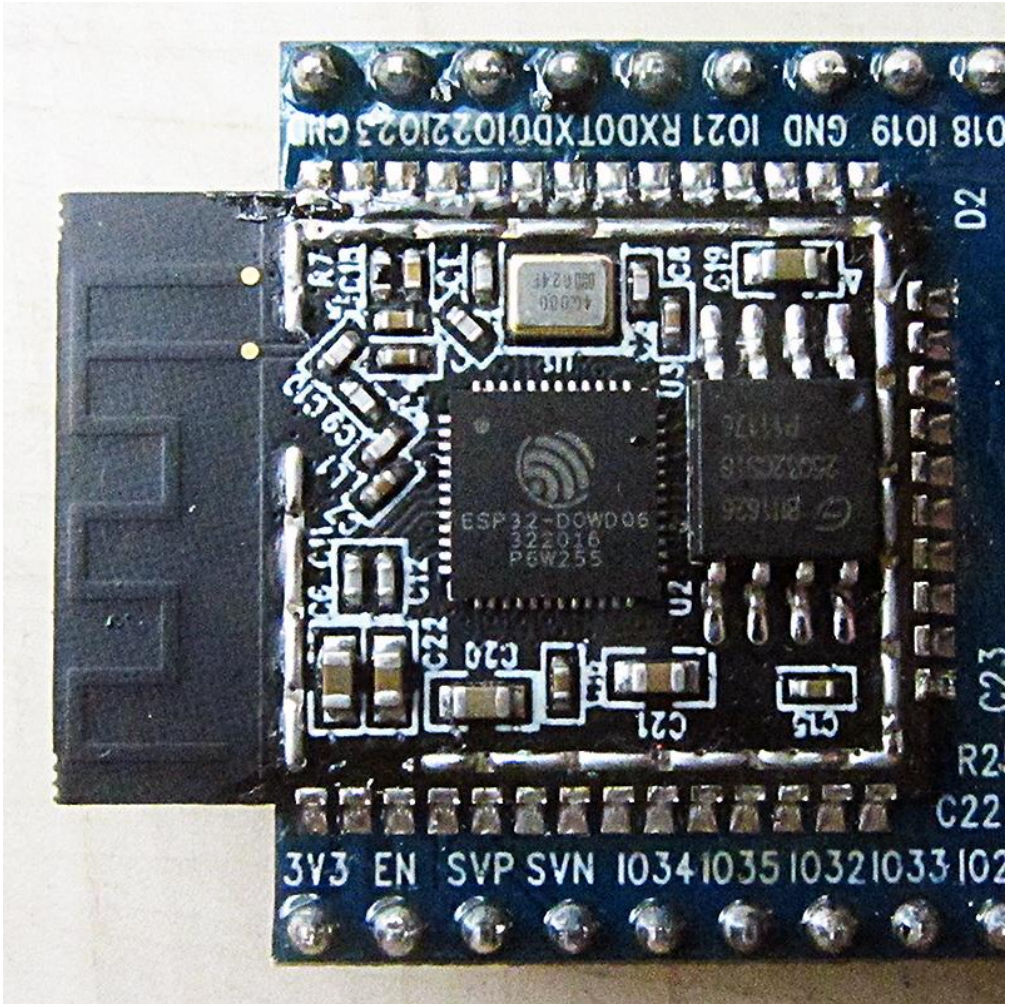
INTERNET PAMETINIH UREĐAJA

prof. dr Dejan S. Aleksić

Prirodno-matematički fakultet, Niš

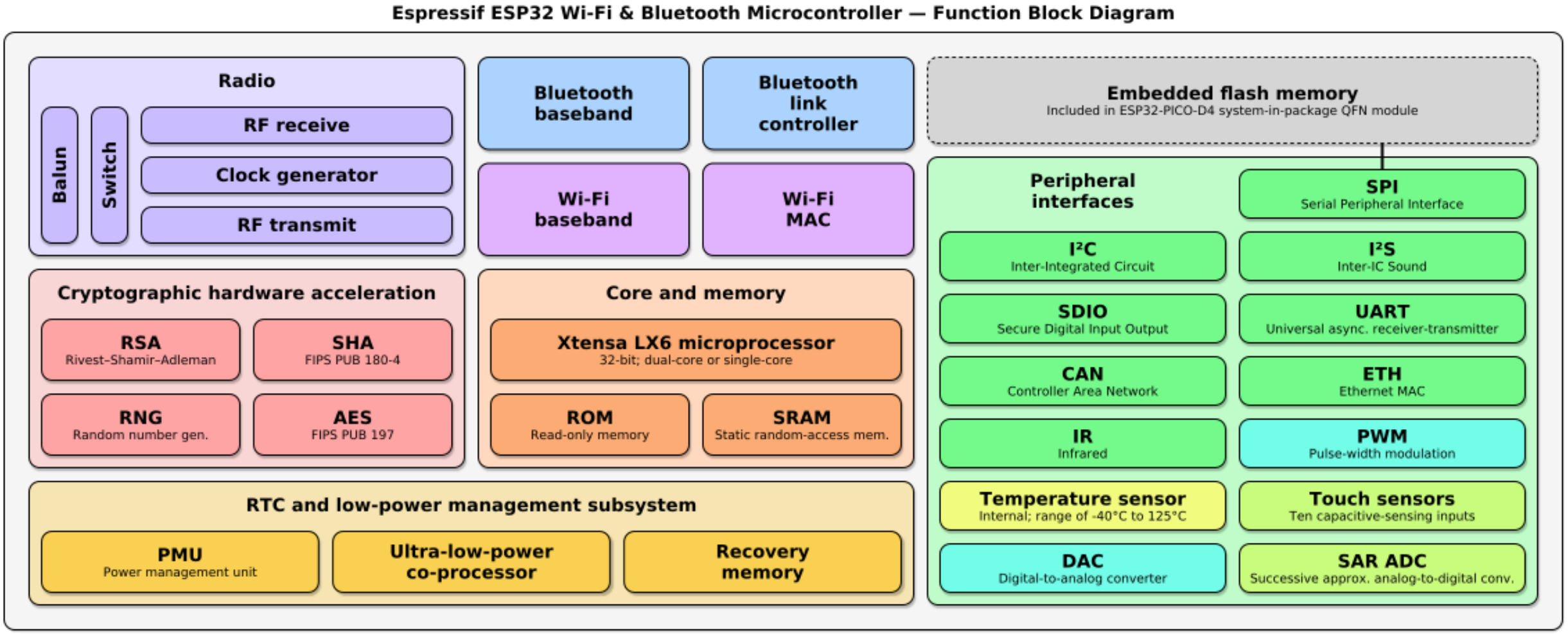
01. ESP 32

ESP 32 – INTERNA STRUKTURA



By Brian Krent - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=57745131

ESP 32 – INTERNA STRUKTURA



By Brian Krent (talk · contribs) - Own work, CC0, https://commons.wikimedia.org/w/index.php?curid=72304119

ESP 32 – INTERNA STRUKTURA

Processors:
- CPU: Xtensa dual-core (or single-core) 32-bit LX6 microprocessor, operating at 160 or 240 MHz and performing at up to 600 DMIPS
- Ultra low power (ULP) co-processor

