ESP8266 și ESP32
(overview)

Microcontrolere cu Wi-Fi

Învățarea arhitecturii și programării multi-tasking a microcontrolerului dual-core pe 32 de biți folosind ESP32 Dev Module, Arduino IDE, ESP-IDF și FreeRTOS - Comunitate / Educație - Arduino Forum

Cuprins

Introducere în ESP8266

- Microchip Wi-Fi cu cost redus, dezvoltat de Espressif Systems
- Lansat în 2014, a revoluționat dezvoltarea IoT
- Stivă TCP/IP completă și capabilități de microcontroler
- CPU RISC like pe 32 de biți: Tensilica (Cadence) Xtensa LX106
- Funcționează la 80 MHz și poate fi overclocked la 160 MHz.
- Preț tipic: $2-5 USD
- Module populare: ESP-01, ESP-12E, ESP-12F, NodeMCU

Specificații cheie ESP8266

Specificație	Valoare
CPU	Tensilica Xtensa LX106 @ 80/160 MHz
RAM	~80 KB disponibili pentru utilizator
Flash	512 KB până la 4 MB (extern)
Pini GPIO	17 (limitat de modul)
ADC	1 canal, 10 biți
Wi-Fi	802.11 b/g/n (2.4 GHz)
Tensiune de operare	3.3V
Consum de curent	~70 mA (mediu), 170 mA (vârf)

- Xtensa LX106 de la Tensilica nu face parte din generațiile RISC tradiționale (RISC-I, RISC-II, MIPS, ARM etc.). Este mai bine descris ca o arhitectură **de tip RISC configurabilă/extensibilă**

Poziția unică a Xtensa:

- ***Caracteristici de tip RISC:***

Arhitectură load-store, operații bazate pe registre, instrucțiuni de bază simple

- ***Caracteristici non-RISC:***
  - **Instrucțiuni de lungime variabilă** (de 16 și 24 biți) - RISC tradițional folosește lungime fixă
  - **Arhitectură configurabilă** - se pot adăuga instrucțiuni și funcționalități personalizate
  - **Cod dens** optimizare (mai aproape de CISC)

Nu este o tradițională RISC generație:

- Generațiile RISC se referă la: Berkeley RISC-I/II (anii 1980) → MIPS, SPARC, ARM (1-a generație comercială) → evoluția ARM → RISC-V (“a 5-a generație”).
- Xtensa a adoptat o abordare diferită în anii 1990, prioritizând personalizarea în defavoarea regulilor RISC stricte.

Clasificare mai precisă:

- arhitectură **Post-RISC**
- **Procesor configurabil de tip RISC**
- Parte din familia de **procesoare VLIW/configurabile** de la Tensilica (acum Cadence)
- Deși Xtensa LX106 are caracteristici RISC și este adesea numit informal "procesor RISC", nu se încadrează în generațiile RISC numerotate. Este un design arhitectural special inspirat de RISC, dar oferă personalizare și o mai bună compactare a codului, de care arhitecturile RISC tipice s-au ferit.

1. Configurabilitate

Aceasta înseamnă că arhitectura Xtensa este **personalizabilă înainte de fabricarea chipului** - proiectanții de chipuri pot adăuga sau elimina funcționalități în funcție de necesități.

Ce poate fi configurat:

- **Numărul de registre** (poate avea 32 sau 64 de registre de uz general)
- **Instrucțiuni personalizate** - adăugarea de instrucțiuni specializate pentru sarcini specifice
  - Exemplu: O companie care produce procesoare audio poate adăuga instrucțiuni DSP personalizate
  - Exemplu: Adăugarea de instrucțiuni specializate de accelerare criptografică
- **Funcționalități opționale** care pot fi incluse sau excluse:
  - Unitate în virgulă mobilă (FPU)
  - Unitate de multiplicare-acumulare (MAC)
  - Unitate de protecție a memoriei (MPU)
  - Interfețe periferice specifice
- **Dimensiunea cache** și arhitectura memoriei
- **Numărul de niveluri de întrerupere**

De ce este important:

- **Reducerea costurilor**: Plătești doar pentru ce ai nevoie (chip mai mic = mai ieftin)
- **Eficiență energetică**: Mai puțini tranzistori neutilizați = consum mai mic de energie
- **Performanță**: Adaugi exact instrucțiunile de care are nevoie aplicația ta

- RISC tradițional vs Xtensa (ARM Cortex-M3 vs Xtensa LX106

2. Caracteristici de densitate ale codului

Aceasta înseamnă încadrarea mai multor funcționalități ale programului în **mai puțin spațiu de memorie**. Important pentru sistemele încorporate cu flash/RAM limitat.

Instrucțiuni de lungime variabilă:

RISC tradițional (ca ARM, MIPS):

Toate instrucțiunile sunt pe 32 de biți (4 octeți)
ADD r1, r2, r3 → 32 biți
MOV r1, #5 → 32 biți
NOP → 32 biți (risipă!)

Xtensa LX106:

Instrucțiunile pot fi de 16 biți sau 24 biți
ADD a2, a3, a4 → 24 biți
MOVI a2, 5 → 24 biți
NOP → 16 biți (s-au economisit 16 biți!)

!!! Operații comune/ frecvente → 16 biți (reducere de 50% a dimensiunii!)

De ce contează densitatea codului pentru ESP8266:

- Doar ~80 KB RAM disponibil
- Spațiu flash limitat (512 KB până la 4 MB)
  - Cod mai mic = mai mult spațiu pentru date/funcționalități
  - Cod mai mic = mai puține citiri din flash = consum mai mic de energie
  - Cod mai mic = utilizare mai bună a cache-ului = execuție mai rapidă

Alte tehnici de densitate a codului în Xtensa:

- **Codificare densă a instrucțiunilor**: Operațiile comune primesc o codificare mai scurtă
- **Gruparea literalelor (Literal pooling)**: Constantele pot fi încorporate eficient în fluxul de instrucțiuni

