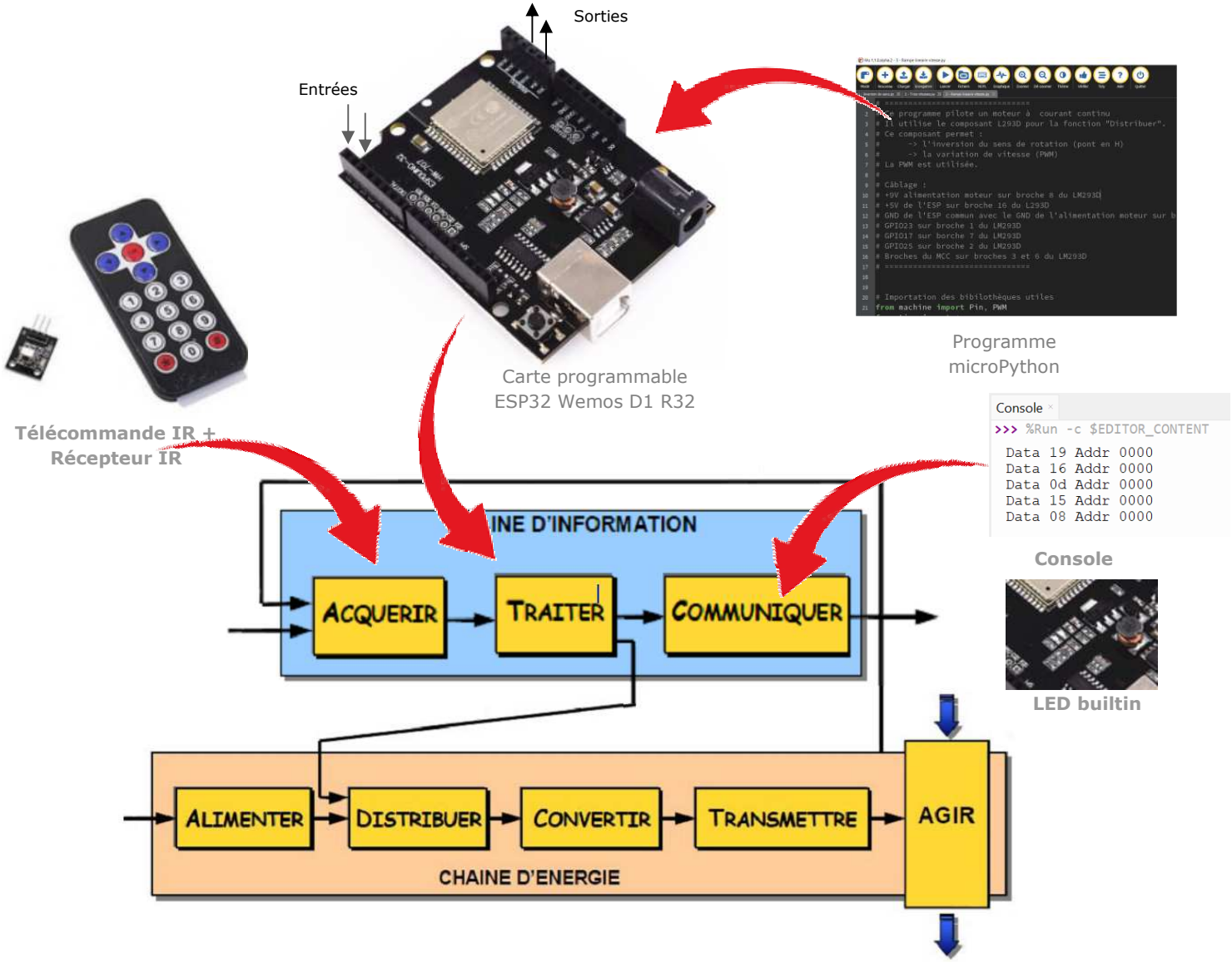
MISE EN ŒUVRE

→ **TRAITER : ESP32 WEMOS (EDI MU)**

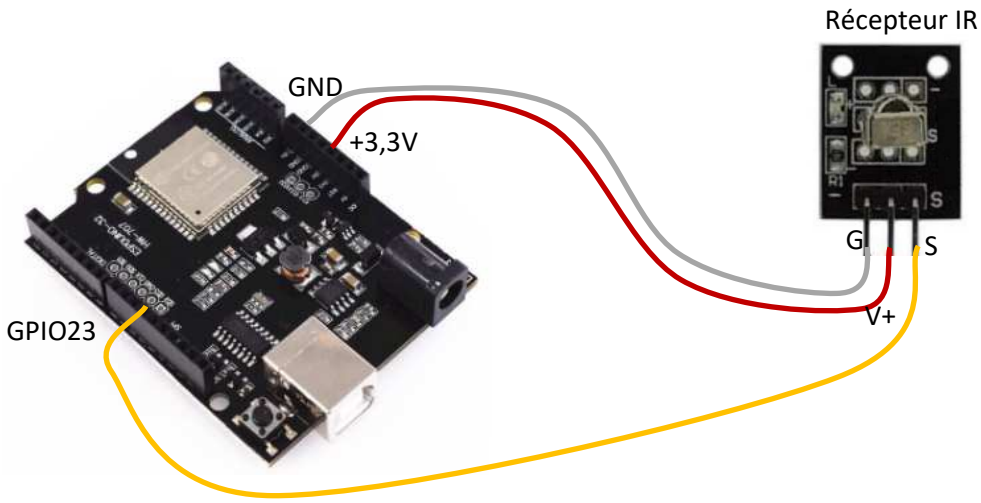
→ **ACQUERIR : Télécommande IR GT017 + récepteur IR 38 kHz**

1 – Mise en situation



2 – Plan de câblage / Montage

Raccorder avec la carte ESP au récepteur IR :



3 – Principe de fonctionnement de la télécommande IR et du récepteur IR

La télécommande envoie une suite de 0 et de 1 sous forme d'onde lumineuse infra rouge sur le récepteur qui transcrit ces 0 et 1 lumineux en 0 et 1 en tension. Cette suite de 0 et 1 forme une donnée hexadécimale qui dépend de la touche appuyée et de la télécommande utilisée.

Il existe plusieurs formats de données. Notre télécommande utilise le format NEC.

Dans le programme qui suit, la donnée *addr* dépend du type de télécommande et la donnée *data* dépend de la touche appuyée.

Voir le blog : https://techtotinker.com/2021/08/044-micropython-technotes-infrared-receiver/

4 – Programme (IR Exemple 3.py fourni sur le réseau)

ESP32 Micropython programme qui :

- **éteint la LED de la carte ESP32 si la touche 1 de la télécommande est appuyé,**
- **allume la LED si la touche 2 est appuyée,**
- **fait clignotter la LED di la touche 3 est appuyée.**

En parallèle, il affiche dans la console les données reçues de la télécommande.

```python
##################################################################
#                                                                #
# Programme qui renvoie dans la console 0 si le capteur est en   #
# face d'une zone claire et 1 s'il est en face d'une zone foncée #
#                                                                #
#    Connecter le récepteur sur 23 utilise les interuptions      #
#              utilise les timer                                 #
#   Besoin d'une bibliothèque ir_rx installée dans la carte ESP  #
#              CG 9/3/23                                         #
#    d'après https://techtotinker.com/2021/08/044-micropython    #
#            -technotes-infrared-receiver/                       #
#                                                                #
##################################################################

