



# Raspberry Pi (5) - gestion des INPUT (boutons poussoir) -

## Gestion des INPUT (boutons poussoirs)

|          |                |        |
|----------|----------------|--------|
| Nom :    | Appréciation : | Note : |
| Prénom : |                |        |
| Classe : |                |        |
| Date :   |                |        |

|            |            |
|------------|------------|
| Objectif : | durée : 4h |
| Utilité :  |            |

**Matériel :** plaque labdec – composants électroniques

**Prérequis :** Connexion à distance avec SSH, commande GPIO

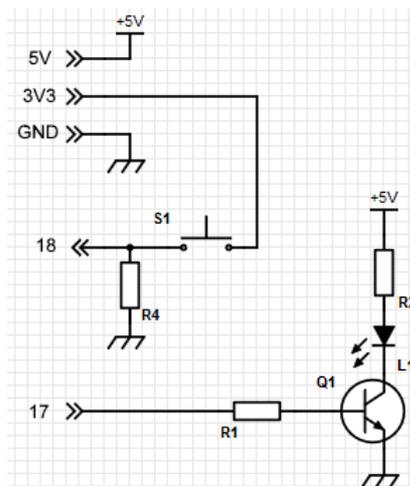
**Compétences et savoirs principalement visées :**  
C2-1, C2-2 (page 3a), C3-2, C3-3 (page 3b à 6)

**Travail à réaliser :**

- 
- 
- 

**Schéma du système :**

| Pi B+ GPIO Ref |   |      |
|----------------|---|------|
| 3.3V           | ● | 5V   |
| 2              | ● | 5V   |
| 3              | ● | GND  |
| 4              | ● | 14   |
| GND            | ● | 15   |
| 17             | ● | 18   |
| 27             | ● | GND  |
| 22             | ● | 23   |
| 3.3V           | ● | 24   |
| 10             | ● | GND  |
| 9              | ● | 25   |
| 11             | ● | 8    |
| GND            | ● | 7    |
| IDSD           | ● | IDSC |
| 5              | ● | GND  |
| 6              | ● | 12   |
| 13             | ● | GND  |
| 19             | ● | 16   |
| 26             | ● | 20   |
| GND            | ● | 21   |

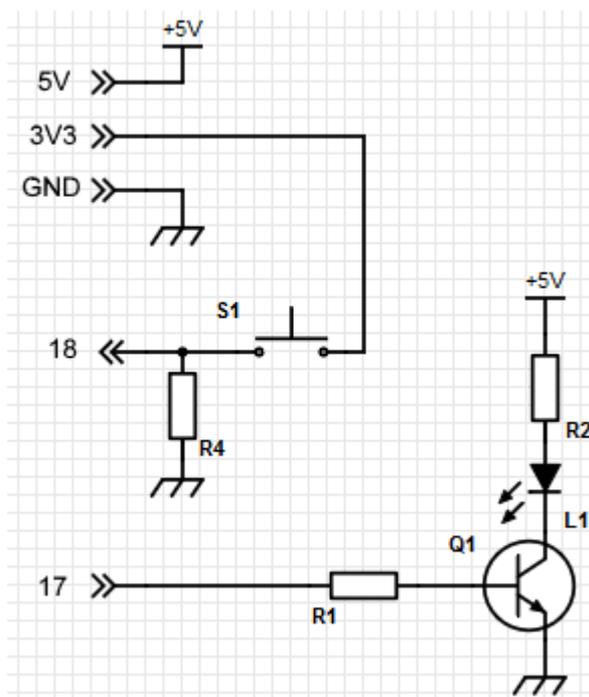


- Réalise le montage suivant sur la plaque Labdec

### AVERTISSEMENT

**Attention** : en cas d'erreur de branchement, ton Raspberry Pi risque d'être **détruit** !!! Ne mets pas le circuit sous tension **avant** que le professeur l'ai vérifié.

| Pi B+ GPIO Ref |   |      |
|----------------|---|------|
| 3.3V           | ● | 5V   |
| 2              | ● | 5V   |
| 3              | ● | GND  |
| 4              | ● | 14   |
| GND            | ● | 15   |
| 17             | ● | 18   |
| 27             | ● | GND  |
| 22             | ● | 23   |
| 3.3V           | ● | 24   |
| 10             | ● | GND  |
| 9              | ● | 25   |
| 11             | ● | 8    |
| GND            | ● | 7    |
| IDSD           | ● | IDSC |
| 5              | ● | GND  |
| 6              | ● | 12   |
| 13             | ● | GND  |
| 19             | ● | 16   |
| 26             | ● | 20   |
| GND            | ● | 21   |



### AVERTISSEMENT

**Attention** : Il ne faudra surtout jamais mettre le signal 18 en mode OUT avec ce circuit ; en cas d'appui sur S1, ton Raspberry Pi serait immédiatement **détruit** !!!

