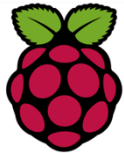




Atelier Raspi

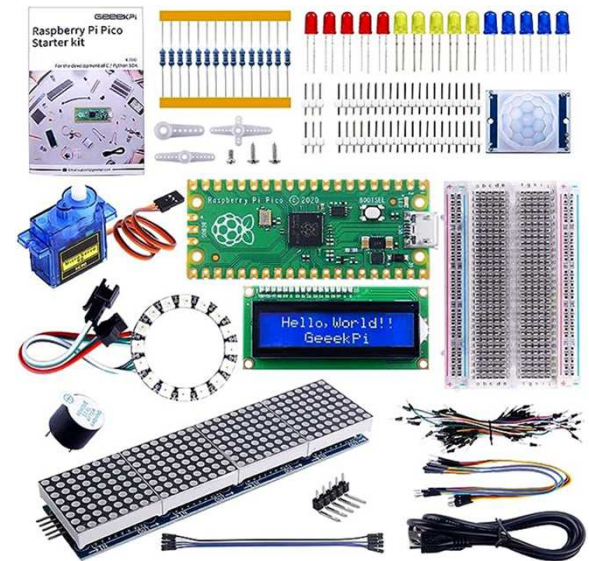
Atelier N°3 Les signaux électriques et les Entrées TOR (Tout ou Rien)



Logo du Raspberry Pico



Logo du MicroPython



L'atelier a pour valeurs, le partage, l'aide, la formation, le faire et construire ensemble à partir de l'expérience des participants

