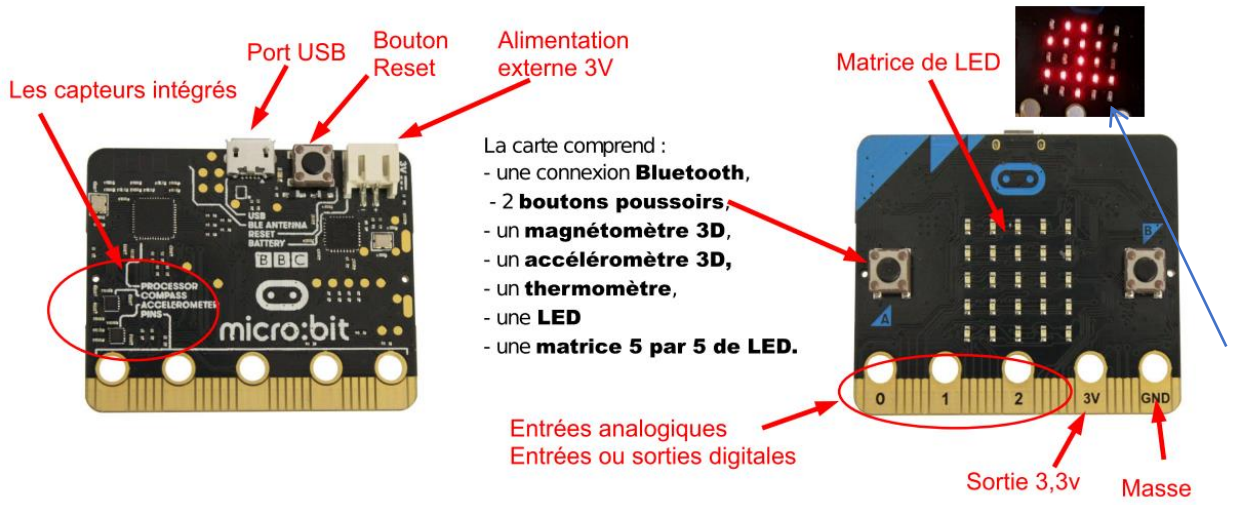
Objectif : illustrer les opérateurs logiques par leur mise en œuvre dans un fonctionnement électronique

1. Prise en main de la carte micro:bit

1.1. Présentation

La carte micro:bit est une carte électronique (nano-ordinateur) créée par la BBC en 2016 pour promouvoir l'apprentissage du codage auprès des élèves. C'est une carte microcontrôleur, programmable, ayant des capteurs et actionneurs intégrés. Elle est plus puissante que la carte Arduino Uno et se programme en Python grâce au logiciel Mu.



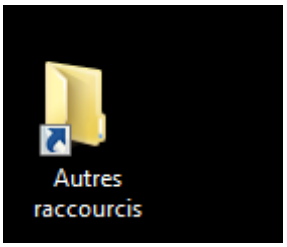
La carte peut fonctionner de manière autonome ou elle peut rester connectée en USB à un ordinateur. Elle peut alimenter des capteurs en 3,3V. Quand on la branche à un ordinateur, elle est détectée comme une carte SD ou une clé USB : il n'y a donc pas de drivers à installer (sous win10) et il suffit simplement de déposer le micro-programme (fichier .hex) dans sa mémoire. La carte exécute ensuite ce programme.

La carte peut-être programmée dans un langage dérivé de Python, mais très proche : le Micropython. Toutefois, la bibliothèque matplotlib et certains modules comme numpy ne sont pas fonctionnels avec la carte Micro:bit.

