Le Moniteur Série sur ESP32

Houssem-eddine LAHMER

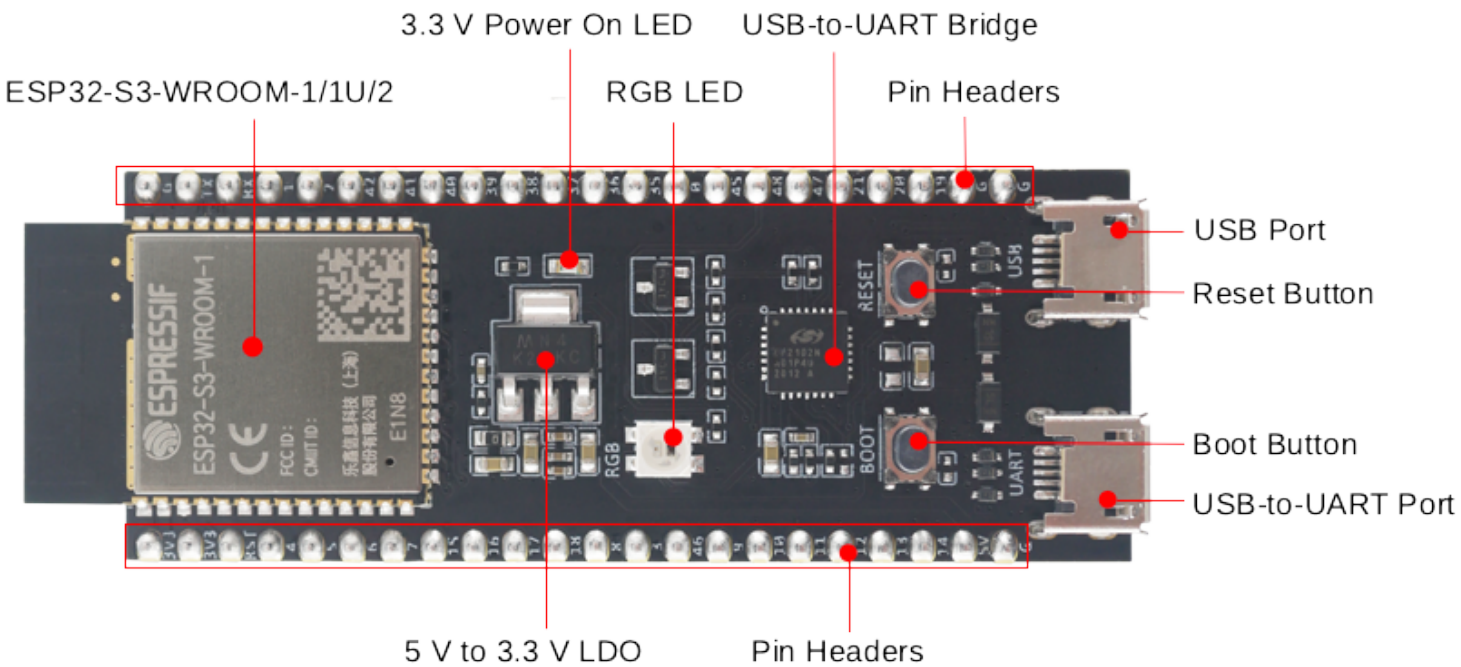
Plan du cours

Qu'est-ce que le Moniteur Série ?

- Le Moniteur Série est une fonctionnalité essentielle des environnements de développement (IDE)
- Il permet la communication entre un ordinateur et un microcontrôleur via une connexion série
- Applications principales :
  - Débogage de programmes
  - Affichage de données en temps réel
  - Envoi de commandes au microcontrôleur
  - Surveillance des capteurs et des périphériques
- Essentiel pour le développement sur ESP32 et autres microcontrôleurs

Communication Série sur ESP32

- L'ESP32 dispose de 3 interfaces UART (Universal Asynchronous Receiver/Transmitter)
- UART0 : Généralement utilisée pour la programmation et le débogage
- UART1 et UART2 : Disponibles pour d'autres périphériques
- Connexion via USB à l'ordinateur grâce au convertisseur USB-UART intégré à la carte