Atelier N°3 Les signaux électriques – les entrées TOR

Etape 1 Les signaux électriques dans le temps

Etape 2 L'oscilloscope, fréquence, période et longueur d'onde

Etape 3 Les signaux électriques sur le circuit LED avec pile

Etape 4 Le rayonnement Infra Rouge IR

Etape 5 Le capteur IR de mouvement

Etape 6 Rajout du câblage du BP, du capteur IR et de l'oscilloscope

Etape 7 Proposition de programme pour le chenillard 4 LEDs

Etape 8 Détection des Entrées TOR BP et capteur IR

Etape 9 Câblage du sapin

Etape 10 Utilisation de la librairie Neopixel

Etape 11 Les fonctions suite, variable globale, init et return

Etape 12 Les fonctions du module time

Etape 13 La structure du programme de commande de la bande LED

Etape 14 Glossaire électronique

Etape 15 Glossaire informatique1

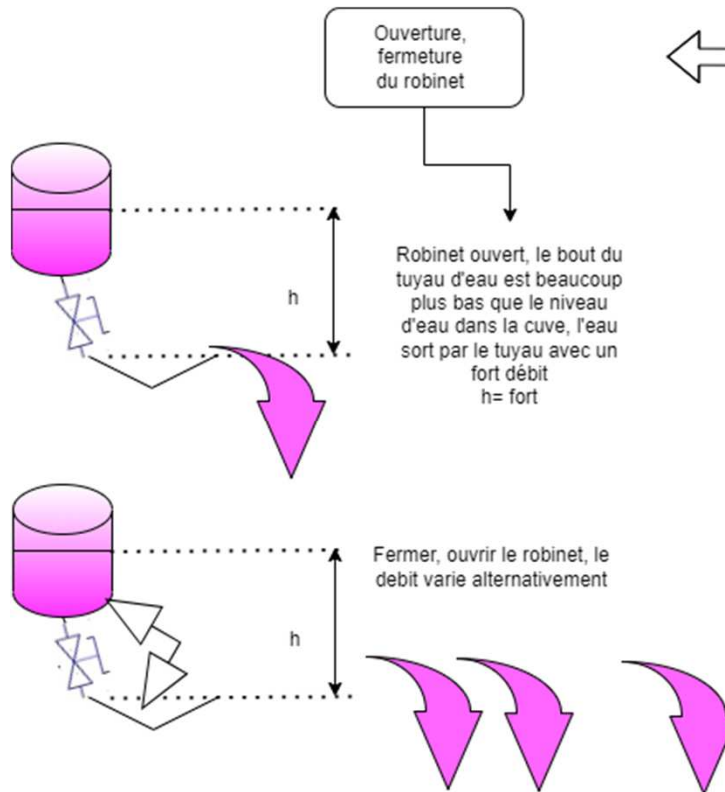
Etape 16 Glossaire informatique2

A3E1 Les signaux électriques dans le temps

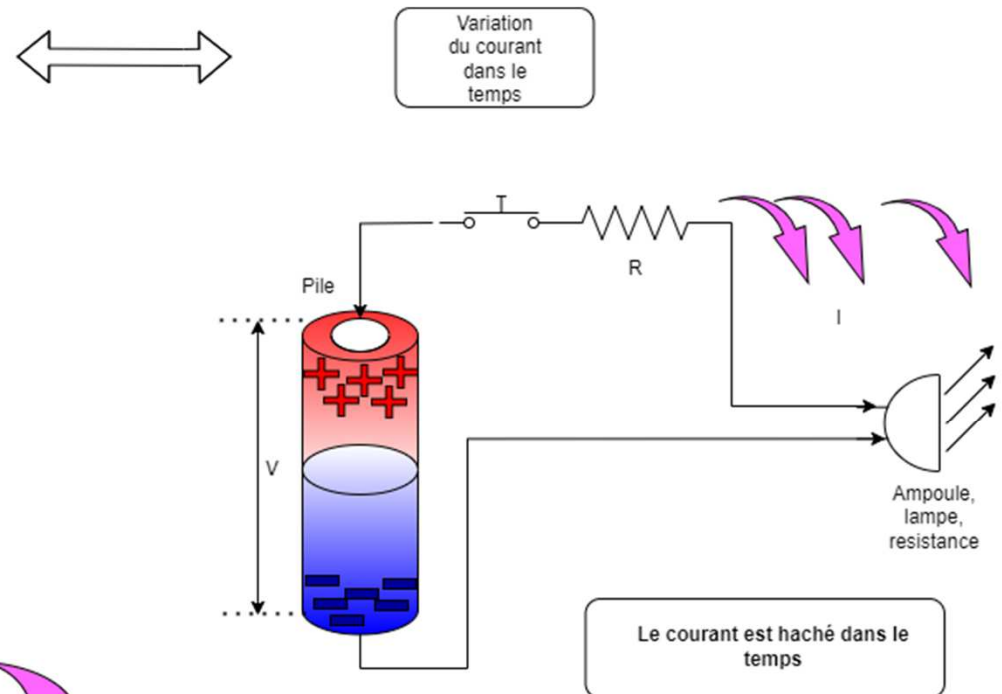
Retour sur la bouteille d'eau avec le tuyau

* positionner le tuyau en bas pour avoir le débit le plus fort

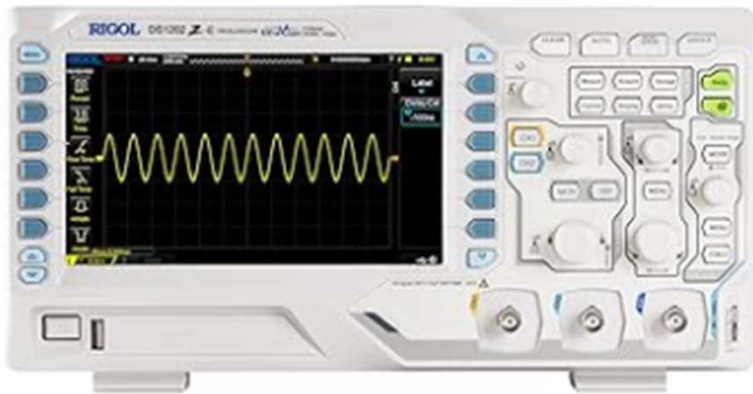
* ouvrir puis fermer le robinet pour avoir un débit d'eau haché



