

ESPurna



**Free Libre Open Source Firmware for
ESP8266-based Smart Devices**



Xose Pérez
@xoseperez
@xoseperez@mastodont.cat
xose@espurna.io
http://espurna.io
http://tinkerman.cat



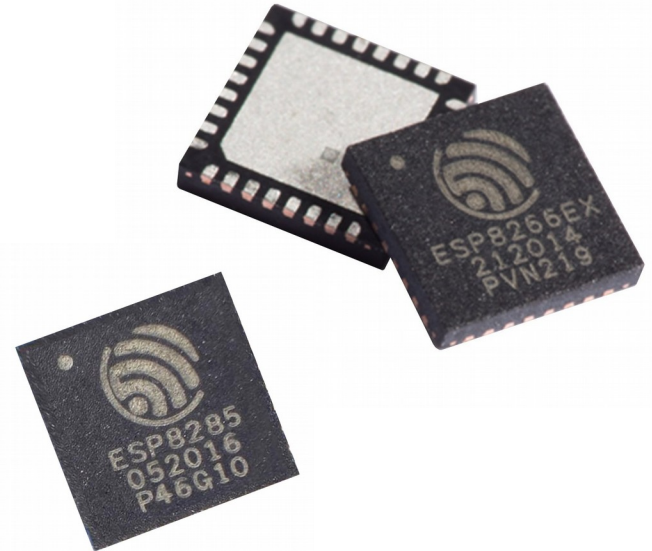
This work is licensed under a
Creative Commons Attribution-ShareAlike 4.0
International License

Introducción

ESP8266

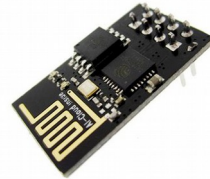
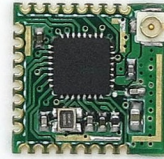
ESP8266 & ESP8285 de Espressif

- Salió a la luz en 2014
- Uso inicial como bridge wifi
- Microprocesador L106 32-bit RISC basado en el Tensilica Xtensa Diamond Standard 106 Micro
- Corriendo a 80 MHz
- 32KiB IRAM & RAM de caché
- 80KiB RAM de usuario
- Flash:
 - ESP8266: chip externo QSPI flash hasta 16 MiB
 - ESP8285: 1MiB de flash embebida
- IEEE 802.11 b/g/n Wi-Fi (solo 2.4GHz, WEP & WPA/WPA2)
- 16 GPIO, SPI, I2C (software) y I2S con DMA
- UART en pins dedicados (y un UART TX extra)
- ADC de 10-bits



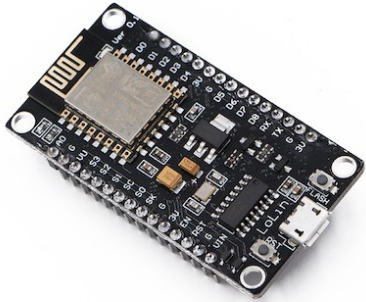
Módulos

Ya que en el ESP8266 no incluye la memoria flash (donde se guarda el código a ejecutar) y requiere de unos pocos pasivos para funcionar correctamente, hay muchos módulos que lo integran con diferentes factores de forma y GPIOs expuestos.



Placas de desarrollo

También han surgido un número de placas de desarrollo con un ESP8266 y conexión directa via USB o hardware específico.



ESP8266 inside

Pero el momento de inflexión se produjo cuando diferentes fabricantes (chinos) empezaron a usar el ESP8266 en sus “productos inteligentes”.



De momento todo bien, pero...

La mayoría de los productos de consumo están basados en soluciones “end to end”. El fabricante te vende su producto, que se conecta a su nube y se controla con su app. Esto significa que:

- Puedes acabar necesitando diferentes apps para diferentes dispositivos, a no ser que lo compres todo al mismo fabricante (ecosistemas aislados o único)
- Diferentes dispositivos de diferentes fabricantes no interoperan (ecosistema cerrado)
- Si una mariposa bate sus alas cerca de un servidor en Shenzhen, puede que no puedas encender las luces de casa...
- Los fabricantes pueden estar haciendo un gran trabajo protegiendo tus datos, pero no están obligados por nuestras leyes
- La seguridad no es prioritaria

En definitiva, puedes acabar de comprar un moderno dispositivo IoT, pero no sabes qué información genera ni cómo la gestiona. Por lo tanto, en realidad no es tuyo.

Soberanía Tecnológica

Soberanía tecnológica

“Tecnología es todo el conjunto de conocimiento técnico que nos ayuda a adaptarnos a nuestro entorno. Al principio la tecnología era un bien público, pero ahora está controlada por las leyes del mercado. Somos sujetos pasivos, meros consumidores de productos y servicios.

(...)

Es muy importante que nos empoderemos tecnológicamente, para cubrir nuestras necesidades y proteger el medio ambiente. Ser soberanos tecnológicamente significa tener los conocimientos y las capacidades para decidir sobre cómo usamos la tecnología. Y esto significa luchar contra aquellos que intentan controlarnos a través de la tecnología.” (*)

Esto, que quizá suene muy radical/político/..., pero no es otra cosa que

lo que hacemos los makers.

Soberanía tecnológica – Soft & Hard

Cuando hablamos de “tecnología digital”, tenemos muy presentes proyectos como la Free Software Foundation (1985) que impulsa “liberar” el código del software que usamos. También hay proyectos en el mundo del hardware que trabajan impulsando el hardware libre (OSHW, Arduino) o el silicio libre (RISC-V, FPGAWars,...).



Soberanía tecnológica - Redes

Y los datos se mueven a través de cañerías virtuales. Conotrolar esas cañerías también significa controlar la información que las usa, privilegiando unos datos sobre otros o simplemente bloqueando aquellos que no interesan.

Por eso también se ven proyectos de redes de telecomunicaciones libres como Guifi.net o The Things Network. El manifiesto de TTN explícitamente dice: “Controlar la red (...) significa controlar el mundo. Nosotros creemos que esta capacidad no debería estar restringida a unas pocas personas, empresas o países. En lugar de esto, debería ser distribuída a la mayor cantidad de gente posible, sin que nadie pueda quitárnoslo.”

guifi·net



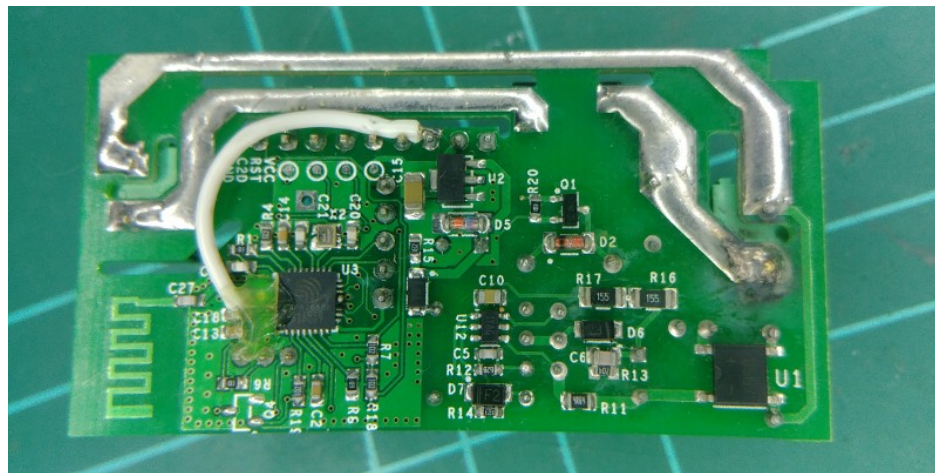
ESPurna

libre firmware

**Mirando hacia
el pasado...**

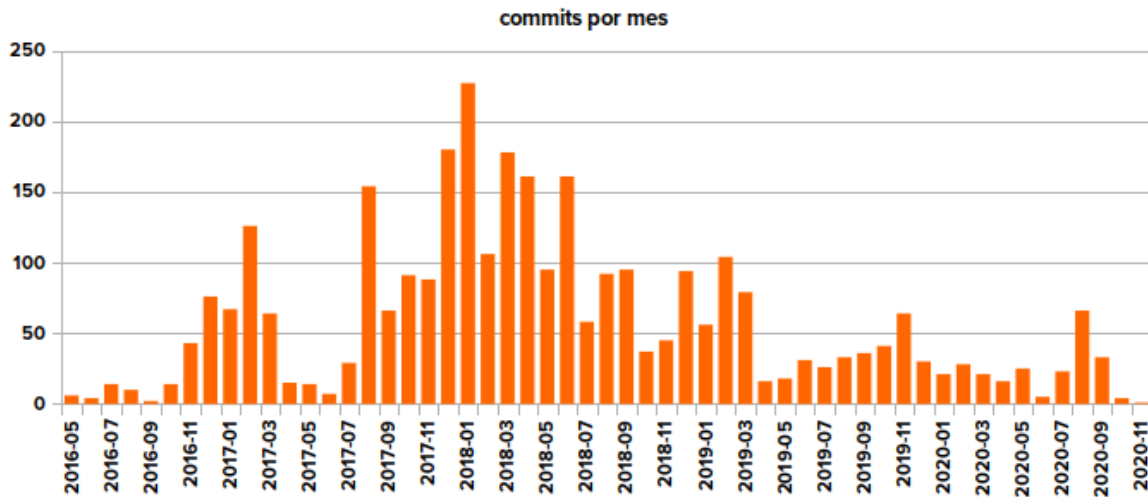
Historia

- El proyecto nace a principios de 2016 y recibe el nombre de ESPurna
- El primer commit (en Bitbucket) es del 20 de Mayo de 2016
 - Soporte para Sonoff Basic
 - Interfaz web muy simple
 - Botón físico
 - MQTT
- Se migra a GitHub el 24 de Enero de 2018
- Mantenido por Max Prokhorov desde Noviembre de 2019
- GPLv3



