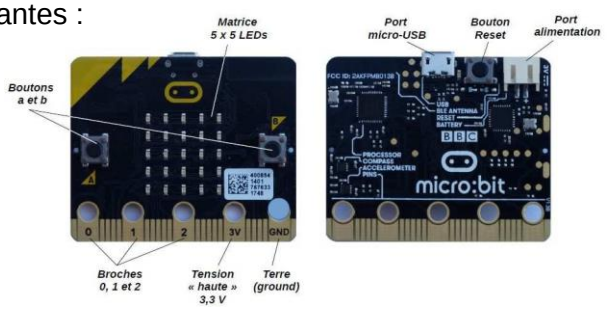
Ainsi, elle s'apparente à ce que l'on nomme l'Internet des objets : Internet of Things, abrégé IoT.

B. FONCTIONNEMENT

La carte BBC Micro:bit peut se programmer en utilisant plusieurs langages : Javascript (blocs ou texte), Scratch3, LISP, C++ avec l'environnement Arduino mais aussi MicroPython. Nous nous intéresserons dans cette séquence uniquement à la programmation de la carte sous MicroPython.

La carte Micro:bit dispose des spécificités techniques suivantes :

25 LEDs programmables individuellement
2 boutons programmables (A et B)
Broches de connexion
Capteurs de lumière et de température
Capteurs de mouvements (accéléromètre et boussole)
Communication sans fil, via Radio et Bluetooth
Interface USB

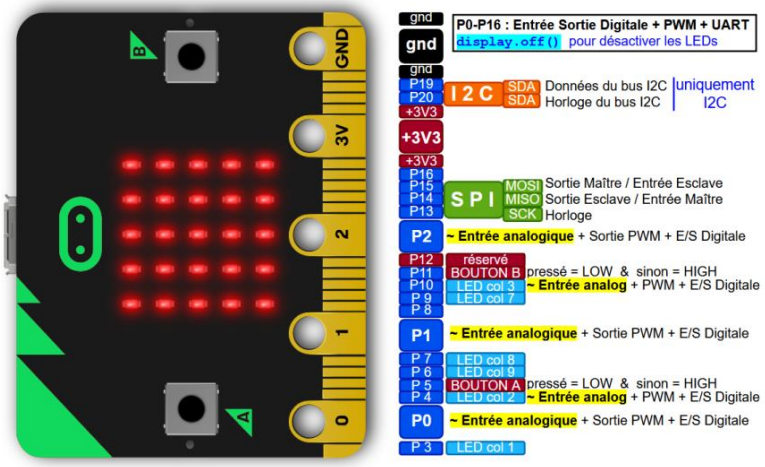


Ouvrir le logiciel *Mu*. Au démarrage, choisir le mode BBC micro:bit. Brancher la carte Micro:bit sur un port USB.

Pour apprendre à programmer sur la carte Micro:bit, on utilisera :

- en cours, un exemplaire de la carte Micro:bit et le logiciel *Mu* pour programmer en MicroPython
- à la maison, l'*émulateur* en ligne (pour programmer et visualiser) : https://create.withcode.uk

Sur le logiciel *Mu*, une fois le programme écrit, il faut soit déposer le programme microPython (format .hex) sur la carte, soit plus simplement Flasher le programme sur la carte.



II. Prise en main

Commande MicroPython

`display.scroll(string, delay=400)` : affiche une chaîne de caractères (*string*, du texte) en défilement avec une certaine vitesse (*delay*, plus le délai est grand, moins le texte défilera rapidement)

1. Dans le logiciel *Mu*, écrire le programme ci-dessous puis flasher le programme sur la carte.

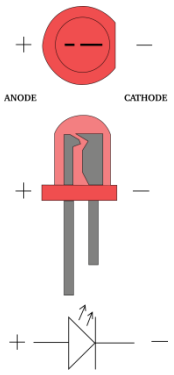
```
from microbit import *

display.scroll("BIENVENUE")
```

2. Modifier le programme pour que le texte « I LOVE NSI » défile assez lentement.

III. Lumière !

A. Connecter une DEL



Une diode électroluminescente (abrégé en DEL en français, ou LED, de l'anglais : *light-emitting diode*) est un composant électronique capable d'émettre de la lumière lorsqu'il est parcouru par un courant électrique. Une diode électroluminescente ne laisse passer le courant électrique que dans un seul sens. *(wikipedia)*

Pour connecter la DEL à la carte, utiliser les fils de connexion avec les pinces « croco » qui se clipsent sur les broches de la carte Micro:bit.
L'anode de la DEL se reconnaît à sa patte plus courte. Elle doit se connecter au pôle « + ».
Utiliser la broche 0 de la carte Micro:bit.
La cathode de la DEL doit se connecter au pôle « - ». Utiliser la broche terre (*ground* en anglais, abrégée en GND).

B. Clignoter

Sortie : écrire sur une broche

`pin2.write_digital(1)` impose 3V sur la broche 2 (en mode sortie).
`pin2.write_digital(0)` abaisse la tension à 0V

Attendre (delay) – pause

`sleep(10)` pour attendre 10ms

Boutons A et B

`was_pressed()` : cette méthode renvoie `True` si le bouton a été actionné pendant que le programme était occupé à une autre tâche.
`is_pressed()` : renvoie `True` si le bouton est actionné à l'instant où cette méthode est invoquée.