* On suppose que la cuve d'eau est alimentée en permanence



A3E2 L'oscilloscope



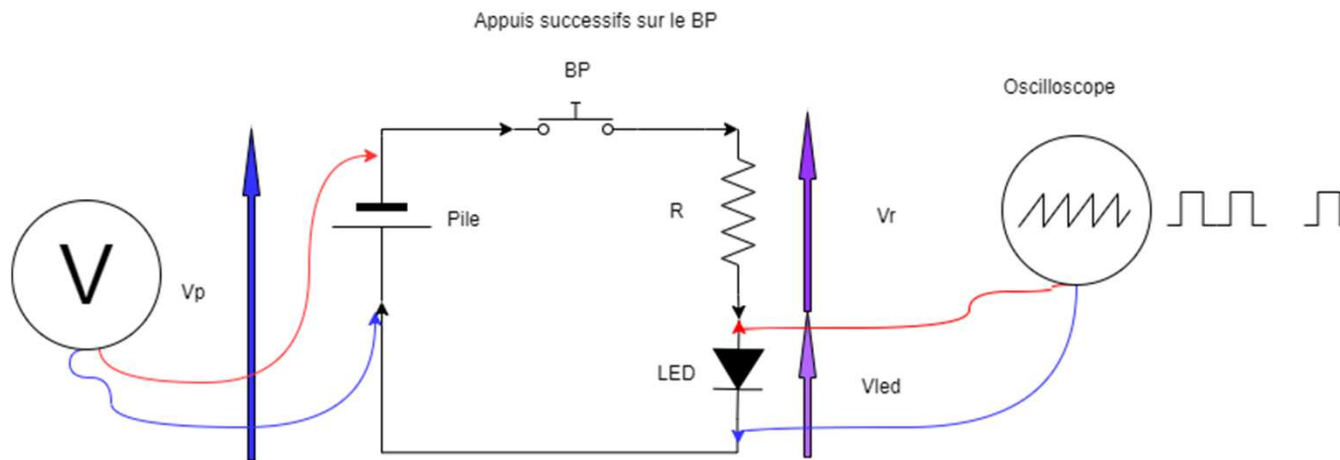
Cet appareil permet de visualiser la forme des signaux et de mesurer des différences de potentiel.

Il se connecte sur le circuit électrique comme le multimètre
C'est-à-dire que l'on vient prendre la différence de potentiel en 2 points du circuit électrique.

Si l'on appuie régulièrement sur le bouton BP ou fermer et ouvrir régulièrement le robinet, on va avoir un signal périodique.
c'est-à-dire que l'on va garder toujours les mêmes temps d'ouverture et de fermetures du robinet ou appui et relâche sur le BP.
Si j'appuie toutes les secondes puis relâche, j'aurai une période de 1 s plus le temps de relâcher.

<https://youtu.be/kK4wkk6lT9U>

A3E3 Signaux sur le circuit électrique « LED avec pile »



* Mesure de la tension V_{pile} avec le voltmètre

- La LED s'éclaire lorsque le BP est appuyé, le courant passe dans la LED
- L'oscilloscope montre la tension aux bornes de la LED
- Lors de la succession des appuis sur le BP on voit la tension aux bornes de la LED varier

A3E4 Le rayonnement infra rouge IR

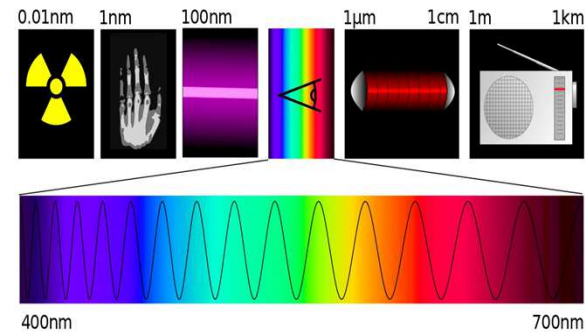
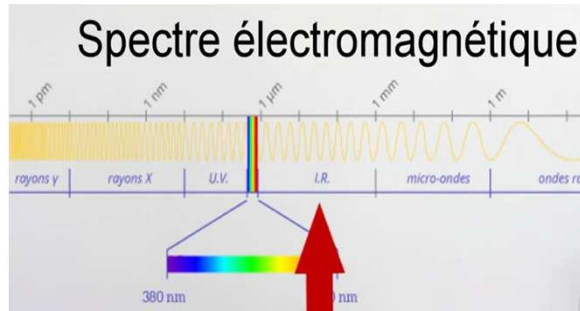


Photo
infra rouge



Télécommande
avec LED
infra rouge

Le rayonnement **infrarouge (IR)** est un rayonnement électromagnétique de longueur d'onde supérieure à celle du spectre visible mais plus courte que celle des micro-ondes ou des ondes radios.

