Du feu simple vers un feu un peu intelligent

Conception d’une maquette fonctionnelle

Pré-requis

Prise en main du micro contrôleur Raspberry Pico Pi

Description

Dans cette séance, nous allons mettre en place une maquette de feu tricolore simple, puis le faire évoluer vers un système équipé de capteurs et d’éléments de restitution afin de permettre la prise en compte d’évènements externes et d’informer les usagers.
La séance se compose de 4 activités à réaliser seul et à faire valider par le professeur. Il n’y a rien à rendre à la fin de la séance

Objectifs :

- Comprendre comment les feux de circulation sont contrôlés et gérés pour optimiser le trafic.
- Explorer les différents aspects du contrôle et de la gestion des feux de circulation intelligents.
- Simuler le fonctionnement avec le micro-contrôleur Pico Pi.
- Réaliser une maquette qui servira de base pour la suite de la séance.

Compétences

- CO2.1 Décoder le cahier des charges d’un produit, participer, si besoin, à sa modification
- CO5.8 SIN1 Proposer/choisir l’architecture d’une solution logicielle et matérielle au regard de la définition d’un produit
- CO5.8 SIN2 Rechercher et écrire l’algorithme de fonctionnement puis programmer la réponse logicielle relative au traitement d’une problématique posée
- CO7.1 Réaliser et valider un prototype ou une maquette obtenus en réponse à tout ou partie du cahier des charges initial.
- CO7.3 SIN2 Des architectures matérielles et logicielles en réponse à une problématique posée

Connaissances

- 5.3. Constituants de l’information
  - Composants programmables
    - Cartes électroniques à microcontrôleur
- 1.2. Outils de l'ingénierie système
  - Ingénierie système
    - Analyse du besoin :besoin initial, mission principale, contexte, cas d’utilisations, scénarios d’utilisation, besoins des parties prenante
- 1.2. Outils de l'ingénierie système
  - Ingénierie système
    - Spécification technique, conception de l’architecture

- 6.1. Moyens de prototypage rapide
  - Prototypage de pièces et de la chaîne d’information
- 6.2. Expérimentations et essais

  - Expérimentations de constituants de la chaîne d’information
- 3.4. Comportement informationnel des produits
  - Inter-opérabilité des produits
    - Système temps-réel
- 4.3. Conception des produits
  - Conception informationnelle des produits
    - Structures de programmation
    - Codage dans un langage spécifique

Évaluation de la séance

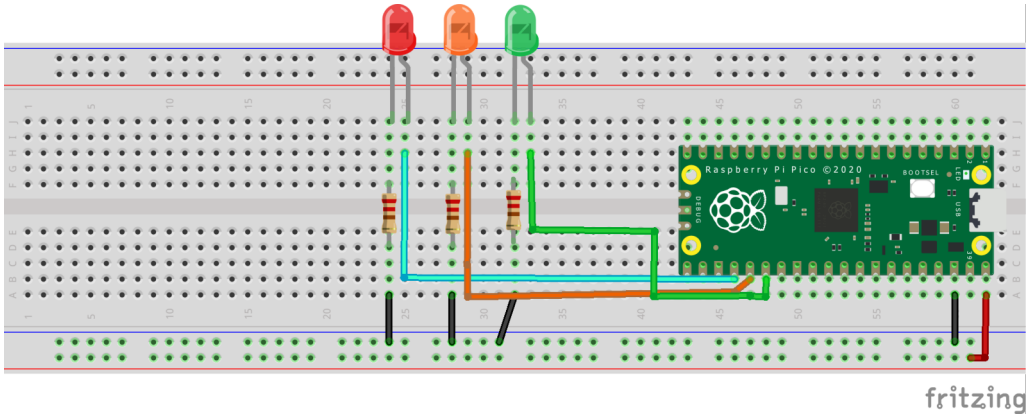
L’évaluation se fait au fil de l’eau sur toute la durée des activités pratiques. Vous devez montrer à l’enseignant que le montage fonctionne à chaque étape.

Il sera tenu compte de votre implication et de votre sérieux durant la séance.

N’hésitez pas solliciter l’enseignant si vous êtes bloqué.

