Repurposing

ESP32 Based

Commercial Products

GO Jimmy PI

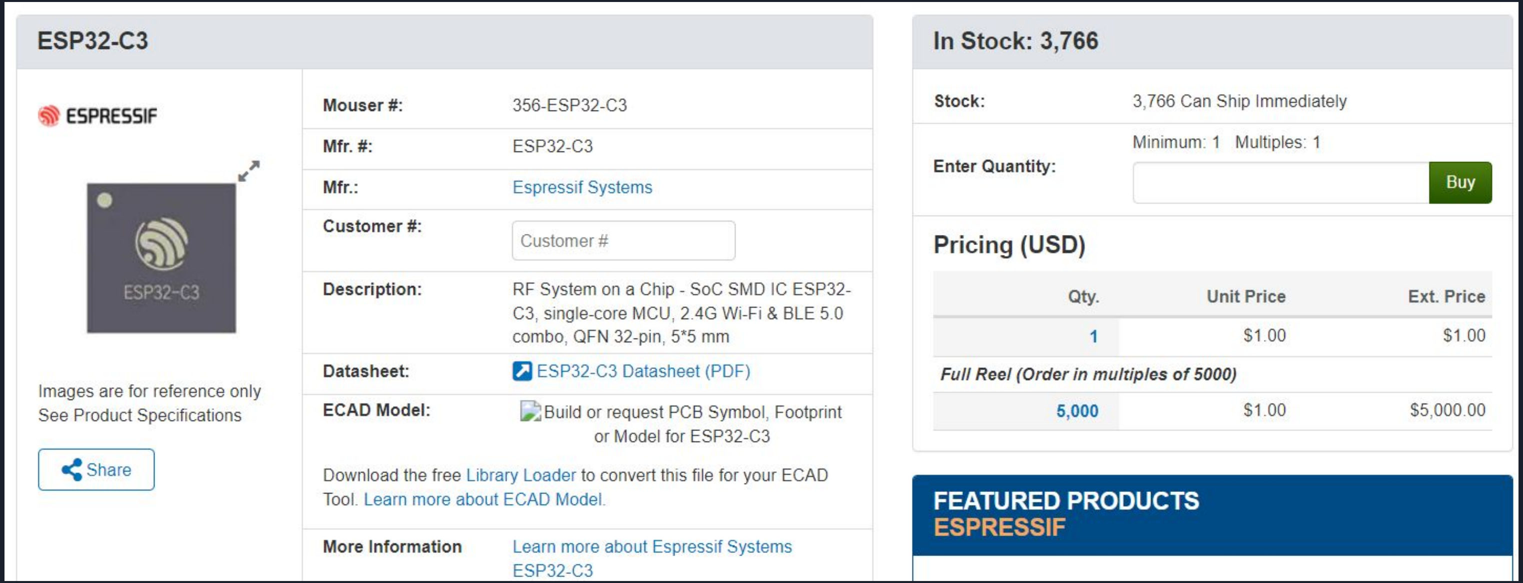
https://hackaday.io/superconference/index.html

gojimmypi.github.io

Topics Today

- What is an ESP32?
- Why repurpose a product?
- How to get started & what is needed
- Where to find example code
- Examples!

What is an ESP32?



ESP32-C3

ESPRESSIF

Images are for reference only
See Product Specifications

Share

Mouser #:	356-ESP32-C3
Mfr. #:	ESP32-C3
Mfr.:	Espressif Systems
Customer #:	Customer #
Description:	RF System on a Chip - SoC SMD IC ESP32-C3, single-core MCU, 2.4G Wi-Fi & BLE 5.0 combo, QFN 32-pin, 5*5 mm
Datasheet:	ESP32-C3 Datasheet (PDF)
ECAD Model:	Build or request PCB Symbol, Footprint or Model for ESP32-C3

Download the free Library Loader to convert this file for your ECAD Tool. Learn more about ECAD Model.

More Information	Learn more about Espressif Systems ESP32-C3

In Stock: 3,766

Stock:	3,766 Can Ship Immediately
Enter Quantity:	Minimum: 1 Multiples: 1 Buy

Pricing (USD)

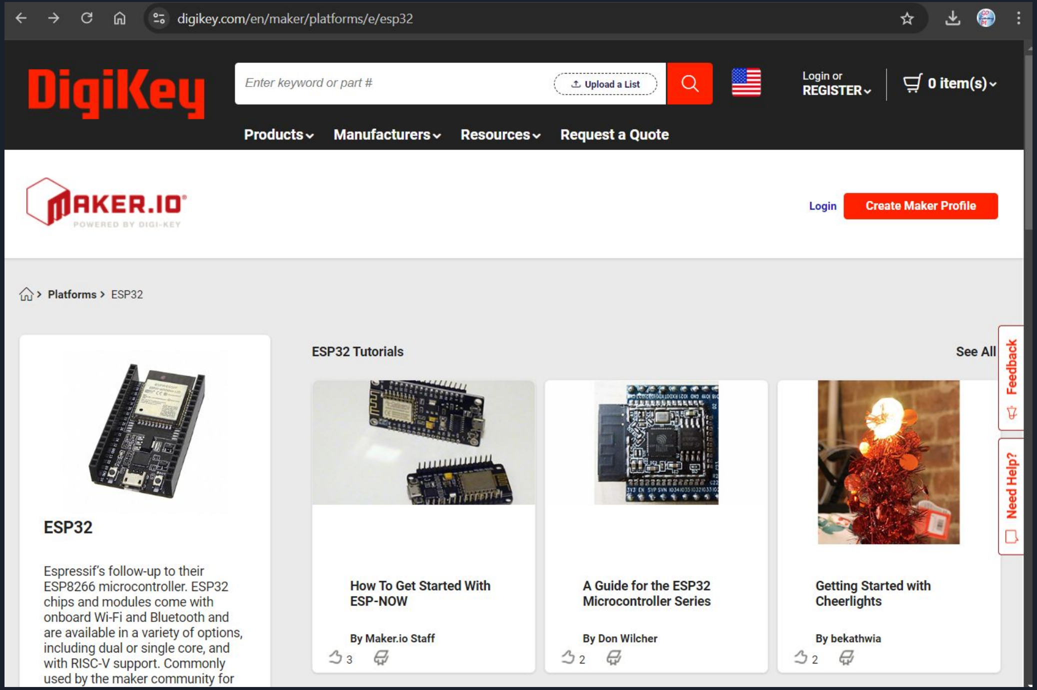
Qty.	Unit Price	Ext. Price
1	$1.00	$1.00
Full Reel (Order in multiples of 5000)		
5,000	$1.00	$5,000.00

FEATURED PRODUCTS
ESPRESSIF

Add WiFi to a project for about $1, plus small amount of PCB real estate. Chip is 5x5 mm!!

Image credit: screen snip from mouser.com

DigiKey has ESP32 maker.io content



ESP32

Espressif's follow-up to their ESP8266 microcontroller. ESP32 chips and modules come with onboard Wi-Fi and Bluetooth and are available in a variety of options, including dual or single core, and with RISC-V support. Commonly used by the maker community for

ESP32 Tutorials

See All

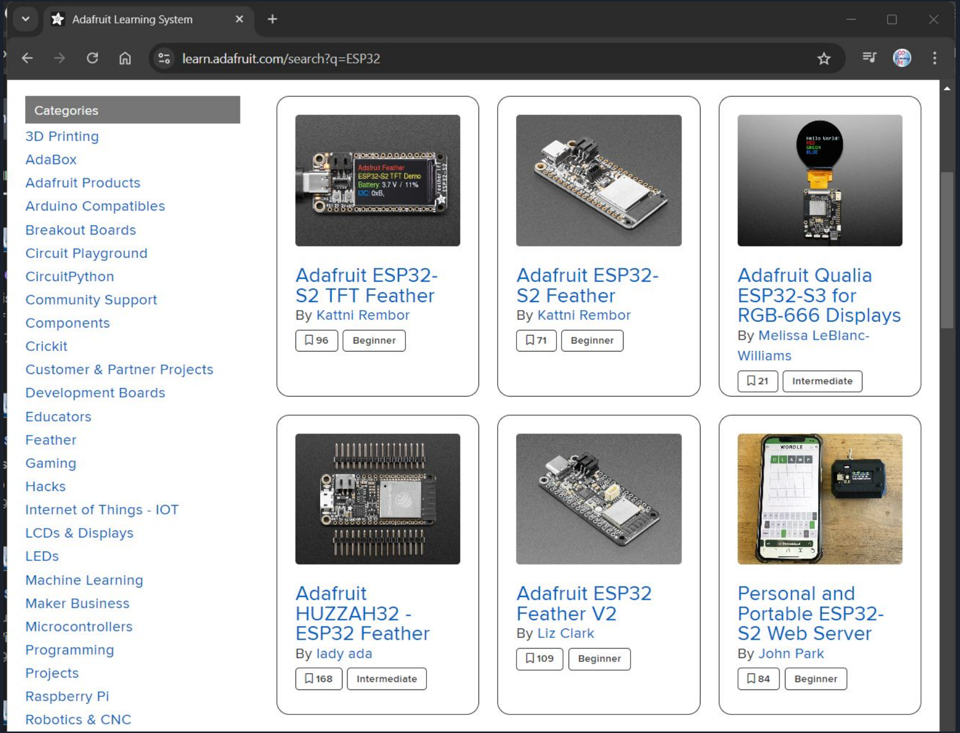
How To Get Started With ESP-NOW
By Maker.io Staff
3

A Guide for the ESP32 Microcontroller Series
By Don Wilcher
2

Getting Started with Cheerlights
By bekathwia
2

Image credit: screen snip from digikey.com/en/maker/platforms/e/esp32

Adafruit: learn.adafruit.com



Categories

- 3D Printing
- AdaBox
- Adafruit Products
- Arduino Compatibles
- Breakout Boards
- Circuit Playground
- CircuitPython
- Community Support
- Components
- Crickit
- Customer & Partner Projects
- Development Boards
- Educators
- Feather
- Gaming
- Hacks
- Internet of Things - IOT
- LCDs & Displays
- LEDs
- Machine Learning
- Maker Business
- Microcontrollers
- Programming
- Projects
- Raspberry Pi
- Robotics & CNC

Adafruit ESP32-S2 TFT Feather
By Kattni Rembor
96 | Beginner

Adafruit ESP32-S2 Feather
By Kattni Rembor
71 | Beginner

Adafruit Qualia ESP32-S3 for RGB-666 Displays
By Melissa LeBlanc-Williams
21 | Intermediate

Adafruit HUZZAH32 - ESP32 Feather
By lady ada
168 | Intermediate

Adafruit ESP32 Feather V2
By Liz Clark
109 | Beginner

Personal and Portable ESP32-S2 Web Server
By John Park
84 | Beginner

Image credit: screen snip from learn.adafruit.com

ESP32 Block Diagram

Wi-Fi 802.11

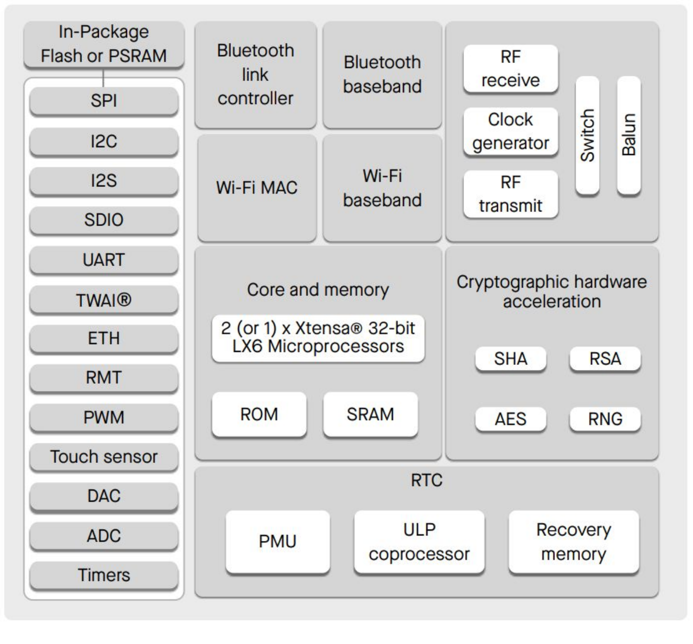
Bluetooth

RAM / ROM

GPIO

UART / SPI / I2C

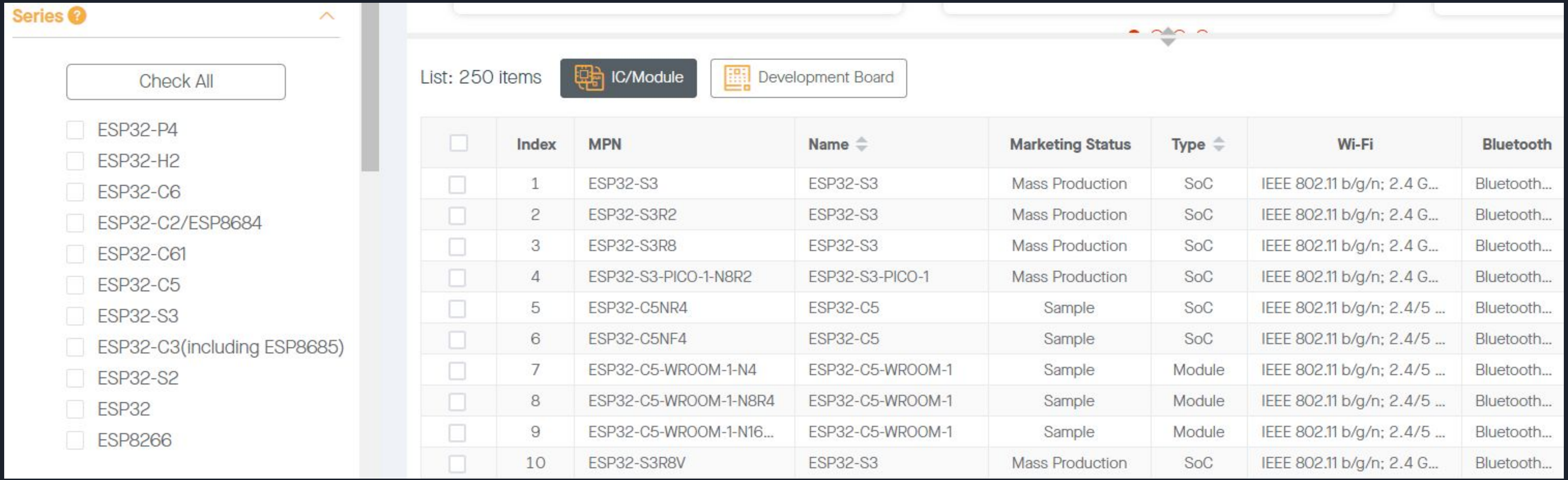
Hardware Accelerated Cryptographic Functions



ESP32 Functional Block Diagram

Image credit: Espressif ESP32 Series Datasheet

Over 200 devices
Each having different power, GPIO, RAM, Flash, etc



Series ?

Check All

- □ ESP32-P4
- □ ESP32-H2
- □ ESP32-C6
- □ ESP32-C2/ESP8684
- □ ESP32-C61
- □ ESP32-C5
- □ ESP32-S3
- □ ESP32-C3(including ESP8685)
- □ ESP32-S2
- □ ESP32
- □ ESP8266

List: 250 items

IC/Module | Development Board

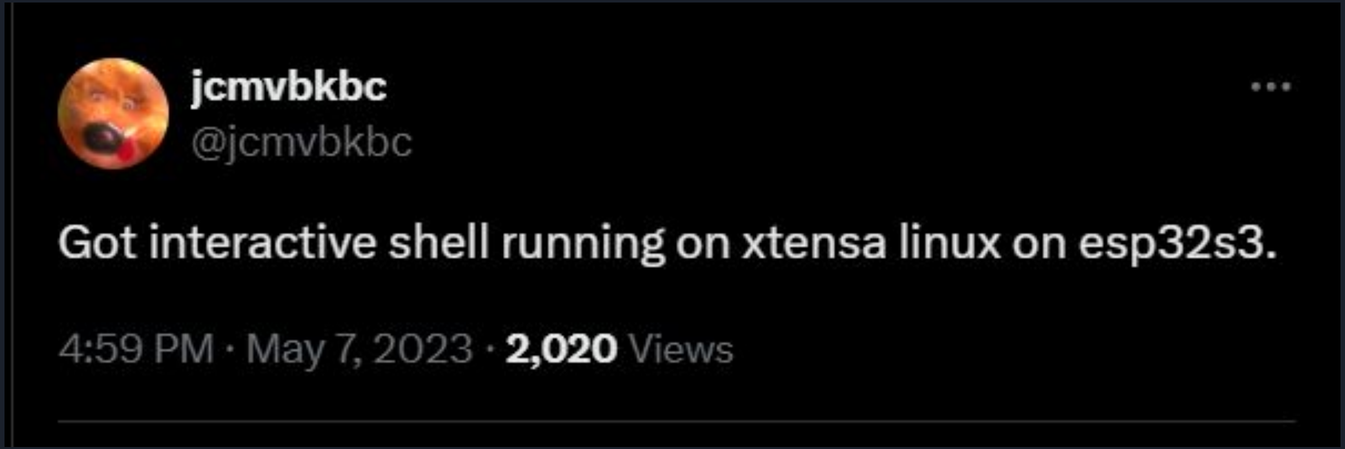
□	Index	MPN	Name	Marketing Status	Type	Wi-Fi	Bluetooth
□	1	ESP32-S3	ESP32-S3	Mass Production	SoC	IEEE 802.11 b/g/n; 2.4 G...	Bluetooth...
□	2	ESP32-S3R2	ESP32-S3	Mass Production	SoC	IEEE 802.11 b/g/n; 2.4 G...	Bluetooth...
□	3	ESP32-S3R8	ESP32-S3	Mass Production	SoC	IEEE 802.11 b/g/n; 2.4 G...	Bluetooth...
□	4	ESP32-S3-PICO-1-N8R2	ESP32-S3-PICO-1	Mass Production	SoC	IEEE 802.11 b/g/n; 2.4 G...	Bluetooth...
□	5	ESP32-C5NR4	ESP32-C5	Sample	SoC	IEEE 802.11 b/g/n; 2.4/5 ...	Bluetooth...
□	6	ESP32-C5NF4	ESP32-C5	Sample	SoC	IEEE 802.11 b/g/n; 2.4/5 ...	Bluetooth...
□	7	ESP32-C5-WROOM-1-N4	ESP32-C5-WROOM-1	Sample	Module	IEEE 802.11 b/g/n; 2.4/5 ...	Bluetooth...
□	8	ESP32-C5-WROOM-1-N8R4	ESP32-C5-WROOM-1	Sample	Module	IEEE 802.11 b/g/n; 2.4/5 ...	Bluetooth...
□	9	ESP32-C5-WROOM-1-N16...	ESP32-C5-WROOM-1	Sample	Module	IEEE 802.11 b/g/n; 2.4/5 ...	Bluetooth...
□	10	ESP32-S3R8V	ESP32-S3	Mass Production	SoC	IEEE 802.11 b/g/n; 2.4 G...	Bluetooth...

Image credit: Screen snip from https://products.espressif.com/#/product-selector

Microcomputer vs Microcontroller

Often the distinguishing characteristic is “*does it run one or many programs*”.

Well:



jcmvbkbc
@jcmvbkbc

Got interactive shell running on xtensa linux on esp32s3.

4:59 PM · May 7, 2023 · **2,020** Views

See also: gojimmypi.github.io/ESP32-S3-Linux/

Screen snip from: x.com/jcmvbkbc/status/1655361863947808768

Why?

?

Why repurpose a commercial product?

The obvious answer for Hackaday Attendees

IT’S FUN!

Beyond the Fun aspect… a product:

- Doesn’t do what you want
- Has known security vulnerabilities
- No longer supported by manufacturer

GO Jimmy PI

e-waste

- 15,000 tons of e-waste every year in “Agbogbloshie” (see links, below)
- Local companies send e-waste to the local landfills
- No end to examples

npr.org/sections/goats-and-soda/2024/10/05/g-s1-6411/electronics-public-health-waste-ghana-phones-computers

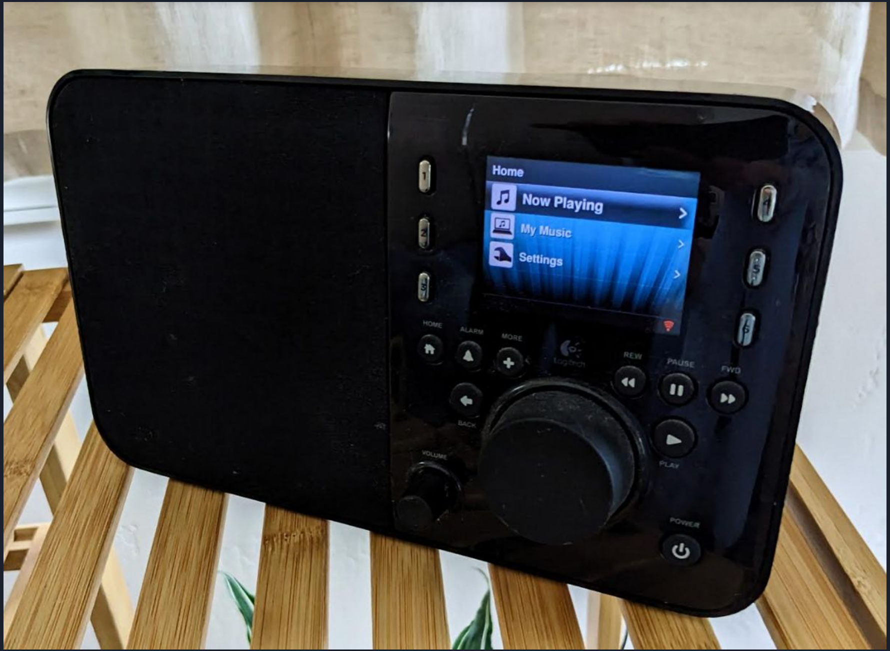
wikipedia.org/wiki/Agbogbloshie

Remember when things would LAST?

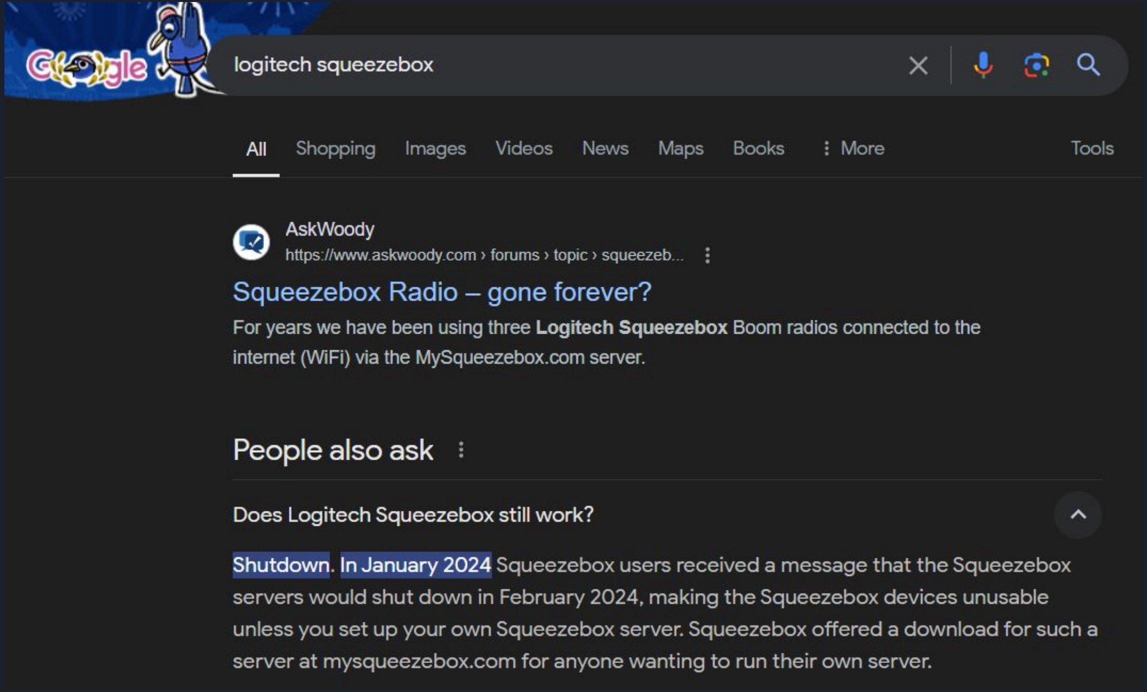
Sony Dream Machine was Sony's long-running line of clock radios. The line was introduced in the early 1960s and ran until the early 2010's

-- https://en.wikipedia.org/wiki/Sony_Dream_Machine

Squeezebox


Home
Now Playing
My Music
Settings
1
2
3
4
5
6
HOME
ALARM
MORE
Logitech
REW
PAUSE
FWD
BACK
VOLUME
PLAY
POWER

Logitech Squeezebox



AskWoody
https://www.askwoody.com › forums › topic › squeezeb...

Squeezebox Radio – gone forever?

