UNIVERSITAT JAUME I

EI1062 – IR2162
Diseño de sistemas empotrados y de tiempo real

Tema 5 – Introducción a los microcontroladores
El microcontrolador ESP32

Grado en Ingeniería Informática

Grado en Inteligencia Robótica

Estructura del tema

- Introducción
- Estructura y características
- Doble CPU
- Comunicaciones inalámbricas
  - Bluetooth
  - WiFi
  - ESP-NOW
- JTAG

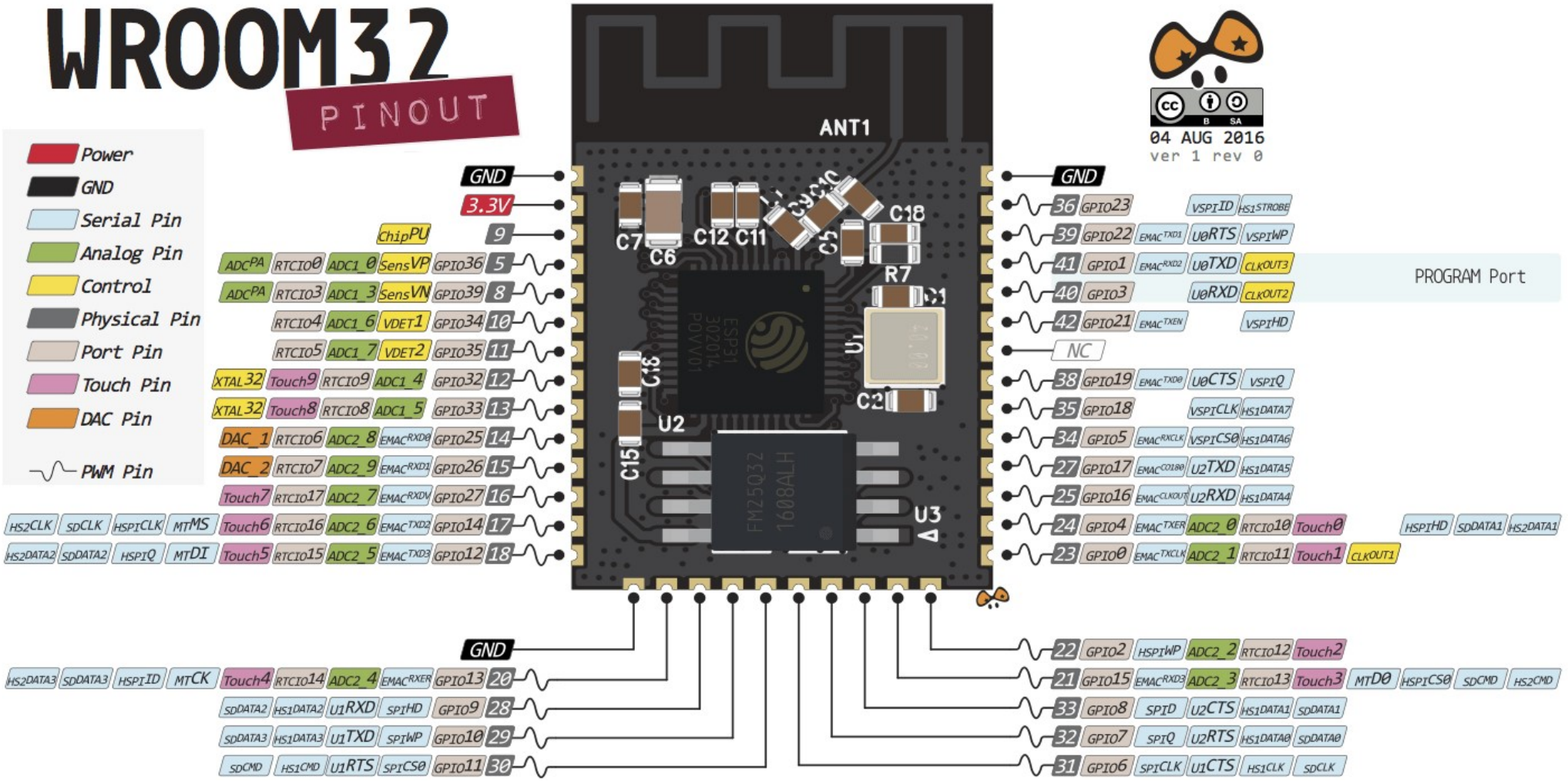
INTRODUCCIÓN

- Microcontrolador diseñado para aplicaciones móviles, wearable computing e IoT.
- Incorpora comunicación inalámbrica a 2.4 GHz soportando WiFi, Bluetooth y BLE (Low Energy Bluetooth).
- Antena incorporada.
- Posee dos procesadores Xtensa LX6 de 32 bits de hasta 240 Mhz.
- RTC ULP (*Ultra Low Power processor*).

APLICACIONES

- Generic low power IoT sensor hub.
- Generic low power IoT loggers.
- Video streaming from camera.
- Music players.
- WiFi enabled toys.
- Speech recognition devices.
- Audio headsets
- Smart power plugs.
- Home automation.
- Industrial wireless control.
- Baby monitors.
- Wearable electronics.
- Security ID tags.
- WiFi location aware products.
- Health care.