Activité 1 - Feu tricolore basique

Réaliser le montage suivant : Les résistances sont des 220Ω (pas nécessaire avec le pico pi).



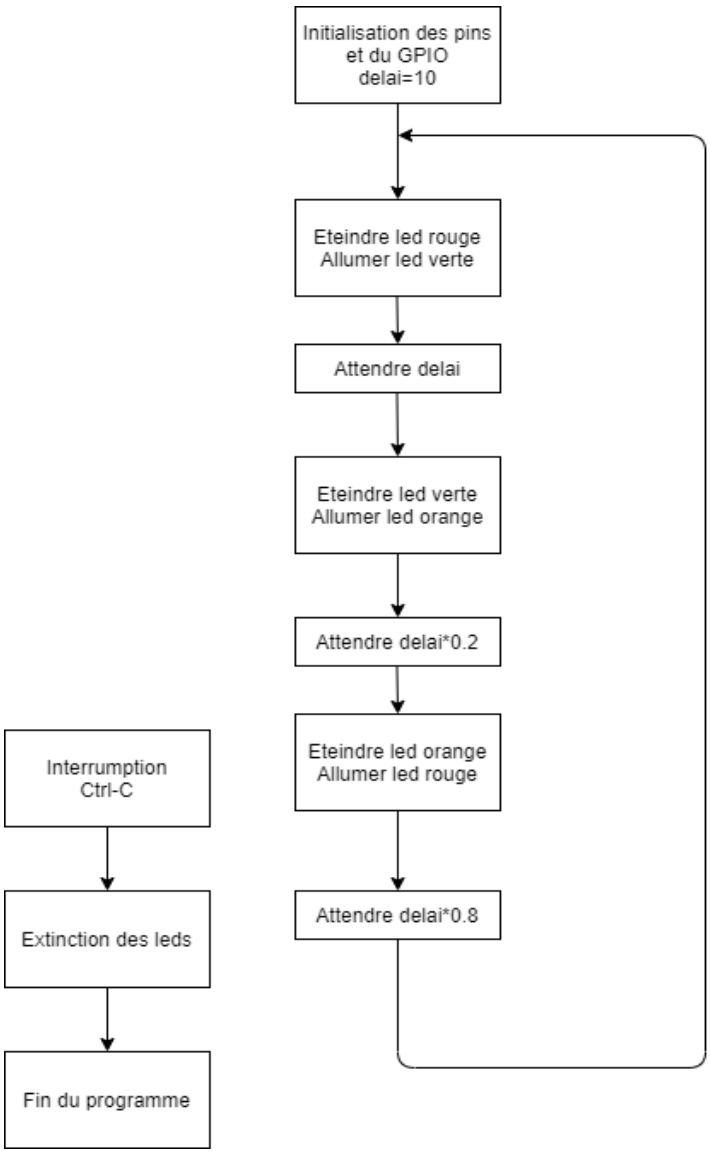
Programmer le Pico Pi afin qu'il réalise l'algorigramme présente à coté.

Pour la durée de base on prendra délai=10s

Pour la gestion de l'arrêt du script de façon propre, nous allons utiliser un bloc try / except qui permet de gérer les exceptions (erreurs qui pourraient se produire).
Exemple :

```
try :
    votre code principal

except KeyboardInterrupt :
    code à exécuter lorsque
    l'exception se produit
```

Remarque : Il est conseillé d'utiliser des fonctions (non décrites ici) afin de rendre votre code facilement réutilisable.

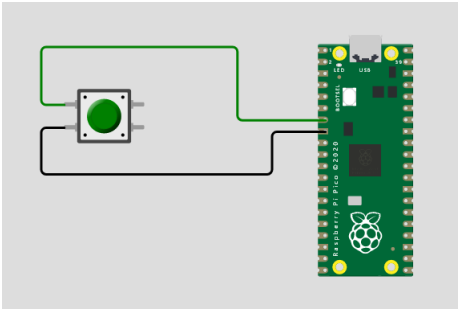


Faire constater par le professeur que le système fonctionne !

