BlueRC : télécommande IR avec smartphone et ESP32

adaptative et universelle

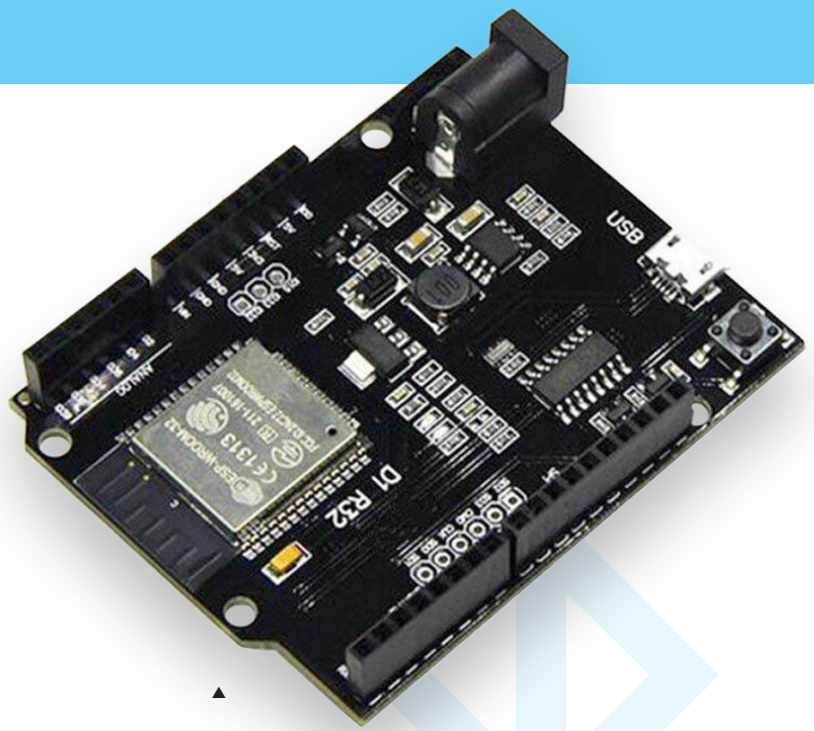


Figure 1. Carte de développement WeMos D1 R32 ESP32 Wi-Fi/ Bluetooth.

Xin Wang (Allemagne)

Il fut un temps où les émetteurs infrarouges sur les téléphones portables étaient courants, mais ce n'est plus le cas. Ce projet propose une solution : utilisez votre téléphone portable avec une connectivité Bluetooth pour contrôler une carte ESP32 qui peut à la fois envoyer des commandes infrarouges et en apprendre de nouvelles.

Je souhaitais commander mes appareils à infrarouge avec mon téléphone. J'ai fait quelques recherches pour comparer plusieurs projets sur internet. La plupart sont basés sur des codes prédéfinis, ce qui signifie que les codes IR devaient être prédéfinis et stockés dans un tableau. Je voulais créer une télécommande à apprentissage qui ne nécessiterait aucune connaissance préalable de codes. Les codes doivent progressivement être mis à jour par le circuit en temps réel.

Figure 2. Interface utilisateur de l'application Android BlueRC V1.1.

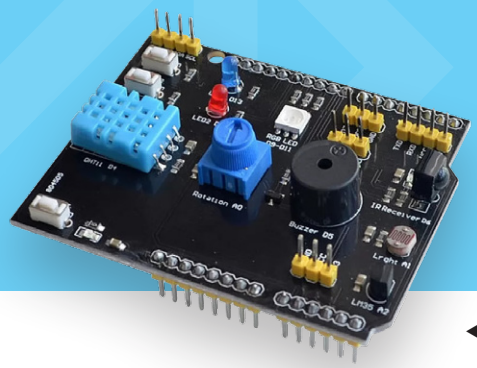


Figure 3. Le shield Keyestudio Easy Module V1 pour Arduino. Source : flyrobo.in.