Diagrama bloc ESP8266

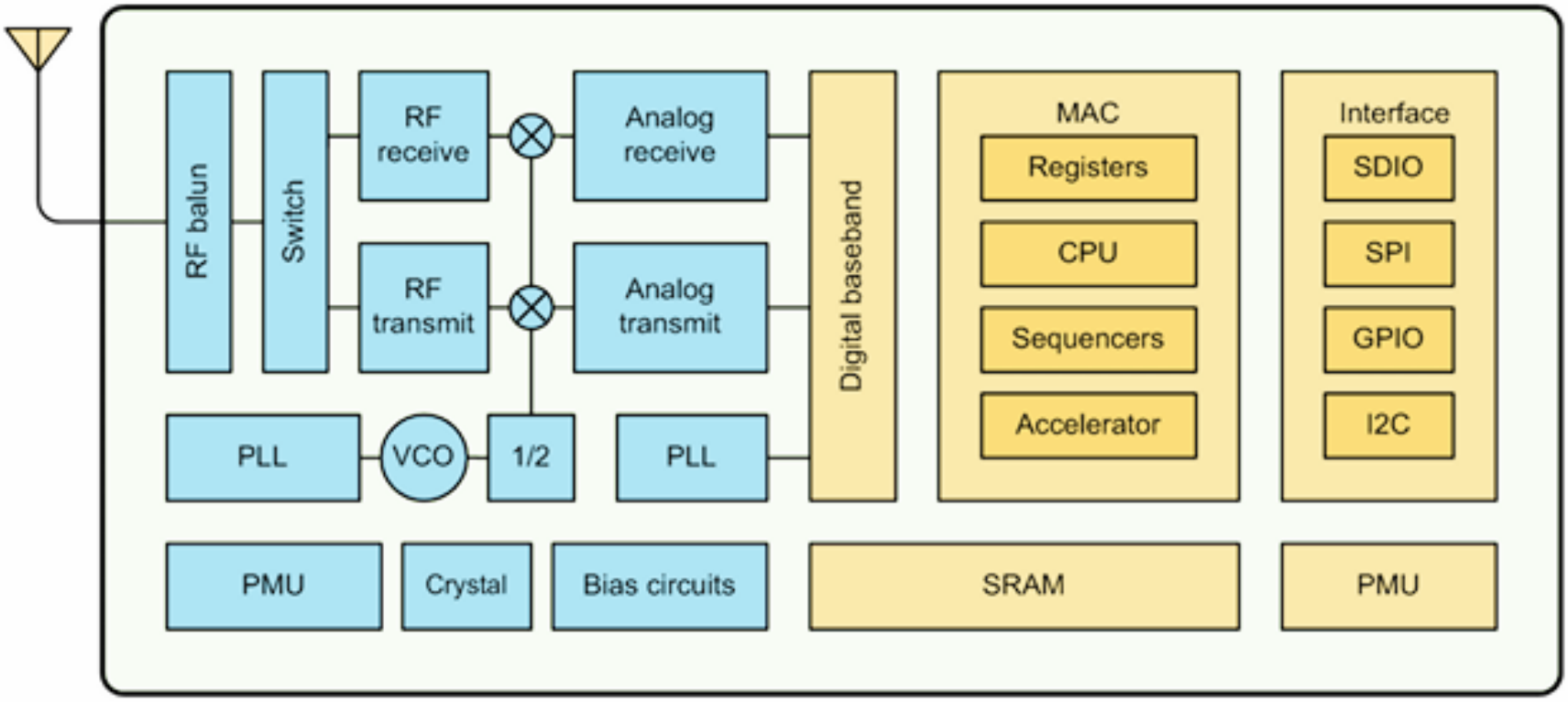


Figure 1 ESP8266EX Block Diagram

- Xtensa L106 este un **nucleu RISC configurabil / de tip like-RISC pe 32 de biți**: urmează principiile RISC (load/store, ALU pe 32 de biți, pipeline compact), dar le extinde cu instrucțiuni de lățime variabilă și configurabilitate a instrucțiunilor personalizate, așa că este mai bine descris ca un ***nucleu configurabil de tip RISC*** decât ca un design RISC clasic.