## Le GPIO en mode INPUT

Tu connais maintenant les GPIO en mode **OUTPUT**, qui permet d'allumer ou éteindre un équipement électrique.

Le mode **INPUT**, quant à lui, permet **d'entrer des informations dans le RPi** ; par exemple : l'état d'un bouton poussoir (appuyé/relâché), l'état d'un capteur quelconque (actif/inactif), ou simplement des bits provenant d'un autre équipement électronique via un câble.

Sur ta plaque LabDec, tu as câblé un bouton-poussoir (BP) relié au signal 18 (bcm)

- ◆ Tapes la commande suivante et indiques ce que tu constates :

*Note importante* : tout ce qui est situé sur la ligne après le caractère « # » est un commentaire ; il ne faut pas le taper

```
gpio readall # Vérifier l'état des broches
```

Dans quel état est la broche 18: (IN ou OUT ?) \_\_\_\_\_, (0 ou 1 ?) \_\_\_\_\_

- ◆ La broche 18 est-elle bien en mode INPUT ? Sinon quelle commande aurait-tu tapé pour la passer en mode INPUT ?

- ◆ Tapes la commande suivante, **tout en appuyant sur le BP** et indiques ce que tu constates :

```
gpio readall # Vérifier l'état des broches
```

Dans quel état est la broche 18: (IN ou OUT ?) \_\_\_\_\_, (0 ou 1 ?) \_\_\_\_\_

Pour comprendre ce qui se passe au niveau électrique, rempli le tableau suivant en faisant les mesures correspondantes :

| Bouton S1 | Tension au point 18<br><i>en Volts *</i> | Niveau logique<br>signal 18 # |
|-----------|--|-------------------------------|
| relâché   |  |                               |
| appuyé    |  |                               |

\* à mesurer

# à déduire

## Détection de l'appui et du relachement depuis un script

La commande pour lire l'état (0 ou 1) d'un GPIO en mode input est :

```
gpio -g read n°du signal
```

- ◆ Tapes la commande suivante, **tout en laissant le BP relâché**, et indiques quelle est la valeur renvoyée par la commande :

```
gpio -g read 18 # lecture de l'état de l'entrée 18
```

quelle est la valeur renvoyée par la commande : \_\_\_\_\_

- ◆ Tapes la commande suivante, **tout en appuyant sur le BP** et indiques quelle est la valeur renvoyée par la commande :

```
gpio -g read 18 # lecture de l'état de l'entrée 18
```

quelle est la valeur renvoyée par la commande : \_\_\_\_\_

- ◆ A partir de WinSCP, crée le fichier «**read.sh**» dans le dossier «**/root**»

### Fichier read.sh

```
#!/bin/bash
gpio -g mode 18 in
b=0
while [ 1==1 ]
do
# attendre que le BP soit poussé
while [ $b -eq 0 ]
do
b=$(gpio -g read 18)
done
echo "le bouton est appuyé..."

# attendre que le BP soit relâché
while [ $b -eq 1 ]
do
b=$(gpio -g read 18)
done
echo "le bouton est relâché..."

done
```

Condition de sortie :  $\$b \neq 0$

$\$b$  va prendre la valeur 1 quand BP sera appuyé

La boucle s'arrête quand  $\$b=1$ , c'est-à-dire quand BP est appuyé

- ◆ Rendre «**read.sh**» exécutable, puis teste-le.