Matériel

Ce projet repose sur deux éléments principaux :

- Une carte à microcontrôleur (**figure 1**) avec un récepteur infrarouge connecté pour apprendre les codes existants d'autres télécommandes, et un émetteur infrarouge pour effectuer la tâche principale en envoyant des commandes à votre téléviseur, boîtier décodeur, etc. La carte stocke également les codes appris.
- Un téléphone Android pour exécuter une application, qui sert simplement d'interface utilisateur pour la carte à microcontrôleur. L'application vous offre une interface de télécommande (**figure 2**), qui vous permet d'appuyer sur les boutons dont vous avez besoin pour contrôler vos appareils infrarouges. Vous pouvez également lancer la séquence d'apprentissage en appuyant sur une touche de cette interface utilisateur, mais la procédure d'apprentissage elle-même est exécutée sur la carte à microcontrôleur.

J'ai choisi le Bluetooth pour la communication entre le téléphone et l'émetteur IR, pour sa faible consommation d'énergie et sa faible latence par rapport au wifi.
J'ai cherché une solution disponible sur le marché qui répondrait à toutes les exigences matérielles, mais sans succès. Donc, en ce qui concerne le microcontrôleur, j'ai choisi une carte ESP32, puisqu'elle dispose à la fois de Bluetooth et de wifi. La carte de développement D1 R32 de WeMos a une disposition des broches compatible avec Arduino UNO et fonctionne bien avec l'EDI Arduino.
J'ai ajouté un *shield* à capteurs multiples (**figure 3**). Ce *shield* a une pléthore d'autres capteurs : un capteur d'humidité, un buzzer, un potentiomètre, une cellule photoélectrique, une LED RGB, et quelques boutons, entre autres — tous ces éléments offrent une grande polyvalence pour de futures idées de domotique. J'ai utilisé les capteurs pour afficher la température et l'humidité dans l'application Android, mais notre capteur intégré le plus important est le récepteur infrarouge !
Alors que le *shield* a un récepteur IR intégré, il ne dispose pas d'un émetteur, j'ai donc dû construire mon propre module de commande/émetteur rapide et pratique. Il faut seulement 3 composants - une douille à 3 broches, un transistor BC547 NPN, et une LED IR, montrés très simplement assemblés dans la **figure 4**. Après soudure,

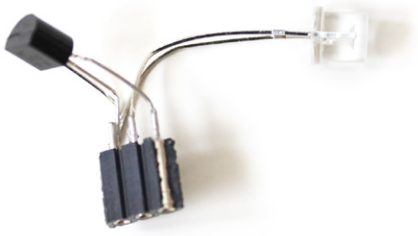


Figure 4. Mon module de commande/transmission infrarouge «maison».