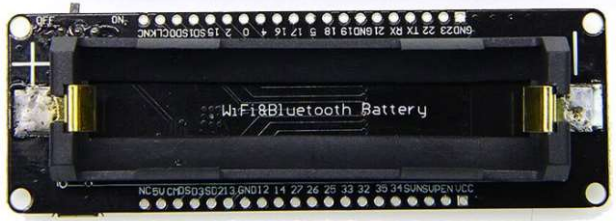
PINOUT

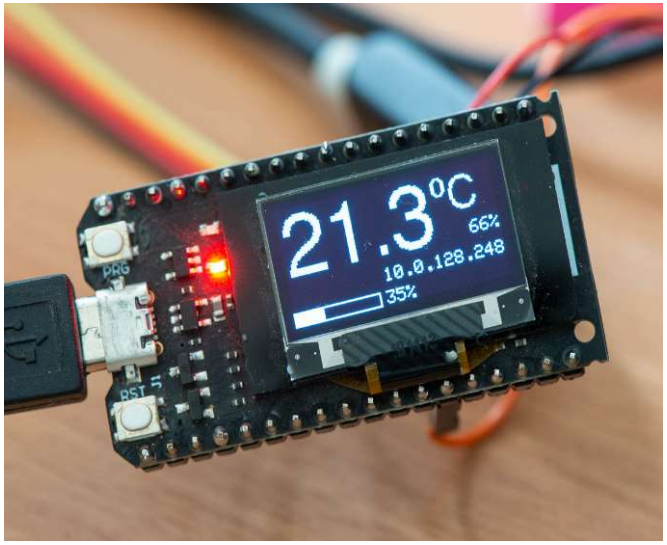

WROOM32
PINOUT
04 AUG 2016
ver 1 rev 0
Power
GND
Serial Pin
Analog Pin
Control
Physical Pin
Port Pin
Touch Pin
DAC Pin
PWM Pin
ANT1
PROGRAM Port

TARJETAS DE DESARROLLO

- El chip suele presentarse instalado en tarjetas de desarrollo, existiendo diversas versiones con pinouts específicos para cada una de ellas.
- Suelen disponer de un controlador USB para programación y comunicación de consola.
- Algunas disponen de un LED integrado, display OLED o batería incorporada con circuito de carga.

TARJETAS DE DESARROLLO

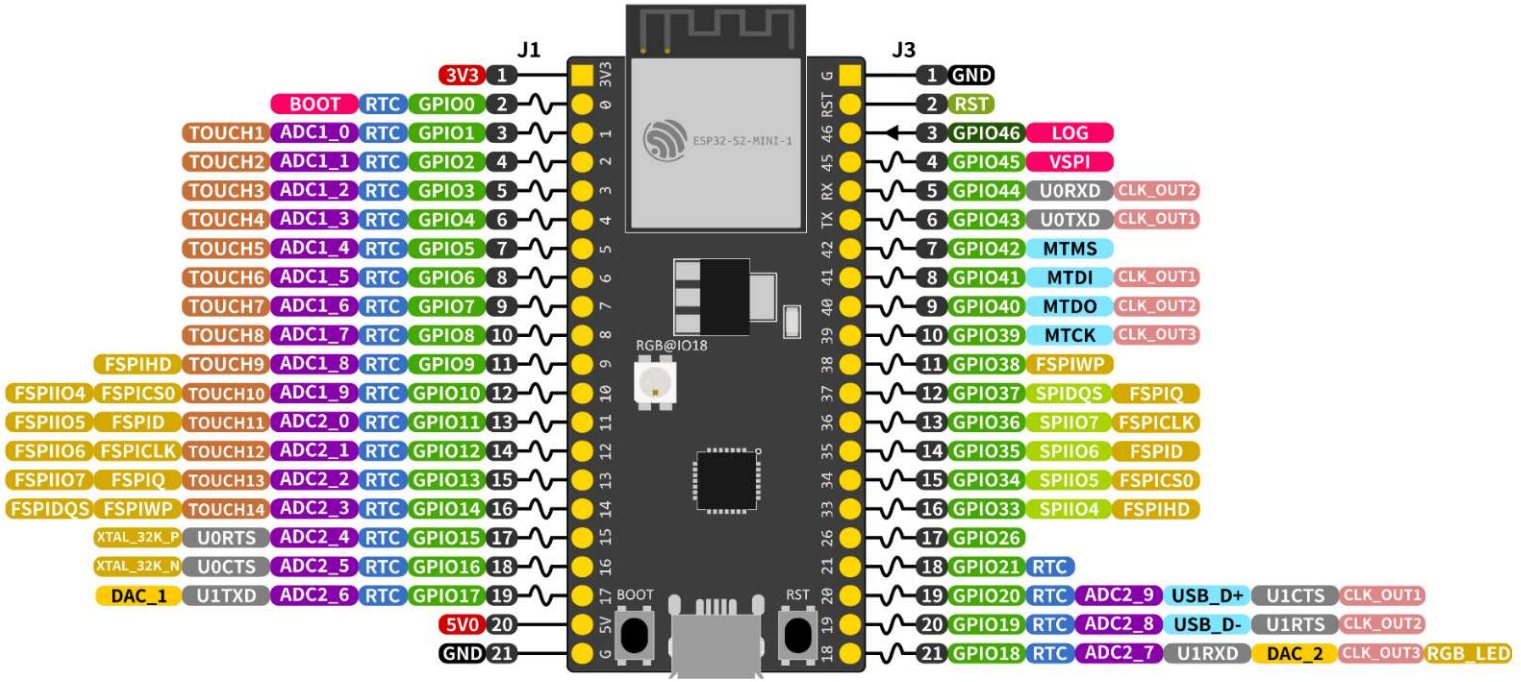




PINOUT TARJETAS

ESP32-S2-DevKitM-1

ESPRESSIF



ESP32-S2 Specs

32-bit Xtensa® single-core @240MHz
Wi-Fi IEEE 802.11 b/g/n 2.4GHz
320 KB SRAM (16 KB SRAM in RTC)
128 KB ROM
43 GPIOs, 4x SPI, 2x UART, 2x I2C,
Touch, I2S, RMT, LED PWM, USB-OTG,
TWAI®, 2x 8-bit DAC, 12-bit ADC

