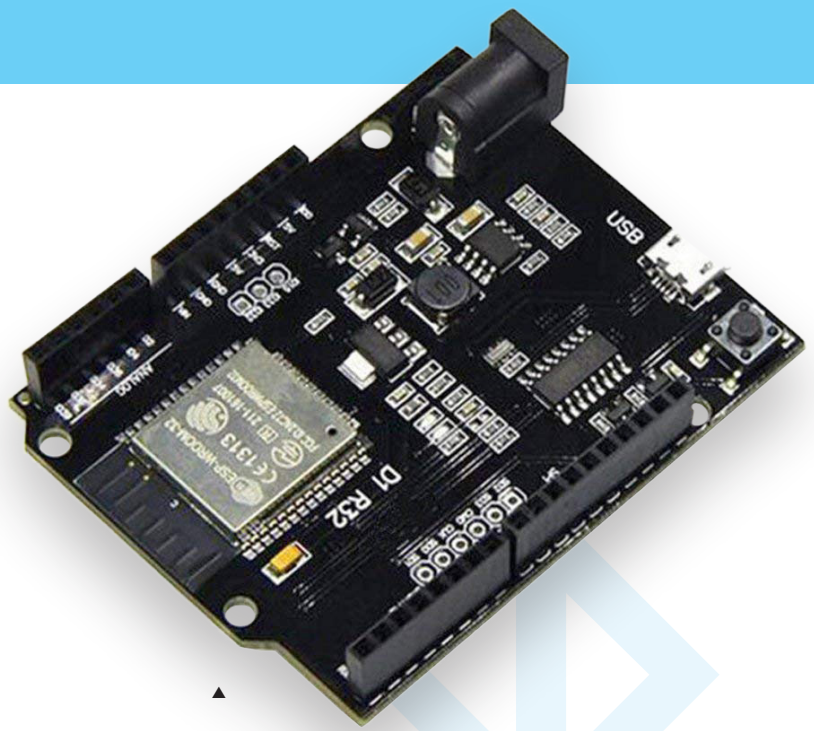
BlueRC:

IR-afstandsbediening met smartphone en ESP32

universeel en aanpasbaar



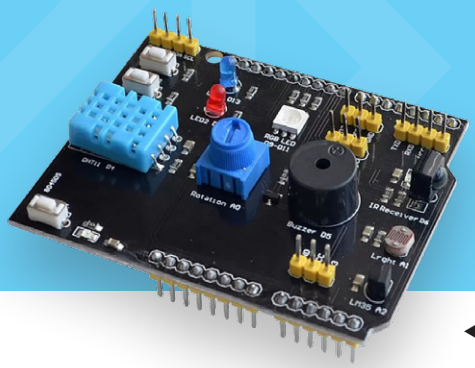
Figuur 1. WeMos D1 R32 ESP32 WiFi/Bluetooth-ontwikkelboard.

Xin Wang (Duitsland)

Ooit waren IR-zenders op mobiele telefoons heel gebruikelijk, maar dat is nu niet meer het geval. Dit project biedt een oplossing: gebruik je mobiele telefoon met zijn Bluetooth-connectiviteit om een ESP32-board aan te sturen dat zowel bekende infraroodcommando's kan versturen als nieuwe kan leren.

Ik wilde mijn IR-gestuurde apparaten ook kunnen bedienen met mijn telefoon. Daarom deed ik wat onderzoek en vergeleek verschillende projecten op het web. De meeste daarvan werkten met voorgedefinieerde codes, wat betekent dat de betreffende IR-codes vooraf bekend moesten zijn en in een matrix moesten worden opgeslagen. Ik wilde een zelflerende afstandsbediening bouwen waarvoor geen voorkennis van de codes van bestaande afstandsbedieningen nodig was. De codes moesten in real time en 'tijdens bedrif' door de schakeling worden bijgewerkt.

Figuur 2. Gebruikersinterface van de BlueRC V1.1 Android-app.



Figuur 3. Het Keyestudio Easy Module Shield V1 voor Arduino (bron: flyrobo.in).

