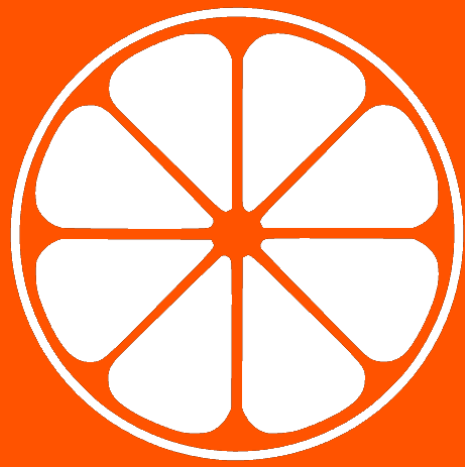




ORANGECON

4.06.2026 AMSTERDAM



SECOND FLASH: LONG LIVE THE ORANGECON BADGE!

Sławomir Jasek

SMARTLOCKPICKING.COM, BLESPlO.it



BLESPlo.it

SMARTLOCKPICKING.COM

SŁAWOMIR (SŁAWEK) JASEK

<suavomeer>

<suaveck>

<yaseck>

#HCE

#NFC

#KNX

#SDR

#BLE

#embedded

#RFID

#IoT



HITB SecConf



Creator of the
OrangeCon 2026 badge



HITB+ CyberWeek
Abu Dhabi, UAE: 12-17 October 2019

DEEPSEC



hardware.io
Hardware Security Conference and Training



ESP32S3

ESP32 - popular, low-cost MCU,
widely used IoT and embedded DIY

ESP32S3

- Dual core
- Crypto, AI hardware acceleration
- USB OTG
- Bluetooth 5
- 45 GPIO



WaveShare, Lilygo, M5Stack, ...



Manufacturers wrap ESP32 chip into useful prototype-friendly hardware, let you get from idea to working demo quickly.

Display, battery, buttons, peripherals, LoRa, ...

Our device: M5Stack M5StickS3

- 8M Flash, 8M PSRAM
- Integrated battery
- IMU (gyro)
- Speaker + microphone
- IR

<https://docs.m5stack.com/en/core/StickS3>



First things first: we will still need the badge for the conference!

Your badge is currently flashed with badge firmware, and configured with your individual Badge ID.

Before flashing anything else, make a “backup” to recover later.

The easiest: use esptool.py <https://github.com/espressif/esptool>

We can help you with it.

There are also several devices available for you to try without the need to tamper your badge.

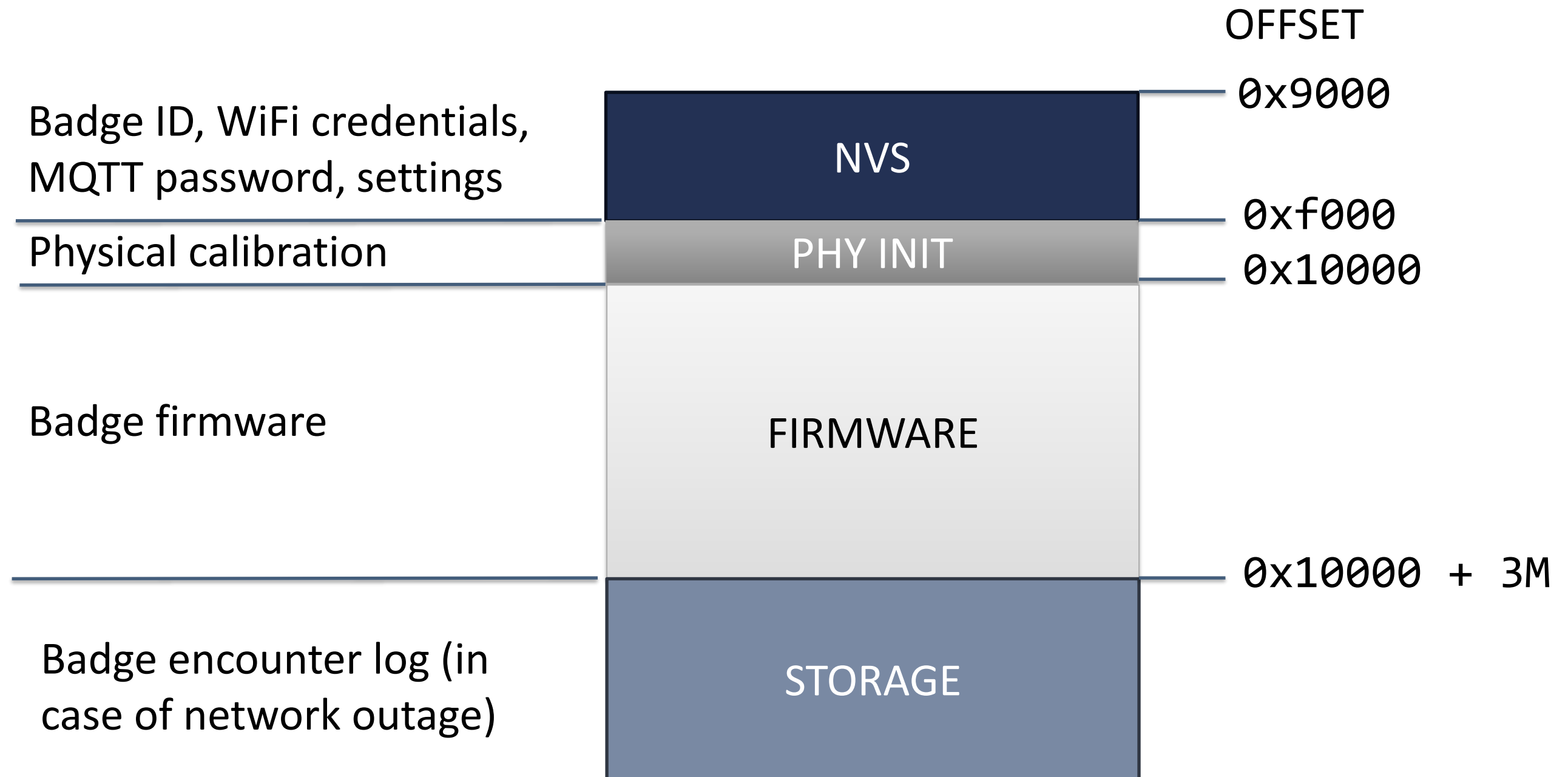
Worst case scenario fallback

Make a picture of your badge showing the ID.

We can recover from this.



Our badge: 4M flash used of 8M total



Grab the current flash content

1. Install esptool.py (python 3.10 needed)

```
pip install esptool
```

(or in venv)

```
python -m venv esptoolenv
```

On Windows: esptoolenv\Scripts\activate

On Linux or macOS: source esptoolenv/bin/activate

<https://docs.espressif.com/projects/esptool/en/latest/esp32/installation.html>

If you have esp-idf installed, it is already there.

Grab the current flash content

2. Read 4M of the flash:

Faster baud rate, skip in case of errors

```
esptool.py -b 460800 read_flash 0x0 0x400000 mybadge.bin
```

```
c:\tmp>esptool.py -b 460800 read_flash 0x0 0x400000 flash_backup.bin
esptool.py v4.8.1
Found 2 serial ports
Serial port COM3
Connecting.....
COM3 failed to connect: Failed to connect to Espressif device: No serial data received.
For troubleshooting steps visit: https://docs.espressif.com/projects/esptool/en/latest/troubleshooting.html
Serial port COM13
Connecting...
Detecting chip type... ESP32-S3
Chip is ESP32-S3-PICO-1 (LGA56) (revision v0.2)
Features: WiFi, BLE, Embedded Flash 8MB (GD), Embedded PSRAM 8MB (AP_3v3)
Crystal is 40MHz
MAC: 70:04:1d:da:87:70
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 460800
Changed.
Configuring flash size...
139264 (3 %)
```

Confirm the whole 4M is correctly read

```
Changing baud rate to 460800
Changed.
Configuring flash size...
4194304 (100 %)
4194304 (100 %)
Read 4194304 bytes at 0x00000000 in 95.8 seconds (350.1 kbit/s)...
Hard resetting via RTS pin...
```

```
2026-05-31  21:58    <DIR>      .
2026-05-31  21:57      4,194,304 flash_backup.bin
              1 File(s)      4,194,304 bytes
```

Re-flash the badge firmware

After the workshop, flash the badge firmware back using this command:

```
esptool.py write_flash 0x0 mybadge.bin
```

(also possible using for example <https://espressif.github.io/esptool-js/>)

If you have any issues with it, we can help.

Flashing ESP32S3 via USB

The ESP32 must be in “download” mode.

Most cases: majority of tools (like esptool.py) can turn it automatically, just plug the USB and it works.

Fallback: if it does not work, or in some cases (e.g. BLESPlo.it flash from Android app), it may be necessary to manually enter bootloader mode:

- Hold the left button until it flashes green
- Flash
- Press once the left button to reset



Flashing options

- Command-line: esptool.py
- esptool-js web: browser (web usb)
<https://espressif.github.io/esptool-js/>
- Many tools have their own web flashers (no need to provide .bin manually)
- M5Burner: multi-firmware flasher
- BLESPlO.it: flash directly from the mobile application



M5BURNER

M5Burner

Binary executable for Windows, Linux, Mac:

<https://docs.m5stack.com/en/uiflow/m5burner/intro>

M5Burner

| Download



M5Burner is a firmware burning software that integrates firmware burning, exporting, publishing, sharing and other functions. The software provides the burning of official UIFlow firmware and product-related demos, which is convenient for development and testing.

Fast and convenient way to install various binaries (demos, tools, ...) – official and community.

Note: you can also just download these binaries directly and flash them manually.

M5Burner

Select
"StickS3"



M5Burner - v202604291800

Device Type

- ALL
- CORE
- CORE2 & TOUGH
- CORES3
- STACKCHAN
- TAB5
- STICK
- STICKS3**
- ATOM
- ATOMS3
- STICKV & UNITV
- T-LITE
- CAMERA
- PAPER
- COREINK

Search

☐ Only Official



UIFlow2.0 StickS3 official

For StickS3

v2.4.5

M5Stack
Published At: 2026-05-15

Download

2772 0



StickS3 小智语音助手 official

Xiao Zhi Voice Assistant

v2.1.0

M5Stack
Published At: 2026-01-21

Download

1686 0



StickS3 PMIC Upgrade

1. After startup, the device will display the current PMIC version.

2. Click the button on the front of the Stick to begin the update process.

3. After the update is complete, the device will automatically restart. You can then check the currently displayed PMIC version.

Update Completed

Do you want to [restart](#) now ?



AirMouseS3

v1.0

A computer mouse that uses a gyroscope and accelerometer, based on the StickS3. Can be connected to any device via Bluetooth.
BUTTON A—LMB
BUTTON B—Freezes movement to straighten the stick if the cursor moves.

675 0 Github



Lightsaber for M5StickS3

1.0.0

Mini lightsaber toy. You can change the color of the lightsaber and connect external RGB modules. It produces realistic lightsaber sound effects when you move it. Here is video <https://youtu.be/okxXLD2phOA>



Bruce A/C Version

1.0.1

Patched some bugs should be working now:

Bruce A/C Edition is a custom Bruce firmware build focused on air conditioner control and experimentation. It adds an AC-B-Gone feature that can send valid OFF commands across multiple supported A/C protocols, plus a full Air Conditioner menu with brand selection, power control, temperature adjustment, mode

Volos23

Published At: 2026-02-28

MindMeld



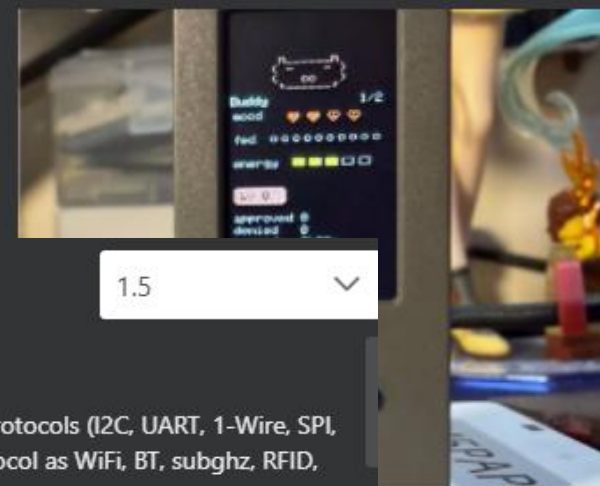
Bus Pirate for M5StickS3

1.5

A Hardware Hacking Tool with Web-Based CLI That Speaks Every Protocol.

It supports sniffing, sending, scripting, and interacting with various digital protocols (I2C, UART, 1-Wire, SPI, etc.) via a serial terminal or web-based CLI. It communicates with radio protocol as WiFi, BT, subghz, RFID, RF24.

geo_tp



Claude Desktop Buddy for StickS3

Claude Desktop Buddy firmware for StickS3

PR unmerged

<https://github.com/anthropics/claude-desktop-buddy/pull/3>

225 0 Github

Download and burn



Bruce 1.14-m5stack-sticks3

1.14

Bruce is a versatile ESP32 firmware that supports a ton of offensive features focusing on facilitating Red Team operations. It also supports M5stack and Lilygo products and works great with Cardputer, Sticks, M5Cores, T-Decks and T-Embeds.

Apead
Published At: 2026-02-04

3027 0 Github

[Download](#)



Bruce 1.14-m5stack-sticks3

1.14

Bruce is a versatile ESP32 firmware that supports a ton of offensive features focusing on facilitating Red Team operations. It also supports M5stack and Lilygo products and works great with Cardputer, Sticks, M5Cores, T-Decks and T-Embeds.

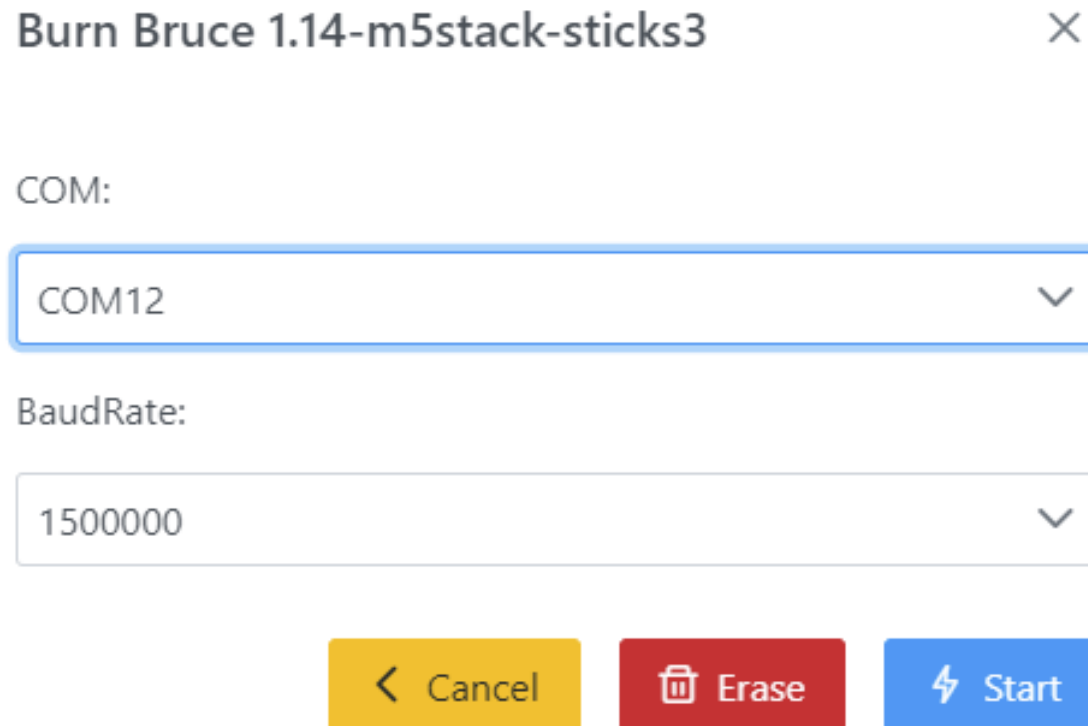
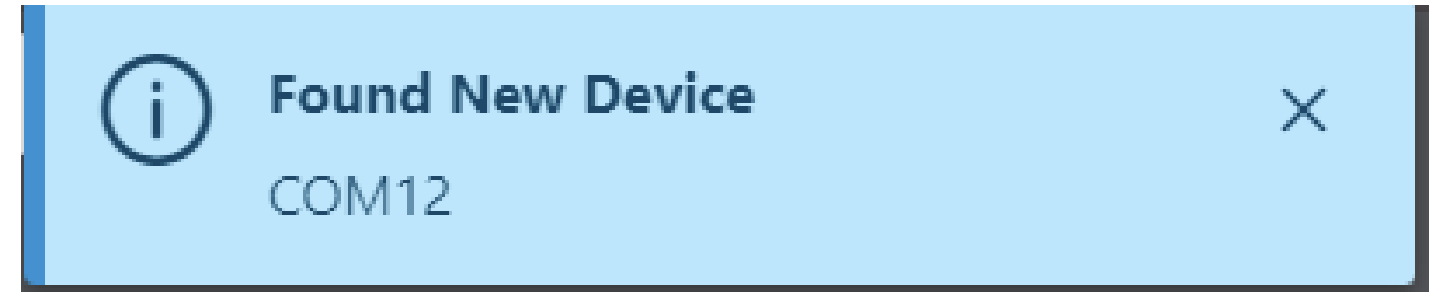
Apead
Published At: 2026-02-04

3027 0 Github

[Remove](#) [Burn](#)

Flashing

1. Connect the StickS3 via USB
2. Select the matching COM port and “Start”.



Successful “burn”

Burning

Writing at 0x0008daea... (75 %)
Writing at 0x00093404... (79 %)
Writing at 0x0009c3fe... (83 %)
Writing at 0x000a2b58... (87 %)
Writing at 0x000a8a74... (91 %)
Writing at 0x000ae59b... (95 %)
Writing at 0x000b47f6... (100 %)
Wrote 8388608 bytes (384960 compressed) at 0x00000000 in 29.1 seconds (effective 2306.5 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...

Burn successfully, click here to return

Troubleshooting

Error: serial exception.

- Confirm COM port number.
- Turn manually in Bootloader mode (hold left button until in blinks green).
- Note: in this mode it will stay at “100%” and “Waiting for burning”. Just reset it (click once the left button).

Burning

...

A serial exception error occurred: Write timeout

Note: This error originates from pySerial. It is likely not a problem with esptool, but with the hardware connection or drivers. For troubleshooting steps visit: <https://docs.espressif.com/projects/esptool/en/latest/troubleshooting.html>

Occurred errors, click here to return

Burning

writing at 0x00071038... (34 %)
Writing at 0x00076e93... (58 %)
Writing at 0x0007c743... (62 %)
Writing at 0x000820cc... (66 %)
Writing at 0x00087cac... (70 %)
Writing at 0x0008daea... (75 %)
Writing at 0x00093404... (79 %)
Writing at 0x0009c3fe... (83 %)
Writing at 0x000a2b58... (87 %)
Writing at 0x000a8a74... (91 %)
Writing at 0x000ae59b... (95 %)
Writing at 0x000b47f6... (100 %)

Waiting for burning

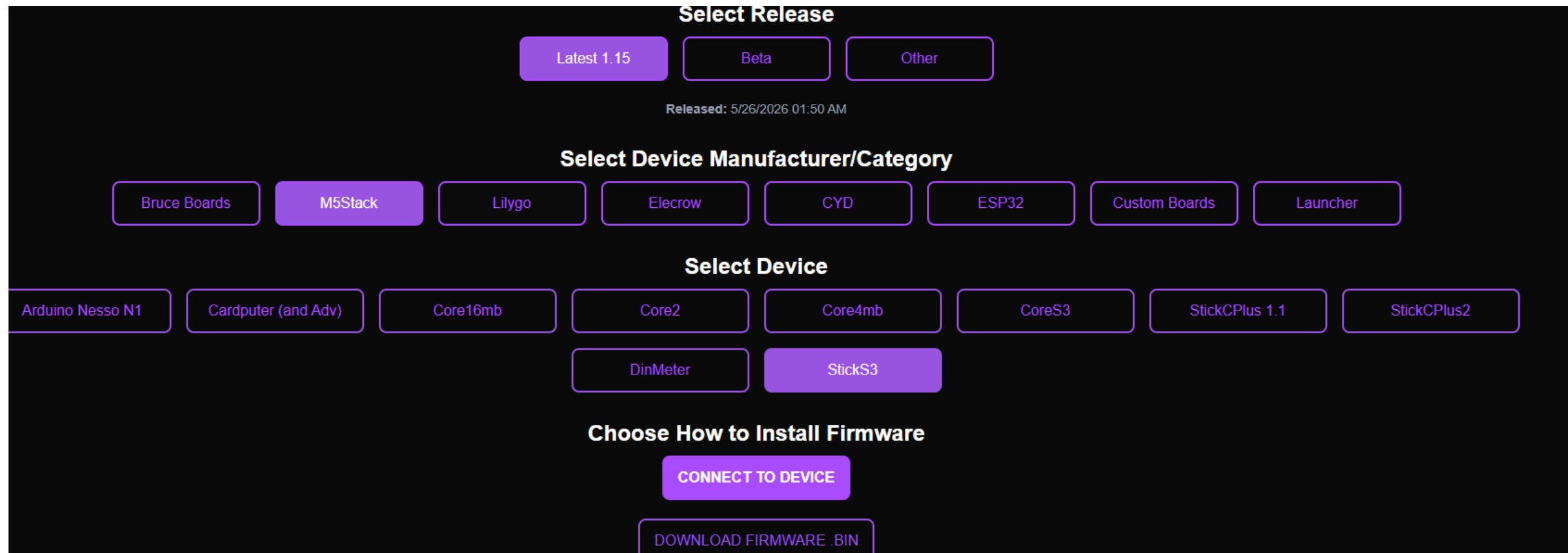


BRUCE

“hacking” multi-tool

<https://bruce.computer/> .

Possible to flash using m5burner or directly www:
<https://bruce.computer/flasher>



The screenshot shows a web interface for flashing firmware. It has a dark background with red buttons and text. The interface is divided into four main sections: 'Select Release', 'Select Device Manufacturer/Category', 'Select Device', and 'Choose How to Install Firmware'. In the 'Select Release' section, 'Latest 1.15' is selected. In the 'Select Device Manufacturer/Category' section, 'M5Stack' is selected. In the 'Select Device' section, 'StickS3' is selected. In the 'Choose How to Install Firmware' section, 'CONNECT TO DEVICE' is selected.

Select Release

Latest 1.15 Beta Other

Released: 5/26/2026 01:50 AM

Select Device Manufacturer/Category

Bruce Boards M5Stack Lilygo Elecrow CYD ESP32 Custom Boards Launcher

Select Device

Arduino Nesso N1 Cardputer (and Adv) Core16mb Core2 Core4mb CoreS3 StickCPlus 1.1 StickCPlus2

DinMeter StickS3

Choose How to Install Firmware

CONNECT TO DEVICE

DOWNLOAD FIRMWARE .BIN

WiFi features

- Karma Attack
- Beacon Spam
- Deauth flood
- Evil Portal
- ...