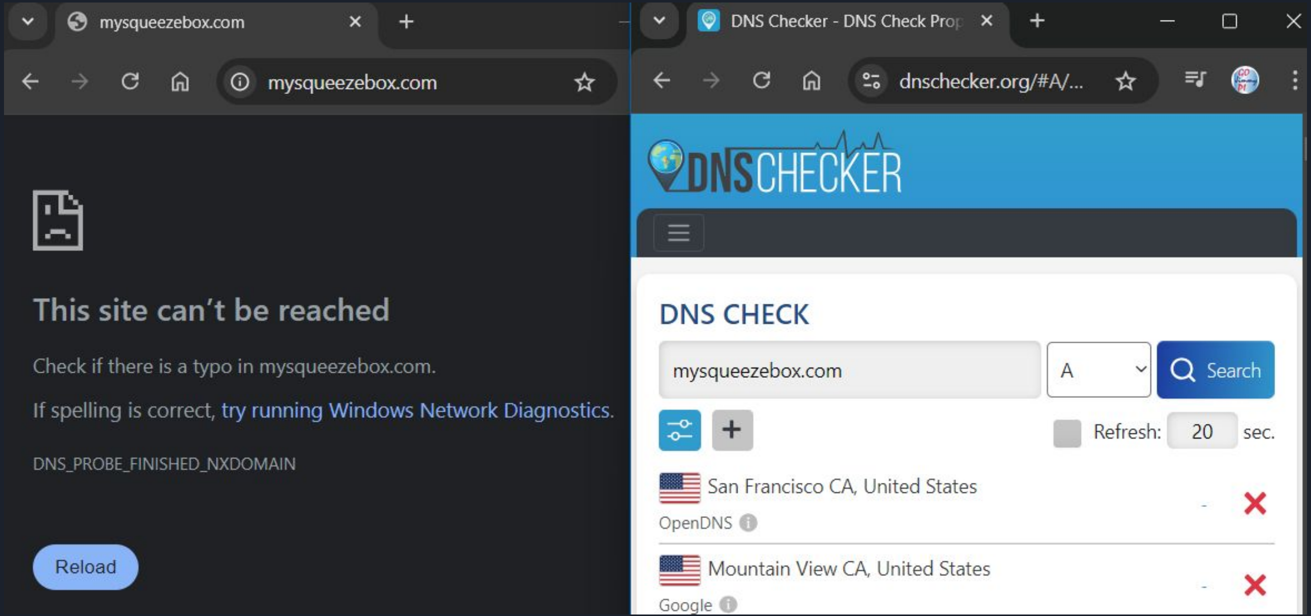
For years we have been using three **Logitech Squeezebox** Boom radios connected to the internet (WiFi) via the MySqueezebox.com server.

People also ask

Does Logitech Squeezebox still work?

Shutdown. In January 2024 Squeezebox users received a message that the Squeezebox servers would shut down in February 2024, making the Squeezebox devices unusable unless you set up your own Squeezebox server. Squeezebox offered a download for such a server at mysqueezebox.com for anyone wanting to run their own server.

Ya, but no.



mysqueezebox.com

This site can't be reached

Check if there is a typo in mysqueezebox.com.

If spelling is correct, try running Windows Network Diagnostics.

DNS_PROBE_FINISHED_NXDOMAIN

DNS CHECK

mysqueezebox.com | A

Refresh: 20 sec.

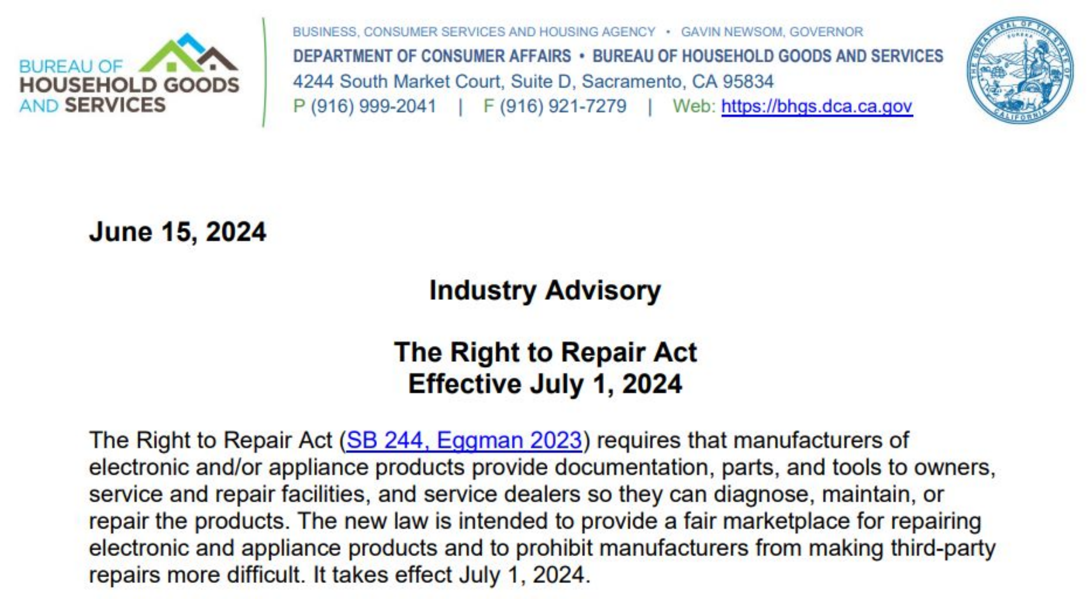
San Francisco CA, United States OpenDNS	-	✗
Mountain View CA, United States Google	-	✗

Oh, the irony

GO Jimmy PI

The list goes on...

Government Regulation?



BUREAU OF HOUSEHOLD GOODS AND SERVICES

BUSINESS, CONSUMER SERVICES AND HOUSING AGENCY • GAVIN NEWSOM, GOVERNOR
DEPARTMENT OF CONSUMER AFFAIRS • BUREAU OF HOUSEHOLD GOODS AND SERVICES
4244 South Market Court, Suite D, Sacramento, CA 95834
P (916) 999-2041 | F (916) 921-7279 | Web: https://bhgs.dca.ca.gov

June 15, 2024

Industry Advisory

The Right to Repair Act
Effective July 1, 2024

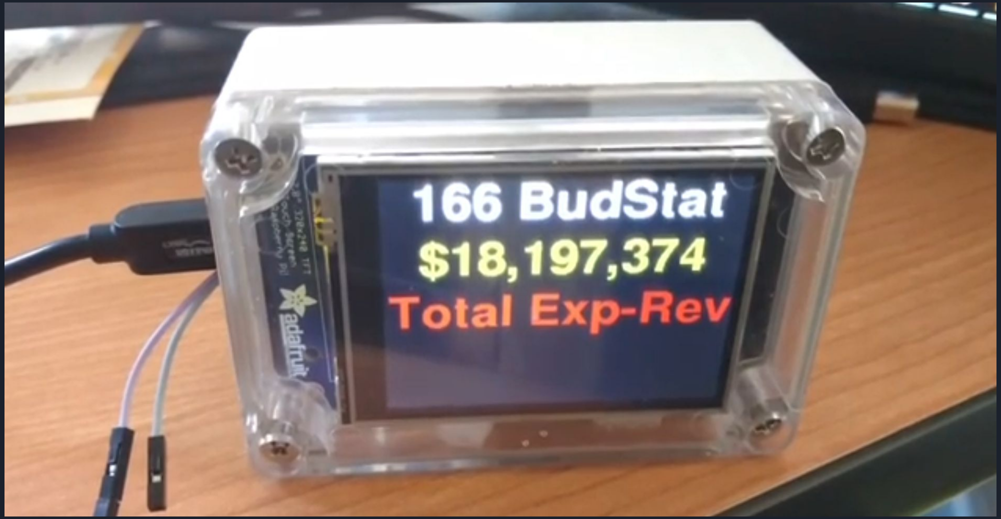
The Right to Repair Act (SB 244, Eggman 2023) requires that manufacturers of electronic and/or appliance products provide documentation, parts, and tools to owners, service and repair facilities, and service dealers so they can diagnose, maintain, or repair the products. The new law is intended to provide a fair marketplace for repairing electronic and appliance products and to prohibit manufacturers from making third-party repairs more difficult. It takes effect July 1, 2024.

https://bhgs.dca.ca.gov/forms_pubs/sb_244_industry_advisory_english.pdf

Repurposing: add or use existing

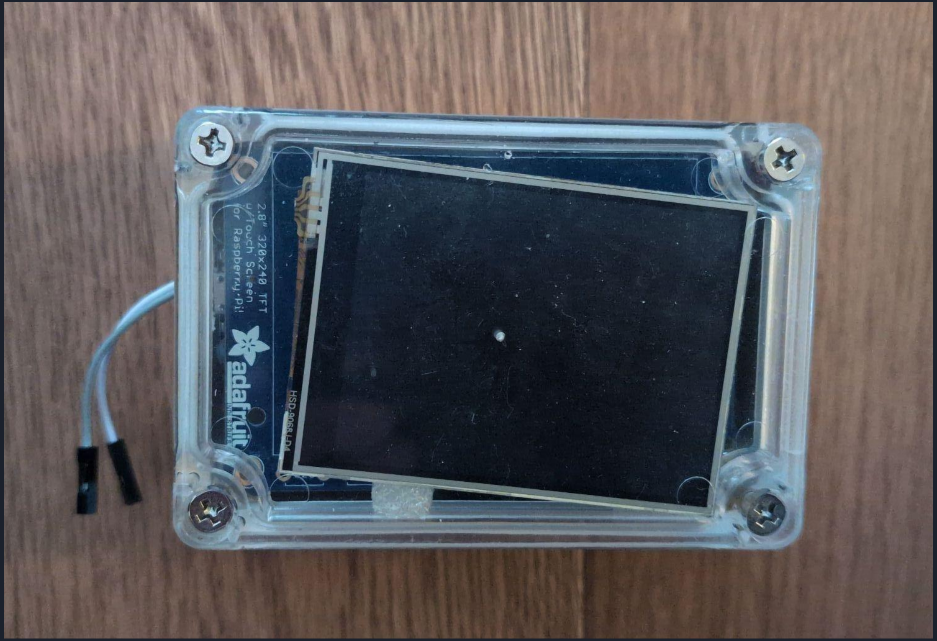
ENCLOSURE

Off-the-shelf enclosures!

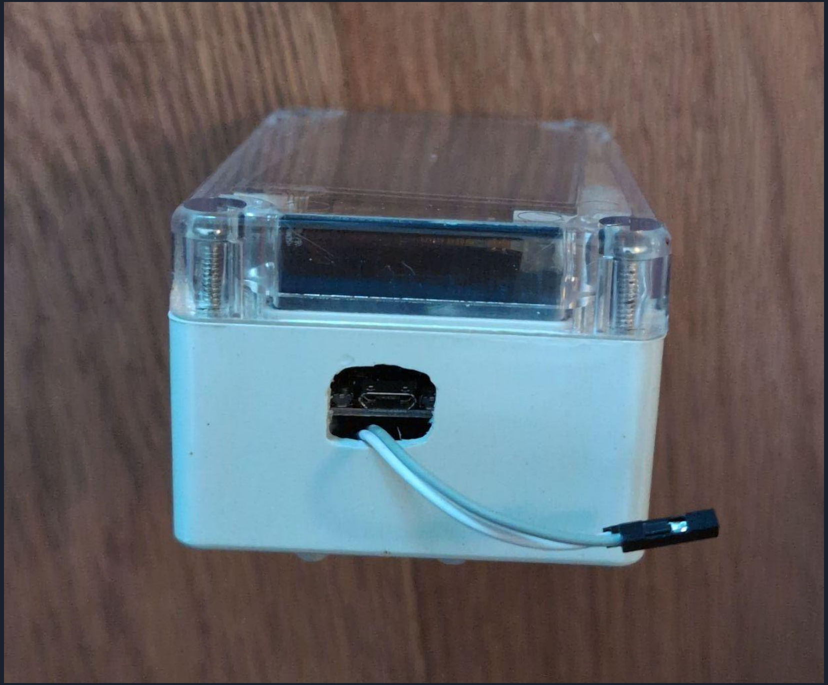


youtube.com/@gojimmypi/videos youtu.be/TmvaU6EQsAc

My enclosure is less than ideal



Machining can be… challenging



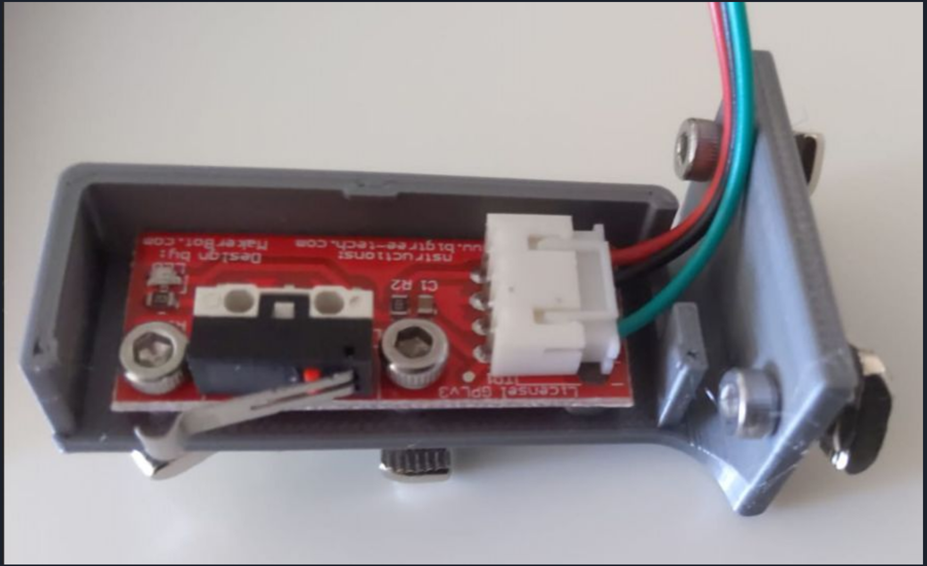
Repurposing

Reuse an awesome enclosure!

(and any built-in relays, sensors, power supply, display, etc.)

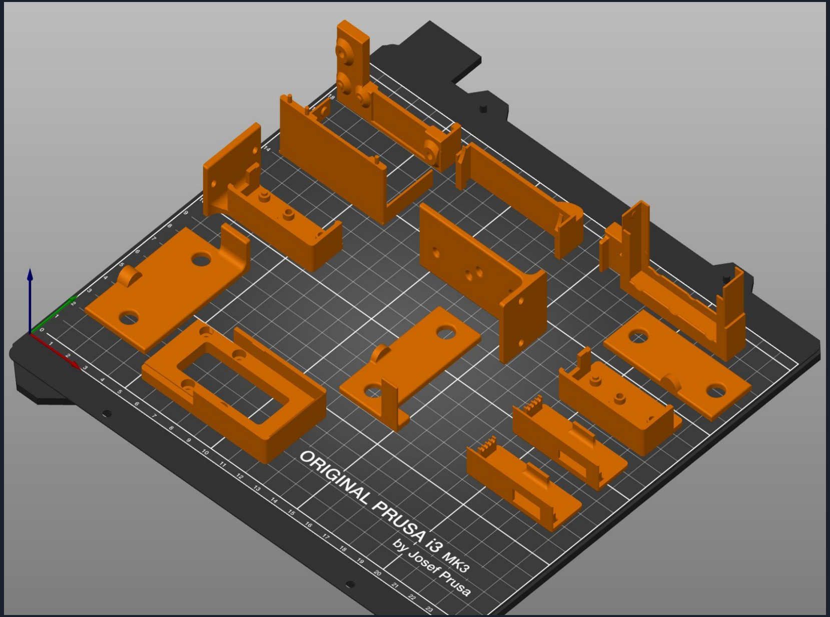
Custom enclosure for CNC Limit Switch

“*repurposing*” includes “**adding features**”!



github.com/gojimmypi/CNC-Endstop/

Arbitrary complexity, at home



gojimmypi.github.io/cnc-3018-makerbot-limit-switch-wiring/

How to get started

- Get an ESP32 dev board to practice
- Research product specifications
- Experiment, create prototype
- Have a security plan from BEGINNING

A few words on SAFETY 💥

IMPORTANT: SAFETY

Working with electricity in the home: line power or “mains” voltages are dangerous and

CAN KILL YOU or START A FIRE or *WORSE*.

(this is not fun nor funny)

If you are inexperienced with working directly with this type of electrical equipment,

DON’T DO IT.

Find a professional that can help; there are also plenty of battery-operated devices.

Proceed AT YOUR OWN RISK.

(I am not responsible for modifications that you make, and the resulting consequences)

Safety tips

- Never work on “live” equipment. Unplug from outlet
- Reach with one hand
- Use insulated tools
- Safety goggles; it’s not only magic smoke that might escape!
- Rubber-soled shoes

Beyond Safety: Expected GPIO Power Up State

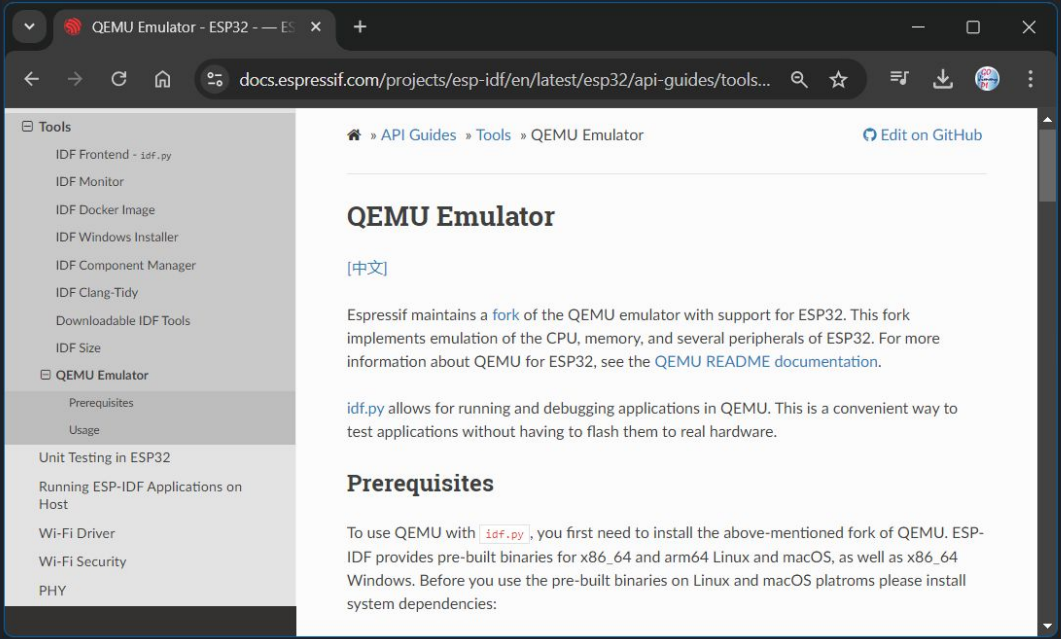
Original software may have set certain GPIO pins at startup time

Alternate software may have unintended and undesired actions

ESP32 has both 5v and 3.3v

- Ensure the GND pin is connected
- Don’t power the ESP32 from the Serial Adapter, aka TTY USB
- Serial is 3.3v
- Caution with stray wires & loose components

Get started without an ESP32 using QEMU



QEMU Emulator

[中文]

Espressif maintains a fork of the QEMU emulator with support for ESP32. This fork implements emulation of the CPU, memory, and several peripherals of ESP32. For more information about QEMU for ESP32, see the QEMU README documentation.

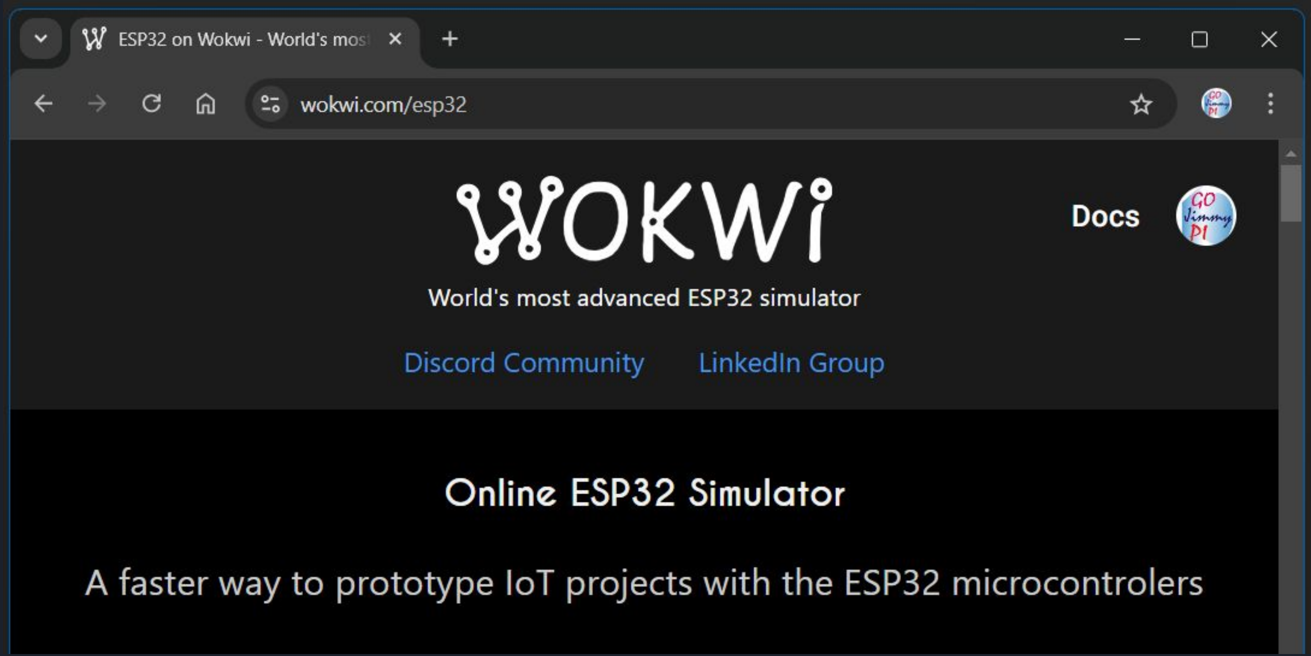
idf.py allows for running and debugging applications in QEMU. This is a convenient way to test applications without having to flash them to real hardware.

Prerequisites

To use QEMU with `idf.py`, you first need to install the above-mentioned fork of QEMU. ESP-IDF provides pre-built binaries for x86_64 and arm64 Linux and macOS, as well as x86_64 Windows. Before you use the pre-built binaries on Linux and macOS platroms please install system dependencies:

Screen snip from: docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/tools/qemu.html

Get started without an ESP32 - use Wokwi



World's most advanced ESP32 simulator

Discord Community LinkedIn Group

Online ESP32 Simulator

A faster way to prototype IoT projects with the ESP32 microcontrolers

www.wokwi.com

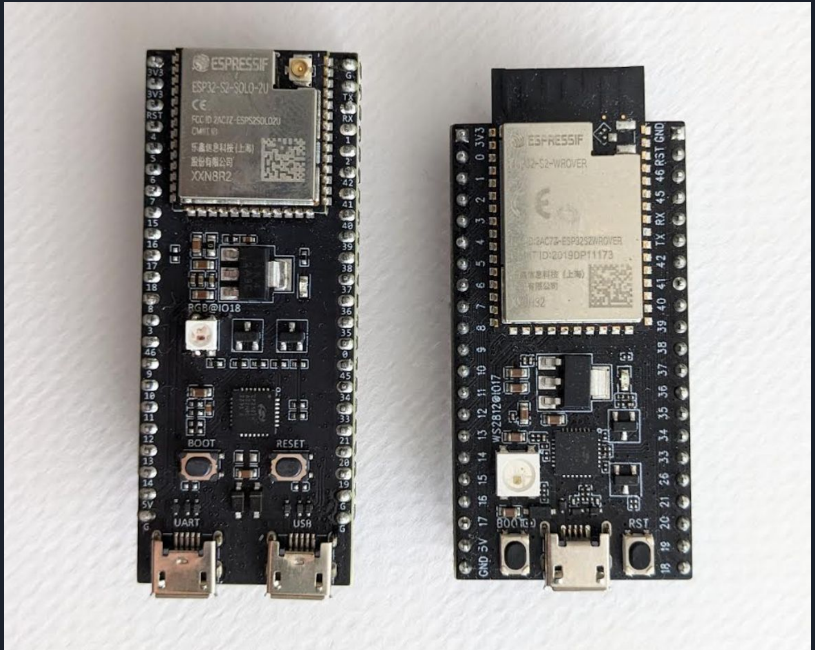
Sample Dev Board

External IPEX/U.FL Antenna

- Better range and signal strength
- Requires more space, extra cost

PCB Antenna

- Compact, low-cost
- Limited range



ulx3s.github.io

Image credit: Radiona. See also: crowdsupply.com/radiona/ulx3s

ESP32 is on the other side!