PWM Capable Pin
GPIOX GPIO Input Only
GPIOX GPIO Input and Output
JTAG/USB JTAG for Debugging and USB
ADCX_CH Analog-to-Digital Converter
TOUCHX Touch Sensor Input Channel
OTHER Other Related Functions
SERIAL Serial for Debug/Programming
DAC_X Digital-to-Analog Converter
STRAP Strapping Pin Functions
RTC RTC Power Domain (VDD3P3_RTC)
MISC Miscellaneous/Secondary functions
GND Ground
CLK_OUTx Clock Output
PWD Power Rails (3V3 and 5V)

PERIFÉRICOS

- 34 GPIOs.
- 18 ADC 12 bits.
- 2 DAC 8 bits.
- 10 touch sensors.
- Sensor temperatura.
- Sensor Hall.
- 16 canals PWM.
- 2 PWM motor.
- Ethernet MAC.
- 4 SPI.
- 2 I2C.
- 2 I2S.
- 3 UART.
- 1 Host SD/eMMC/SDIO.
- 1 esclavo SDIO.
- 1 TWAI (CAN 2.0).
- 1 IR (TX/RX).

PROCESADOR Y MEMORIA

- Procesador doble Xtensa L6 32 bits hasta 600 MIPS (240 MHz).
- 448 Kb ROM.
- 520 Kb SRAM.
- 16 Kb RAM en RTC (ULP).
- QSPI flash/SRAM hasta 4 x 16 Mb.

USO DE PINES

GPIO	INPUT	OUTPUT	COMENTARIO
0	BOOT	BOOT	Salida de señal PWM durante BOOT
1	TX	BOOT	Salida debug durante BOOT
2	OK	OK	Onboard LED (algunas tarjetas)
3	BOOT	RX	HIGH en BOOT
4	OK	OK	
5	OK	BOOT	Salida de señal PWM durante BOOT
6 – 11	NO	NO	SPI flash integrada
12	BOOT	OK	BOOT falla si entrada a nivel alto
13	OK	OK	
14 – 15	OK	BOOT	Salida de señal PWM durante BOOT
16 – 19	OK	OK	
21- 23	OK	OK	
25 – 27	OK	OK	
32 – 33	OK	OK	
34 – 36	OK	NO	Solo entrada. No pull-up ni pull-down
39	OK	NO	Solo entrada. No pull-up ni pull-down

AHORRO ENERGÍA

- 5 modos de bajo consumo:
  - Active
  - Modem
  - Light
  - Deep
  - Hibernation
- Coprocesador ULP (*Ultra Low Power*) activo en todos ellos.
- El sistema puede despertar por cambios en la mayoría de GPIOs o mediante timer (RTC).

SENSORES INTERNOS

- 10 sensores capacitivos (*Touch*):
  - GPIOs 4, 0, 2, 15, 13, 12, 14, 27, 33, 32
  - Detectan contacto por cambio de carga eléctrica.
  - Conectados al ULP-coprocessor para poder despertar el sistema cuando está en modo *deep sleep*.
- Sensor de temperatura.
- Sensor Hall
  - Posición
  - Proximidad
  - Velocidad

CONVERSIÓN D/A Y A/D

- 18 conversores A/D de 12 bits (3.3V, 0-4095).
- ADC1: GPIOs 36, 37, 38, 39, 32, 33, 34, 35
- ADC2: GPIOs 4, 0, 2, 15, 13, 12, 14, 27, 25, 26
- ADC2 y WiFi no pueden usarse a la vez.
- Los GPIOs 0, 2, 4 y 15 del ADC2 no pueden usarse libremente porque están asociados a otras funciones en determinadas tarjetas de desarrollo.
- Los GPIOs 25 y 26 están conectados a los DAC de 8 bits DAC1 y DAC2 respectivamente.

SPI, I2C, I2S, UART, CAN, RMT

- 4 buses SPI
  - SPI0 acceso a memoria externa.
  - SPI1 master.
  - SPI2 y SPI3 master o esclavo.
- 1 controlador I2C master o esclavo.
- 2 controladores I2S (*streaming digital data*).
- 3 controladores UART (RS232, RS485, IrDA).
- 1 controlador TWAI (CAN 2.0).
- 1 RMT (Remote Control) de 8 canales.

SDIO / SD / MMC

- 1 controlador SDIO esclavo
  - Un master externo puede acceder a la memoria compartida y al DMA del ESP32.
- 2 controladores SD/MMC:
  - CARD0 y CARD1.
  - Conexión posible a dos tarjetas de memoria.
  - Formatos SD, MMC, CE-ATA.
  - CARD0: modos de 1, 4 y 8 bits
  - CARD1: modos de 1 y 4 bits

PWM

- 16 canales PWM LED controller
  - Pueden controlar cualquier dispositivo
  - 8 de alta velocidad y 8 de baja velocidad
  - Se especifica frecuencia y ciclo de trabajo
- 2 módulos MCPWM (control de motores).
  - 6 salidas PWM por módulo.
- 8 PCNT de 16 bits (contador de pulsos)