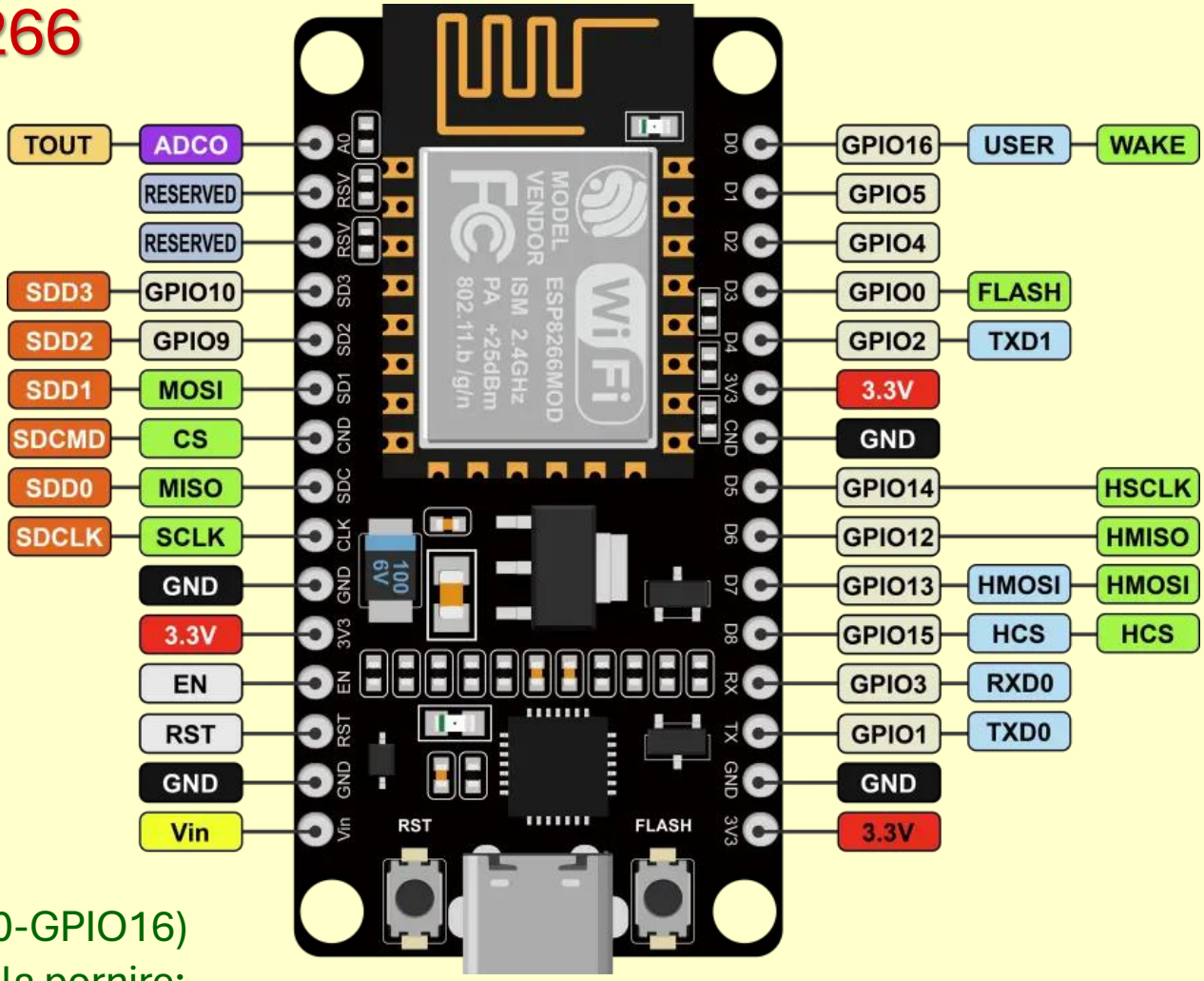
Nucleul Tensilica Xtensa LX106

- Arhitectură RISC pe 32 de biți
- Arhitectură Harvard (magistrale separate pentru instrucțiuni și date)
- 16 registre de uz general pe 32 de biți
- Format de instrucțiuni pe 16/24 de biți (cod compact)
- Înmulțire și împărțire hardware
- Operații load/store pe 16 biți și 8 biți
- Performanță tipică: ~80 MIPS @ 80/160 MHz
- Fără FPU (unitate în virgulă mobilă) - emulare software

Arhitectura memoriei ESP8266

- SRAM intern: ~160 KB total
  - ~80 KB disponibili pentru datele utilizatorului
  - ~35 KB utilizați de SDK și stiva Wi-Fi
  - RAM pentru instrucțiuni (IRAM) și RAM pentru date (DataRAM)
- Flash SPI extern: 512 KB până la 4 MB
  - Stochează codul programului și constantele
  - Sistem de fișiere (SPIFFS sau LittleFS)
- Regiuni I/O mapate în memorie
- Fără EEPROM - se folosește Flash sau stocare externă

GPIO și pini ESP8266



- 17 pini GPIO în total (GPIO0-GPIO16)
- Unii pini au funcții speciale la pornire:
  - GPIO0: Selecția modului de boot (LOW = mod Flash)
  - GPIO2: trebuie să fie HIGH la pornire
  - GPIO15: Trebuie să fie LOW la pornire
- Toți pinii GPIO sunt la 3.3V (NU toleranți la 5V!)
- GPIO16 este special: poate trezi din deep sleep, dar nu are întrerupere/PWM
- Pinii TX și RX sunt utilizați pentru programare/depanare serială

Interfețe periferice ESP8266

- SPI (Interfață periferică serială)
  - O interfață SPI hardware
  - Poate fi emulat prin software pe orice GPIO pentru SPI suplimentar
- I^2C (Circuit inter-integrat)
  - Implementat prin software (bit-banging)
  - Se pot folosi oricare doi pini GPIO.
- UART (Receptor/Transmițător asincron universal)
  - Două interfețe UART (UART0 și UART1)
  - UART1 este doar TX
- ADC: 1 canal, rezoluție de 10 biți (interval 0-1V)
- I2S- Un singur controler I2S (un singur dispozitiv periferic)
  - Utilizează DMA pentru un transfer eficient de date fără intervenția procesorului

Capabilități WiFi ESP8266

- Suport IEEE 802.11 b/g/n (doar 2.4 GHz)
- Trei moduri de operare:
  - Stație (STA): Conectare la rețeaua WiFi existentă
  - Punct de acces (AP): Crearea propriei rețele Wi-Fi
  - Ambele (AP+STA): Moduri simultane
- Stivă TCP/IP completă (lwIP)
- Securitate WPA/WPA2
- Până la 5 conexiuni TCP/UDP simultane
- Suportă protocoalele DHCP, DNS, HTTP, MQTT, CoAP

Gestionarea energiei ESP8266

Mod	Consum de curent	Descriere
Activ (TX)	~170 mA	WiFi în transmisie
Activ (RX)	~60 mA	WiFi în recepție
Modem Sleep	~15 mA	CPU pornit, WiFi oprit
Light Sleep	~0.9 mA	CPU suspendat, WiFi oprit
Deep Sleep	~20 µA	Totul oprit cu excepția RTC