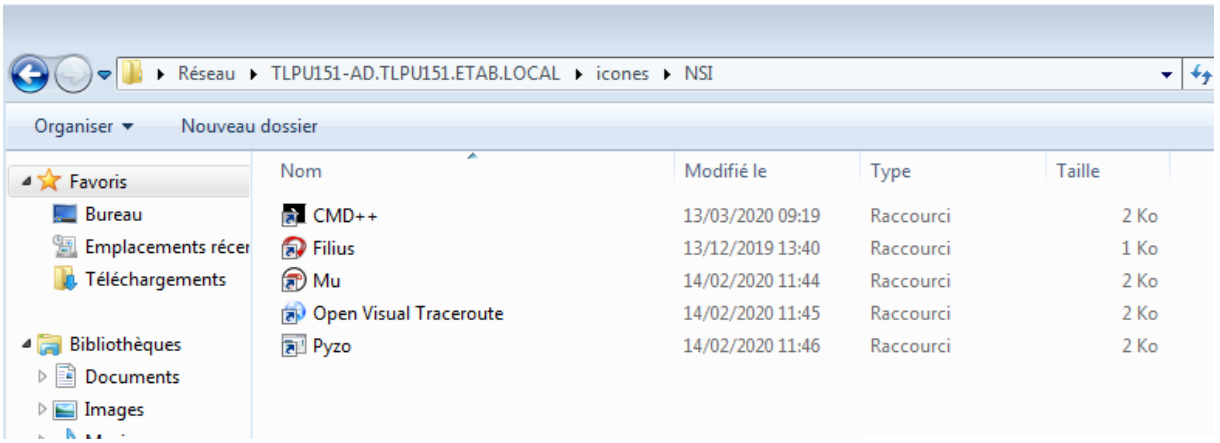
1.2. Utilisation de MU editor

MU editor (https://codewith.mu/en/tutorials/1.0/microbit) est un logiciel permettant de déposer directement le microprogramme sur la carte, sans avoir à passer par l'étape manuelle de dépôt du fichier .HEX et il permet également de recevoir et d'envoyer des données en temps réel à la carte (on appelle cela la console).

1) Au lycée sur le bureau dans le répertoire



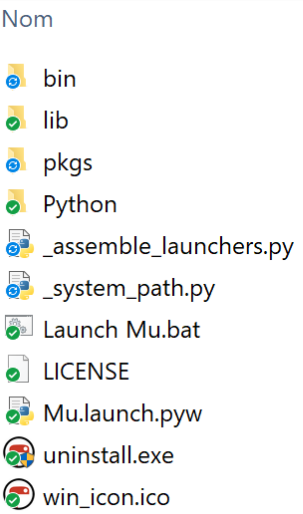
Puis ouvrir /NSI et choisir Mu



2) Au lycée récupérer sur le répertoire la version portable de MU :