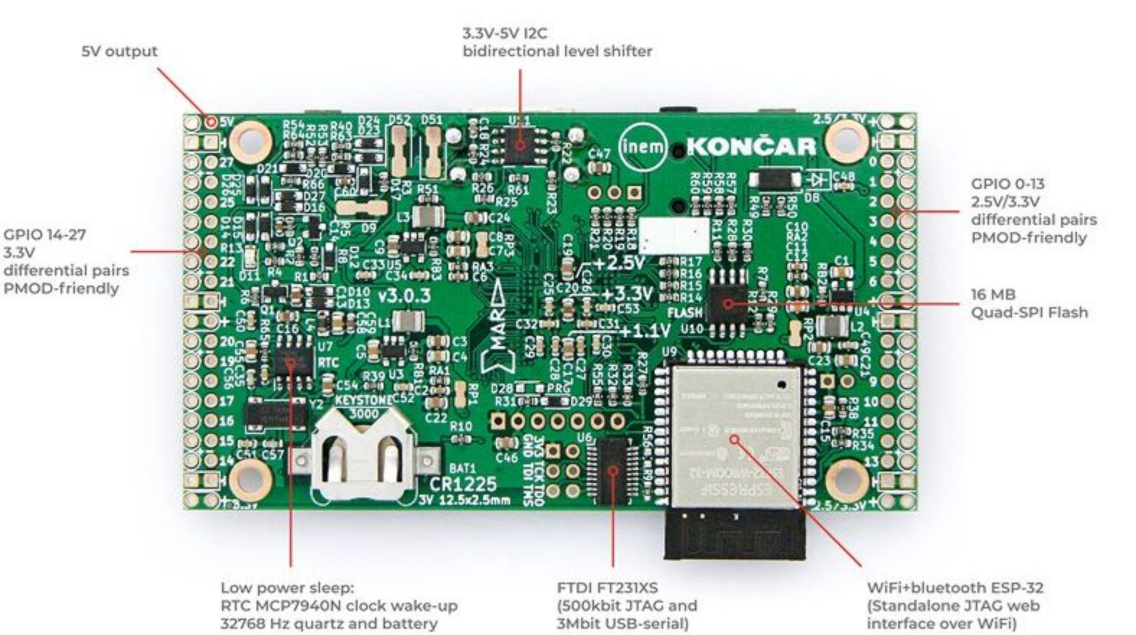
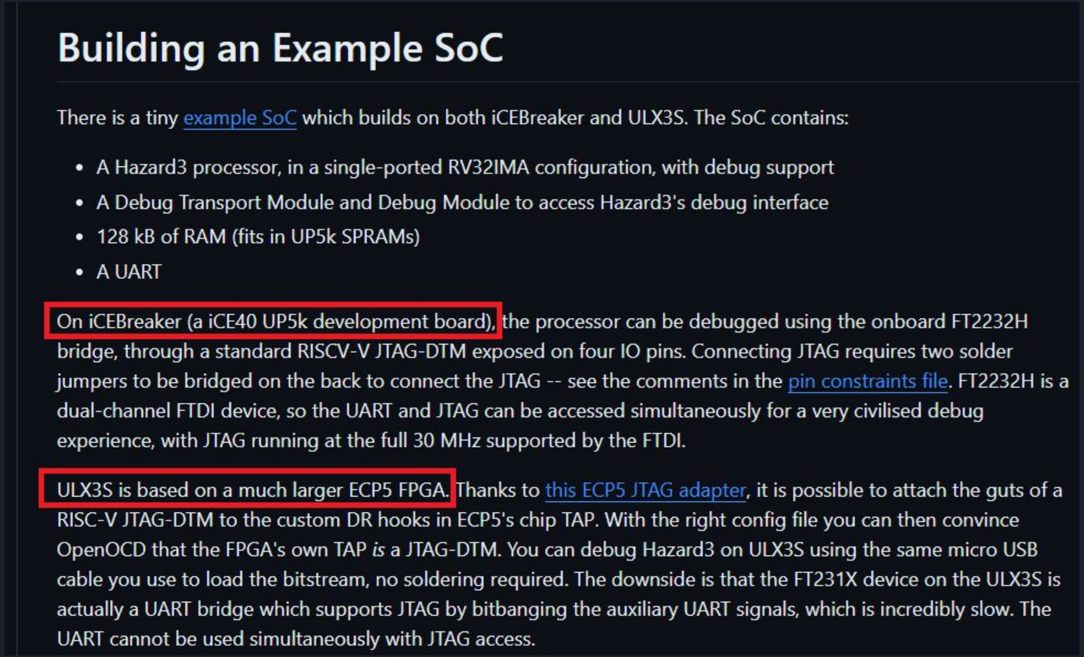


Image credit: Radiona

Side note on ULX3S: Hazard3 soft RISC-V



Building an Example SoC

There is a tiny example SoC which builds on both iCEBreaker and ULX3S. The SoC contains:

- A Hazard3 processor, in a single-ported RV32IMA configuration, with debug support
- A Debug Transport Module and Debug Module to access Hazard3's debug interface
- 128 kB of RAM (fits in UP5k SPRAMs)
- A UART

On iCEBreaker (a iCE40 UP5k development board), the processor can be debugged using the onboard FT2232H bridge, through a standard RISCV-V JTAG-DTM exposed on four IO pins. Connecting JTAG requires two solder jumpers to be bridged on the back to connect the JTAG -- see the comments in the pin constraints file. FT2232H is a dual-channel FTDI device, so the UART and JTAG can be accessed simultaneously for a very civilised debug experience, with JTAG running at the full 30 MHz supported by the FTDI.

ULX3S is based on a much larger ECP5 FPGA. Thanks to this ECP5 JTAG adapter, it is possible to attach the guts of a RISC-V JTAG-DTM to the custom DR hooks in ECP5's chip TAP. With the right config file you can then convince OpenOCD that the FPGA's own TAP *is* a JTAG-DTM. You can debug Hazard3 on ULX3S using the same micro USB cable you use to load the bitstream, no soldering required. The downside is that the FT231X device on the ULX3S is actually a UART bridge which supports JTAG by bitbanging the auxiliary UART signals, which is incredibly slow. The UART cannot be used simultaneously with JTAG access.

Image: screen snip from github.com/Wren6991/Hazard3/

Flashing: put program code on the device

- USB Port
- JTAG

□□ DANGER □□

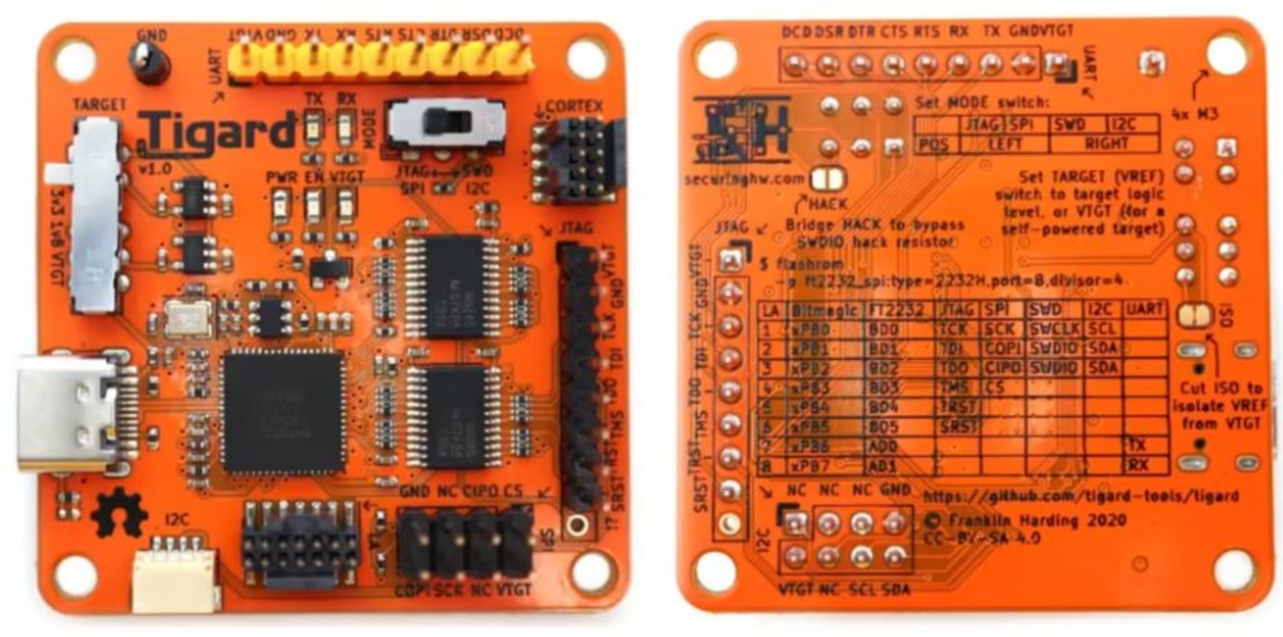
NEVER

PROGRAM A DEVICE

WHILE CONNECTED TO

AC “MAINS” LINE POWER

Tigard


GND
UART
DCD DSR DTR CTS RTS RX TX GND VTGT
TARGET
Tigard
v1.0
TX RX
MODE
PWR EN VTGT
JTAG SWD
SPI I2C
CORTEX
JTAG
SRST TRST TMS TDO TDI TCK GND VTGT
I2C
GND NC CIPO CS
COPI SCK NC VTGT
Set MODE switch:
securinghw.com
HACK
Bridge HACK to bypass SWDIO hack resistor
Set TARGET (VREF) switch to target logic level, or VTGT (for a self-powered target)
4x M3
ISO
Cut ISO to isolate VREF from VTGT
NC NC NC GND
VTGT NC SCL SDA
https://github.com/tigard-tools/tigard
© Franklin Harding 2020
CC-BY-SA 4.0

	JTAG	SPI	SWD	I2C
POS	LEFT		RIGHT	

```
$ flashrom
  -p ft2232_spi:type=2232H,port=B,divisor=4
```

LA	Bitmagic	FT2232	JTAG	SPI	SWD	I2C	UART
1	xPB0	BD0	TCK	SCK	SWCLK	SCL	
2	xPB1	BD1	TDI	COPI	SWDIO	SDA	
3	xPB2	BD2	TDO	CIPO	SWDIO	SDA	
4	xPB3	BD3	TMS	CS			
5	xPB4	BD4	TRST				
6	xPB5	BD5	SRST				
7	xPB6	AD0					TX
8	xPB7	AD1					RX

https://github.com/tigard-tools/tigard

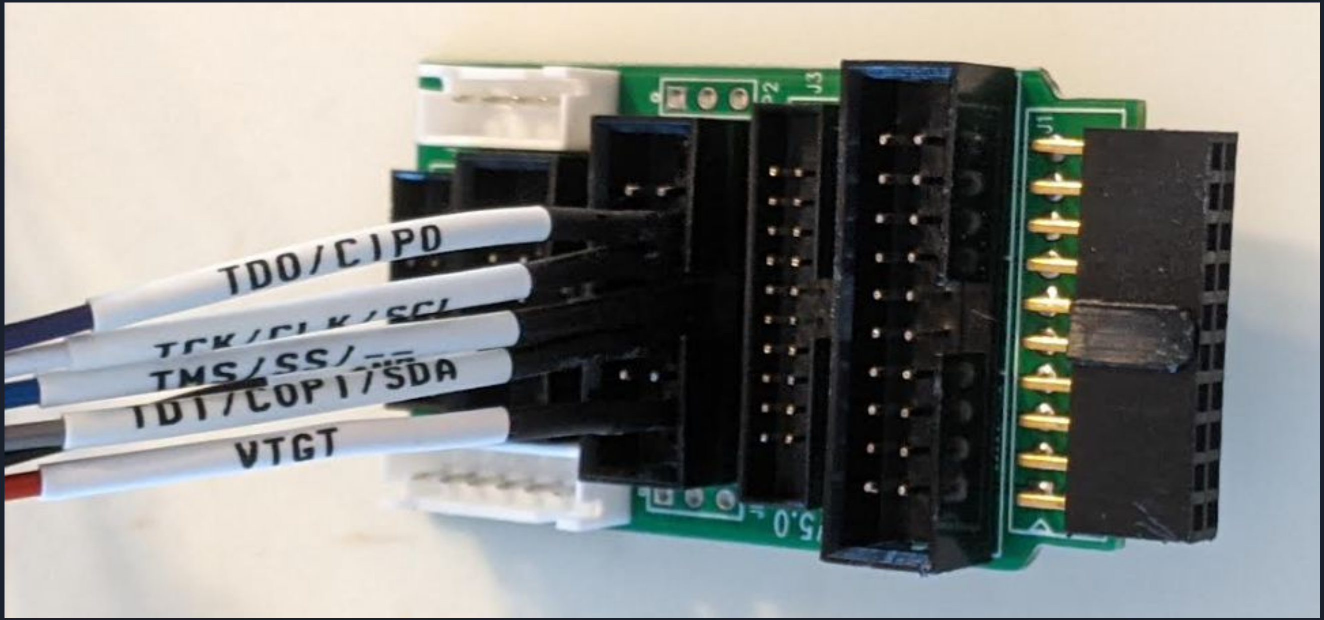
Image credit: https://www.crowdsupply.com/securinghw/tigard

JTAG Pins on ESP32

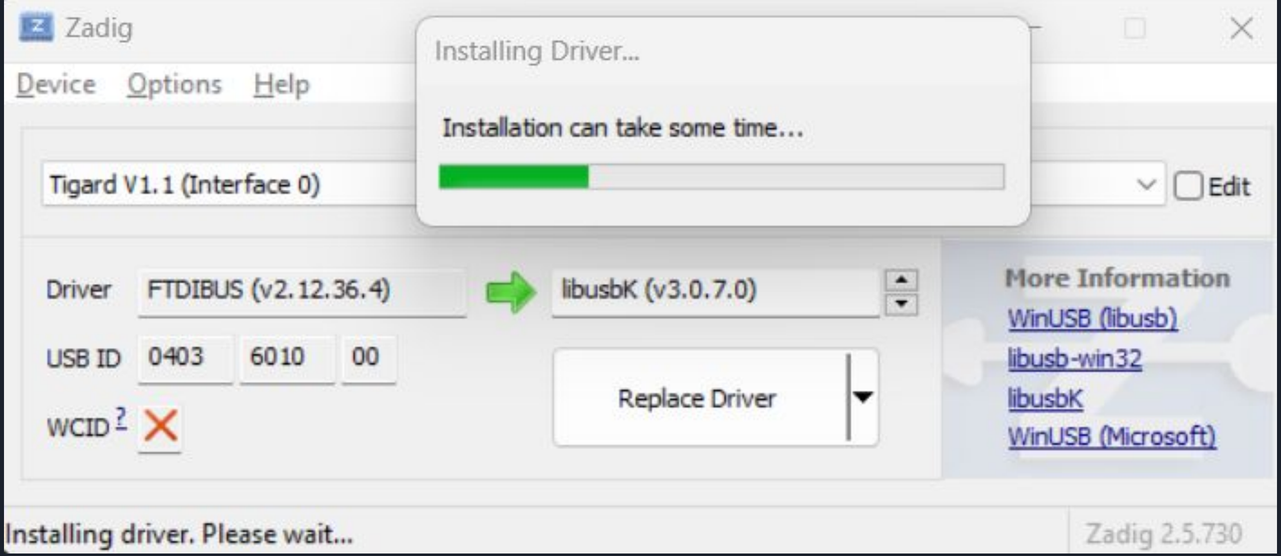
```
TDI -> GPIO12

TCK -> GPIO13

TMS -> GPIO14

TDO -> GPIO15

TRST -> EN / RST (Reset)

GND -> GND
```

See also: gojimmypi.github.io/Tigard-JTAG-SingleStep-Debugging-ESP32/

Tigard has JTAG labels



Default Tigard Drivers: needs to be libusbK



Installing Driver...

Installation can take some time...

Tigard V1.1 (Interface 0)

Driver FTDIBUS (v2.12.36.4) → libusbK (v3.0.7.0)

USB ID 0403 6010 00

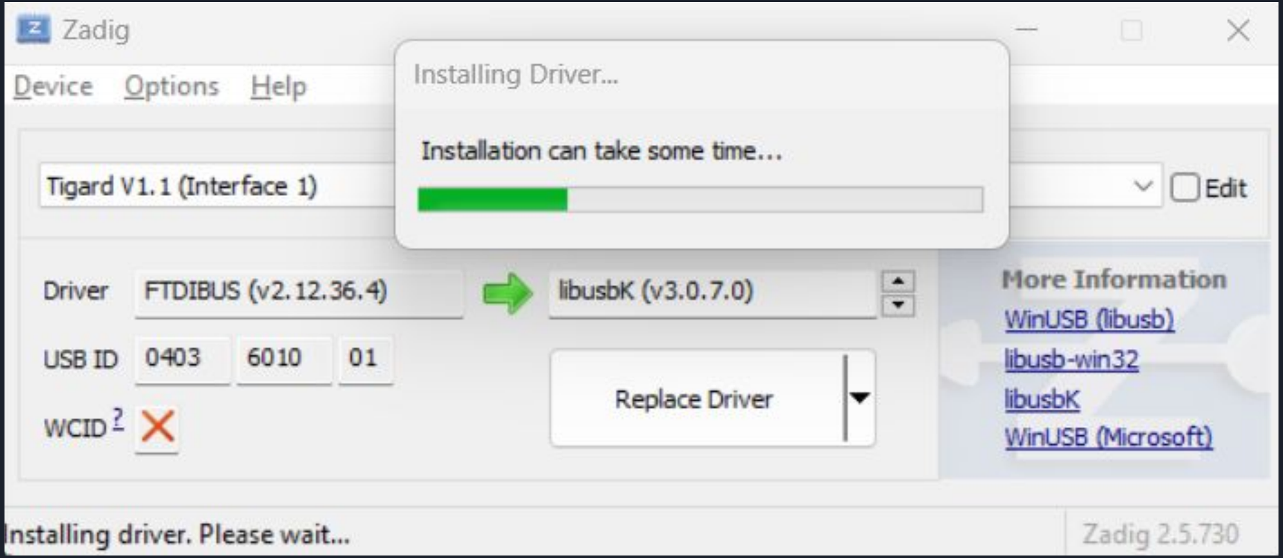
WCID ? ☒

Installing driver. Please wait...

Zadig 2.5.730

Download Zadig at: zadig.akeo.ie

There are TWO interfaces



Installation can take some time...

Tigard V1.1 (Interface 1)

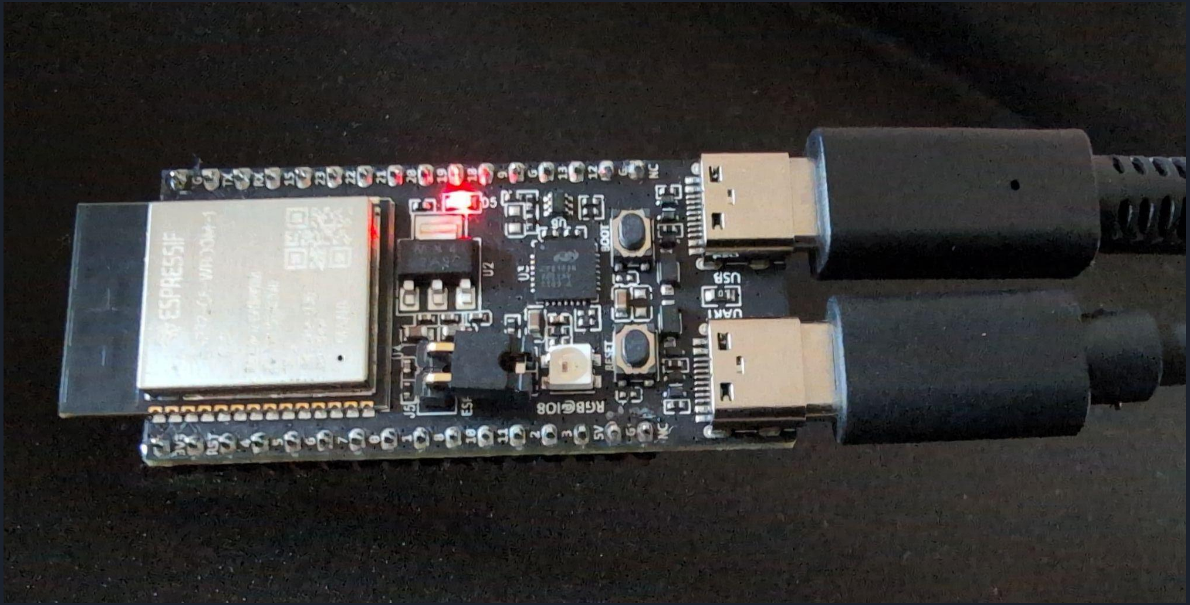
Driver FTDIBUS (v2.12.36.4) → libusbK (v3.0.7.0)

USB ID 0403 6010 01

WCID ? ✗

Installing driver. Please wait...

Some Espressif boards have JTAG included



See also: gojimmypi.github.io/FTDI2232HL-Dual-Interface-fix/

JTAG on Windows - Troubleshooting

- Unplug & replug USB after updating drivers to libusbK
- Try lower speeds
- Ensure quality USB cable used; not charging “Power only”
- Drivers revert after Windows update
- See blog:

  gojimmypi.github.io/Tigard-JTAG-SingleStep-Debugging-ESP32/

Software Tools

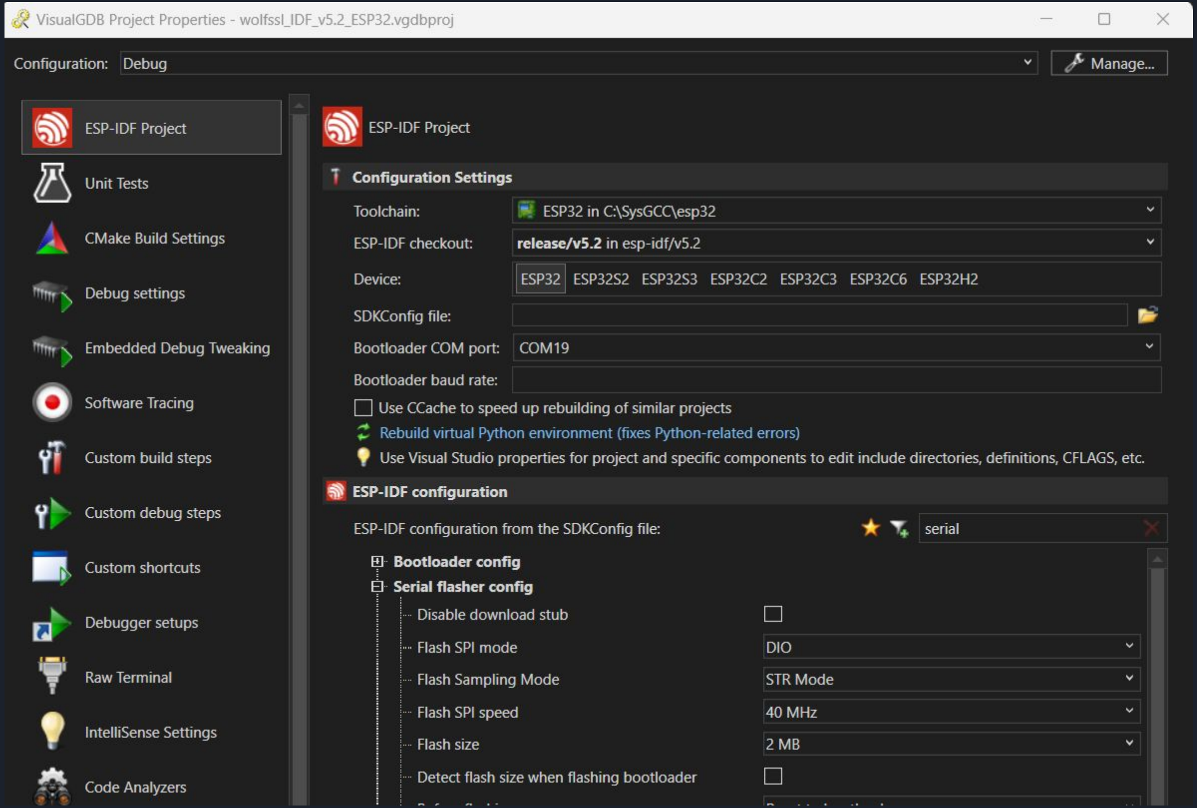
Espressif ESP-IDF	idf.espressif.com
PlatformIO	docs.platformio.org/en/latest/platforms/espressif32.html
Arduino	docs.arduino.cc/hardware/nano-esp32
Espressif Arduino	docs.espressif.com/projects/arduino-esp32/en/latest/installing.html
VisualGDB	visualgdb.com

Many different IDE packages can be used

IDE = Integrated Development Environment

- Espressif IDE (ESP-IDF Eclipse Plugin)
- VS Code
- Arduino IDE
- Visual Micro (Arduino IDE for Visual Studio)
- VIM / VI / Nano / etc.

VisualGDB Extension for Visual Studio



VisualGDB Project Properties - wolfssl_IDF_v5.2_ESP32.vgdbproj

Configuration: Debug

ESP-IDF Project

Configuration Settings

Toolchain: ESP32 in C:\SysGCC\esp32

ESP-IDF checkout: **release/v5.2** in esp-idf/v5.2

Device: ESP32 ESP32S2 ESP32S3 ESP32C2 ESP32C3 ESP32C6 ESP32H2

SDKConfig file:

Bootloader COM port: COM19

Bootloader baud rate:

□ Use CCache to speed up rebuilding of similar projects

Rebuild virtual Python environment (fixes Python-related errors)

Use Visual Studio properties for project and specific components to edit include directories, definitions, CFLAGS, etc.