Pré requis à l’activité suivante : Utilisation d’un bouton poussoir

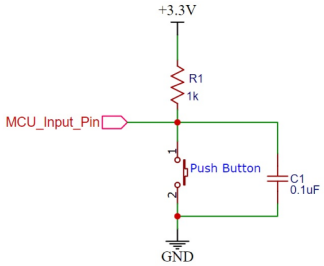
Essayer avec le simulateur le montage suivant :

```
from machine import Pin
counter=0
pin = Pin(5, Pin.IN, Pin.PULL_UP)
while True:
    if pin.value()==0:
        print("Bouton appuyé")
        counter+=1
        print("Count={}".format(counter))
```



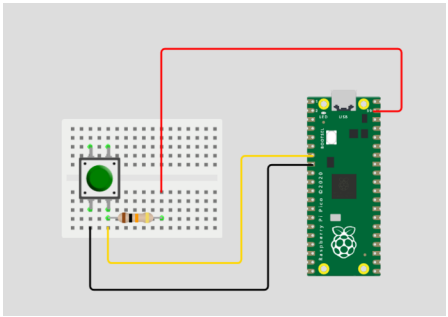
On remarque que lors d’un appui le moniteur renvoie plusieurs réponses. Ceci est dû à la nature mécanique du contact au niveau de l’interrupteur. C’est l’effet de rebond (bouncing). Pour palier à cet effet indésirable il existe plusieurs solutions :

- solution matérielle : utilisation d’un condensateur en parallèle entre le + et le -



- solution logicielle : on mesure le temps écoulé entre deux appuis de boutons consécutifs, et si plusieurs entrées se produisent dans un certain intervalle de temps, une seule entrée est enregistrée.

Nous allons choisir la solution logicielle.



Un pin en mode input qui n’est connecté à rien n’est pas forcément à l’état 0. Son état fluctue à cause des perturbations électromagnétiques ambiante.
Afin de forcer l’état du signal à un état stable, on utilise une résistance dite de pull-up comme sur le montage dessus pour forcer l’état à 1 (ou de pull-down pour forcer l’état à 0). On prendra une résistance de 10kΩ.

Dans cet exemple, on définit le pin 5 en mode input Pin.IN et Pin.PULL_UP car la résistance est en pull up.

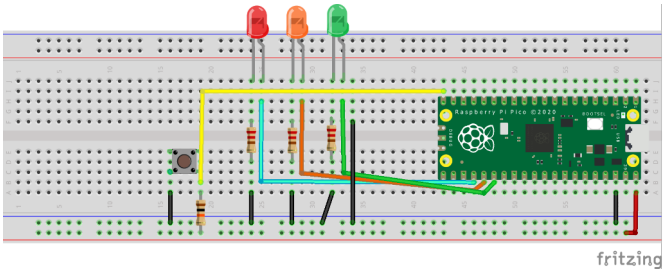
Code exemple :

```
from machine import Pin

def callback() :
     print("Bouton appuyé")

btn_pin = Pin(15, Pin.IN, Pin.PULL_UP)
btn_pin.irq(trigger=Pin.IRQ_FALLING, handler=callback)
```

La ligne d’après met en place une interruption (IRQ Interrupt ReQuest) sur le front descendant (Pin.IRQ_FALLING). Lorsque cette interruption se produit, le programme exécute la fonction spécifiée par handler=callback

Activité 2 - Feu tricolore basique avec bouton poussoir - 30min

Modifier le montage précédent pour ajouter un bouton poussoir comme ci-dessous.