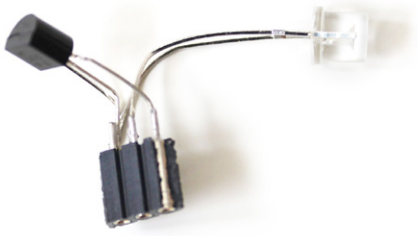
De hardware

Er zijn slechts twee belangrijke bouwstenen voor dit project:

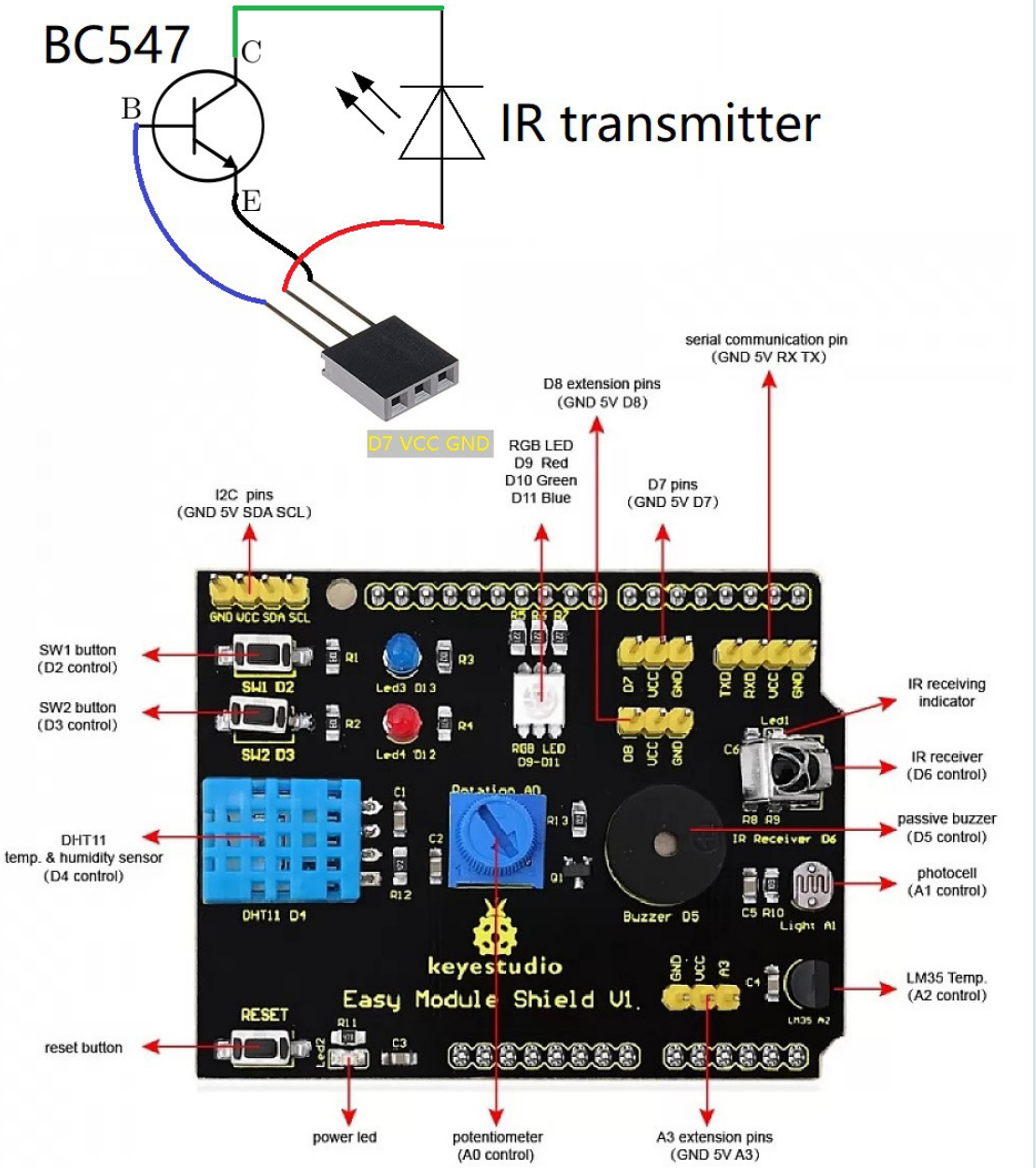
- Een microcontroller-board (**figuur 1**) met een aangesloten infrarood-ontvanger voor het aanleren van bestaande codes van andere afstandsbedieningen, en een infrarood-zender voor de hoofdtaak: het verzenden van commando's naar de TV, set-top box enzovoort. Het board slaat ook de geleerde codes op.
- Een Android-telefoon om een app te draaien die enkel fungeert als de gebruikersinterface voor het microcontroller-board. De app voorziet in een afstandsbediening op het display (**figuur 2**) waarop je de toetsen kunt indrukken die nodig zijn om je IR-gestuurde apparaten te bedienen. Je kunt de leerprocedure starten met een toetsdruk in deze gebruikersinterface, maar die leerprocedure zelf wordt uitgevoerd op de microcontroller-print.

