

LA BIBLIOTHÈQUE GPIO

Configuration des entrées-sorties

Afin d'initialiser une E/S, il suffit de préciser son numéro, fonction de la configuration précédemment choisie, son paramétrage (entrée ou sortie) et éventuellement son état initial (pour une sortie).

```
GPIO.setup(12, GPIO.IN)      # broche 12 est une entree numerique
GPIO.setup(12, GPIO.OUT)     # broche 12 est une sortie numerique
GPIO.setup(12, GPIO.OUT, initial=GPIO.HIGH)  # broche 12 est une sortie
                                              initialement a l'etat haut
```

Lire une entrée numérique

Afin de lire l'état d'une entrée, il suffit de préciser le numéro de l'E/S.

```
GPIO.input(12)
```

LA BIBLIOTHÈQUE GPIO

Changer l'état d'une sortie numérique

2. To set an output high:

```
GPIO.output(12, GPIO.HIGH)
# or
GPIO.output(12, 1)
# or
GPIO.output(12, True)
```

3. To set an output low:

```
GPIO.output(12, GPIO.LOW)
# or
GPIO.output(12, 0)
# or
GPIO.output(12, False)
```

4. To output to several channels at the same time:

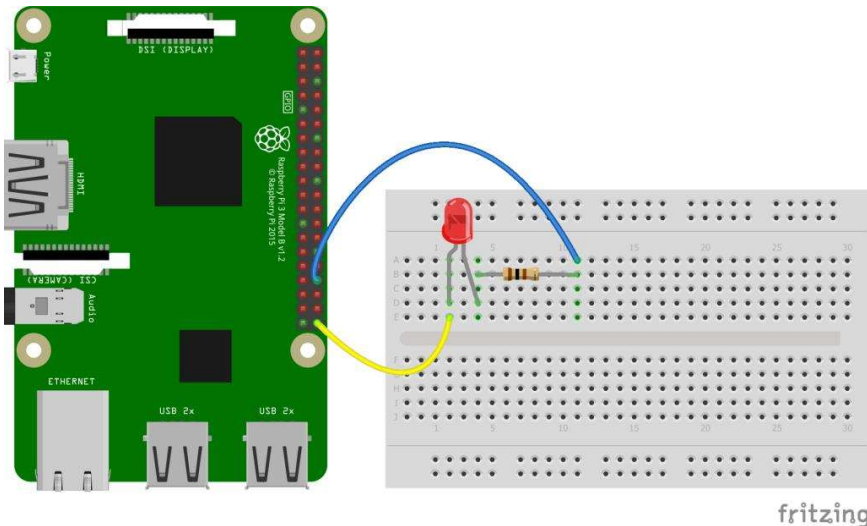
```
chan_list = (11,12)
GPIO.output(chan_list, GPIO.LOW) #
all LOW
GPIO.output(chan_list,
(GPIO.HIGH,GPIO.LOW)) # first LOW,
second HIGH
```

5. Clean up at the end of your program

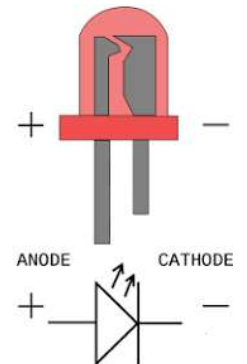
```
GPIO.cleanup()
```

Exercice 1 :