Cette gamme de longueurs d'onde varie dans le vide de 700 nm à 0,1 mm

A3E5 Le capteur de mouvement infra rouge IR



Le capteur IR permet de détecter un mouvement par analyse des rayonnement infra rouge émis par le corps humain. Sa sortie change en fonction du dépassement d'un seuil réglé à l'avance. Il a donc une sortie TOR, Basse (0 volt) ou haute (3,3V) en fonction de sa détection.

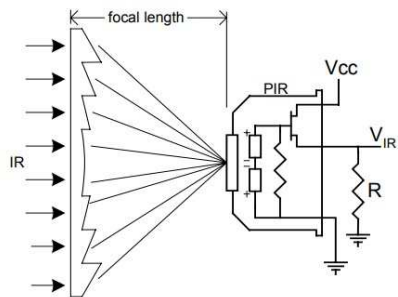
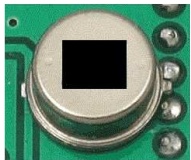
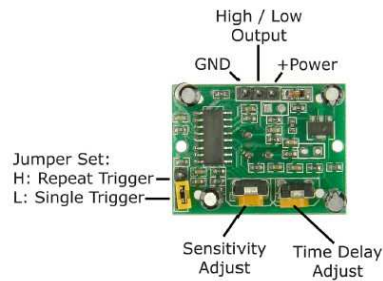


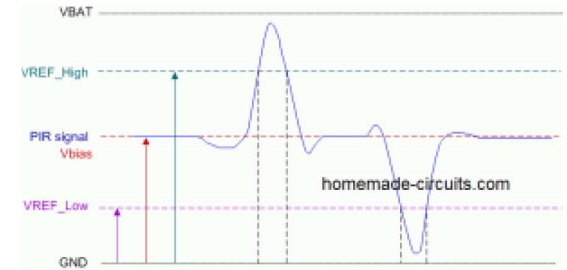
Figure 7: PIR Detector with Fresnel Lens



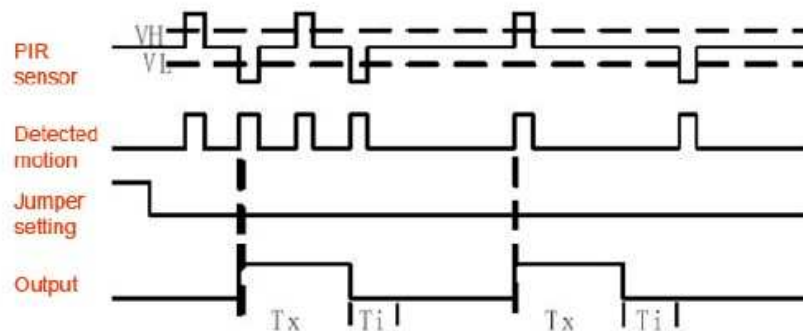
Capteur IR



Différents réglages de la sensibilité et du temps de réaction.