Memory: 520 KiB SRAM

Wireless connectivity:
- Wi-Fi: 802.11 b/g/n
- Bluetooth: v4.2 BR/EDR and BLE

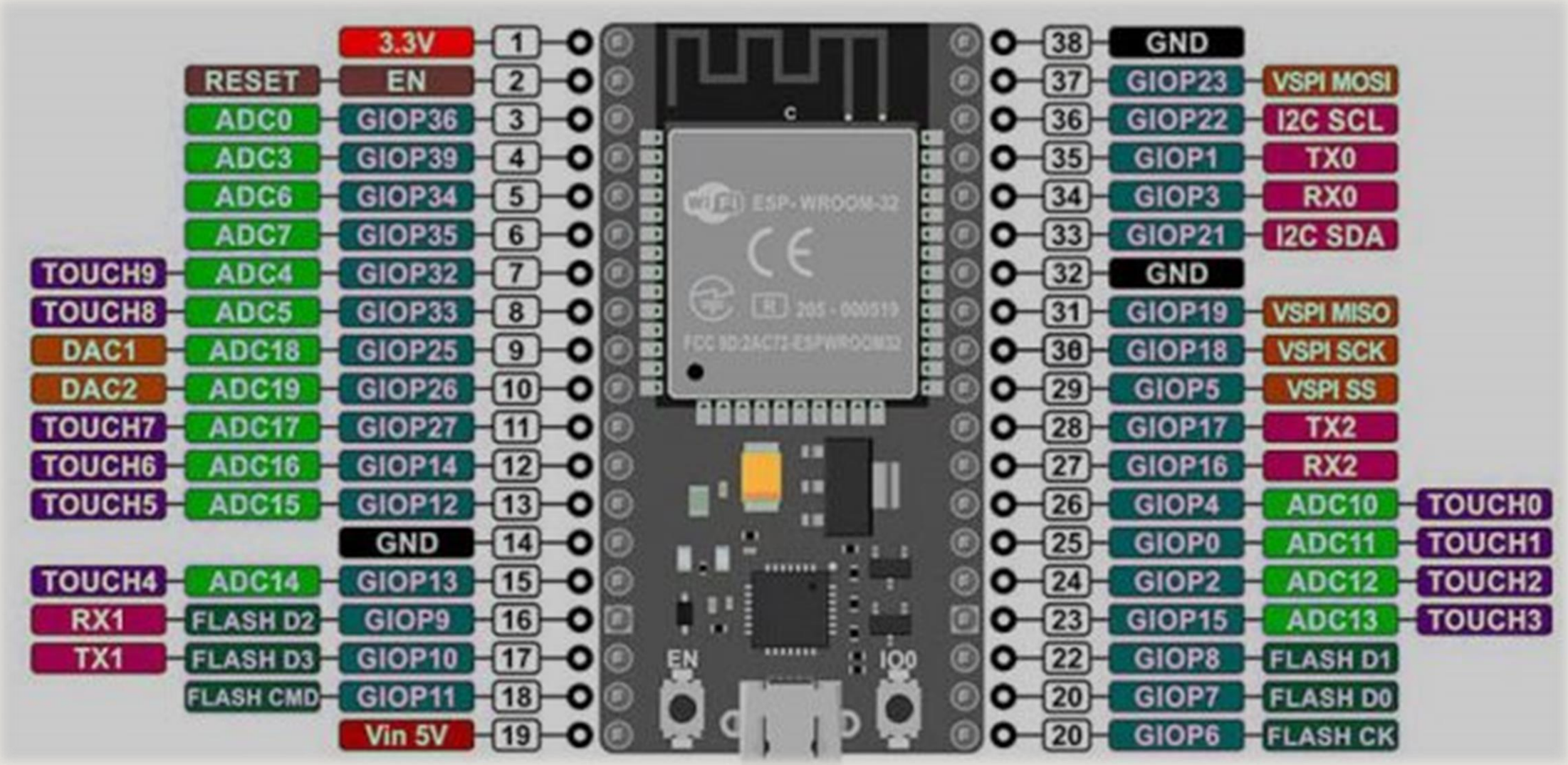
Peripheral interfaces:
- 12-bit SAR ADC up to 18 channels
- 2 × 8-bit DACs
- 10 × touch sensors (capacitive sensing GPIOs)
- Temperature sensor
- 4 × SPI
- 2 × I²S interfaces
- 2 × I²C interfaces
- 3 × UART
- SD/SDIO/CE-ATA/MMC/eMMC host controller
- SDIO/SPI slave controller
- Ethernet MAC interface with dedicated DMA and IEEE 1588 Precision Time Protocol support
- CAN bus 2.0
- Infrared remote controller (TX/RX, up to 8 channels)
- Motor PWM
- LED PWM (up to 16 channels)
- Hall effect sensor
- Ultra low power analog pre-amplifier

Security:
- IEEE 802.11 standard security features all supported, including WFA, WPA/WPA2 and WAPI
- Secure boot
- Flash encryption
- 1024-bit OTP, up to 768-bit for customers
- Cryptographic hardware acceleration: AES, SHA-2, RSA, elliptic curve cryptography (ECC), random number generator (RNG)