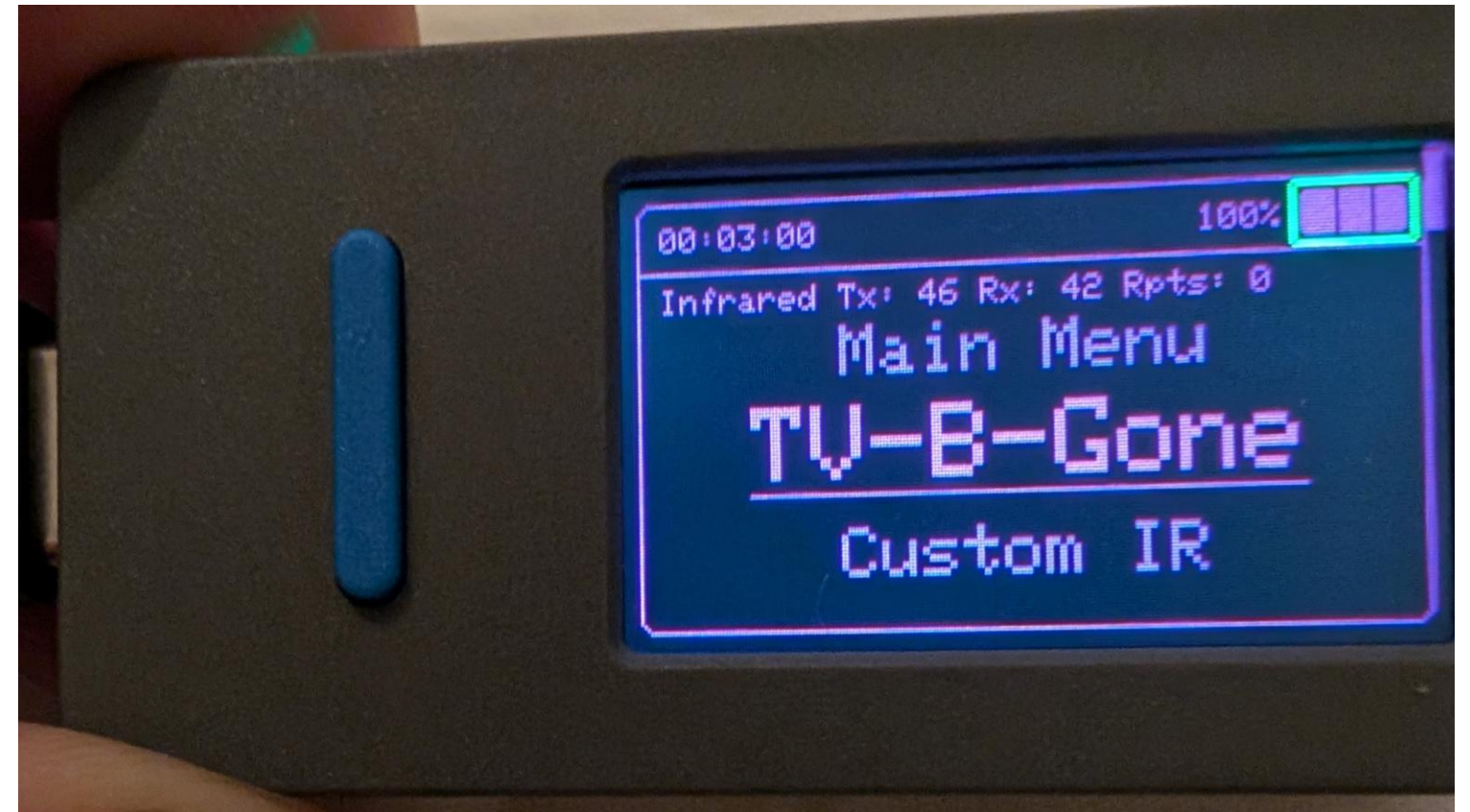
<https://wiki.bruce.computer/features/wifi/>

IR

Clone IR, TV-B-GONE, ...

A/C edition:

[GitHub - tutyr2-jpg/Bruce-A-C-Edition: Predatory ESP32 Firmware · GitHub](#)



Turn off the AC demo

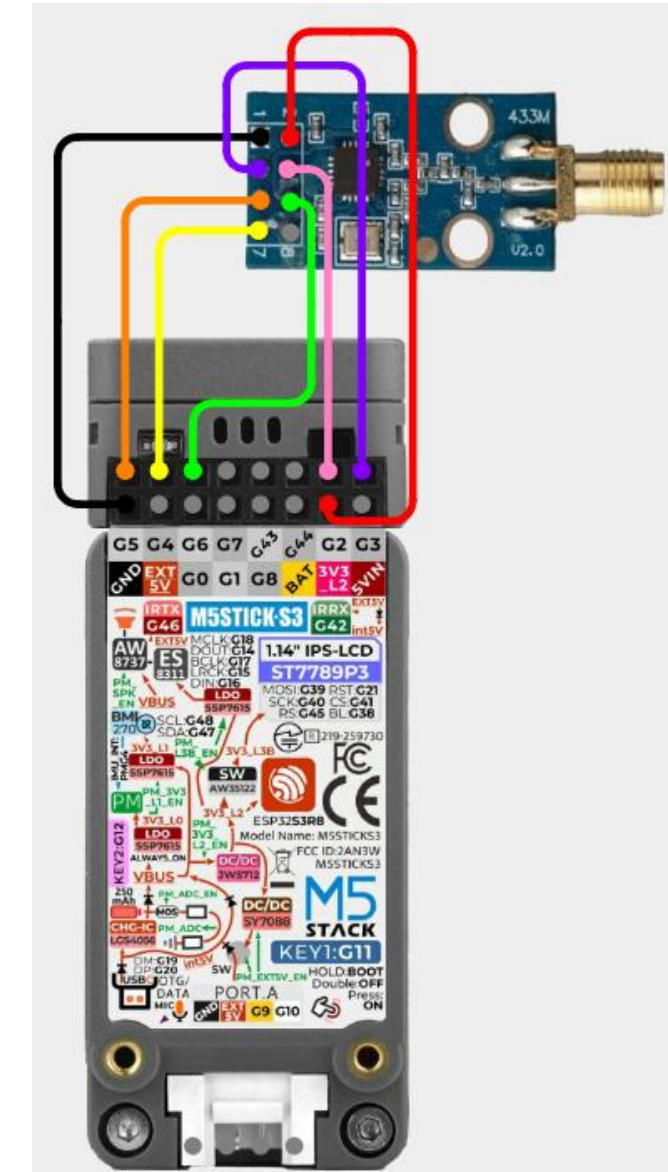
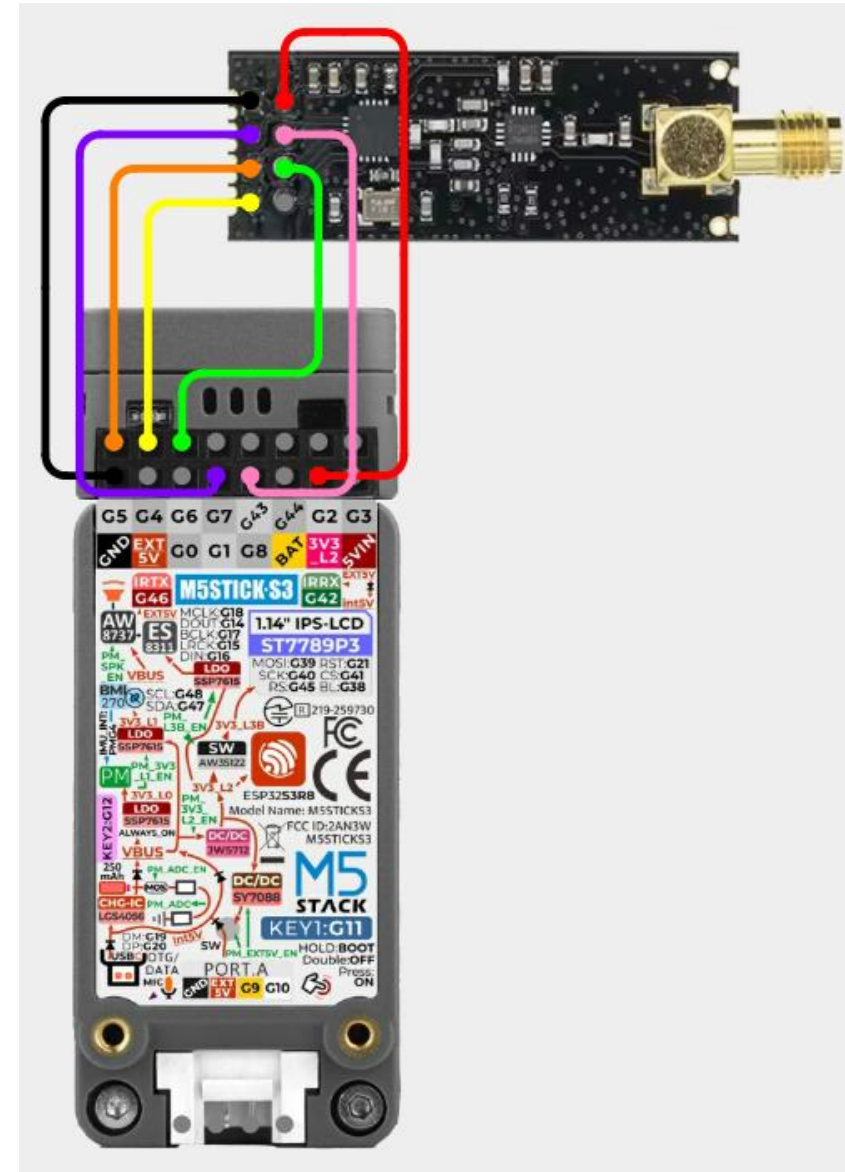


<https://youtu.be/UBIsMDq4UPU>

External components

More features with external components:

- CC1101: sub-GHz
- nRF24: jam 2.4GHz signals and view the spectrum.



<https://wiki.bruce.computer/wiring-diagrams/m5sticks3/cc1101/>
<https://wiki.bruce.computer/wiring-diagrams/m5sticks3/nrf24/>



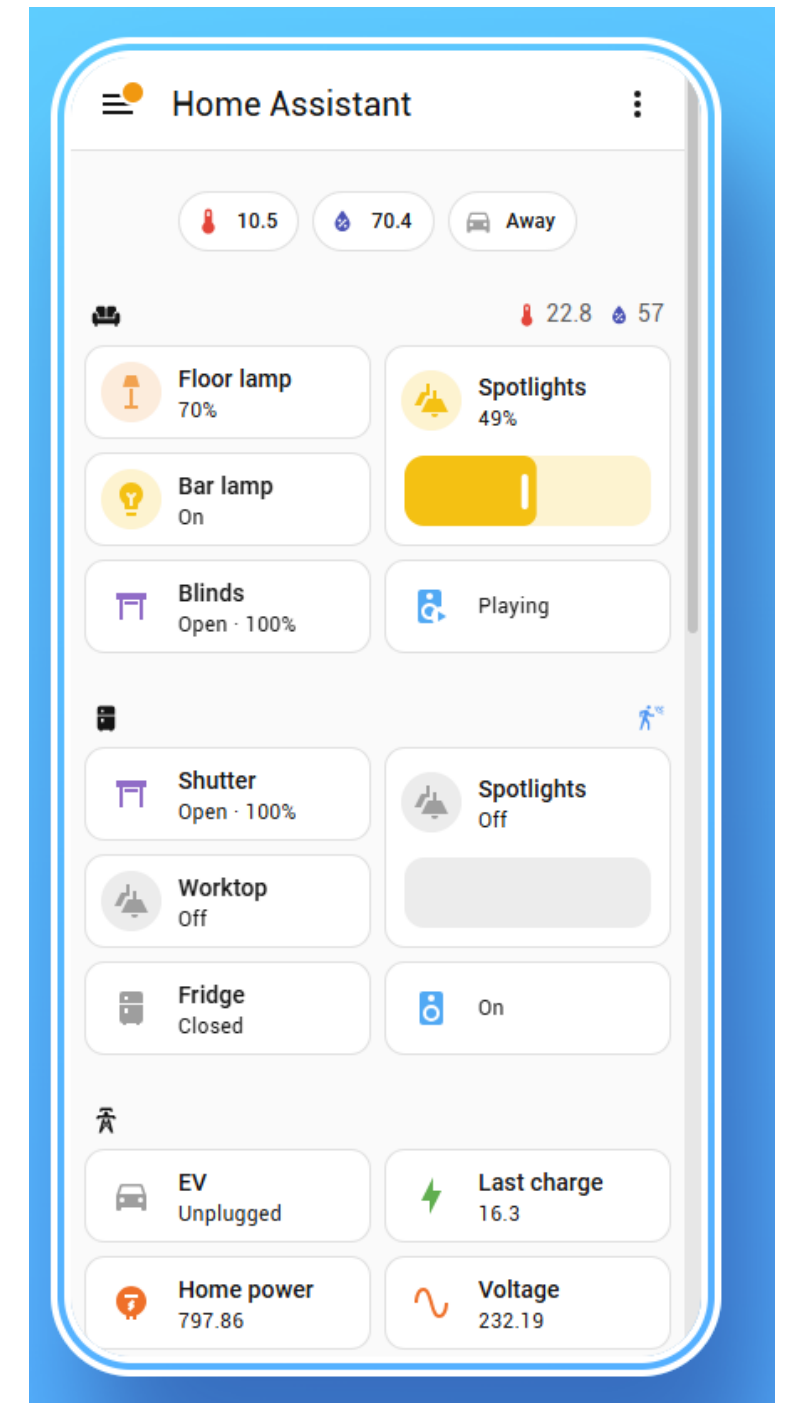
HOME ASSISTANT

What is Home Assistant?

A free, open-source home automation platform designed to act as a centralized, private, and local smart home hub.

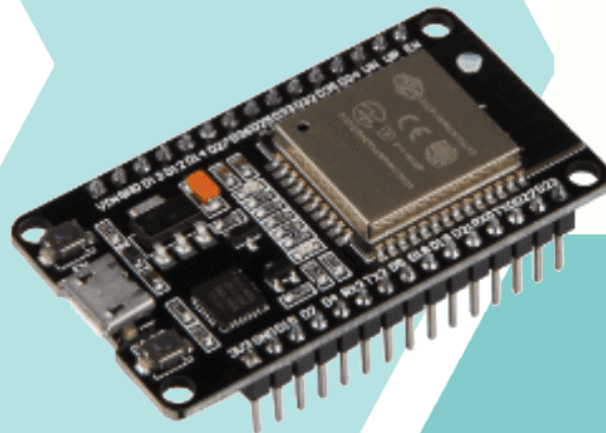
Connects devices from different brands allowing them to work together locally without relying on external cloud services.

<https://www.home-assistant.io/>



What is ESPHome?

Open-source system that lets you create custom firmware for ESP32 (for connecting to Home Assistant) with simple YAML configuration.



```
sensor:  
  - platform: dht  
    pin: D2  
    temperature:  
      name: "Temperature"  
    humidity:  
      name: "Humidity"
```

Living Room



Temperature

15.6 °C



Humidity

63%

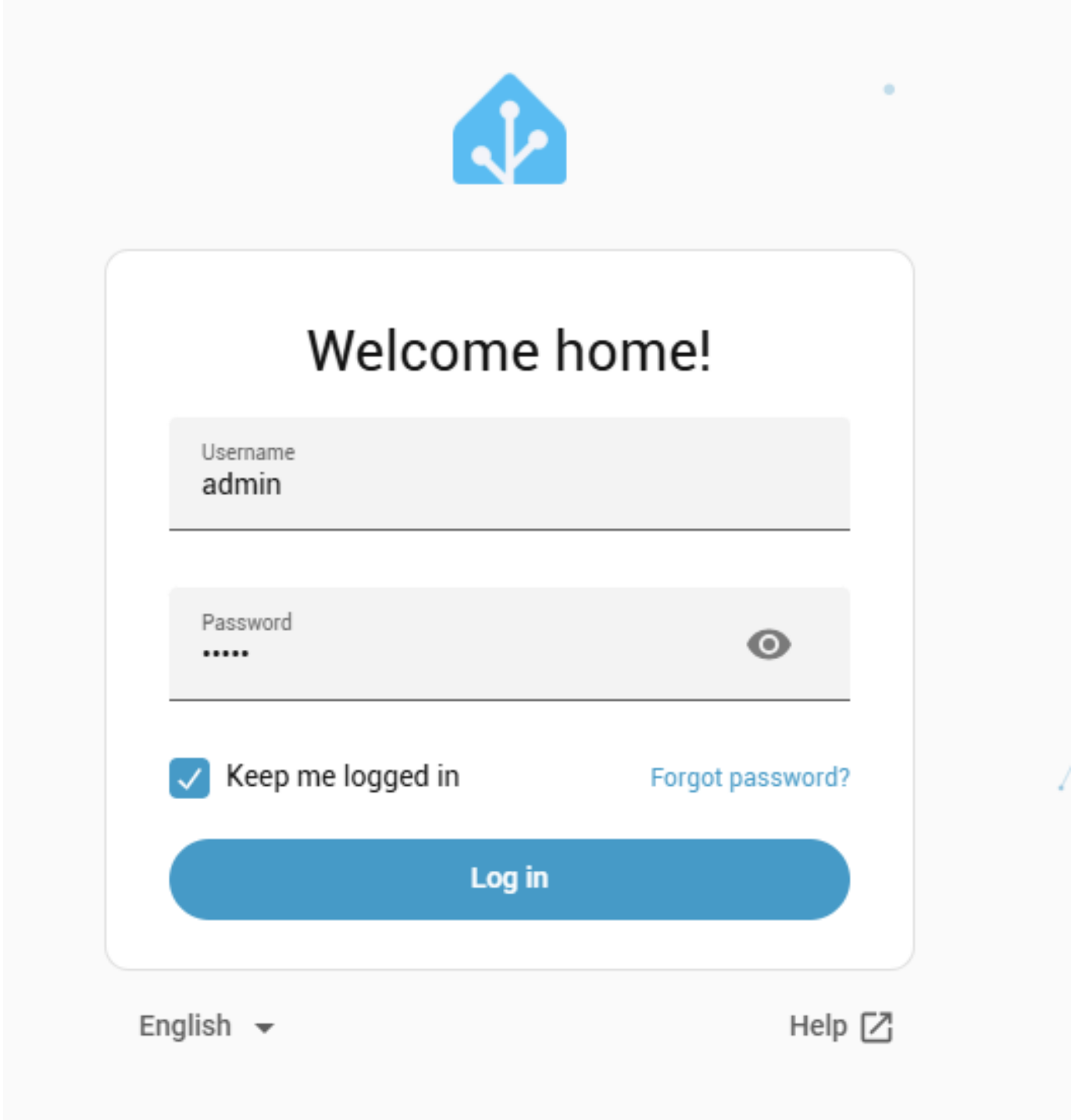
Home Assistant for our workshop

Wifi: smarthome, pass: smarthome

<http://10.5.5.2:8123>

admin/admin

Raspberry Pi running latest Home Assistant OS.



The image shows the Home Assistant login interface. At the top, there is a blue house icon with a white network symbol inside. Below this, the text "Welcome home!" is displayed. There are two input fields: "Username" with the value "admin" and "Password" with five dots. To the right of the password field is an eye icon. Below the fields, there is a checked checkbox labeled "Keep me logged in" and a link "Forgot password?". A large blue "Log in" button is at the bottom. At the very bottom, there is a language selector showing "English" with a dropdown arrow, and a "Help" link with an external link icon.

Username
admin

Password
.....

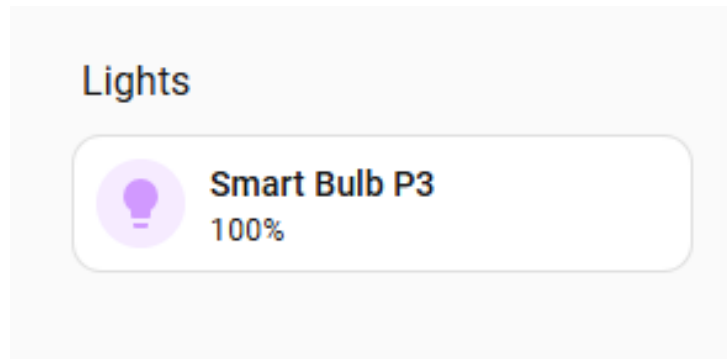
☒ Keep me logged in [Forgot password?](#)

Log in

English ▼ [Help](#)

Our setup

Matter smart lightbulb connected to Home Assistant.



We will add StickS3 as a remote control button for the light.



Sample StickS3 already paired

Press the button to switch our lightbulb.

Check in Overview -> Devices the button and light state.

mystick > Assign area

☒	Button KEY1	On
☐	Button KEY2	Off

Lights

 **Smart Bulb P3**
100%

ESPHome Builder

Converts yaml text config into binary image.

Possible to run on your laptop:

https://esphome.io/guides/installing_esphome/

Easier: installed in our Home Assistant as an “addon” application (but it takes more time to build on the Raspberry Pi)

≡< Home Assistant

 Overview

 Map

 Energy

 Activity

 History

 ESPHome Builder

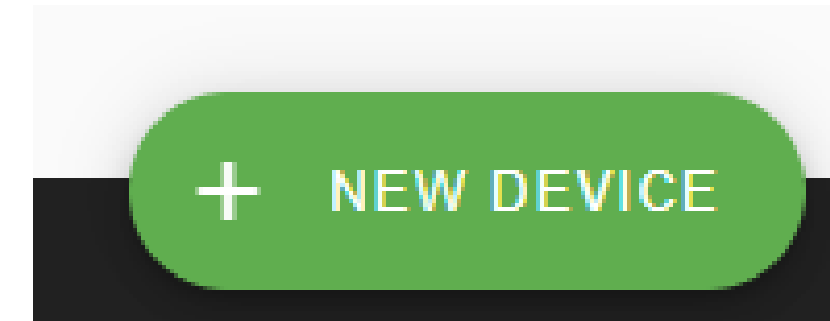
 File editor

 Media

 To-do lists

Steps to add a new device

1. Select “+ New Device” in bottom right corner



2. Select “New Device Setup”, name your device and select “ESP32-S3” as a template

ESP32-S3

Create configuration

How would you like to create your configuration?

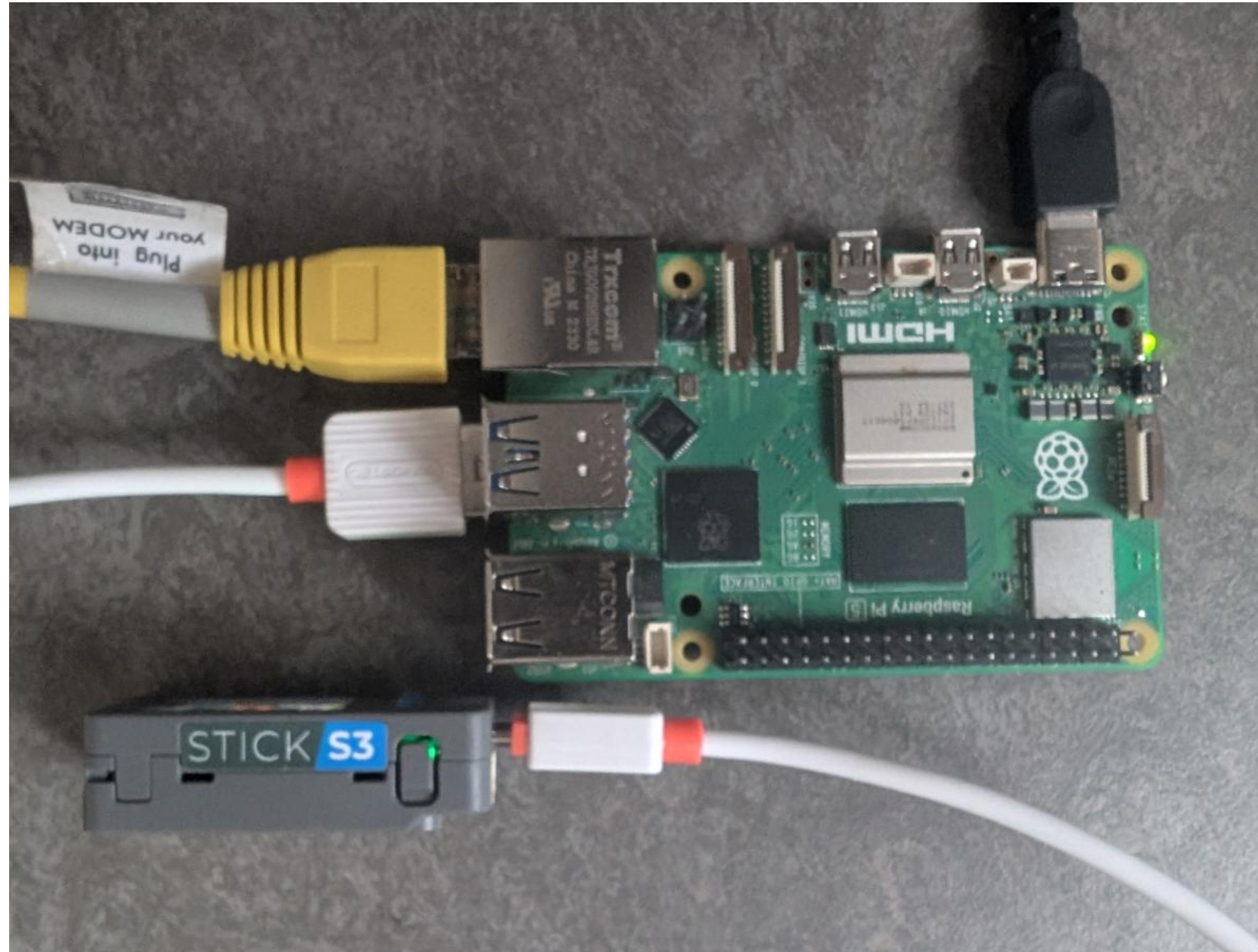
New Device Setup >
A guided process to get you started.