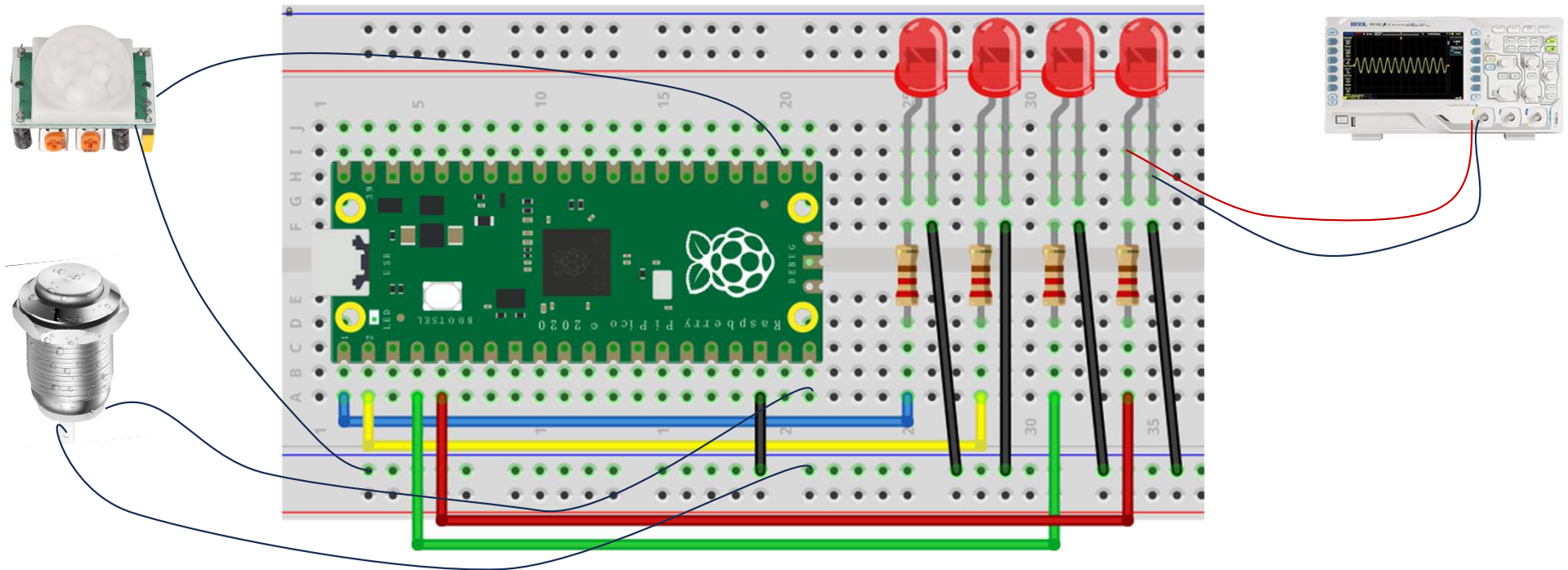
Le signal issu du capteur, correspond à l'image simplifiée vue par chacune des zones sensibles du capteurs lorsqu'une personne passe devant.



Output est le signal de sortie TOR de la carte, on va détecter lorsqu'il sera à 1 ou avec une différence de potentiel « haut » soit 3,3v ou 5v.