portamu_1.0.2_win64

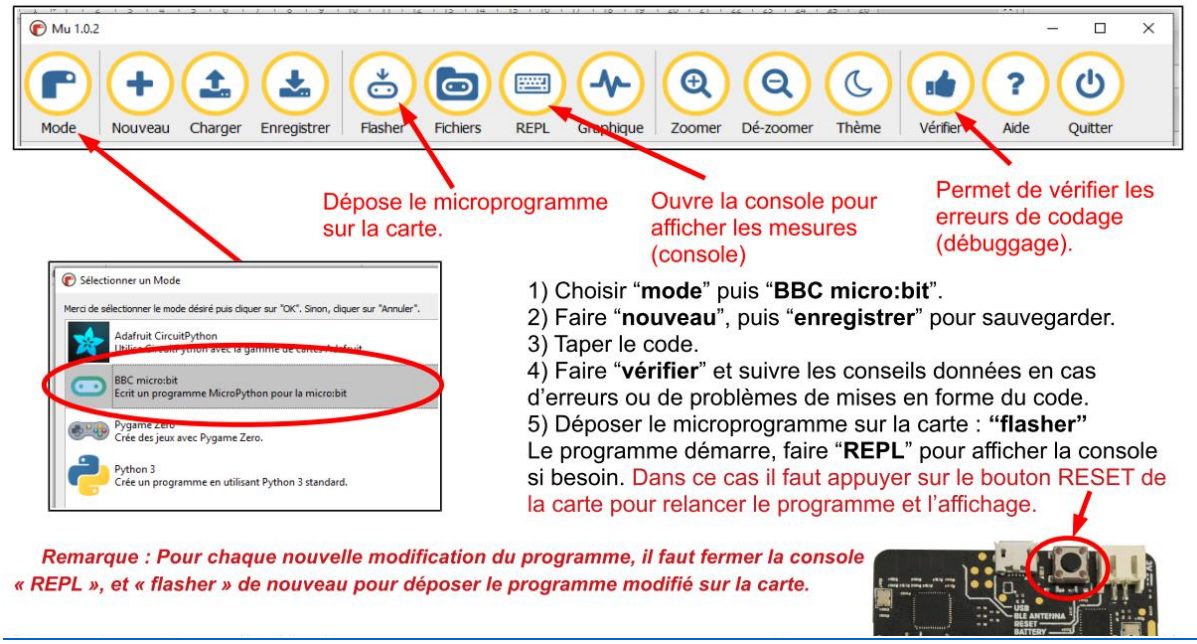
Puis cliquer sur : Launch Mu.bat



3) Installation : hors lycée

Aller sur le site https://codewith.mu/ et suivre les instructions pour l'ins
(Pour les versions inférieures à windows10, il faut également installer le
https://os.mbed.com/docs/mbed-os/v5.7/tutorials/windows-serial-drive……,
Les principales commandes « micropython » pour micro:bit se trouvent en ligne, ne pas hésiter à consulter : https://microbit-micropython.readthedocs.io/en/v1.0.1/index.html Les commandes Python génériques ne sont pas non plus détaillées. https://docs.python.org/fr/3/index.html

4) Utilisation du logiciel



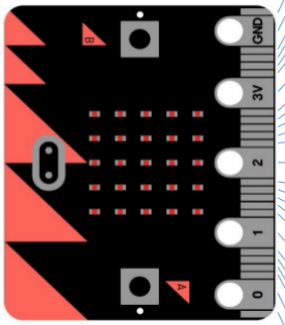
1) Choisir "**mode**" puis "**BBC micro:bit**".
2) Faire "**nouveau**", puis "**enregistrer**" pour sauvegarder.
3) Taper le code.
4) Faire "**vérifier**" et suivre les conseils données en cas d'erreurs ou de problèmes de mises en forme du code.
5) Déposer le microprogramme sur la carte : **"flasher"**
Le programme démarre, faire "**REPL**" pour afficher la console si besoin. Dans ce cas il faut appuyer sur le bouton RESET de la carte pour relancer le programme et l'affichage.

Remarque : Pour chaque nouvelle modification du programme, il faut fermer la console « REPL », et « flasher » de nouveau pour déposer le programme modifié sur la carte.

1.3. Brochage de la carte Micro:bit

Edge Connector Pinout

Note: A number of these pins may not be accessible in all editors.



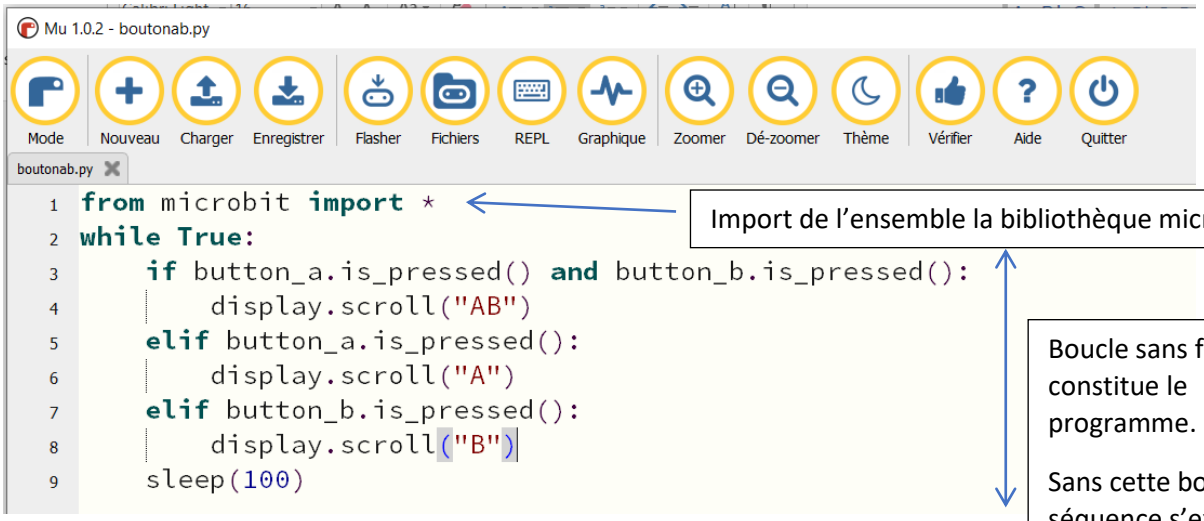
0V
Special function pin
3V
Digital input / output
Analogue input / digital IO
Digital input (shared with a button)
Digital output (shared with LED matrix)

