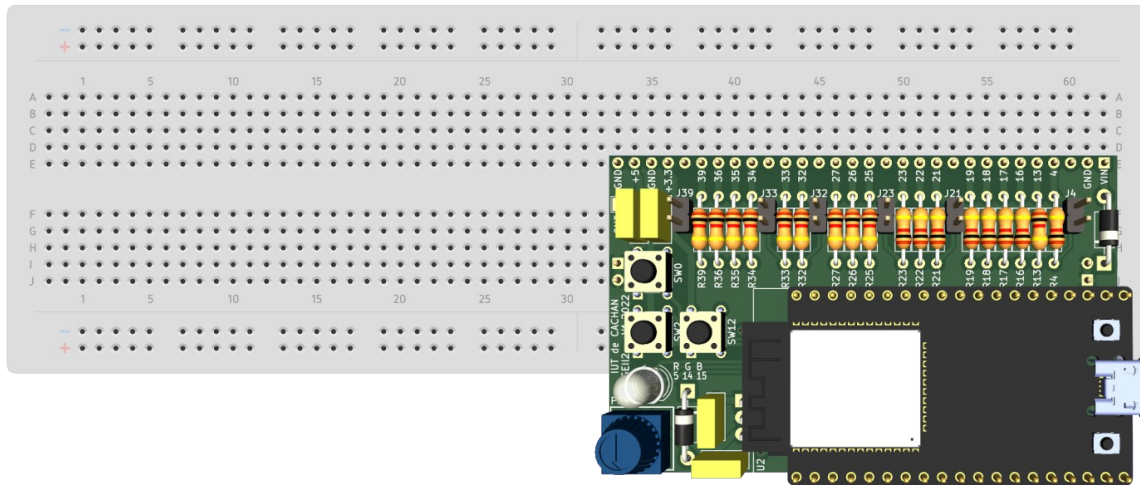


La plaque de test avec l'« Espressif ESP32 Dev Module »



ESP32 sur plaque de test

Vous rendrez un compte-rendu comportant les **documents réponses** et seulement les **programmes demandés** (avec des commentaires). Vous copierez ces programmes dans un document que vous imprimerez en fin de séance

Tous les documents se trouvent dans Y:_ER_S2 et aussi depuis le serveur du département **profge2** puis **semestre 2** et **Page E&R-S2** ce qui vous permet d'y accéder depuis internet. Pour cette séance, consultez en particulier le dossier **esp32\doc**.

1. En utilisant le tutoriel **tuto_esp32.pdf**, créer un projet pour l'ESP32. Notez bien que les logiciels **VSCode** et **PlatformIO** sont normalement déjà installés sur votre poste. Vous devez faire les étapes d'installation des logiciels si vous souhaitez travailler avec votre ordinateur personnel.

2. Connectez le câble USB et vérifiez que vous pouvez programmer le microcontrôleur avec un programme vide.

3. Effectuez le tutoriel pour la communication série. Modifiez le programme pour écrire votre prénom toutes les 5 secondes sur le terminal.

Commentez le programme et copiez-le dans le document à rendre. N'oubliez pas de sauvegarder le document.

4. En utilisant le tutoriel pour les entrées TOR, câblez un bouton poussoir sur l'entrée 23 de l'ESP32 avec une résistance de pull-up externe (on pourra prendre 10 kΩ par exemple). Complétez le **DR1**. Vérifiez le fonctionnement grâce au terminal.

Retirez la résistance et activez le pull-up interne. Vérifiez que le programme fonctionne toujours.

Commentez le programme et copiez-le dans le document à rendre. N'oubliez pas de sauvegarder le document.

Conservez le câblage du bouton pour la question suivante.

5. En utilisant le tutoriel pour les sorties TOR, câblez une led rouge avec une résistance de $470\ \Omega$ sur la sortie 19 de l'ESP32. Complétez le **DR2**. Utilisez le bouton câblé précédemment pour allumer et éteindre la led.

Commentez le programme et copiez-le dans le document à rendre. N'oubliez pas de sauvegarder le document.

6. A partir du schéma de la carte, ***schémaMaquette.pdf***, repérez sur quelles broches sont câblés les trois boutons poussoirs. En déduire s'ils doivent être configurés en pull-up ou pull-down, interne ou externe. Écrivez un programme pour valider le fonctionnement des trois boutons poussoirs avec le terminal. Ce programme n'est pas à rendre.