Initialisation du Moniteur Série

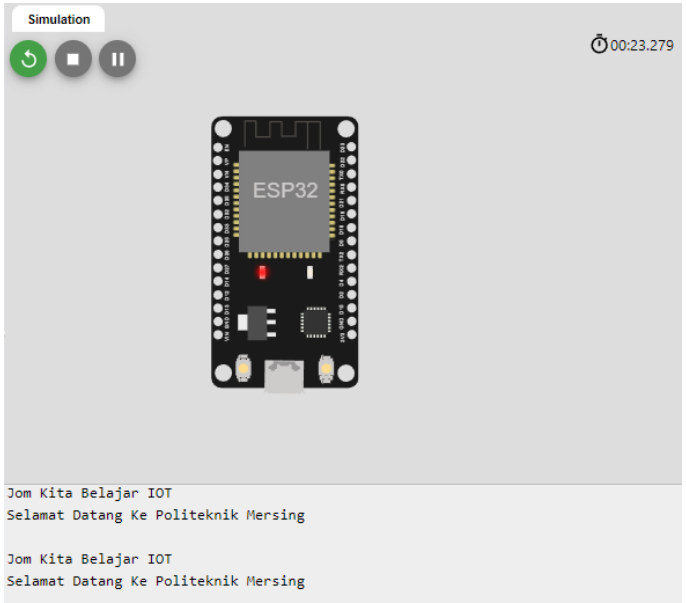
- Configuration dans la fonction `setup()`
- Vitesse de transmission spécifiée en bauds
  - Valeurs courantes : 9600, 115200 bps

```
1 void setup() {
2   // Initialise le moniteur s rie     9600 bauds
3   Serial.begin(9600);
4
5   // ou    vitesse plus   leve   pour un d bit plus rapide
6   // Serial.begin(115200);
7 }
```

- Important : La vitesse doit correspondre à celle configurée dans l'IDE

Configuration dans l'IDE Arduino

- Configuration du Moniteur Série dans l'IDE Arduino :
  1. Ouvrir le Moniteur Série via le menu ou icône
  2. Sélectionner la même vitesse que dans le code (ex : 9600 bauds)
  3. Choisir la configuration de fin de ligne appropriée
- Si les caractères sont illisibles, vérifier la vitesse de transmission



Interface du Moniteur Série

Fonctions d'Écriture de Base

Fonctions de sortie principales

Fonction	Description
`Serial.print(data)`	Affiche les données sans retour à la ligne
`Serial.println(data)`	Affiche les données avec retour à la ligne
`Serial.write(byte)`	Envoie des données brutes (octets)

```
1 void loop() {
2   Serial.print("Valeur: ");      // Sans retour    la ligne
3   Serial.println(42);           // Avec retour    la ligne
4   Serial.write(65);             // Envoie le caract re 'A'
      (ASCII 65)
5   delay(1000);
6 }
```

Formatage des Données

```
1  void loop() {
2    int valeur = 42;
3    float temperature = 23.5;
4
5    // Affichage de nombres en diff rents formats
6    Serial.print("D cimal: ");
7    Serial.println(valeur);
8
9    Serial.print("Hexad cimal: ");
10   Serial.println(valeur, HEX);
11
12   Serial.print("Binaire: ");
13   Serial.println(valeur, BIN);
14
15   Serial.print("Temp rature: ");
16   Serial.println(temperature, 2);  // 2 d cimales
17
18   delay(1000);
19 }
```

Fonctions de Lecture

Fonctions de lecture principales

Fonction	Description
`Serial.available()`	Retourne le nombre d'octets disponibles
`Serial.read()`	Lit un octet (retourne -1 si aucun disponible)
`Serial.readString()`	Lit une chaîne complète
`Serial.readStringUntil(char)`	Lit jusqu'à un caractère spécifique

```
1 void loop() {
2   // V rifie si des donn es sont disponibles
3   if (Serial.available() > 0) {
4     // Lit un caract re
5     char c = Serial.read();
6
7     // Echo du caract re re u
```

Lecture de Commandes Complètes

```
String commande = "";
boolean commandeComplete = false;

void setup() {
  Serial.begin(9600);
  Serial.println("Pr t    recevoir des commandes:");
}

void loop() {
  // Lecture caract re par caract re
  while (Serial.available() > 0) {
    char caractere = Serial.read();

    // Ajoute    la commande si ce n'est pas un retour
    chariot
    if (caractere != '\n' && caractere != '\r') {
      commande += caractere;
    }
    // Fin de la commande
    else if (commande.length() > 0) {
      commandeComplete = true;
```

Récapitulatif

- Le moniteur série est un outil essentiel pour :
  - Débogage de code
  - Communication avec l'ESP32
  - Affichage de données des capteurs
  - Contrôle interactif
- Fonctions principales :
  - `Serial.begin()` - Initialisation
  - `Serial.print()/println()` - Affichage
  - `Serial.available()/read()` - Lecture
- Applications avancées :
  - Journalisation
  - Interfaces utilisateur textuelles
  - Débogage de performance
  - Communication multi-périphérique

Ressources et Documentation

- Documentation officielle ESP32 :
  - `https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/uart.html`
- Tutoriels complémentaires :
  - Arduino - Serial : `https://www.arduino.cc/reference/en/language/functions/communication/serial/`
  - ESP32 - Communication série : `https://randomnerdtutorials.com/esp32-serial-communication-arduino-ide/`
- Simulateurs en ligne :
  - Wokwi : `https://wokwi.com/`
  - TinkerCAD : `https://www.tinkercad.com/`
- Outils utiles :
  - Analyseur série : PuTTY, CoolTerm, Serial Monitor Studio
  - Visualisateurs graphiques : Serial Plotter, Processing