Import from File >
Use an existing ESPHome configuration (.yaml).

Empty Configuration >
For manually writing or pasting a configuration.

You can also drag and drop your .yaml file here

Connect the M5StickS3 directly to our home assistant RPi



Select “install” and then “Plug into the computer running ESPHome”



Configuration created!

You can now install the configuration to your device. The first time this requires a cable.

Once the device is installed and connected to your network, you will be able to manage it wirelessly.

Each ESPHome device has a unique encryption key to talk to other devices. You will need this key to include your device in Home Assistant. You can find the key later in the device menu.

Encryption key

oQPD7JI1Dt0GkHC9L2hvjTCqywYv+LCZdY55RJ0C8YQ=

SKIP

INSTALL

How do you want to install mystick.yaml on your device?

Wirelessly

Requires the device to be online



Plug into this computer

For devices connected via USB to this computer



Plug into the computer running ESPHome Device Builder

For devices connected via USB to the server



Manual download

Install it yourself using ESPHome Web or other tools



CANCEL

Flash

Pick Server Port

USB JTAG/serial debug unit
/dev/ttyACM0



Select the USB device, wait for build

BACK

Install mystick.yaml

```
INFO ESPHome 2026.5.1
INFO Reading configuration /config/esphome/mystick.yaml...
INFO Generating C++ source...
INFO Setting CONFIG_LWIP_MAX_SOCKETS to 12 (TCP=6 [api=3, captive_portal=3], UDP=3 [captive_portal=1, mdns=2], TCP_LISTEN=3 [api=1, ota=1, web_server_base=1])
INFO Compiling app... Build path: /data/build/mystick
Processing mystick (board: esp32-s3-devkitc-1; framework: espidf; platform: https://github.com/pioarduino/platform-espressif32/releases/download/55.03.38-1/platform-espressif32.zip)
-----
Created pioarduino Python virtual environment using uv: /root/.platformio/penv
Installed esptool from tl-install path: /data/cache/platformio/packages/tool-esptoolpy
Library Manager: Installing esphome/noise-c @ 0.1.11
Unpacking [#####] 100%
Library Manager: noise-c@0.1.11 has been installed!
Library Manager: Resolving dependencies...
Library Manager: Installing esphome/libsodium @ 1.10021.0
Unpacking [#####] 100%
Library Manager: libsodium@1.10021.0 has been installed!
HARDWARE: ESP32S3 240MHz, 320KB RAM, 4MB Flash
- contrib-piohome @ 3.4.4
- framework-espidf @ 3.50504.0 (5.5.4)
- tool-cmake @ 4.0.3
- tool-esp-rom-elfs @ 2024.10.11
- tool-esptoolpy @ 5.2.0
- tool-ninja @ 1.13.1
- tool-scons @ 4.40801.0 (4.8.1)
- tool-xtensa-esp-elf-gdb @ 16.3.0+20250913
- toolchain-xtensa-esp-elf @ 14.2.0+20260121
Creating a new virtual environment for IDF Python dependencies using uv
Using Python 3.12.10 environment at: /root/.platformio/penv/.espidf-5.5.4
```

Successful flash

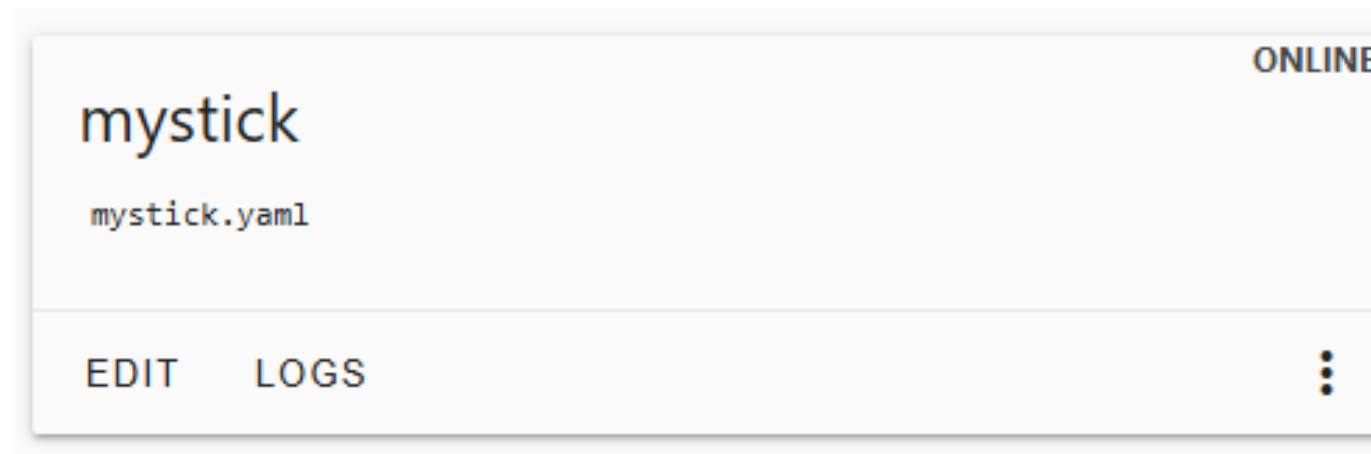
```
Merging binaries into /data/build/mystick/.pioenvs/mystick/firmware.factory.bin
Merging binaries with esptool
SHA digest in image updated.
Wrote 0xdee90 bytes to file '/data/build/mystick/.pioenvs/mystick/firmware.factory.bin', ready to flash to offset 0x0.
Successfully created /data/build/mystick/.pioenvs/mystick/firmware.factory.bin
esp32_copy_ota_bin([".pioenvs/mystick/firmware.bin"], [".pioenvs/mystick/firmware.elf"])
Copied firmware to /data/build/mystick/.pioenvs/mystick/firmware.ota.bin
===== [SUCCESS] Took 266.74 seconds =====
Using Python 3.12.10 environment at: /root/.platformio/penv/.espidf-5.5.4
```

```
Writing at 0x000cbfb2 [=====> ] 90.6% 524288/578616 bytes...
Writing at 0x000d18c6 [=====> ] 93.4% 540672/578616 bytes...
Writing at 0x000d7bf0 [=====> ] 96.3% 557056/578616 bytes...
Writing at 0x000dcbb1 [=====> ] 99.1% 573440/578616 bytes...
Writing at 0x000dee90 [=====] 100.0% 578616/578616 bytes...
Wrote 847504 bytes (578616 compressed) at 0x00010000 in 5.9 seconds (1147.0 kbit/s).
Verifying written data...
```

```
Hard resetting via RTS pin...
INFO Successfully uploaded program.
INFO Starting log output from /dev/ttyACM0 with baud rate 115200
[18:57:27.309][D][wifi:1458]: Found networks:
[18:57:27.310][I][wifi:1429]: - 'smarthome' (02:F2:AD:C5:BF:71)  Ch: 3 -38dB P:0
[18:57:27.311][D][wifi:1852]: Retry phase: INITIAL_CONNECT → SCAN_CONNECTING
[18:57:27.312][I][wifi:1097]: Connecting to 'smarthome' (02:F2:AD:C5:BF:71) (priority 0, attempt 1/2 in phase SCAN_CONNECTING)...
[18:57:31.637][I][wifi:1570]: Connected
[18:57:31.638][D][wifi:1587]: Disabling AP
[18:57:31.639][C][wifi:1237]: IP Address: 10.5.5.216
[18:57:31.640][C][wifi:1248]: SSID: 'smarthome'
[18:57:31.640][C][wifi:1248]: BSSID: 02:F2:AD:C5:BF:71
[18:57:31.640][C][wifi:1248]: Hostname: 'mystick'
[18:57:31.641][C][wifi:1248]: Signal strength: -35 dB 
[18:57:31.641][C][wifi:1248]: Channel: 3
```

Your device is now connected

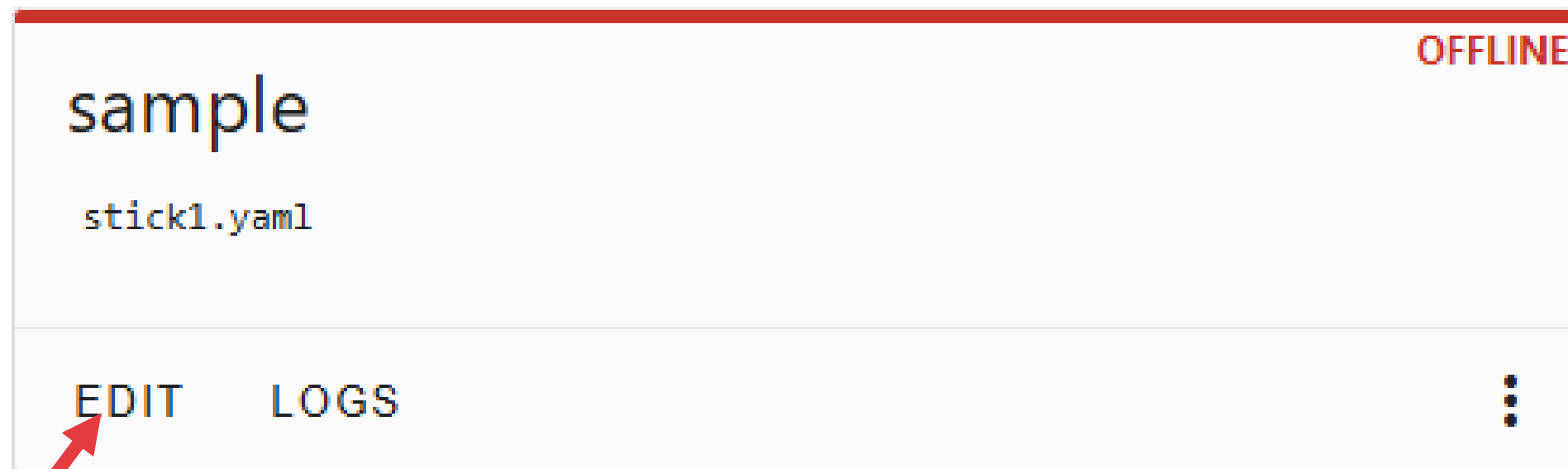
You have the ESPHome running on the StickS3 and connected to our Home Assistant.



But it is still generic, let's configure it to use the M5StickS3 buttons.

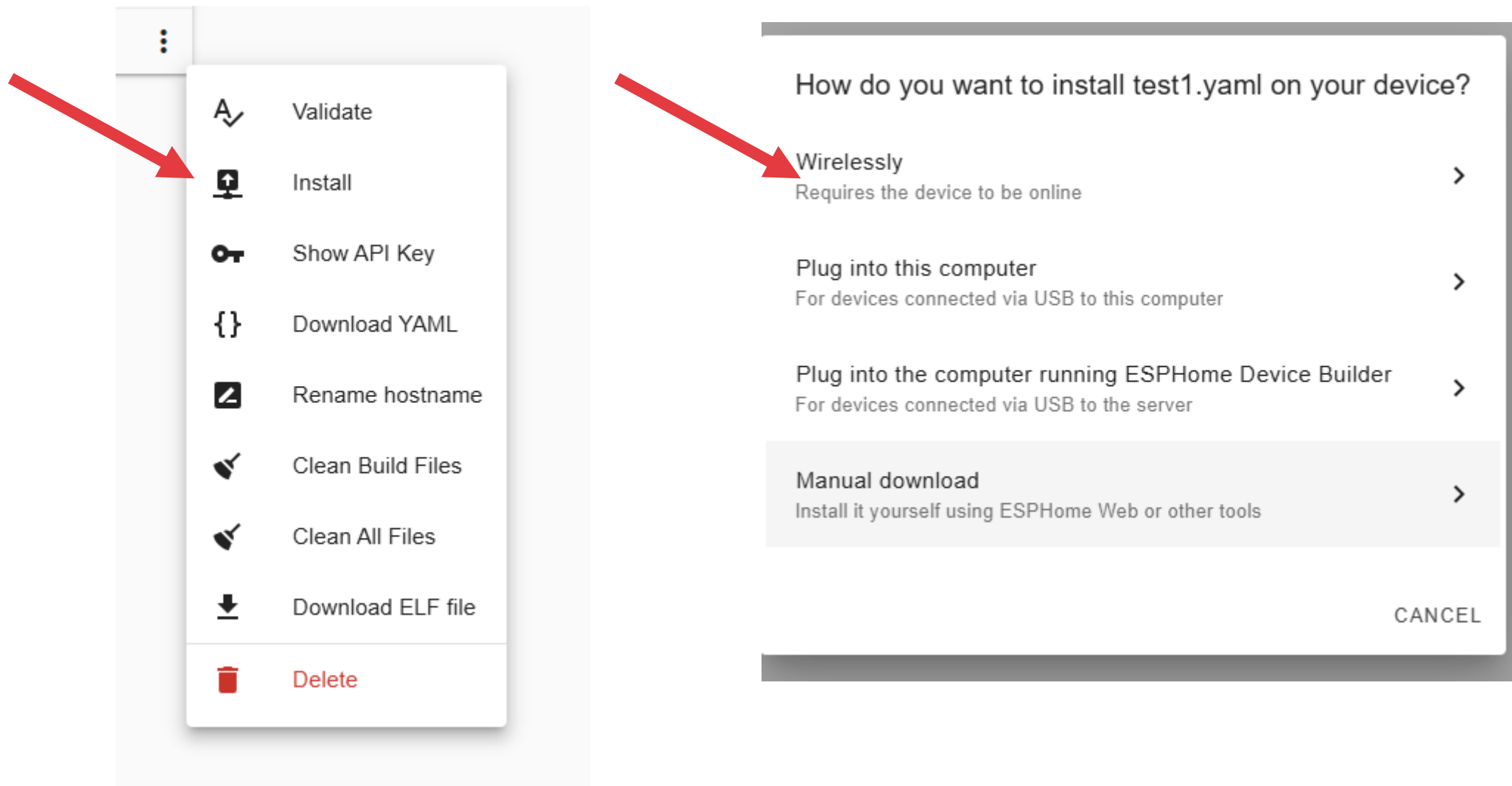
ESPHome Builder

Copy the “binary_sensor” section from sample to your new one (paste at the end)



```
binary_sensor:
- platform: gpio
  name: "Button KEY1"
  pin:
    number: GPIO11
    mode:
      input: true
      pullup: true
      inverted: true
- platform: gpio
  name: "Button KEY2"
  pin:
    number: GPIO12
    mode:
      input: true
      pullup: true
      inverted: true
```

Deploy the new configuration (Install -> Wirelessly)



Now go to Overview->Discovered Devices and add your device

The screenshot shows the Home Assistant mobile app's 'What do you want to add?' screen. At the top, there's a title bar with a close button (X) and the text 'What do you want to add?'. Below this, the 'Discovered' section lists three items: 'iBeacon Tracker' (with a circular icon), 'mystick' (with a house icon and 'ESPHome' below it), and 'Manage discovered devices' (with a right arrow). A modal dialog titled 'Discovered ESPHome device' is overlaid on the screen, asking 'Do you want to add the device mystick to Home Assistant?'. Below the dialog, there's a 'Name and assign' section with a dropdown menu showing 'esp32-s3-devkitc-1' and 'Espressif'. A 'Submit' button is visible on the right side of the screen.

Your new device

Overview->Devices.

Press the main or side button
to see if it works

mystick > [Assign area](#)

☐	Button KEY1	Off
☐	Button KEY2	Off

mystick > [Assign area](#)

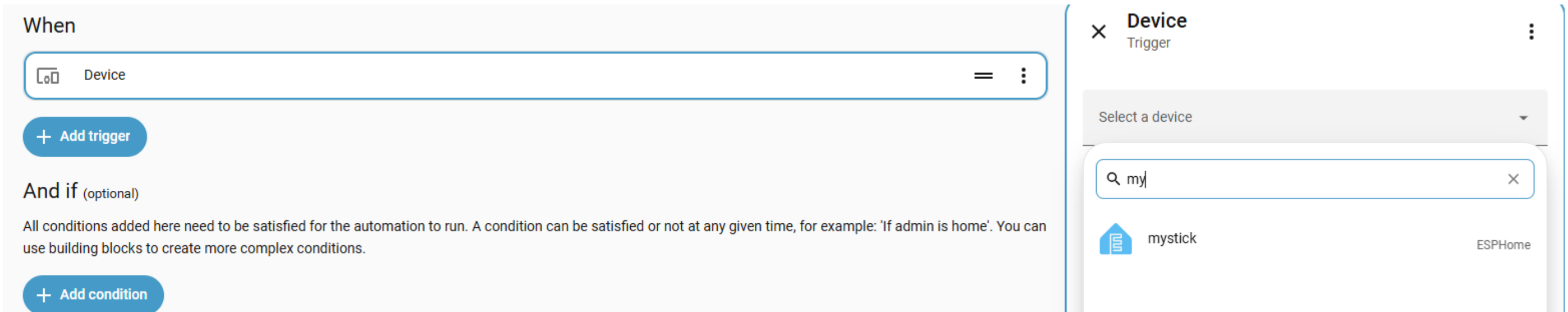
☒	Button KEY1	On
☐	Button KEY2	Off

Now just set the button to control the light

Settings -> Automations and Scenes -> Create Automation (bottom right)

+ Create automation

When -> Add Trigger -> Device -> select your device



The image shows a screenshot of an automation setup interface. On the left, under the 'When' section, there is a search bar with 'Device' entered, followed by an equals sign and a three-dot menu icon. Below this is a blue button labeled '+ Add trigger'. Under the 'And if (optional)' section, there is a text box explaining that all conditions must be satisfied for the automation to run, and a blue button labeled '+ Add condition'. On the right, a 'Device Trigger' panel is open, showing a dropdown menu 'Select a device' and a search bar with 'my' entered. Below the search bar, a list of devices is shown, including 'mystick' with an ESPHome icon.

Automation trigger

Select button (button 1 is main, button 2 is side)

Optionally add “duration” (for example 2s) if you want the “long press” action

×

Device

Trigger

⋮

 Device

mystick

×

▼

Trigger

mystick Button KEY1 turned on

▼

Duration

hh

:

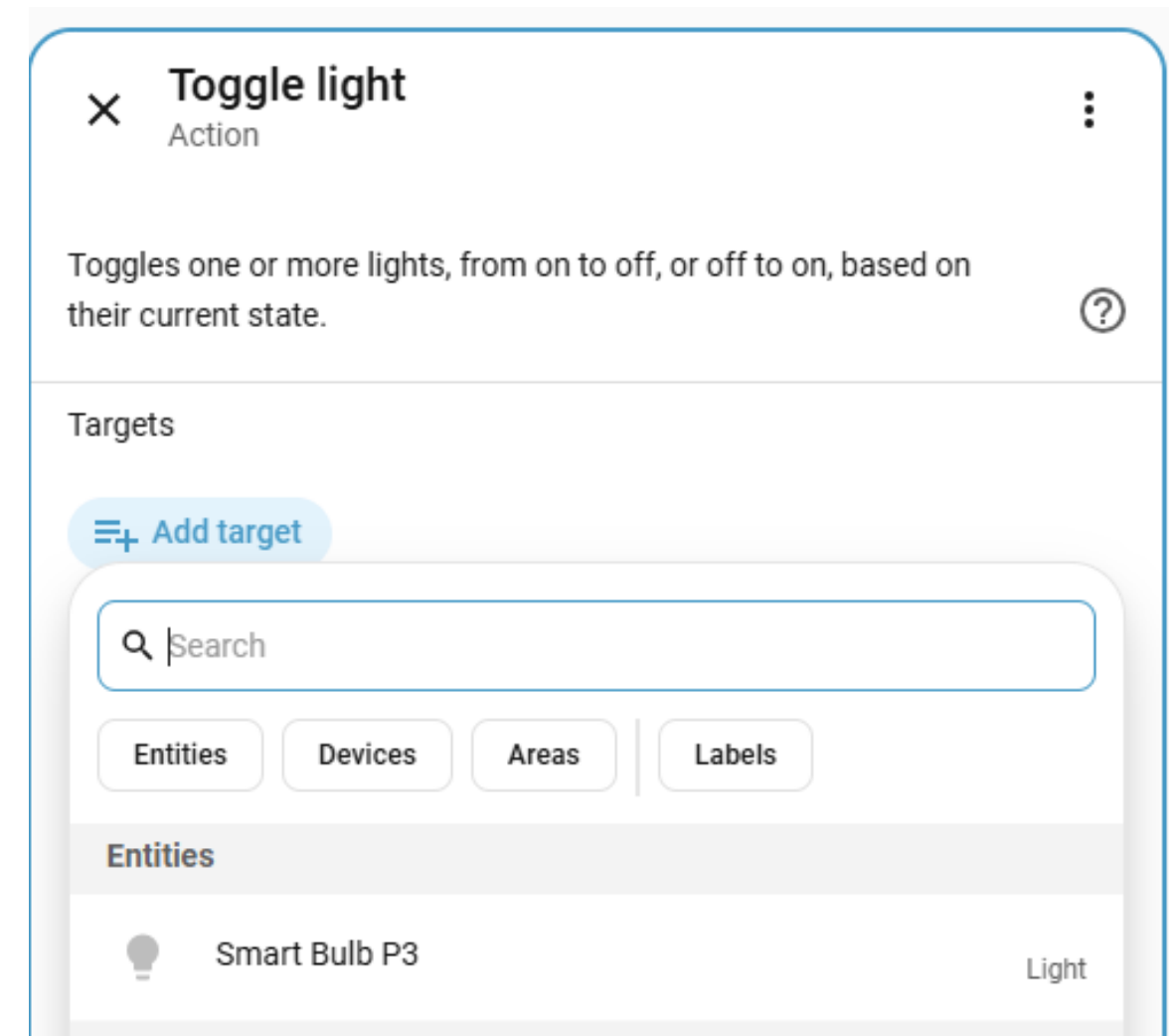
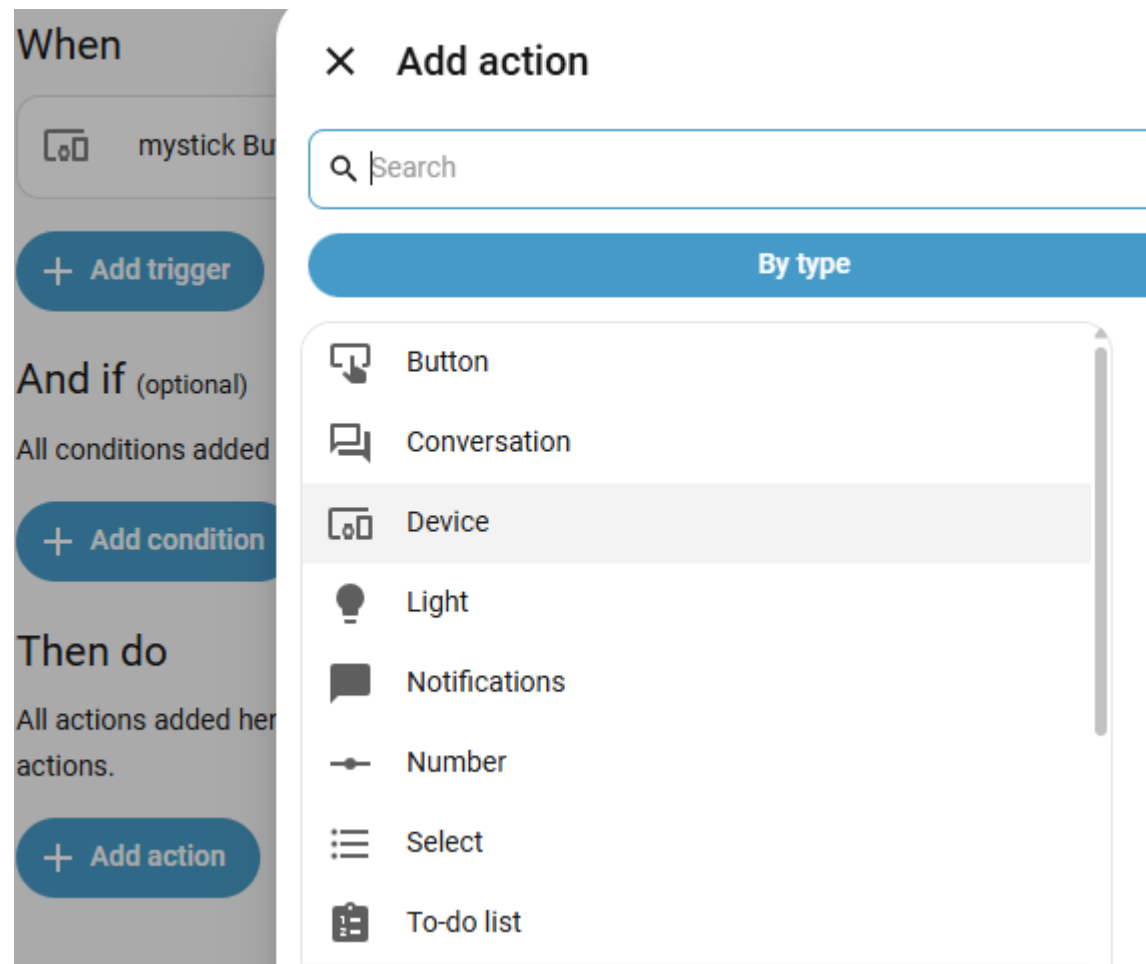
mm

:

ss

Automation action

Add action -> Light -> Toggle Light -> Add target -> Our lightbulb



Optional color

☐ Transition

Duration it takes to get to next state.

seconds

☐ Color

The color in RGB format. A list of three integers between 0 and 255 representing the values of red, green, and blue.