Métricas

- 76946 líneas de código
- 3196 commits
- 170 autores diferentes
- 60 releases
- ~145k descargas de binarios
- 2346 estrellas
- 558 forks
- ~1700 issues
- 586 PRs incorporados

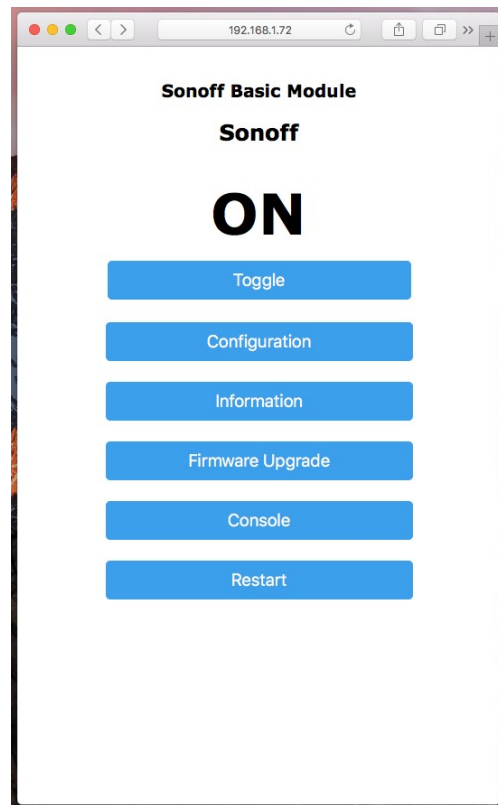


Datos a 2020-11-13 github.com / gitstats

**¿Solo puede
quedar uno?**

Sonoff-Tasmota

- <https://github.com/arendst/Sonoff-Tasmota>
- Activo desde Enero de 2016 (como Sonoff-MQTT-OTA)
- Mantenido por Theo Arends
- 1181094 líneas de código
- 9135 commits
- 257 colaboradores
- 83 releases
- >3M descargas de binarios
- >13400 estrellas
- >2900 forks
- >7200 issues
- >1900 PRs incorporados
- GPLv3



ESP Easy

- <https://github.com/letscontrolit/ESPEasy>
- Activo desde Mayo de 2015
- Mantenido por Let's Control It
- 592349 líneas de código
- 5779 commits
- 174 colaboradores
- 347 releases
- ~2300 estrellas
- ~1700 forks
- ~1800 issues
- 1248 PRs incorporados
- GPLv3

Welcome to ESP Easy : COM9

[Main](#) [Config](#) [Hardware](#) [Devices](#) [Tools](#)

<	>	Task	Device	Name	Port	IDX/Variable	GPIO	Values
Edit		1	Temperature DS18B20			240	GPIO-12	Temperature: 20.75
Edit		2	Temp + Hum DHT			237	GPIO-13	Temperature: 19.00 Humidity: 21.00
Edit		3	Temp + Baro BMP085			243	GPIO-4 GPIO-5	Temperature: 21.91 Pressure: 1020.62
Edit		4	LUX BH1750			244	GPIO-4 GPIO-5	Lux: 330.83

Task Settings	Value
Device:	<input type="text" value="Temp + Hum DHT"/> ?
Name:	<input type="text"/>
IDX / Var:	<input type="text" value="237"/>
1st GPIO:	<input type="text" value="GPIO-13"/>
DHT Type:	<input type="text" value="DHT 11"/>

Optional Settings	Value
Formula Temperature:	<input type="text"/> ?
Formula Humidity:	<input type="text"/>
Value Name 1:	<input type="text" value="Temperature"/>
Value Name 2:	<input type="text" value="Humidity"/>

[Close](#) [Submit](#)

Powered by www.esp8266.nu

Otras opciones

- ESPruino (<https://github.com/espruino/Espruino>)
- Souliss (<https://github.com/souliss/souliss>)
- Sonoff WiFi Switch (https://github.com/tretyakovsa/Sonoff_WiFi_switch)
- Mongoose OS Smart Light (<https://github.com/cesanta/mongoose-os-smart-light>)
- AiLight (<https://github.com/stelgenhof/AiLight>)
- KmanSonoff (<https://github.com/KmanOz/KmanSonoff>)
- OpenMQTTGateway (<https://github.com/1technophile/OpenMQTTGateway>)
- Open-Home-Automation (<https://github.com/mertenats/Open-Home-Automation>)

**Dispositivos
compatibles**

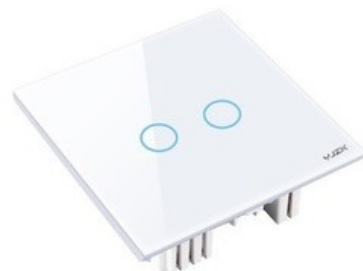
Monitorización de consumo eléctrico



Enchufes “inteligentes”



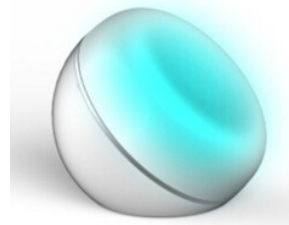
Interruptores



Dispositivos embebidos



Bombillas y lámparas



Drivers LED



Curiosidades



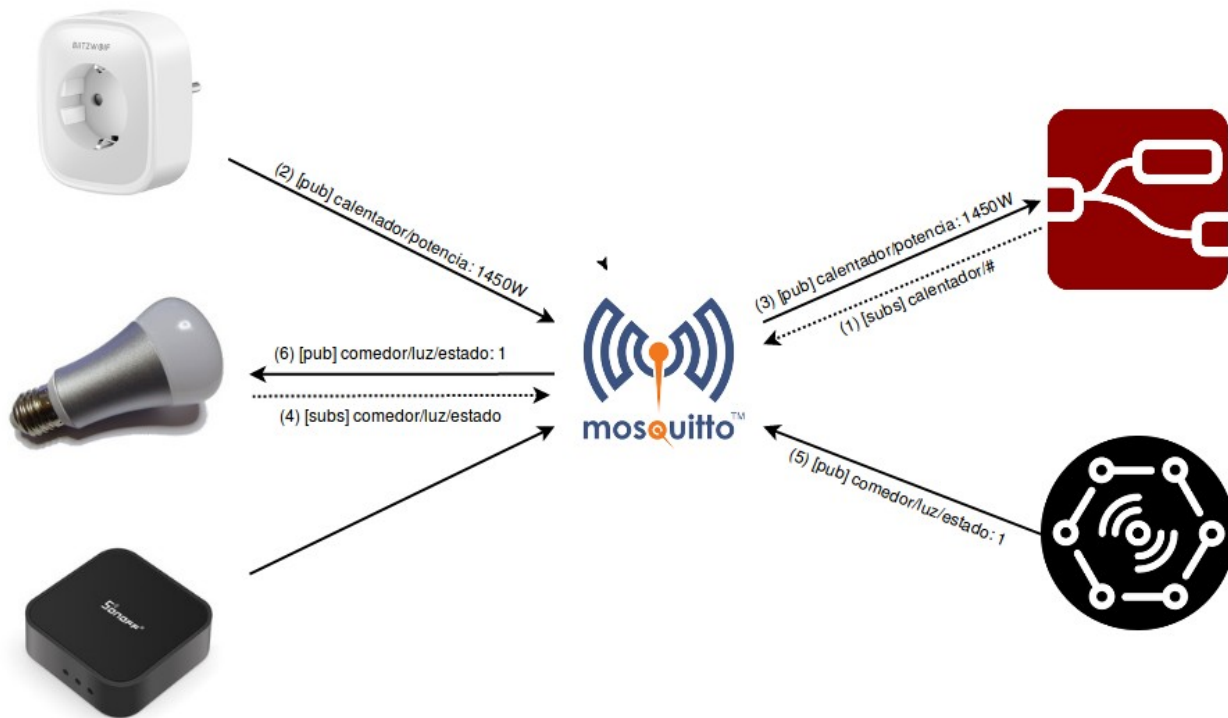
En total...

186

**dispositivos diferentes
(y subiendo)**

Funcionalidades

Comunicaciones - MQTT



Comunicaciones - Interfaz web

HEATER
ESPURNA 1.13.5-dev

STATUS

GENERAL

DOMOTICZ

HASS

MQTT

NTP

SCHEDULE

SENSORS

SWITCHES

THINGSPEAK

WIFI

ADMIN

DEBUG

Save

Reconnect

Reboot

© 2016-2019
Xosé Pérez

STATUS

Current configuration

Switch #0 OFF ON

Current #0 HLW8012 @ GPIO(12.5,14)

Voltage #0 HLW8012 @ GPIO(12.5,14)

Active Power #0 HLW8012 @ GPIO(12.5,14)

Reactive Power #0 HLW8012 @ GPIO(12.5,14)

Apparent Power #0 HLW8012 @ GPIO(12.5,14)

Power Factor #0 HLW8012 @ GPIO(12.5,14)

Energy #0 HLW8012 @ GPIO(12.5,14) (since 2018-09-01 06:37:25)

Manufacturer	BLITZWOLF	Network	daolz
Device	BW5HPX	BSSID	██████████
Chip ID	22EFF1	Channel	6
Wifi MAC	██████████	RSSI	-65
SDK version	1.5.3(aec24ac9)	IP	192.168.1.193 (telnet)
Core version	2.3.0	Free heap	19384 bytes
Firmware name	ESPURNA	Load average	3%
Firmware version	1.13.5-dev	VCC	3148mV
Firmware revision		MOTT Status	CONNECTED