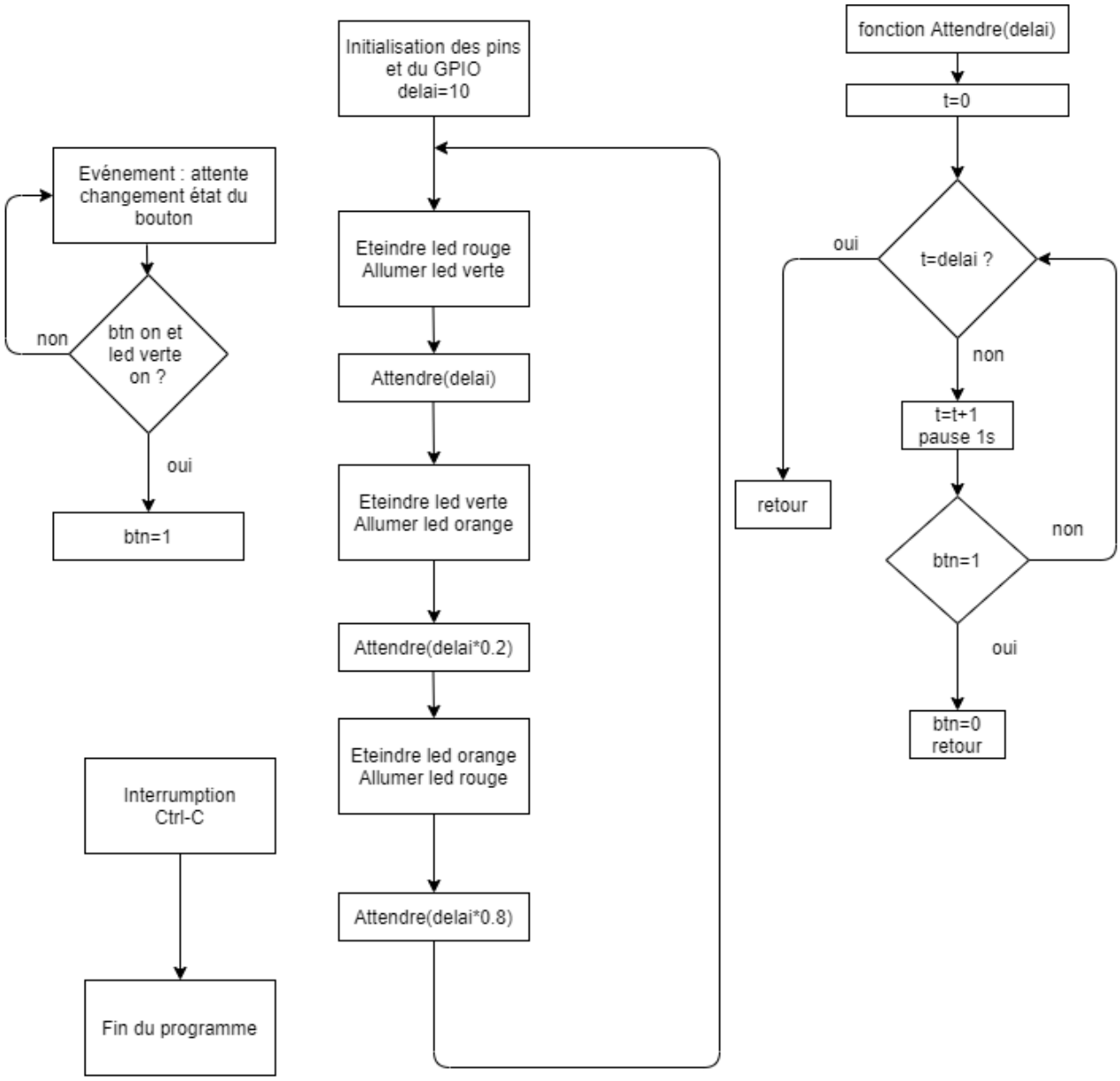


En vous basant sur le code précédent, ajouter la fonctionnalité de demande de passage pour les piétons à l'aide d'un bouton poussoir. Comme vous utilisez un BP de type Grove, il n'est pas nécessaire de mettre la résistance.

Dans cette partie, il vous est demandé de créer une fonction *Attendre(delai)* qui se substituera à la fonction *sleep(delai)* du module time. En effet cette dernière met le programme en pause complète pendant la durée du délai donnée par la variable *delai*. Si vous appuyez sur le bouton poussoir pendant que le programme est dans un *sleep(delai)*, la prise en compte ne sera effective qu'après la durée de *delai*. Par conséquent, la demande de passage ne sera pas prise en compte correctement.

La fonction *Attendre* permettra de vérifier toutes les secondes si on a appuyé sur le BP regardant la valeur de la variable associée à l'appui sur le BP.