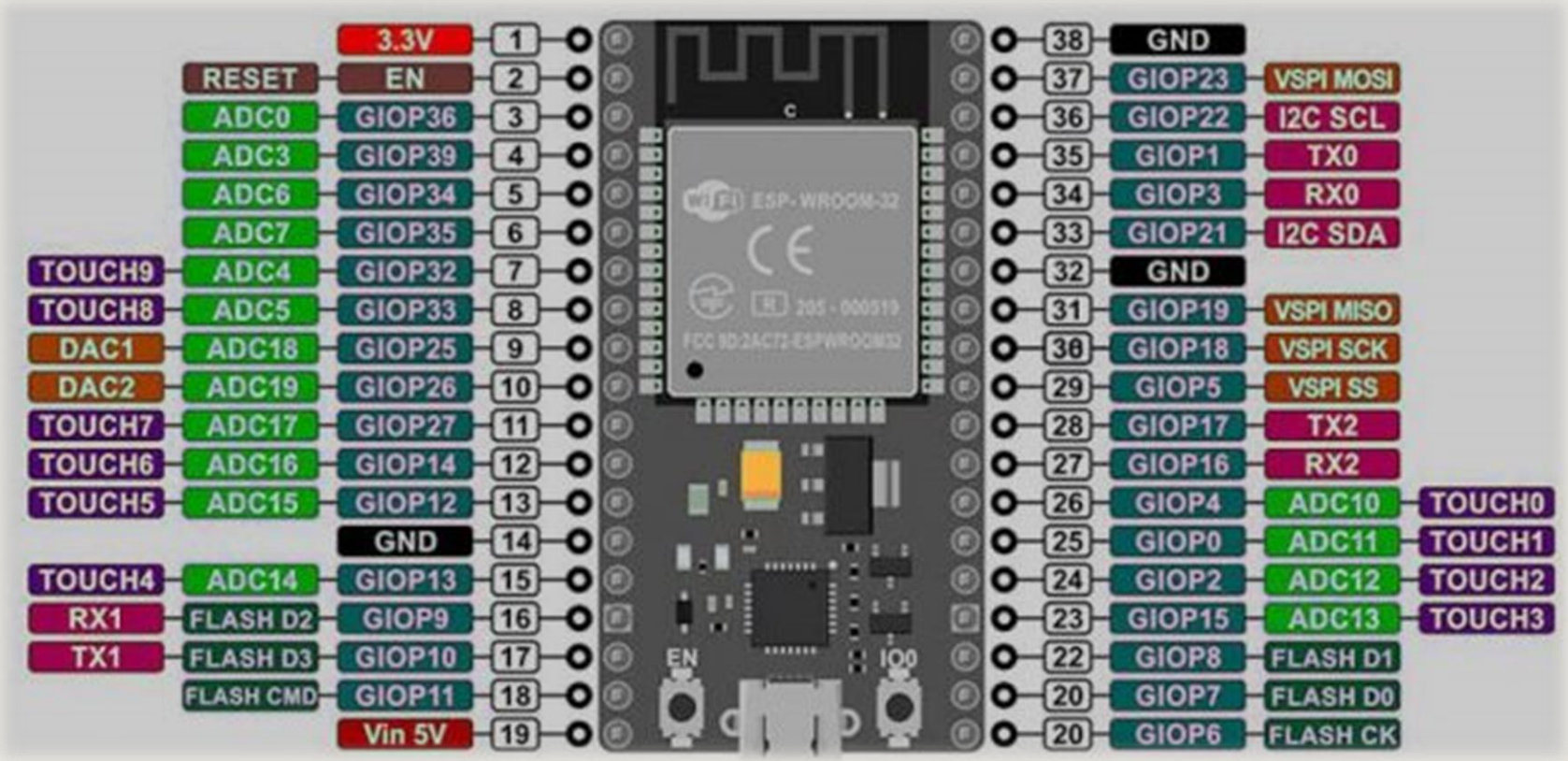
Power management:
- Internal low-dropout regulator
- Individual power domain for RTC
- 5uA deep sleep current
- Wake up from GPIO interrupt, timer, ADC measurements, capacitive touch sensor interrupt

preuzeto sa: https://en.wikipedia.org/wiki/ESP32

ESP 32 SOC


3.3V 1
RESET EN 2
ADC0 GIOP36 3
ADC3 GIOP39 4
ADC6 GIOP34 5
ADC7 GIOP35 6
TOUCH9 ADC4 GIOP32 7
TOUCH8 ADC5 GIOP33 8
DAC1 ADC18 GIOP25 9
DAC2 ADC19 GIOP26 10
TOUCH7 ADC17 GIOP27 11
TOUCH6 ADC16 GIOP14 12
TOUCH5 ADC15 GIOP12 13
GND 14
TOUCH4 ADC14 GIOP13 15
RX1 FLASH D2 GIOP9 16
TX1 FLASH D3 GIOP10 17
FLASH CMD GIOP11 18
Vin 5V 19
38 GND
37 GIOP23 VSPI MOSI
36 GIOP22 I2C SCL
35 GIOP1 TX0
34 GIOP3 RX0
33 GIOP21 I2C SDA
32 GND
31 GIOP19 VSPI MISO
30 GIOP18 VSPI SCK
29 GIOP5 VSPI SS
28 GIOP17 TX2
27 GIOP16 RX2
26 GIOP4 ADC10 TOUCH0
25 GIOP0 ADC11 TOUCH1
24 GIOP2 ADC12 TOUCH2
23 GIOP15 ADC13 TOUCH3
22 GIOP8 FLASH D1
20 GIOP7 FLASH D0
20 GIOP6 FLASH CK
ESP-WROOM-32
EN
IO0