A3E6 Rajout du câblage des Entrées et de l'oscilloscope sur la plaquette

* Rajout du BP pour lancer la séquence d'allumage des LED

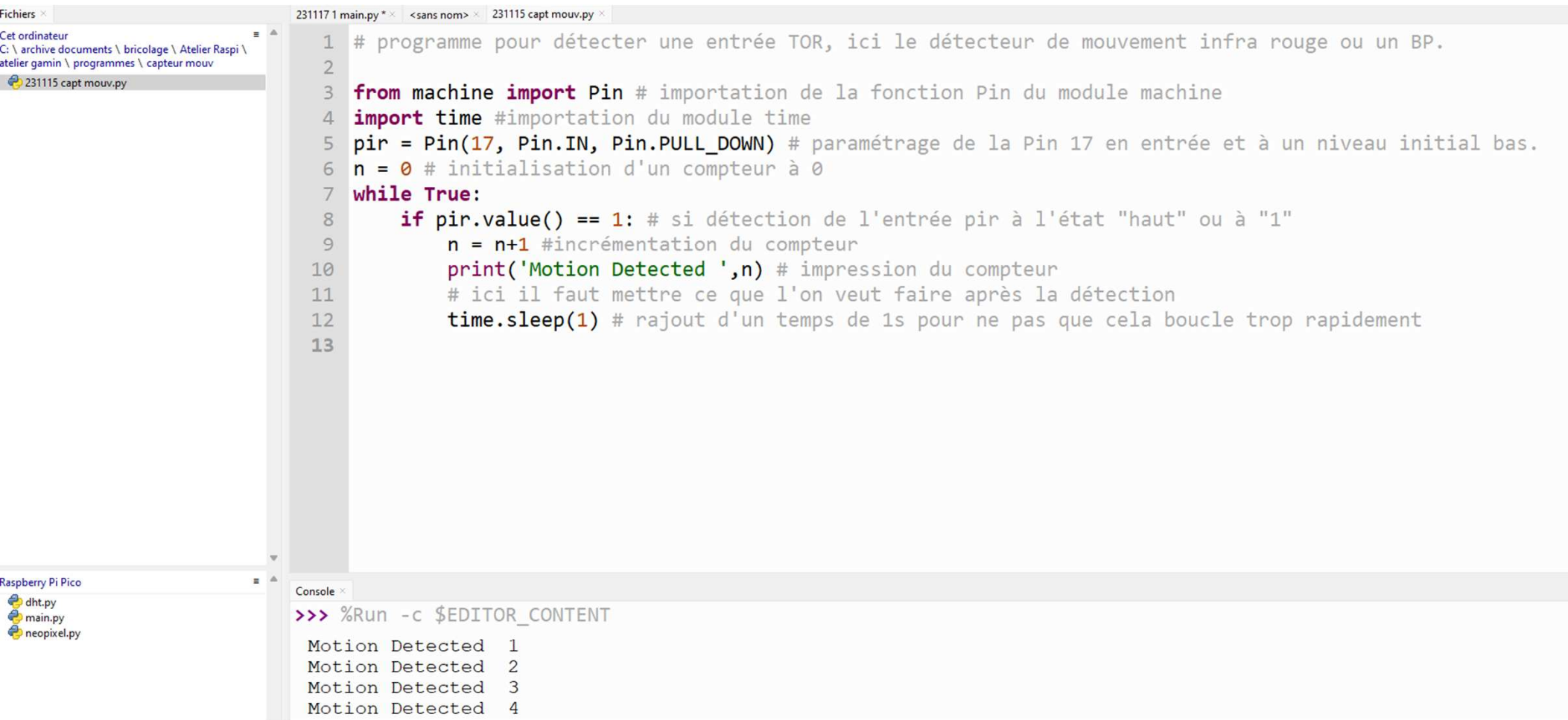


A3E7 Proposition de programme pour le chenillard

```
main.py  diagram.json  ▼

1  # Exemple chenillard list simple
2
3  #import des modules
4  import time
5  from machine import Pin
6
7  # initialisation de I/O
8  led_0 = Pin(0, Pin.OUT)
9  led_1 = Pin(1, Pin.OUT)
10 led_2 = Pin(2, Pin.OUT)
11 led_3 = Pin(3, Pin.OUT)
12
13 # declaration d'un chenillard sous forme d'une liste de led
14 chenillard = [led_0,led_1,led_2,led_3]
15
16 # éteint toutes les led
17 for led in chenillard:
18     led.value(0)
19
20 # boucle principale
21 while True:
22     # pour chaque led de la liste on inverse l'état et on attend 1 second
23     for led in chenillard:
24         led.toggle()
25         time.sleep(1)
26
```