7. Repérez sur le schéma la led RGB. En déduire que son fonctionnement est similaire au fonctionnement de 3 leds. Quelles sorties de l'ESP32 sont utilisées ? Écrivez un programme pour que chacun des trois boutons commande l'allumage d'une led différente.

Commentez le programme et copiez-le dans le document à rendre. N'oubliez pas de sauvegarder le document.

8. En utilisant le tutoriel pour les entrées analogiques, câblez un potentiomètre sur l'entrée 33 de l'ESP32. Complétez le **DR3**. Écrivez un programme pour valider le fonctionnement du potentiomètre grâce au terminal.

Commentez le programme et copiez-le dans le document à rendre. N'oubliez pas de sauvegarder le document.

9. Repérez sur le schéma le potentiomètre. Sur quelle entrée est-il câblé ? Écrivez un programme pour valider son fonctionnement avec le terminal. Ce programme n'est pas à rendre.

10. En utilisant le tutoriel sur les sorties PWM, écrivez un programme qui génère un signal PWM avec une fréquence de 500Hz et un rapport cyclique de 25 % sur la sortie 19 de l'ESP32. Vérifiez en visualisant le signal sur l'oscilloscope. Modifiez le programme pour que le potentiomètre règle le rapport cyclique. Ce programme n'est pas à rendre.

11. Modifiez le programme précédent pour envoyer le signal généré précédemment sur la led verte de la carte. Ce programme n'est pas à rendre.

12. Écrivez un programme qui utilise les 3 boutons, le potentiomètre et la led rgb pour régler la couleur de la led. On pourra, par exemple, régler le rapport cyclique du rouge quand on appuie sur un premier bouton, régler le vert quand on appuie sur un deuxième bouton et régler le bleu quand on appuie sur le troisième bouton.

Commentez le programme et copiez-le dans le document à rendre. N'oubliez pas de sauvegarder le document.

13. En utilisant le tutoriel pour les bibliothèques Arduino, ajoutez la bibliothèque suivante :

Grove - LCD RGB Backlight by Seeed Studio

à votre projet. Elle va nous permettre de faire fonctionner un écran LCD à 2 lignes et 16 colonnes similaire à l'écran que vous aviez sur la carte du *Game/Trophy*. Cette bibliothèque permet également de piloter la couleur du rétroéclairage. L'écran que nous utilisons ne possède pas cette possibilité.

Il fonctionne avec une liaison I2C. C'est une liaison série synchrone que nous ne détaillerons pas. Elle nécessite deux fils pour l'alimentation, un fil pour l'horloge, SCL et un fil pour les données, SDA.

Le câblage de l'écran doit être le suivant :

noir	GND
rouge	+5V ou +3,3V
blanc	SDA
jaune	SCL

En vous aidant du brochage de l'ESP32 (voir fichier **Pin Layout.jpg**), repérez les broches SDA et SCL et complétez le **DR4**. Testez le fonctionnement du programme d'exemple *HelloWorld*. Ce programme n'est pas à rendre.

14. Reprenez le programme de la question 12 et ajoutez l'affichage sur l'écran LCD conforme à l'écran ci-dessous avec les xxx remplacés par les pourcentages respectifs de rouge, de vert et de bleu :

	I	U	T		d	e		C	A	C	H	A	N		
R	x	x	x	%	V	x	x	x	%	B	x	x	x	%	

Commentez le programme et copiez-le dans le document à rendre. N'oubliez pas de sauvegarder le document.

15. Nous allons maintenant récupérer un caractère tapé dans le terminal. Le caractère R majuscule mettra le pourcentage de rouge à 100 % et le caractère r minuscule mettra le pourcentage de rouge à 0 %.

La méthode `Serial.available()` renvoie vrai si un caractère a été reçu. La méthode `Serial.read()` permet de lire le caractère. On peut donc utiliser le code suivant :

```
if (Serial.available()) {  
    char caractere = Serial.read();  
    if (caractere == 'R') {  
        // code pour passer le rouge à 100 %  
    } else if (caractere == 'r') {  
        // code pour passer le rouge à 0 %  
    }  
}
```

Faites fonctionner le programme complet.

Commentez le programme et copiez-le dans le document à rendre. N'oubliez pas de sauvegarder le document.