ESP-IDF configuration

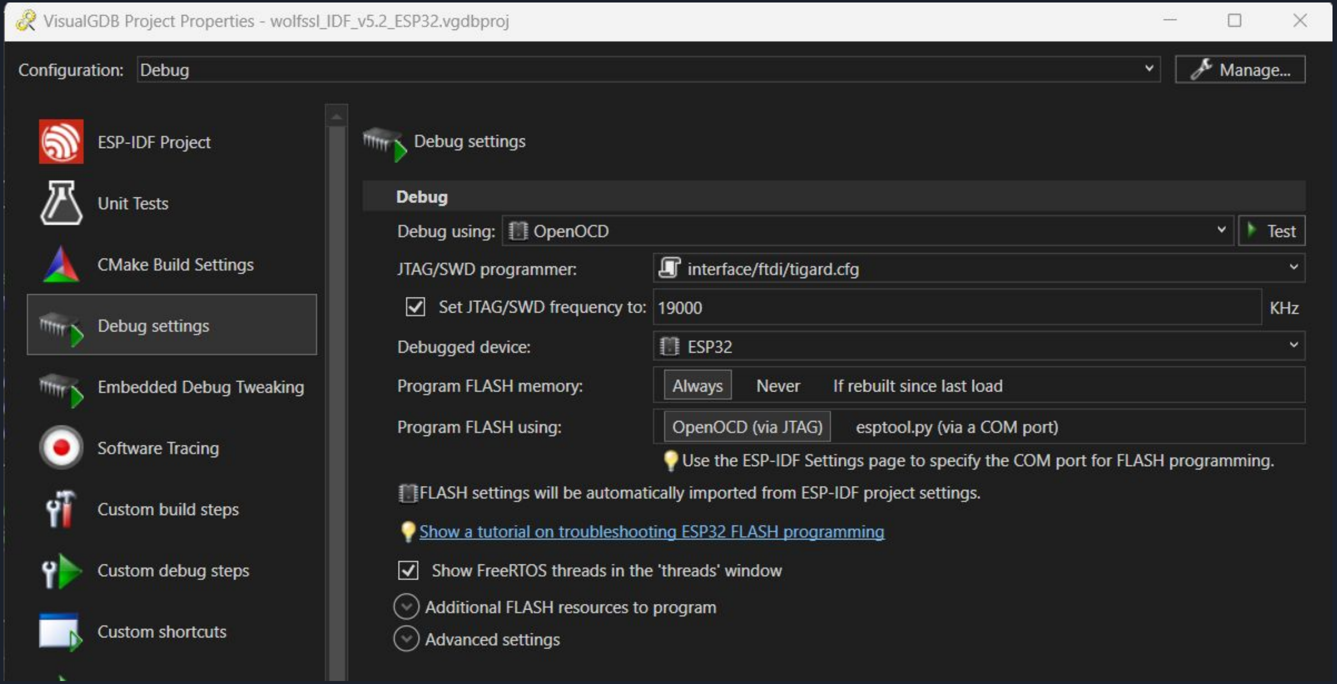
ESP-IDF configuration from the SDKConfig file: serial

⊞ **Bootloader config**

⊟ **Serial flasher config**

Disable download stub	□
Flash SPI mode	DIO
Flash Sampling Mode	STR Mode
Flash SPI speed	40 MHz
Flash size	2 MB
Detect flash size when flashing bootloader	□

JTAG Debug Settings: Tigard



VisualGDB Project Properties - wolfssl_IDF_v5.2_ESP32.vgdbproj

Configuration: Debug

Debug settings

Debug

Debug using: OpenOCD

JTAG/SWD programmer: interface/ftdi/tigard.cfg

☑ Set JTAG/SWD frequency to: 19000 KHz

Debugged device: ESP32

Program FLASH memory: [✓ Always] Never If rebuilt since last load

Program FLASH using: [✓ OpenOCD (via JTAG)] esptool.py (via a COM port)

Use the ESP-IDF Settings page to specify the COM port for FLASH programming.

FLASH settings will be automatically imported from ESP-IDF project settings.

Show a tutorial on troubleshooting ESP32 FLASH programming

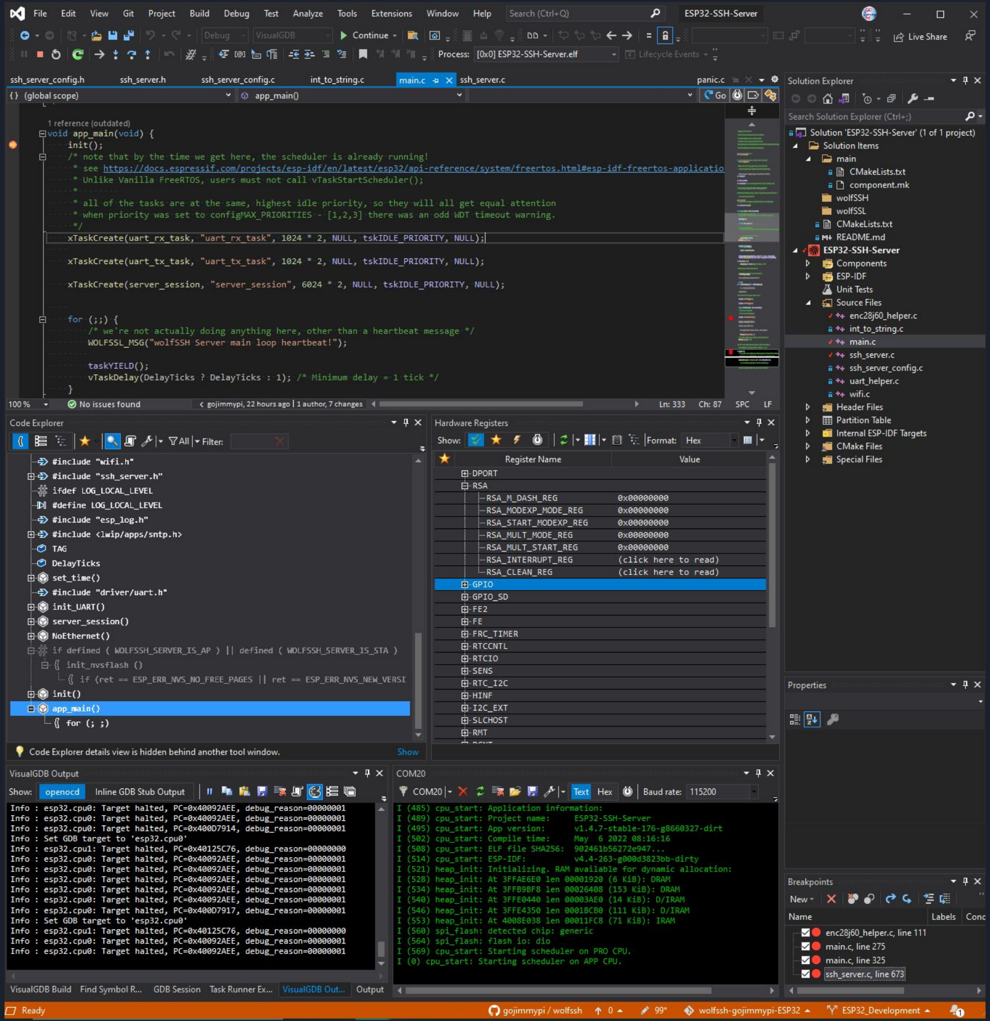
☑ Show FreeRTOS threads in the 'threads' window

Additional FLASH resources to program

Advanced settings

VisualGDB

- Breakpoints
- Single-step
- Variable inspection
- Register inspection
- Code explorer
- Call Stack
- Output window
- Project files
- GitHub integration
- Intellisense
- Much more….



```c
void app_main(void) {
    init();
    /* note that by the time we get here, the scheduler is already running!
     * see https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/system/freertos.html#esp-idf-freertos-applicatio
     * Unlike Vanilla FreeRTOS, users must not call vTaskStartScheduler();
     *
     * all of the tasks are at the same, highest idle priority, so they will all get equal attention
     * when priority was set to configMAX_PRIORITIES - [1,2,3] there was an odd WDT timeout warning.
     */
    xTaskCreate(uart_rx_task, "uart_rx_task", 1024 * 2, NULL, tskIDLE_PRIORITY, NULL);

    xTaskCreate(uart_tx_task, "uart_tx_task", 1024 * 2, NULL, tskIDLE_PRIORITY, NULL);

    xTaskCreate(server_session, "server_session", 6024 * 2, NULL, tskIDLE_PRIORITY, NULL);


    for (;;) {
        /* we're not actually doing anything here, other than a heartbeat message */
        WOLFSSL_MSG("wolfSSH Server main loop heartbeat!");

        taskYIELD();
        vTaskDelay(DelayTicks ? DelayTicks : 1); /* Minimum delay = 1 tick */
    }
```

```
#include "wifi.h"
#include "ssh_server.h"
ifdef LOG_LOCAL_LEVEL
#define LOG_LOCAL_LEVEL
#include "esp_log.h"
#include <lwip/apps/sntp.h>
TAG
DelayTicks
set_time()
#include "driver/uart.h"
init_UART()
server_session()
NoEthernet()
if defined ( WOLFSSH_SERVER_IS_AP ) || defined ( WOLFSSH_SERVER_IS_STA )
  init_nvsflash ()
    if (ret == ESP_ERR_NVS_NO_FREE_PAGES || ret == ESP_ERR_NVS_NEW_VERSI
init()
app_main()
  for (; ;)
```

Code Explorer details view is hidden behind another tool window. Show

Register Name	Value
DPORT	
RSA	
RSA_M_DASH_REG	0x00000000
RSA_MODEXP_MODE_REG	0x00000000
RSA_START_MODEXP_REG	0x00000000
RSA_MULT_MODE_REG	0x00000000
RSA_MULT_START_REG	0x00000000
RSA_INTERRUPT_REG	(click here to read)
RSA_CLEAN_REG	(click here to read)
GPIO	
GPIO_SD	
FE2	
FE	
FRC_TIMER	
RTCCNTL	
RTCIO	
SENS	
RTC_I2C	
HINF	
I2C_EXT	
SLCHOST	
RMT	

```
Info : esp32.cpu0: Target halted, PC=0x40092AEE, debug_reason=00000001
Info : esp32.cpu0: Target halted, PC=0x40092AEE, debug_reason=00000001
Info : esp32.cpu0: Target halted, PC=0x400D7914, debug_reason=00000001
Info : Set GDB target to 'esp32.cpu0'
Info : esp32.cpu1: Target halted, PC=0x40125C76, debug_reason=00000000
Info : esp32.cpu0: Target halted, PC=0x40092AEE, debug_reason=00000001
Info : esp32.cpu0: Target halted, PC=0x40092AEE, debug_reason=00000001
Info : esp32.cpu0: Target halted, PC=0x40092AEE, debug_reason=00000001
Info : esp32.cpu0: Target halted, PC=0x40092AEE, debug_reason=00000001
Info : esp32.cpu0: Target halted, PC=0x40092AEE, debug_reason=00000001
Info : esp32.cpu0: Target halted, PC=0x400D7917, debug_reason=00000001
Info : Set GDB target to 'esp32.cpu0'
Info : esp32.cpu1: Target halted, PC=0x40125C76, debug_reason=00000000
Info : esp32.cpu0: Target halted, PC=0x40092AEE, debug_reason=00000001
Info : esp32.cpu0: Target halted, PC=0x40092AEE, debug_reason=00000001
```

```
I (485) cpu_start: Application information:
I (489) cpu_start: Project name:     ESP32-SSH-Server
I (495) cpu_start: App version:      v1.4.7-stable-176-g8660327-dirt
I (502) cpu_start: Compile time:     May  6 2022 08:16:16
I (508) cpu_start: ELF file SHA256:  902461b56272e947...
I (514) cpu_start: ESP-IDF:          v4.4-263-g000d3823bb-dirty
I (521) heap_init: Initializing. RAM available for dynamic allocation:
I (528) heap_init: At 3FFAE6E0 len 00001920 (6 KiB): DRAM
I (534) heap_init: At 3FFB9BF8 len 00026408 (153 KiB): DRAM
I (540) heap_init: At 3FFE0440 len 00003AE0 (14 KiB): D/IRAM
I (546) heap_init: At 3FFE4350 len 0001BCB0 (111 KiB): D/IRAM
I (553) heap_init: At 4008E038 len 00011FC8 (71 KiB): IRAM
I (560) spi_flash: detected chip: generic
I (564) spi_flash: flash io: dio
I (569) cpu_start: Starting scheduler on PRO CPU.
I (0) cpu_start: Starting scheduler on APP CPU.
```

Name	Labels	Conc
☑ enc28j60_helper.c, line 111		
☑ main.c, line 275		
☑ main.c, line 325		
☑ ssh_server.c, line 673		

Device software

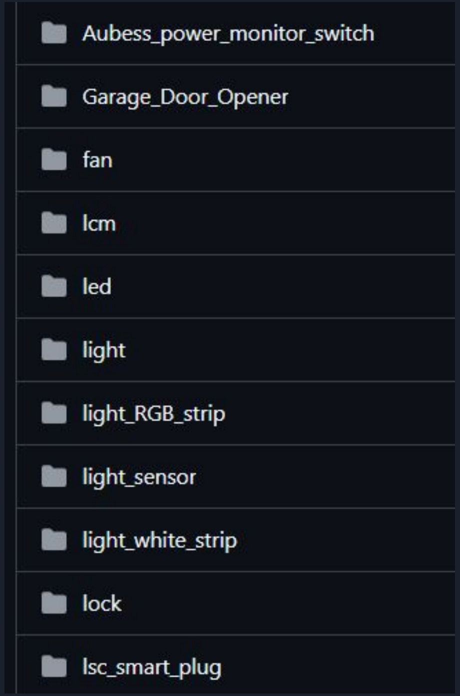
“File - New Project” ?

Device software

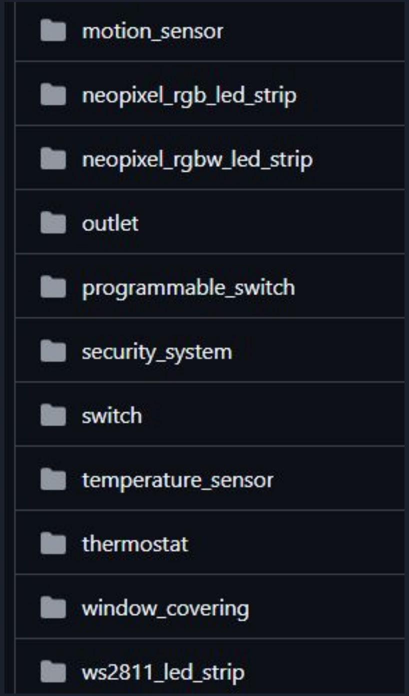
“File - New Project” ?

(probably not)

Apple HomeKit



- Aubess_power_monitor_switch
- Garage_Door_Opener
- fan
- lcm
- led
- light
- light_RGB_strip
- light_sensor
- light_white_strip
- lock
- lsc_smart_plug



- motion_sensor
- neopixel_rgb_led_strip
- neopixel_rgbw_led_strip
- outlet
- programmable_switch
- security_system
- switch
- temperature_sensor
- thermostat
- window_covering
- ws2811_led_strip

github.com/AchimPieters/esp32-homekit-demo/

How to secure your project communication

- Unlike the clock radio, many ESP32 projects will use network communication
- Old devices will have old, insecure cryptographic features
- Use a commercial-grade open source cryptographic library such as wolfSSL for TLS 1.3

Examples run on wolfSSL

Secure Your Apple HomeKit Espressif ESP32 Devices with wolfSSL

wolfssl.com/secure-your-apple-homekit-espressif-esp32-devices-with-wolfssl/

wolfSSL

Open Source
Internet Security

LIGHTWEIGHT. PORTABLE. C-BASED.

- Up to TLS 1.3 and DTLS 1.3
- 20-100 kB footprint
- 1-36 kB RAM per session
- Up to 20X Smaller than OpenSSL
- Long list of supported operating systems
- Certified FIPS 140-3, DO-178 Support, MISRA-C
- Best-tested crypto
- 24x7 Support
- Dual-licensed
- Secure boot, MQTT, SSH, TPM 2.0, JSSE, IDPS, commercial support for curl

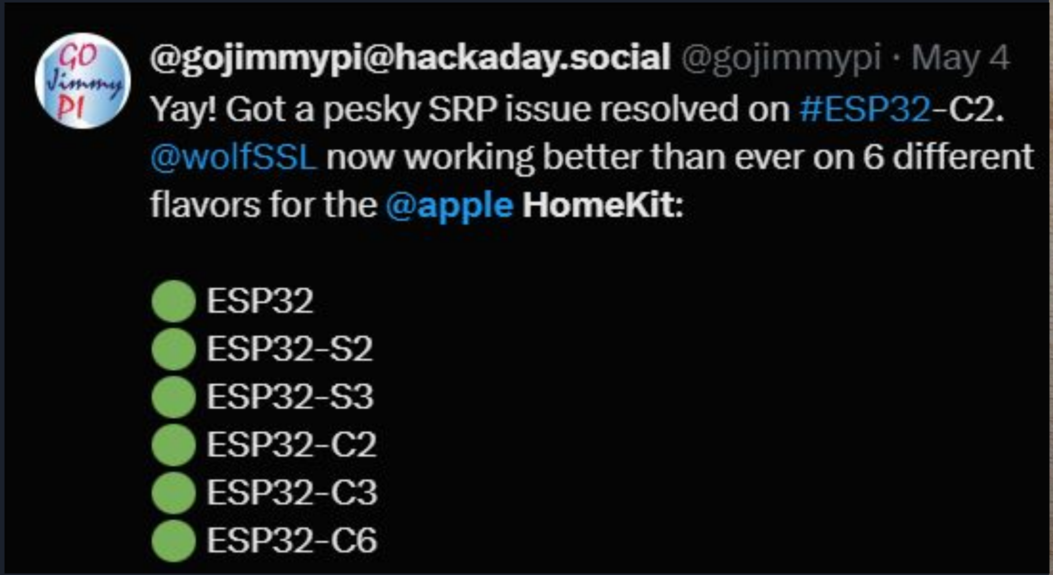
SYSGO EMBEDDING INNOVATIONS · RENESAS · Mentor Graphics · arm · Texas Instruments · Green Hills SOFTWARE · 0x5 HEX-Five · MICROCHIP · NXP · ST life.augmented · XILINX · intel · DDC-I

Certified, commercial grade, open source

- github.com/wolfSSL/wolfssl
- wolfssl.com/espressif
- Dual licensed, free for makers (GPLv2)
- Custom licensing terms also available

Any ESP32 can be part of Apple HomeKit

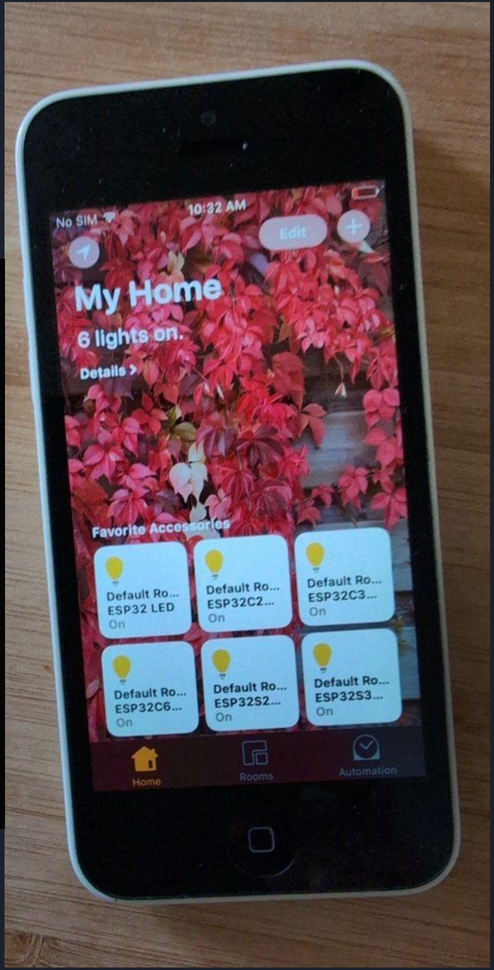
Tweet of Success 🎉



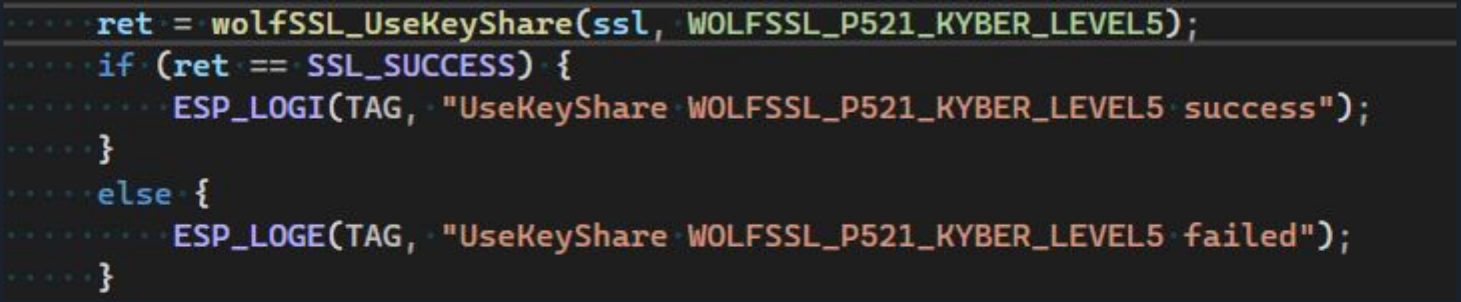
@gojimmypi@hackaday.social @gojimmypi · May 4

Yay! Got a pesky SRP issue resolved on #ESP32-C2. @wolfSSL now working better than ever on 6 different flavors for the @apple HomeKit:

🟢 ESP32
🟢 ESP32-S2
🟢 ESP32-S3
🟢 ESP32-C2
🟢 ESP32-C3
🟢 ESP32-C6


No SIM
10:32 AM
Edit
My Home
6 lights on.
Details >
Favorite Accessories
Default Ro... ESP32 LED On
Default Ro... ESP32C2... On
Default Ro... ESP32C3... On
Default Ro... ESP32C6... On
Default Ro... ESP32S2... On
Default Ro... ESP32S3... On
Home
Rooms
Automation

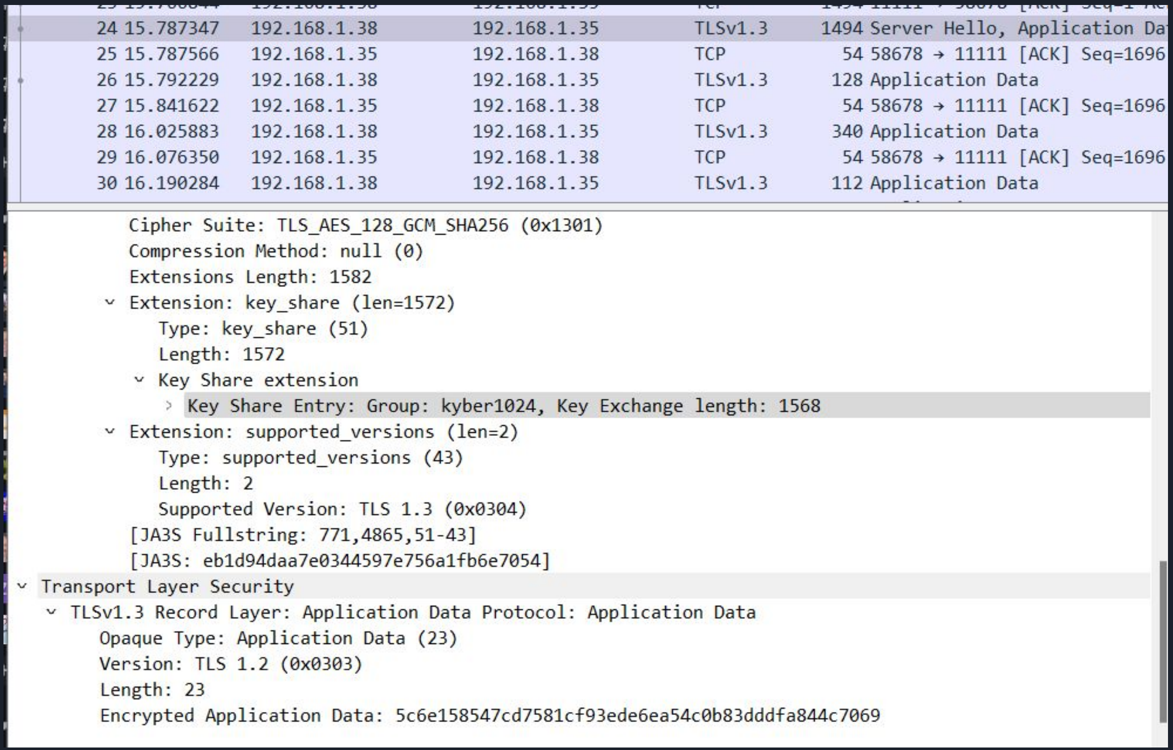
Post Quantum Cryptography



```
    ret = wolfSSL_UseKeyShare(ssl, WOLFSSL_P521_KYBER_LEVEL5);
    if (ret == SSL_SUCCESS) {
        ESP_LOGI(TAG, "UseKeyShare WOLFSSL_P521_KYBER_LEVEL5 success");
    }
    else {
        ESP_LOGE(TAG, "UseKeyShare WOLFSSL_P521_KYBER_LEVEL5 failed");
    }
```

www.wolfssl.com/post-quantum-key-share-on-the-espressif-esp32/

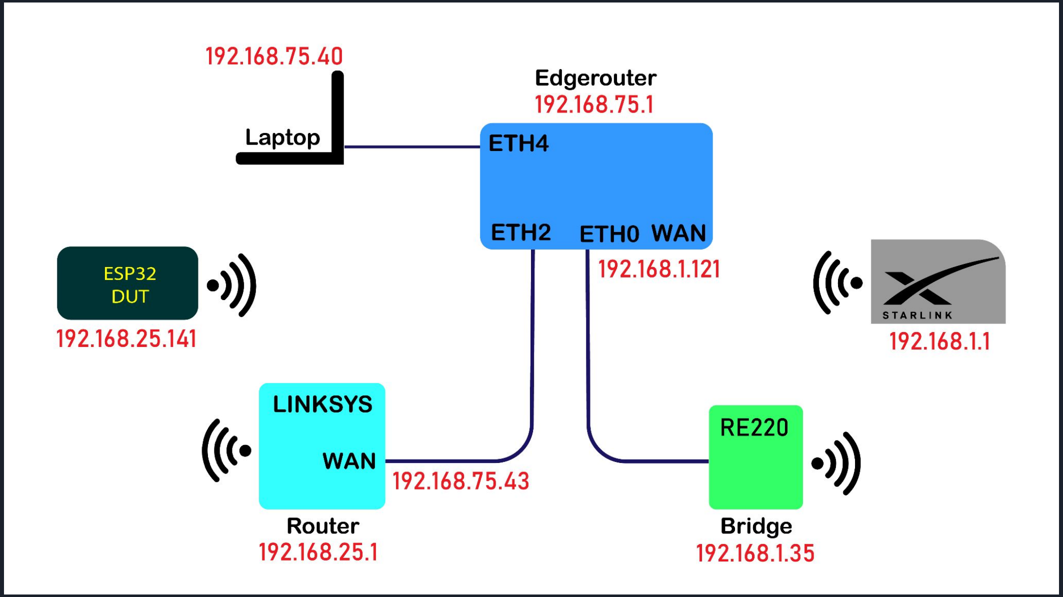
Wireshark to inspect network packets



```
24 15.787347   192.168.1.38     192.168.1.35     TLSv1.3    1494 Server Hello, Application Da
25 15.787566   192.168.1.35     192.168.1.38     TCP          54 58678 → 11111 [ACK] Seq=1696
26 15.792229   192.168.1.38     192.168.1.35     TLSv1.3     128 Application Data
27 15.841622   192.168.1.35     192.168.1.38     TCP          54 58678 → 11111 [ACK] Seq=1696
28 16.025883   192.168.1.38     192.168.1.35     TLSv1.3     340 Application Data
29 16.076350   192.168.1.35     192.168.1.38     TCP          54 58678 → 11111 [ACK] Seq=1696
30 16.190284   192.168.1.38     192.168.1.35     TLSv1.3     112 Application Data
```