Exercice 1 :

A l'aide des commandes ci-dessus,

- créer un programme pour faire clignoter la DEL.
- le programme ci-dessous affiche une image gaie à l'appui sur le bouton A et une image triste à l'appui sur le bouton B. En s'inspirant de cet exemple, créer un programme qui allume la DEL à l'appui sur le bouton A et l'éteint à l'appui sur le bouton B.

```
from microbit import *
while True:
    if button_a.was_pressed():
        display.show(Image.HAPPY)
    else:
        display.show(Image.SAD)
     sleep(500)
```

C. A l'aide !

Exercice 2 :
A l'aide du code morse , créer un programme qui produit un SOS lumineux avec la DEL.

Exercice 3 :
Pour améliorer ce programme,

- créer une fonction « morse_S » qui code la lettre S en morse sur la DEL.
- créer une fonction « morse_O » qui code la lettre O en morse sur la DEL.
- créer un programme qui lance indéfiniment un appel SOS en utilisant ces deux fonctions.

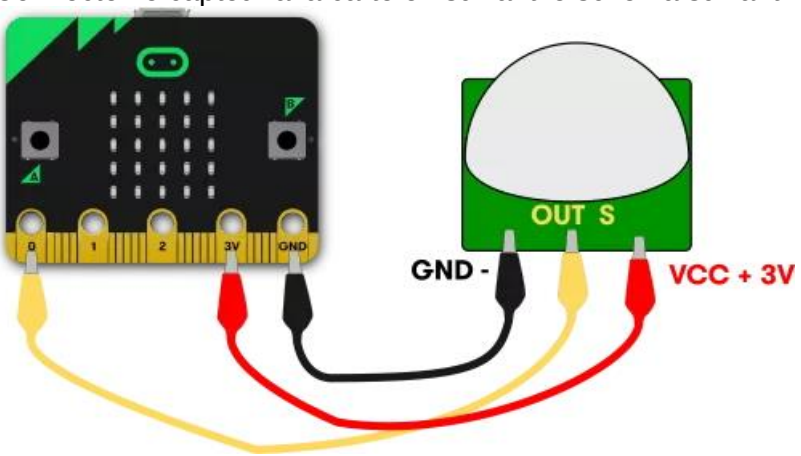
Exercice 4 :
Améliorer le programme précédent pour que l'appel SOS se lance à l'appui sur le bouton A et s'arrête à l'appui sur le bouton B

IV. Y-a-t-il quelqu'un ici ?

On remplace la DEL par un capteur de présence (SR602). C'est un capteur infrarouge pyroélectrique pour détecter un mouvement à une distance de 0 à 3,5m.

A. Connecter le capteur

Connecter le capteur à la carte en suivant le schéma suivant.



B. Détection

Entrée : mesure le signal sur une broche
`pin2.read_digital()` renvoie 0 ou 1 selon la tension mesurée sur la broche 2

```
from microbit import *
while True:
    print(pin0.read_digital())
    sleep(500)
```

Recopier le programme ci-dessus et télécharger-le sur la carte.
Pour obtenir l'affichage, cliquer sur le bouton « REPL » abréviation de « *Read-Evaluate-Print-Loop* »
Mais cliquer sur REPL crée une interruption clavier . Pour y remédier, il faut faire Ctrl+D (soft reboot) pour redémarrer le programme.

Vérifier que le capteur fonctionne correctement.

C. Détection/Action

Exercice 5 :
Créer un programme qui affiche une image gaie si le détecteur capte un mouvement et une image triste sinon.

D. Alarme de présence connectée

Il y a un émetteur/récepteur radio sur la carte Micro:Bit. Il permet à deux cartes de communiquer à distance. Plusieurs commandes sont disponibles :

- `radio.on()` : allume la radio. À écrire obligatoirement avant toute utilisation de la radio
- `radio.off()` : éteint la radio
- `radio.send(message)` : envoie la chaîne de caractères (*texte*) "message"
- `radio.receive()` : reçoit le prochain message à être diffusé sur le canal de réception
- `radio.config(channel=7)` : configure le canal radio sur lequel vous allez émettre et recevoir (comme les *Talkie-Walkies* !). Les canaux disponibles vont de 0 à 83 (celui par défaut étant le numéro 7)

1. Former un binôme avec un autre élève et choisir un canal radio commun (compris entre 0 et 83). L'un des membres sera « émetteur » du message radio et l'autre « récepteur ».
2. Recopier les parties du programmes suivant la répartition des rôles.
3. Flasher ce programme sur la carte. Tester-le.

```
Commun aux 2 :
.............
from microbit import *
import radio

radio.on()
radio.config(channel=19)  # Choisir un n° de channel
radio.config(power=7)     # Règle la force du signal


Receveur :
..........
msg_recu = None
while msg_recu is None:
    msg_recu = radio.receive()
    sleep(500)
display.show(msg_recu)
radio.off()

Emetteur :
..........
message = "Hello NSI !"
continuer = True
while continuer:
    if button_b.was_pressed():
        radio.send(message)
    sleep(500)
radio.off()
```

Exercice 6 :

L'objectif est de simuler un capteur de présence connecté.

L'un des membres du binôme joue le rôle du détecteur de présence qui émet une alerte, l'autre membre du binôme jour le rôle du receveur de message d'alerte sur son smartphone.
Créer un programme qui :

- pour l'émetteur affiche un cœur si une présence est détectée et envoie le message 'mouvement', sinon il n'affiche rien et envoie 'RAS'.
- pour le récepteur affiche une image « STICKFIGURE » si le message 'mouvement' est reçu et n'affiche rien sinon.

NB : les différentes méthodes utilisées dans ce TP sont répertoriées sur le site : https://nsirennes.fr/os-archi/bbc-microbit/