LIVINGLAMPDOWN
ESPURNA 1.12.7a

STATUS

GENERAL

DOMOTICZ

HASS

LIGHTS

MQTT

NTP

SCHEDULE

SWITCHES

THINGSPEAK

WIFI

ADMIN

DEBUG

Save

Reconnect


Reboot

© 2016-2018
Xosé Pérez

STATUS

Current configuration

Switch #0 ON OFF

Color 

Channel #3

Manufacturer	AITHINKER	Network	daolz
Device	AI_LIGHT	BSSID	██████████
Chip ID	0F2216	Channel	6
Wifi MAC	██████████	RSSI	-50
SDK version	1.5.3(aec24ac9)	IP	192.168.1.194 (telnet)
Core version	2.3.0	Free heap	9999 bytes
Firmware name	ESPURNA	Load average	1%
Firmware version	1.12.7a	VCC	3207mV
Firmware build date	2018-06-11 23:10:34	MQTT Status	CONNECTED
Firmware size	520560 bytes	NTP Status	SYNC'D
Free space	503808 bytes	Current time	2019-03-23 22:30:14

Comunicaciones - Telnet

```
$ telnet 192.168.1.193
Trying 192.168.1.193...
Connected to 192.168.1.193.
Escape character is '^'.
Password: *****
Welcome!
help
[412125] Available commands:
[412126] > COMMANDS
[412126] > CONFIG
[412127] > CRASH
[412127] > DEL
[412127] > DICTIONARIES
[412127] > EEPROM
[412127] > EEPROM.COMMIT
[412129] > EEPROM.DUMP
[412131] > ERASE.CONFIG
[412133] > FACTORY.RESET
[412136] > FLASH.DUMP
[412137] > GET
[412139] > GPTO
[412140] > HA.CLEAR
[412142] > HA.CONFIG
[412143] > HA.SEND
[412145] > HEAP
[412146] > HELP
[412148] > INFO
```

```
[412149] > KEYS
[412150] > MAGNITUDES
[412153] > MQTT.RESET
[412154] > OTA
[412155] > PUBLISH
[412157] > RELAY
[412159] > RELOAD
[412160] > RESET
[412162] > RESET.SAFE
[412164] > SELECT
[412166] > SET
[412166] > STACK
[412168] > SUBSCRIBE
[412170] > UNSUBSCRIBE
[412172] > UPTIME
[412174] > WIFI
[412175] > WIFI.AP
[412176] > WIFI.RESET
[412178] > WIFI.SCAN
[412181] +OK
```

Comunicaciones – API REST

```
$ curl "http://192.168.1.108/api?apikey=C62ED7BE7593B658"
relay0 -> /api/relay/0
relay1 -> /api/relay/1
temperature -> /api/temperature
humidity -> /api/humidity

$ curl http://192.168.1.108/api/relay/0?apikey=C62ED7BE7593B658
Relay0 => 1

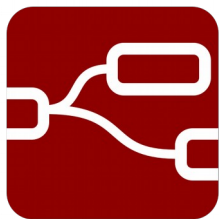
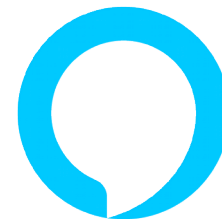
$ curl -H "Accept: application/json" http://192.168.1.108/api/temperature?apikey=C62ED7BE7593B658
{ "temperature": 21.5 }

$ curl -X PUT -H "Accept: application/json" http://192.168.1.108/api/relay/0 --data "apikey=C62ED7BE7593B658&value=2"
{ "relay0": 0 }

$ curl "http://192.168.1.108/api/relay/0?apikey=C62ED7BE7593B658&value=2"
Relay0 => 0

$ curl -X PUT -H "Accept: application/json" http://192.168.1.109/api/rgb --data "apikey=E45FFE7593658012&value=%23FF0000"
{ "rgb": "#FF0000" }
```

Integraciones



Sensores

- Temperatura (DHT, DALLAS, AM2320, SI7021, BMP280, BME280, BME680, TMP35, TMP36, MAX6675, NTC, SHT3X)
- Humedad (DHT, BME280, BME680, AM2320, SI7021, SHT3X)
- Presión atmosférica (BMP180, BMP280, BME680, BMD280)
- CO2 (T6613, MHZ19, MICS2710, MICS5525, SenseAir S8)
- NO2 (MICS2710)
- Polvo (PMSX003, PMS5003T, SDS011)
- Luz (BH1750)
- UV (GUVAS12SD, VEML6075, SI1145)
- PH (EZO pH)
- Distancia (HC-SR04, SRF0X, DYP-ME007, Parallax PING, VL53L1X)
- Consumo (ADE7953, HLW8012, CSE7766, CSE7759B, HJL-01, BL0937, V9261F, ECH1560, PZEM004T, PZEM004Tv3, CT + ADC, Pulsos)
- Básicos (Digital, Analógico, Contador, Eventos)
- Serie (lectores códigos de barras, QR, ...)

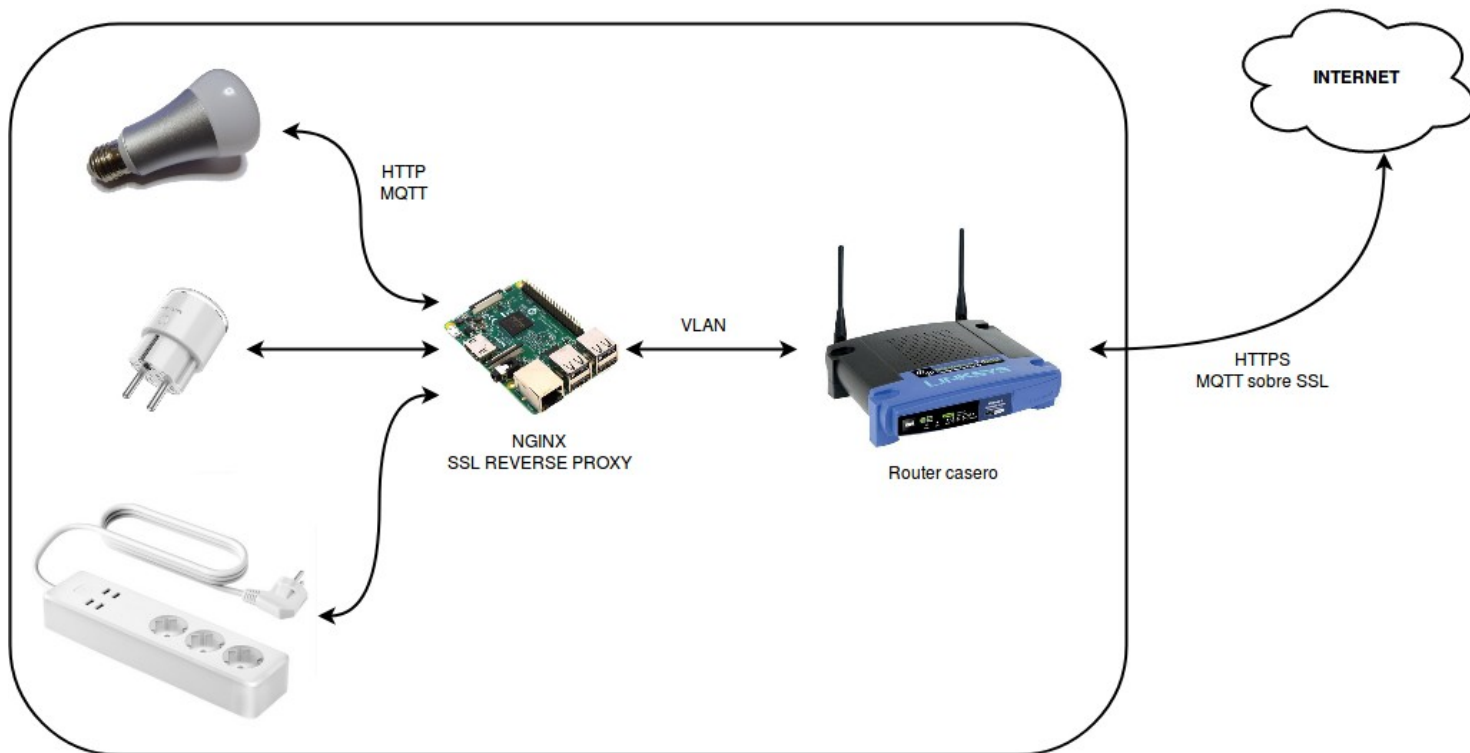
Otras

- **OTA (actualización del firmware over-the-air):**
 - Usando el IDE de Arduino o PlatformIO
 - Usando el ESPurna OTA Manager
 - Usando NoFUSS
 - Usando el interfaz web (sección ADMIN)
- **Scheduler (ejecutar acciones en momentos dados)**
- **Actualización de la hora del dispositivo via NTP**
- **Configuración y sincronización de interruptores (local y remota via MQTT)**
- **Configuración de múltiples redes WiFi (se conecta a la de mejor señal)**

Seguridad

- El ESP8266 no tiene un sistema de encriptación por hardware
- Se puede implementar SSL fingerprinting por software
- Pero consume mucha memoria y limita otras funciones
- ESPurna soporta:
 - MQTT sobre SSL
 - OTA sobre SSL
 - Integración con Thingspeak sobre SSL
- Y, por supuesto, WiFi con WPA2

Seguridad



Manos a la masa

Introducción

Repositorio

<http://espurna.io>

<https://github.com/xoseperez/espurna>

The screenshot shows the GitHub repository page for `xoseperez/espurna`. At the top, there is a search bar and navigation links for Pull requests, Issues, Marketplace, and Explore. The repository name is `xoseperez/espurna`, with 131 unwatched items, 1,407 stars, and 317 forks. Below this, there are tabs for Code, Issues (271), Pull requests (40), Projects (2), Wiki, Insights, and Settings. The repository description is "Home automation firmware for ESP8266-based devices" with a link to `http://tinkerman.cat`. It has 2,602 commits, 17 branches, 59 releases, and 94 contributors. The license is GPL-3.0. There are buttons for "New pull request", "Create new file", "Upload files", "Find File", and "Clone or download". A commit by `xoseperez` is highlighted, titled "Allow to configure all LEDs from UI (#1429)". Below the commit, there are file listings for `.github`, `code`, `images`, and `RF`. The `README.md` file is expanded, showing the title "ESPurna Firmware" and a description: "ESPurna ('spark' in Catalan) is a custom firmware for ESP8285/ESP8266 based smart switches, lights and sensors. It uses the Arduino Core for ESP8266 framework and a number of 3rd party libraries." At the bottom, there are badges for version (1.13.6-dev), branch (dev), license (GPL-3.0), build (passing), code quality (A), and downloads (67k total). There are also links for donate (PayPal), chat (on Gitter), and follow (@xoseperez).