☐ Color temperature

Color temperature in Kelvin.

☐ Brightness

Number indicating the percentage of full brightness, where 0 turns the light off, 1 is the minimum brightness, and 100 is the maximum brightness.

%

✓ Advanced options

^ Advanced options

☐ RGBW-color

The color in RGBW format. A list of four integers between 0 and 255 representing the values of red, green, blue, and white.

☐ RGBWW-color

The color in RGBWW format. A list of five integers between 0 and 255 representing the values of red, green, blue, cold white, and warm white.

☐ Color name

A human-readable color name.

More features expected soon

Our example just used buttons, but the M5StickS3 has also display, microphone, speaker, ...

It is expected that the device will support among others Voice Assistant, like other M5Stack modules:

| 🤖 Voice Assistant



<https://docs.m5stack.com/en/homeassistant/overview>

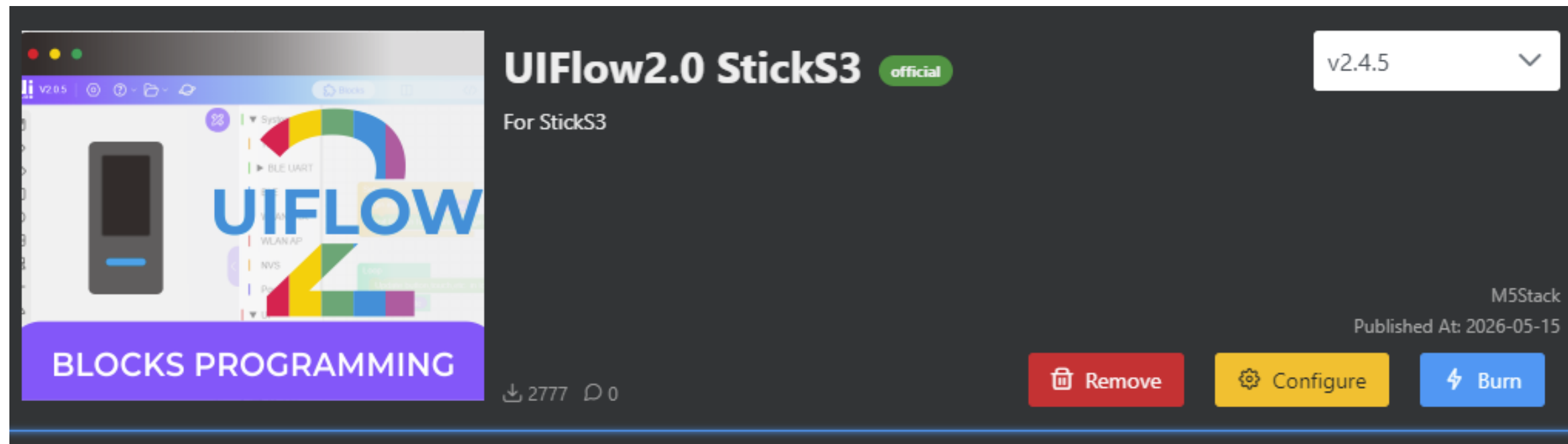


UIFLOW

Simple GUI programming

Installation

Step 1: install the “UIFlow” firmware, for example using M5Burner



<https://uiflow.m5stack.com>

1. Interface navigation

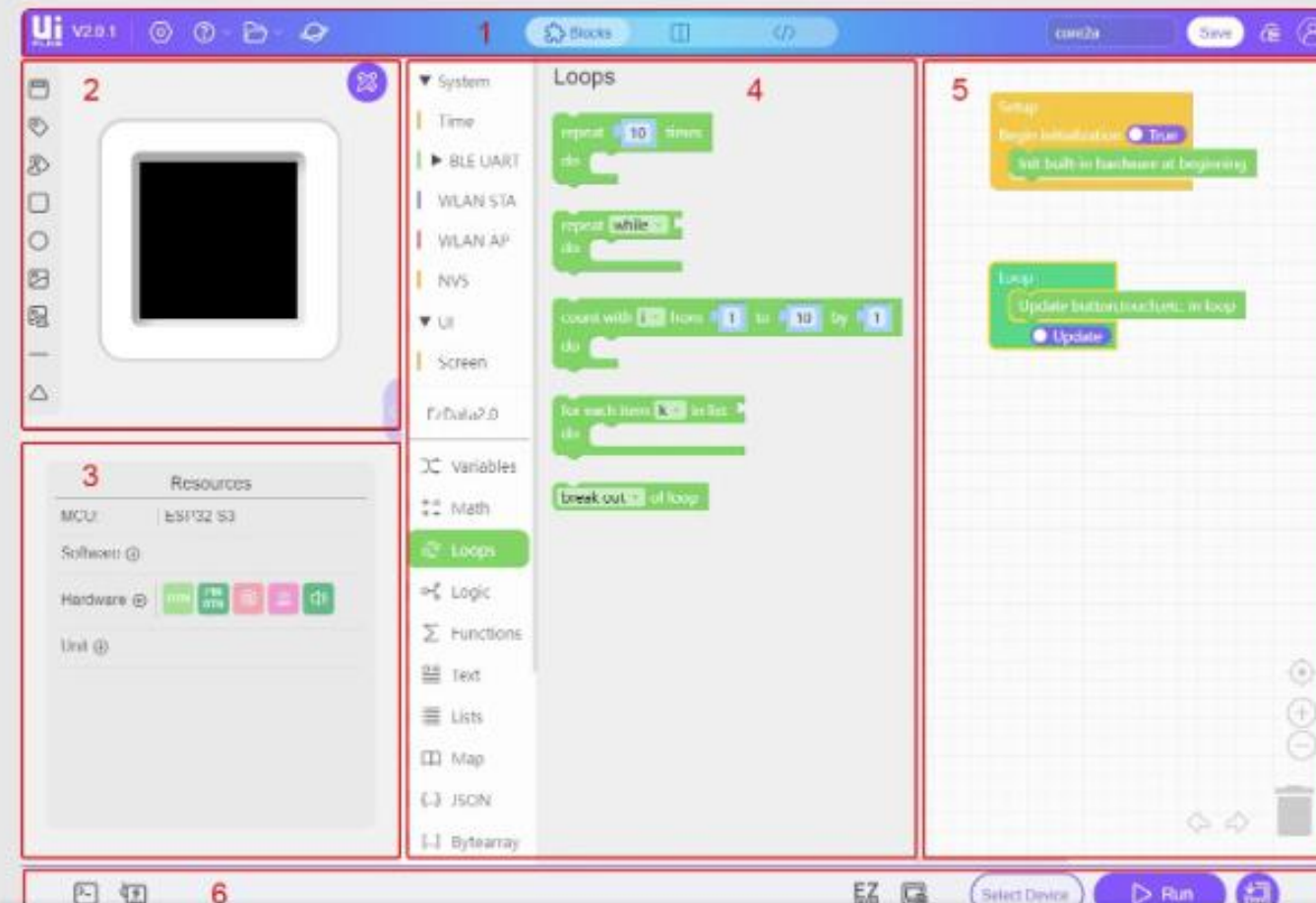
Interface navigation

The UIFlow2.0 interface includes: the top menu bar [1], UI editing area [2], resource panel [3], Block toolbox [4], programming area [5], and bottom function bar [6].

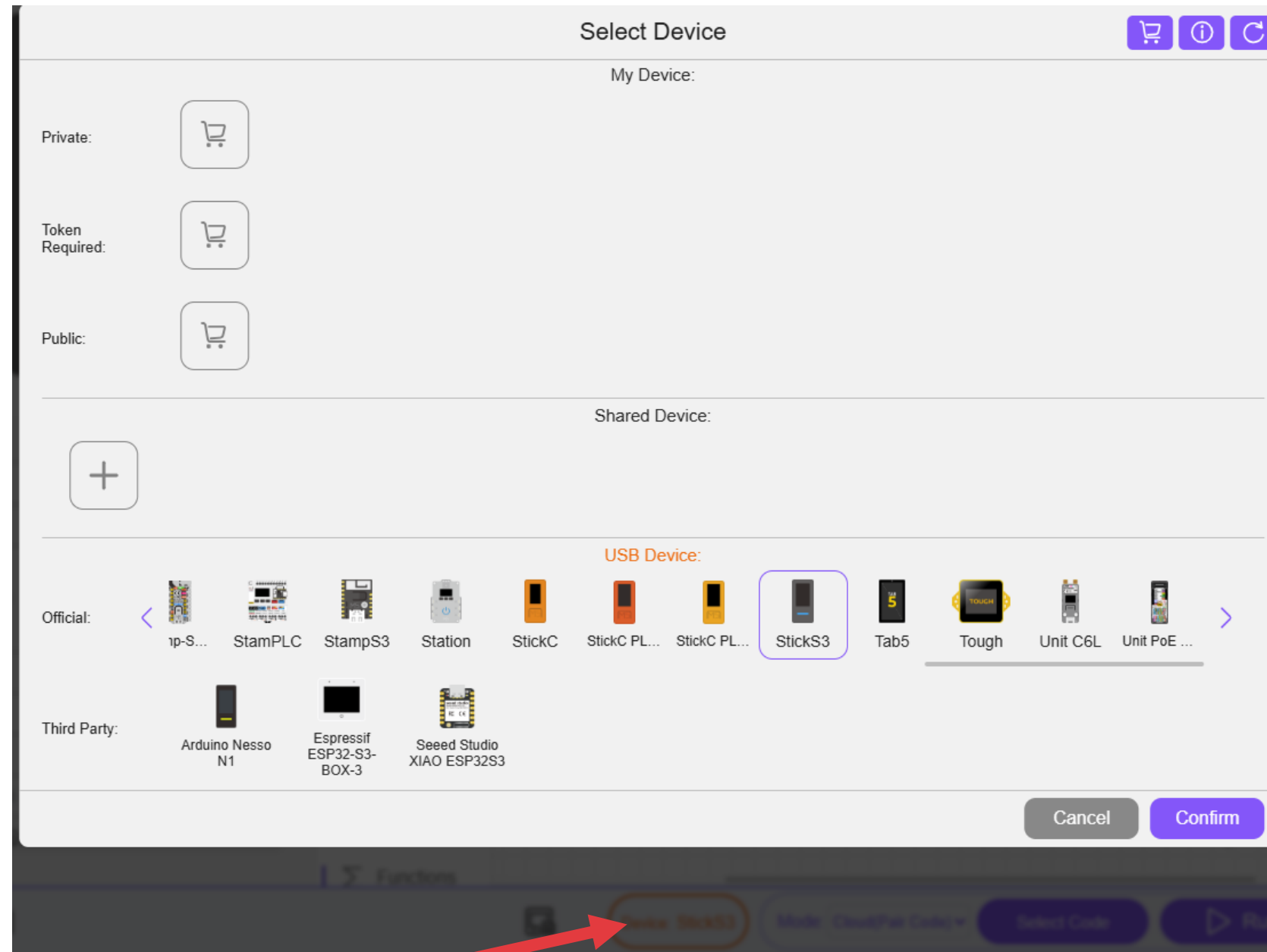
2. Login account

3. Firmware burning

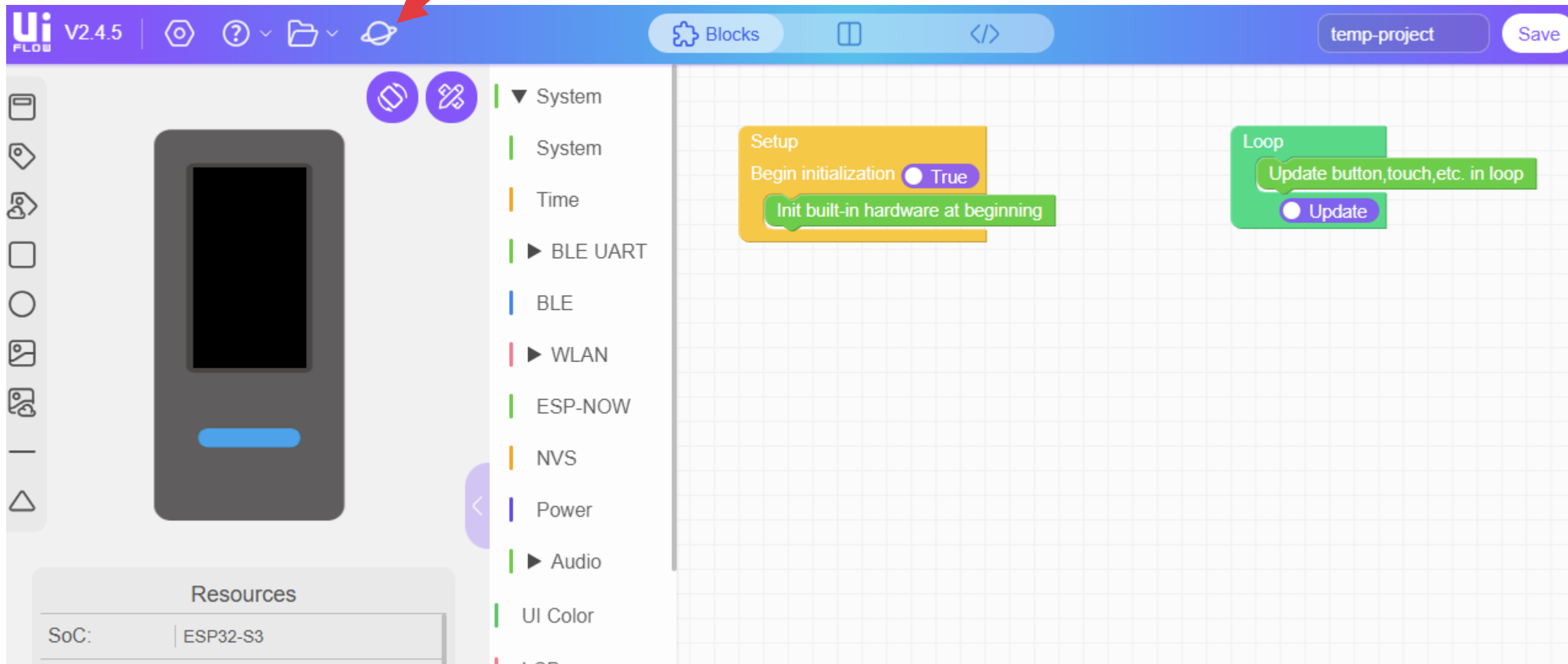
4. Develop



“Select device” (bottom), StickS3



Develop your own or browse sample projects



Sample projects


Project Zone

Name / description
Search...

Newest
Hottest

All
Official
User
Recommend

Official:
3
Station
StickC
StickC PL...
StickC PL...
StickS3
Tab5
To

Third Party:
Arduino Ness...
Espressif ES...
Seed Studio...

Selected:
StickS3



IMU test

displays gyro/accel data on the screen

V2.4.4



ID 151491
darksquare
2026-05-14 00:04:56
15 0

Import



Beeper v1

Index beeper inspired by Limbus Company

V2.4.4



ID 152419
Yumi-mp4
2026-04-24 18:54:13
1 0

Import



TickDemo-StickS3

A simulator of ticking sounds.(btnA:start/stop)

V2.4.3



ID 147659
Tower
2026-04-15 16:15:06
10 0

Import



Project

ID 145925

Import

Sample project

uiflow2.m5stack.com

Ui FLOW V2.4.5

Blocks

TickDemo-StickS: Save Update

System

System

Time

BLE UART

BLE

WLAN

ESP-NOW

NVS

Power

Audio

UI Color

LCD

M5GFX

Screen

Label

Image

Resources

SoC: ESP32-S3

Software +

Hardware +

Unit +

Buttons: BTN, PIN BTN, @, 🔊, IR, 🎤

Setup

Begin initialization ☒ True

Init built-in hardware at beginning

Speaker set volume 30 %

set BtnA_flag to false

set var_dita to false

When button BtnA was clicked

if BtnA_flag == true

do

set BtnA_flag to false

Set label1 text "Closed"

else

set BtnA_flag to true

Set label1 text "Alarm"

set var_dita to false

set Time_A to get ticks in milliseconds

Loop

Update button,touch,etc. in loop

if BtnA_flag == true

do

if ticks diff get ticks in milliseconds Time_A > 1000

do

Speaker play tone freq 2000 millisecond 50

set var_dita to true

if ticks diff get ticks in milliseconds Time_A > 2000

do

Speaker play tone freq 4000 millisecond 50

set Time_A to get ticks in milliseconds

set var_dita to false

else

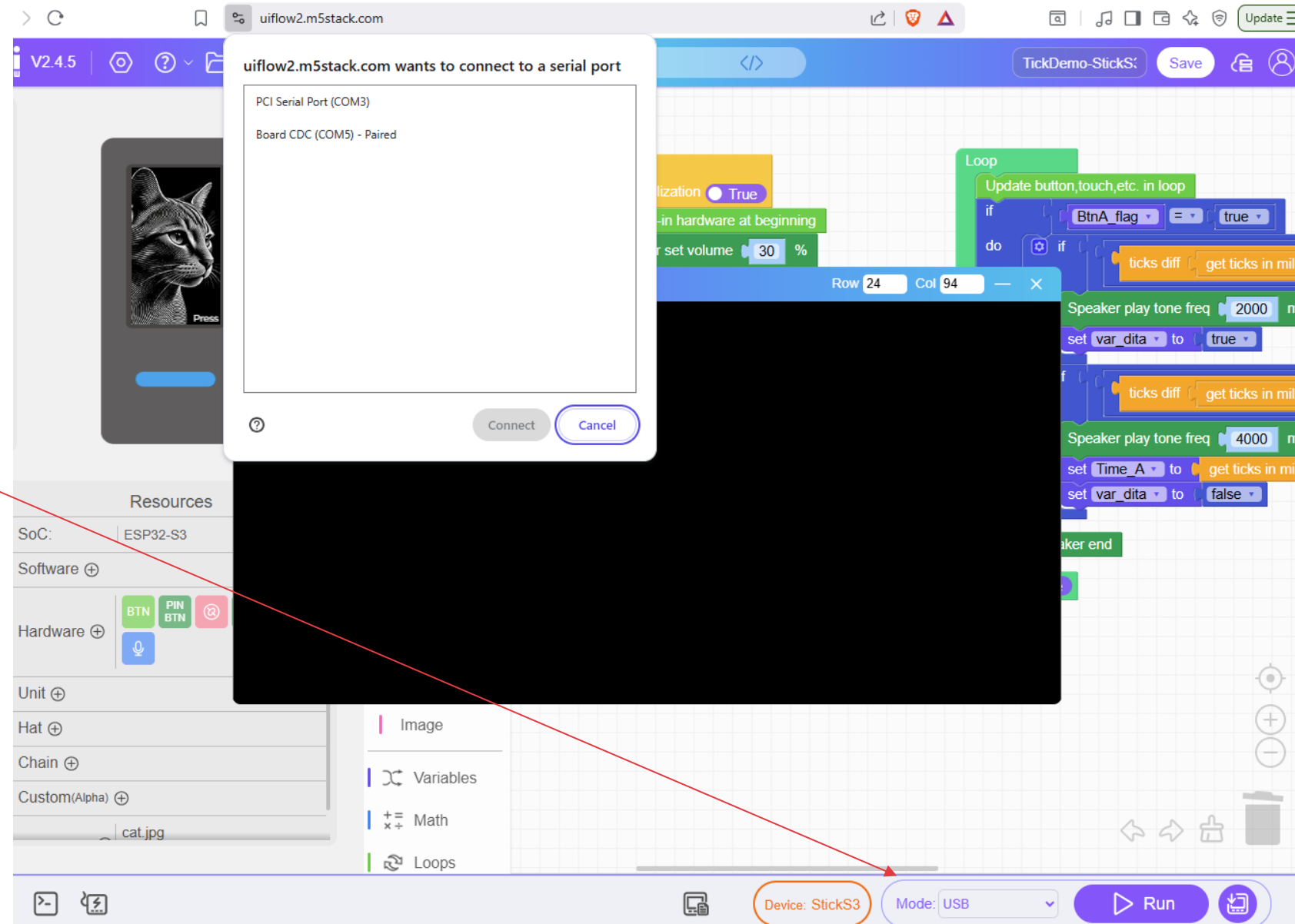
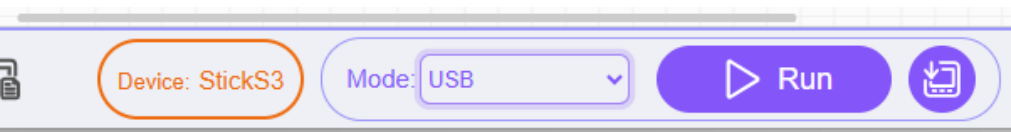
Speaker end

Update

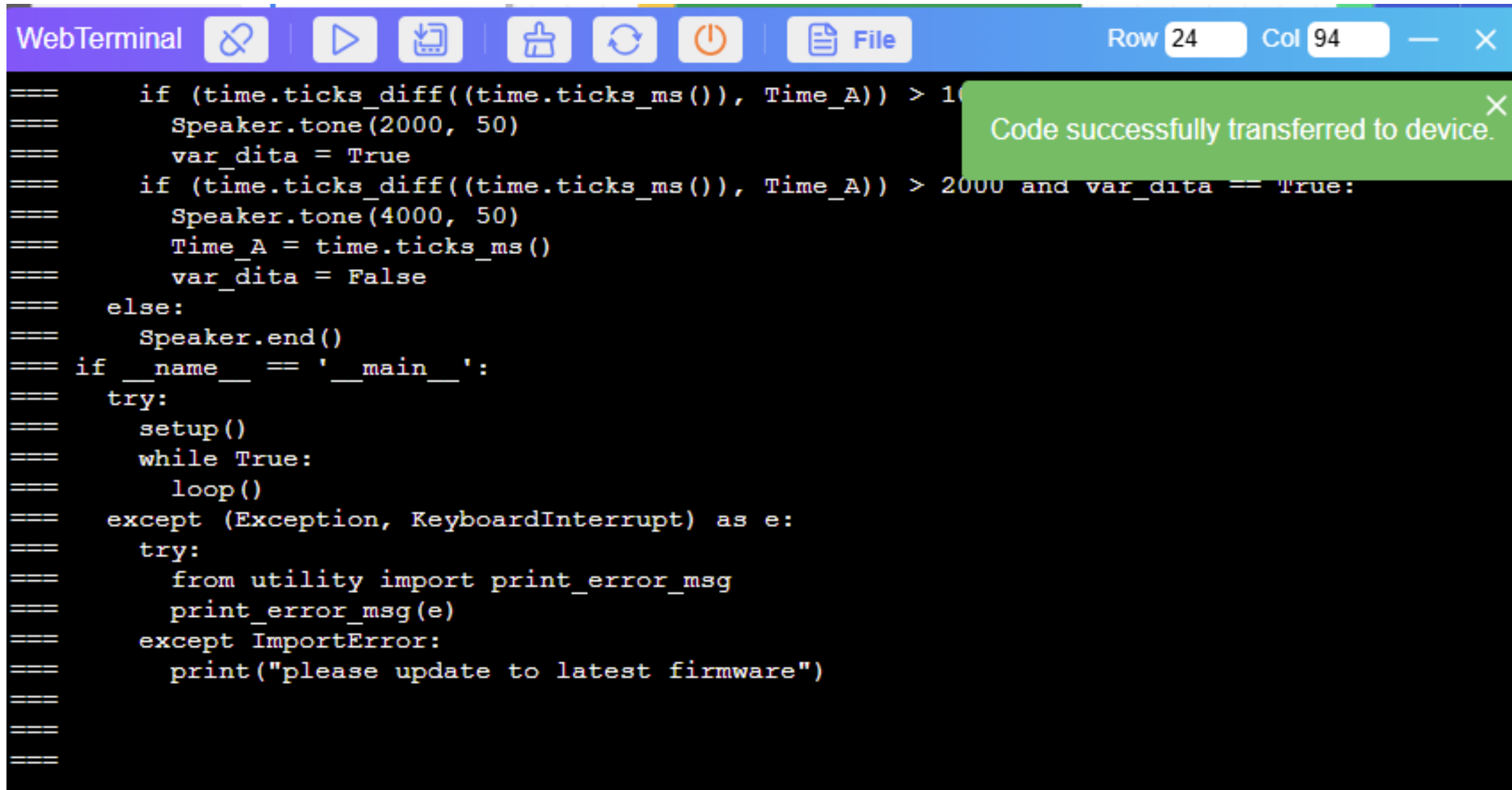
Device: StickS3 Mode: Cloud(Pair Code) Select Code Run

Run the project

Select mode “USB”



Code transferred



A screenshot of a WebTerminal window. The title bar is purple and contains the text 'WebTerminal' and several icons: a refresh icon, a play icon, a download icon, a thumbs up icon, a circular arrow icon, a power icon, and a document icon labeled 'File'. To the right of the icons, it shows 'Row 24' and 'Col 94'. The terminal area has a black background with white text. The code is a Python script for a microcontroller, likely an ESP8266, using the 'time' and 'Speaker' modules. It includes a main loop that checks the time difference and plays tones. A green notification box with a close button (X) is overlaid on the right side of the terminal, containing the text 'Code successfully transferred to device.'.

```
WebTerminal Row 24 Col 94
if (time.ticks_diff((time.ticks_ms()), Time_A)) > 1000:
    Speaker.tone(2000, 50)
    var_dita = True
if (time.ticks_diff((time.ticks_ms()), Time_A)) > 2000 and var_dita == True:
    Speaker.tone(4000, 50)
    Time_A = time.ticks_ms()
    var_dita = False
else:
    Speaker.end()
if __name__ == '__main__':
    try:
        setup()
        while True:
            loop()
    except (Exception, KeyboardInterrupt) as e:
        try:
            from utility import print_error_msg
            print_error_msg(e)
        except ImportError:
            print("please update to latest firmware")
```

Code successfully transferred to device.