Ik heb Bluetooth gekozen voor de communicatie tussen de telefoon en de IR-zenderhardware, vanwege het lage vermogen en de geringe latentie, vergeleken met WiFi. Een commercieel verkrijgbare oplossing die aan alle hardwarevoorwaarden voldeed bleek helaas onvindbaar: niets was 100% geschikt. Voor de microcontroller koos ik daarom voor een ESP32-board, omdat dat zowel Bluetooth als WiFi aan boord heeft. Het D1 R32 ontwikkelboard van WeMos heeft een Arduino UNO-compatibele pin-layout en werkt goed met de Arduino IDE.

Ik voegde er een shield met meerdere sensoren (**figuur 3**) aan toe. Dit shield heeft talloze zaken aan boord, zoals een vochtigheidssensor, een zoemer, een potentiometer, een fotocel, een RGB-LED en een paar knoppen – een heleboel veelzijdigheid ten behoeve van toekomstige domotica-projecten. Ik heb de sensoren gebruikt om de temperatuur en vochtigheid in de Android-app weer te geven, maar de belangrijkste sensor op het shield is natuurlijk de infrarood-ontvanger!
Hoewel het shield een ingebouwde IR-ontvanger heeft, is er geen zender, dus moest ik zelf een driver/zender in elkaar knutselen. Deze bestaat uit slechts 3 componenten – een 3-polige busstrip, een BC547 NPN-transistor en een IR-LED, gemonteerd als in **figuur 4**. Na het solderen prikte ik de busstrip op poort-pin D7 op het sensor-shield, dat poort D14 gebruikt als uitgang om mijn IR-zender aan te



Figuur 4. Mijn zelfbouw IR driver/zender-module.



Figuur 5. Het shield en de aansluiting van de IR driver/zender-module (bron: flyrobo.in).

sturen, terwijl de IR-ontvanger van het schild standaard poort D27 gebruikt als ingang.

In het 'schema' van **figuur 5** kun je zien hoe ik een en ander heb aangesloten.

De software

De software is relatief eenvoudig: de ESP32 fungeert als Bluetooth-server, en een Android-app doet dienst als client. De ESP32 wacht gewoon op Bluetooth-berichten van de Android-app. De Bluetooth-communicatie tussen de twee apparaten is minimaal, omdat toetsaanslagen van de afstandsbediening – en het commando om de leercurve te beginnen ("^") – door de Android-app als afzonderlijke tekens naar de ESP32 worden gestuurd.

Eerst probeerde ik de standaard IR-bibliotheek voor Arduino. Die werkte prima, maar wordt niet volledig ondersteund op de ESP32, omdat deze gebruik maakt van een eigen Remote Control Transceiver (RMT) in de vorm van onboard-hardware, met veel protocol en modulatielogica ingebouwd voor IR-signalen. Na veel vallen en opstaan vond ik eindelijk een goed werkende IR-bibliotheek voor de ESP32 (*IRremoteESP8266*) van Ken Shirriff et al [1]. Bedankt, Ken.