SEGURIDAD

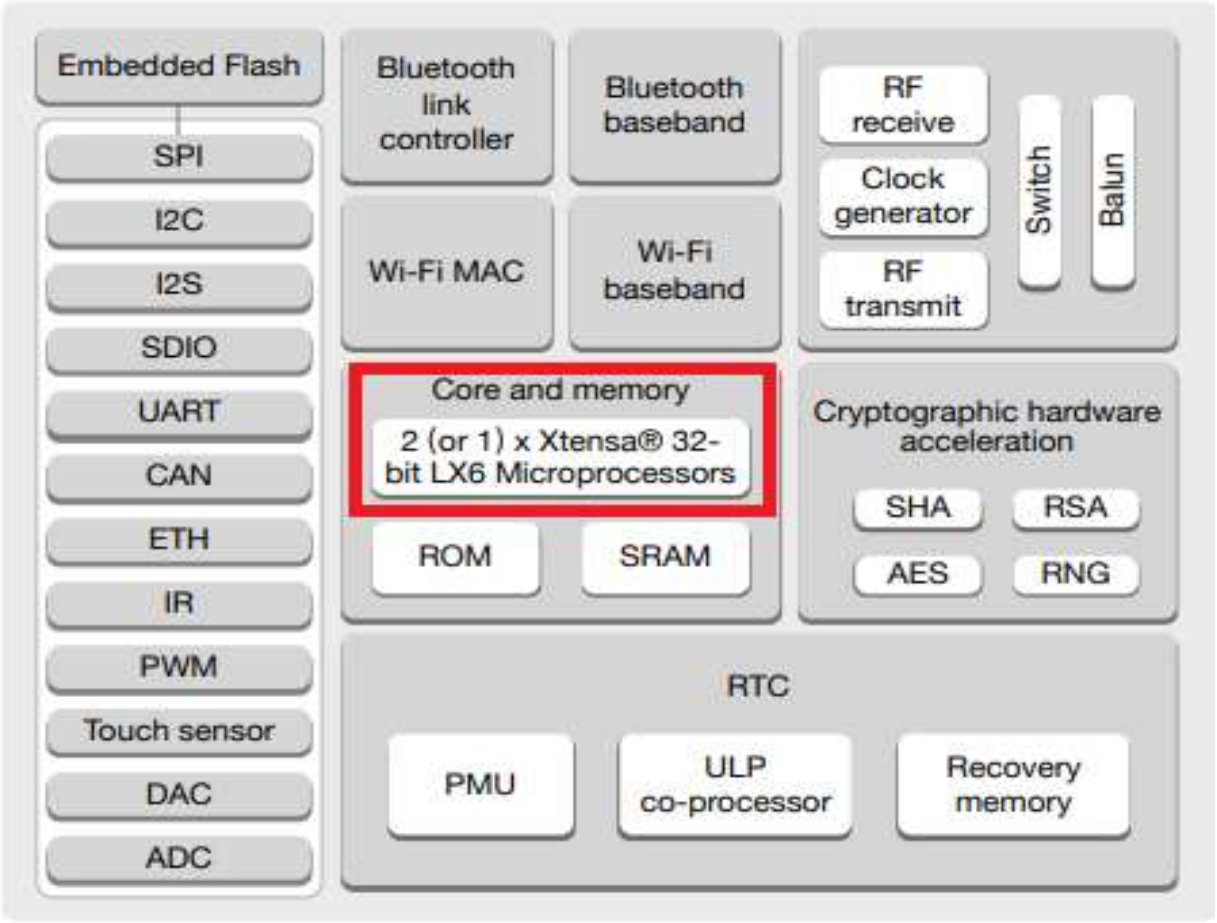
- Acelerador AES (128, 192 y 256 bits).
- Acelerador SHA (256, 384 y 512 bits).
- Acelerador RSA.
  - Exponenciación y módulo.
  - Multiplicación y módulo.
  - Multiplicación.
  - Diversas longitudes de operadores comprendidas entre 512 bits y 4096 bits.
- RNG (*Random Generator*) de 32 bits.

MISCELÁNEA

- Ethernet MAC (*Media Access Controller*).
  - Conexión a Ethernet cableada RJ45.
- Efuse.
  - 32 palabras de 32 bits.
  - Una vez escrito un “1” no puede cambiarse.
- PID (*Process ID*).
  - Hasta 8 processos simultáneos (0...7) en dos CPUs.
  - Conmutación entre procesos por interrupción o solicitud de los procesos 0 o 1.
- Controlador DMA (*Direct Memory Access*).

DOBLE CPU

- El microcontrolador ESP32 dispone de dos procesadores Xtensa LX6 (Core0 y Core1).



DOBLE CPU

- Por defecto, el código se ejecuta en el procesador Core0.
- Se puede determinar en qué procesador se está ejecutando una tarea con la función **`xPortGetCoreID()`**:

```
void loop() {
  Serial.print("loop() running on core ");
  Serial.println(xPortGetCoreID());
}
```

DOBLE CPU

- Se pueden crear tareas y asignarlas a uno de los procesadores con:

```
TaskHandle_t Tarea1;

void setup() {
    xTaskCreatePinnedToCore(
        Task1code, /* Función para invocar la tarea */
        "Tarea1",  /* Nombre de la tarea */
        10000,     /* Tamaño en words de la pila */
        NULL,      /* Parámetro de la tarea */
        0,         /* Prioridad de la tarea (0...7) */
        Tarea1,    /* Handle de la tarea */
        0);        /* Core en que se ejecutará */
    }
}

void Task1code( void * param) {
    /* Código de la tarea */
}

void loop() {
}
```

DOBLE CPU

- Si en algún momento se desea finalizar la tarea, se puede conseguir invocando:

```
vTaskDelete(Tarea1);
```

donde “Tarea1” es el handle previamente definido.