BLESPLO.IT

BLESPlo.it



Mobile app

- Rich UI, complex parsing/fingerprinting, ...
- Uses phone's BLE capabilities

Device library with dynamic scripts

ESP32 "companion"

- Connect via WiFi or USB-C*
- Scan/connect remotely
- Get more low-level details on packets
- Simulate any device (optionally with display/buttons)
- Fully control the pairing parameters

* Not all features yet via USB



Mobile application

Currently Android only (iOS coming soon!)

<https://play.google.com/store/apps/details?id=it.blesplo.app>



Don't have Android phone? We have several available for the class.

You can try the app solo on the sample devices (robotic dog, dildo, lightbulb, Bluetooth car, ...).

More options when you connect it to your badge.



Use the application solo: control smart lightbulb

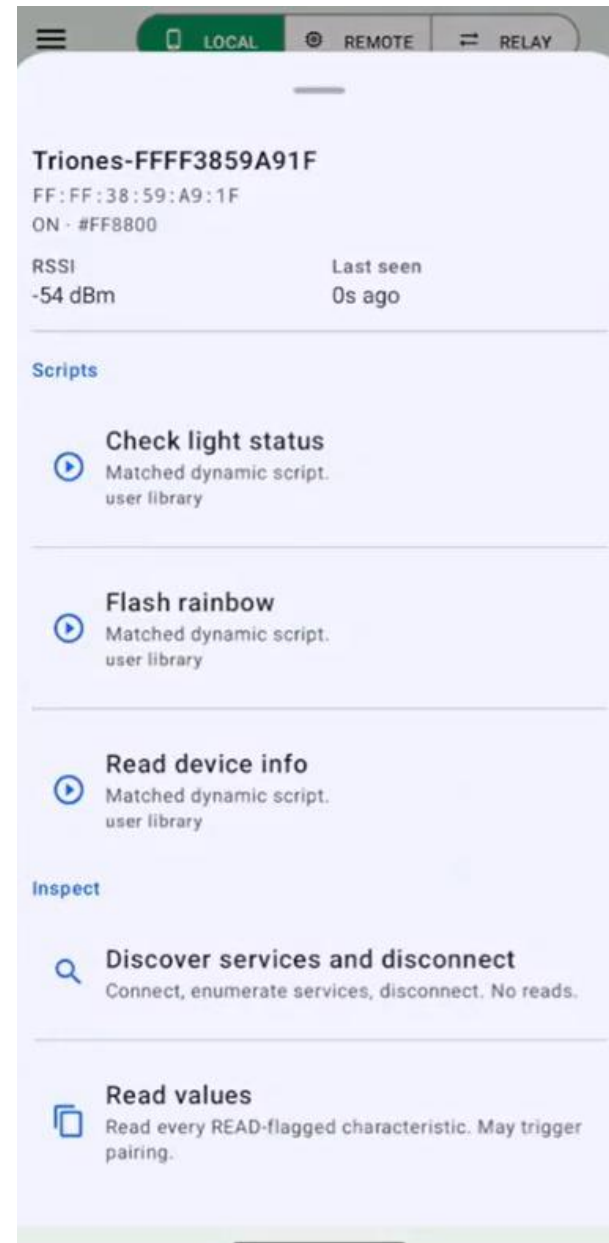
Find "Triones" lamp



Use Quick action to check light status or flash rainbow

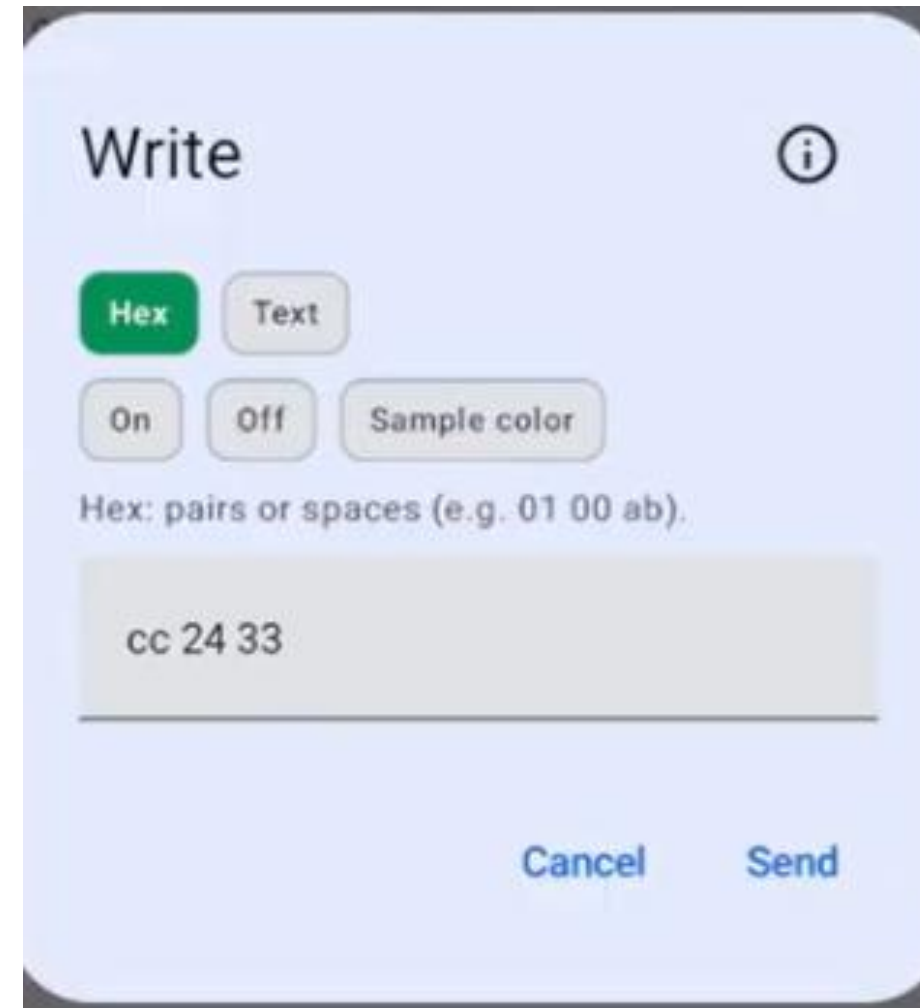
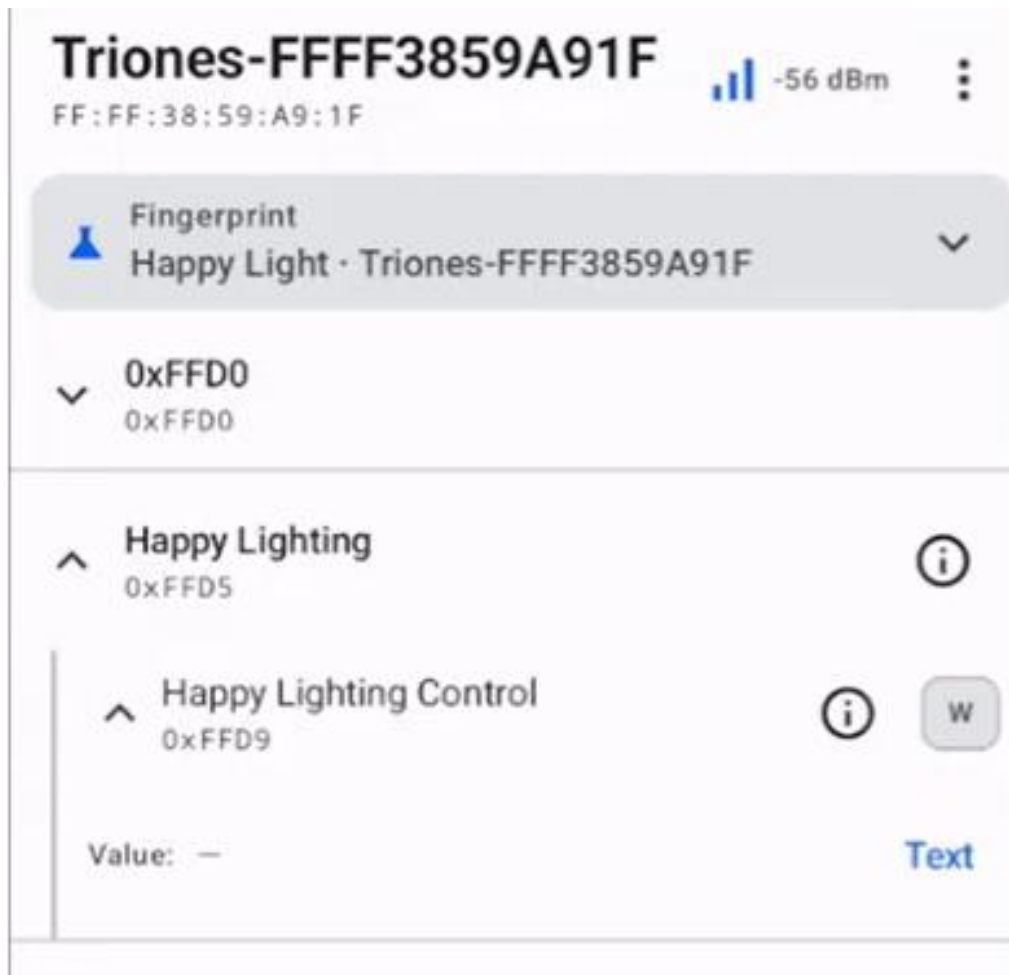


Use connection for manual mode control



<https://www.youtube.com/watch?v=zUJHDkzPpjU>

Lamp manual control after connecting



Use the application solo: control robotic dog

Find robotic dog

-66



Triones-FFFF3859A91F
 Happy Light: Triones
 FF:FF:38:59:A9:1F






Connect to the device

Control manually or select “Scripts” at the bottom to control with BLESPlo.it menu.

Rapidpower-dog-fire

-47 dBm

9A:CE:45:BF:A2:99

Capture (this session) · svc

GAP

0x1800

Write

Hex

Text

Stop

Move forward

Turn left

Turn right

Hex: pairs or spaces (e.g. 01 00 ab).

f0 2a 00 01 d5 ff fe

Cancel

Send

Manual

Scripts

Logs

Rapidpower-dog-fire

-72 dBm

70:04:1D:DA:73:4A

Rapidpower dog

Read values

Read device info

Actions

State

Last cmd: unknown

Swimming

Sit down

Greetings / Hello

Get down

Act cute

Handshake

Disconnect

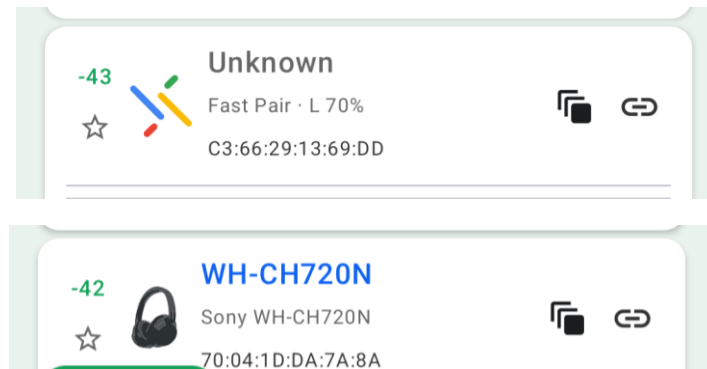
Manual

Scripts

Logs

Use the application solo: Airoha vulnerability against Sony headset

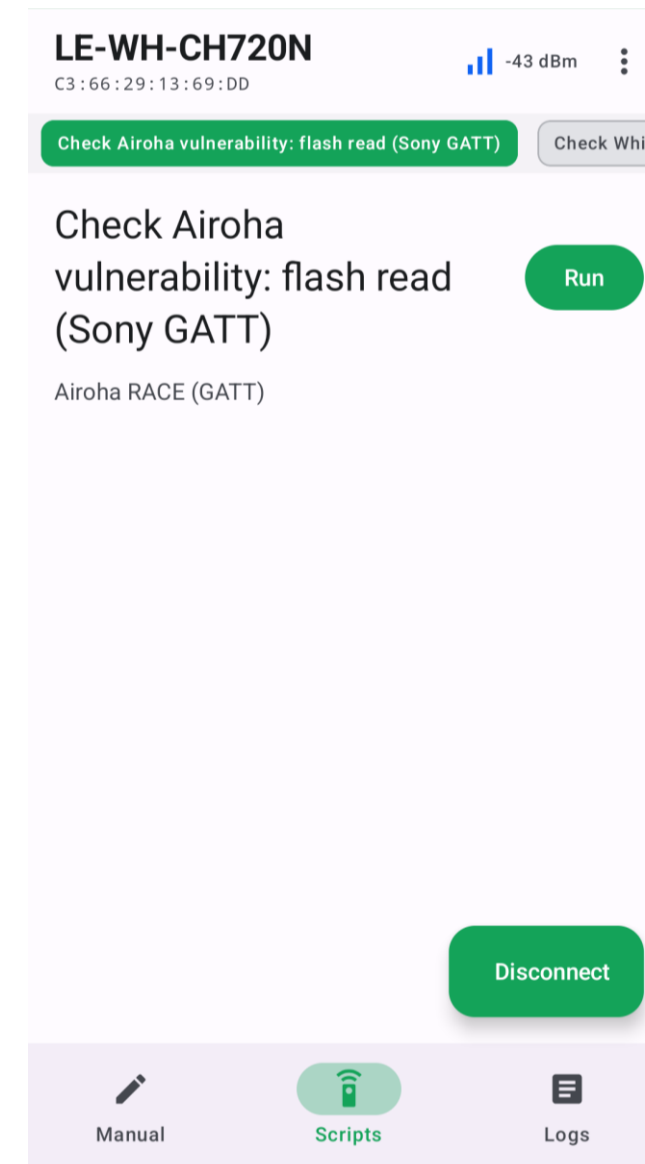
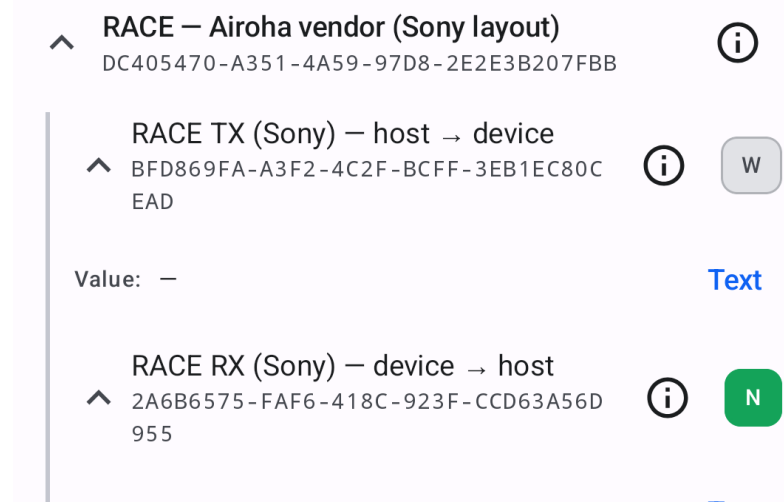
Find our headset (visible both as Fast Pair and WH-CH720N)



 Connect to the headset

Run “Scripts” check, or perform it manually (subscribe “N” to RX, then “W” to transmit).

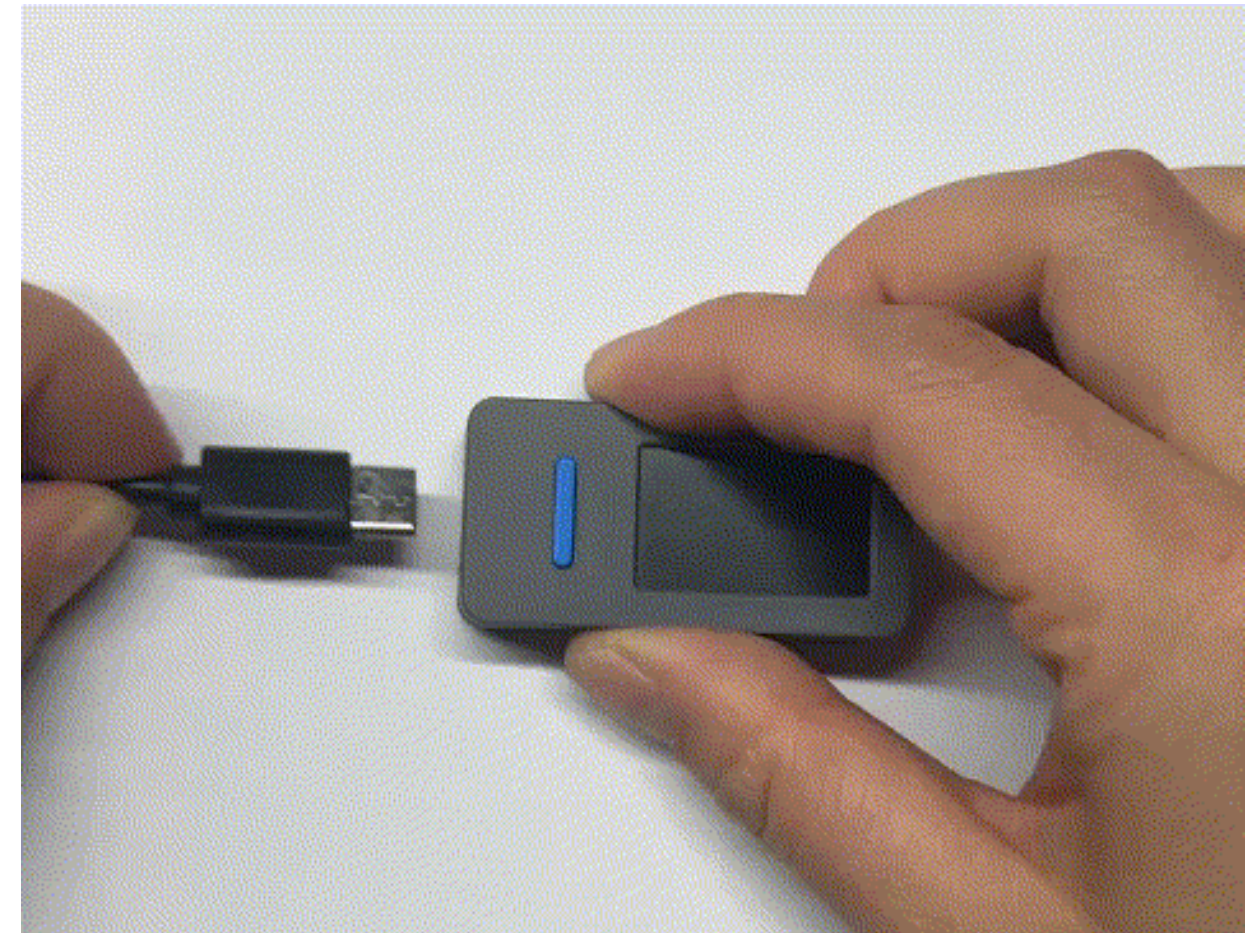
Note: to check WhisperPair simulate the device.