A3E8 Détection des entrées TOR



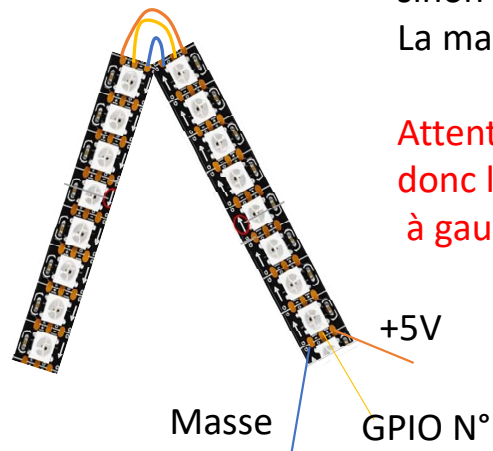
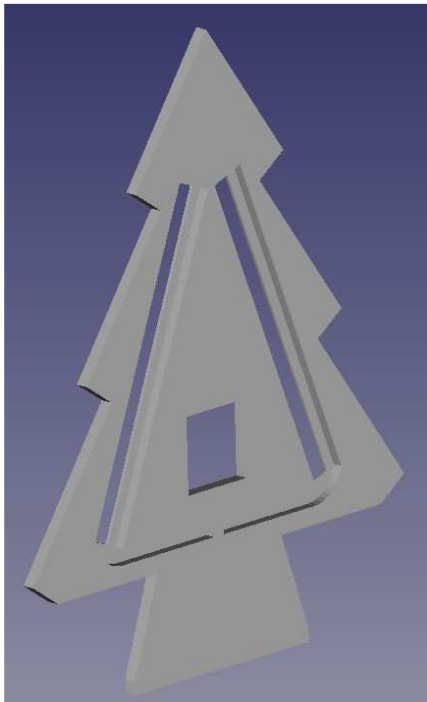
```
Fichiers x 231117 1 main.py * x <sans nom> x 231115 capt mouv.py x
Cet ordinateur
C:\archive documents\bricolage\Atelier Raspi\
atelier gamin\programmes\capteur mouv
231115 capt mouv.py

1 # programme pour détecter une entrée TOR, ici le détecteur de mouvement infra rouge ou un BP.
2
3 from machine import Pin # importation de la fonction Pin du module machine
4 import time #importation du module time
5 pir = Pin(17, Pin.IN, Pin.PULL_DOWN) # paramétrage de la Pin 17 en entrée et à un niveau initial bas.
6 n = 0 # initialisation d'un compteur à 0
7 while True:
8     if pir.value() == 1: # si détection de l'entrée pir à l'état "haut" ou à "1"
9         n = n+1 #incrémentatation du compteur
10        print('Motion Detected ',n) # impression du compteur
11        # ici il faut mettre ce que l'on veut faire après la détection
12        time.sleep(1) # rajout d'un temps de 1s pour ne pas que cela boucle trop rapidement
13

Raspberry Pi Pico
dht.py
main.py
neopixel.py

Console x
>>> %Run -c $EDITOR_CONTENT
Motion Detected 1
Motion Detected 2
Motion Detected 3
Motion Detected 4
```

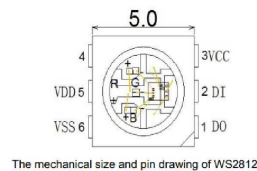
A3E9 Câblage des LEDs du sapin



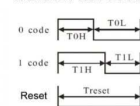
Le 5V peut être une alimentation extérieure
sinon câbler sur le Vbus pin 40

La masse doit être commune avec la carte Pico

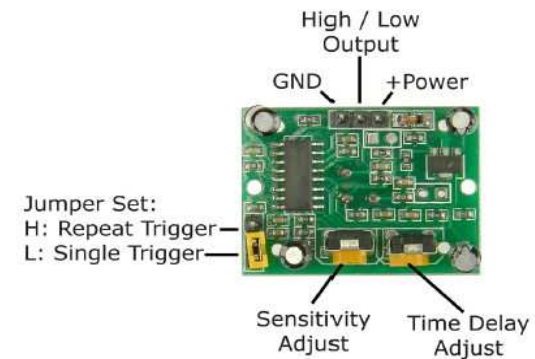
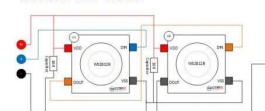
Attention aux branchements, on va voir les fils de derrière
donc la masse sera à droite (en bleu en général) et le +5V
à gauche, la couleur des fils n'a pas forcément de signification.



WS2812 PROTOCOL



WS2812 LED CHAIN



A3E10 Utilisation de la librairie Néopixel.

Fonctions utilisées:

pixels = Neopixel(i_nb_leds, 0, 0, "RGB") -> dans la variable pixels, on va utiliser la classe Néopixel avec les paramètres suivants: le nombre de LEDS à piloter (i_nb_leds) qui est une variable de notre programme, le N° de la machine d'état, le N° de la Pin utilisée, le mode de couleur RVB.

pixels.set_pixel(i_led, i_value, i_brightness): -> on fait appel à la fonction set_pixel() avec 3 paramètres que sont les variables i_led, i_value et i_brightness.

pixels.fill((i_value, i_brightness): -> on fait appel à la fonction fill() avec un parameter couleur et luminosité
exemple: "rouge avec (255,0,0) et 10 pour 10/255 de luminosité"

pixels.show(): -> on fait appel à la fonction d'afficher les LEDs (éteindre, allumer avec la couleur déjà programmée)

pixels.rotate_left (num_of_pixels): -> Après avoir défini avec pixels.set_pixel() les pixels, cette fonction incrémente vers la gauche de num_of_pixels. Idem pour la droite avec **pixels.rotate_right():**, right est dans le sens de parcours des LEDs.