## Bouton PUSH

l'appui sur le bouton PUSH provoque l'allumage de la led 17 ; elle s'éteint dès qu'on relâche le bouton

- ◆ A partir de WinSCP, crée le fichier « **push.sh** » dans le dossier « **/root** »

### Fichier push.sh

```
#!/bin/bash
```

```
gpio -g mode 17 out
```

```
gpio -g mode 18 in
```

```
b=1
```

```
while [ 1==1 ]
```

```
do
```

```
  b=$(gpio -g read 18)
```

```
  gpio -g write 17 $b
```

```
  sleep 0.1
```

```
done
```

L'instruction **while** => **do** => **done** permet de faire une boucle de programme ; une boucle est une série d'instructions qui va s'exécuter en boucle jusqu'à ce que la condition de sortie soit remplie ;

- ◆ Dans un terminal Putty connecté au Raspberry Pi, tapes les commandes suivantes :

```
cd /root
```

```
chmod +x push.sh
```

- ◆ Lances ton script avec la commande : **./ push.sh**
- ◆ Que constate-tu ? (Relance le script autant de fois que nécessaire)
- ◆ Rempli le tableau pour vérifier que tu as compris le fonctionnement :

| Bouton S1 | Niveau logique<br>broche 18 | État de la LED<br><i>allumée/éteinte</i> |  |
|-----------|-----------------------------|--|--|
| relâché   |                             |  |  |
| appuyé    |                             |  |  |

- ◆ A ton avis, est-il nécessaire d'utiliser un Raspberry Pi pour obtenir ce fonctionnement ?

- ◆ Si le script est arrêté, la LED s'allume-t-elle encore quand on appui sur le bouton ? Pourquoi ?

## Bouton MONOSTABLE (temporisé)

l'appui sur le bouton MONOSTABLE provoque l'allumage de la led ; elle s'éteindra au bout de 7 secondes.

- ◆ A partir de WinSCP, crée le fichier «**monostable.sh**» dans le dossier «**/root**»

### Fichier monostable.sh

```
#!/bin/bash
gpio -g mode 17 out
gpio -g mode 18 in

while [ 1==1 ]
do
# éteindre la led
gpio -g write 17 0
b=0

# attendre que le bouton S1 soit poussé
while [ $b -eq 0 ]
do
    b=$(gpio -g read 18)
done

# allumer la led
gpio -g write 17 1

# attendre 7 secondes
sleep 7
done
```

- ◆ Dans un terminal Putty connecté au Raspberry Pi, tapes les commandes suivantes :

```
cd /root
```

```
chmod +x monostable.sh
```

- ◆ Lances ton script avec la commande : **./monostable .sh** , que constate-tu ?

- ◆ Au bout de combien de temps la LED s'éteint-elle ? \_\_\_\_\_ secondes

- ◆ Modifie légèrement le script pour que la LED reste à l'**état allumé pendant 20 secondes** ; testes ce script modifié ; que constates-tu ?

Recopie ici la ligne modifiée

## Bouton BISTABLE

l'appui sur le bouton BISTABLE provoque l'allumage de la LED; elle s'éteint quand on appui à nouveau sur ce bouton.

- ◆ A partir de WinSCP, crée le fichier «**bistable.sh**» dans le dossier «**/root**»

### Fichier bistable.sh

```
#!/bin/bash

gpio -g mode 17 out
gpio -g mode 18 in
b=0
# la variable $on prendra les valeurs 0 ou 1 pour "allumer" ou "éteindre"
# la led est initialement éteinte
on=0
gpio -g write 17 $on

while [ 1==1 ]
do

# attendre que le bouton S1 soit poussé
while [ $b -eq 0 ]
do
    b=$(gpio -g read 18)
done

# inverser la valeur de $on
if [ $on -eq 0 ]
then
    on=1
else
    on=0
fi

# éteindre ou allumer la led selon la valeur de $on
gpio -g write 17 $on

# attendre que le bouton S1 soit relâché
while [ $b -eq 1 ]
do
    b=$(gpio -g read 18)
done

sleep 0.5

done
```

◆ Dans un terminal Putty connecté au Raspberry Pi, tapes les commandes suivantes :

```
cd /root
```

```
chmod +x bistable.sh
```

◆ Lances ton script avec la commande : `./ bistable .sh` , comment le bouton fonctionne-t-il ?

◆ **BONUS** (*pour les élèves le plus rapides !!*) Modifie le script pour que la LED clignote 2 fois avant de s'allumer, ou de s'éteindre...