16. Pour terminer, nous allons remplacer la liaison filaire avec le terminal par une liaison Bluetooth avec un appareil Android. Les appareils Apple ne sont pas compatibles.

Ajoutez la ligne suivante dans les « *include* » :

```
#include <BluetoothSerial.h>
```

Puis la ligne pour créer l'objet « liaison série Bluetooth » :

```
BluetoothSerial SerialBT;
```

Dans la fonction `setup`, ajoutez la ligne :

```
SerialBT.begin("Votre nom");
```

« Votre nom » va vous permettre d'identifier votre ESP32 et de le distinguer de ceux de vos voisins quand vous allez rechercher un appareil Bluetooth.

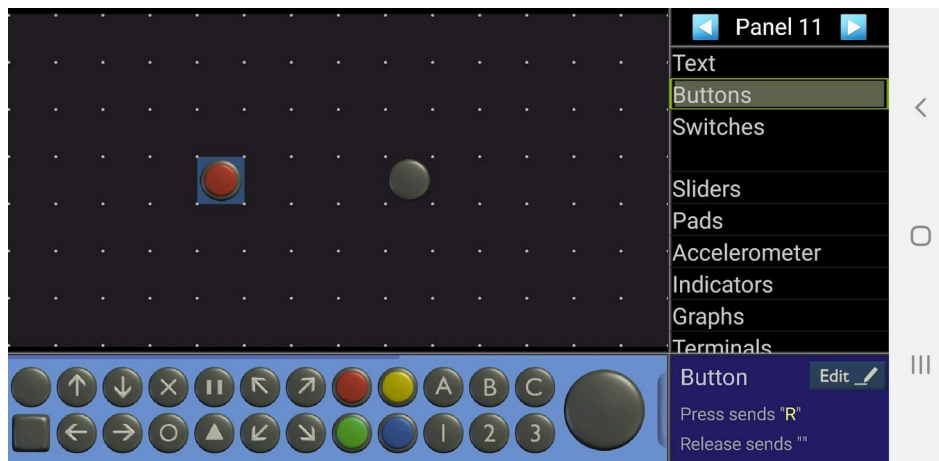
Dans la fonction `loop`, remplacez `Serial` par `SerialBT`.

Installez l'application « Bluetooth Electronics » sur votre appareil Android.

Créez un nouveau « Panel » avec un bouton qui envoie le caractère R majuscule et un bouton qui envoie le caractère r minuscule.

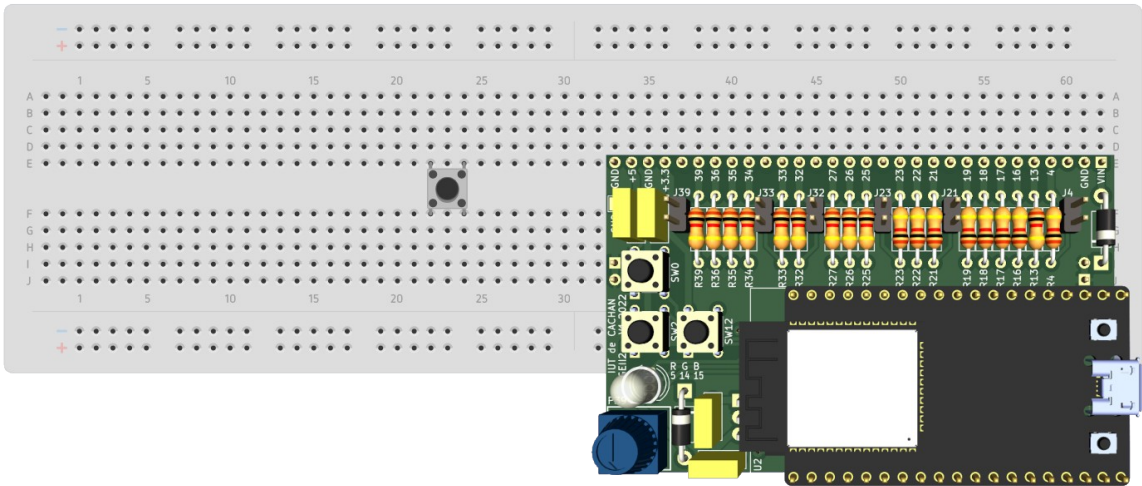
Appairez votre ESP32 avec votre appareil Android. Vous devriez pouvoir allumer ou éteindre le rouge de la led RGB. Ce programme n'est pas à rendre.