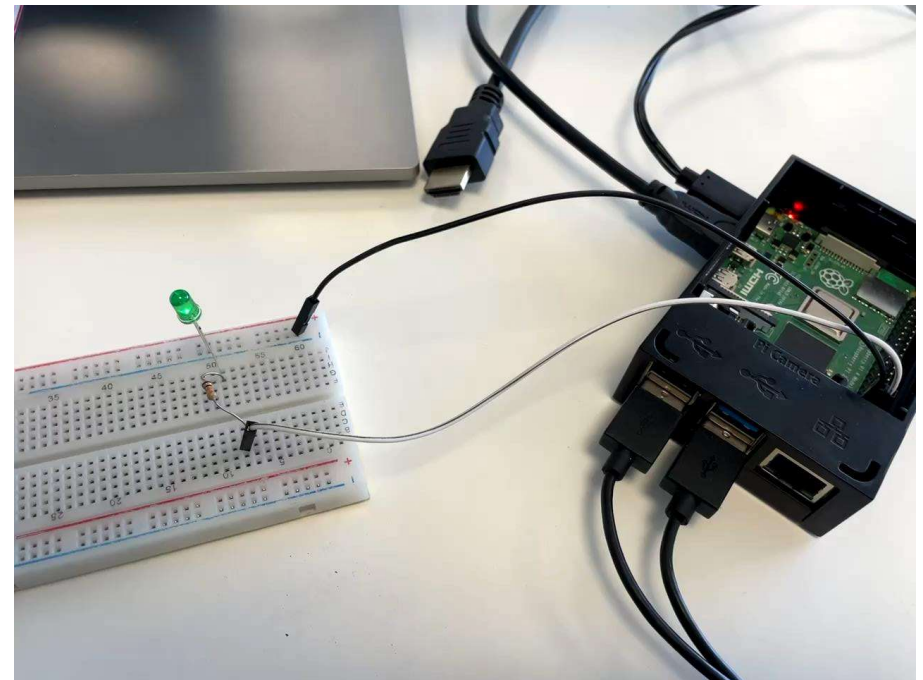
- Ecrivez un programme simple qui fait clignoter une LED avec une période d'une seconde.
- Réalisez le montage et écrivez le programme approprié.



```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.OUT)
while True:
    print("Allume")
    GPIO.output(4,1) #
    Allume la LED
    time.sleep(0.5)
    print("Eteind")
    GPIO.output(4,0)
    time.sleep(0.5)
```

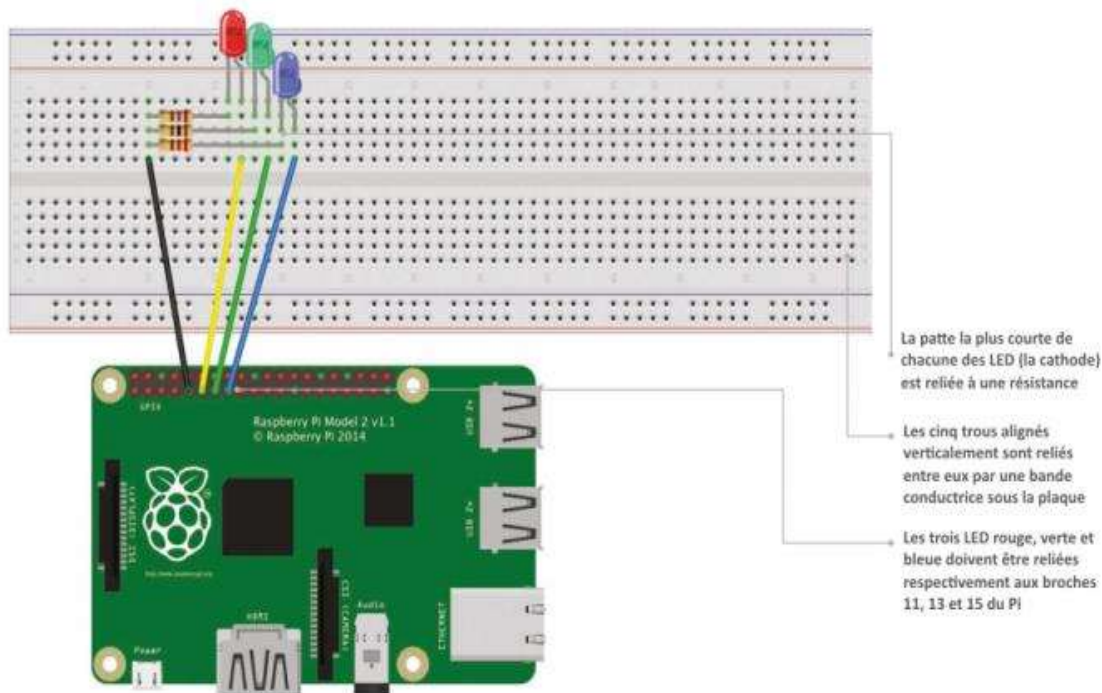


Solution:



Exercice 2:

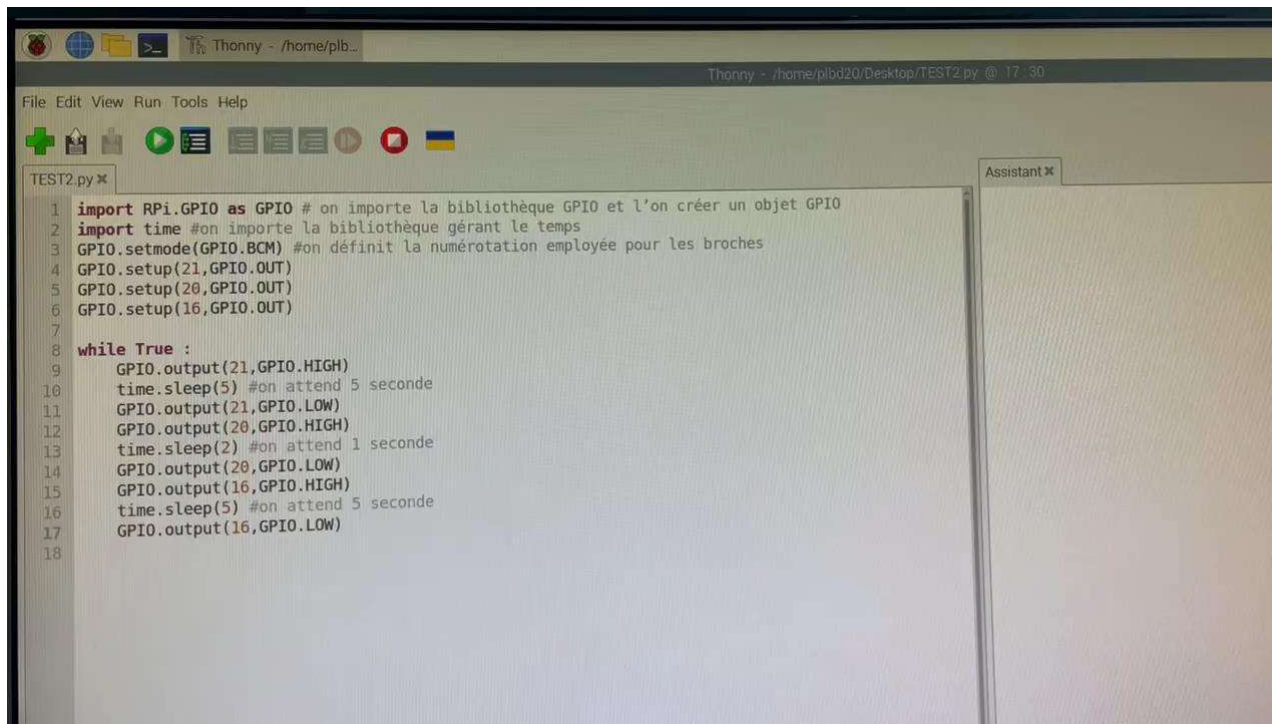
- Ecrire un programme simple qui simule les feux de circulation en utilisant 3 LED Rouge, Vert et Bleu.
- Réaliser le montage et écrire le programme approprié.



```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.OUT)
GPIO.setup(3, GPIO.OUT)
GPIO.setup(2, GPIO.OUT)
```

```
while True:
    print("rouge")
    GPIO.output(2,1)
    time.sleep(5)
    GPIO.output(2,0)
    print("vert")
    GPIO.output(4,1)
    time.sleep(5)
    GPIO.output(4,0)
    print("orange")
    GPIO.output(3,1)
    time.sleep(1)
    GPIO.output(3,0)
```

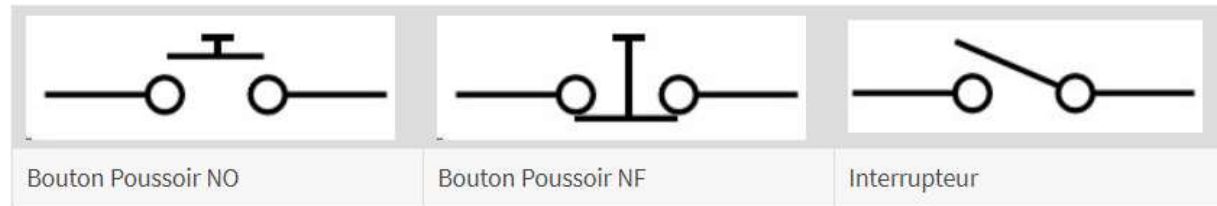
Solution:



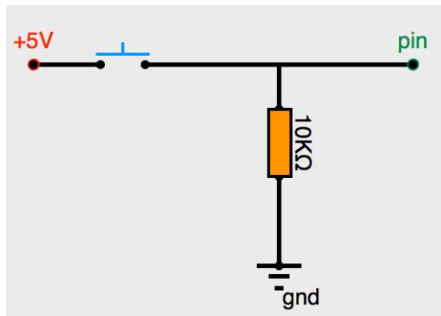
The image shows a screenshot of the Thonny Python IDE. The window title is 'Thonny - /home/plb...'. The menu bar includes 'File', 'Edit', 'View', 'Run', 'Tools', and 'Help'. Below the menu bar is a toolbar with icons for file operations, running, and debugging. The main editor area displays a Python script named 'TEST2.py'. The script imports the RPi.GPIO and time modules, sets up GPIO pins 21, 20, and 16 as outputs, and enters a while loop that toggles the output states of these pins with specific delays. A small 'Assistant' window is visible on the right side of the IDE.

```
1 import RPi.GPIO as GPIO # on importe la bibliothèque GPIO et l'on crée un objet GPIO
2 import time #on importe la bibliothèque gérant le temps
3 GPIO.setmode(GPIO.BCM) #on définit la numérotation employée pour les broches
4 GPIO.setup(21,GPIO.OUT)
5 GPIO.setup(20,GPIO.OUT)
6 GPIO.setup(16,GPIO.OUT)
7
8 while True :
9     GPIO.output(21,GPIO.HIGH)
10    time.sleep(5) #on attend 5 seconde
11    GPIO.output(21,GPIO.LOW)
12    GPIO.output(20,GPIO.HIGH)
13    time.sleep(2) #on attend 1 seconde
14    GPIO.output(20,GPIO.LOW)
15    GPIO.output(16,GPIO.HIGH)
16    time.sleep(5) #on attend 5 seconde
17    GPIO.output(16,GPIO.LOW)
18
```


LE BOUTON POUSSOIR

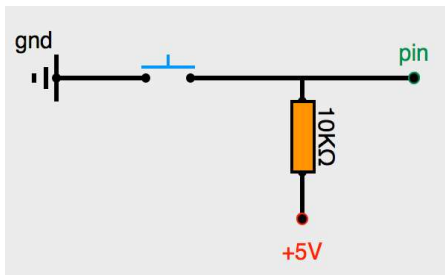


Résistance pull-down



- Si le poussoir est baissé, le courant va du +5V au pin de Raspberry. Il ne prendra pas le chemin du Ground car la résistance lui demande un effort. Le pin de Raspberry recevra du +5V et indiquera HIGH (ou 1).
- Si le poussoir est levé, donc le circuit ouvert, le très faible courant résiduel qui sortira du pin de l'Arduino sera absorbé par le Gnd, le pin sera donc bien en LOW (ou 0).