**Info** : utilise un délai de 0.2s pour l'état allumé/éteint

Recopie ici ton script modifié :

## Bouton BISTABLE : exemple d'un bouton Play/Pause

Le fonctionnement doit être conforme à ce qui suit :

- l'appui sur le BP provoque la lecture (« Play ») d'un fichier mp3 et en même temps l'allumage de la LED
- Si on appuie à nouveau sur le BP pendant la lecture, celle-ci se met en « Pause » et la LED se met à clignoter
- Quand la lecture est terminée la LED s'éteint

- ◆ Installe le logiciel omxplayer : **apt-get install omxplayer**
- ◆ Télécharge le fichier « mp3 » : wget [http://10.3.14.254/eleve/multimedia/Cruel\\_Summer.mp3](http://10.3.14.254/eleve/multimedia/Cruel_Summer.mp3)
- ◆ A partir de WinSCP, crée le fichier « **mp3clignote.sh** » dans le dossier « **/root** », après l'avoir complété et fait vérifier par le professeur

Fichier mp3clignote.sh

```
#!/bin/bash

delai=$1

# le signal 17 doit être dans le bon mode
gpio -g mode [REDACTED]

# boucle infinie
while true
do
    # allumer la LED et attendre pendant $delai
    [REDACTED]

    # éteindre la LED et attendre pendant $delai
    [REDACTED]
done
```

- ◆ A partir de WinSCP, crée le fichier «mp3.sh » dans le dossier « /root », après l'avoir complété et fait vérifier par le professeur

#### Fichier mp3.sh

```
#!/bin/bash
killall omxplayer.bin
killall omxplayer
killall mp3clignote.sh
if [ -e /root/omxplayer.pipe ]; then
    rm /root/omxplayer.pipe
fi
mkfifo /root/omxplayer.pipe

#mettre les signaux 17 et 18 dans le bon mode ("in" ou "out")
gpio -g mode 17
gpio -g mode 18
b=0
# la LED doit être éteinte
# on attend que le bouton play soit appuyé
while [ $b -eq 0 ]
do
    b=$(gpio -g read 18)
done
# attendre que le bouton play soit relâché
while [ $b -eq 1 ]
do
    b=$(gpio -g read 18)
done
# allumer la LED
# lancer la lecture
omxplayer -o local "/root/cruel_summer.mp3" < /root/omxplayer.pipe &
echo -n . > /root/omxplayer.pipe &
echo -n "1" > /root/omxplayer.pipe &
on=1
sleep 1

#
while true
do
    # vérifier si le bouton play/pause est poussé
    # en même temps vérifier si la chanson est terminée
    while [ $b -eq 0 ]
    do
        b=$(gpio -g read 18)
        #la lecture est-elle terminée ?
        # si encours == " cela signifie que la lecture est terminée
        encours=`ps ax | grep omxplayer.bin | grep -v grep`
```

```

if [[ $encours == " " ]]; then
# éteindre la LED
# quitter le script
exit 0
fi
# on met le script en sommeil pendant 100ms pour ne pas surcharger le CPU
sleep 0.1
done

# attendre que le bouton play/pause soit relâché
while [ $b -eq 1 ]
do
b=$(gpio -g read 18)
done

# mettre en pause ou reprendre la lecture ?
if [ $on -eq 0 ]
then
on=1
killall mp3clignote.sh
# allumer la LED
echo -n "p" > /root/omxplayer.pipe &
echo "Reprise de la lecture..."
else
on=0
# on lance mp3clignote avec le paramètre 0.2 (pour 0.2 secondes)
echo -n "p" > /root/omxplayer.pipe &
echo "Lecture en pause..."
fi
done
exit 0

```

- ◆ **BONUS 2** (*pour les élèves super rapides !!*) Télécharge le morceau « Pink\_Floyd.mp3 » et modifie le script de façon à ce qu'il soit joué à la place de « Cruel\_Summer.mp3 »

Recopie ici les lignes modifiées :

◆ **BONUS 3** (*pour les élèves super rapides !!*) Écris un script « fpush.sh » pour que la LED clignote rapidement quand on appuie sur le bouton et qu'elle s'éteigne quand on le relâche

Recopie ici ton script modifié :