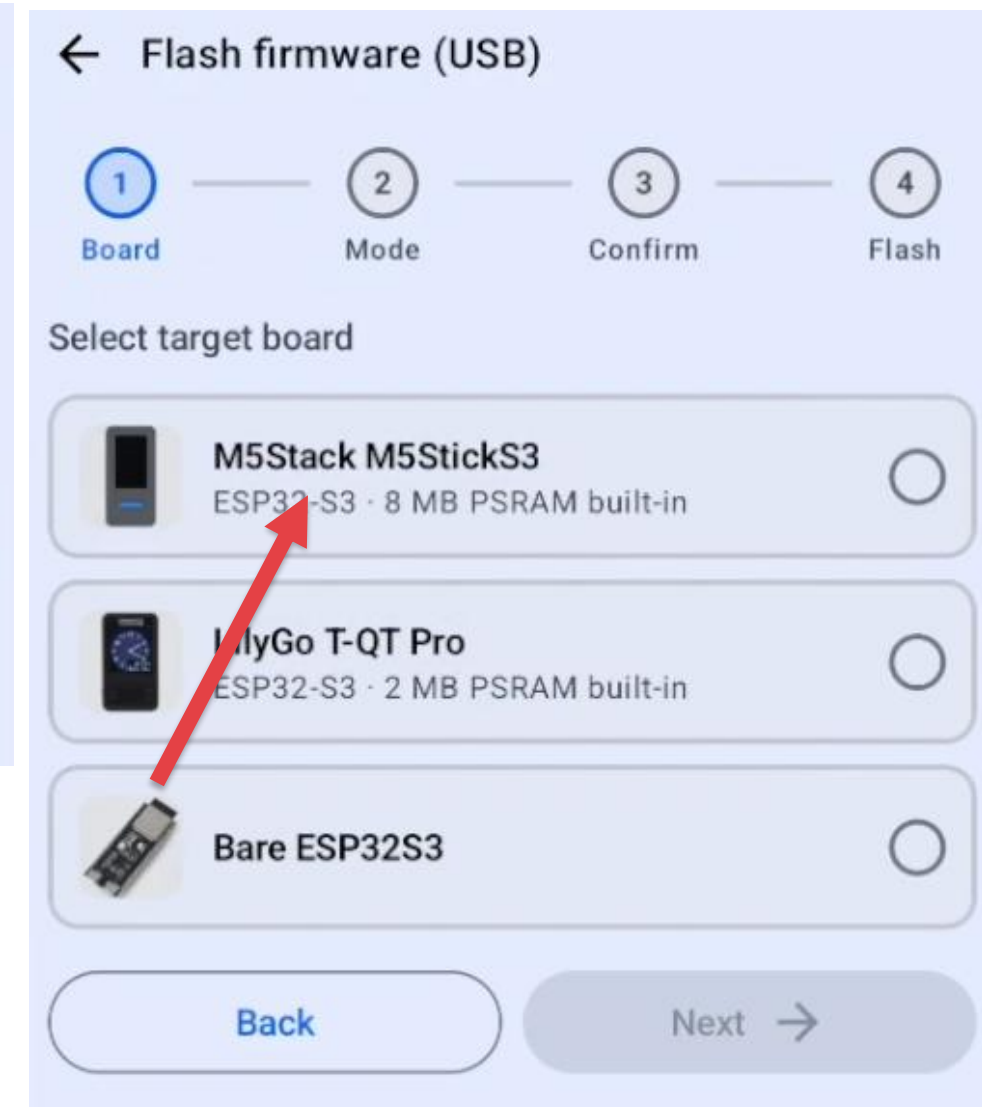
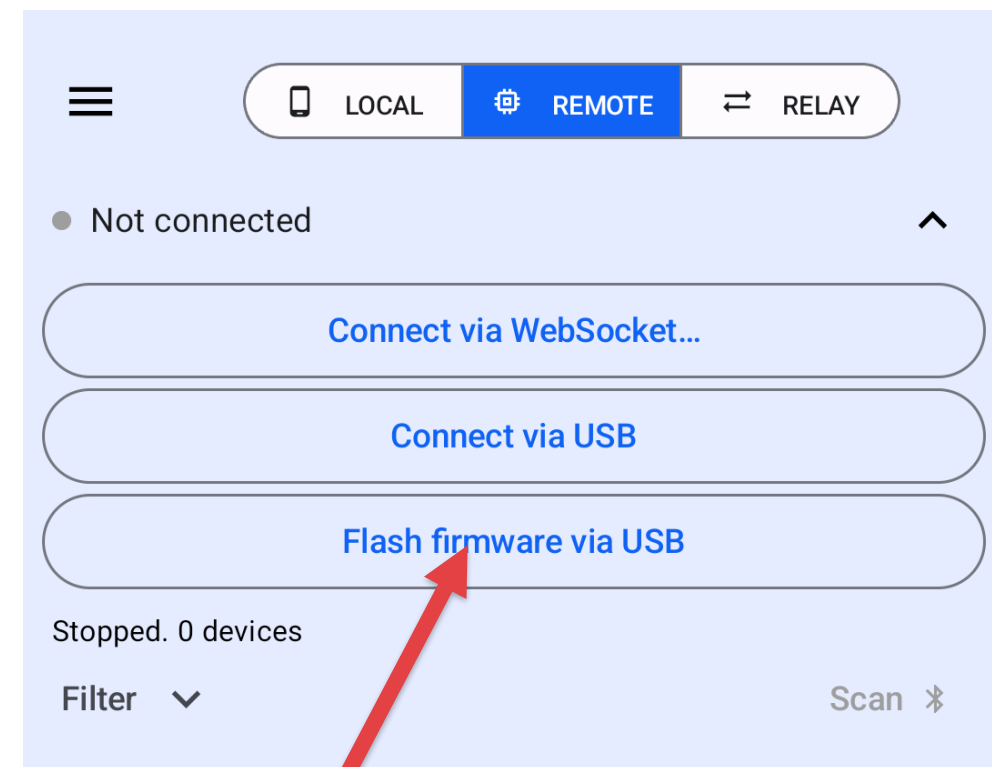
Flashing your badge with BLESPlO.it

Set the badge in the bootloader mode

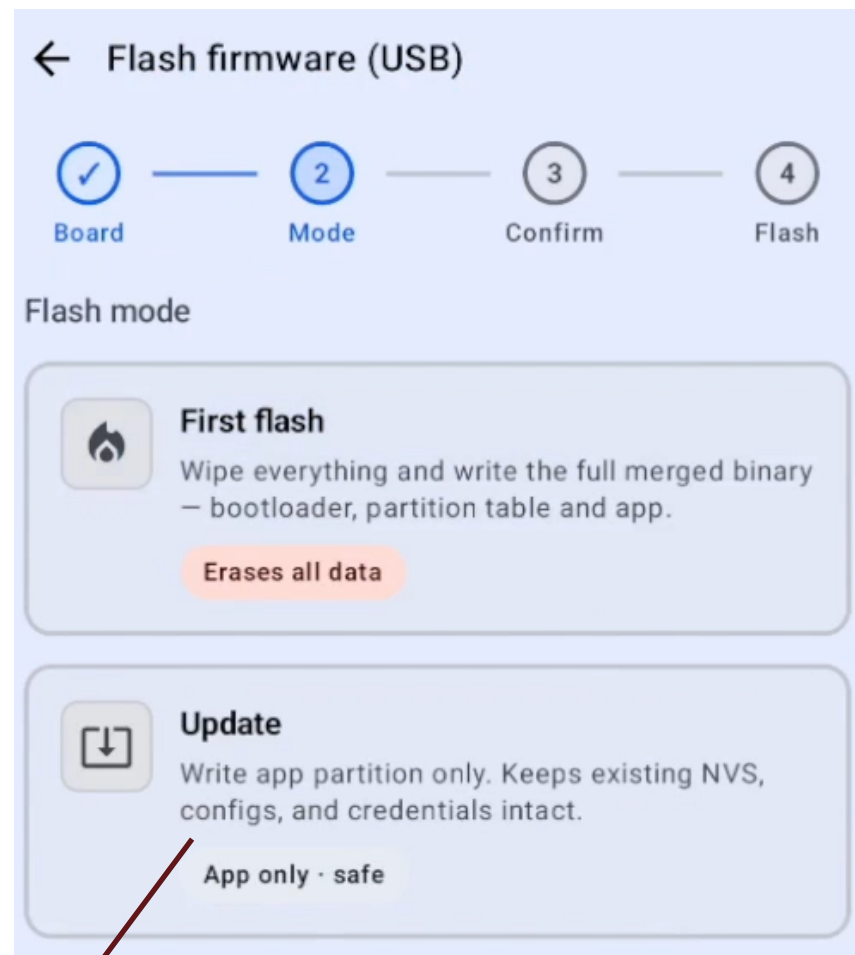
- Hold the left button until it flashes green
- Connect it via USB-C to the phone



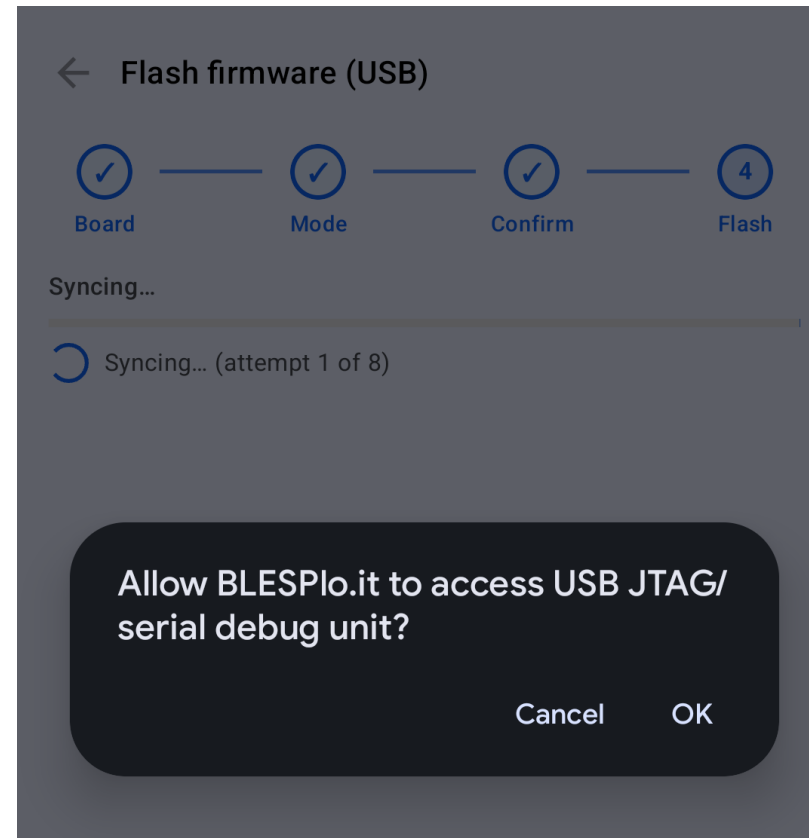
Flashing



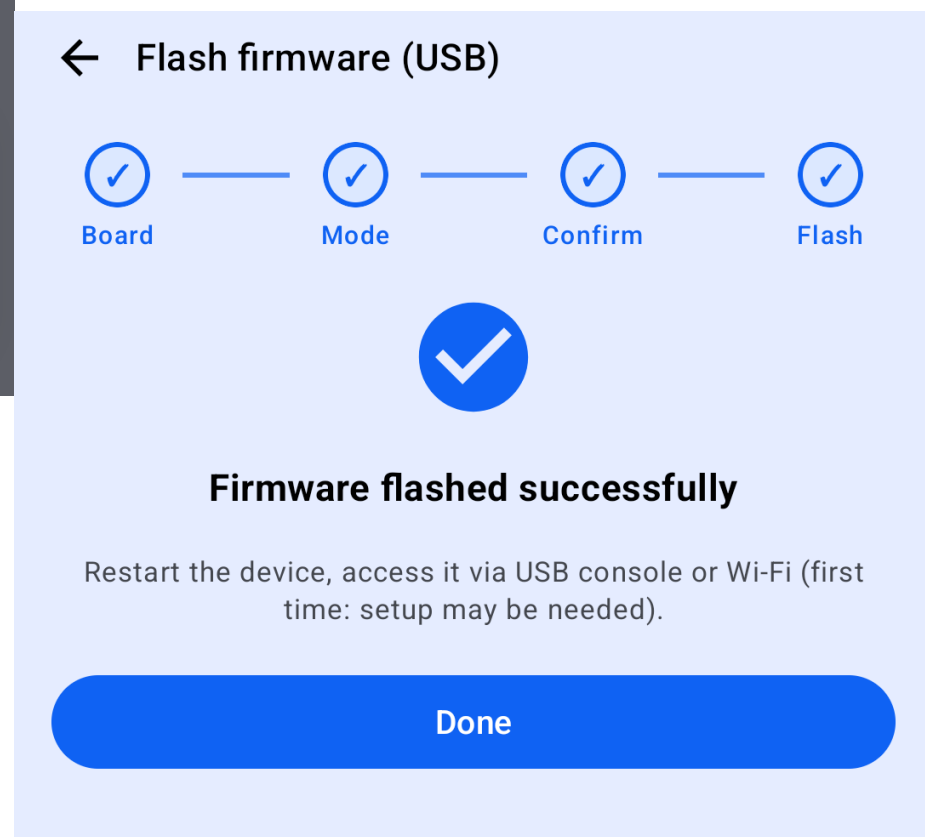
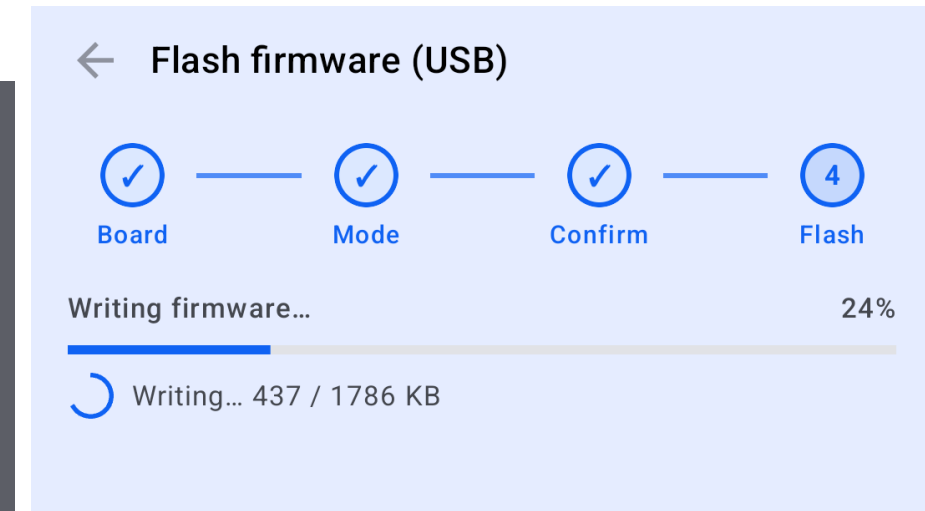
Flash “update” over our badge (it will keep wifi)



Update will keep the wifi settings



Connect ESP32 in Bootloader mode (blinking green)



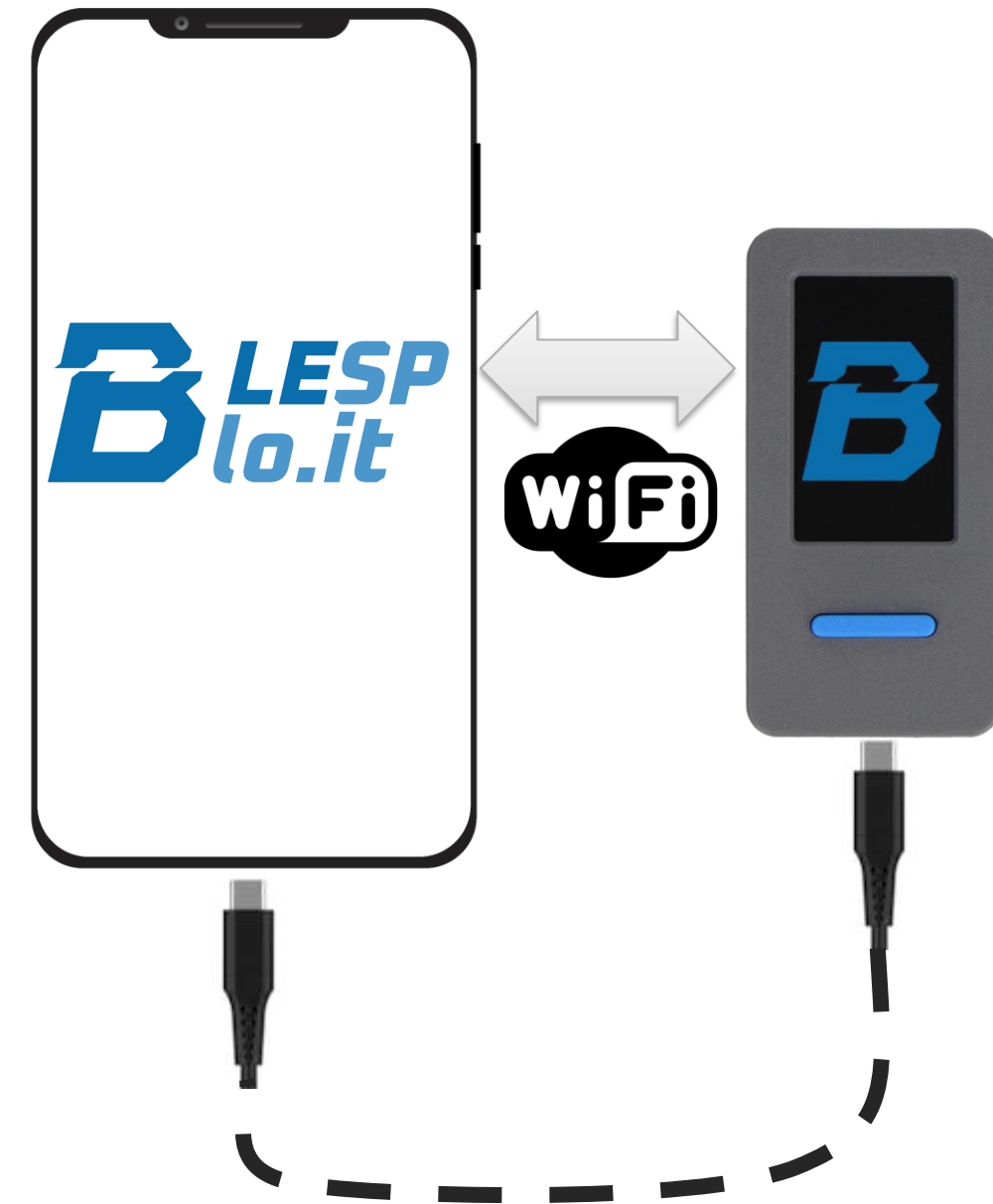
Connecting to ESP32

1. WiFi

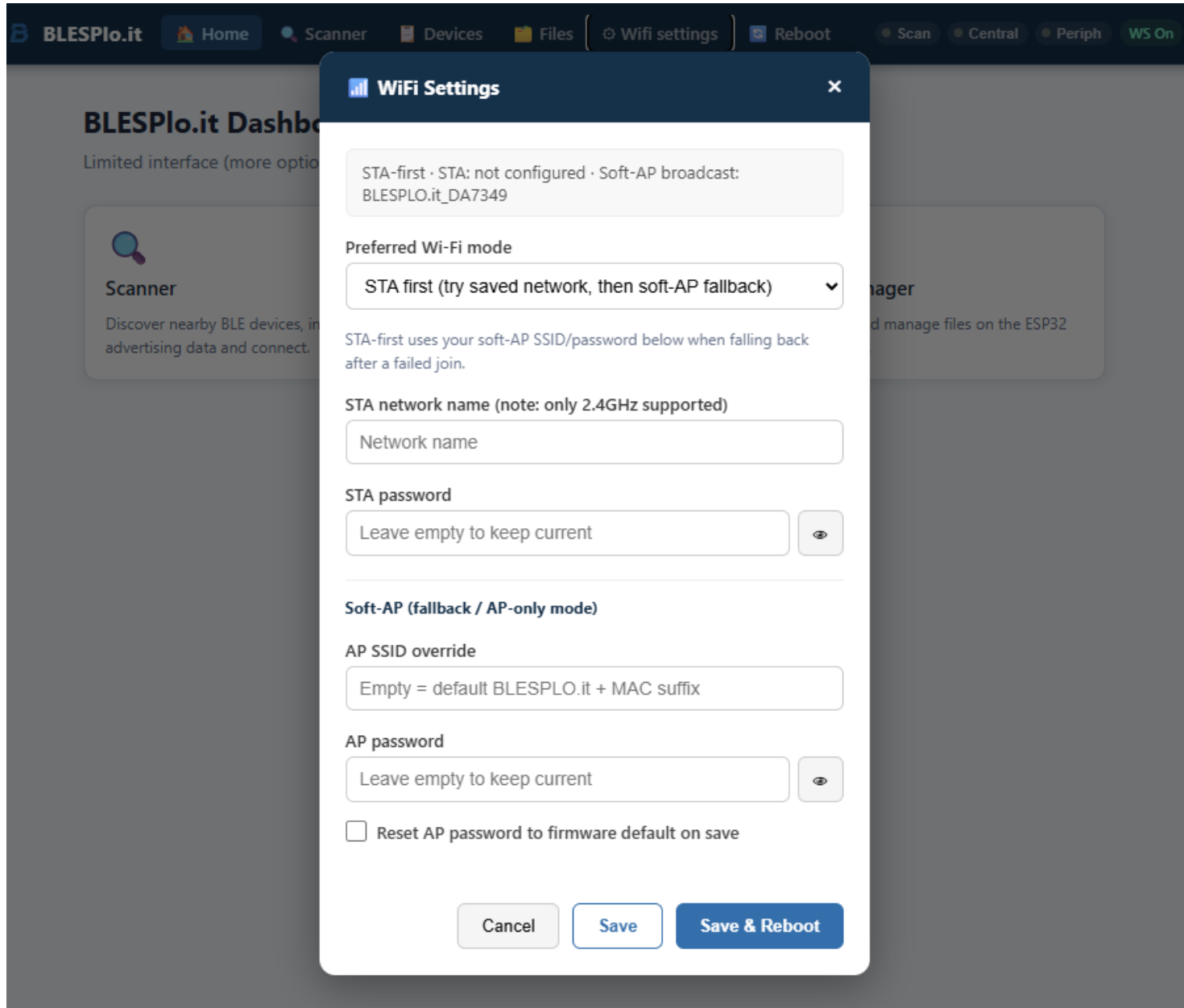
- Our badge: should connect to badge wifi
- At home starts in AP mode, ssid: BLESPlo.it_AABB, pass: BLESPlo.it
- Possible to adjust: after connecting to wifi AP or in the console (note: 2.4GHz only supported)

2. USB serial: config + debug log

- From the phone directly or from laptop (115200 baud)



Option: wifi settings



WiFi Settings

STA-first · STA: not configured · Soft-AP broadcast: BLESPLo.it_DA7349

Preferred Wi-Fi mode

STA first (try saved network, then soft-AP fallback) ▼

STA-first uses your soft-AP SSID/password below when falling back after a failed join.

STA network name (note: only 2.4GHz supported)

Network name

STA password

Leave empty to keep current

Soft-AP (fallback / AP-only mode)

AP SSID override

Empty = default BLESPLo.it + MAC suffix

AP password

Leave empty to keep current

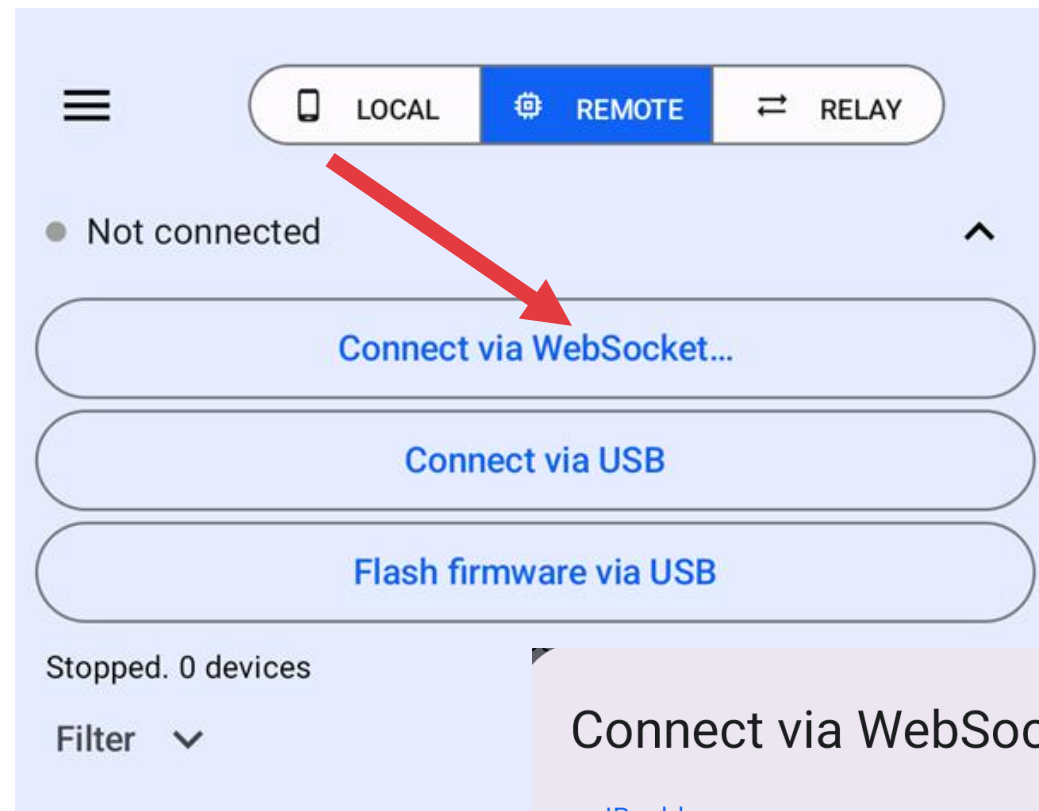
☐ Reset AP password to firmware default on save

Cancel Save Save & Reboot

```

BLESPlo.it > wifi
Current wifi settings:
Mode: STA-first
Networking: enabled
Current IP: 192.168.4.1
Soft-AP SSID (effective): BLESPLo.it_DA7349
Soft-AP SSID (custom NVS): (none - MAC suffix default)
Soft-AP PSK: (firmware default)
STA: no credentials
wifi -s -p | -M sta|ap | -a AP-SSID | -q AP-PSK | -e | -f
STA/AP WiFi config, soft-AP overrides, mode preference
-s, --ssid=<ssid> STA SSID
-p, --PSK=<psk> STA password
-a, --ap-ssid=<ssid> Soft-AP SSID (omit value or "" clears → MAC default)
-q, --ap-psk=<psk> Soft-AP WPA2 password (empty clears → firmware default)
-M, --mode=<sta|ap> sta = STA-first + AP fallback, ap = AP-only
-e, --on Enable networking (uses existing credentials)
-f, --off Disable networking
  
