



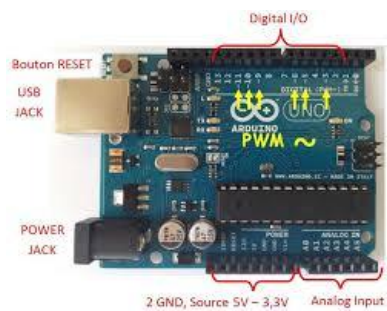
@attach_ayoub.py @robothings_fstm

Projet : Commande d'une LED et d'un Buzzer à Distance via Infrarouge (IR) avec Arduino

Ce projet vous permettra de contrôler une **LED** et un **buzzer** à distance en utilisant une **télécommande IR** et un **récepteur IR** connecté à un Arduino.

1. Matériel requis

- Arduino Uno/Nano/Mega
- Télécommande IR (TV, climatiseur, ou module IR spécifique)
- Module récepteur IR (KY-022 ou TSOP1738)
- LED + Résistance 220Ω
- Buzzer actif (ou passif avec PWM si nécessaire)
- Câbles de connexion
- Carte de prototypage (breadboard)



2. Principe de fonctionnement

1. L'utilisateur appuie sur un bouton de la **télécommande IR**.
 2. Le **récepteur IR** reçoit le signal et le transmet à l'Arduino.
 3. L'Arduino analyse le code reçu et active/désactive la LED ou le buzzer en fonction du bouton pressé.
-

3. Schéma de connexion

3.1 Connexion du récepteur IR (KY-022 ou TSOP1738)

Broche Récepteur IR	Connexion Arduino
VCC	5V
GND	GND
OUT	D2

3.2 Connexion de la LED

Broche LED	Connexion Arduino
Anode (+, longue patte)	Résistance 220Ω → D4
Cathode (-, courte patte)	GND

3.3 Connexion du Buzzer

Broche Buzzer	Connexion Arduino
+	D5
-	GND

4. Installation de la bibliothèque IRremote

Avant d'écrire le programme, installez la bibliothèque **IRremote** :

1. Ouvrez l'IDE Arduino.
 2. Allez dans **Outils** → **Gérer les bibliothèques**.
 3. Recherchez **IRremote by Armin Joachimsmeier** et installez-la.
-

5. Programme Arduino

Le code suivant permet d'allumer/éteindre la LED et d'activer/désactiver le buzzer avec une télécommande IR.

```
#include <IRremote.h> // Bibliothèque pour IR

#define RECV_PIN 2 // Broche du récepteur IR
#define LED_PIN 4 // LED connectée à D4
#define BUZZER_PIN 5 // Buzzer connecté à D5

bool ledState = false; // État de la LED
bool buzzerState = false; // État du buzzer

void setup() {
    Serial.begin(9600); // Démarrer la communication série
    IrReceiver.begin(RECV_PIN, ENABLE_LED_FEEDBACK); // Initialiser le récepteur
    IR
    pinMode(LED_PIN, OUTPUT);
    pinMode(BUZZER_PIN, OUTPUT);
    digitalWrite(LED_PIN, LOW);
    digitalWrite(BUZZER_PIN, LOW);
}

void loop() {
    if (IrReceiver.decode()) {
        unsigned long code = IrReceiver.decodedIRData.decodedRawData; // Lire le code
        reçu
        Serial.println(code, HEX); // Afficher le code en hexadécimal
        IrReceiver.resume(); // Préparer la réception du prochain signal

        // Remplacez ces codes par ceux de votre télécommande
        if (code == 0x1FE48B7) { // Exemple: Bouton "1" allume/éteint la LED
            ledState = !ledState;
            digitalWrite(LED_PIN, ledState);
        }
        else if (code == 0x1FEA857) { // Exemple: Bouton "2" active/désactive le buzzer
            buzzerState = !buzzerState;
            digitalWrite(BUZZER_PIN, buzzerState);
        }
    }
}
```

```
}  
}
```

6. Explication du code

- On initialise la bibliothèque **IRremote** et on assigne les broches de connexion.
 - On active la communication série pour afficher les codes IR reçus.
 - Lorsque la télécommande envoie un signal, l'Arduino capture son code et l'affiche dans le moniteur série.
 - On utilise **deux boutons** pour **allumer/éteindre la LED** et **activer/désactiver le buzzer**.
 - Les codes **0x1FE48B7** et **0x1FEA857** sont des exemples. **Vous devez remplacer ces valeurs par les codes réels de votre télécommande** (voir l'étape 7).
-

7. Trouver les codes IR de votre télécommande

Avant d'utiliser les boutons de votre télécommande, vous devez identifier leurs codes.

1. **Chargez ce programme dans votre Arduino :**

```
#include <IRremote.h>
```

```
#define RECV_PIN 2 // Broche du récepteur IR
```

```
void setup() {  
  Serial.begin(9600);  
  IrReceiver.begin(RECV_PIN, ENABLE_LED_FEEDBACK);  
}
```

```
void loop() {  
  if (IrReceiver.decode()) {  
    Serial.println(IrReceiver.decodedIRData.decodedRawData, HEX);  
    IrReceiver.resume();  
  }  
}
```

2. **Ouvrez le Moniteur Série (9600 bauds).**

3. Appuyez sur les boutons de la télécommande et notez les codes affichés.
 4. Remplacez les valeurs dans le programme principal.
-

8. Améliorations possibles

- **Ajouter plus de boutons** pour contrôler d'autres composants (ex: moteur, relais, etc.).
 - **Utiliser des modes PWM** pour contrôler l'intensité de la LED et la tonalité du buzzer.
 - **Ajouter un affichage LCD/OLED** pour voir quel périphérique est actif.
-

9. Conclusion

Ce projet est un excellent moyen de découvrir la **communication infrarouge avec Arduino** et permet de contrôler des périphériques à distance facilement. 🚀