BLUETOOTH

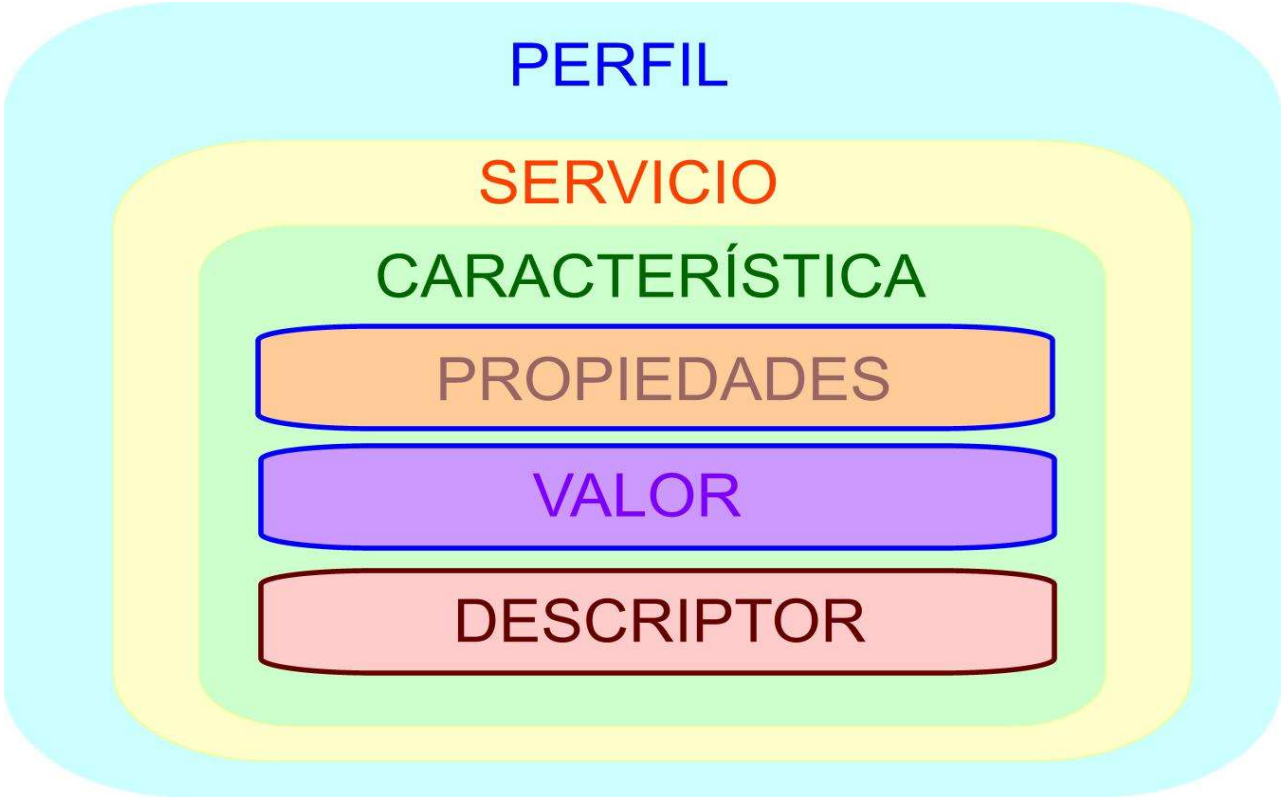
- El microcontrolador ESP32 incorpora los protocolos Bluetooth y BLE (Bluetooth Low Energy).
- Bluetooth puede asimilarse a un puerto serie.
- BLE es una evolución de bajo consumo.
  - Mantiene el dispositivo en modo sleep mientras no haya comunicación.
  - Reduce el consumo a la centésima parte de Bluetooth.
  - Soporta modos broadcast (uno a varios) y mesh network (varios a varios).
  - Se emplea en salud, fitness, balizas y automatización.

BLUETOOTH

```
//This example code is in the Public Domain (or CC0 licensed, at your option.)
//By Evandro Copercini - 2018
//This example creates a bridge between Serial and Classical Bluetooth (SPP)
//and also demonstrate that SerialBT have the same functionalities of a normal Serial

#include "BluetoothSerial.h"
#if !defined(CONFIG_BT_ENABLED) || !defined(CONFIG_BLUEDROID_ENABLED)
#error Bluetooth is not enabled! Please run `make menuconfig` and enable it
#endif

BluetoothSerial SerialBT;
void setup() {
  Serial.begin(115200);
  SerialBT.begin("ESP32test"); //Bluetooth device name
  Serial.println("The device started, now you can pair it with bluetooth!");
}

void loop() {
  if (Serial.available()) {
    SerialBT.write(Serial.read());
  }
  if (SerialBT.available()) {
    Serial.write(SerialBT.read());
  }
  delay(20);
}
```

BLE

- Emplea una estructura jerárquica de datos para la comunicación, denominada GATT (Generic ATTributes):



BLE

- El nivel superior de la jerarquía lo ocupa el perfil, que posee uno o más servicios.
- Cada servicio contiene al menos una característica.
- Existen servicios predefinidos por SIG (*Bluetooth Special Interest Group*).
- Las características contienen descriptores, metadatos de la declaración de la característica.
- Cada servicio, característica y descriptor tiene un UUID (*Universally Unique Identifier*) de 128 bits.
- Los UUID se pueden obtener del SIG (genéricos) o generar en el *UUID generator website* (propios).

WiFi

- ESP32 incorpora el protocolo WiFi.
- Puede actuar como *Station*, como *AccessPoint* o ambos.
- Solamente es compatible con la banda de 2'4 GHz.
- Soporta los protocolos 802.11b y 802.11g.
- Soporta el protocolo 802.11n con ambos anchos de banda de 20MHz y 40MHz.
- Transferencia de datos de hasta 150 Mbps.
- Potencia de transmisión ajustable.
- Selector de antena integrada/externa.
- RSSI (Radio Signal Strenght Information). Medidor de intensidad de la señal de radio recibida.

ESP-NOW

- ESP-NOW es un protocolo de comunicación inalámbrica desarrollado por Espressif para ransmisión de paquetes pequeños de forma sencilla.
- Una vez emparejados, si pierden conexión entre ellos por cualquier motivo, los dispositivos vuelven a conectarse automáticamente.
- Permite la comunicación entre dos dispositivos, de uno a varios o de varios a uno.

ESP-NOW

- ESP-NOW posee las siguientes características:
  - Comunicación encriptada o no encriptada.
  - Dispositivos mezclados (encriptados – no encriptados).
  - *Callback* función opcional para confirmar el éxito o fallo de la transmisión.
- Y las siguientes limitaciones:
  - Máximo de 10 dispositivos encriptados en modo *Station*.
  - Máximo 6 dispositivos encriptados en modos *AccessPoint* o *Station* + *Access Point*.
  - Máximo 20 dispositivos entre encriptados y no encriptados.
  - Máximo payload de 250 bytes por paquete.