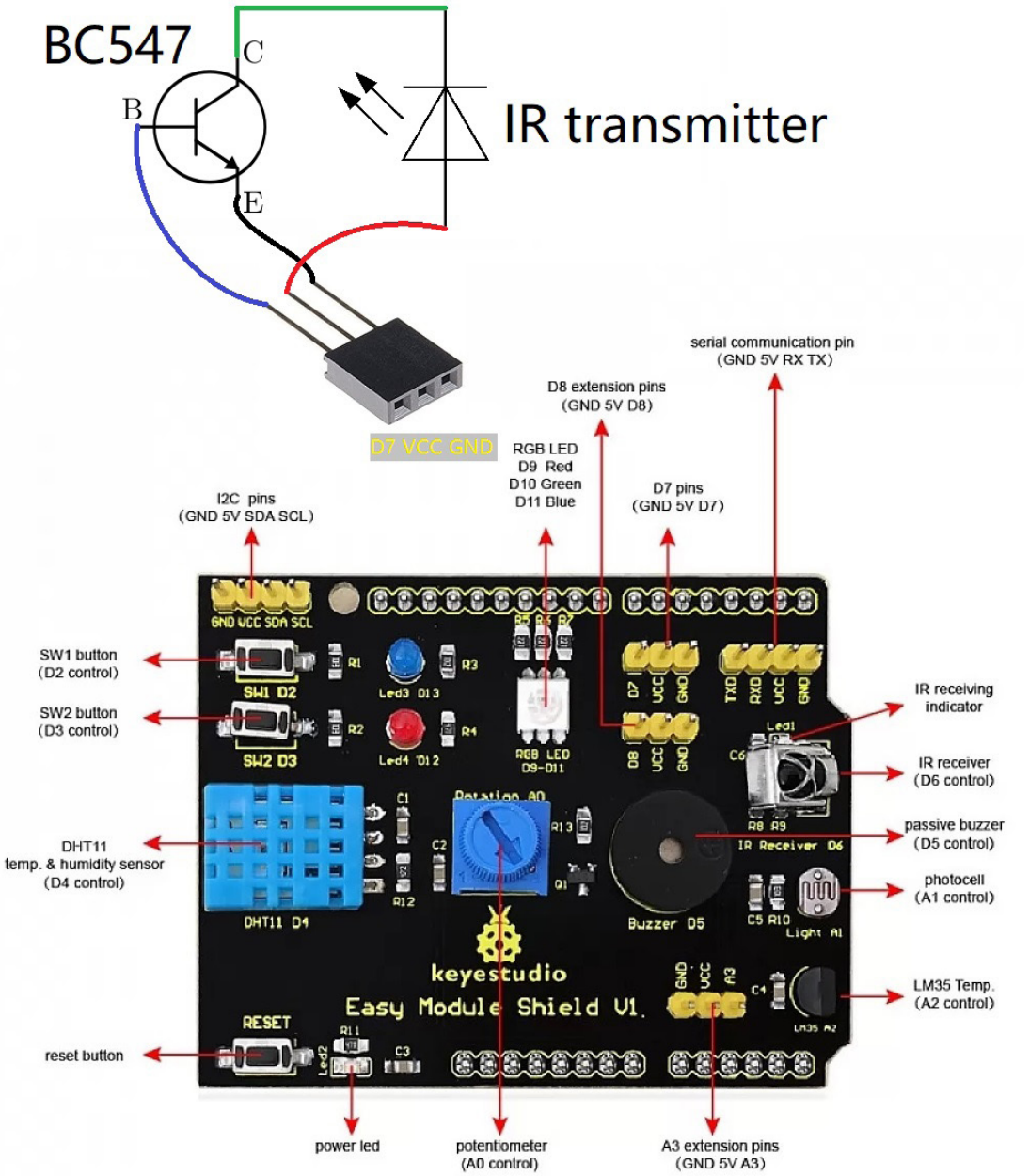


Figure 5. Le shield et la façon dont j'y ai connecté mon module de commande/transmission IR. Source : flyrobo.in.

j'ai branché le connecteur dans la broche du port D7 sur le *shield* du capteur, qui utilise la broche du port D14 comme sortie pour contrôler mon émetteur IR, alors que le récepteur IR du *shield* utilise la broche du port D27 comme entrée par défaut.

La **figure 5** présente le schéma de base.

Logiciel

Mon logiciel est relativement simple : l'ESP32 sert de serveur Bluetooth, et l'application Android sert de client. L'ESP32 attend simplement les messages Bluetooth de l'application. La communication Bluetooth entre les deux appareils est limitée, car les pressions sur les touches de la télécommande – ainsi que la commande pour commencer la séquence d'apprentissage (^) – de l'application Android sont envoyées à l'ESP32 sous la forme de caractères uniques.

J'ai d'abord utilisé la bibliothèque IR standard pour Arduino. Elle fonctionnait bien, mais n'était pas totalement compatible avec l'ESP32, car ce dernier utilise un émetteur-récepteur de télécommande (RMT) propriétaire avec beaucoup de protocoles et de logique de modulation intégrés pour les signaux infrarouges. Après de nombreux essais et erreurs, j'ai finalement trouvé une bibliothèque IR entièrement fonctionnelle pour l'ESP32, *IRremoteESP8266*, par Ken Shirriff et al [1]. Merci à Ken. À l'origine, je voulais stocker les codes IR dans l'EEPROM, mais lorsque j'ai écrit mon micrologiciel, j'ai réalisé que l'ESP32 n'avait pas d'EEPROM. Cependant, il a la possibilité de sauvegarder des données de façon permanente dans

la mémoire flash grâce à la bibliothèque *Preferences.h* [5]. Comme ma télécommande d'origine avait un codage RC5, je n'ai implémenté que le schéma de codage RC5. Vous devriez peut-être modifier le croquis Arduino si votre télécommande utilise un autre protocole.