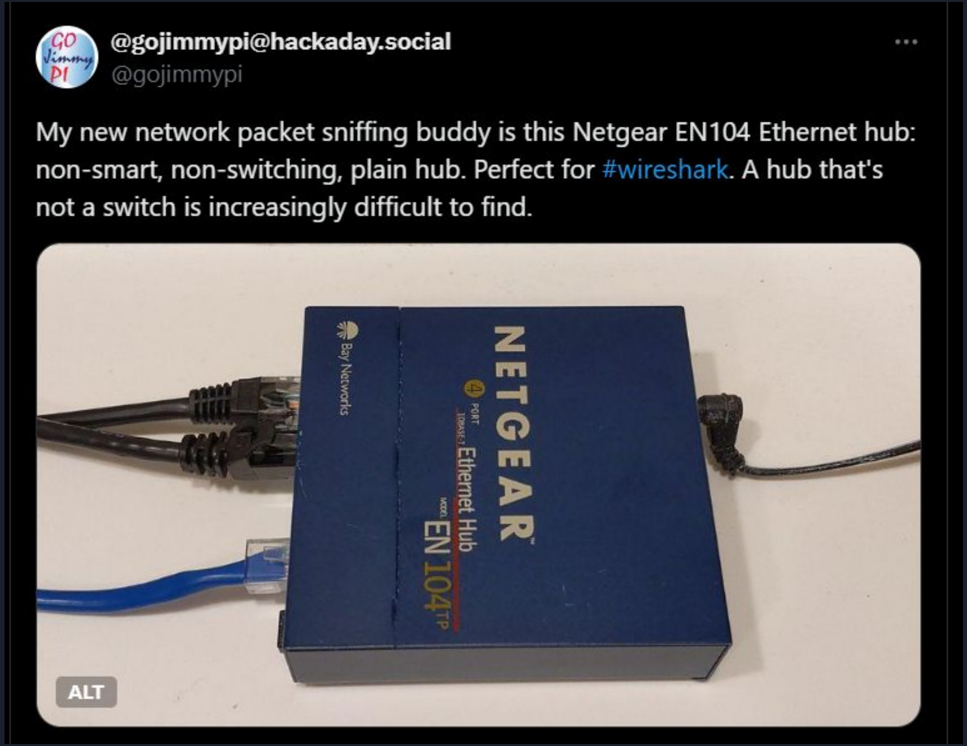
```
          Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
          Compression Method: null (0)
          Extensions Length: 1582
        ˅ Extension: key_share (len=1572)
            Type: key_share (51)
            Length: 1572
          ˅ Key Share extension
            › Key Share Entry: Group: kyber1024, Key Exchange length: 1568
        ˅ Extension: supported_versions (len=2)
            Type: supported_versions (43)
            Length: 2
            Supported Version: TLS 1.3 (0x0304)
          [JA3S Fullstring: 771,4865,51-43]
          [JA3S: eb1d94daa7e0344597e756a1fb6e7054]
˅ Transport Layer Security
  ˅ TLSv1.3 Record Layer: Application Data Protocol: Application Data
      Opaque Type: Application Data (23)
      Version: TLS 1.2 (0x0303)
      Length: 23
      Encrypted Application Data: 5c6e158547cd7581cf93ede6ea54c0b83dddfa844c7069
```

Monitoring ESP32 packets



gojimmypi.github.io/Espressif-ESP32-WiFi-Port-Sniffing-DUT/

Consider a plain hub (not a switch!)



@gojimmypi@hackaday.social
@gojimmypi

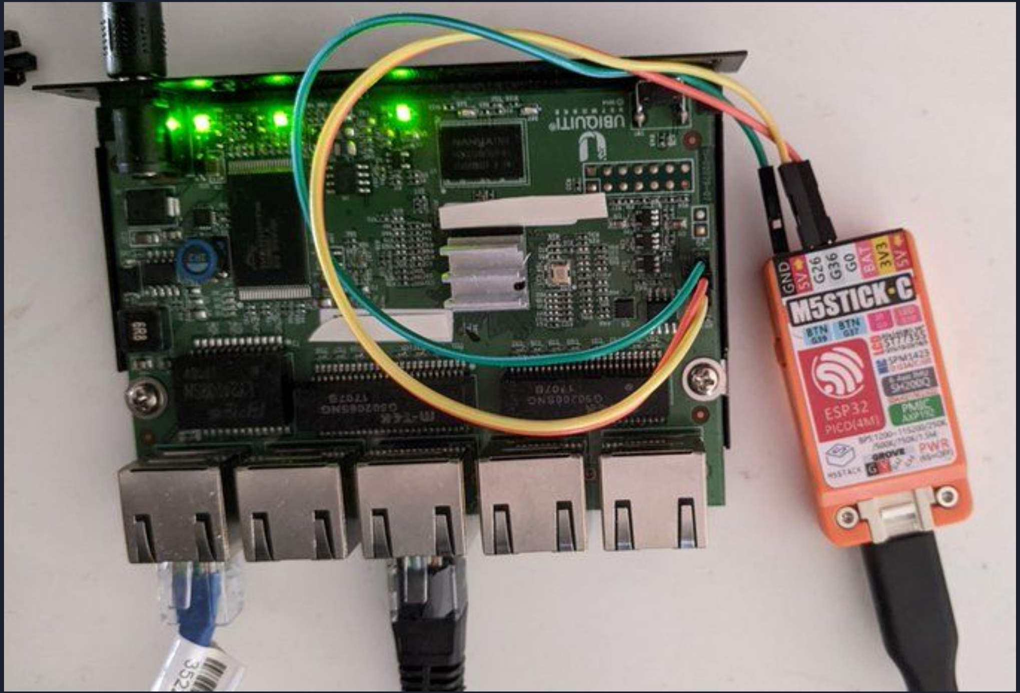
My new network packet sniffing buddy is this Netgear EN104 Ethernet hub: non-smart, non-switching, plain hub. Perfect for #wireshark. A hub that's not a switch is increasingly difficult to find.

x.com/gojimmypi/status/1693438341000380824

SSH on a Stick - Add functionality

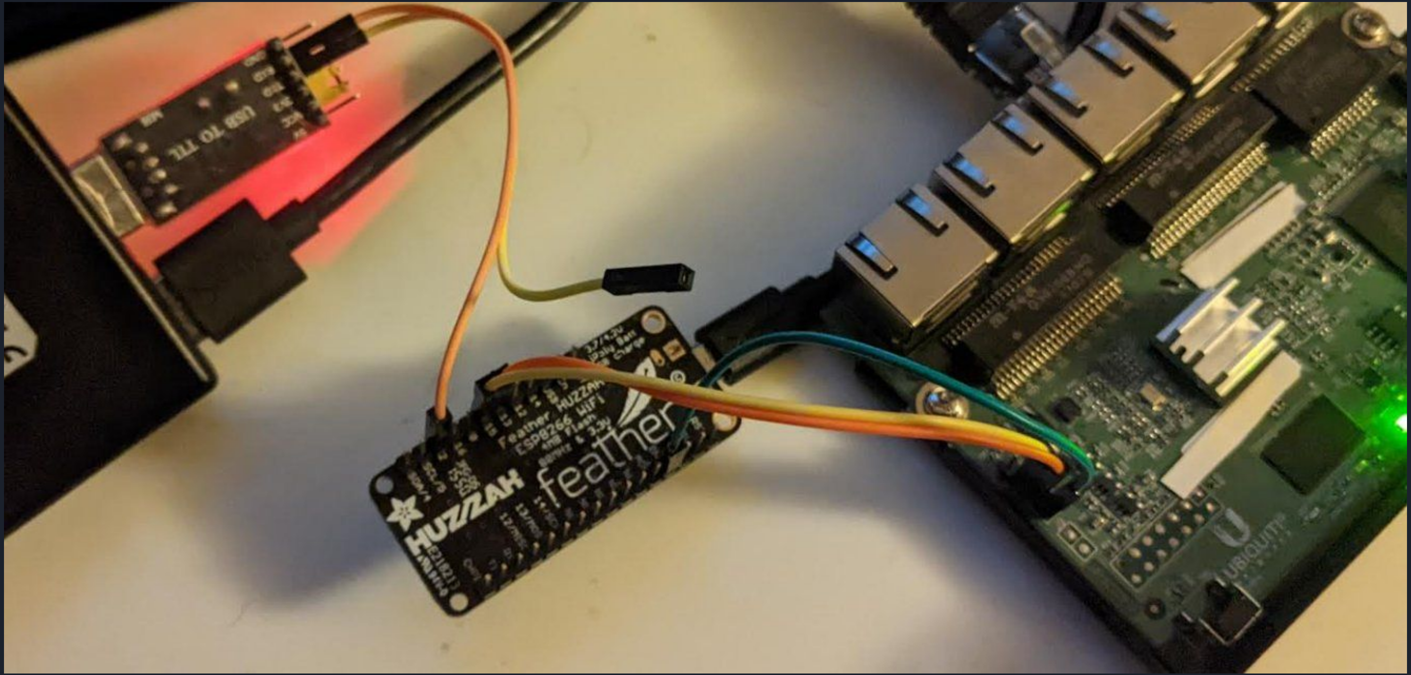
Add features:

- Access your favorite UART
- ESP32 WiFi hotspot or station
- Login via network SSH
- Uses wolfSSH



github.com/wolfSSL/wolfssh-examples/tree/main/Espressif

The SSH project also works on the ESP8266



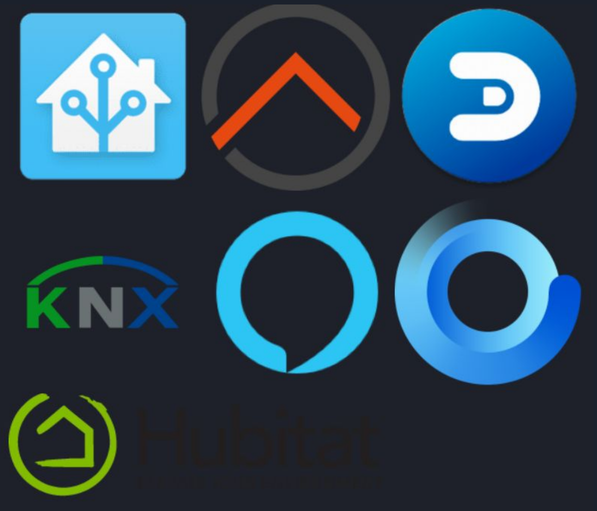
gojimmypi.github.io/SSH-to-ESP8266/

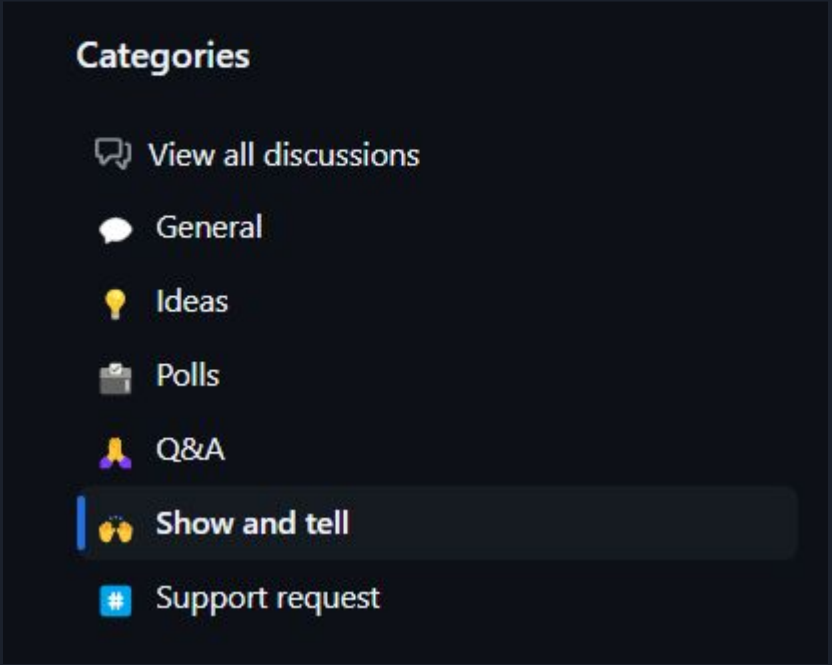
Tasmota

tasmota.github.io

Project Showcase

“Show and tell”

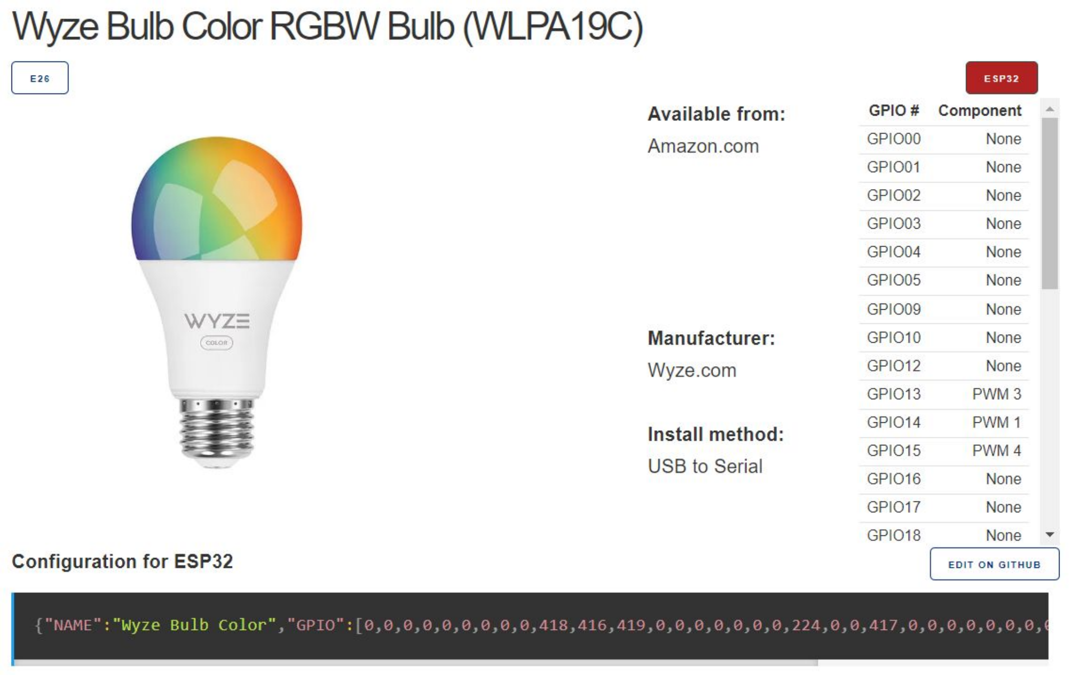




Categories

- View all discussions
- General
- Ideas
- Polls
- Q&A
- **Show and tell**
- Support request

Let’s try a multi-color Wi-Fi bulb.



Wyze Bulb Color RGBW Bulb (WLPA19C)

Available from:
Amazon.com

Manufacturer:
Wyze.com

Install method:
USB to Serial

GPIO #	Component
GPIO00	None
GPIO01	None
GPIO02	None
GPIO03	None
GPIO04	None
GPIO05	None
GPIO09	None
GPIO10	None
GPIO12	None
GPIO13	PWM 3
GPIO14	PWM 1
GPIO15	PWM 4
GPIO16	None
GPIO17	None
GPIO18	None

Configuration for ESP32

```
{"NAME":"Wyze Bulb Color","GPIO":[0,0,0,0,0,0,0,0,0,418,416,419,0,0,0,0,0,0,0,224,0,0,417,0,0,0,0,0,0,0,
```

Image credit: screen snip: templates.blakadder.com/wyze_WLPA19C.html

One might not ever guess there’s an ESP32 inside

Pop off the lamp cover

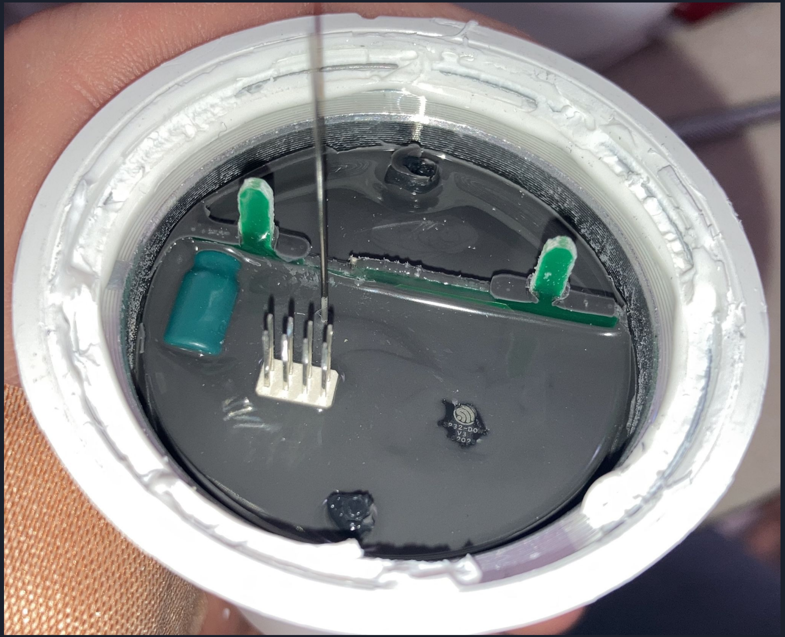


Image credit: Tasmota walk-through imgur.com/a/wyze-bulb-color-UIB1Eux

What I expected

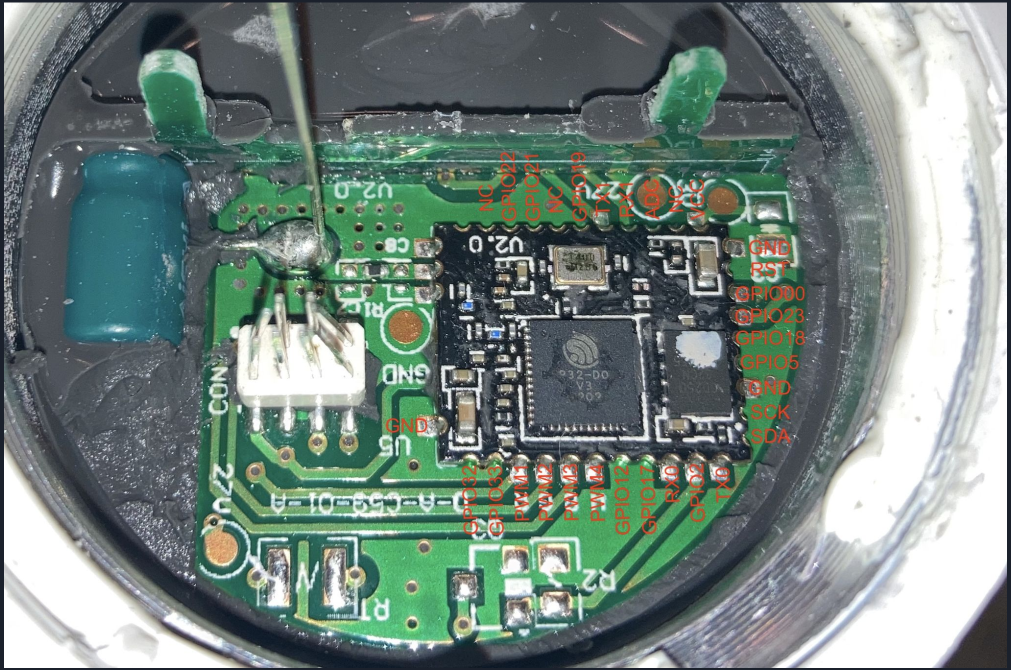
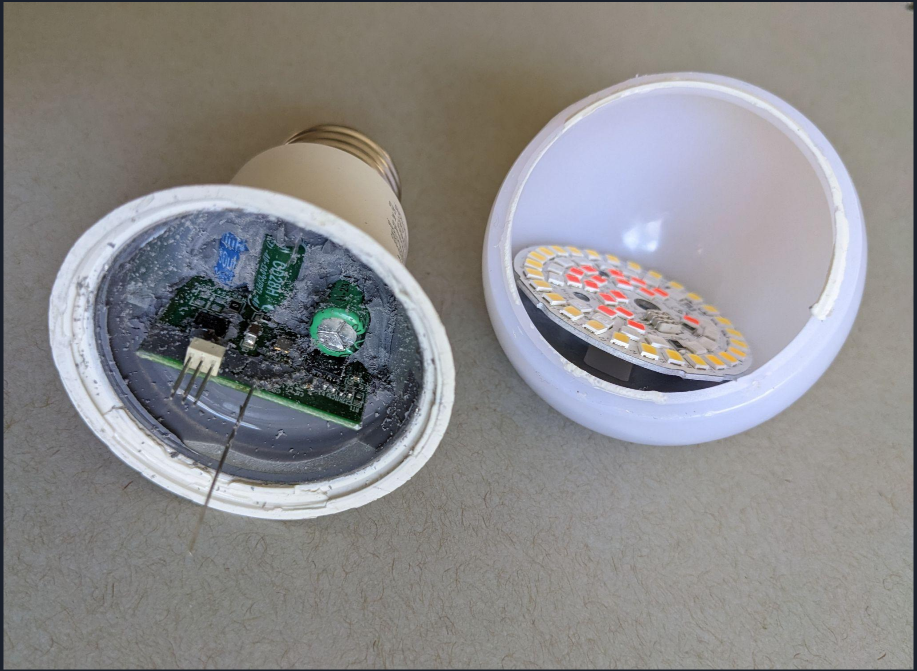


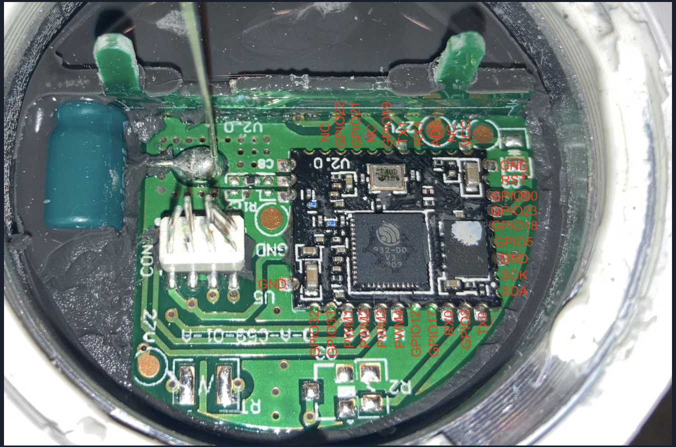
Image credit: Tasmota walk-through. imgur.com/a/wyze-bulb-color-UIB1Eux

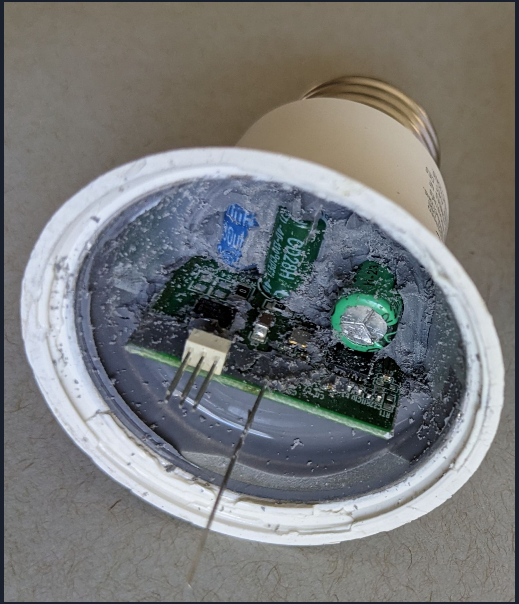
What I actually found



Nothing ever goes as planned

Expected vs Actual pics



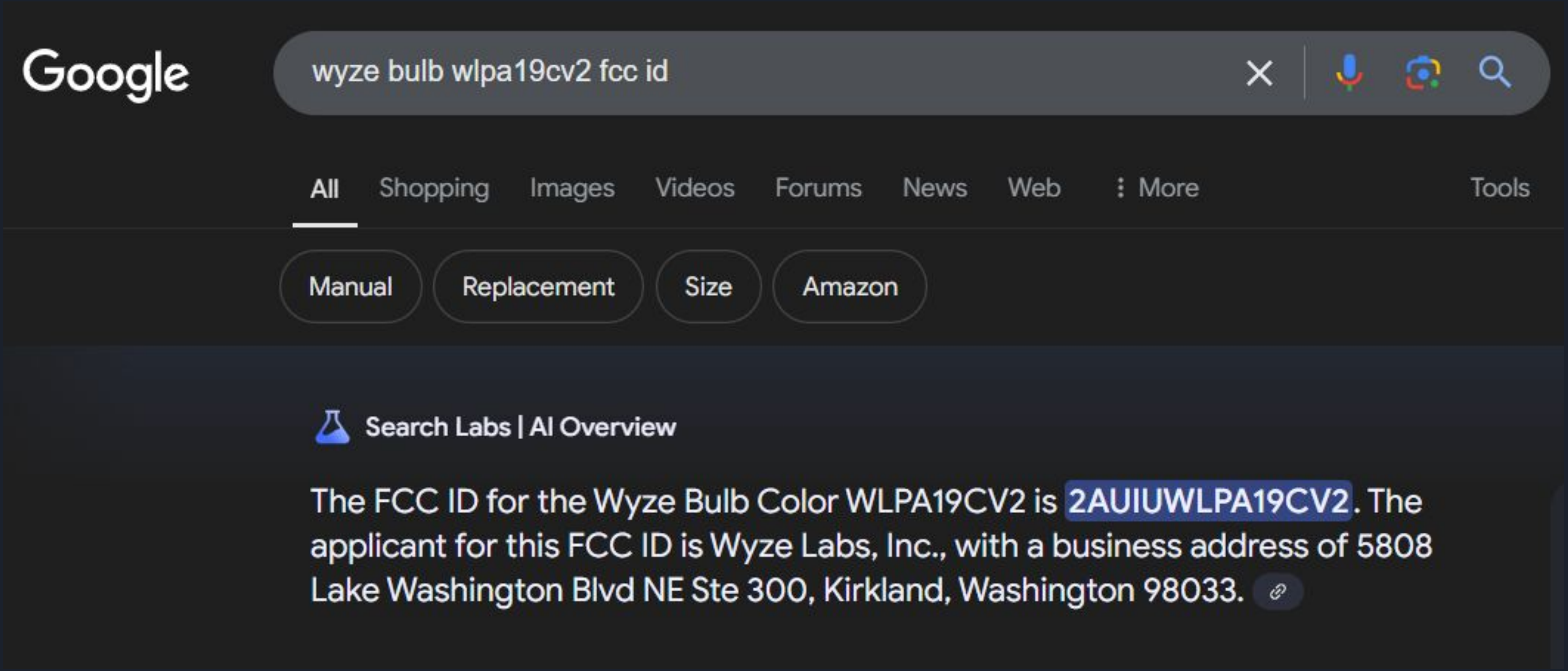


Lessons learned, ask yourself:

- How easy is it to disassemble / reassemble?
- What, exactly will be the new purpose?
- Is there physical room for alternative or extra devices?
- Will you need to use and existing or additional display?
- How to better plan for a repurposing project?