Algorigramme simplifié :

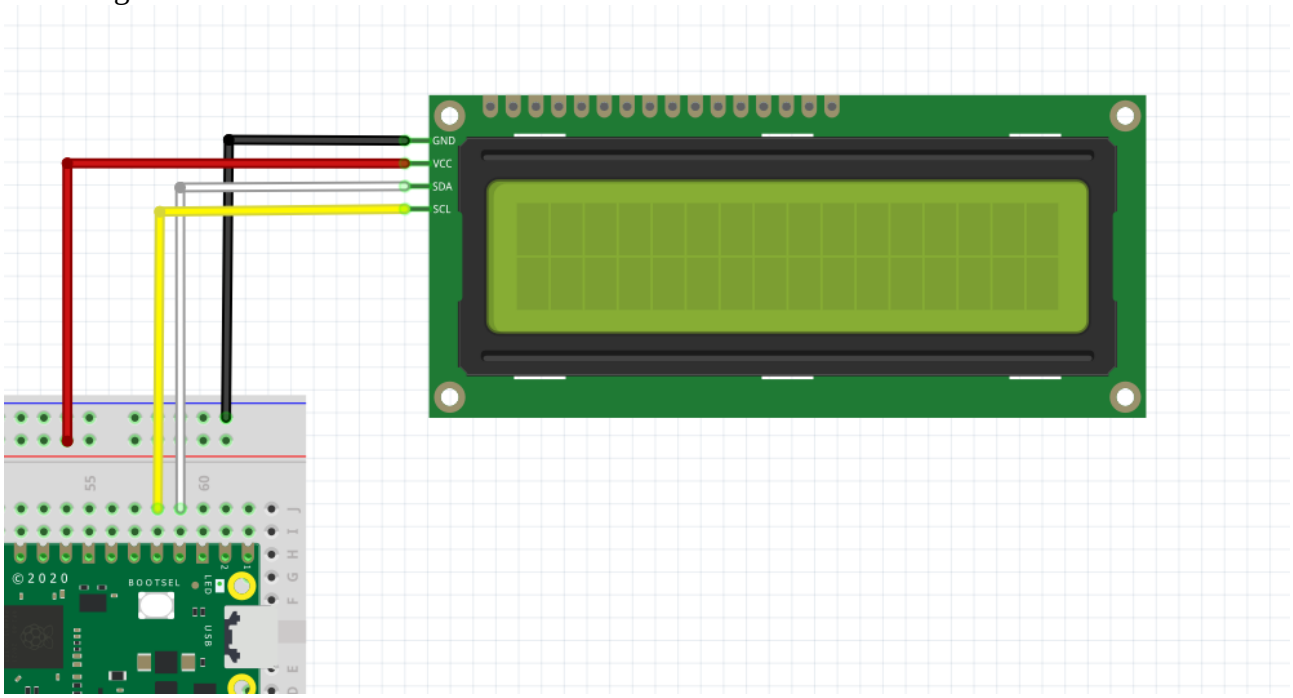


Faire constater par le professeur que le système fonctionne !

Activité 3 - Feu tricolore ajout d’un écran

On va ajouter un écran afin que le système puisse informer les utilisateurs.

On utiliserons un écran LCD 16x2 de type i2c
Le câblage se fait comme ci-dessous



Pour pouvoir utiliser cet écran avec le Pico Pi, il faut utiliser une libraire spécifique.
Cette libraire devra être placer dans un dossier lib à la racine du Pico Pi.

La librairie a utiliser avec cet écran se nomme : grove_lcd_i2c.py
Elle est dans fichier ressources.zip
Vous trouverez aussi un fichier exemple nommé test_lcd.py pour vérifier si l’écran fonctionne.