ESP 32 SOC



- 18 Analog-to-Digital Converter (ADC) channels
- 3 SPI interfaces
- 3 UART interfaces
- 2 I2C interfaces
- 16 PWM output channels
- 2 Digital-to-Analog Converters (DAC)
- 2 I2S interfaces
- 10 Capacitive sensing GPIOs

ESP 32 SOC

GPIO	Ulaz	Izlaz	Napomena
0	Pulled up	✓	outputs PWM signal at boot, must be LOW to enter flashing mode
1	Tx pin	✓	debug output at boot
2	✓	✓	connected to on-board LED, must be left floating or LOW to enter flashing mode
3	✓		
4			
5			
6		✗	
7			
8			
9			
10			
11			
12			
13			
14			

ESP 32 SOC

GPIO	Ulaz	Izlaz	Napomena
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			

ESP 32 SOC

GPIO	Ulaz	Izlaz	Napomena
0			
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			

ESP 32 – WIFI PODRŠKA

- WiFi modovi: station, access point, station + access point
- Skeniranje WiFi mreža
- Povezivanje na WiFi mrežu
- Pribavljanje statusa WiFi konekcije
- Jačina signala WiFi konekcije
- Pribavljanje IP adrese
- Postavljanje statičke IP adrese
- Prekidanje WiFi konekcije
- Povezivanje na WiFi posle gubitka konekcije
- ESP32 WiFi događaji
- Povezivanje na WiFi posle gubitka konekcije (WiFi događaji)

ESP 32 – WIFI PODRŠKA

- Kako bi mogli da koristimo ESP32 Wi-Fi funkcionalnost moramo najpre da ukljucimo odgovarajuću biblioteku:

```
#include <WiFi.h>
```

ESP 32 – WIFI MODOVI

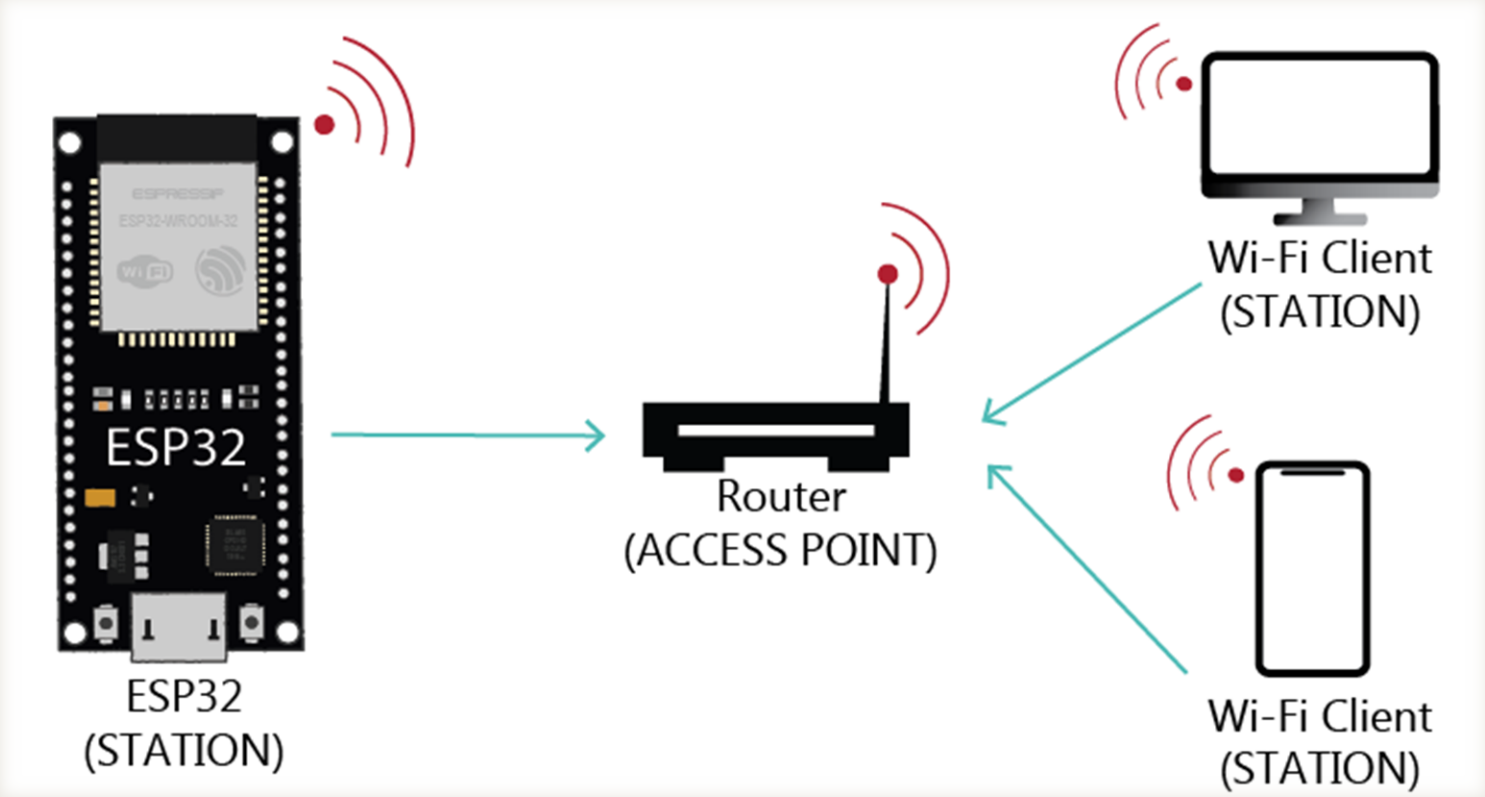
- ESP32 može da radi u sledeća tri režima koji se „biraju“ pozivom funcije `WiFi.mode()` sa odgovarajućim argumentom.