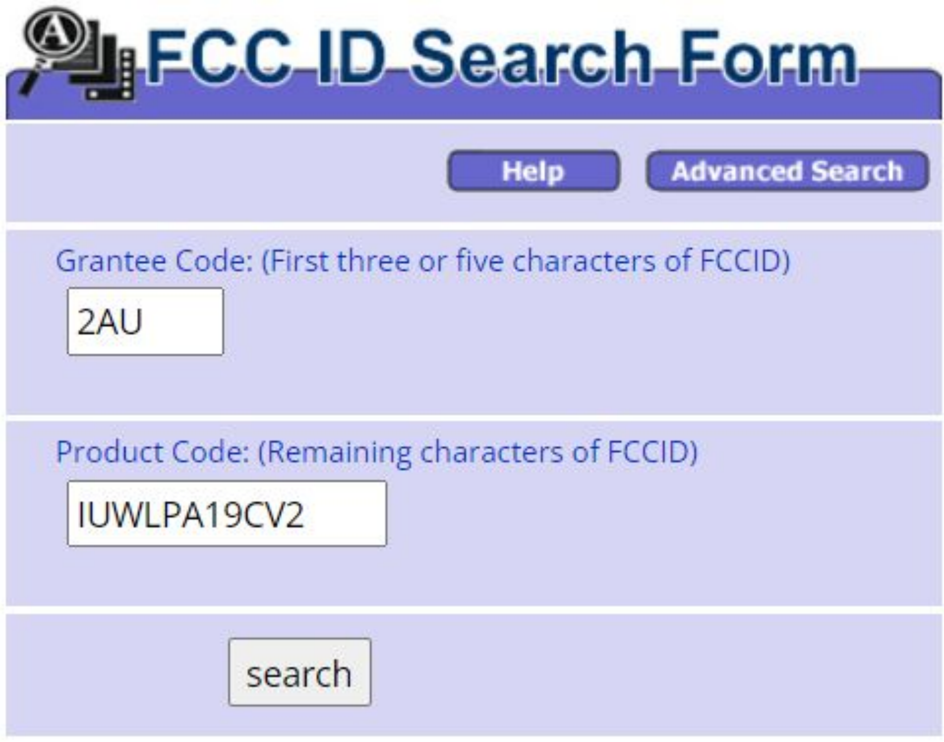
GO Jimmy PI

Find the FCC ID



The FCC ID for the Wyze Bulb Color WLPA19CV2 is 2AUIUWLPA19CV2. The applicant for this FCC ID is Wyze Labs, Inc., with a business address of 5808 Lake Washington Blvd NE Ste 300, Kirkland, Washington 98033.

Visit www.fcc.gov/oet/ea/fccid



FCC ID Search Form

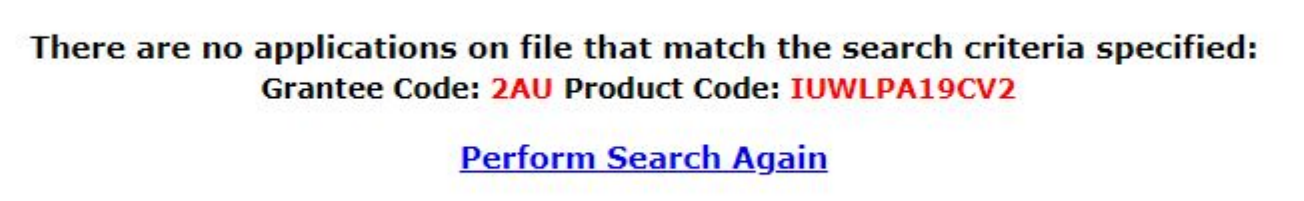
Grantee Code: (First three or five characters of FCCID)

2AU

Product Code: (Remaining characters of FCCID)

IUWLPA19CV2

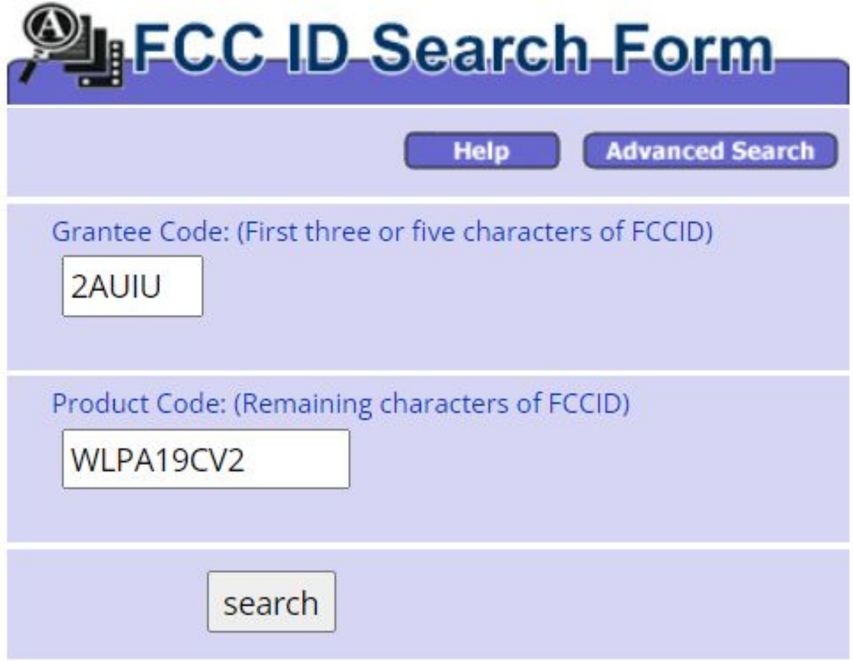
Three or FIVE digits



There are no applications on file that match the search criteria specified:

Grantee Code: 2AU Product Code: IUWLPA19CV2

Search with first 5 digits



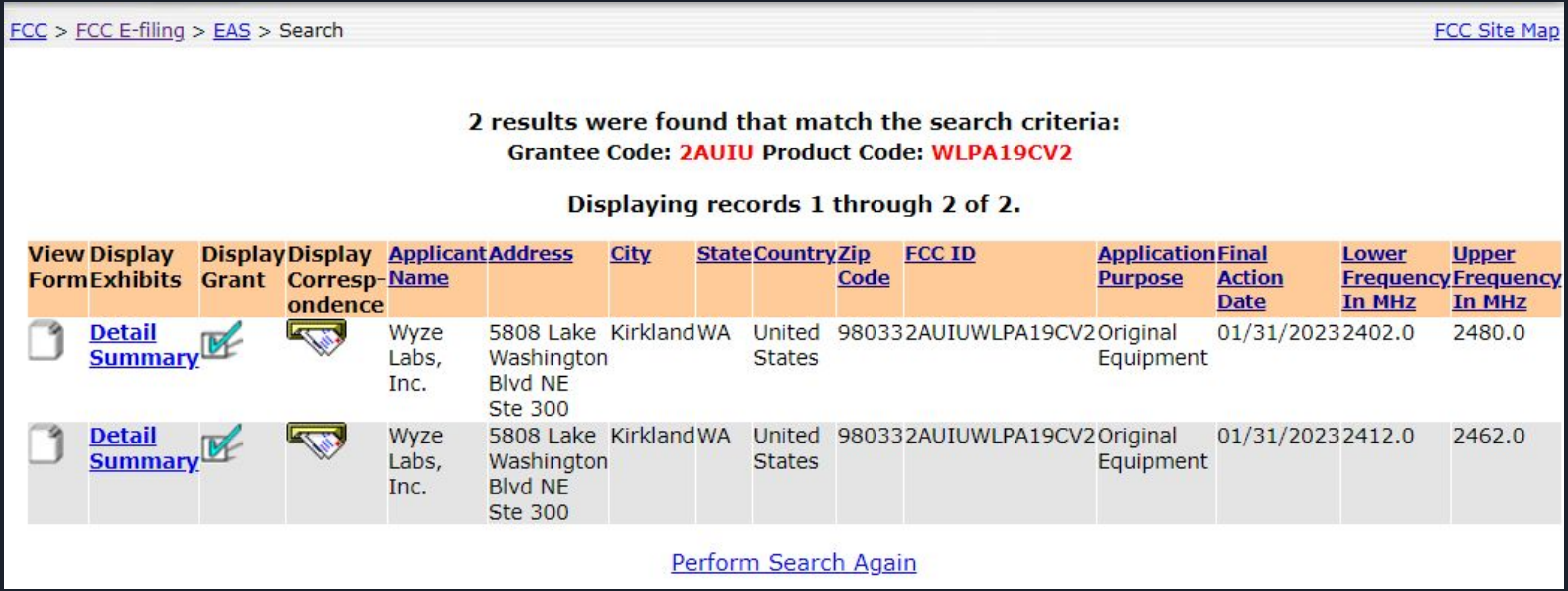
Grantee Code: (First three or five characters of FCCID)

2AUIU

Product Code: (Remaining characters of FCCID)

WLPA19CV2

Search Results!



FCC > FCC E-filing > EAS > Search

FCC Site Map

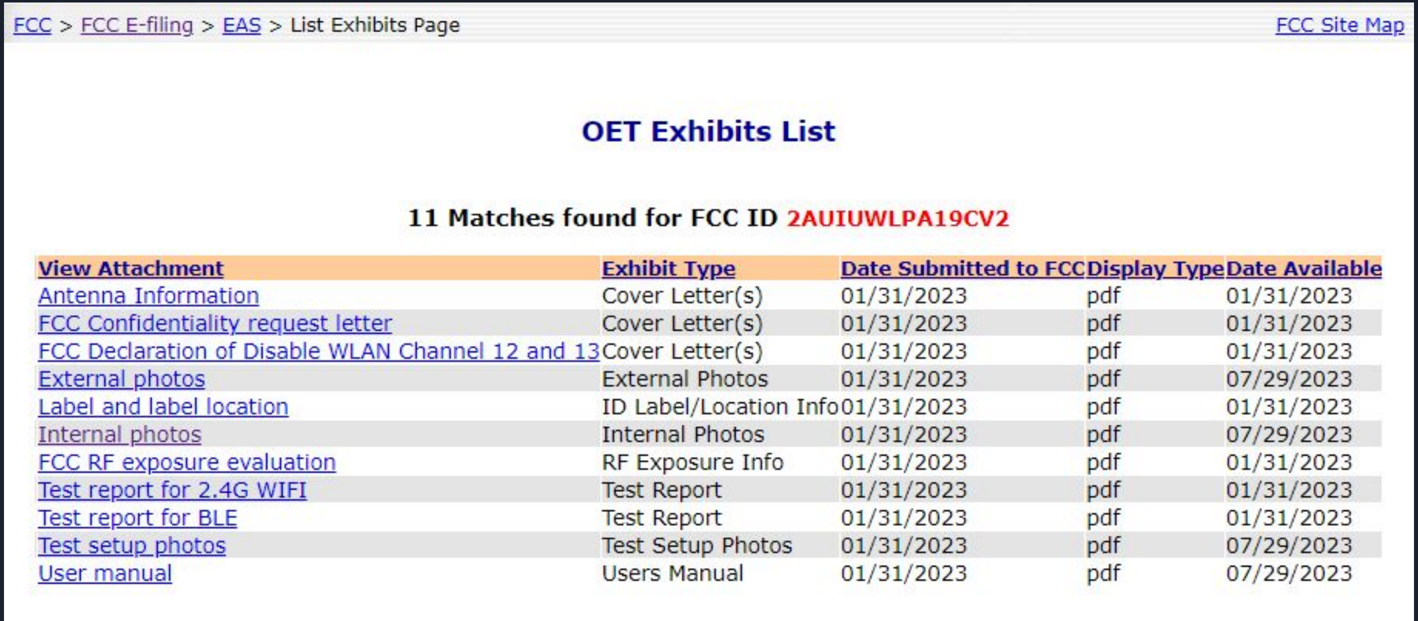
2 results were found that match the search criteria:
Grantee Code: 2AUIU Product Code: WLPA19CV2

Displaying records 1 through 2 of 2.

View Form	Display Exhibits	Display Grant	Display Corresp-ondence	Applicant Name	Address	City	State	Country	Zip Code	FCC ID	Application Purpose	Final Action Date	Lower Frequency In MHz	Upper Frequency In MHz
📄	Detail Summary	✓	✉	Wyze Labs, Inc.	5808 Lake Washington Blvd NE Ste 300	Kirkland	WA	United States	98033	2AUIUWLPA19CV2	Original Equipment	01/31/2023	2402.0	2480.0
📄	Detail Summary	✓	✉	Wyze Labs, Inc.	5808 Lake Washington Blvd NE Ste 300	Kirkland	WA	United States	98033	2AUIUWLPA19CV2	Original Equipment	01/31/2023	2412.0	2462.0

Perform Search Again

OET Exhibits List



FCC > FCC E-filing > EAS > List Exhibits Page

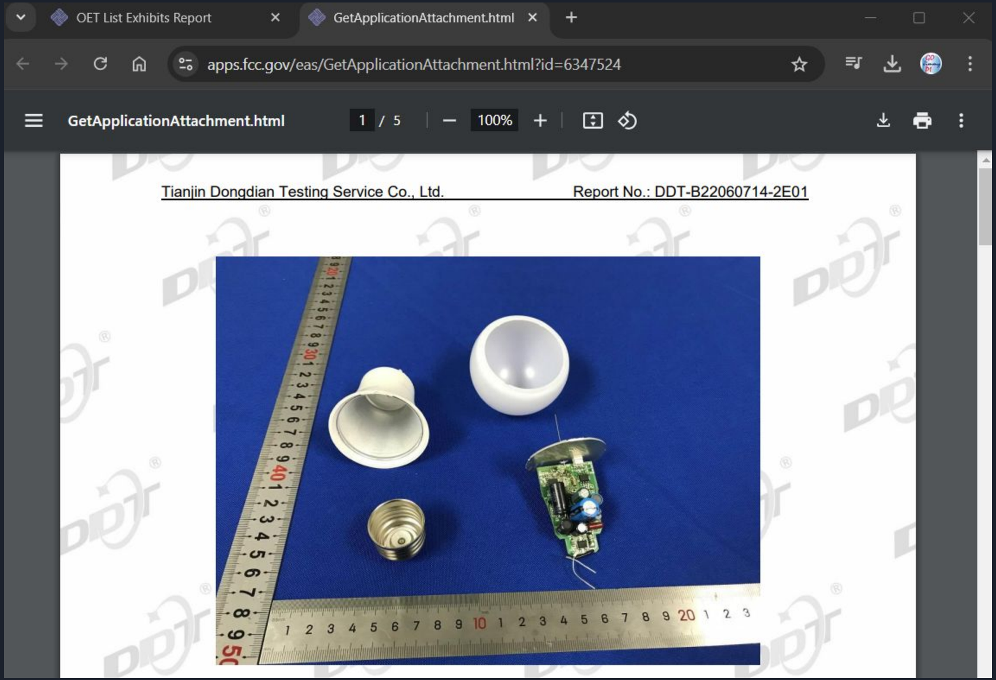
FCC Site Map

OET Exhibits List

11 Matches found for FCC ID 2AUIUWLPA19CV2

View Attachment	Exhibit Type	Date Submitted to FCC	Display Type	Date Available
Antenna Information	Cover Letter(s)	01/31/2023	pdf	01/31/2023
FCC Confidentiality request letter	Cover Letter(s)	01/31/2023	pdf	01/31/2023
FCC Declaration of Disable WLAN Channel 12 and 13	Cover Letter(s)	01/31/2023	pdf	01/31/2023
External photos	External Photos	01/31/2023	pdf	07/29/2023
Label and label location	ID Label/Location Info	01/31/2023	pdf	01/31/2023
Internal photos	Internal Photos	01/31/2023	pdf	07/29/2023
FCC RF exposure evaluation	RF Exposure Info	01/31/2023	pdf	01/31/2023
Test report for 2.4G WIFI	Test Report	01/31/2023	pdf	01/31/2023
Test report for BLE	Test Report	01/31/2023	pdf	01/31/2023
Test setup photos	Test Setup Photos	01/31/2023	pdf	07/29/2023
User manual	Users Manual	01/31/2023	pdf	07/29/2023

First photo would have been useful to see

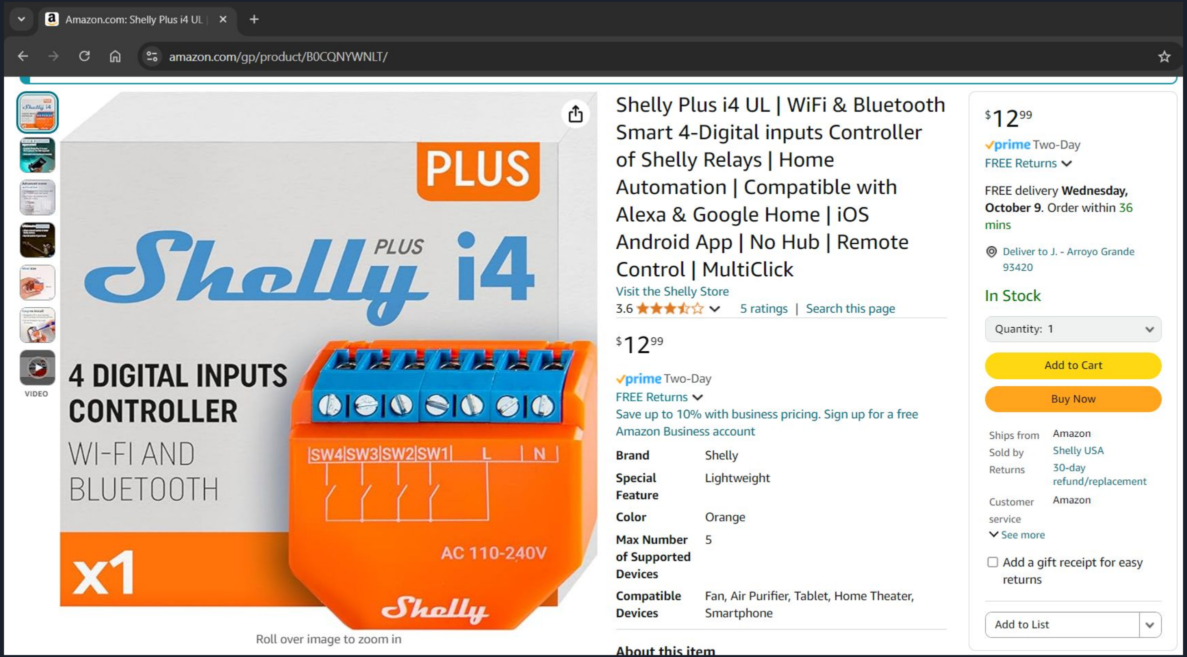

OET List Exhibits Report
GetApplicationAttachment.html
apps.fcc.gov/eas/GetApplicationAttachment.html?id=6347524
GetApplicationAttachment.html
1 / 5
100%

Tianjin Dongdian Testing Service Co., Ltd. Report No.: DDT-B22060714-2E01

Oh no,

So…This is all impossible?

Let’s look at another commercial product


Amazon.com: Shelly Plus i4 UL |
amazon.com/gp/product/B0CQNYWNLT/
PLUS
Shelly PLUS i4
4 DIGITAL INPUTS CONTROLLER
WI-FI AND BLUETOOTH
x1
SW4 SW3 SW2 SW1 L N
AC 110-240V
Shelly
VIDEO
Roll over image to zoom in

Shelly Plus i4 UL | WiFi & Bluetooth Smart 4-Digital inputs Controller of Shelly Relays | Home Automation | Compatible with Alexa & Google Home | iOS Android App | No Hub | Remote Control | MultiClick

Visit the Shelly Store

3.6 ★★★★☆ 5 ratings | Search this page

$12.99

✓prime Two-Day

FREE Returns

Save up to 10% with business pricing. Sign up for a free Amazon Business account

Brand	Shelly
Special Feature	Lightweight
Color	Orange
Max Number of Supported Devices	5
Compatible Devices	Fan, Air Purifier, Tablet, Home Theater, Smartphone

About this item

$12.99

✓prime Two-Day

FREE Returns

FREE delivery **Wednesday, October 9**. Order within 36 mins

Deliver to J. - Arroyo Grande 93420

In Stock

Quantity: 1

Add to Cart

Buy Now

Ships from	Amazon
Sold by	Shelly USA
Returns	30-day refund/replacement
Customer service	Amazon

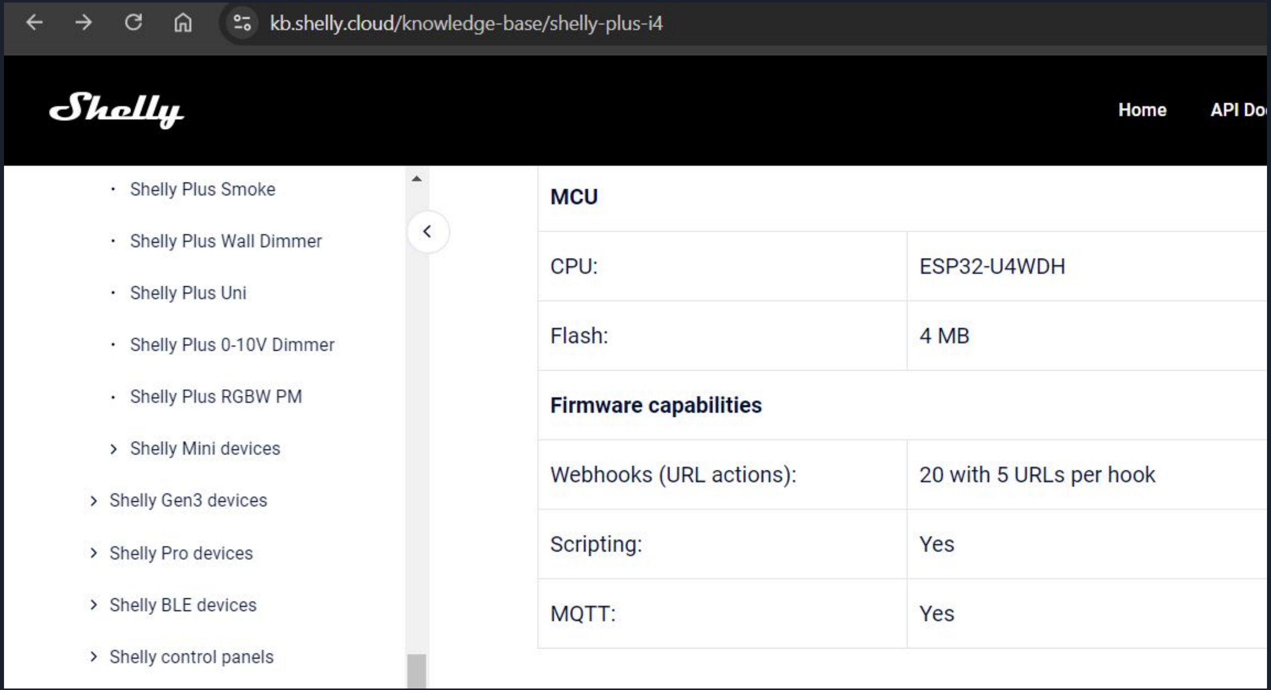
See more

□ Add a gift receipt for easy returns

Add to List

Image credit screen snip from amazon.com/gp/product/B0CQNYWNLT/

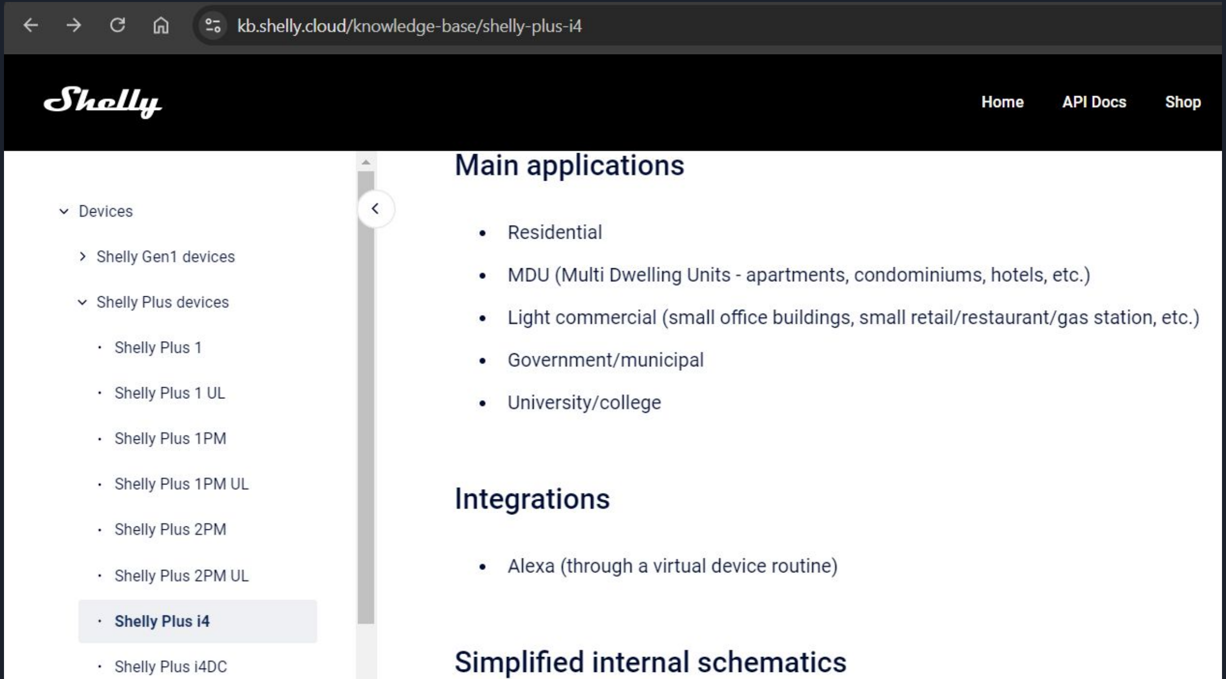
ESP32 Inside!