Breakout PCB Ref (if applicable)	Name	Description
22	0V	0V / ground
0V	0V	0V / ground
21	0V	0V / ground
20	SDA	Serial data pin connected to the magnetometer & accelerometer
19	SCL	Serial clock pin connected to the magnetometer & accelerometer
18	3V	3V / positive supply
3V	3V	3V / positive supply
17	3V	3V / positive supply
16	DIO	General purpose digital IO **(P16 in editors)**
15	MOSI	Serial connection - Master Output / Slave Input
14	MISO	Serial connection - Master Input / Slave Output
13	SCK	Serial connection - Clock
2	PAD2	General purpose digital / analogue IO **(P2 in editors)**
12	DIO	General purpose digital IO **(P12 in editors)**
11	BTN_B	Button B – Normally high, going low on press **(Button B in editors)**
10	COL3	Column 3 on the LED matrix
9	COL7	Column 7 on the LED matrix
8	DIO	General purpose digital IO **(P8 in in editors)**
1	PAD1	General purpose digital / analogue IO **(P1 in editors)**
7	COL8	Column 8 on the LED matrix
6	COL9	Column 9 on the LED matrix
5	BTN_A	Button A – Normally high, going low on press **(Button A in editors)**
4	COL2	Column 2 on the LED matrix
0	PAD0	General purpose digital / analogue IO **(P0 in editors)**
3	COL1	Column 1 on the LED matrix

2. Un premier programme : essai des boutons et affichage sur écran

Dans un premier temps nous allons utiliser uniquement les entrées sorties disponibles sur la carte elle-même.

Compléter le tableau pour identifier les entrés/sorties numéros de broches



Import de l'ensemble la bibliothèque microbit

Boucle sans fin qui constitue le programme.

Sans cette boucle la séquence s'exécute une seule fois !

3. Visualisation des fonctions logiques OUI, NON, OU et ET

Pour chaque fonction écrivez un programme qui permet en fonction de l'appui sur un ou deux boutons de faire défiler le cas échéant « OUI »,« NON » , « OU » et « ET » mais rien quand la fonction n'est pas valide.

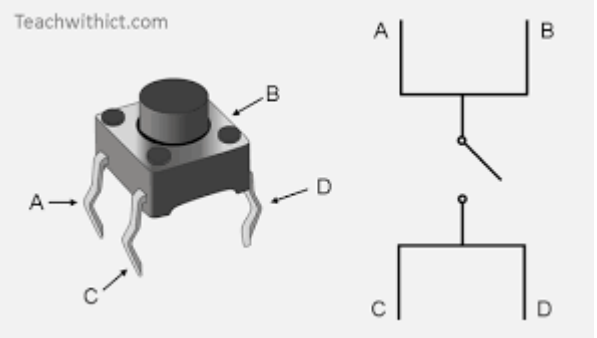
Compléter les 4 tableaux du document réponses pour chacune de ces fonctions.

4. Utilisation des entrées-sorties physiques (les numéros de broches) avec les éléments électroniques extérieurs

4.1. Quelques informations sur l'électronique

Les entrées logiques :