Trezire din deep sleep: GPIO16 conectat la pinul RST

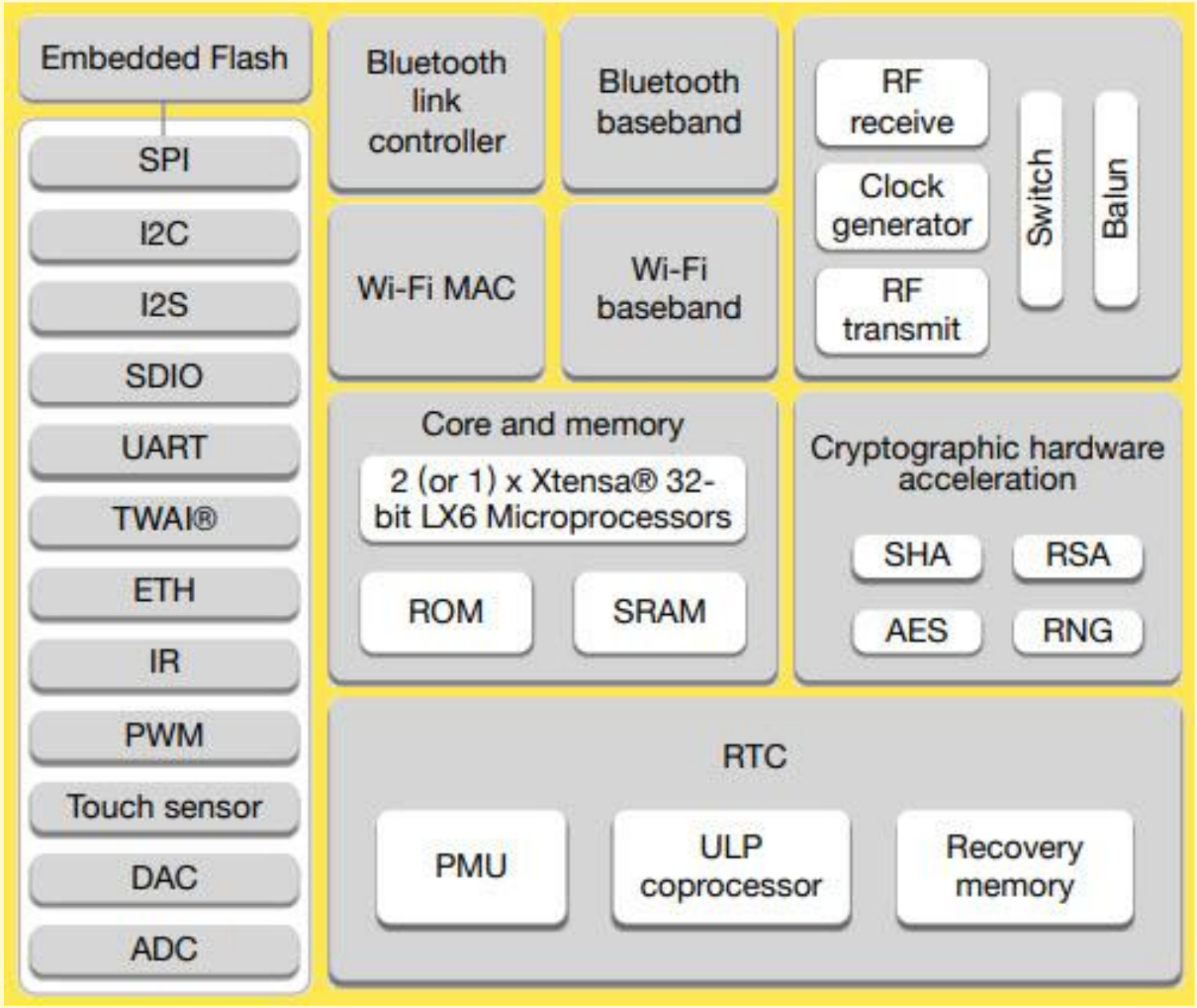
Introducere în ESP32

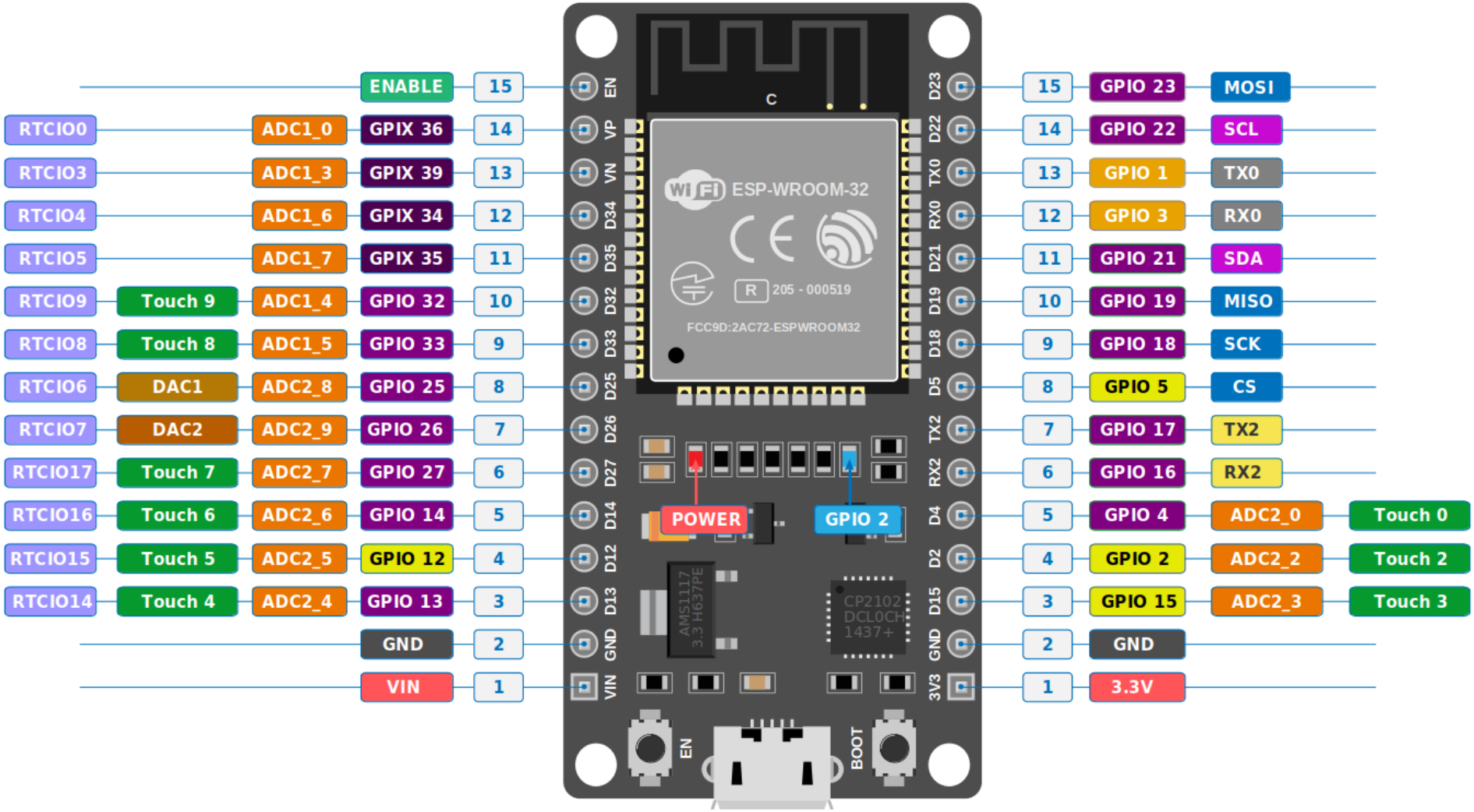
- Succesor al ESP8266, lansat în 2016
- ESP32 este mai puternic și mai bogat în funcționalități.
- CPU dual-core: Tensilica Xtensa LX6
- Funcționează la 160/240 MHz
- Wi-Fi + Bluetooth 4.2 (BLE și Classic)
- Mai mult RAM și mai multe periferice
- Funcții de securitate mai bune (criptare hardware)

