

Savoir-faire : Expérimenter des architectures matérielles et logicielles en réponse à une problématique.

Problématique : Vous devez mettre au point un programme qui permettent d'indiquer le niveau de remplissage d'un volume.

1. Programmation du module potentiomètre

✍ Connecter le potentiomètre au GPIO32 de l'ESP32.

✍ Ecrire le script ci-dessous et tester le fonctionnement.

```
from machine import ADC
import time

pot = ADC(32) # GPIO32 en entrée analogique

while True:
    val_ana = pot.read_uv() # lecture de la valeur analogique
    val_num = pot.read()    # lecture de la valeur numérique
    print(f"la valeur analogique de la tension est de {val_ana} microvolts")
    print(f"la valeur numérique de la tension est de {val_num}")
    print("")
    time.sleep_ms(100)
```

L'instruction ADC(32) permet de rendre actif le Convertisseur Analogique Numérique (CAN) (Analog to Digital Conversion : ADC en anglais) de la broche 32.

Le CAN convertit la valeur de la tension analogique présente sur la broche 32 (tension donnée par le potentiomètre) en valeur numérique (figure 1).

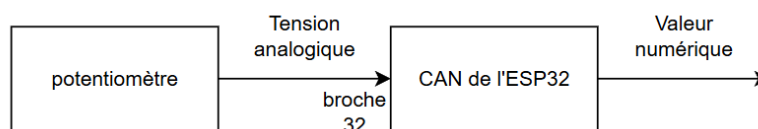


Figure 1 : CAN

✍ Quelles sont les valeurs extrêmes de la valeur numérique ?

✍ Quelles sont les valeurs analogique correspondantes ?

2. Indicateur de niveau d'eau à LED

Un particulier veut connaître le niveau de remplissage de son récupérateur d'eau de 1 000 litres. Pour ceci, il place au fond de la cuve un capteur de pression qui délivre une information électrique proportionnelle à la **hauteur d'eau**.

Deux LEDs vont lui indiquer le niveau dans la cuve :

- La **LED rouge** s'allume si la **cuve est vide** (hauteur d'eau inférieure à 10 cm).
- La **LED verte** s'allume si la **cuve est pleine** (hauteur d'eau supérieure à 90 cm).

✍ Compléter l'algorithme (figure 2) permettant de connaître le niveau d'eau (à partir des LEDs) en fonction de l'information de la hauteur d'eau.