Ci-dessous le « panel » de « Bluetooth Electronics » :

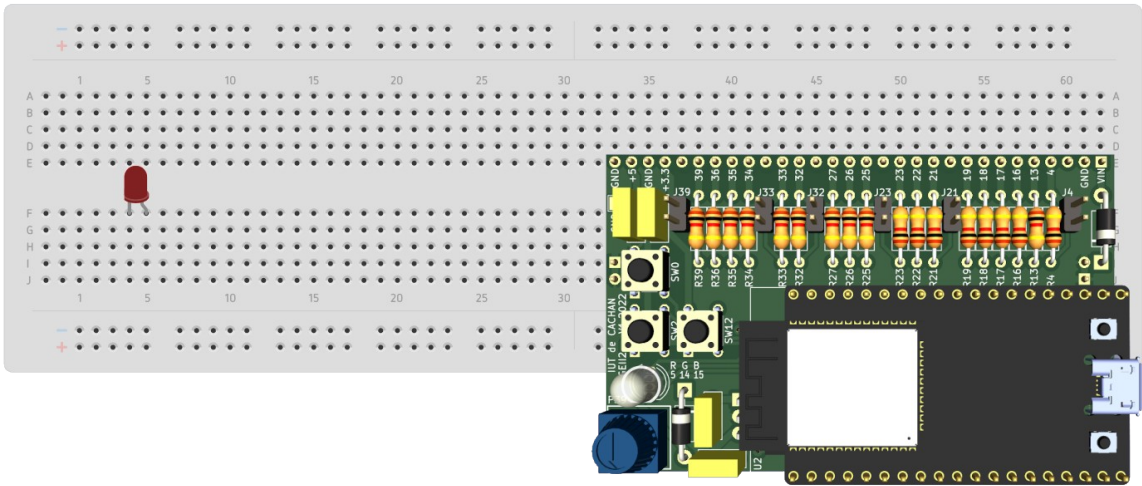


S'il vous reste du temps, vous pouvez utiliser les « Sliders » pour piloter les pourcentages RGB de la led RGB avec votre appareil Android.

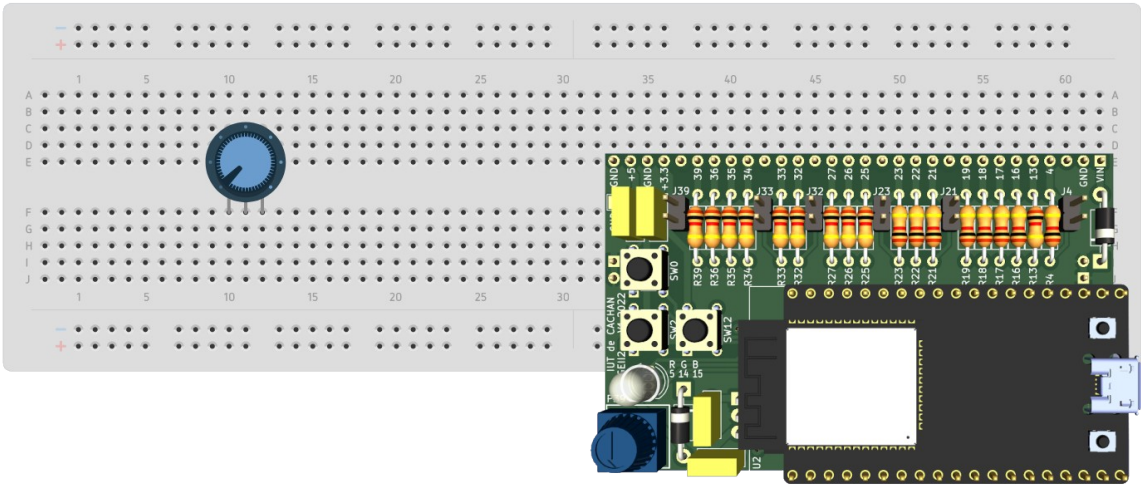
Document Réponse 1 – Bouton poussoir avec pull-up externe



Document Réponse 2 – Led rouge



Document Réponse 3 – Potentiomètre



Document Réponse 4 – Écran LCD