Specificații cheie ESP32

Specificație	Valoare
CPU	Dual Xtensa LX6 @ 160/240 MHz
RAM	520 KB SRAM
Flash	4 MB până la 16 MB (extern)
Pini GPIO	34 (multifuncțional)
ADC	18 canale, 12 biți
DAC	2 canale, 8 biți
WiFi	802.11 b/g/n (2.4 GHz)
Bluetooth	4.2 BR/EDR și BLE
Tensiune de operare	3.3V

Diagrama bloc ESP32




ENABLE
15
EN
RTCIO0
ADC1_0
GPIX 36
14
VP
RTCIO3
ADC1_3
GPIX 39
13
VN
RTCIO4
ADC1_6
GPIX 34
12
D34
RTCIO5
ADC1_7
GPIX 35
11
D35
RTCIO9
Touch 9
ADC1_4
GPIO 32
10
D32
RTCIO8
Touch 8
ADC1_5
GPIO 33
9
D33
RTCIO6
DAC1
ADC2_8
GPIO 25
8
D25
RTCIO7
DAC2
ADC2_9
GPIO 26
7
D26
RTCIO17
Touch 7
ADC2_7
GPIO 27
6
D27
RTCIO16
Touch 6
ADC2_6
GPIO 14
5
D14
RTCIO15
Touch 5
ADC2_5
GPIO 12
4
D12
RTCIO14
Touch 4
ADC2_4
GPIO 13
3
D13
GND
2
GND
VIN
1
VIN
C
ESP-WROOM-32
R 205 - 000519
FCC9D:2AC72-ESPWROOM32
POWER
GPIO 2
AMS1117 3.3 H637PE
CP2102 DCL0CH 1437+
EN
BOOT
D23
15
GPIO 23
MOSI
D22
14
GPIO 22
SCL
TX0
13
GPIO 1
TX0
RX0
12
GPIO 3
RX0
D21
11
GPIO 21
SDA
D19
10
GPIO 19
MISO
D18
9
GPIO 18
SCK
D5
8
GPIO 5
CS
TX2
7
GPIO 17
TX2
RX2
6
GPIO 16
RX2
D4
5
GPIO 4
ADC2_0
Touch 0
D2
4
GPIO 2
ADC2_2
Touch 2
D15
3
GPIO 15
ADC2_3
Touch 3
GND
2
GND
3V3
1
3.3V

Xtensa LX6 dual-core

- Două nuclee like-RISC Xtensa LX6 identice pe 32 de biți
- Ceas configurabil: 80, 160 sau 240 MHz
- Execuție independentă a sarcinilor pe fiecare nucleu
- Utilizare tipică: Core 0 pentru Wi-Fi/Bluetooth, Core 1 pentru aplicație
- Planificatorul FreeRTOS gestionează ambele nuclee
- Fiecare nucleu are 16 registre de uz general pe 32 de biți
- Operații hardware de înmulțire, împărțire și MAC
- Fără virgulă mobilă hardware (emulare software)

Arhitectura memoriei ESP32

- SRAM intern: 520 KB total
  - SRAM0: 192 KB (instrucțiuni și date)
  - SRAM1: 128 KB (instrucțiuni și date)
  - SRAM2: 200 KB (doar date)
- Flash SPI extern: 4 MB până la 16 MB tipic
- RAM SPI extern (PSRAM): Opțional, până la 8 MB
- Memorie RTC rapidă: 8 KB (pentru stocare în deep sleep)
- Memorie RTC lentă: 8 KB (acces cu consum ultra-redus)
- E-Fuse: 1024 biți pentru securitate și configurare

GPIO și pini ESP32

- 34 de pini GPIO programabili (GPIO0-GPIO39)
- GPIO34-39 sunt doar intrare (fără pull-up/down intern)
- Logică la 3.3V (NU tolerant la 5V)
- Majoritatea pinilor au funcții multiple și pot fi remapați.
- Pini de configurare (strapping - comportament special la pornire):
  - GPIO0, GPIO2, GPIO5, GPIO12, GPIO15
- Rezistoare pull-up/pull-down integrate (configurabile prin software)
- Pini cu senzor tactil: GPIO0, 2, 4, 12-15, 27, 32, 33

Interfețe periferice ESP32

- SPI: 4 controlere SPI independente (2 pentru Flash, 2 pentru uz general)
- I^2C: 2x interfețe I^2C (mod master sau slave)
- I^2S: 2x interfețe I^2S (pentru aplicații audio)
- UART: 3x UART interfaces (full-duplex)
- CAN 2.0: 1 controler magistrală CAN (auto/industrial)
- Ethernet MAC: Pentru rețea cu fir (cu PHY extern)
- SD/SDIO/MMC host: Interfață card SD
- Periferic telecomandă: IR TX/RX

ESP32 ADC și DAC

- ADC (Convertor analog-digital)
  - 18 canale, rezoluție de 12 biți
  - Două ADC-uri SAR (ADC1 și ADC2)
  - Nu se poate utiliza ADC2 când Wi-Fi este activ
  - Interval tensiune de intrare: 0-3.3V (cu atenuare)
  - Setări de atenuare integrate pentru diferite intervale

- DAC (Convertor)
  - 2 canale, rezoluție de 8 biți
  - GPIO25 și GPIO26
  - Interval de ieșire: 0-3.3V

ESP32 Wi-Fi și Bluetooth

- Wi-Fi:
  - IEEE 802.11 b/g/n (2.4 GHz)
  - Moduri STA, AP și STA+AP
  - Securitate WPA/WPA2/WPA3

- Bluetooth 4.2:
  - BLE (Bluetooth Low Energy)
  - Bluetooth Classic (BR/EDR)
  - Poate funcționa ca master sau slave
  - Profiluri HID, SPP, A2DP suportate
- Wi-Fi și Bluetooth pot coexista (partajează RF)

