Raspberry : montage et programmation avec une LED

Montage à réaliser

Sur ce montage, la LED est reliée :

- par la patte de la **cathode** à la **masse** (GRD),
- par la patte de **l'anode** à une **résistance** de 1k ohms,
- l'autre patte la résistance à la broche 7 c'est à dire au **GPIO4** du Raspberry.

Code des couleurs

* https://www.apprendre-en-ligne.net/crypto/passecret/ohm.html

Programmation Python avec la bibliothèque RPi.GPIO

Découvrez l'environnement de programmation Thonny Python IDE pour vos scripts Python

```
import RPi.GPIO as GPIO
# Choix de la numerotation BCM
GPIO.setmode(GPIO.BCM)
```

- Définir le GPIO4 en sortie et allumer la led :

```
GPIO.setup(4, GPIO.OUT)                  # le GPIO4 est une sortie numerique
GPIO.setup(4, GPIO.OUT, initial=GPIO.HIGH)# le GPIO4 est une sortie initialement a l'etat haut
GPIO.output(4, GPIO.HIGH)                # allumer la LED : etat haut
GPIO.output(4, GPIO.LOW)                 # eteindre la LED : etat bas
```

- L'état haut correspond à GPIO.HIGH c'est à dire 1 ou encore True.
- L'état bas correspond à GPIO.LOW c'est à dire 0 ou False.

- pour gérer un inversement d'état <code python> GPIO.output(4, not GPIO.input(4)) </code>
- Remettre à zéro une entrée-sortie numérique À la fin de tout programme, il est conseillé de remettre les Entrées/sorties dans l'état où le programme les a trouvées. On n'impactera ainsi que les E/S utilisées. <code python> GPIO.cleanup() </code>

Exercice : faire un programme qui fait clignoter une LED. Pour cela utilisez le module time.

```
import time
# Attendre 1/2 seconde
time.sleep(0.5)
```

Programmation Python avec la bibliothèque gpiozero

La bibliothèque gpiozero permet une programmation simplifiée.

- allumer/éteindre la LED ; la faire clignoter <code python> from gpiozero import LED led = LED(2) led.on() # allumer la led led.off() # eteindre la led led.blink() # faire clignoter la led led.off() # eteindre la led </code>
- les arguments de la méthode blink <code python> led.blink(on*time=x, off*time=y, n=z) # on*time : durée d'éclairage de la LED # off*time : durée d'extinction de la LED # n : nombre de clignotement </code>

Un bouton pour allumer la led

Montage

- Pour **lire** l'état du bouton, le programme va venir lire l'état à intervalle régulier.
- Une fois le bouton appuyé, le programme **attend** le relâchement du bouton.

Le branchement du bouton :

- une patte du bouton est branchée à la **masse** (GRD),
- l'autre patte du bouton :
  - est branchée sur la broche 37 c'est à dire le **GPIO26**,
  - est également reliée à une **résistance de pull-up** qui est branchée à une des **sorties +3,3V** du Raspberry.

Cette résistance pull-up permet :

- de mettre le **potentiel** du **GPIO26** à **HIGH** (valeur 1 - niveau haut),
- de **protéger** le Raspberry en évitant le **court-circuit**.

Lorsque l'on **pousse** le bouton poussoir, le **potentiel** du **GPIO26 est forcé à LOW (valeur 0 - niveau bas) par la mise à la masse. La LED est toujours reliée à la masse et à la broche 40 (GPIO21).**

==== Programmation avec la bibliothèque GPIO ==== <code python> # Import des modules import RPi.GPIO import time # Initialisation de la numerotation et des E/S GPIO.setmode(GPIO.BCM) GPIO.setup(21, GPIO.OUT, initial=GPIO.LOW) # la led GPIO.setup(26, GPIO.IN) # le bouton # Si on detecte un appui sur le bouton, on allume la LED # et on attend que le bouton soit relache while True: state = GPIO.input(26) if not state: # on a appuye sur le bouton connecte sur la broche 26 GPIO.output(21, GPIO.HIGH) while not state: state = GPIO.input(10) time.sleep(0.02) # Pause pour ne pas saturer le processeur GPIO.output(21, GPIO.LOW) time.sleep(0.02) # Pause pour ne pas saturer le processeur </code> ==== Programmation avec la bibliothèque gpiozero ==== <code python> from gpiozero import LED, Button from signal import pause led = LED(21) bouton = Button(26) bouton.when*pressed* = *led.on bouton.when*released = led.off pause() </code>

L'instruction **bouton.when*pressed = led.on n'appelle pas la fonction (méthode) led.on mais crée une référence à cette fonction de telle sorte que la fonction led.on sera appelée seulement quand l'événement bouton.when_pressed se produira. De la même l'instruction bouton.when_released = led.off permet d'indiquer qu'il faut exécuter la fonction (méthode) led.off quand on relâche le bouton. </WRAP>***

Pour en savoir plus sur **gpiozero** : gpiozero.pdf

Travail à faire : réaliser un **feu tricolore** avec un bouton pour **piéton** :

- les feux doivent **passer régulièrement** du vert à l'orange puis au rouge avec la temporisation de votre choix,
- si un piéton **appuie** sur le bouton :
  - le feu passe **immédiatement** à l'orange puis au rouge,
  - le feu rouge doit **durer plus longtemps** que durant le cycle normal,
  - puis le **cycle habituel reprend** avec la **temporisation initiale**.

==== Les activités ... ====

Je reviens à la liste des activités.