# Importation des bibliothèques
from machine import Pin, Timer
from ir_rx import NEC_16

# définition de la fonction d'interruption en cas de réception IR
def ir_callback(data, addr, ctrl):
    global ir_data
    global ir_addr
    if data > 0:
        ir_data = data
        ir_addr = addr
        # affiche les nouvelles données présentes issues du récepteur IR
        print('Data {:02x} Addr {:04x}'.format(data, addr))

# Définition de la fonction appelée par le timer
def timer_callback(timer):
    led.value( not led.value() ) # inverse l'état de la led

# définition du vecteur d'interruption lié à la réception IR
ir = NEC_16(Pin(23, Pin.IN), ir_callback)

# attachament de l'objet led à la broche 2 qui est mise en sortie
# (adresse de la Led sur la carte ESP32)
led = Pin(2, Pin.OUT)

# attachement de l'objet tim0 au timer utilisé
tim0 = Timer(0)

# initialisation des variables globales
isLedBlinking = False
ir_data = 0
ir_addr = 0

while True:
    if ir_data > 0: # si une nouvelle donnée est issue issue du récepteur IR
        if ir_data == 0x16:   # si touche télécommande appuyée est 1
            led.value(0) # éteint la LED
            if isLedBlinking==True:
                tim0.deinit() # arrête le timer
                isLedBlinking = False
        elif ir_data == 0x19: # si touche télécommande appuyée est 2
            led.value(1) # allume la LED
            if isLedBlinking==True:
                tim0.deinit() # arrête le timer
                isLedBlinking = False
        elif ir_data == 0x0d: # si touche télécommande appuyée est 3
            isLedBlinking = True
            # paramétrage et vecteur d'interruption du timer
            tim0.init(period=500, mode=Timer.PERIODIC, callback=timer_callback)
        ir_data = 0 # remet à 0 le nombre de nouvelles données reçues sur le capteur IR
```