Funcții de securitate ESP32

- Accelerare criptografică hardware:
  - AES, SHA-2, RSA, ECC (Curbă eliptică)
- Secure Boot: Verificarea autenticității firmware-ului la pornire
- Criptare Flash: Criptarea codului aplicației în Flash
- eFuse: Biți programabili o singură dată (OTP) pentru chei și configurare
- Generator de numere aleatoare reale (TRNG)-True Random Number Generator
- Periferice de semnătură digitală pentru autentificare
- Stocare securizată pentru chei și certificate

Gestionarea energiei ESP32

Mod	Consum de curent	Descriere
Activ (WiFi TX)	~160-260 mA	Ambele nuclee + WiFi în transmisie
Activ (WiFi RX)	~80-95 mA	Ambele nuclee + WiFi în recepție
Modem Sleep	~20-30 mA	CPU-uri pornite, WiFi/BT oprit
Light Sleep	~0.8 mA	CPU-uri suspendate, RTC pornit
Deep Sleep	~10 µA	Doar RTC + coprocesor ULP
Hibernare	~5 µA	Doar timer-ul RTC activ

Coprocesorul ULP (Ultra-Low-Power) poate rula în timpul deep sleep

Senzor tactil ESP32

- 10 GPIO-uri cu senzor tactil capacitiv
- Pini tactili: GPIO0, 2, 4, 12, 13, 14, 15, 27, 32, 33
- Detectează modificări ale capacitanței la atingere
- Nu necesită componente externe
- Poate trezi ESP32 din deep sleep la atingere
- Prag de sensibilitate ajustabil
- Util pentru:
  - Butoane tactile și slidere
  - Detectarea proximității
  - Declanșatoare de trezire cu consum redus

Comparație: ESP8266 vs ESP32

Caracteristică	ESP8266	ESP32
CPU	Xtensa LX106 singur	Xtensa LX6 dual
Frecvență de ceas	80/160 MHz	160/240 MHz
RAM	~80 KB	520 KB
Flash	512 KB - 4 MB	4 MB - 16 MB
GPIO	17 pins	34 pins
ADC	1 ch, 10-bit	18 ch, 12-bit
DAC	Fără	2 ch, 8-bit
WiFi	802.11 b/g/n	802.11 b/g/n
Bluetooth	Fără	4.2 BR/EDR + BLE
Senzor tactil	Fără	10 pini
Preț	$2-5	$4-10

Prezentare generală a instrumentelor de dezvoltare

- **Arduino IDE** (cel mai ușor pentru începători)
  - IDE familiar, suport comunitar imens
  - Pachete de suport pentru plăci ESP8266 și ESP32

- **MicroPython** (Python pentru microcontrolere)
  - REPL interactiv pentru prototipare rapidă
  - Sintaxă mai simplă decât C/C++

- **ESP-IDF** (Espressif IoT Development Framework)
  - SDK oficial de la Espressif
  - Control total, toate funcționalitățile, curbă de învățare mai abruptă

- **PlatformIO** (extensie IDE profesională)
  - Management avansat de proiecte și depanare

Configurarea Arduino IDE

- Pasul 1: Instalarea Arduino IDE
  - Descărcați de pe arduino.cc
- Pasul 2: Adăugarea URL-ului Board Manager ESP8266
  - File → Preferences → Additional Board Manager URLs
  - http://arduino.esp8266.com/stable/package_esp8266com_index.json
- Pasul 3: Instalarea plăcilor ESP8266/ESP32
  - Tools → Board → Boards Manager
  - Căutați "ESP8266" sau "ESP32" și instalați
- Pasul 4: Selectarea plăcii și a portului COM
  - Tools → Board → Selectați modulul ESP
  - Tools → Port → Selectați portul serial USB

Arduino IDE - Exemplu Blink

```
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

LED_BUILTIN: GPIO2 (ESP8266), GPIO2 (ESP32)

Arduino - Conexiune Wi-Fi

```
#include <ESP8266WiFi.h>  // or <WiFi.h> for ESP32

const char* ssid = "your_SSID";
const char* password = "your_PASSWORD";

void setup()
{
  Serial.begin(115200);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }

  Serial.println("WiFi connected!");
  Serial.print("IP: ");
  Serial.println(WiFi.localIP());
}
void loop() { }
```

Arduino - Server Web simplu

```
#include <ESP8266WebServer.h>

ESP8266WebServer server(80);

void handleRoot() {
  server.send(200, "text/html",
    "<h1>ESP8266 Web Server</h1>");
}

void setup() {
  WiFi.begin(ssid, password);
  // wait for connection...

  server.on("/", handleRoot);
  server.begin();
}

void loop() {
  server.handleClient();
}
```

Prezentare generală Micro Python

- Implementare Python 3 pentru microcontrolere
- REPL interactiv (Buclă Citire-Evaluare-Afișare)
  - Testarea codului imediat, fără compilare/încărcare
- Subset al bibliotecii standard Python
- Module specifice ESP: *machine, network, esp*
- Avantaje:
  - Sintaxă mai ușoară decât C/C++
  - Prototipare și testare rapidă
  - Tipizare dinamică, gestionare automată a memoriei
- Dezavantaje: Mai lent decât C, amprentă de memorie mai mare

Read-Eval-Print Loop (Bucle Citire-Evaluare-Afisare)

Instalarea Micro Python

- Pasul 1: Instalarea esptool
  - pip install esptool
- Pasul 2: Descărcarea firmware-ului Micro Python
  - De pe micropython.org/download
  - Alegeți versiunea ESP8266 sau ESP32
- Pasul 3: Ștergerea flash-ului
  - esptool.py --port /dev/ttyUSB0 erase_flash
- Pasul 4: Programarea firmware-ului
  - esptool.py --port /dev/ttyUSB0 write_flash
    -z 0x1000 esp8266-*.bin
- Pasul 5: Conectarea cu terminalul serial (PuTTY, screen)
  - Rata baud 115200

MicroPython - Exemplu REPL

```
>>> import machine
>>> pin = machine.Pin(2, machine.Pin.OUT)
>>> pin.on()
>>> pin.off()

>>> import network
>>> wlan = network.WLAN(network.STA_IF)
>>> wlan.active(True)
>>> wlan.connect('SSID', 'password')
>>> wlan.ifconfig()
('192.168.1.100', '255.255.255.0',
 '192.168.1.1', '8.8.8.8')
```

Micro Python - Exemplu Blink