Programmation

Pour la carte WeMos, téléchargez le croquis Arduino *BlueRC* sur [2]. Il s'agit d'un seul fichier *.ino* que vous pouvez télécharger avec l'EDI Arduino.
Pour configurer l'application Android, suivez les étapes suivantes :

1. Sur votre appareil Android, téléchargez *BlueRC-app-V1.1.apk* depuis la page GitHub [3]. Vous devez configurer votre téléphone pour qu'il autorise le téléchargement d'applications à partir de sources externes (autres que Google Play Store).
2. Mettez sous tension la carte WeMos sur laquelle est installé le micrologiciel BlueRC *.ino*.
3. Lancez l'application Android. Si le Bluetooth n'est pas actif, l'application demande une autorisation ; appuyez sur *Accept*.
4. Tapez sur le « kebab menu » en haut à droite, puis sur *Connect BT Device*. S'il s'agit de votre premier appairage avec l'ESP32, tapez sur *Scan for devices* et attendez que *ESP32BlueRC-Shield* apparaisse sous *Other Available Devices*. Tapez dessus, puis confirmez l'appairage.
5. Dans la barre d'état en haut, vous devriez maintenant voir *connected:ESP32BlueRC-Shield*.

Une fois que vous avez apparié votre matériel, il n'est pas nécessaire d'effectuer à nouveau le processus de recherche – il suffit d'appuyer sur *ESP32BlueRC-Shield* sous *Paired Devices* à l'avenir.
Une fois l'application installée et fonctionnelle, vous pouvez maintenant suivre le processus d'apprentissage :

1. Appuyez sur la touche LEARN dans l'interface utilisateur, et l'application répondra par « *Learning begins, please press a button within 10 seconds...* » L'application Android envoie alors le caractère spécial « ^ » à l'ESP32, puis la LED bleue du *shield* s'allume.
2. Sur l'interface utilisateur, appuyez sur la touche que vous voulez indiquer à la carte ESP32. Le téléphone envoie le caractère correspondant à l'ESP32, qui attend alors l'étape suivante.
3. Pointez votre ancienne télécommande infrarouge vers la carte ESP32 et appuyez sur le bouton correspondant de la télécommande. Le code IR de ce bouton sera automatiquement stocké dans la mémoire de l'ESP32.
4. Répétez les étapes 1 et 2 pour toutes les touches que vous souhaitez enregistrer.
5. Fermez l'application en appuyant sur *Menu → Exit program*.

Voilà, c'est fait. Vous pouvez maintenant contrôler vos appareils infrarouges à partir de votre appareil BlueRC, grâce à l'application sur votre téléphone Android.
Pour résoudre les problèmes liés à l'ESP32, vous pouvez l'éteindre et le rallumer.
Vous pouvez visionner une vidéo de démo [4] sur ma chaîne YouTube.

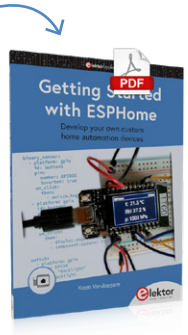
220103-04

Des questions, des commentaires ?

Contactez Elektor (redaction@elektor.fr).

Produits

- **Dogan and Ahmet Ibrahim, *The Official ESP32 Book* (PDF, en anglais, SKU 18330)**
  https://elektor.fr/18330
- **Koen Vervloesem, *Getting Started with ESPHome* (PDF, en anglais, SKU 19739)**
  https://elektor.fr/19739



LIENS

[1] Bibliothèque IRremoteESP8266 : https://www.arduino.cc/reference/en/libraries/irremoteesp8266/
[2] Le croquis Arduino BlueRC pour ESP32: https://elektor.link/BlueRCESP32
[3] Application Android BlueRC : https://elektor.link/BlueRCApp
[4] Le système BlueRC - vidéo : https://youtu.be/Kb-TdF84Oz4
[5] ESP32 Save Data Permanently using Preferences Library:
https://randomnerdtutorials.com/esp32-save-data-permanently-preferences/