Arduino Core para ESP8266

- *Port* del *framework* Arduino para ESP8266
- Te permite programar un ESP8266 como si fuera una placa Arduino
- Incluso desde el IDE de Arduino
- En la mayoría de los casos usando las mismas APIs
- Añade funcionalidades propias del hardware

- Pero cuidado, por que aunque se programa igual, tiene sus peculiaridades...

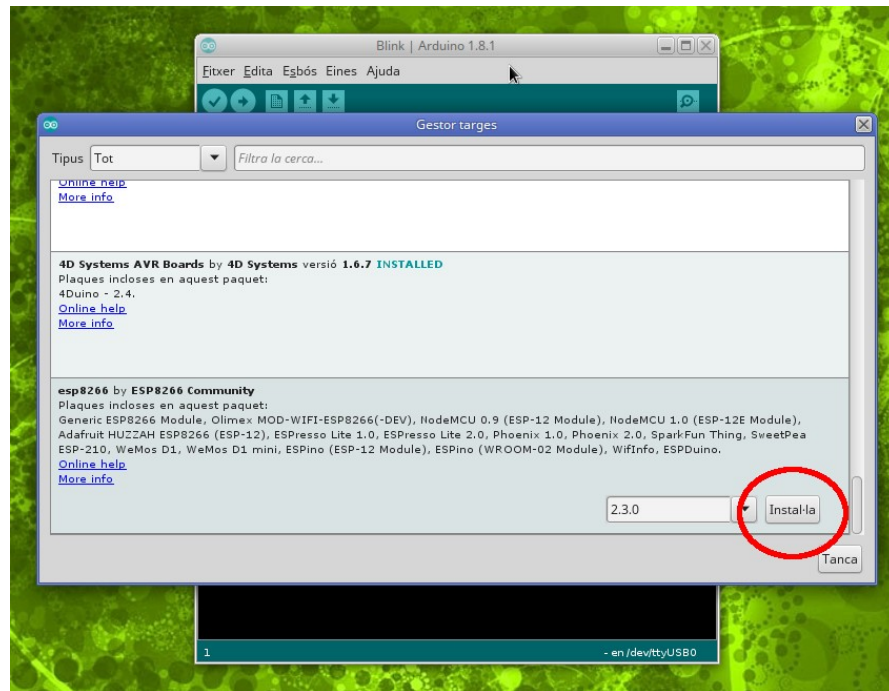
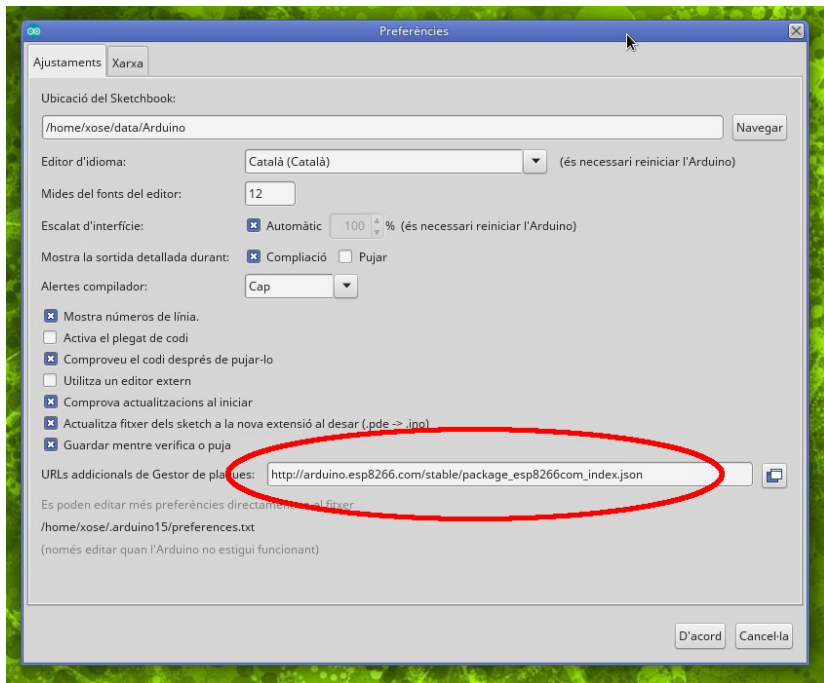
Dependencias - Core

- **ArduinoJson (5.13.4)**
- **Async-MQTT-Client (0.8.1)**
- **DebounceEvent (2.0.5)**
- **EEPROM_Rotate (0.9.2)**
- **Embedis**
- **ESPAsyncTCP (#55cd520)**
- **ESPAsyncWebServer (#05306e4)**
- **FauxmoESP (3.1.0)**
- **JustWiFi (2.0.2)**
- **NTPClient (fork #0942ebc)**
- **Time (fork)**

Dependencias - Opcionales

- Brzo I2C
- Encoder
- ESPSoftwareSerial (3.4.1)
- HLW8012 (1.1.0)
- IRremoteESP8266 (2.2.0)
- mDNSResolver (#4cfcda1)
- my92xx (3.0.1)
- NoFUSS (0.2.5)
- OneWire
- PZEM004T
- PubSubClient
- rc-switch
- RFM69 (1.1.3)
- NewPing
- SparkFun_VEML6075_Arduino_Library (1.0.3)
- vl53l1x-arduino (1.0.1)
- MAX6675-Library (2.0.1)

Arduino IDE



PlatformIO

<https://platformio.org>



Professional collaborative platform for embedded development

A place where Developers and Teams have true Freedom! No more vendor lock-in!



40

Platforms



23

Frameworks



913

Boards



222

Examples



10,546

Libraries

Cross-platform PlatformIO IDE and Unified Debugger · Static Code Analyzer and Remote Unit Testing
Multi-platform and Multi-architecture Build System · Firmware File Explorer and Memory Inspection



Install PlatformIO Now




[🔗 Release Notes](#) · [🔗 Open Source](#) · [🐦 Twitter](#) · [🌐 LinkedIn](#) · [📘 Facebook](#) · [📦 Bintray](#) · [❤️ Donate](#)

PlatformIO - ¿Porqué?

- Multi-OS (escrito en python) y Open Source
- Multiplataforma (30 plataformas diferentes)
- Multientorno (diferentes configuraciones conviviendo en un mismo proyecto)
- Gestión automatizada del “*toolchain*”
- Gestión automatizada de las dependencias
- Muy configurable (hooks a nivel usuario)
- Buena integración con Travis
- CLI o integraciones nativas en VSCode, Atom, Cloud9, Eclipse, Sublime, Emacs, VIM...
- PIO Plus con ejecución remota (entre otras cosas)

PlatformIO - ¿Porqué?

Comparison of ESP8266 Project Build Times

Time to Build Project by IDE	 Arduino IDE 1.6.8	 PlatformIO IDE 1.1.1	 Arduino Eclipse Plugin V3
Entire Project	24s	15s	28s
Single File Changed	8s	3s	3s

Tested Sketch: WiFiWebServer.ino from <https://github.com/esp8266/Arduino> , core 2.1.0

Test PC: Dell OptiPlex 780, Intel Core 2 Duo E8400 @3GHz, 12GB RAM, SSD, Windows 7 x64

Compilando

Requisitos

- **GIT 2.X**
- **Opción A**
 - MS Visual Studio Code
 - Extensión PlatformIO IDE para VSCode
- **Opcion B**
 - Atom IDE
 - Extensión PlatformIO IDE para Atom
- **Opcion C**
 - Python 3.6 & PIP
 - PlatformIO Core 4.X o 5.X
- **Node JS 6.X o superior (opcional, requerido para modificar el interfaz web)**
- **Driver para el programador o placa que usemos (depende del OS)**

GIT

Usa siempre un sistema de gestión de versiones, incluso para proyectos personales, incluso aunque no lo subas a ningún repositorio público, aunque no vaya a salir de tu ordenador. Siempre.

<https://git-scm.com>



git --distributed-even-if-your-workflow-isnt

Search entire site...

Git is a **free and open source** distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is **easy to learn** and has a **tiny footprint with lightning fast performance**. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like **cheap local branching**, convenient **staging areas**, and **multiple workflows**.



About

The advantages of Git compared to other source control systems.



Documentation

Command reference pages, Pro Git book content, videos and other material.



Downloads

GUI clients and binary releases for all major platforms.



Community

Get involved! Bug reporting, mailing list, chat, development and more.



Pro Git by Scott Chacon and Ben Straub is available to [read online for free](#). Dead tree versions are available on [Amazon.com](#).



Linux GUIs



Tarballs



Mac Build



Source Code

Visual Studio Code

The screenshot displays the Visual Studio Code interface with the PlatformIO IDE extension page open. The left sidebar shows the Extensions view with a search bar and a list of installed and recommended extensions. The main panel shows the details for the PlatformIO IDE extension, including its logo, name, version, and a description of its capabilities as an IoT development environment.