```

Wifi connection from the app



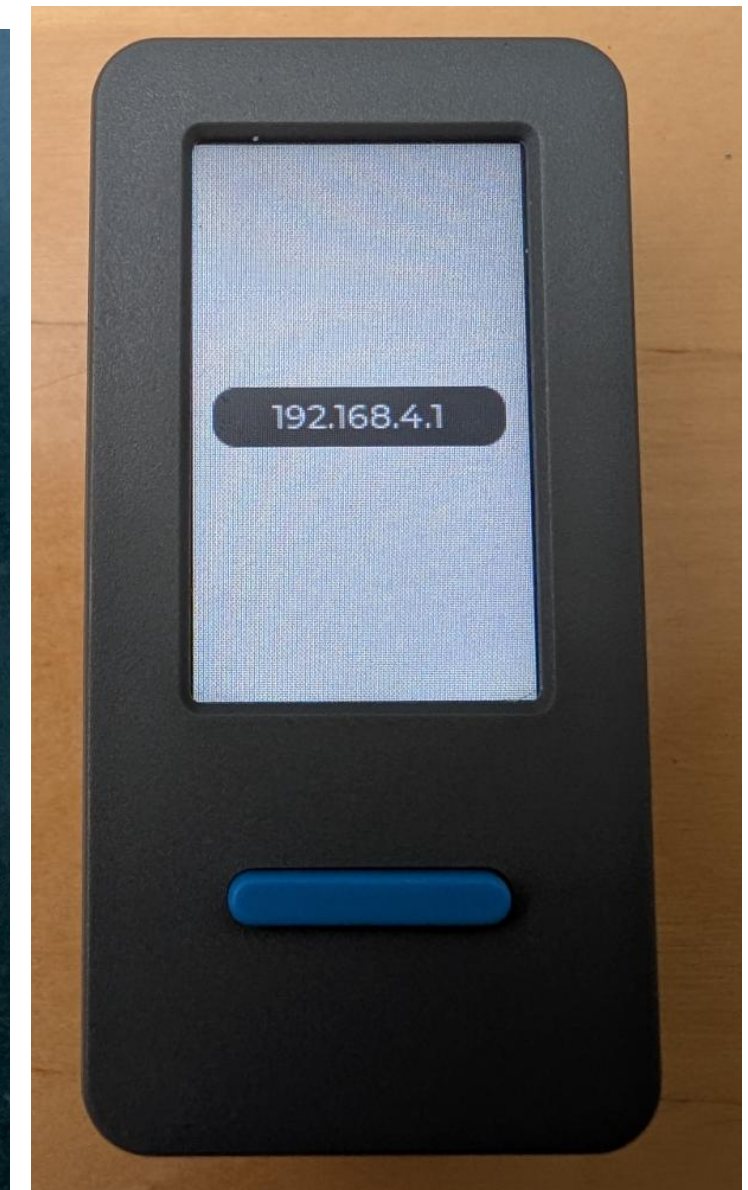
Connect via WebSocket

IP address

Port

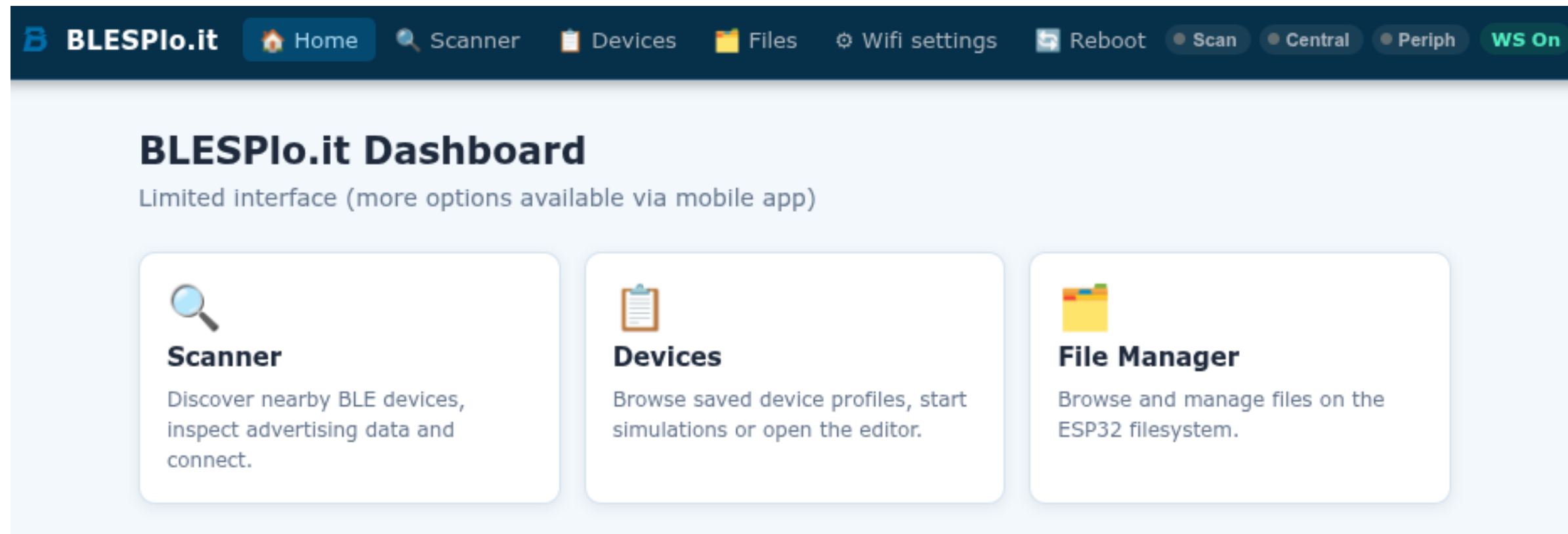
Cancel Connect

Ssid: orangecon-badge
Pass: orangecon



Option: direct access (limited features)

You can also access ESP32 directly (web interface), e.g. <http://192.168.4.1>



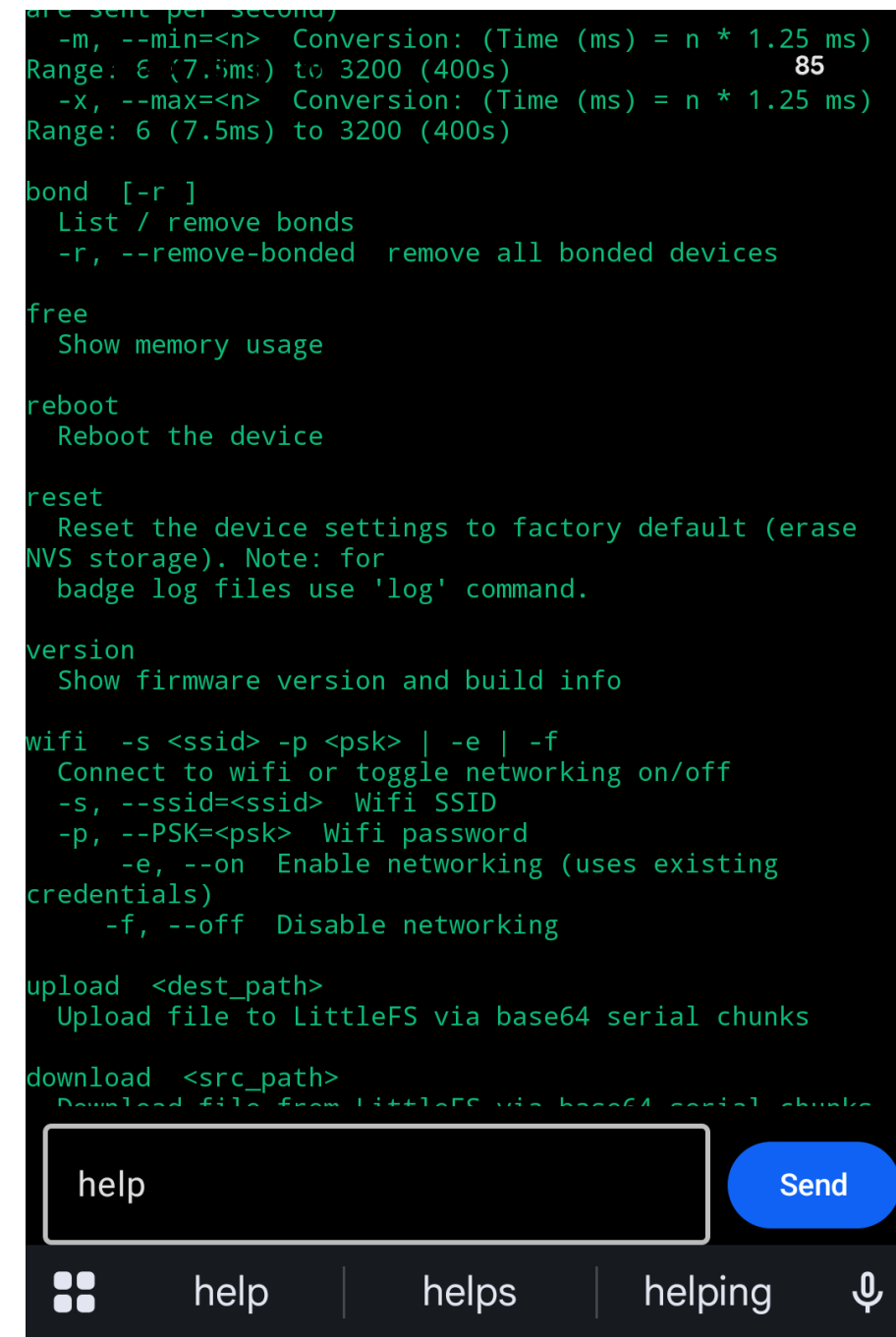
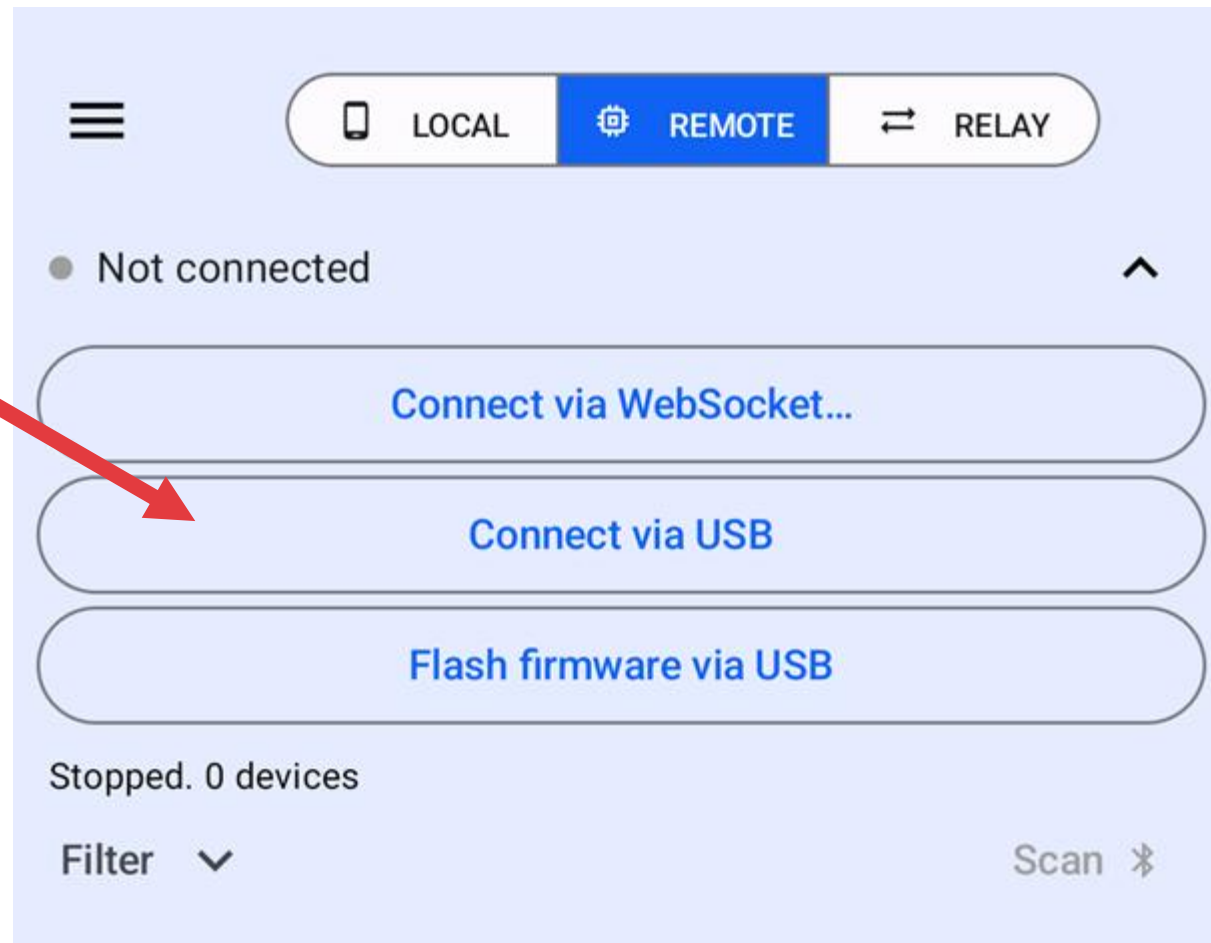
Option: serial console

Mostly useful for diagnostics, but also simple configuration (e.g. set wifi)

```
W (3976836) BLE sim - SMP: No pairing info available
I (3976836) BLE sim - GATT: registered service 0x1800 with handle=1
I (3976836) BLE sim - GATT: registering characteristic 0x2a00 with def_handle=2 val_handle=3 ✓ matches original
I (3976836) BLE sim - GATT: registering characteristic 0x2a01 with def_handle=4 val_handle=5 ✓ matches original
I (3976836) BLE sim - GATT: registering characteristic 0x2a01 with def_handle=6 val_handle=7 ✓ matches original
I (3976836) BLE sim - GATT: registered service 0x1801 with handle=8
I (3976836) BLE sim - GATT: registering characteristic 0x2a05 with def_handle=9 val_handle=10 ✓ matches original
I (3976836) BLE sim - GATT: registering characteristic 0x2a01 with def_handle=12 val_handle=13 ✓ matches original
I (3976846) BLE sim - GATT: registering characteristic 0x2a01 with def_handle=14 val_handle=15 ✓ matches original
I (3976846) BLE sim - GATT: registered service fa879af4-d601-420c-b2b4-07ffb528dde3 with handle=16
I (3976846) BLE sim - GATT: registering characteristic 10e2fde2-d7fe-4845-b3f3-a32010ebb095 with def_handle=17 val_handle=18 ✓ matches original
I (3976846) BLE sim - GATT: registering characteristic b02eaeaa-f6bc-4a7e-bc94-f7b7fc8ded0b with def_handle=20 val_handle=21 ✓ matches original
I (3976846) BLE sim - core: GATT services added successfully
I (3976846) NimBLE: GAP procedure initiated: extended advertise; instance=0

I (3976846) BLE_sim - GAP: Instance 0 (main) started - static
I (3976846) utils: === Memory BLE sim ready - PSRAM: 8038448
I (3976846) LVGL UI: Loading PNG from buf at 13,66 size 108x108
I (3976846) utils: INTERNAL: 88011, largest block: 31744, min ever: 55783
I (3976846) web server - device sim: Started simulation for: rapidpower_robotic_dog
I (3976856) web server websocket: Broadcast {"type":"sim_status","status":"started","device":"rapidpower_robotic_dog","adv":[{"id":"main","instance":0,"running":true,"b
d_addr":"70:04:1D:DA:73:4A","addr_type":"public_hardware"}],"src":"ESP_DA734A"} to 1 clients
I (3976866) web server websocket: Broadcast {"type":"gfx","cmd":"png","id":"dog","path":"/static/rapidpower_robotic_dog/rapidpower_dog_128.png","align":"center","x":0,"
y":0,"w":80,"h":80,"src":"ESP_DA734A"} to 1 clients
I (3976866) websocket gfx: gfx: {"type":"gfx","cmd":"png","id":"dog","path":"/static/rapidpower_robotic_dog/rapidpower_dog_128.png","align":"center","x":0,"y":0,"w":80,
"h":80}
```

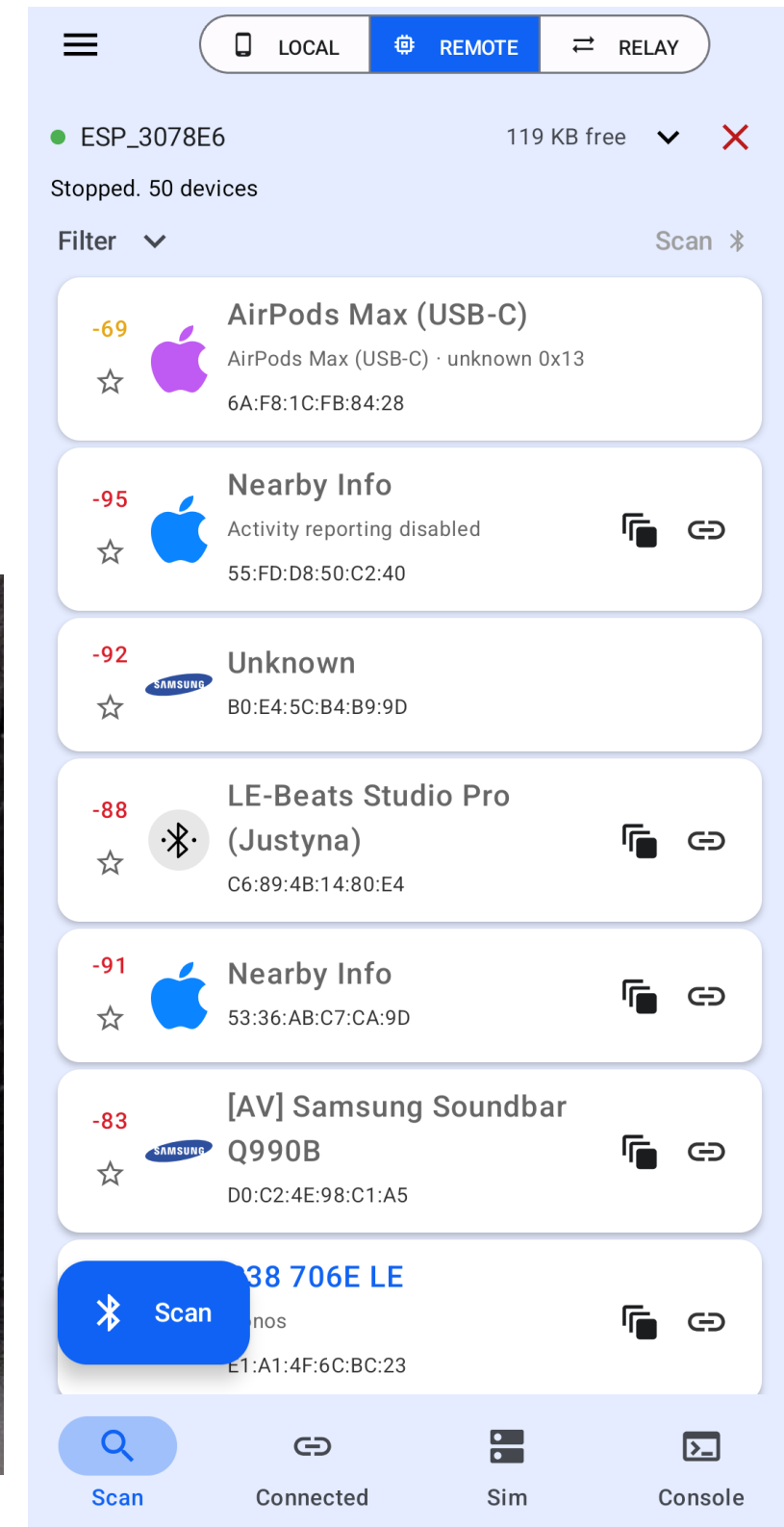
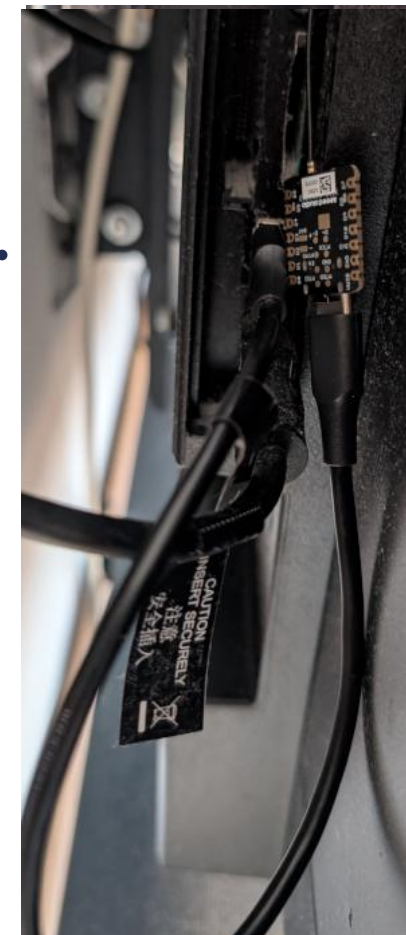
Option: connect to console directly from the phone (USB C)



Remote Scan

Scan: works like local but using ESP32 Bluetooth adapter remotely (works also via Internet).

Remotely scan, connect, control devices (“Bluetooth proxy”). Full relay coming next.