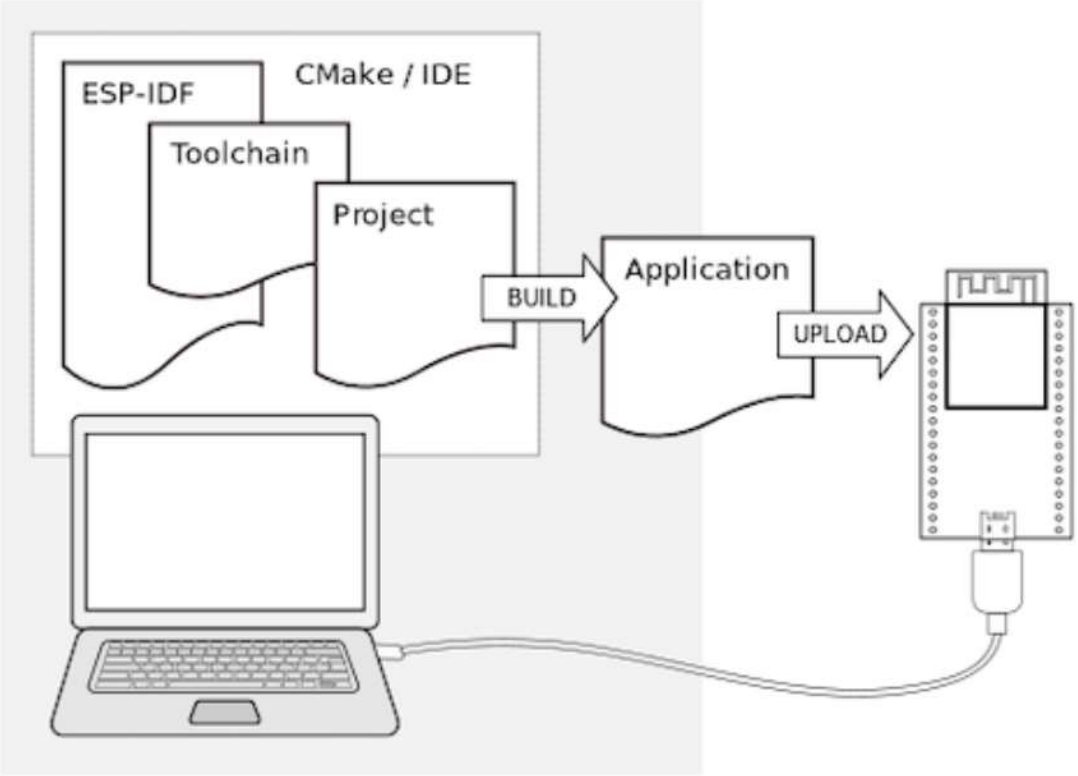
ESP-NOW

- Para el transmisor, se deben seguir los pasos:
  - Inicializar ESP-NOW.
  - Declarar una función para enviar datos `onDataSent()`.
  - Añadir un corresponsal mediante su MAC.
  - Enviar un mensaje al corresponsal con `esp_now_send()`.

- Para el receptor, los pasos a seguir son:
  - Inicializar ESP-NOW.
  - Declarar la función de recepción `onDataRecv()`.
  - Dentro de la función, guardar el paquete recibido.

ESP-IDF

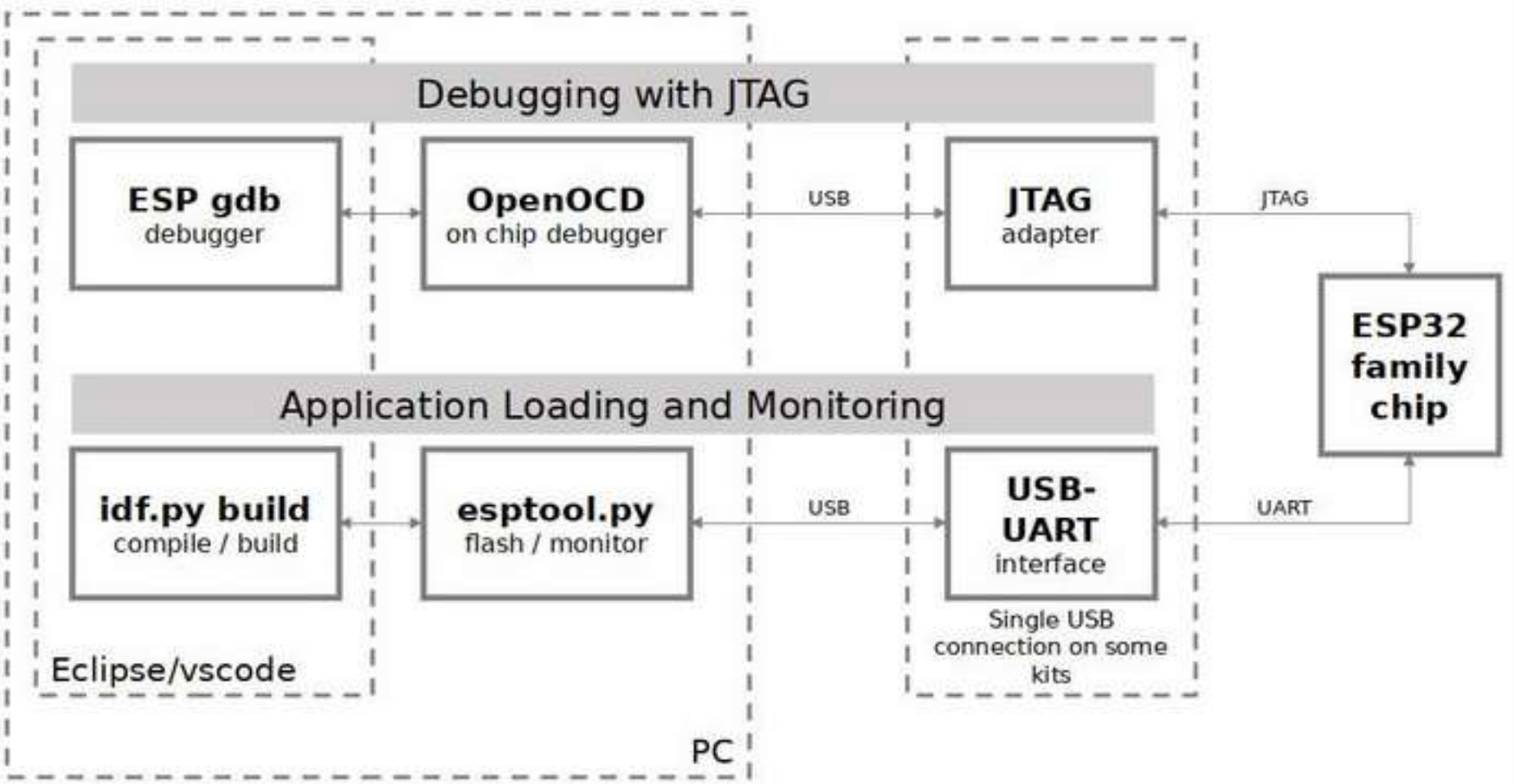
- ESP-IDF (Espressif IoT Development Framework) es el entorno de desarrollo para el ESP32.
- Se ejecuta desde línea de comandos o desde el IDE.