EXTENSIONS

Search Extensions in Marketplace

ENABLED 17

- jslint** 0.10.20 | 1.5M | 4.5
Integrates JSHint into VS Code. JSHint is a li...
Dirk Baeumer
- LaTeX Workshop** 6.2.2 | 2.7M | 4.5
Boost LaTeX typesetting efficiency with pre...
James Yu
- Markdown All in One** 2.2.0 | 2.7M | 5
All you need to write Markdown (keyboard s...
Yu Zhang
- PHP Intellisense** 2.3.10 | 6.3M | 3.5
Advanced Autocompletion and Refactoring ...
Felix Becker
- PlatformIO IDE** 1.6.0 | 826K | 5
Development environment for IoT, Arduino, ...
PlatformIO

RECOMMENDED 4

- markdownlint** 0.25.1 | 3.9M | 4.5
Markdown linting and style checking for Vis...
David Anson [Install](#)
- XML Tools** 2.4.0 | 2.7M | 4
XML Formatting, XQuery, and XPath Tools F...
Josh Johnson [Install](#)
- npm Intellisense** 1.3.0 | 1.3M | 4.5
Visual Studio Code plugin that autocomplet...
Christian Kohler [Install](#)
- npm** 0.3.5 | 2.8M | 3.5
npm support for VS Code
egamma [Install](#)

DISABLED 0

Extension: PlatformIO IDE

PlatformIO IDE platformio.platformio-ide

PlatformIO | 826,448 | ★★★★★ | Repository | License

Development environment for IoT, Arduino, ARM mbed, Espressif (ESP8266/ESP32), RISC-V, STM32, ...

[Disable](#) [Uninstall](#)

[Details](#) [Contributions](#) [Changelog](#) [Dependencies](#)

PlatformIO IDE for VSCode

The next generation integrated development environment for IoT

PlatformIO is an open source ecosystem for IoT development. Cross-platform build system and unified debugger. Remote unit testing and firmware updates.

Platforms: Atmel AVR, Atmel SAM, Espressif 32, Espressif 8266, Freescale Kinetis, Infineon XMC, Intel ARC32, Intel MCS-51 (8051), Lattice iCE40, Maxim 32, Microchip PIC32, Nordic nRF51, Nordic nRF52, NXP LPC, RISC-V, Samsung ARTIK, Silicon Labs EFM32, ST STM32, ST STM8, Teensy, TI MSP430, TI Tiva, WIZNet W7500

Frameworks: Arduino, ARTIK SDK, CMSIS, Energia, ESP-IDF, ESP8266 RTOS SDK, Freedom E SDK, libOpenCM3, mbed, PULP OS, Pumbaa, Simba, SPL, STM32Cube, Tizen RT, WiringPi

Features

- Cross-platform code builder without external dependencies to a system software:
 - 600+ embedded boards
 - 30+ development platforms
 - 15+ frameworks

Command Line Interface - PlatformIO

Actualizamos repositorios:

```
$ sudo apt-get update
```

Instalamos GIT y las dependencias de PlatformIO:

```
$ sudo apt-get install git python3 python3-distutils python3-pip
```

PlatformIO tiene un script de instalación rápido para máquinas Linux, muy cómodo:

```
$ python3 -c "$(curl -fsSL https://raw.githubusercontent.com/platformio/platformio/develop/scripts/get-platformio.py)"
```

Por último, añadimos la ubicación donde se ha instalado al PATH del sistema y permisos necesarios:

```
$ echo "export PATH=~/.platformio/penv/bin" >> ~/.profile  
$ sudo adduser $USER dialout
```

Salimos y entramos de nuestro perfil para que el último cambio tenga efecto.

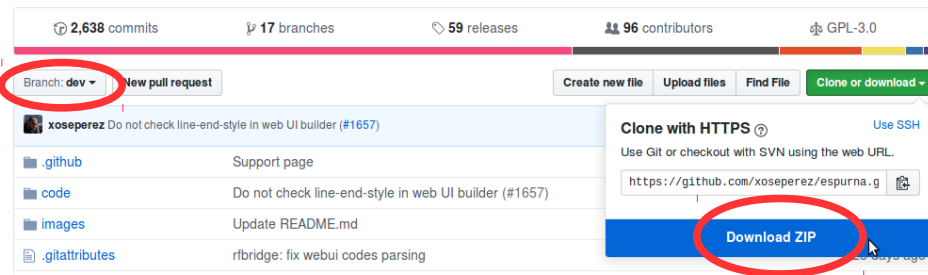
Command Line Interface – Web & Utils

Para “compilar” la web de ESPurna y usar algunas otras herramientas se necesita NodeJS y Gulp. El proceso de instalación dependerá de la plataforma, pero para nuestro Ubuntu haremos:

```
$ curl -sL https://deb.nodesource.com/setup_10.x | sudo -E bash -  
$ sudo apt-get install -y nodejs
```

Obteniendo el código

Opción A: Descargar el ZIP y descomprimir



The screenshot shows the GitHub interface for a repository. At the top, it displays statistics: 2,638 commits, 17 branches, 59 releases, 96 contributors, and GPL-3.0 license. Below this, there are buttons for 'Branch: dev', 'New pull request', 'Create new file', 'Upload files', 'Find File', and 'Clone or download'. The 'Clone or download' button is expanded, showing options for 'Clone with HTTPS' and 'Use SSH'. The 'Clone with HTTPS' option is selected, and the URL 'https://github.com/xoseperez/espurna.g' is visible. The 'Download ZIP' button is highlighted with a red circle.

Opción B: Usando GIT

```
$ git clone https://github.com/xoseperez/espurna
Cloning into 'espurna'...
remote: Enumerating objects: 329, done.
remote: Counting objects: 100% (329/329), done.
remote: Compressing objects: 100% (220/220), done.
remote: Total 21234 (delta 195), reused 190 (delta 109), pack-reused 20905
Receiving objects: 100% (21234/21234), 111.12 MiB | 593.00 KiB/s, done.
Resolving deltas: 100% (15748/15748), done.
Checking connectivity... done.
```

Ramas

Si descargas el ZIP del proyecto, estás descargando una rama concreta. La rama por defecto en GitHub es la rama de desarrollo (dev).

Cuando se hace una *release*, se mueve (*merge*) el código a la rama master y allí se etiqueta la *release* y Travis genera los binarios para cada dispositivo.

Si descargas el proyecto vía GIT puedes cambiar de rama fácilmente:

```
$ cd espurna
$ git checkout master
Switched to branch 'master'
Your branch is up-to-date with 'origin/master'.
```

Estructura del proyecto

```
$ tree -d
├── code
│   ├── espurna
│   │   ├── config
│   │   │   └── arduino.h
│   │   │   └── ...
│   │   ├── data
│   │   ├── filters
│   │   ├── libs
│   │   ├── sensors
│   │   ├── static
│   │   ├── espurna.ino
│   │   └── ...
│   ├── html
│   │   ├── images
│   │   ├── vendor
│   │   │   └── images
│   │   ├── custom.css
│   │   ├── custom.js
│   │   └── index.html
│   ├── platformio.ini
│   ├── ota.py
│   ├── .pioenvs
│   ├── .pio/libdeps
│   └── ...
├── images
├── LICENSE
├── pre-commit
├── README.md
└── SUPPORT.md
```

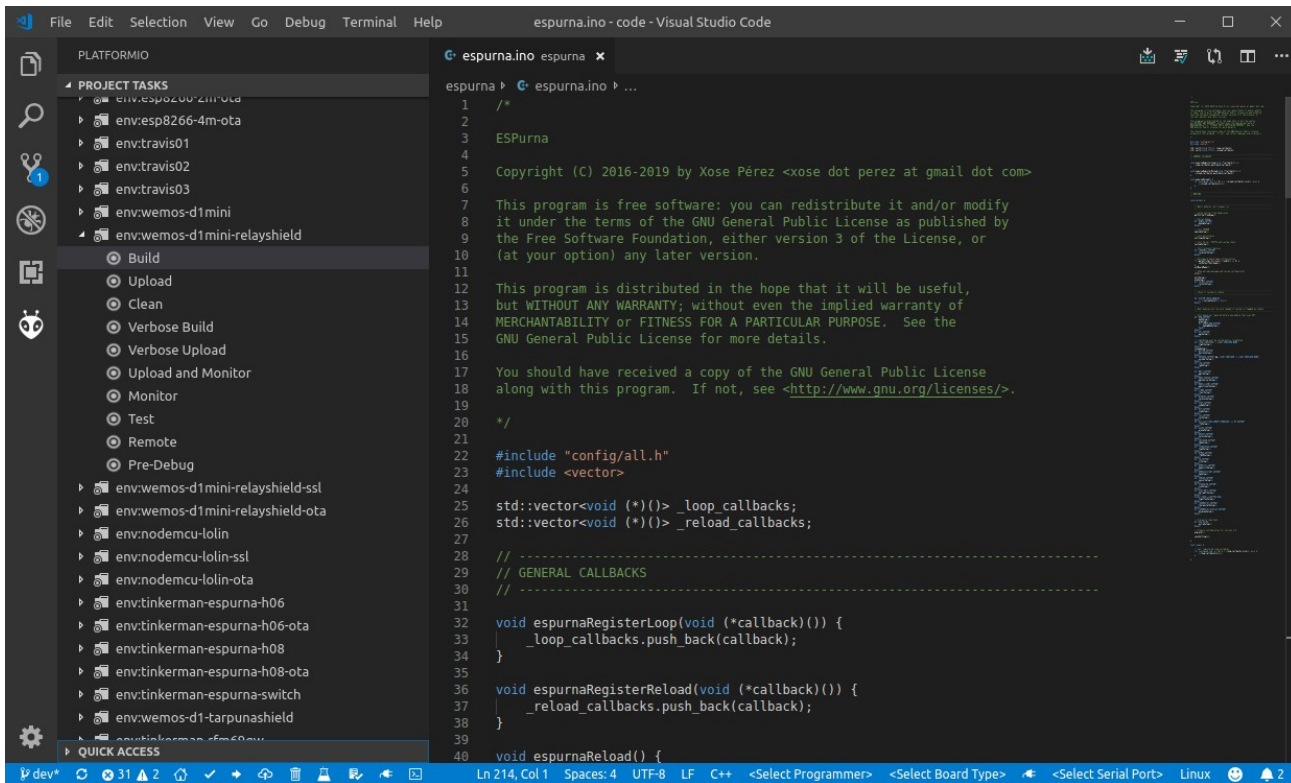
Para cargar el proyecto desde el IDE de Arduino abriremos el archivo `“/code/espurna/espurna.ino”`.