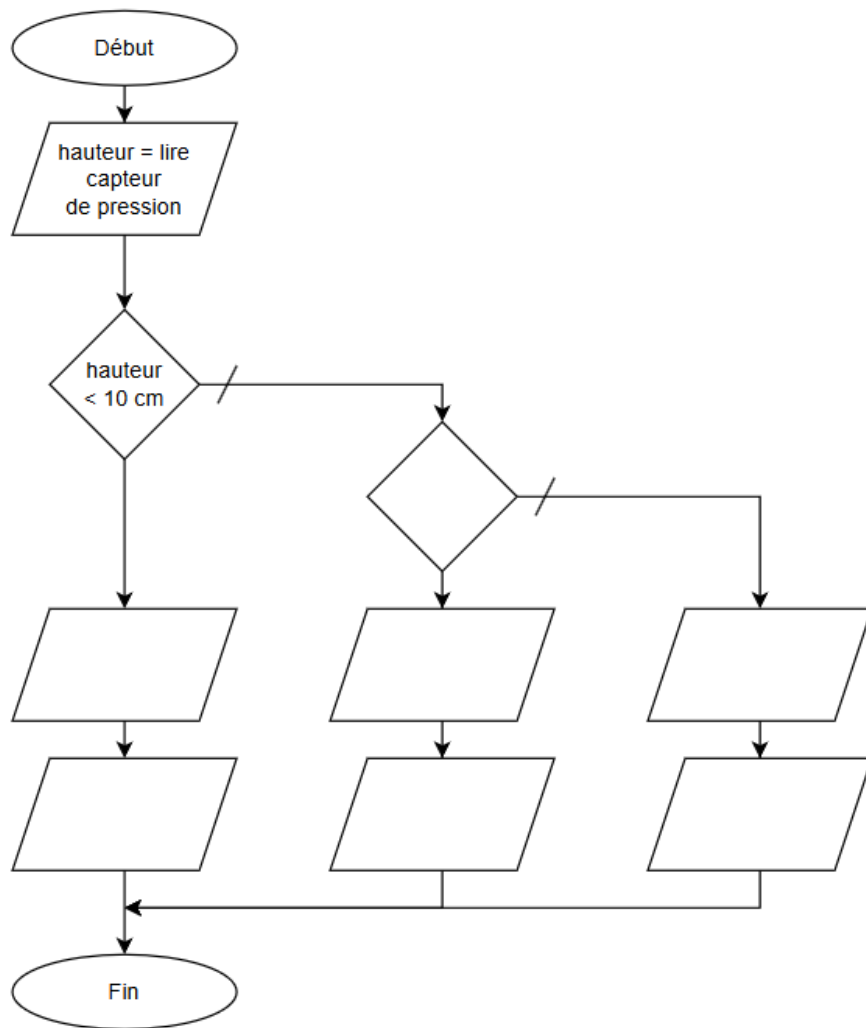


Figure 2 : algorithme

Lorsque le capteur de pression est connecté à une entrée analogique de l'ESP32, la valeur numérique obtenue en sortie du CAN pour une hauteur d'eau de 1 mètre est de 4 000.

✍ Calculer les valeurs numériques pour les hauteurs d'eau de 10 cm et de 90 cm.

Pour la mise au point du programme :

- Le potentiomètre simulera le capteur de pression.
- La LED rouge et la LED verte indiqueront le niveau d'eau.
- La carte ESP32 servira à traiter l'information.

✂ Compléter la chaîne d'information (figure 3) avec le nom des constituants de chaque bloc.

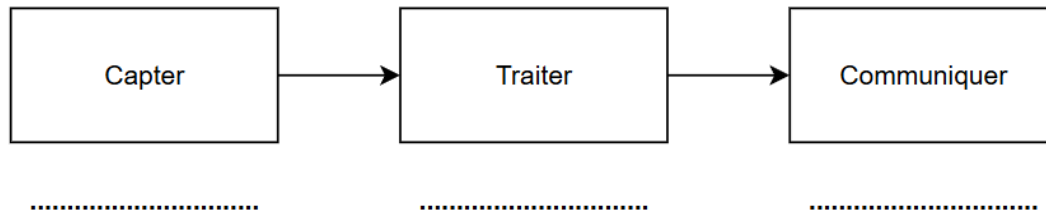


Figure 3 : chaîne d'information

✂ Connecter le potentiomètre et les deux LEDs à la carte ESP32.

✂ Compléter le programme ci-dessous et le tester. La fonction `niveau_eau_led` correspond à l'algorithme de la figure 2.

```

from machine import ADC, Pin
import time

pot = ADC(32) # GPIO32 en entrée analogique
led_rouge = Pin(. . . , Pin.OUT)
led_verte = Pin(. . . , Pin.OUT)

def niveau_eau_led():

```

```

while True:
    niveau_eau_led()
    time.sleep_ms(1)

```

✂ Modifier la fonction `niveau_eau_led` afin de faire clignoter la LED rouge lorsque le niveau d'eau descend en dessous de 5 cm.

3. Indicateur de niveau d'eau à LED RGB

La LED rouge et la LED verte vont être remplacées par une seule LED de type RGB.
Ceci va permettre de n'utiliser qu'une seule LED et d'avoir plus de couleurs disponibles.

La bibliothèque `neopixel` va permettre de gérer la LED RGB

```
from machine import Pin
from neopixel import NeoPixel
```

```
led_rgb = Pin(23, Pin.OUT)          # GPIO23 en sortie
```

```
led_driver = NeoPixel(led_rgb, 1)   # création du driver Neopixel pour 1 LED (pixel)
led_driver[0] = (255, 255, 255)     # configuration de la LED 1 en blanc
led_driver.write()                  # écriture du contenu du driver
```

✂ Connecter la LED RGB conformément au programme ci-dessus et tester le programme.

✂ Implémenter un programme qui indique le niveau de remplissage de la cuve conformément à la figure 4.

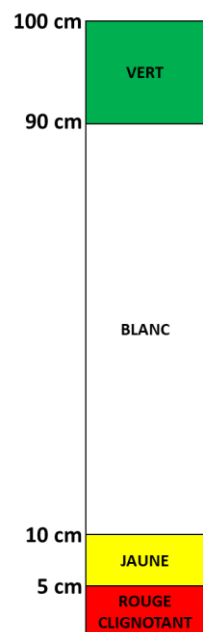


Figure 4 : couleur de la LED RGB en fonction de la hauteur d'eau