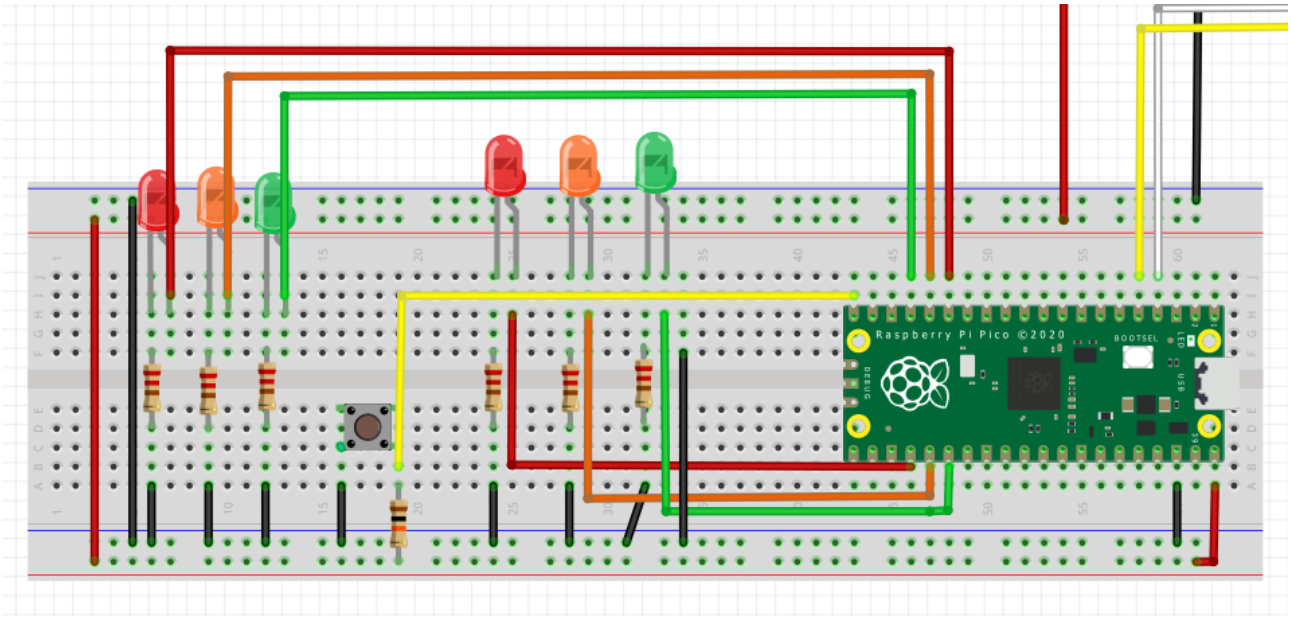
Ensuite modifier le code précédent pour intégrer l’utilisation de l’écran LCD.
On affichera :

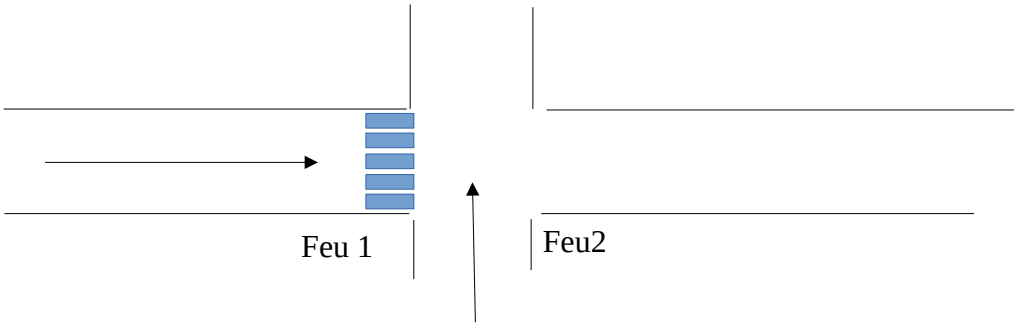
- « Passage demande » lorsque l’on appuie sur le BP
- Lorsque le feu devient rouge :
  - « Passage autorise » sur la 1ere ligne
  - « Piétons : temps-restant-passage-pietons » sur la 2nde ligne. Dans ce cas il y aura un décompte du temps restant.

Faire constater par le professeur que le système fonctionne !

Activité 4 - Avec 2 feux.

Ajouter maintenant une seconde série de leds (R,J,V) avec leur résistance sur les pins 15,16,17 qui correspondent aux GP11 GP12,GP13 et un second bouton poussoir.
Attention sur le schéma ci-dessous il manque un BP





En vous basant ce que vous avez fait précédemment, programmer le Pico Pi pour qu'il gère les deux feux comme à une intersection.

Faire constater par le professeur que le système fonctionne !