MCU	
CPU:	ESP32-U4WDH
Flash:	4 MB
Firmware capabilities	
Webhooks (URL actions):	20 with 5 URLs per hook
Scripting:	Yes
MQTT:	Yes

Image credit screen snip from kb.shelly.cloud/knowledge-base/shelly-plus-i4

Works only with Alexa. Let's change that!



kb.shelly.cloud/knowledge-base/shelly-plus-i4

Shelly

Home API Docs Shop

- Devices
  - Shelly Gen1 devices
  - Shelly Plus devices
    - Shelly Plus 1
    - Shelly Plus 1 UL
    - Shelly Plus 1PM
    - Shelly Plus 1PM UL
    - Shelly Plus 2PM
    - Shelly Plus 2PM UL
    - **Shelly Plus i4**
    - Shelly Plus i4DC

Main applications

- Residential
- MDU (Multi Dwelling Units - apartments, condominiums, hotels, etc.)
- Light commercial (small office buildings, small retail/restaurant/gas station, etc.)
- Government/municipal
- University/college

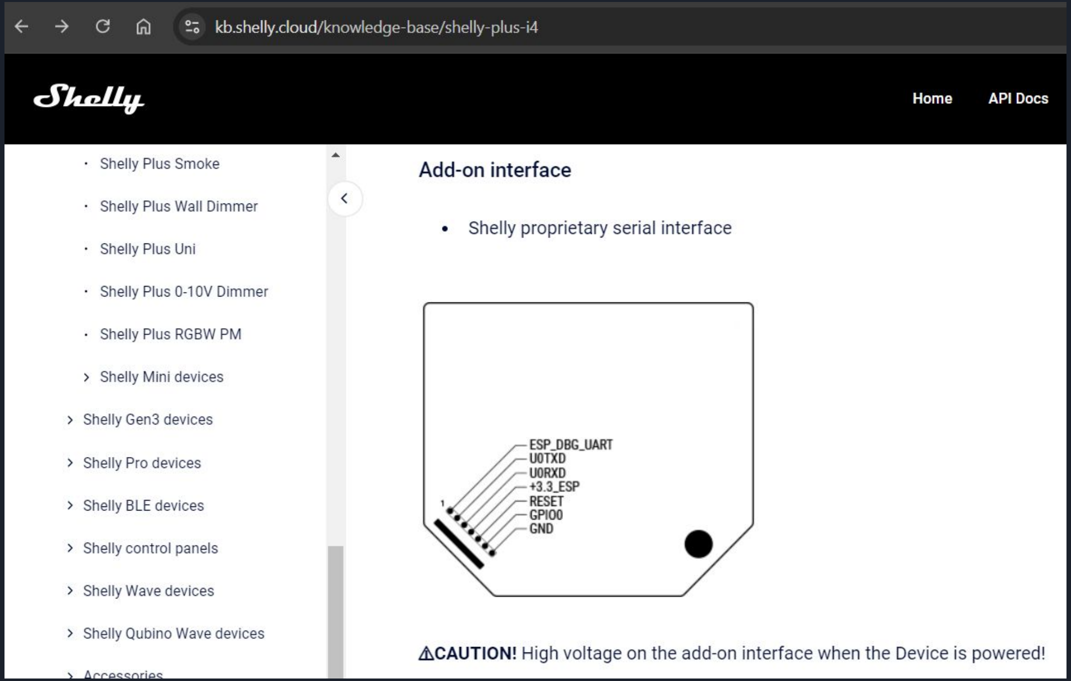
Integrations

- Alexa (through a virtual device routine)

Simplified internal schematics

Image credit screen snip from: kb.shelly.cloud/knowledge-base/shelly-plus-i4

Manufacturer with awesome docs!



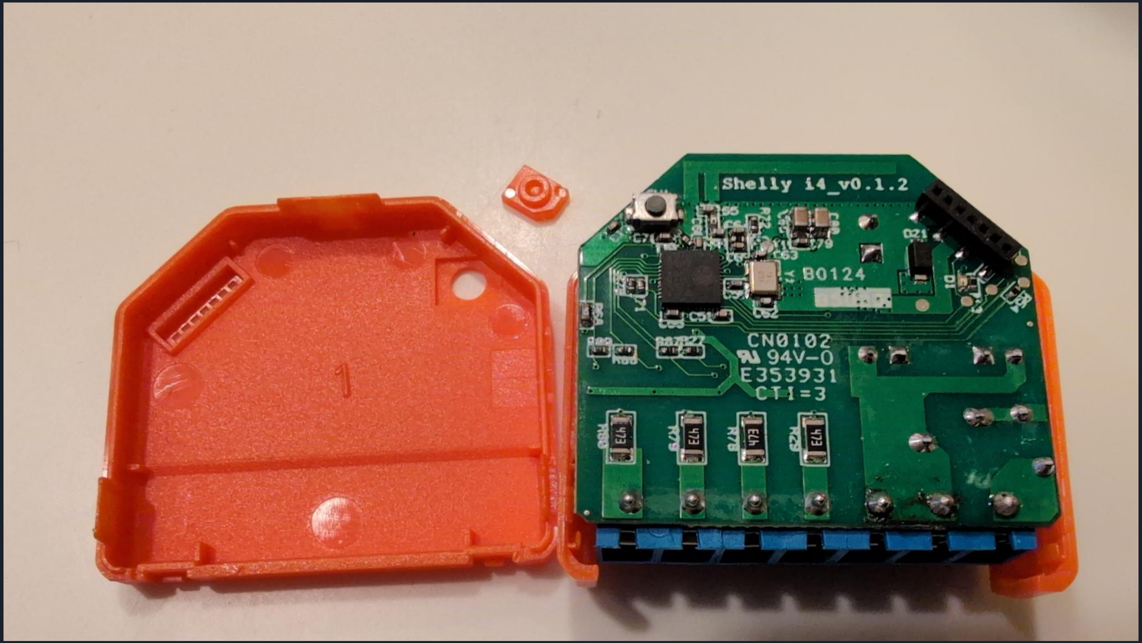
Add-on interface

- Shelly proprietary serial interface

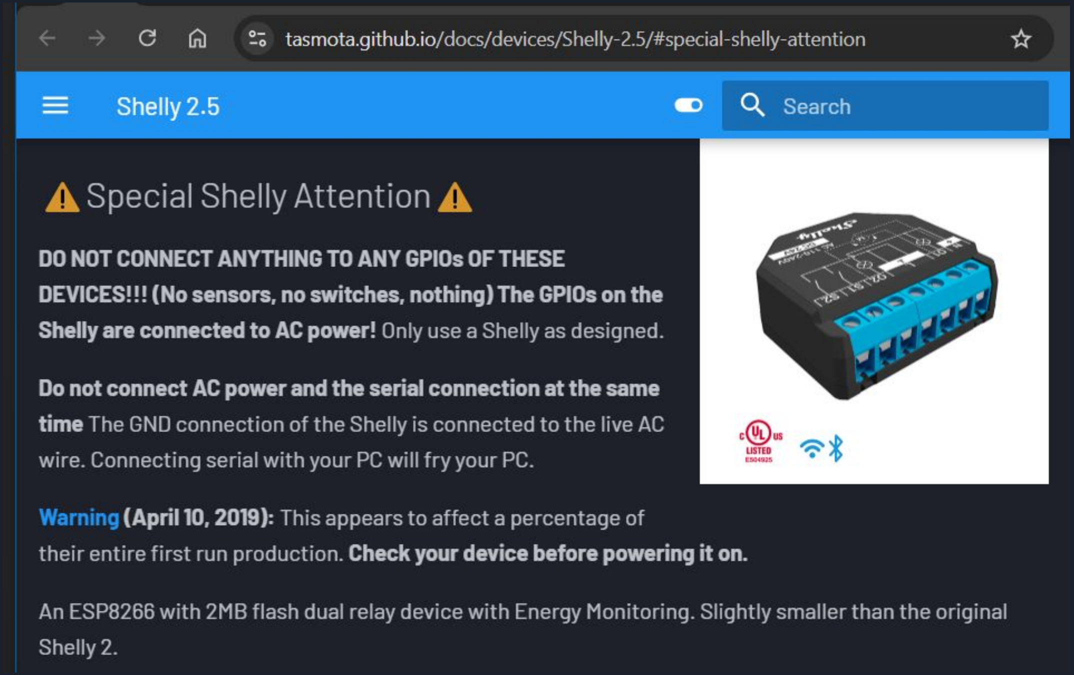
⚠**CAUTION!** High voltage on the add-on interface when the Device is powered!

Image credit: screen snip from kb.shelly.cloud/knowledge-base/shelly-plus-i4

Easily disassembled, or not!



Reminder about safety


tasmota.github.io/docs/devices/Shelly-2.5/#special-shelly-attention
Shelly 2.5
Search

⚠ Special Shelly Attention ⚠

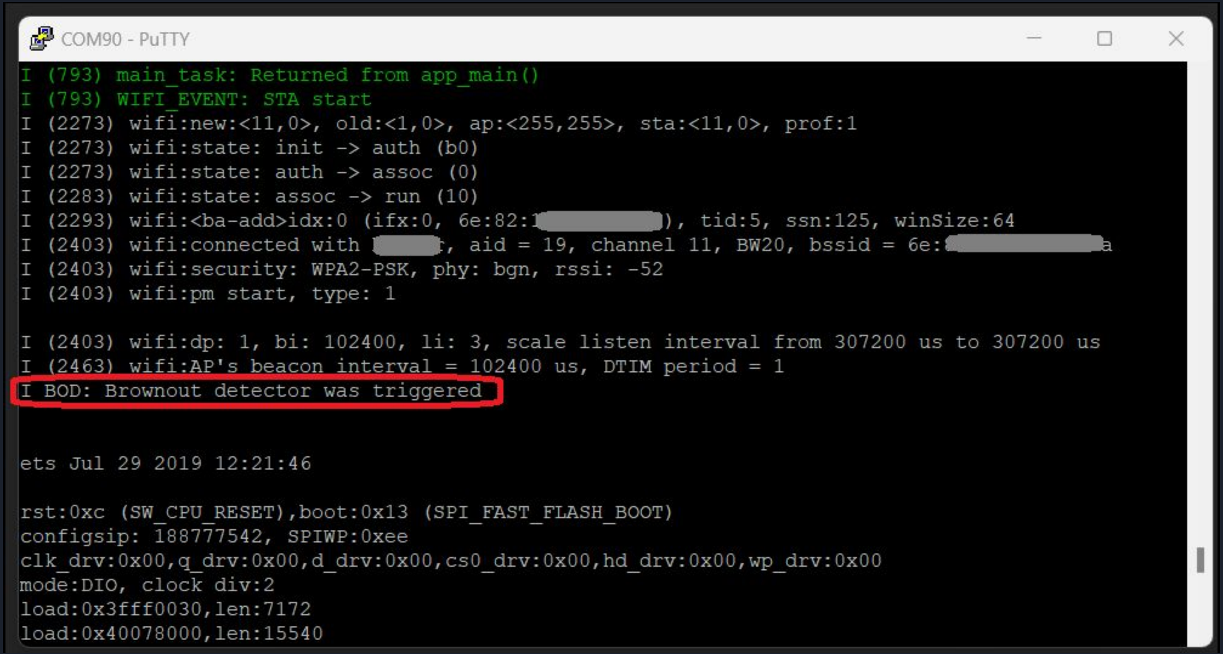
DO NOT CONNECT ANYTHING TO ANY GPIOs OF THESE DEVICES!!! (No sensors, no switches, nothing) The GPIOs on the Shelly are connected to AC power! Only use a Shelly as designed.

Do not connect AC power and the serial connection at the same time The GND connection of the Shelly is connected to the live AC wire. Connecting serial with your PC will fry your PC.

Warning **(April 10, 2019):** This appears to affect a percentage of their entire first run production. **Check your device before powering it on.**

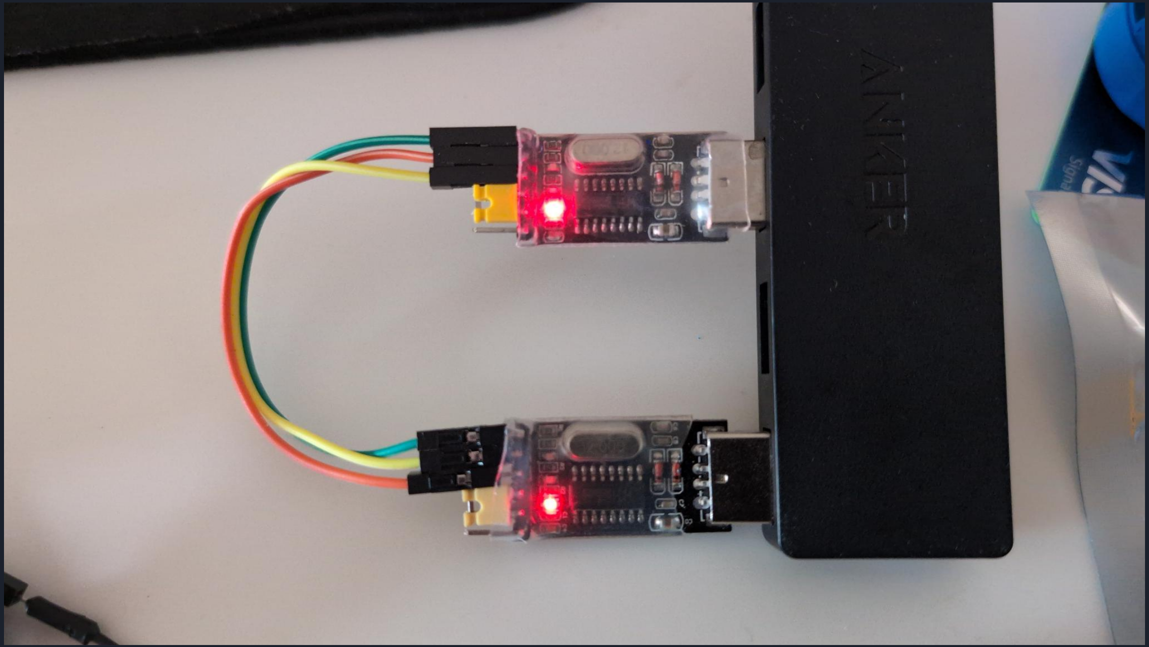
An ESP8266 with 2MB flash dual relay device with Energy Monitoring. Slightly smaller than the original Shelly 2.

Remember “***don’t power from USB TTY serial***” ?

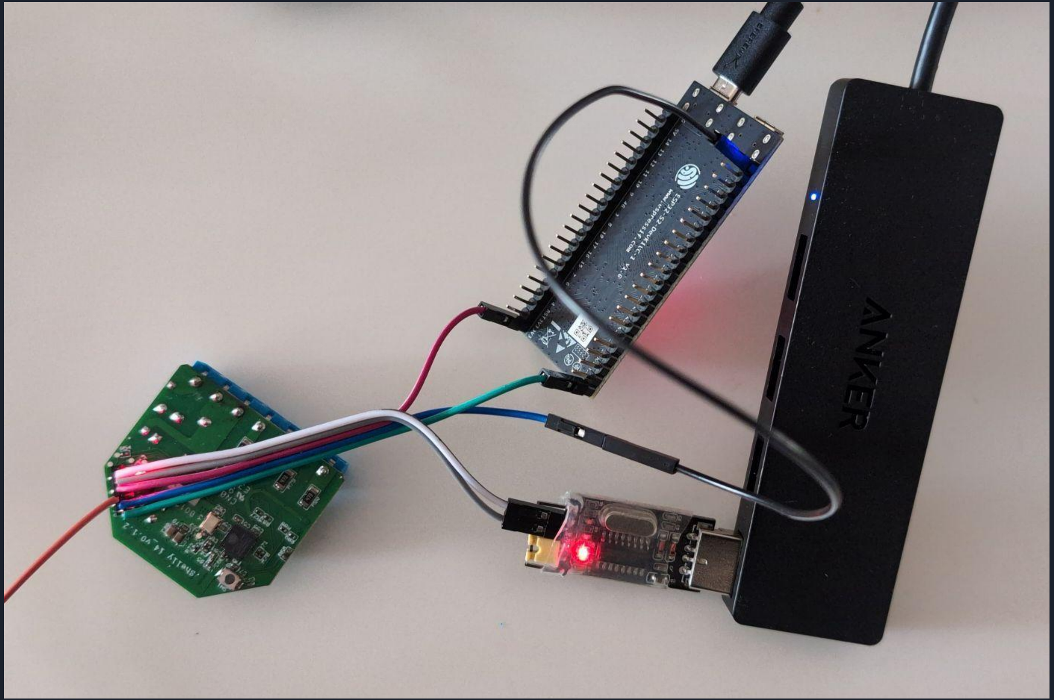


```
I (793) main_task: Returned from app_main()
I (793) WIFI_EVENT: STA start
I (2273) wifi:new:<11,0>, old:<1,0>, ap:<255,255>, sta:<11,0>, prof:1
I (2273) wifi:state: init -> auth (b0)
I (2273) wifi:state: auth -> assoc (0)
I (2283) wifi:state: assoc -> run (10)
I (2293) wifi:<ba-add>idx:0 (ifx:0, 6e:82:[redacted]), tid:5, ssn:125, winSize:64
I (2403) wifi:connected with [redacted], aid = 19, channel 11, BW20, bssid = 6e:[redacted]
I (2403) wifi:security: WPA2-PSK, phy: bgn, rssi: -52
I (2403) wifi:pm start, type: 1

I (2403) wifi:dp: 1, bi: 102400, li: 3, scale listen interval from 307200 us to 307200 us
I (2463) wifi:AP's beacon interval = 102400 us, DTIM period = 1
I BOD: Brownout detector was triggered


ets Jul 29 2019 12:21:46

rst:0xc (SW_CPU_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 188777542, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:2
load:0x3fff0030,len:7172
load:0x40078000,len:15540
```

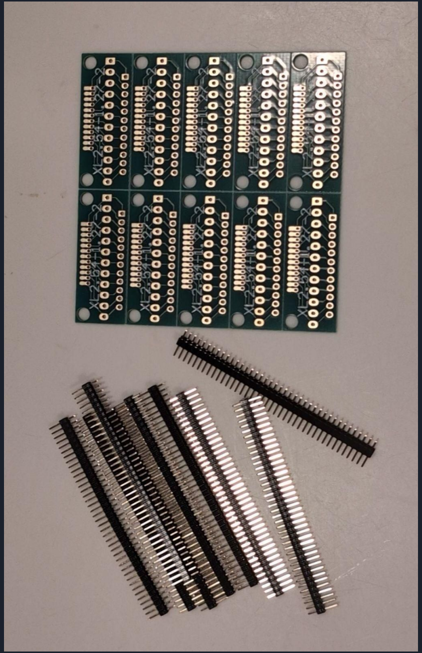
Does my USB-TTY work properly?

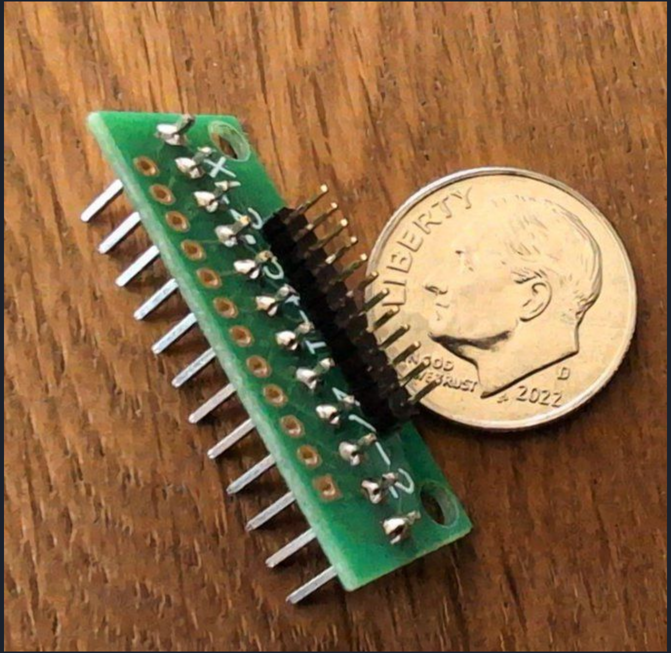


3.3v can come from a variety of sources

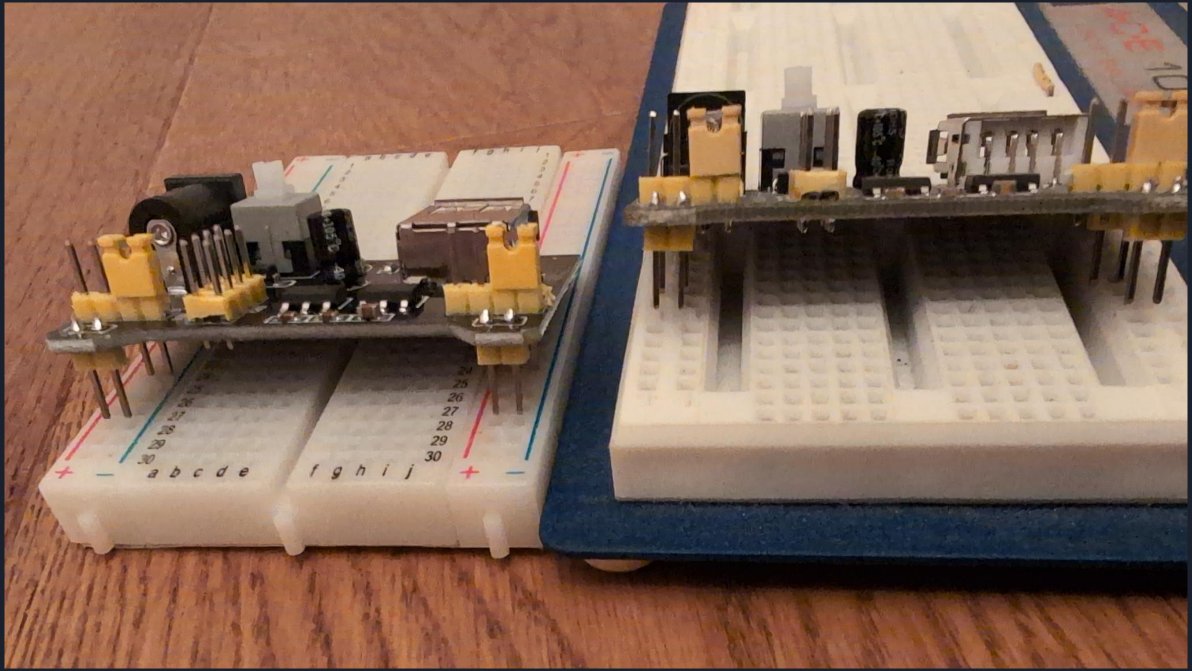


Consider an adapter board

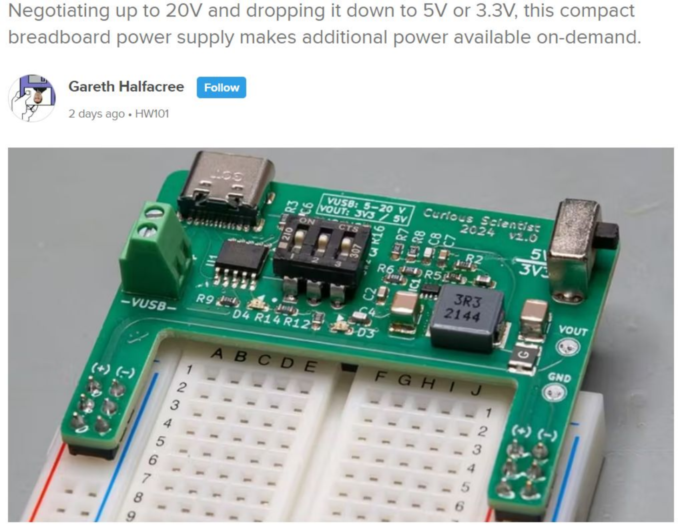




Breadboard power supplies are not all equal



USB Breadboard supply



Negotiating up to 20V and dropping it down to 5V or 3.3V, this compact breadboard power supply makes additional power available on-demand.