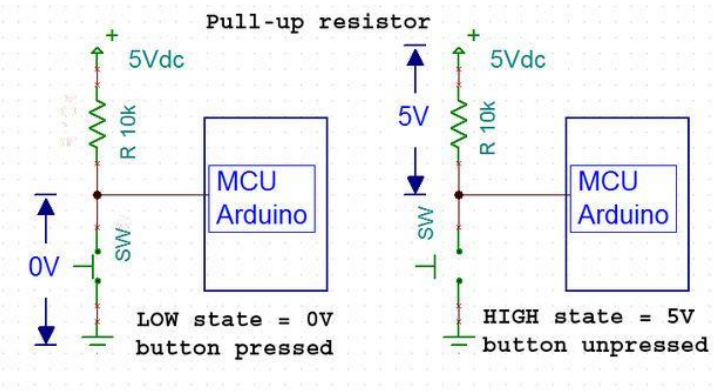
Les boutons/switchs



Bouton A et B brochage Pull-up interne à la carte Mico:bit (voir brochage ci-dessus)

Avec button_a

```
while True:
    if button_a.is_pressed():
            pin0.write_digital(1)
            display.show('Y')
    else:
            pin0.write_digital(0)
            display.show('N')
```

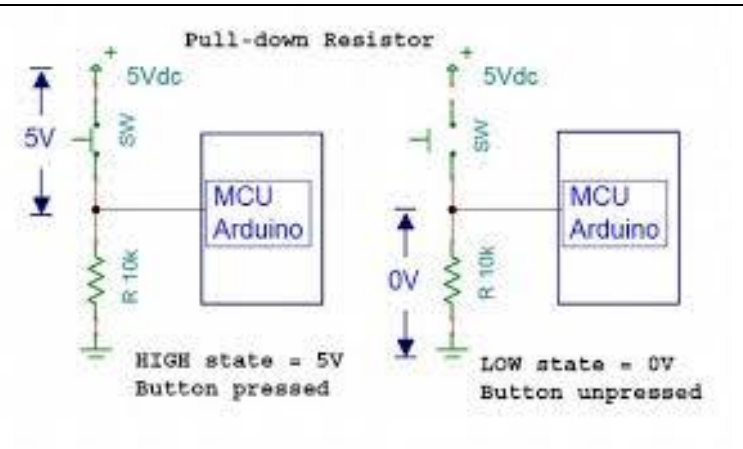


Bouton sur le connecteur

Avec numéro de pin (button_a à la pin 1)

```
from microbit import *

while True:
    if pin1.read_digital() == 1 :
        pin0.write_digital(1)
        display.show('Y')
    else:
        pin0.write_digital(0)
        display.show('N')
```

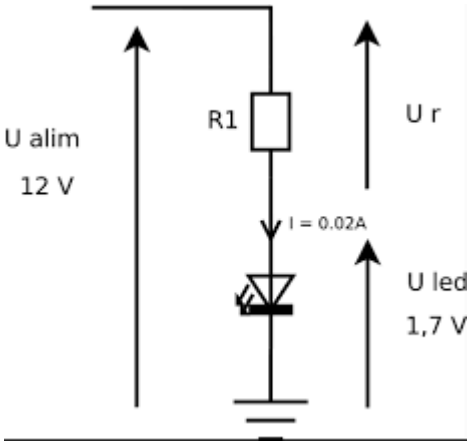


Les sorties logiques : Diode électroluminescente (abrégé en **LED**, de l'anglais : *light-emitting diode*, ou DEL en français) :

Gros-plan d'une diode électroluminescente.

L'anode et la cathode d'une LED. Les signes indiquent la polarisation (courant conventionnel) lorsque la diode est utilisée en sens direct.

La LED nécessite une résistance de protection pour limiter le courant qui la traverse à sa valeur nominale.



R = (12V-1.7V)/0.02A = 515 W

Soit 560 W en E12

- Compléter le tableau du document réponse : table d'adressage des entrées/sorties de la carte MicrotBit

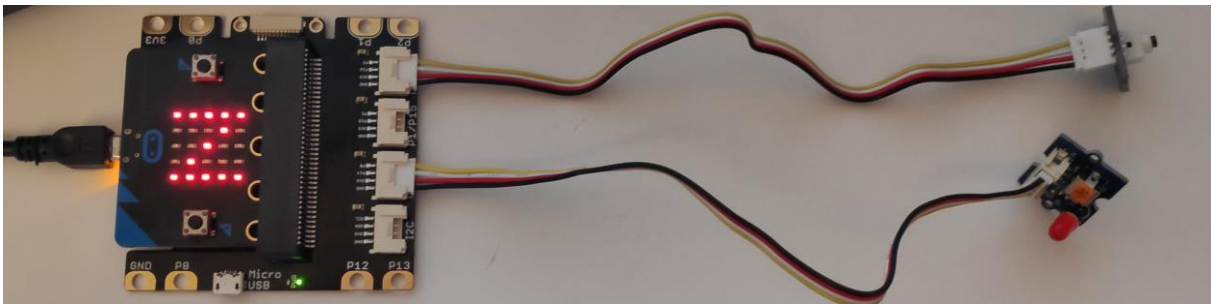
5. Ouverture d'un coffre-fort.