En `“/code/espurna/config/”` tenemos los archivos de configuración:

- `“arduino.h”` permite configurar fácilmente el proyecto cuando se usa el IDE de Arduino.
- `“general.h”` contiene la configuración por defecto de los diferentes módulos
- `“hardware.h”` contiene la definición y configuraciones específicas de los diferentes dispositivos soportados
- `“sensors.h”` contiene la configuración por defecto de los diferentes sensores soportados

Para compilar con PlatformIO lo haremos desde `“/code/”`. En `“/code/html/”` tenemos el interfaz web.

Compiler – VSCode



The screenshot shows the Visual Studio Code interface with the PlatformIO extension. The left sidebar displays the PlatformIO environment list, with 'env:wemos-d1mini-relayshield' selected. The main editor shows the 'espurna.ino' file with the following code:

```
1  /*
2
3  ESPurna
4
5  Copyright (C) 2016-2019 by Xose Pérez <xose dot perez at gmail dot com>
6
7  This program is free software: you can redistribute it and/or modify
8  it under the terms of the GNU General Public License as published by
9  the Free Software Foundation, either version 3 of the License, or
10 (at your option) any later version.
11
12 This program is distributed in the hope that it will be useful,
13 but WITHOUT ANY WARRANTY; without even the implied warranty of
14 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
15 GNU General Public License for more details.
16
17 You should have received a copy of the GNU General Public License
18 along with this program. If not, see <http://www.gnu.org/licenses/>.
19
20 */
21
22 #include "config/all.h"
23 #include <vector>
24
25 std::vector<void (*)()> _loop_callbacks;
26 std::vector<void (*)()> _reload_callbacks;
27
28 // -----
29 // GENERAL CALLBACKS
30 // -----
31
32 void espurnaRegisterLoop(void (*callback)()) {
33     _loop_callbacks.push_back(callback);
34 }
35
36 void espurnaRegisterReload(void (*callback)()) {
37     _reload_callbacks.push_back(callback);
38 }
39
40 void espurnaReload() {
```

Compilar – Línea de comandos

La raíz del código es la carpeta “/code/”

```
$ cd espurna/code
$ pio run -e wemos-dlmini-relayshield
...
$ ls -la .pioenvs/wemos-dlmini-relayshield/firmware.bin
-rw-rw-r-- 1 xose xose 475120 Mar 28 13:41 .pioenvs/wemos-dlmini-relayshield/firmware.bin
```

La primera vez PlatformIO ejecutará las siguientes tareas:

- Leer el archivo platformio.ini
- Analizar el código para construir el árbol de dependencias
- Crear un espurna.ino.cpp (prototipos, main,...)
- Instalar todas las dependencias (librerías)
- Instalar el toolchain (scons, framework, esptool, toolchain-xtensa,...)
- Compilar el código y generar un binario

Línea de comandos - Herramientas

Para compilar la web, desde la carpeta “/code/”

```
$ npm install --only=dev  
$ node node_modules/gulp/bin/gulp.js
```

OTA Manager permite ver qué dispositivos tienes en la red y actualizarlos remotamente:

```
$ pip3 install -r requirements.txt  
$ python3 ota.py
```

Build Manager se usa para automatizar los “builds” pero también lo podemos usar para saber qué entornos tenemos definidos:

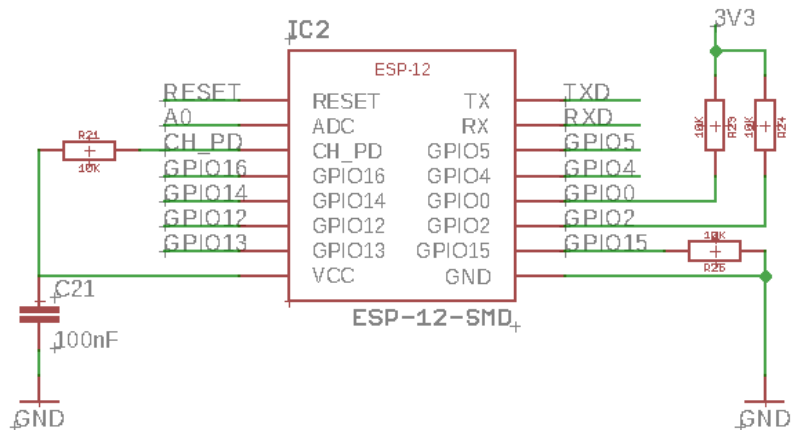
```
$ ./build.sh -l  
* aithinker-ai-light  
* allnet-4duino-iot-wlan-relais  
* ...
```

Problemas?

- Instala GIT con soporte para aplicaciones de terceros (solo aplica a Windows)
- Asegúrate de que tienes GIT instalado antes de arrancar el IDE o ejecutar “pio run”.
- Reinicia VSCode o Atom después de instalar la extensión de PlatformIO
- Asegúrate que la carpeta “/code/” es la raíz de tu proyecto en el IDE.
- En caso de errores desconocidos, aleatorios,.. borra las carpetas de caché que genera PlatformIO y vuelve a compilar el proyecto. La manera más efectiva es borrarlas a mano (carpetas “.pioenv” y “.piolibdeps” bajo “/code/”).

Subiendo

Serial bootloader (aka UART Mode)