WiFi.mode(WIFI_STA)	Station mod: ESP32 se povezuje na neku pristupnu tačku
WiFi.mode(WIFI_AP)	Access point mode: uredjaji mogu da se povezu na ESP32
WiFi.mode(WIFI_AP_STA)	Kombinacija predhodna dva moda

ESP 32 – WI-FI STATION

```
WiFi.mode(WIFI_STA)
```

- Kada je ESP32 podešen kao Wi-Fi stanica, može se povezati na druge mreže (kao što je vaš ruter).
- U ovom slučaju ruter dodeljuje jedinstvenu IP adresu ESP-u.

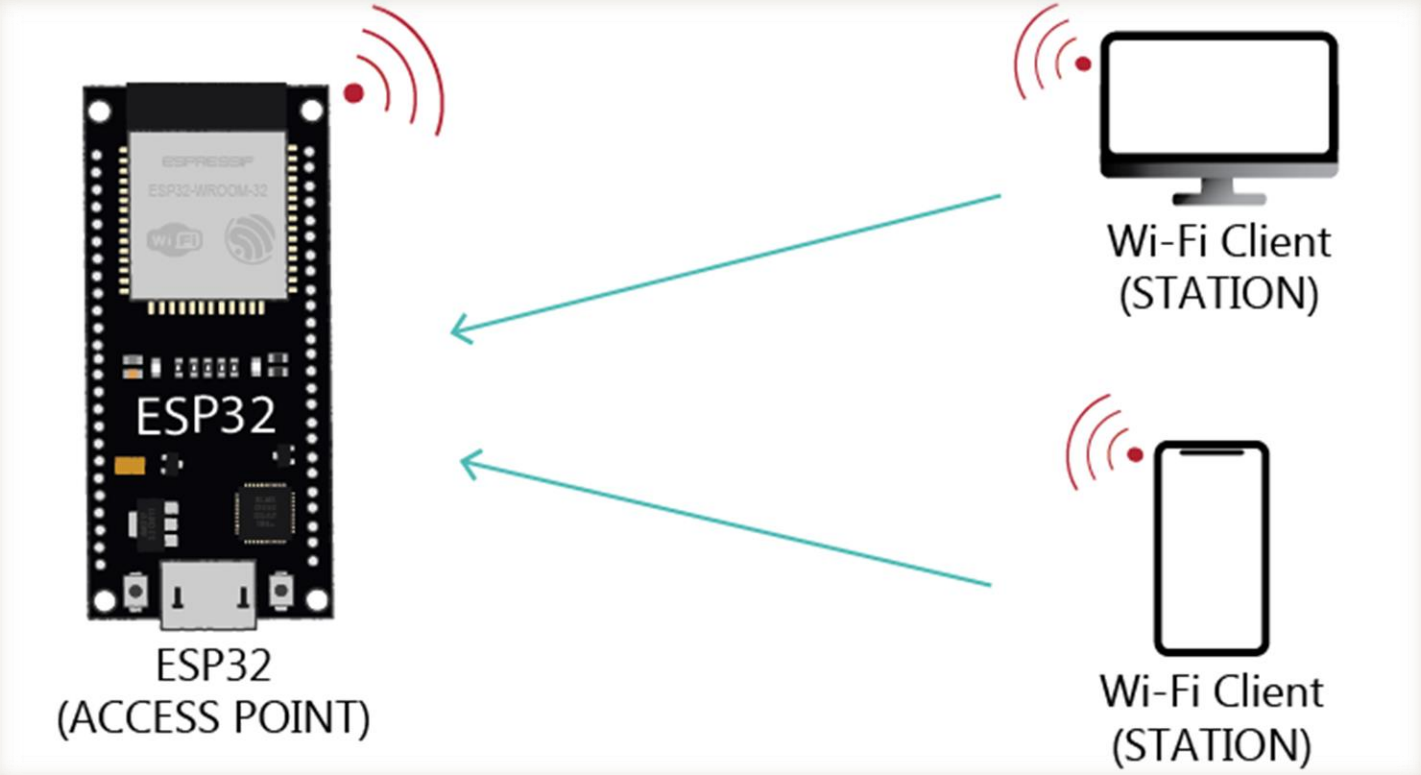


Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

ESP 32 – WI-FI ACCESS POINT

```
WiFi.mode(WIFI_AP)
```

- Kada je ESP32 podešen kao Wi-Fi pristupna stanica možete ostale uređaje povezati na ESP32 bez dodatnog rutera.
- Ovo može biti korisno ako želite da međusobno povežete više ESP32 uređaja



Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

ESP 32 – WI-FI STATION + ACCESS POINT

```
WiFi.mode(WIFI_AP_STA)
```

- Ovako podešen ESP 32 može istovremeno raditi i kao stanica i kao pristupna tačka

Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

ESP 32 – SKENIRANJE WI-FI MREŽA

```
#include "WiFi.h"

void setup() {
  Serial.begin(115200); // Set WiFi to station mode and disconnect from an AP if it was previously connected
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  delay(100);
}

void loop() {
  Serial.println("scan start"); // WiFi.scanNetworks will return the number of networks found
  int n = WiFi.scanNetworks();