JTAG Debug

- FreeRTOS Es el Sistema Operativo de Tiempo Real incluido en ESP-IDF. Permite programación multinúcleo.
- Depurar este tipo de código sin las herramientas apropiadas puede resultar muy complicado.
- La mejor forma de depurar en este caso es emplear un *debugger*, conectado a los procesadores a través de un puerto de depuración.
- Espressif ofrece OpenOCD, que soporta el procesador ESP32 y FreeRTOS, para depurar con GDB (GNU Debugger).
- Como consecuencia, el computador de desarrollo se debe conectar al ESP32 a través de USB y de JTAG.

JTAG Debug


Debugging with JTAG
ESP gdb
debugger
OpenOCD
on chip debugger
USB
JTAG
adapter
JTAG
ESP32
family
chip
Application Loading and Monitoring
idf.py build
compile / build
esptool.py
flash / monitor
USB
USB-
UART
interface
UART
Single USB
connection on some
kits
Eclipse/vscode
PC

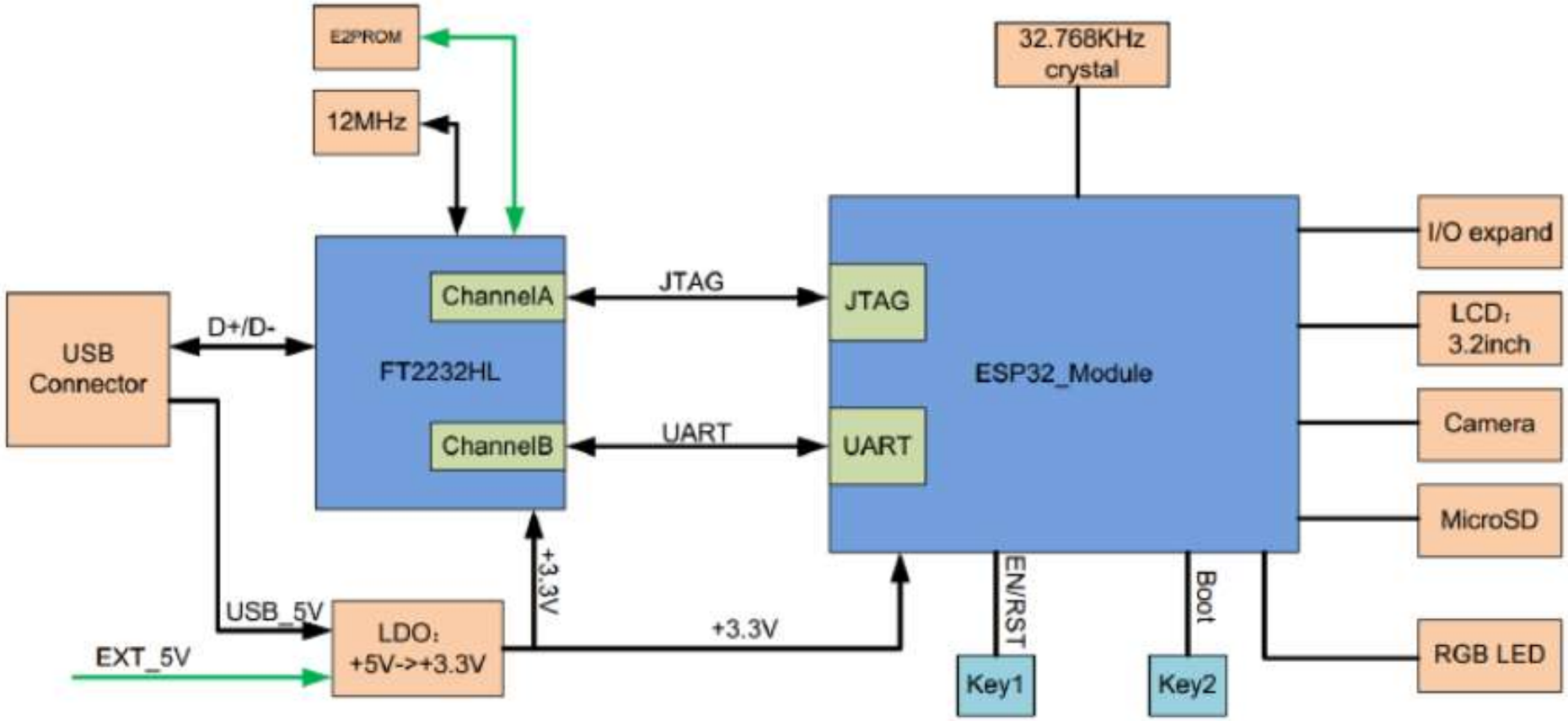
JTAG Debug

- JTAG (Joint Test Action Group) es un interfaz específico para depuración.
- En el diagrama anterior, la parte “Debugging With JTAG” muestra los elementos hw y sw relacionados con la depuración:
  - Xtensa ESP-32 elf-gdb debugger
  - OpenOCD on chip debugger
  - JTAG adapter
- La etiqueta “Application Loading and Monitoring” contiene los elementos necesarios para compilar, ensamblar y copiar en la flash del ESP32 los programas, así como comunicar con los mismos en tiempo de ejecución.