On utilisera un nouveau Bouton Poussoirs « c » sur la broche 2. La LED sur la broche 0 simule le fonctionnement tout ou rien de la serrure soit fermée « 0 » soit ouverte « 1 ».

Fonctionnement – cahier des charges

3 clés différentes peuvent ouvrir le coffre-fort, mais il doit s'ouvrir seulement si on introduit au moins 2 des 3 clés. Les 3 clés sont désignées a, b, c.

- Compléter le document réponse
- Faites les branchements sur Shield Grove pour micro:bit
- Réaliser le programme adéquat



Pour les logigrammes vous pouvez vous aider du simulateur en ligne : https://logic.ly/demo/

Une documentation succincte en français est à l'adresse http://christianpc.fr/simulateur-de-portes-logiques/

Document réponses

2 Un premier programme

Désignation	Entrée/Sortie	Information/commande	Logique, analogique, numérique	Numéro de broche(s)
button_a				
button_b				
Matrice de l'écran				

3 Visualisation des fonctions logiques OUI, NON, OU et ET

- Fonction « OUI »

<table>
<tr><td colspan="2">Logigramme</td><td></td><td colspan="2">Equation :</td></tr>
<tr><td colspan="3"></td><td colspan="2">Table de vérité</td></tr>
<tr><td rowspan="3">Solution Programmée</td><td>Boucle sans fin</td><td></td><td>button_a</td><td>OUI</td></tr>
<tr><td rowspan="2">Code pour la structure du test</td><td rowspan="2"></td><td>0</td><td></td></tr>
<tr><td>1</td><td></td></tr>
</table>

- Fonction « NON »

<table>
<tr><td colspan="2">Logigramme</td><td></td><td colspan="2">Equation :</td></tr>
<tr><td colspan="3"></td><td colspan="2">Table de vérité</td></tr>
<tr><td rowspan="3">Solution Programmée</td><td>Boucle sans fin</td><td></td><td>button_a</td><td>NON</td></tr>
<tr><td rowspan="2">Code pour la structure du test</td><td rowspan="2"></td><td>0</td><td></td></tr>
<tr><td>1</td><td></td></tr>
</table>

- Fonction « OU »

Logigramme		Equation :		
		Table de vérité		
Boucle sans fin		button_a	button_b	OU
Code pour la structure du test		0	0	
		0	1	
		1	0	
		1	0	

- Fonction « ET »

Logigramme		Equation :		
		Table de vérité		
Boucle sans fin		button_a	button_b	OU
Code pour la structure du test		0	0	
		0	1	
		1	0	
		1	0	

4 Utilisation des entrées-sorties physiques

Interprétez le normally high du brochage	A l'état haut sans action
Instruction commande logique broche en sortie	
Instruction commande logique broche en entrée	

5 Ouverture d'un coffre-fort.

5.1 Remplissez la table de vérité

« a »	« b »	« c »	SERRURE
0	**0**	**0**	
0	**0**	**1**	
0	**1**	**0**	
0	**1**	**1**	
1	**0**	**0**	
1	**0**	**1**	
1	**1**	**0**	
1	**1**	**1**	

5.2 Pour chaque combinaison vraie écrivez l'équation logique correspondante

5.3 L'équation totale est la somme logique des combinaisons vraies. Ecrivez cette équation logique

5.4 Programmer ce fonctionnement avec les deux boutons a et b ainsi qu'avec le bouton du module Grove. Faites constater le bon fonctionnement.

5.5 Si vous avez le temps produisez le logigramme complet de cette commande.