```
import machine
import time

led = machine.Pin(2, machine.Pin.OUT)

while True:
    led.on()
    time.sleep(1)
    led.off()
    time.sleep(1)
```

Salvați ca main.py pentru rulare automată la pornire

Micro Python – Conexiune Wi-Fi

```
import network

wlan = network.WLAN(network.STA_IF)
wlan.active(True)

if not wlan.isconnected():
    print('Connecting to network...')
    wlan.connect('SSID', 'password')
    while not wlan.isconnected():
        pass

print('Network config:', wlan.ifconfig())
```

Prezentare generală ESP-IDF

- Espressif IoT Development Framework
- SDK oficial de la Espressif Systems
- Bazat pe FreeRTOS (sistem de operare în timp real)
- Acces complet la toate funcționalitățile hardware
- Mediu de dezvoltare profesional
- Include:
  - Lanț de instrumente (compilator GCC)
  - Sistem de compilare (CMake)
  - Biblioteci și API-uri complete
  - Exemple și documentație
- Curbă de învățare mai abruptă decât Arduino

PWM - Modulația impulsurilor în lățime

- Utilizat pentru ieșire de tip analogic de la pini digitali
- ESP8266:
  - PWM software pe toți pinii GPIO
  - Rezoluție de 10 biți (0-1023)
  - Frecvență tipică: 1 kHz
- ESP32:
  - 16 canale PWM independente (LEDC)
  - Rezoluție de până la 16 biți
  - Frecvență configurabilă (1 Hz - 40 MHz)
- Aplicații: Dimmer LED, controlul vitezei motoarelor, controlul servo

Interfața I^2C

- Protocol de comunicație serială pe două fire
- SDA (Date seriale), SCL (Ceas serial)
- Dispozitive multiple pe aceeași magistrală (până la 127)
- ESP8266: I^2C software (bit-banging)
  - Implicit: GPIO4 (SDA), GPIO5 (SCL)
  - Se pot folosi oricare pini GPIO
- ESP32: Controlere I^2C hardware
  - Două interfețe I^2C
  - Pinii sunt configurabili
- Dispozitive I^2C comune: Ecrane OLED, senzori, RTC-uri

Interfața SPI

- Protocol de comunicație serială pe patru fire
- MOSI (Master Out Slave In)
- MISO (Master In Slave Out)
- SCK (Ceas serial)
- CS/SS (Selecție cip / Selecție slave)
- ESP8266: O interfață SPI hardware
  - Utilizat în principal pentru Flash extern
- ESP32: Patru controlere SPI
  - Două dedicate Flash/PSRAM
  - Două disponibile pentru uz general (HSPI, VSPI)
- Mai rapid decât I^2C, utilizat frecvent pentru SD carduri, ecrane

Interfața UART

- Comunicație serială asincronă
- Două fire: TX (Transmisie), RX (Recepție)
- Fără semnal de ceas (asincron)
- ESP8266:
  - UART0: TX=GPIO1, RX=GPIO3 (utilizat pentru programare)
  - UART1: Doar TX pe GPIO2
- ESP32:
  - Trei interfețe UART
  - Pinii pot fi remapați
- Configurabil: rată baud, paritate, biți de stop, biți de date
- Utilizări comune: GPS, senzori, depanare, comunicație

Wi-Fi - Mod Stație (STA)

- ESP se conectează la rețeaua Wi-Fi existentă (router)
- Funcționează ca dispozitiv client
- Primește adresa IP de la un server DHCP
- Poate accesa internetul prin router
- Cazuri de utilizare:
  - Senzor IoT care trimite date în cloud
  - Dispozitiv smart home controlat prin aplicație
  - Stație meteo care încarcă date
  - Web scraping și cereri API
- Cel mai comun mod pentru aplicații IoT

Wi-Fi - Mod Punct de acces (AP)

- ESP creează propria rețea Wi-Fi
- Alte dispozitive se pot conecta la ESP
- ESP atribuie adrese IP dispozitivelor conectate
- Fără acces la internet (cu excepția modului AP+STA)
- Configurare:
  - SSID și parolă personalizate
  - Selecția canalului Wi-Fi
  - Opțiune SSID ascuns
- Cazuri de utilizare:
  - Portal de configurare pentru configurarea inițială
  - Comunicare directă între dispozitive
  - Server web autonom fără router

Bluetooth pe ESP32

- Suport Bluetooth 4.2 (BR/EDR + BLE)
- BLE (Bluetooth Low Energy):
  - Consum de energie foarte redus
  - Utilizat pentru balizaje, senzori, dispozitive purtabile
  - Servicii GATT (Generic Attribute Profile)
- Bluetooth Classic (BR/EDR):
  - Rate de transfer mai mari
  - Streaming audio (profil A2DP)
  - Comunicație serială (profil SPP)
- Poate opera ca master sau slave
- Coexistă cu Wi-Fi (RF partajat, multiplexat în timp)

Modul Deep Sleep

- Mod cu consum ultra-redus pentru aplicații pe baterii
- ESP8266:
  - Consum de ~20 µA
  - Doar RTC (Ceas în timp real) rămâne activ
  - Trezire prin timer: ESP.deepSleep (µs)
  - Necesită GPIO16 conectat la pinul RST

- ESP32:
  - Consum de ~10 µA
  - Surse multiple de trezire: timer, atingere, GPIO extern
  - Memorie RTC păstrată (până la 8 KB)
  - Coprocesorul ULP poate rula în timpul deep sleep
- Sistemul se resetează la trezire (ca la pornire)

Temporizatoare și întreruperi

- Temporizatoare hardware:
  - ESP8266: Un temporizator hardware
  - ESP32: Patru temporizatoare hardware pe 64 de biți
  - Pot declanșa întreruperi la intervale precise
- Întreruperi GPIO:
  - Declanșare la schimbarea stării pinului
  - Moduri: RISING, FALLING, CHANGE, LOW, HIGH
  - ESP8266: Toți pinii GPIO suportă întreruperi
  - ESP32: Toți pinii GPIO suportă întreruperi
- Reguli ISR (Rutina de servire a întreruperii):
  - Mențineți codul ISR scurt și rapid
  - Folosiți ICACHE_RAM_ATTR (ESP8266) sau IRAM_ATTR (ESP32)