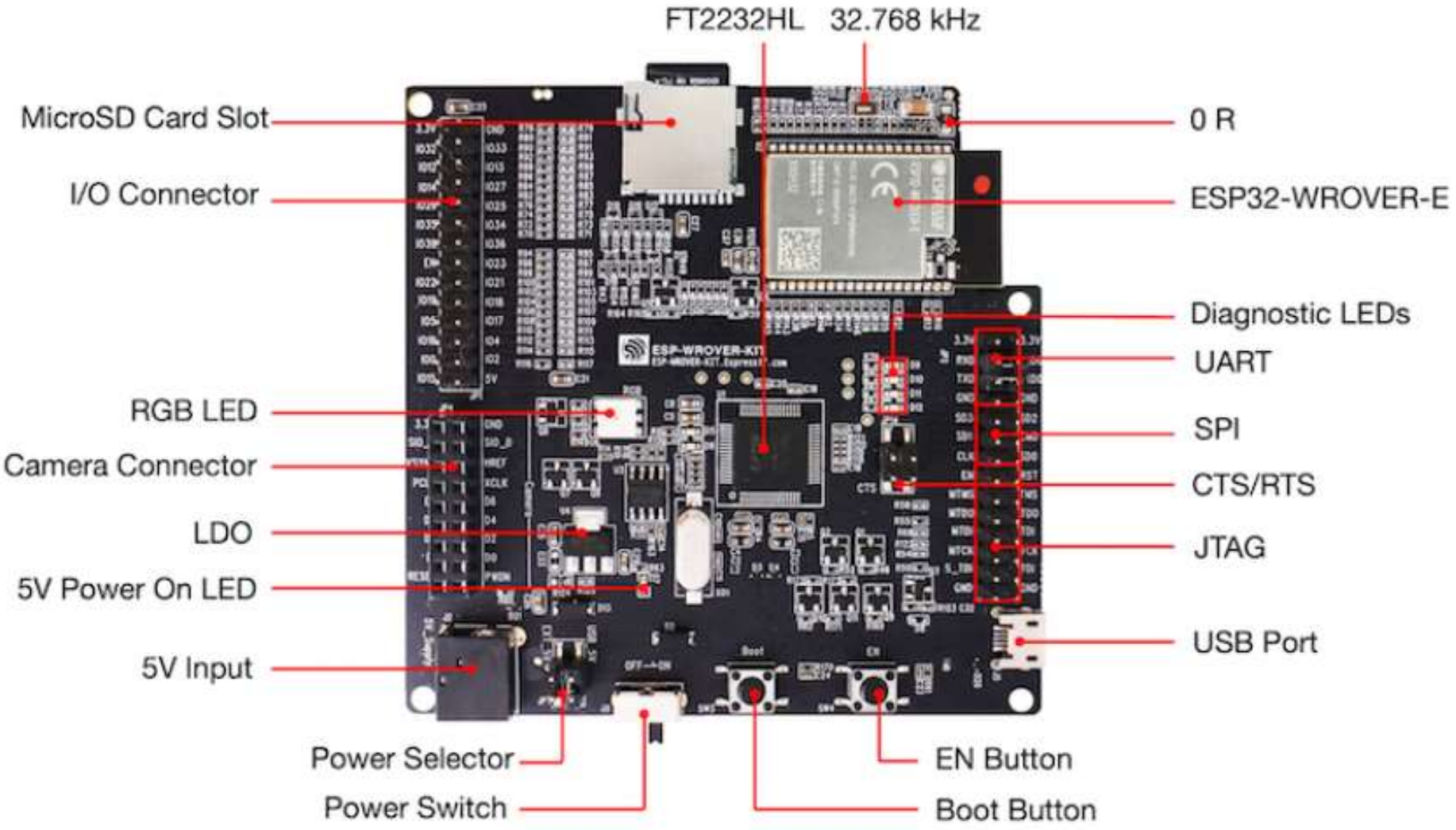
ESP-WROVER-KIT

- ESP32-WROVER-KIT es una tarjeta de desarrollo.
- Su principal característica diferenciadora es que incorpora un doble controlador USB (FTDI FT2232HL) que permite conectar simultáneamente a través de USB y de JTAG.
- Incluye, además:
  - Display LCD SPI de 3.2”.
  - Slot para tarjeta microSD.
  - Conector para cámara estándar OV7670.
  - LED RGB.
  - Conectores de expansión para acceso a los GPIOs del ESP32.
  - Entrada de alimentación de 5V.
  - Regulador de voltaje de 5V a 3.3V y 1A.

ESP-WROVER-KIT



ESP-WROVER-KIT



DIRECCIONES DE INTERÉS

- Espressif ESP32 - Getting started
- Espressif ESP32 - Programming Guide
- Espressif ESP32 - JTAG Debugging
- Espressif ESP32 - ESP32-WROVER-KIT
- Randomnerds ESP32 tutorial
- Savjee – Multitasking on ESP32 with Arduino and FreeRTOS

UNIVERSITAT JAUME I

EI1062 – IR2162
Diseño de sistemas empotrados y de tiempo real

Tema 5 – Introducción a los microcontroladores
El microcontrolador ESP32

Grado en Ingeniería Informática

Grado en Inteligencia Robótica