Oorspronkelijk wilde ik de IR-codes opslaan in EEPROM, maar toen ik de firmware schreef, realiseerde ik me dat de ESP32 helemaal geen EEPROM heeft. Het heeft echter wel de mogelijkheid om gegevens permanent op te slaan in het flash-geheugen via de *Preferences.h*-bibliotheek [5]. Omdat mijn originele afstandsbediening een RC5-exemplaar was, heb ik alleen de RC5-codering geïmplementeerd. Eventueel moet je de Arduino-sketch aanpassen als je eigen afstandsbediening een ander protocol gebruikt.

Programmering

Voor het WeMos-board kun je de BlueRC Arduino-sketch van [2] downloaden. Het is een enkel .ino-bestand dat je kunt uploaden met de Arduino IDE.
Volg deze stappen om de Android-app te installeren:

1. Download op je Android-toestel *BlueRC-app-V1.1.apk* van de GitHub pagina op [3]. Je moet je smartphone zo instellen dat het downloaden van apps van externe bronnen (anders dan Google Play Store) is toegestaan.
2. Schakel het WeMos-board met de *BlueRC.ino*-firmware in.
3. Start de Android-app. Als Bluetooth niet actief is, vraagt de app om toestemming; tik op *Accept*.
4. Tik op het "kebab menu" rechtsboven en vervolgens op *Connect BT Device*. Als dit de eerste keer is dat je een koppeling met de ESP32 maakt, tik dan op *Scan for Devices* en wacht tot het ESP32BlueRC-Shield verschijnt onder *Other Available Devices*. Tik hierop en bevestig de koppeling.
5. In de statusbalk bovenaan moet nu *connected:ESP-32BlueRC-Shield* staan.

Zodra je de hardware hebt gekoppeld, hoef je het scanproces niet opnieuw uit te voeren – tik voortaan gewoon op *ESP32BlueRC-Shield* onder *Paired Devices*. Nu de app is geïnstalleerd en werkt, kun je de codes gaan aanleren:

1. Tik lang op de LEARN-knop in de UI; de app zal reageren met "Learning begins, please press a button within 10 seconds..." De Android-app stuurt dan het speciale teken "^" naar de ESP32, waarna de blauwe LED op het schild oplicht.
2. Tik in de gebruikersinterface op de knop waaronder het ESP32-board de code moet aanleren. De smartphone zal het corresponderende teken naar de ESP32 sturen, waarna deze op de volgende stap wacht.
3. Richt je bestaande IR-afstandsbediening op het ESP32-board en druk op de betreffende knop. De IR-code voor deze knop wordt nu automatisch opgeslagen in het geheugen van de ESP32.
4. Herhaal stappen 1...3 voor alle knoppen die je wilt aanleren.
5. Sluit de app door op *Menu Exit Program* te tikken.

Klaar. Je kunt nu je IR-bestuurde apparaten bedienen vanaf je BlueRC-apparaat via de app op je Android-telefoon.
Om problemen met de ESP32 op te lossen, kun je hem het beste uit- en weer aanzetten!
Een video van het apparaat in actie is te zien op mijn YouTube-kanaal [4].

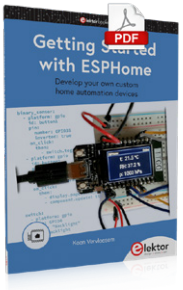
220103-03

Vragen of opmerkingen?

Hebt u technische vragen of opmerkingen naar aanleiding van dit artikel? Stuur een e-mail naar de redactie van Elektor via redactie@elektor.com.

Gerelateerde producten

- **Dogan and Ahmet Ibrahim, *The Official ESP32 Book* (PDF, SKU 18330)**
  www.elektor.nl/18330
- **Koen Vervloesem, *Getting Started with ESPHome* (PDF, SKU 19739)**
  www.elektor.nl/19739



WEBLINKS

[1] IRremoteESP8266-bibliotheek: https://www.arduino.cc/reference/en/libraries/irremoteesp8266/
[2] BlueRC Arduino-sketch voor ESP32: https://elektor.link/BlueRCESP32
[3] BlueRC Android-app: https://elektor.link/BlueRCApp
[4] Het BlueRC-systeem in actie: https://youtu.be/Kb-TdF84Oz4
[5] ESP32 Save Data Permanently using Preferences Library:
https://randomnerdtutorials.com/esp32-save-data-permanently-preferences/