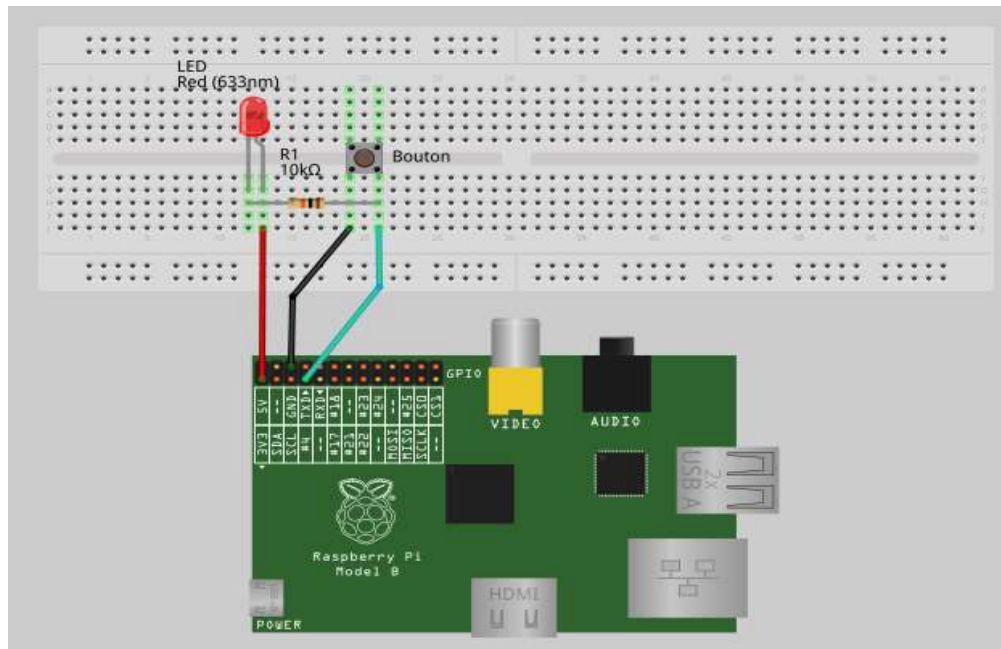
Résistance Pull-Up



- Quand le poussoir est ouvert, le +5V nourrit le pin de Raspberry, qui donnera HIGH comme résultat.
- Lorsqu'il est fermé, le +5V et le pin sont absorbés par le ground, le pin donnera LOW comme résultat.

Exercice 3 :

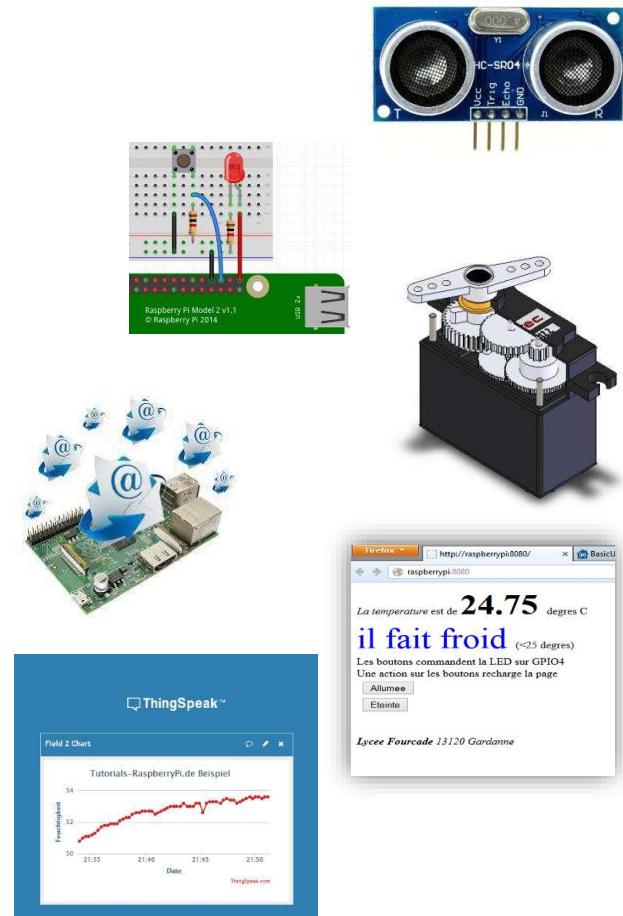
- Développez un programme qui permet d'allumer ne LED branchée sur le GPIO xx durant 5 secondes une fois le bouton poussoir actionné.
- Le bouton poussoir est lié au GPIO 12.
- Réalisez le montage et écrivez le programme approprié.



```
import RPi.GPIO as GPIO
import time
GPIO.setmode(GPIO.BCM)
GPIO.setup(4, GPIO.OUT)
GPIO.setup(12, GPIO.IN)
while True:
    if
GPIO.input(12)==GPIO.LOW:
    print("Allume")
    GPIO.output(4,1) #
Allume la LED
    time.sleep(5)
    print("Eteind")
    GPIO.output(4,0)
```


APPLICATIONS RASPBERRY (TP)

- ❖ PROGRAMMATION : BLINK, LED, BOUTON POUSSOIR,
- ❖ CAPTEUR DE MESURE DE DISTANCE (ULTRASON)
- ❖ CONTROL DU SERVO-MOTEURS
- ❖ CAPTEUR DE TEMPÉRATURE
- ❖ ENVOIE D'UN E-MAIL
- ❖ PLATEFORME GRATUITE POUR IOT (THINGSPEAK)
- ❖ APPLICATION CAMÉRA
- ❖ APPLICATION MOBILE POUR CONTRÔLE D'UNE SERRURE



FIN DU COURS