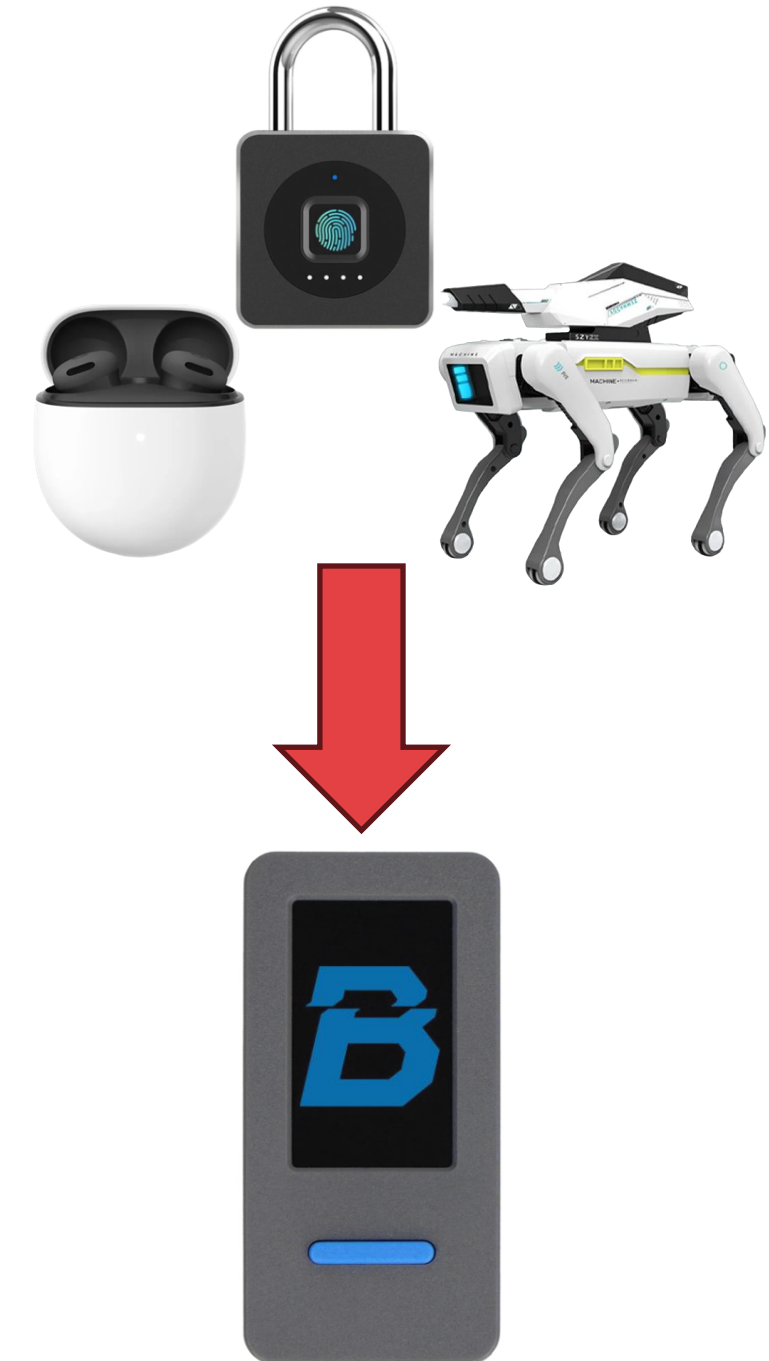


Remote Sim

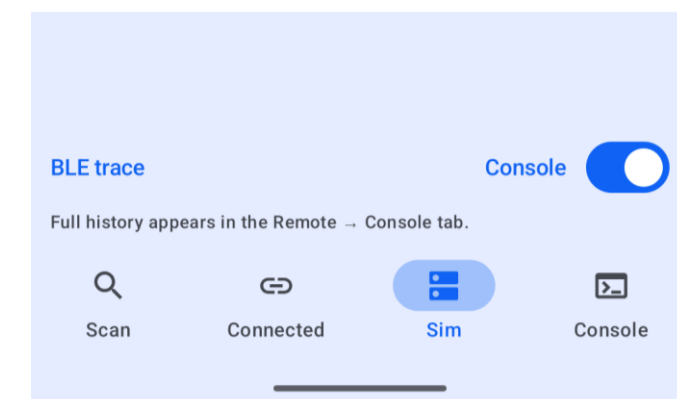
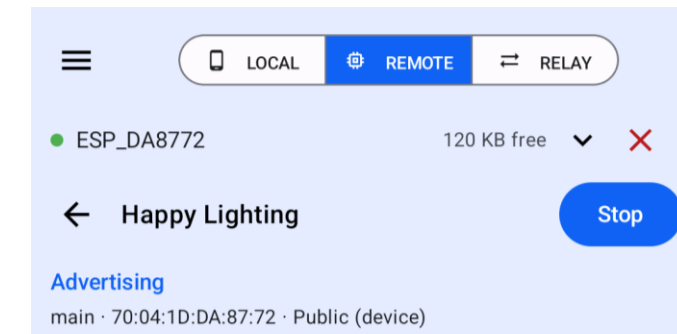
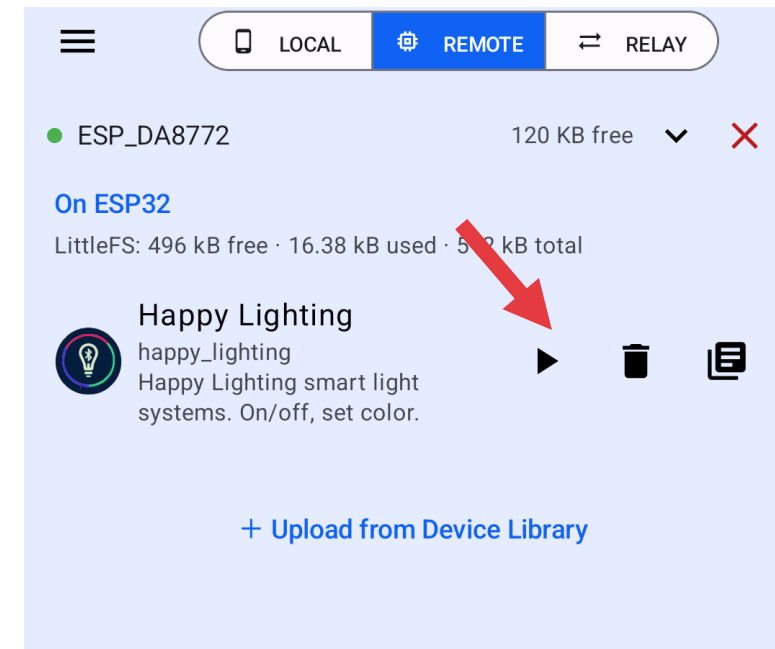
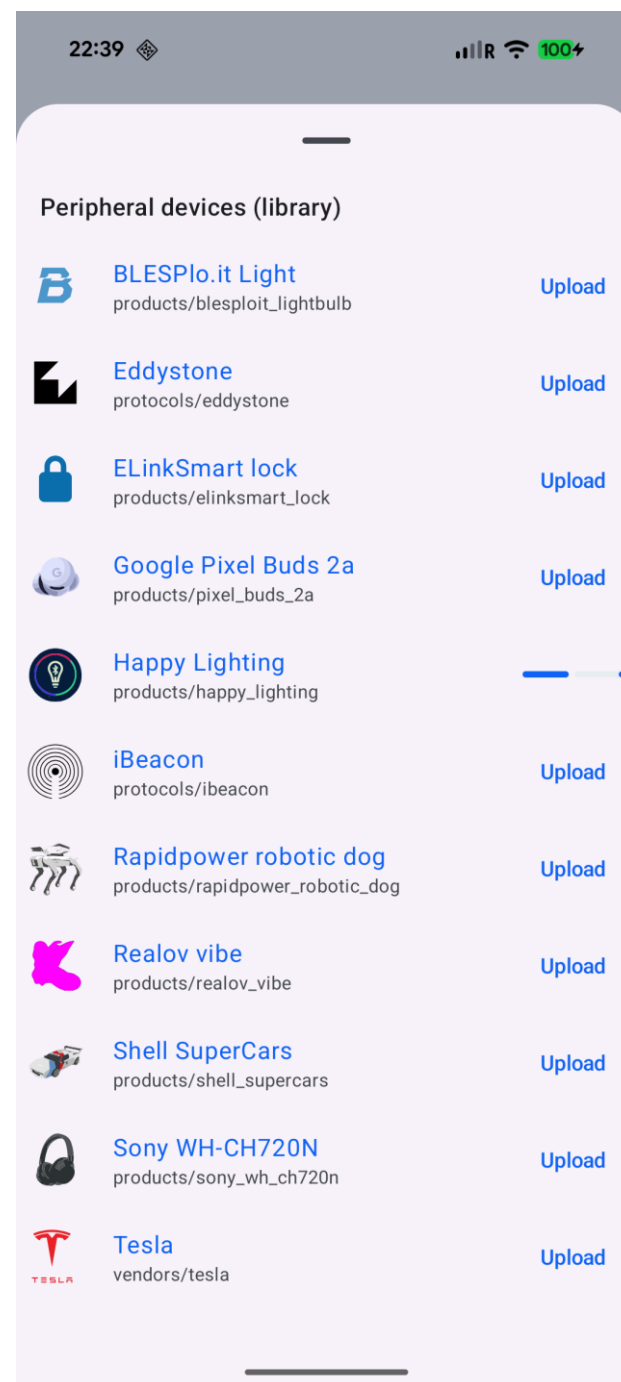
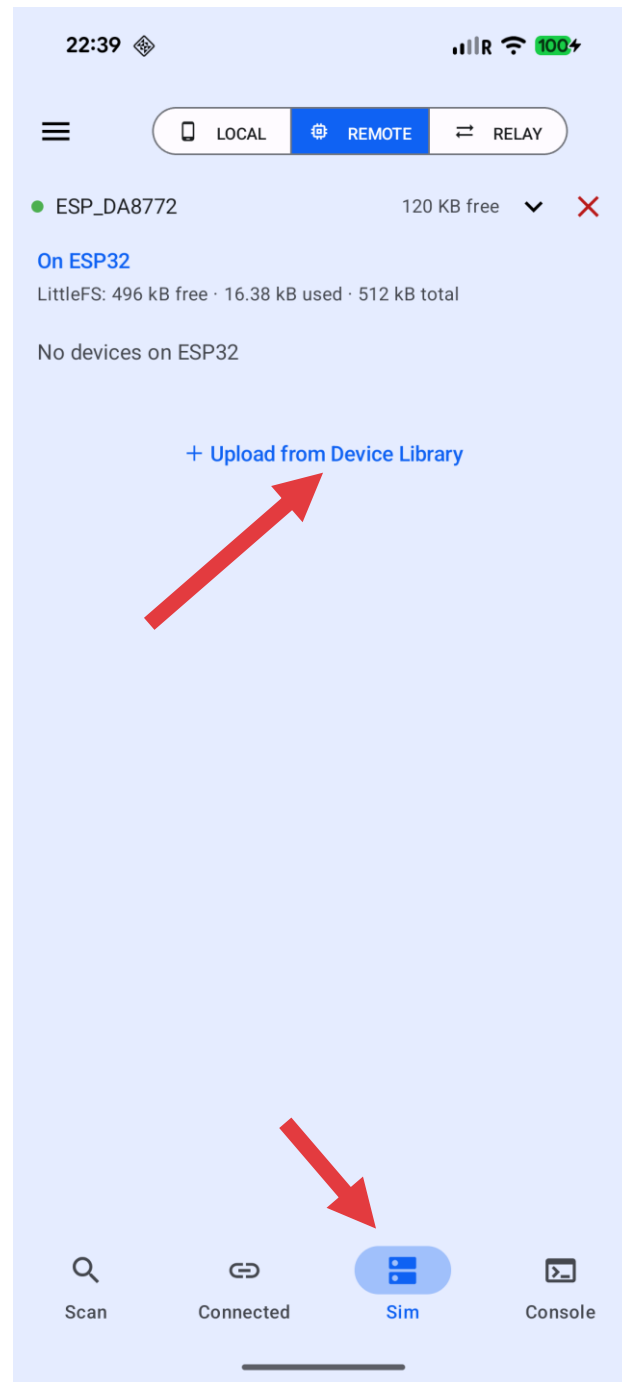
Simulate any BLE device on ESP32

- Test application without the need to have real hardware
- Prototype, test tools in a stable environment
- Simply capture application traffic in various device states
- Remove link-layer pairing (app will not notice).
- Attack the mobile app by malicious device

Several example “peripheral” profiles in the Device Library



Upload device to ESP32 for simulation



Popular smart lightbulb

HappyLighting



Happy Lighting

happy_lighting

Happy Lighting smart light systems. On/off, set color.



Control from original application:

2.9★
15.2K reviews ⓘ

5M+
Downloads

<https://play.google.com/store/apps/details?id=com.xiaoyu.hlight>

<https://apps.apple.com/us/app/happylighting-life-with-smart/id1145694075>



Happy Lighting

products/happy_lighting

v1 bundled

Slawomir Jasek



Happy Lighting smart light systems. On/off, set color.

Product / model URL
<https://happylightingapp.com/>

Companion apps



Direct download links



confirmed APK

v1.6.49

<https://apkpure.com/happylighting/com.xiaoyu.hlight/download/1.6.49>



Google Play

There are also several unofficial ones available.

<https://play.google.com/store/apps/details?id=com.xiaoyu.hlight>



Apple App Store

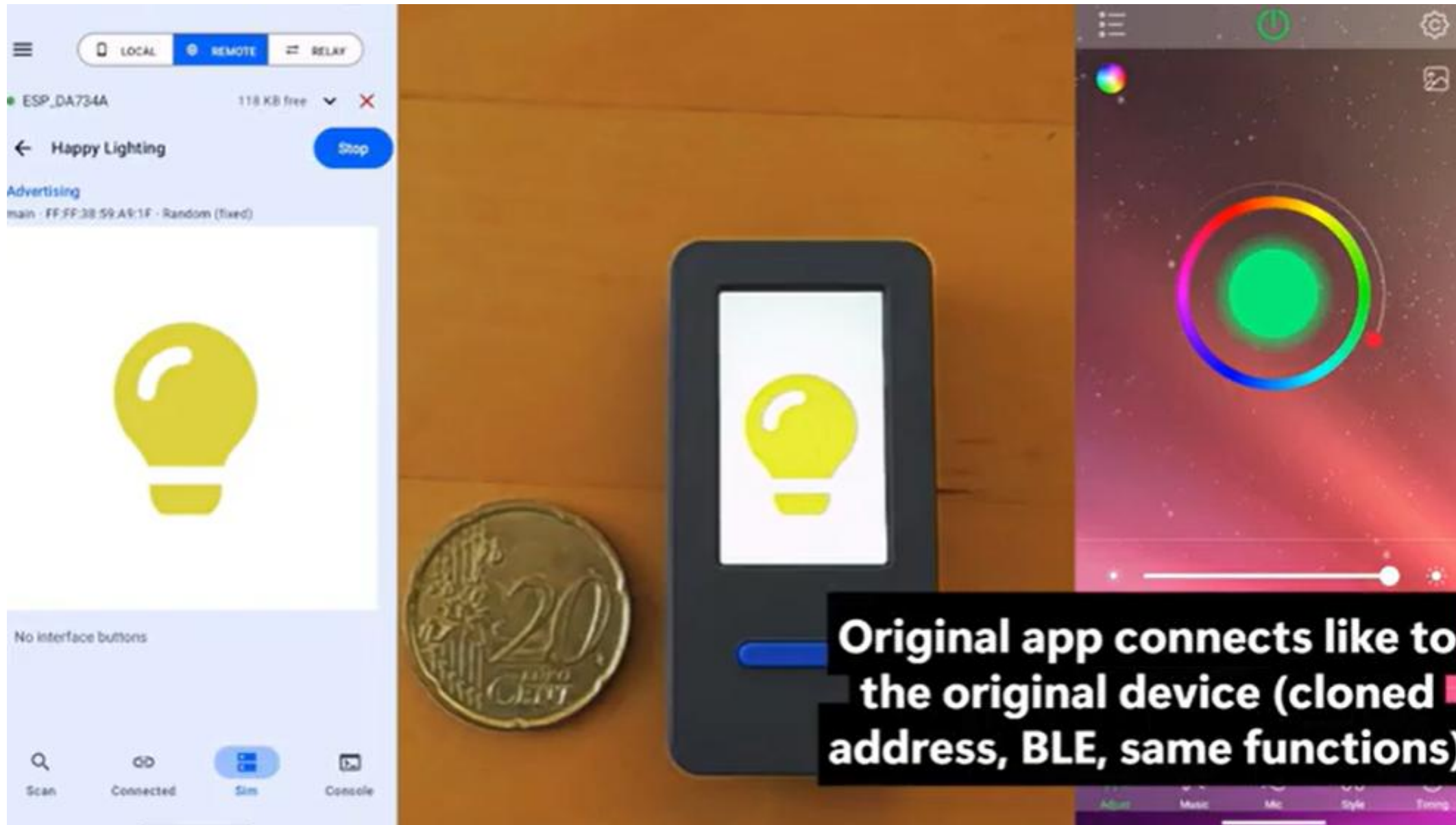
<https://apps.apple.com/us/app/happylighting-life-with-smart/id1145694075>

Notes

Confirmed working with 'Triones' lightbulb. Other device names extracted from the app (unconfirmed). The device seems to have a PIN authentication, but it is not enforced nor visible anywhere the user interface. The a...

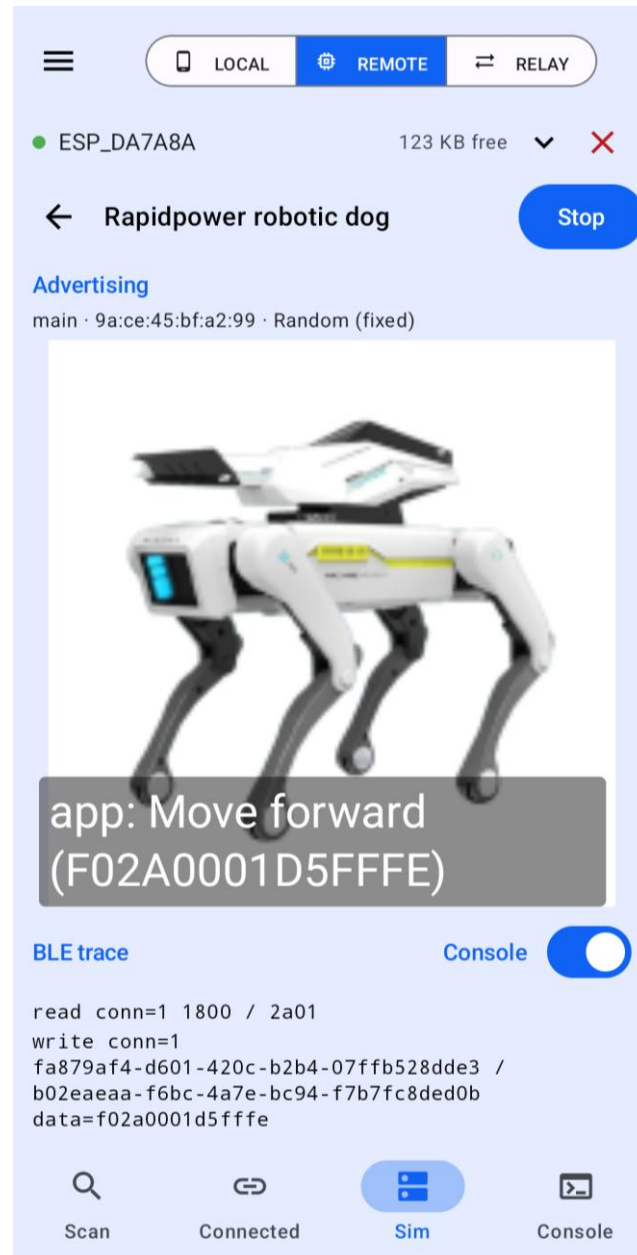
Show more

Happy Lighting demo



<https://www.youtube.com/watch?v=zUJHDkzPpjU>

Simulate robotic dog



Control from the official app:

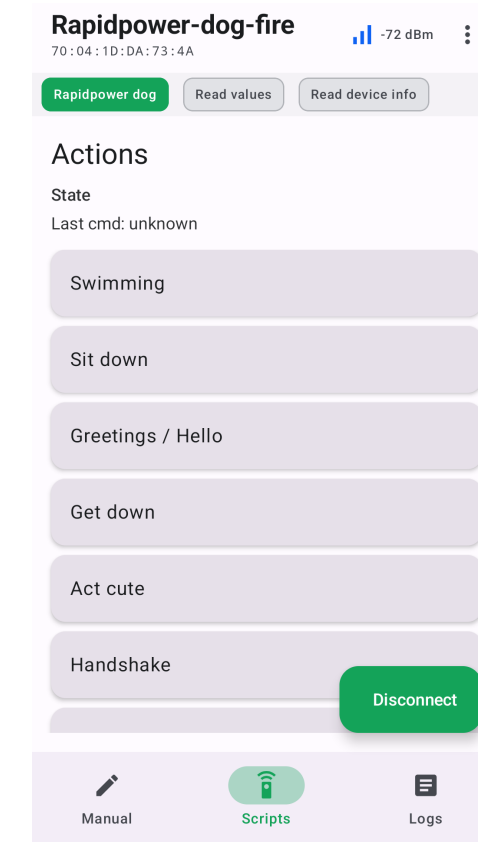
<https://play.google.com/store/apps/details?id=com.zhongrun.robotdog>



Use the controller

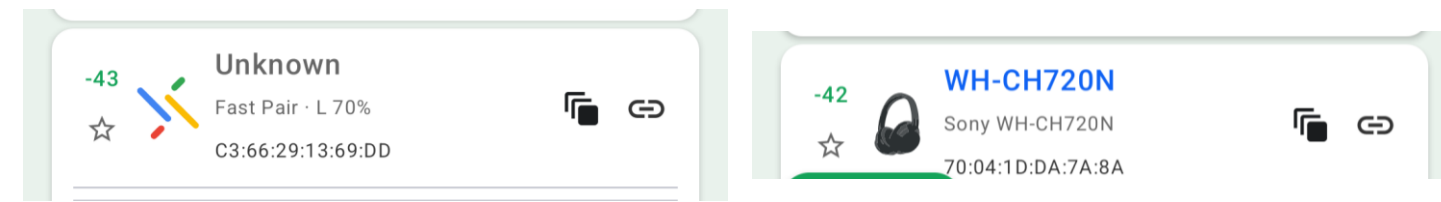


Use the “Local” mode to connect and control it from the menu

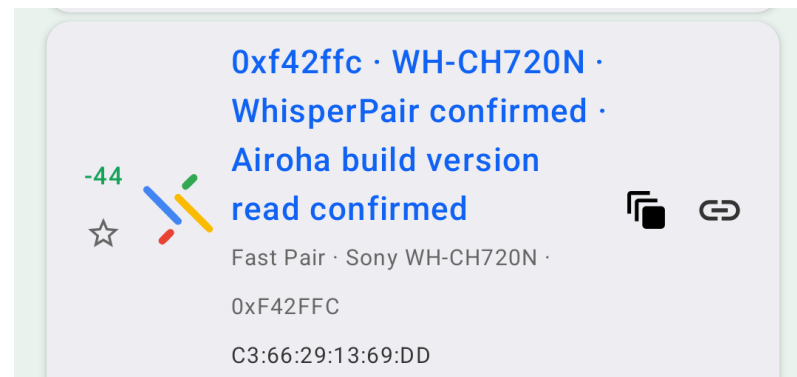


WhisperPair and Airoha

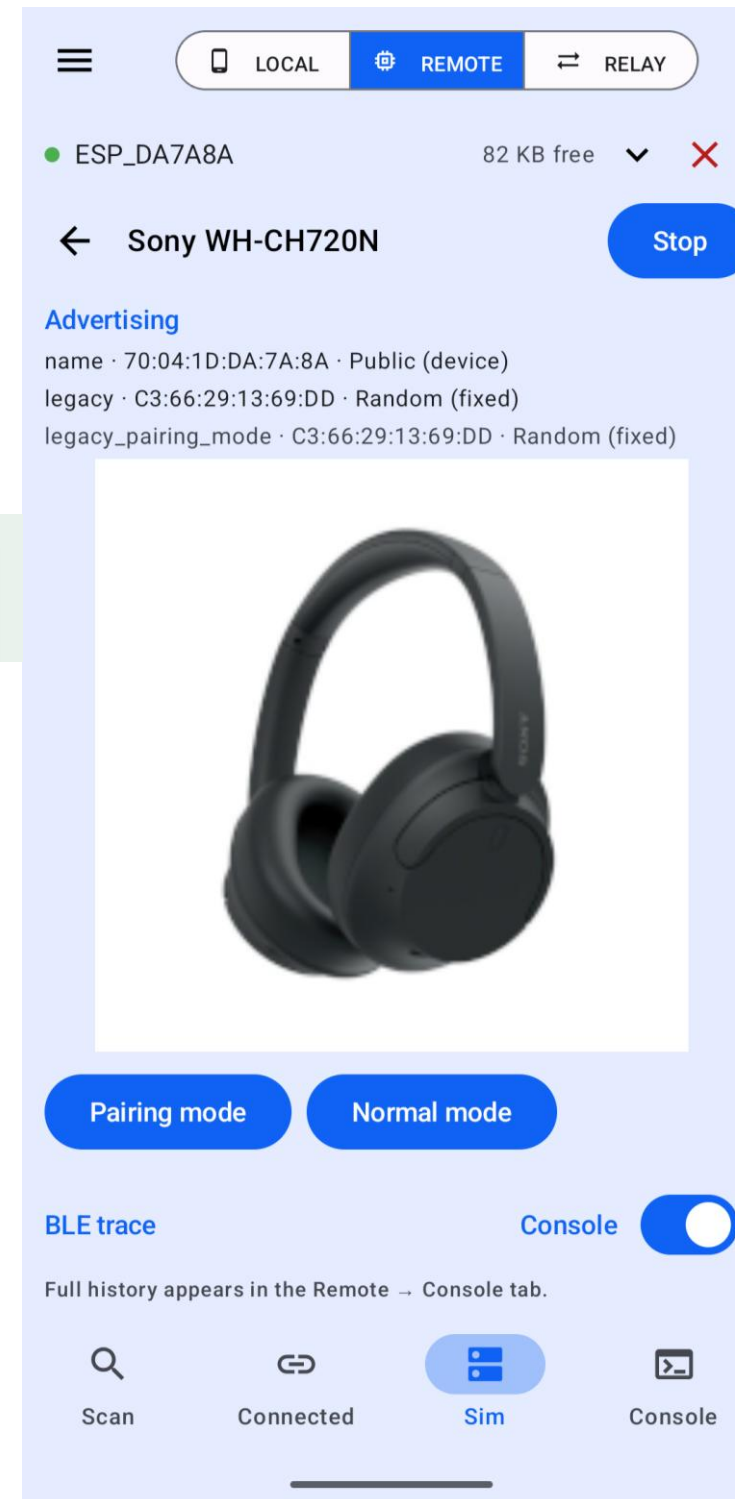
Upload “Sony WH-CH720N”, Find your simulated device (strongest signal)



Check for WhisperPair and Airoha (Quick Action or manually).



Optionally set “Pairing mode” to see the device Fast Pair pop-up



Sony

C3:66:29:13:69:CC
Fast Pair · L 51%

RSSI
-96 dBm

Last seen
0s ago

Stay connected

On: after a successful quick script, switches to Manual and keeps this peripheral session. Off: disconnect and go back (device list).



✓ Capture (this session) · no svc

Scripts

▶ Check Airoha vulnerability: flash read (Sony GATT)
Matched dynamic script.

▶ Check WhisperPair vulnerability
Matched dynamic script.

▶ Check Airoha vulnerability: build version (Sony GATT)
Matched dynamic script.

▶ Check Fast Pair model
Matched dynamic script.

More to come!

B **LESP**
lo.it

<https://blesplo.it>