Gareth Halfacree Follow

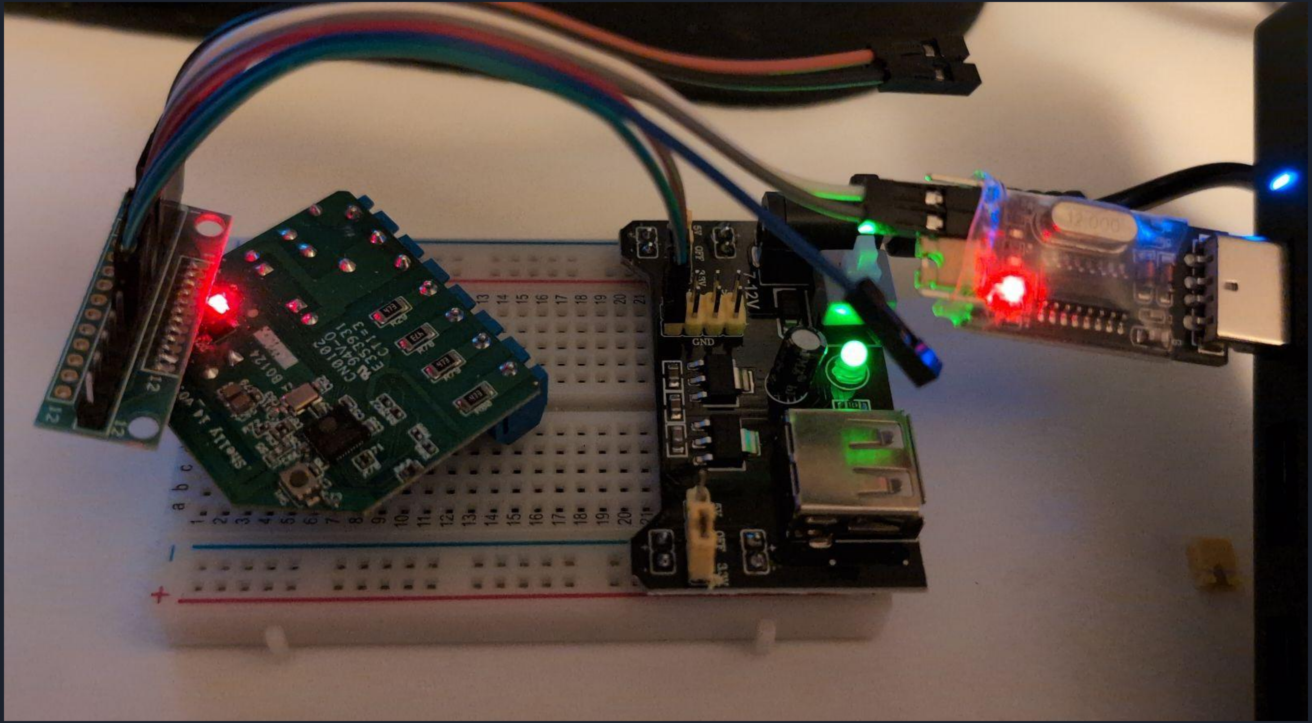
2 days ago • HW101

See also: curiousscientist.tech/blog/usb-pd-decoy-breadboard-power-supply

Image Credit screen snip from:
www.hackster.io/news/curious-scientist-takes-a-new-approach-to-breadboard-power-supplies-with-a-usb-pd-decoy-board-50e5f757d146

Hold down GPIO-0 to ground the entire time from power up to flash completion.

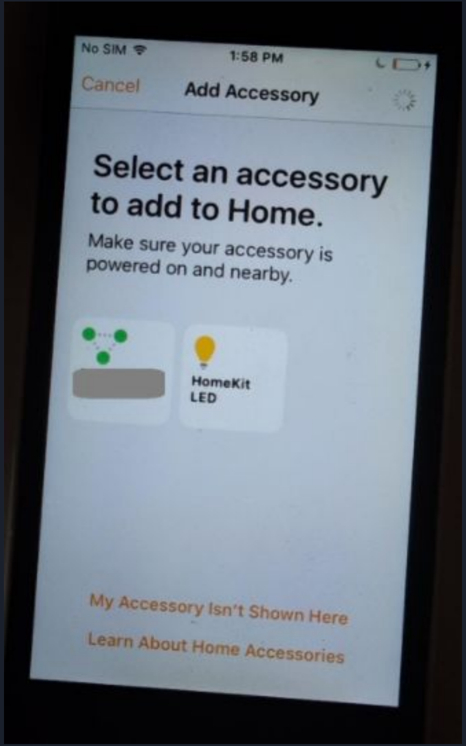
Release jumper to ground, reboot.



Programming steps

```
WRK_IDF_PATH=/mnt/c/SysGCC/esp32/esp-idf/v5.2-master

. ${WRK_IDF_PATH}/export.sh

idf.py menuconfig

idf.py erase-flash -p /dev/ttys90 -b 115200

idf.py -p /dev/ttys90 -b 115200 flash
```

See the HomeKit LED on the Apple iPhone



No SIM 1:58 PM

Cancel Add Accessory

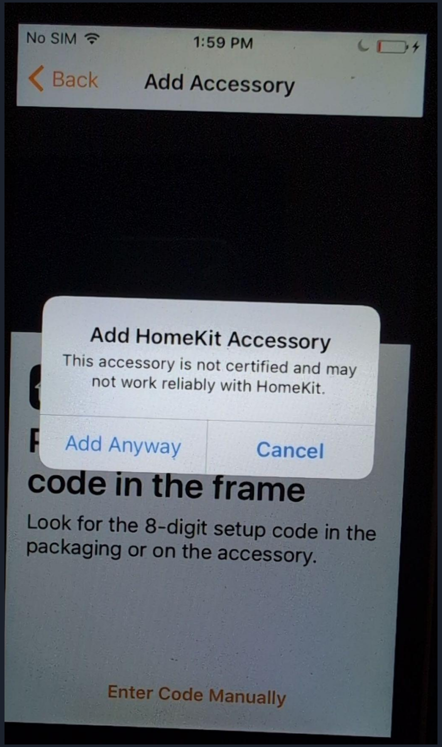
Select an accessory to add to Home.

Make sure your accessory is powered on and nearby.

My Accessory Isn't Shown Here

Learn About Home Accessories

Add the accessory. Not certified, add anyway!



No SIM 1:59 PM

Back Add Accessory

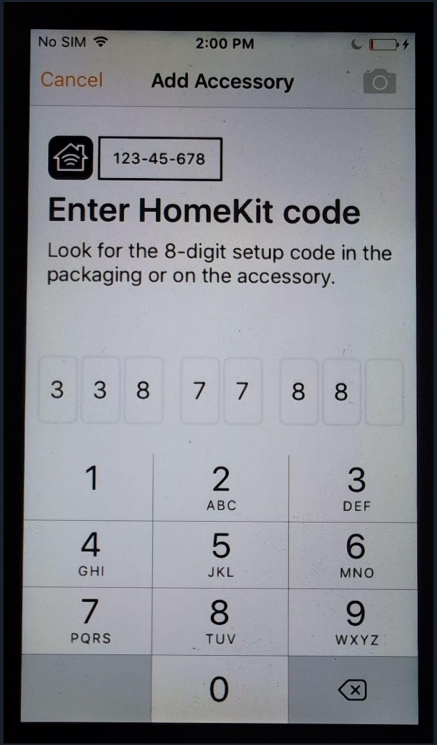
Add HomeKit Accessory

This accessory is not certified and may not work reliably with HomeKit.

code in the frame

Look for the 8-digit setup code in the packaging or on the accessory.

Default code set in `idf.py menuconfig`
The last digit is a “3”:



No SIM 2:00 PM

Cancel **Add Accessory**

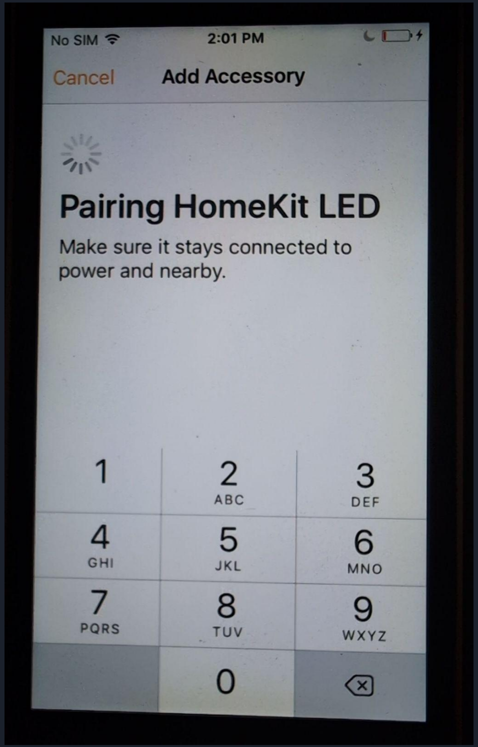
123-45-678

Enter HomeKit code

Look for the 8-digit setup code in the packaging or on the accessory.

3 3 8 7 7 8 8

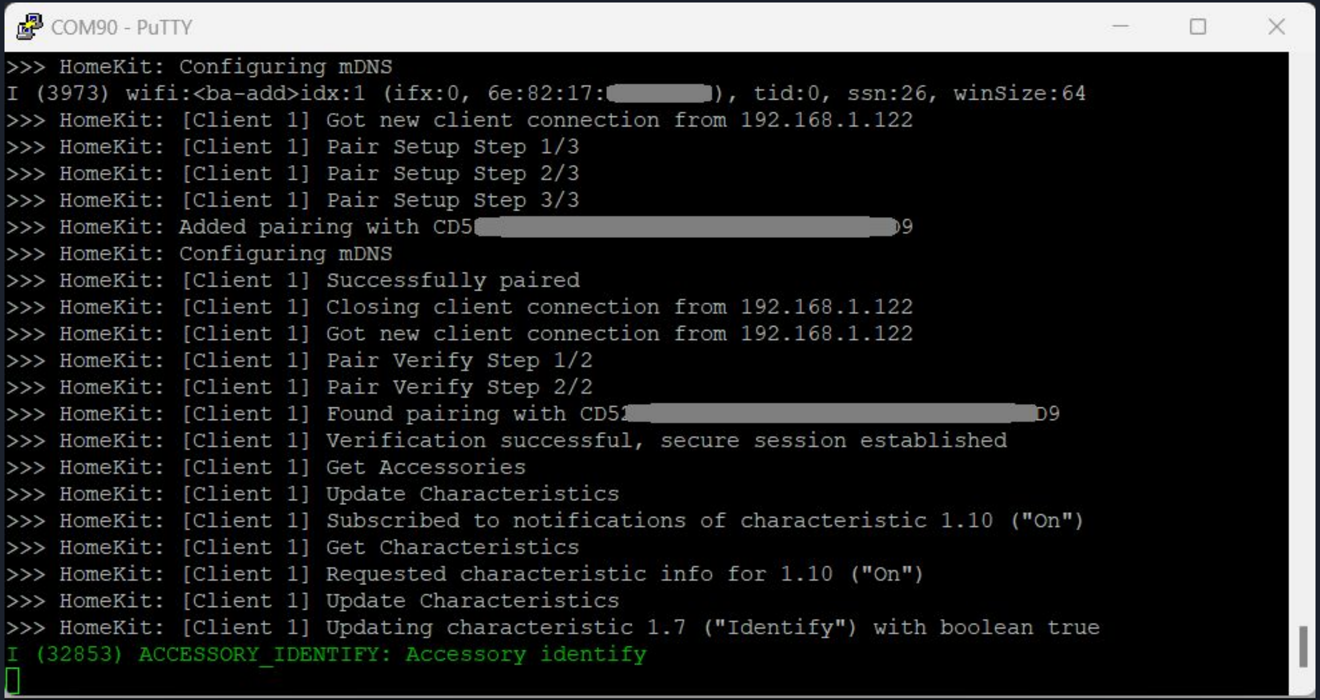
Wait a short time



Pairing HomeKit LED

Make sure it stays connected to power and nearby.

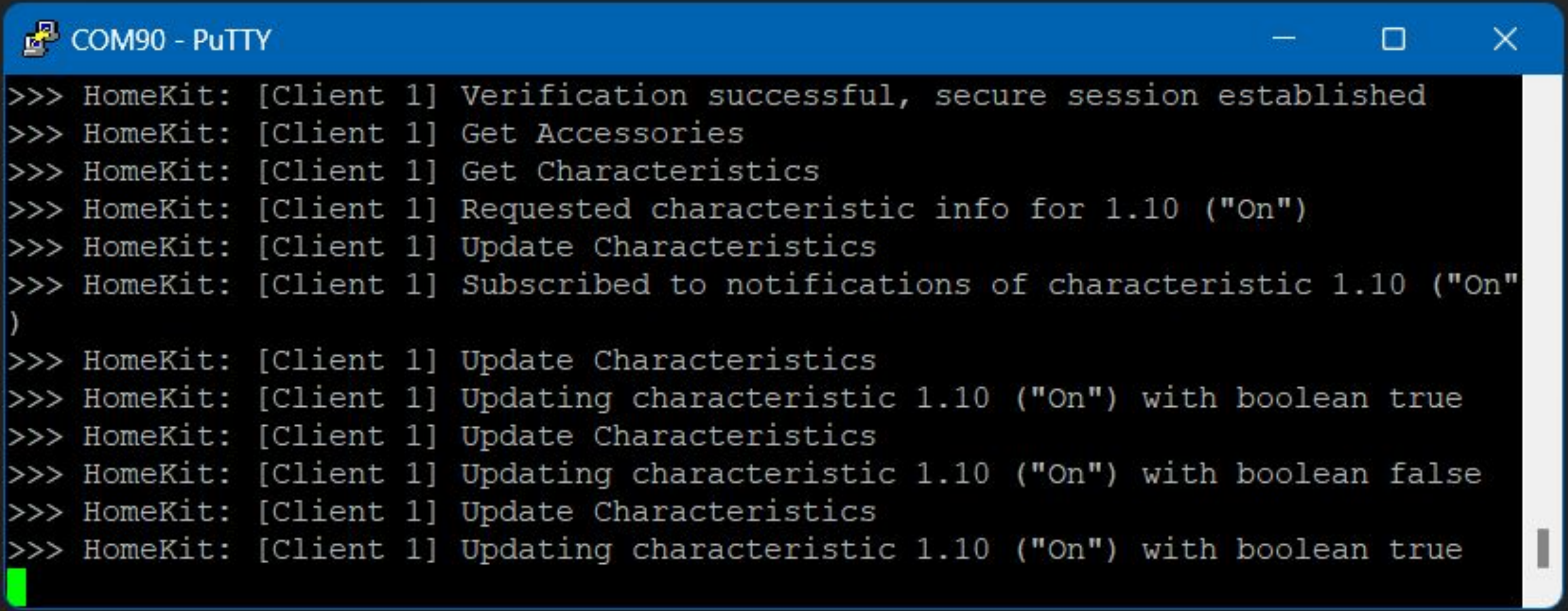
Success!



```
>>> HomeKit: Configuring mDNS
I (3973) wifi:<ba-add>idx:1 (ifx:0, 6e:82:17:[redacted]), tid:0, ssn:26, winSize:64
>>> HomeKit: [Client 1] Got new client connection from 192.168.1.122
>>> HomeKit: [Client 1] Pair Setup Step 1/3
>>> HomeKit: [Client 1] Pair Setup Step 2/3
>>> HomeKit: [Client 1] Pair Setup Step 3/3
>>> HomeKit: Added pairing with CD5[redacted]9
>>> HomeKit: Configuring mDNS
>>> HomeKit: [Client 1] Successfully paired
>>> HomeKit: [Client 1] Closing client connection from 192.168.1.122
>>> HomeKit: [Client 1] Got new client connection from 192.168.1.122
>>> HomeKit: [Client 1] Pair Verify Step 1/2
>>> HomeKit: [Client 1] Pair Verify Step 2/2
>>> HomeKit: [Client 1] Found pairing with CD5[redacted]D9
>>> HomeKit: [Client 1] Verification successful, secure session established
>>> HomeKit: [Client 1] Get Accessories
>>> HomeKit: [Client 1] Update Characteristics
>>> HomeKit: [Client 1] Subscribed to notifications of characteristic 1.10 ("On")
>>> HomeKit: [Client 1] Get Characteristics
>>> HomeKit: [Client 1] Requested characteristic info for 1.10 ("On")
>>> HomeKit: [Client 1] Update Characteristics
>>> HomeKit: [Client 1] Updating characteristic 1.7 ("Identify") with boolean true
I (32853) ACCESSORY_IDENTIFY: Accessory identify
```

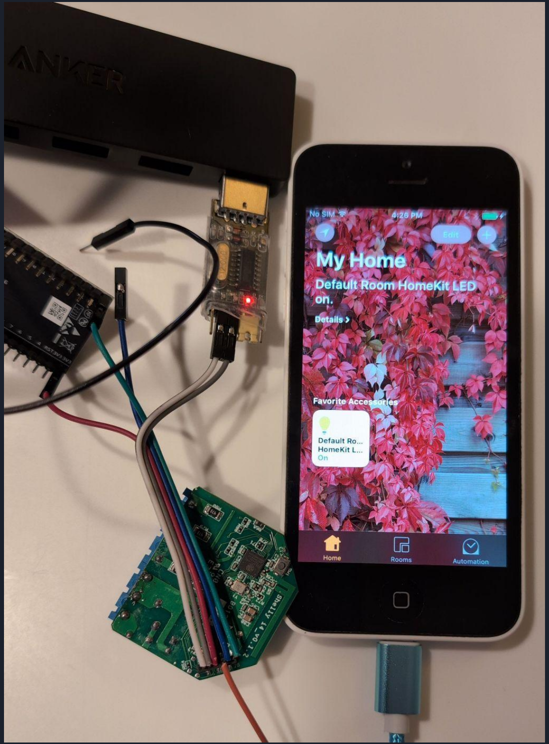
COM port will show toggle “on”: true or false:

(active low: “off” will turn ***on*** the LED!)



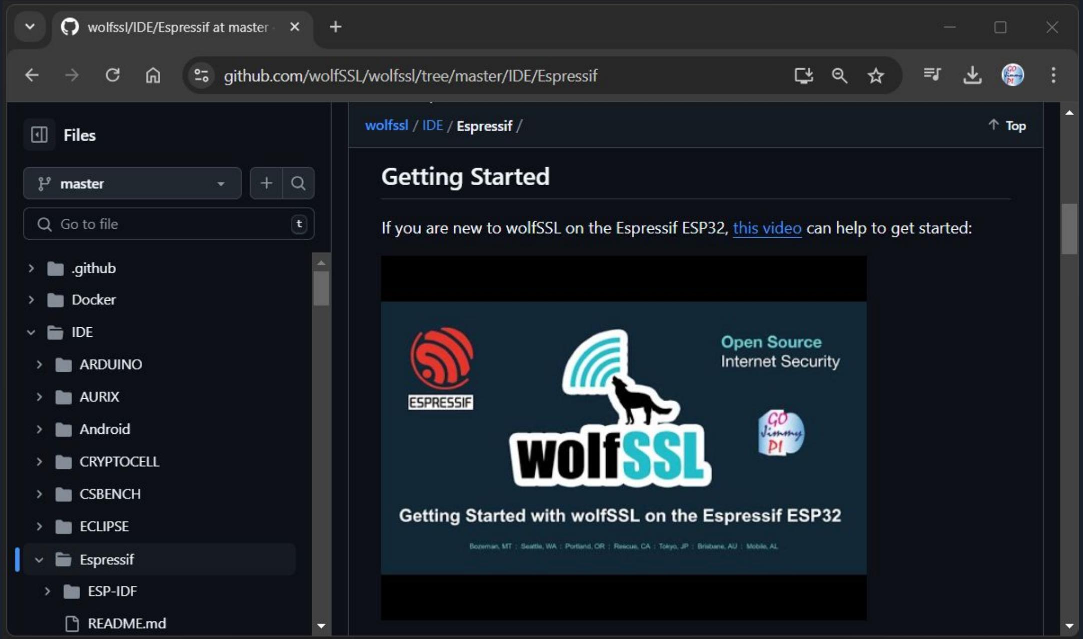
```
>>> HomeKit: [Client 1] Verification successful, secure session established
>>> HomeKit: [Client 1] Get Accessories
>>> HomeKit: [Client 1] Get Characteristics
>>> HomeKit: [Client 1] Requested characteristic info for 1.10 ("On")
>>> HomeKit: [Client 1] Update Characteristics
>>> HomeKit: [Client 1] Subscribed to notifications of characteristic 1.10 ("On"
)
>>> HomeKit: [Client 1] Update Characteristics
>>> HomeKit: [Client 1] Updating characteristic 1.10 ("On") with boolean true
>>> HomeKit: [Client 1] Update Characteristics
>>> HomeKit: [Client 1] Updating characteristic 1.10 ("On") with boolean false
>>> HomeKit: [Client 1] Update Characteristics
>>> HomeKit: [Client 1] Updating characteristic 1.10 ("On") with boolean true
```

TADA!


ANKER
No SIM
4:26 PM
Edit
My Home
Default Room HomeKit LED on.
Details >
Favorite Accessories
Default Ro...
HomeKit L...
On
Home
Rooms
Automation

YAY!

Getting Started with wolfSSL on the ESP32



wolfssl/IDE/Espressif at master

github.com/wolfSSL/wolfssl/tree/master/IDE/Espressif

Files

master

Go to file

- .github
- Docker
- IDE
  - ARDUINO
  - AURIX
  - Android
  - CRYPTOCELL
  - CSBENCH
  - ECLIPSE
  - Espressif
    - ESP-IDF
    - README.md

wolfssl / IDE / Espressif /

Top

Getting Started

If you are new to wolfSSL on the Espressif ESP32, this video can help to get started:

ESPRESSIF
Open Source
Internet Security
wolfSSL
GO Jimmy PI
Getting Started with wolfSSL on the Espressif ESP32
Bozeman, MT : Seattle, WA : Portland, OR : Rescue, CA : Tokyo, JP : Brisbane, AU : Mobile, AL

www.youtube.com/watch?v=CzwA3ZBZBZ8

Getting Started with wolfSSL

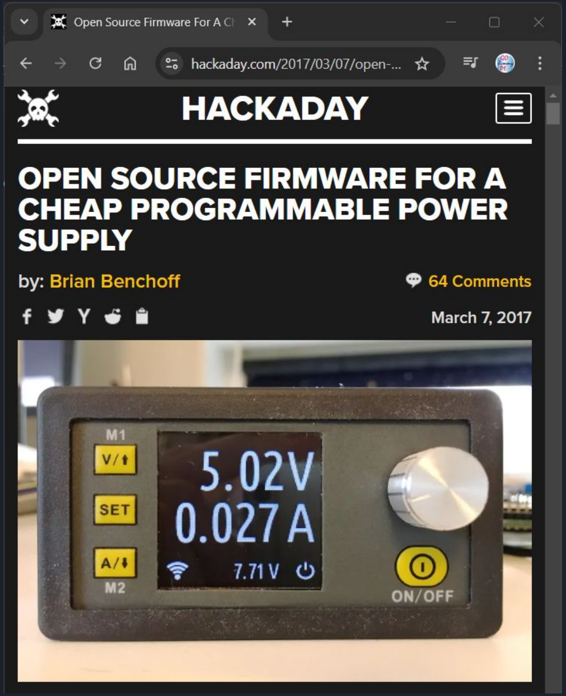
Best-Tested Commercial-Grade Cryptographic Libraries

TLS 1.3

Presented by Jim Scarletta

www.youtube.com/watch?v=04DGXkZ1IC4

Beyond the ESP32


Open Source Firmware For A Ch
hackaday.com/2017/03/07/open-...
HACKADAY
M1
V/⬆
SET
A/⬇
M2
5.02V
0.027 A
7.71 V
ON/OFF

OPEN SOURCE FIRMWARE FOR A CHEAP PROGRAMMABLE POWER SUPPLY

by: Brian Benchoff

64 Comments

March 7, 2017

See: gojimmypi.github.io/opendps-with-dps5015

github.com/kanflo/opendps

Thank you

Thank you

- Hackaday
- SupplyFrame
- wolfSSL

Standing on the Shoulders of Giants

Slides and information will be available:

gojimmypi.github.io/Hackaday-Supercon-2024

(case sensitive)

gojimmypi@gmail.com