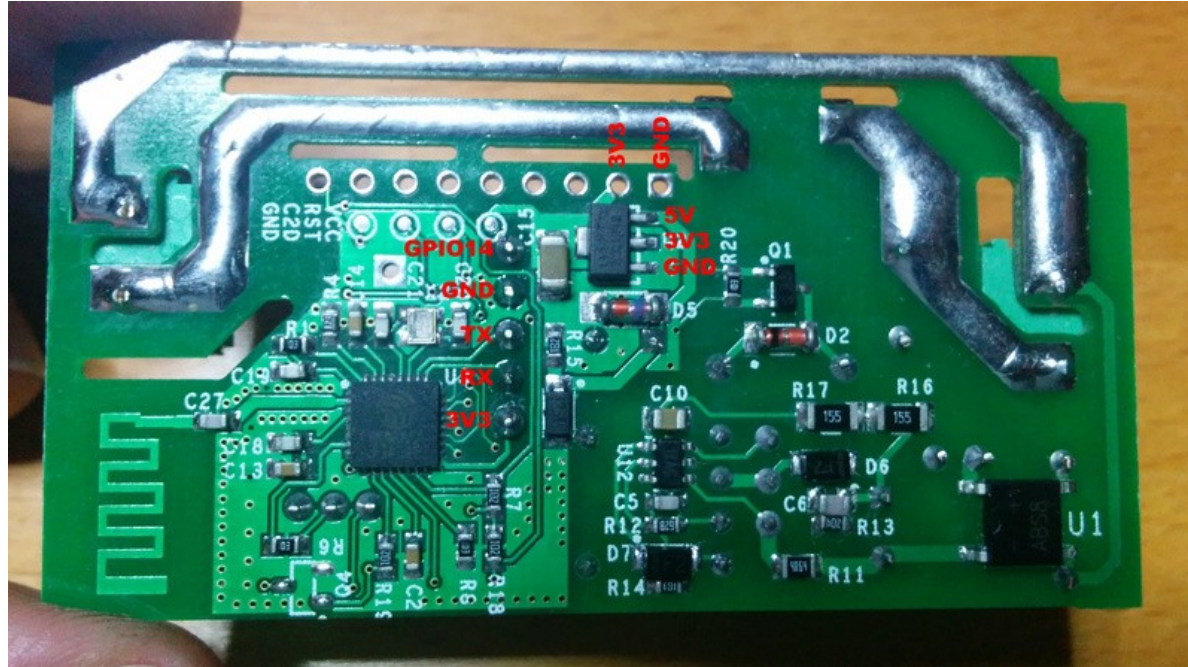


ESP8266 Boot Modes

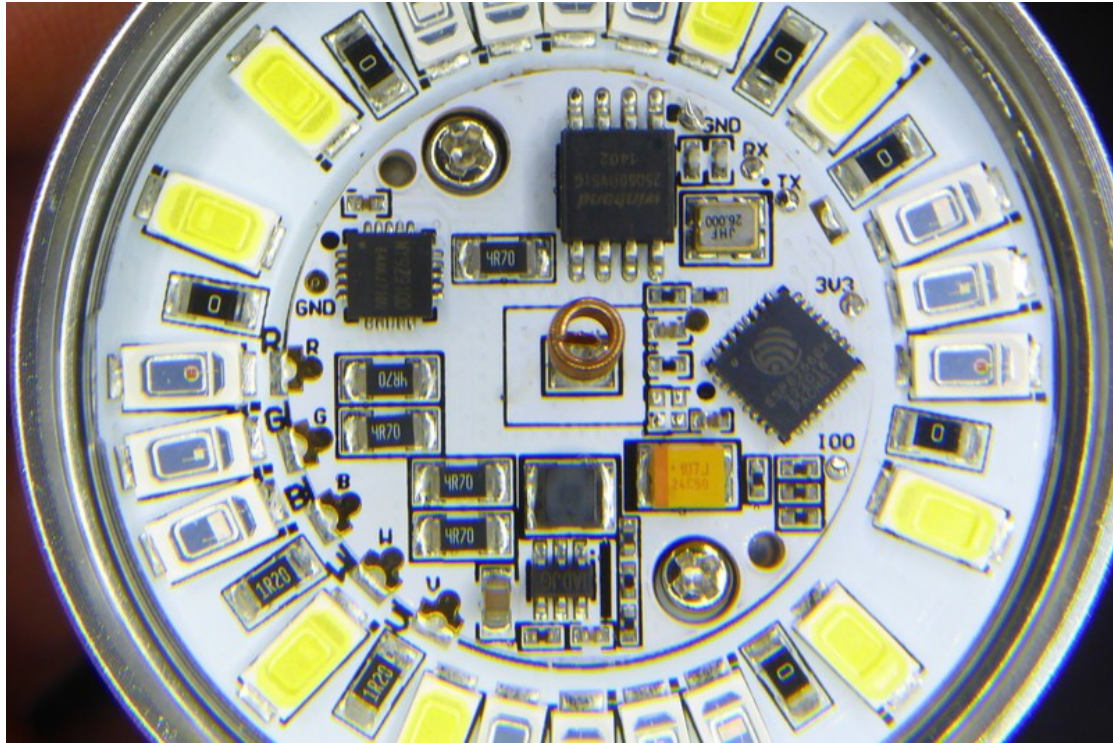
The Espressif code can boot in different modes, selected on power-up based on GPIO pin levels.

GPIO15	GPIO0	GPIO2	Mode	Description
L	L	H	UART	Download code from UART
L	H	H	Flash	Boot from SPI Flash
H	x	x	SDIO	Boot from SD-card

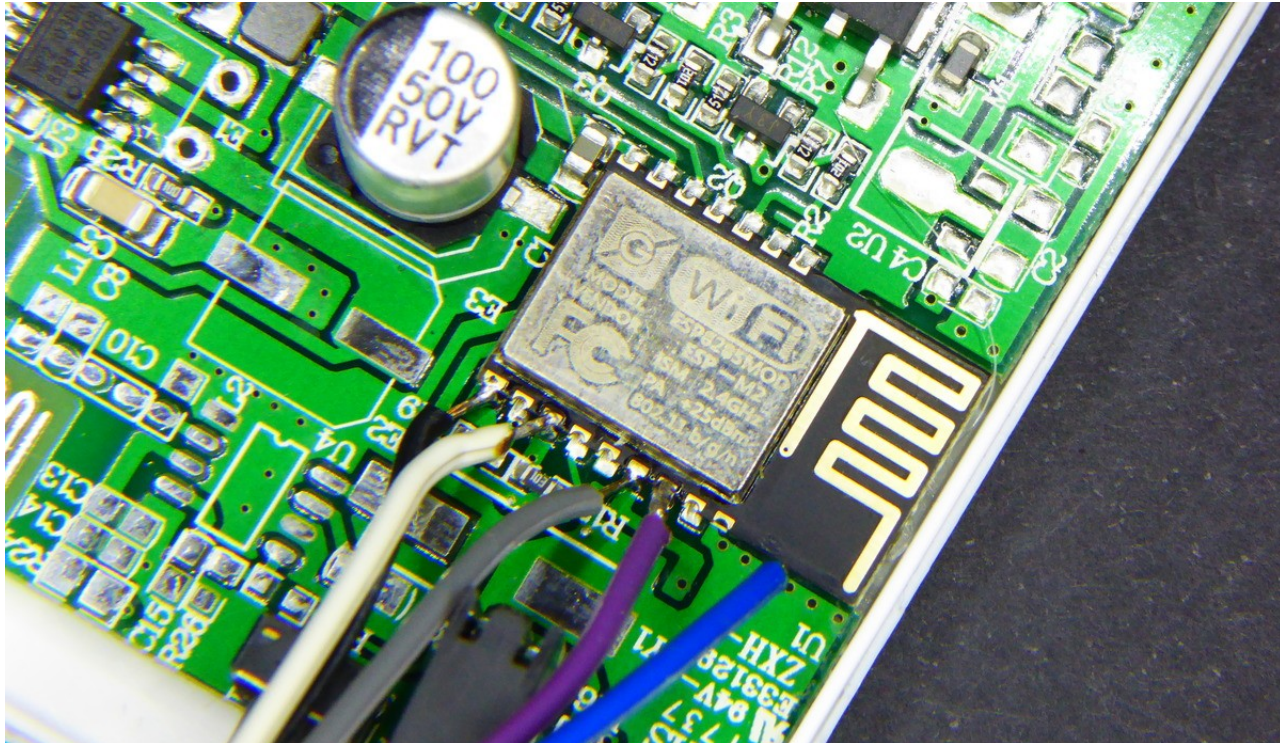
Serial bootloader (Sonoff Basic)



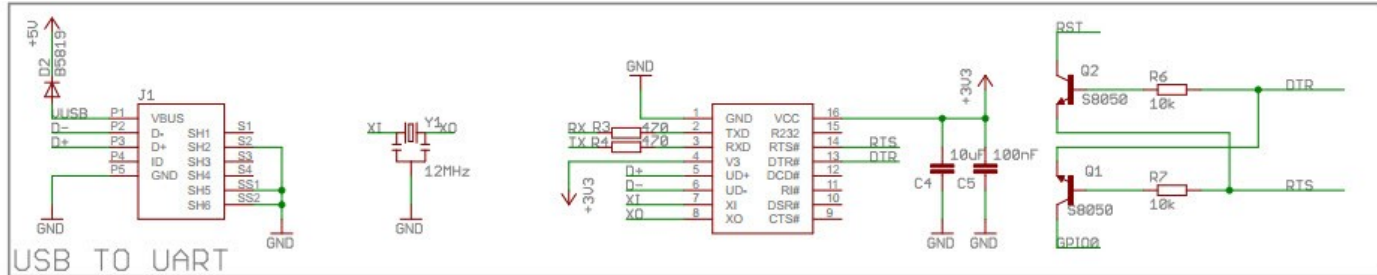
Serial bootloader (Ai Light)



Serial bootloader (Arilux AL-LC06)



Serial bootloader (Wemos D1 Mini)



PlatformIO – VSCode

The screenshot shows the Visual Studio Code interface with PlatformIO installed. The left sidebar displays the 'PROJECT TASKS' panel, where the 'Upload' task is selected. The main editor shows the source code for 'espurna.ino'. The bottom terminal window displays the output of the upload process, including progress bars for data and program upload, and a list of upload progress for various files.

```
espurna.ino espurna x
1 /*
2
3 ESPurna
4
5 Copyright (C) 2016-2019 by Xose Pérez <xose dot perez at gmail dot com>
6
7 This program is free software; you can redistribute it and/or modify
8 it under the terms of the GNU General Public License as published by
9 the Free Software Foundation, either version 3 of the License, or
10 (at your option) any later version.
11
12 This program is distributed in the hope that it will be useful,
13 but WITHOUT ANY WARRANTY; without even the implied warranty of
14 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
15 GNU General Public License for more details.
16
17 You should have received a copy of the GNU General Public License
18 along with this program. If not, see <http://www.gnu.org/licenses/>.
19
20 */
21
22 #include "config/all.h"
23 #include <vector>
24
25 std::vector<void (*)()> _loop_callbacks;
26 std::vector<void (*)()> _reload_callbacks;
27
```

```
3: Task - Upload (wemo)
DATA: [===== ] 52.8% (used 43220 bytes from 81920 bytes)
PROGRAM: [==== ] 11.2% (used 470975 bytes from 4194304 bytes)
Configuring upload protocol...
Looking for upload port...
Auto-detected: /dev/ttyUSB0
Uploading pioenvs/wemos-d1mini-relayshield/firmware.bin
Uploading 475120 bytes from pioenvs/wemos-d1mini-relayshield/firmware.bin to flash at 0x00000000
..... [ 17% ]
..... [ 34% ]
..... [ 51% ]
..... [ 68% ]
```

PlatformIO – CLI

PlatformIO intenta por defecto detectar el target de nuestro programa:

```
$ pio device list
```

PlatformIO intenta por defecto detectar el target de nuestro programa:

```
$ pio run -e wemos-dlmini-relayshield -t upload
```

También le podemos proporcionar la ubicación del dispositivo:

```
$ pio run -e wemos-dlmini-relayshield -t upload --upload-port /dev/ttyUSB0
```

O la IP para hacer OTA (si ya tiene ESPurna instalado):

```
$ pio run -e wemos-dlmini-relayshield-ota -t upload --upload-port 192.168.20.57
```


Tuya-convert

<https://github.com/ct-Open-Source/tuya-convert>

```
$ # checkout and build espurna

$ cd $HOME/workspace
$ git clone https://github.com/xoseperez/espurna
(...)
$ cd espurna/code
$ pio run -e blitzwolf-bwshpx
(...)
$ ls -la .pioenvs/blitzwolf-bwshpx/firmware.bin
-rw-rw-r-- 1 xose xose 475120 Mar 28 13:41 .pioenvs/blitzwolf-bwshpx/firmware.bin

$ # checkout and start tuya-converter to flash espurna binary

$ cd $HOME/workspace
$ git clone https://github.com/ct-Open-Source/tuya-convert
(...)
$ cd tuya-convert
$ ./install_prereq.sh
(...)
$ ln -s $HOME/espurna/code/.pioenvs/blitzwolf-bwshpx/firmware.bin files/blitzwolf-bwshpx.bin
$ ./start_flash.sh
(...)
$ curl http://10.42.42.42/flash3?url=http://10.42.42.1/files/blitzwolf-bwshpx.bin
```