Ceas în timp real (RTC)

- Atât ESP8266 cât și ESP32 au RTC intern
- Funcții:
  - Menținerea timpului în timpul deep sleep
  - Temporizator de trezire
  - Memorie RTC pentru retenția datelor în sleep
- Fără baterie de rezervă (pierde ora la întreruperea alimentării)
- Sincronizarea timpului:
  - NTP (Network Time Protocol) prin Wi-Fi
  - Module RTC externe (DS1307, DS3231) prin I^2C
- Funcții RTC ESP32:
  - GPIO RTC pentru trezire
  - Coprocesorul ULP poate accesa perifericele RTC

Actualizări OTA

- Actualizări firmware Over-The-Air prin Wi-Fi
- Nu este necesar cablu USB pentru actualizări
- Actualizarea dispozitivelor instalate la distanță.
- Două metode OTA:
  - Arduino OTA: Simplu, integrat în Arduino IDE
  - HTTP OTA: Descărcare firmware de pe server web
- Partiționarea Flash:
  - Două partiții de aplicație (curentă + actualizare)
  - Bootloader-ul comută între partiții
- Considerații de securitate:
  - HTTPS pentru descărcări securizate
  - Verificarea semnăturii firmware-ului

Aplicații comune - IoT

- **Monitorizarea mediului:**
  - Senzori de temperatură, umiditate, calitate a aerului
  - Înregistrarea datelor în cloud (ThingSpeak, Blynk, AWS)

- **Automatizare Smart Home:**
  - Controlul luminilor, termostate, încuietori
  - Integrare cu Alexa, Google Home

- **Control și monitorizare la distanță:**
  - Camere de securitate, automatizări porți garaj
  - Sisteme de irigare plante

- **Dispozitive portabile și trackere de fitness** (ESP32 + BLE)

- **Rețea Mesh (ESP32 ESP-MESH)**

Aplicații - Prototipuri

- Prototipare IoT rapidă:
  - Ciclu de dezvoltare rapid
  - Suport comunitar și de biblioteci extins
  - Cost redus pentru proof-of-concept

- Proiecte educaționale:
  - Învățarea programării Wi-Fi/Bluetooth
  - Concepte și protocoale IoT
  - Dezvoltarea sistemelor incorporate

- Achiziție de date:
  - Rețele de senzori
  - Înregistratoare de date la distanță

- Dispozitive Gateway: Punte între diferite protocoale

Observatii

- Alimentare:
  - Alimentare stabilă de 3.3V (nu 5V!)
  - Capacitate suficientă de curent (300+ mA vârf)
  - Condensatoare de decuplare (10µF + 100nF)

- Conexiune Wi-Fi:
  - Gestionarea eșecurilor de conexiune
  - Implementarea logicii de reconectare
  - Utilizarea timer-elor watchdog pentru recuperare din blocări

- Gestionarea memoriei:
  - Monitorizarea heap-ului liber cu ESP.getFreeHeap()
  - Evitarea scurgerilor de memorie și fragmentării

- Utilizați deep sleep pentru proiecte alimentate pe baterie

- Implementați OTA pentru actualizări facile ale dispozitivelor instalate

Depanare și rezolvarea problemelor

- Monitor serial:
  - Utilizați Serial.print() pentru ieșire de depanare
  - Verificați rata baud (de obicei 115200)
- Probleme frecvente:
  - Bucle de boot: Verificați alimentarea, stările GPIO
  - Nu se conectează la Wi-Fi: Verificați credențialele, puterea semnalului
  - Blocări: Depășire memorie, depășire stivă, reset watchdog
- Decodor de excepții:
  - Decodarea urmelor stivei la blocaj
  - Disponibil ca plugin Arduino IDE
- Utilizați ESP Exception Decoder pentru analiza blocajelor
- Verificați problemele GitHub și forumurile ESP8266/ESP32

Resurse și documentație

- Documentație oficială:
  - Espressif Systems: www.espressif.com
  - ESP8266 Arduino Core: github.com/esp8266/Arduino
  - ESP32 Arduino Core: github.com/espressif/arduino-esp32
  - ESP-IDF: docs.espressif.com/projects/esp-idf

- Micro Python:
  - micropython.org/download (firmware)
  - docs.micropython.org/en/latest/esp8266

- Comunitate:
  - Forumul comunității ESP8266
  - Reddit: r/esp8266, r/esp32
  - Arduino Forum

Module și plăci populare

- Module ESP8266:
  - ESP-01: Cel mai mic, 2 GPIO, de bază
  - **ESP-12E/F: Cel mai popular, 11 GPIO**
  - NodeMCU: Placă de dezvoltare cu USB, regulator de tensiune
  - Wemos D1 Mini: Placă de dezvoltare compactă, factor de formă Arduino

- Module ESP32:
  - ESP32-WROOM: Modul de uz general
  - ESP32-WROVER: Cu PSRAM (RAM suplimentar)
  - ESP32 DevKitC: Placă oficială de dezvoltare de la Espressif
  - ESP32-CAM: Cu cameră și card SD
  - TTGO T-Display: Cu ecran TFT integrat

Variante ESP32

- ESP32-S2:
  - Single-core, fără Bluetooth, suport USB OTG
- ESP32-S3:
  - Dual-core, accelerare AI, USB OTG, doar BLE
- ESP32-C3:
  - Nucleu RISC-V (nu Xtensa), Wi-Fi + BLE, cost redus
- ESP32-C6:
  - RISC-V, Wi-Fi 6, Bluetooth 5.3, Zigbee/Thread
- ESP32-H2:
  - Bluetooth 5.2, Zigbee, Thread (fără Wi-Fi)
- Fiecare variantă vizează cazuri de utilizare și cerințe specifice

Rezumat

- ESP8266 și ESP32 au revoluționat dezvoltarea IoT
- Cost redus, cu WiFi, ușor de programat
- ESP8266: Single-core, bun pentru proiecte simple
- ESP32: Dual-core, Bluetooth, mai multe funcționalități
- Opțiuni multiple de dezvoltare:
  - Arduino IDE (cel mai ușor)
  - Micro Python (bazat pe Python)
  - ESP-IDF (control total)
- Suport periferic bogat: GPIO, SPI, I^2C, UART, I^2S, ADC, PWM
- Funcții de gestionare a energiei pentru operare pe baterie
- Comunitate și ecosistem imens