A3E11 Les fonctions suite, variable globale, return

global ma_variable -> permet d'avoir accès à ma_variable du programme principal dans la def d'une fonction
def ma_fonction():

```
    global ma_variable  
    print( ma_variable)  
    ma_variable += 2  
    return ma_variable
```

Return ma_variable -> permet de donner au reste du programme la nouvelle valeur de ma_variable
ou d'une nouvelle variable si créée dans la fonction

A3E12 Les fonctions du module time

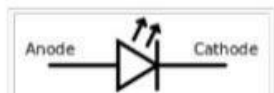
time.time() -> donne le temps présent en ms

time.time_diff(ticks1, ticks2) -> donne la différence entre ticks1 et 2

A3E13 La structure du programme du chenillard bande LED

- Importer les modules nécessaires
 - Machine : pour avoir accès aux IO (Pin)
 - Time : pour pouvoir compté du temps
 - Neopixel
 - ...
- Déclaration
 - Ruban led
 - Capteur de présence
 - ...
- Initialisation
 - Ruban led à un état connu
 - ...
- Boucle principale
 - Si détection
 - Effectuer une séquence « chenillard » sur le ruban led

A2E14 Glossaire électronique



Symbole de la diode LED



Pile



Résistance



BP

A2E15 Glossaire informatique1

Instruction du langage : ne nécessite pas d'importer de module pour les utiliser

print('Bonjour') : affiche Bonjour sur la console du PC

While test : Boucle tant que test est VRAI

import : pour importer des modules contenant des fonctions

import time : permet d'appeler les fonctions relatives au temps

time.sleep(x) : arrête l'exécution du programme pendant x secondes

import machine : module contenant les fonctions permettant d'agir sur les périphériques du pi pico

from : pour simplifier l'écriture on importe certaines fonctions directement dans une variable à l'aide de « from »

from machine import Pin : permet d'accéder à la fonction Pin du module machine uniquement.

ma_led_verte = Pin(n, Pin.OUT) : permet d'indiquer comment on souhaite utiliser un GPIO,
ici on a configuré le p I/O N° n en sortie tout ou rien

ma_led_verte.Toggle() : inverse l'état de l'I/O associé à la variable.

Ici l'IO n passera de l'état haut (+3.3v) à l'état bas (0v) ou inversement

: En début de ligne permet d'écrire un commentaire, ce qui est écrit n'est pas une instruction et donc n'est pas pris en compte par le programme.

Variable $A=A+1$ -> Nouvelle valeur = ancienne valeur+1 , peut s'écrire $A+=1$

print (" texte " , A) , input ("texte" , B)

Opérateurs : +, -, *, /, %, **Données** : int(), float(), str(), bool()=True or False

A2E16 Glossaire informatique2

Liste -> Création d'une liste, `liste_de_noms=['Toto', 'Hubert', 'Jacques', 'Philippe']`
`liste_quelconque = ['Toto', 1, 'Philippe', 4,5,6]`
`print ('Nombre d'articles dans la liste : ', len(liste_quelconque))`
`for items in liste_quelconque:`
 `print (items)` -> parcourt la liste et imprime chaque item de la liste

Dictionnaire-> liste d'éléments sous la forme d'une paire key:valeur.

`N = {'voiture':'4 roues', 'moto':'2 roues'}`

On accède à la valeur d'un key en l'utilisant comme index (clé de recherche):

`print(N['voiture'])` -> '4 roues'

Fonction-> `def nom_de_ma_fonction(args) :`

on peut ensuite l'appeler dans un programme avec: `nom_de_ma_fonction()`

Module-> `from machine import Pin` -> importe la fonction Pin du module machine

`P0 = Pin(0, Pin.OUT)` -> on paramètre le GPIO 0 en sortie

`P0.value(1)` -> on lui donne la valeur 1 ou état « haut »