Primeros pasos

Conexión serie

```
$ pio device monitor -b 115200 --echo $@
```

```
---8<-----
```

```
[000325] [MAIN] ESPURNA 1.13.6-dev
[000325] [MAIN] xose.perez@gmail.com
[000326] [MAIN] http://tinkerman.cat

[000326] [MAIN] CPU chip ID: 0xXXXXXX
[000329] [MAIN] CPU frequency: 80 MHz
[000332] [MAIN] SDK version: 1.5.3(aec24ac9)
[000336] [MAIN] Core version: 2.3.0
[000339] [MAIN] Core revision: 159542381
[000343]
[000344] [MAIN] Flash chip ID: 0xXXXXXX
[000347] [MAIN] Flash speed: 40000000 Hz
[000351] [MAIN] Flash mode: DOUT
[000354]
[000355] [MAIN] Flash size (CHIP)   : 4194304 bytes / 1024 sectors ( 0 to 1023)
[000362] [MAIN] Flash size (SDK)     : 4194304 bytes / 1024 sectors ( 0 to 1023)
[000369] [MAIN] Reserved             : 4096 bytes / 1 sectors ( 0 to 0)
[000376] [MAIN] Firmware size       : 475120 bytes / 116 sectors ( 1 to 116)
[000383] [MAIN] Max OTA size         : 2666496 bytes / 651 sectors ( 117 to 767)
[000391] [MAIN] SPIFFS size          : 1015808 bytes / 248 sectors ( 768 to 1015)
[000398] [MAIN] EEPROM size          : 16384 bytes / 4 sectors (1016 to 1019)
[000405] [MAIN] Reserved             : 16384 bytes / 4 sectors (1020 to 1023)
[000412]
```

Conexión serie

```
[000413] [MAIN] EEPROM sectors: 1019, 1018, 1017, 1016
[000418] [MAIN] EEPROM current: 1018
[000421]
[000422] [MAIN] EEPROM: 4096 bytes initially | 957 bytes used (23%) | 3139 bytes free (76%)
[000431] [MAIN] Heap : 35464 bytes initially | 5328 bytes used (15%) | 30136 bytes free (84%)
[000439] [MAIN] Stack : 4096 bytes initially | 768 bytes used (18%) | 3328 bytes free (81%)
[000447]
[000448] [MAIN] Boot version: 31
[000451] [MAIN] Boot mode: 1
[000453] [MAIN] Last reset reason: External System
[000458] [MAIN] Last reset info: Fatal exception:0 flag:6 (EXT_SYS_RST) epc1:0x00000000 epc2:0x00000000 epc3:0x00000000
excvaddr:0x00000000 depc:0x00000000
[000471]
[000472] [MAIN] Board: WEMOS_D1_MINI_RELAYSHIELD
[000476] [MAIN] Support: ALEXA_API_BROKER_BUTTON_DEBUG_SERIAL_DEBUG_TELNET_DEBUG_WEB_DOMOTICZ_HOMEASSISTANT_LED_MDNS_SERVER_MQTT
NTP_SCHEDULER_TELNET_TERMINAL_THINGSPEAK_WEB
[000491] [MAIN] WebUI image: SMALL
[000494]
[000710] [MAIN] Firmware MD5: 2bfd40d62cdd5b1e9c0a7e29fc090be8
[000711] [MAIN] Power: 2766 mV
[000712] [MAIN] Power saving delay value: 1 ms
[000712] [MAIN] WiFi Sleep Mode: MODEM
[000715]

---8<-----
```

Conexión serie

```
[000730] [TELNET] Listening on port 23
[000732] [WEBSERVER] Webserver running on port 80
[000735] [RELAY] Retrieving mask: 0
[000736] [RELAY] Relay #0 boot mode 0
[000736] [RELAY] #0 set to OFF
[000737] [RELAY] Number of relays: 1
[000740] [BUTON] Number of buttons: 1
[000745] [LED] Number of leds: 1
[000746] [MQTT] Async ENABLED, SSL DISABLED, Autoconnect ENABLED
[000760] [NTP] Update intervals: 65s / 2074s
[FAUXMO] Device 'ESPURNA-XXXXXX' added as #0
[000762] [THINGSPEAK] Async ENABLED, SSL DISABLED
[000821] [WIFI] Creating access point
[000880] [WIFI] Captive portal enabled
[000880] [WIFI] ----- MODE AP
[000881] [WIFI] SSID  ESPURNA-XXXXXX
[000881] [WIFI] PASS  fibonacci
[000884] [WIFI] IP    192.168.4.1
[000887] [WIFI] MAC   XX:XX:XX:XX:XX:XX
[000890] [WIFI] -----
[000897] [MDNS] OK
[001739] [RELAY] Setting relay mask: 0
[060001] [MAIN] System OK
```

Conectarse a la WiFi

- Buscar una red con nombre ESPURNA-XXXXXX
- Usar la contraseña por defecto “fibonacci”
- Navegar a <http://192.168.4.1>
- Cambiar la contraseña por defecto
- Volver a conectarse a la WiFi con la nueva contraseña y volver a 192.168.4.1
- Ir al apartado de WIFI y configurar la red a la que quieras conectar el dispositivo

- Alternativamente se puede compilar ESPurna con soporte para WPS y SmartConfig (hay apps compatibles en Google Play)

Interfaz Web

- FRIDGE
- ESPURNA 1.13.5-dev
- STATUS
- GENERAL
- DOMOTICZ
- HASS
- MQTT
- NTP
- SCHEDULE
- SENSORS
- SWITCHES
- THINGSPEAK
- WIFI
- ADMIN
- DEBUG

Save

Reconnect

Reboot

© 2016-2019
Xose Pérez
@xoseperez
http://tinkerman.cat
ESPurna @ GitHub
GPLv3 license

STATUS

Current configuration

Switch #0	OFF ON	
Current #0	0.337A	HLW8012 @ GPIO(5,14,13)
Voltage #0	219V	HLW8012 @ GPIO(5,14,13)
Active Power #0	71W	HLW8012 @ GPIO(5,14,13)
Reactive Power #0	16W	HLW8012 @ GPIO(5,14,13)
Apparent Power #0	73W	HLW8012 @ GPIO(5,14,13)
Power Factor #0	97%	HLW8012 @ GPIO(5,14,13)
Energy #0	75900729J	HLW8012 @ GPIO(5,14,13) (since 2019-01-28 20:03:...

Manufacturer	TINKERMAN	Network	daoiz
Device	ESPURNA_H08	BSSID	XXXXXXXXXX
Chip ID	XXXXXXXXXX	Channel	6
Wifi MAC	XXXXXXXXXX	RSSI	-61
SDK version	1.5.3(aec24ac9)	IP	192.168.1.167 (telnet)
Core version	2.3.0	Free heap	19336 bytes
Firmware name	ESPURNA	Load average	4%
Firmware version	1.13.5-dev	VCC	3196mV
Firmware revision		MQTT Status	CONNECTED
Firmware build date	2019-02-26 03:42:27	NTP Status	SYNC'D
Firmware size	491840 bytes	Current time	2019-03-28 18:23:35
Free space	2650112 bytes	Uptime	23d 03h 23m 42s
		Last update	21 seconds ago

OTA Manager

```
$ cd espurna/code
$ python install -r requirements.txt
$ python ota.py
```

ESPurna OTA Manager v0.3

#	HOSTNAME	IP	MAC	APP	VERSION	DEVICE	MEM_SIZE	SDK_SIZE	FREE_SPACE
1	3DPRINTER	192.168.1.120	XXXX	ESPURNA	1.13.6-dev	BLITZWOLF_BWSHPX	1024	1024	532480
2	DISHWASHER	192.168.1.169	XXXX	ESPURNA	1.13.5-dev	BLITZWOLF_BWSHPX	1024	1024	532480
3	ESPURNA-B93C93	192.168.1.192	XXXX	ESPURNA	1.13.6-dev	WEMOS_D1_MINI_RELAYSHIELD	4096	4096	2646016
4	FRIDGE	192.168.1.167	XXXX	ESPURNA	1.13.5-dev	TINKERMAN_ESPURNA_H08	4096	4096	2650112
5	HEATER	192.168.1.193	XXXX	ESPURNA	1.13.5-dev	BLITZWOLF_BWSHPX	1024	1024	532480
6	LIVINGLAMPDOWN	192.168.1.194	XXXX	ESPURNA	1.12.7a	AITHINKER_AI_LIGHT	1024	1024	503808
7	LIVINGLAMPUP	192.168.1.195	XXXX	ESPURNA	1.12.7a	AITHINKER_AI_LIGHT	1024	1024	503808
8	LIVINGLIGHT	192.168.1.196	XXXX	ESPURNA	1.13.5-dev	ITEAD_S20	1024	1024	552960
9	MICROWAVES	192.168.1.200	XXXX	ESPURNA	1.13.5-dev	BLITZWOLF_BWSHPX	1024	1024	532480
10	OFFICE	192.168.1.107	XXXX	ESPURNA	1.13.5-dev	BLITZWOLF_BWSHPX	1024	1024	532480
11	OFFICELIGHT	192.168.1.129	XXXX	ESPURNA	1.13.5-dev	TINKERMAN_ESPURNA_SWITCH	4096	4096	2670592
12	RFM69GW	192.168.1.144		RFM69GW	1.0.1	TINKERMAN_RFM69GW	0	0	0
13	THERMOMIX	192.168.1.148	XXXX	ESPURNA	1.13.5-dev	BLITZWOLF_BWSHPX	1024	1024	532480
14	WASHER	192.168.1.198	XXXX	ESPURNA	1.13.5-dev	ITEAD_SONOFF_POW_R2	4096	1024	536576
15	WORKLIGHT	192.168.1.178	XXXX	ESPURNA	1.13.5-dev	NEO_COOLCAM_NAS_WR01W	1024	1024	552960

```
$ python ota.py -f FRIDGE
```