  Serial.println("scan done");
  if (n == 0) {
    Serial.println("no networks found");
  } else {
    Serial.print(n);
    Serial.println(" networks found");
    for (int i = 0; i < n; ++i) { // Print SSID and RSSI for each network found
      Serial.print(i + 1); Serial.print(WiFi.SSID(i)); Serial.print(WiFi.RSSI(i));
      Serial.println((WiFi.encryptionType(i) == WIFI_AUTH_OPEN)?" ":"*");

      delay(10);
     }
  }

  Serial.println(""); // Wait a bit before scanning again
  delay(5000);
}
```

Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

ESP 32 – POVEZIVANJE NA WI-FI MREŽU

```
const char* ssid = "REPLACE_WITH_YOUR_SSID";
const char* password = "REPLACE_WITH_YOUR_PASSWORD";


.
.
.


void initWiFi() {
   WiFi.mode(WIFI_STA);
   WiFi.begin(ssid, password);
   Serial.print("Connecting to WiFi ..");

   while (WiFi.status() != WL_CONNECTED) {
     Serial.print('.');
     delay(1000);
   }

   Serial.println(WiFi.localIP());
}
```

Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

ESP 32 – POVEZIVANJE NA WI-FI MREŽU

```
const char* ssid = "REPLACE_WITH_YOUR_SSID";
const char* password = "REPLACE_WITH_YOUR_PASSWORD";


.
.
.


void initWiFi() {
   WiFi.mode(WIFI_STA);
   WiFi.begin(ssid, password);
   Serial.print("Connecting to WiFi ..");

   while (WiFi.status() != WL_CONNECTED) {
     Serial.print('.');
     delay(1000);
   }

   Serial.println(WiFi.localIP());
}
```

Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

ESP 32 – POSTAVLJANJE STATIČKE IP ADRESE

```
// Set your Static IP address
IPAddress local_IP(192, 168, 1, 184);

// Set your Gateway IP address
IPAddress gateway(192, 168, 1, 1);
IPAddress subnet(255, 255, 0, 0);
IPAddress primaryDNS(8, 8, 8, 8); // optional
IPAddress secondaryDNS(8, 8, 4, 4); // optional

// Configures static IP address
if (!WiFi.config(local_IP, gateway, subnet, primaryDNS, secondaryDNS)) {
  Serial.println("STA Failed to configure");
}
```

Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

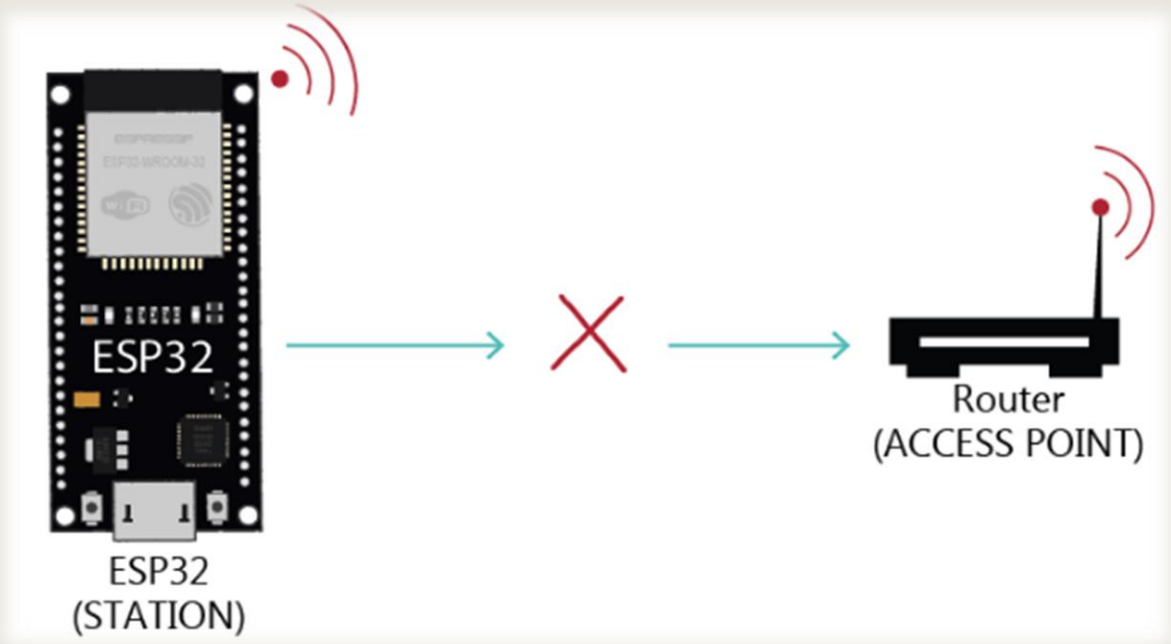
ESP 32 – PREKID/USPOSTAVLJANJE WI-FI VEZE

```
WiFi.disconnect();
```

```
WiFi.reconnect();

ili

WiFi.disconnect();
WiFi.begin(ssid, password);
```



Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

ESP 32 – PREKID/USPOSTAVLJANJE WI-FI VEZE

```
#include <WiFi.h>
const char* ssid = "REPLACE_WITH_YOUR_SSID";
const char* password = "REPLACE_WITH_YOUR_PASSWORD";
unsigned long previousMillis = 0;
unsigned long interval = 30000;
void initWiFi() {
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  Serial.print("Connecting to WiFi ..");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print('.’);
    delay(1000); }
    Serial.println(WiFi.localIP());
  }
void setup() {
  Serial.begin(115200);
  initWiFi();
}
void loop() {
  unsigned long currentMillis = millis();
  if ((WiFi.status() != WL_CONNECTED) && (currentMillis - previousMillis >=interval)) {
    Serial.print(millis());
    Serial.println("Reconnecting to WiFi...");
    WiFi.disconnect();
    WiFi.reconnect();
    previousMillis = currentMillis;
  }
}
```

Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

ESP 32 –WI-FI DOGAĐAJI

```
0  ARDUINO_EVENT_WIFI_READY                < ESP32 WiFi ready
1  ARDUINO_EVENT_WIFI_SCAN_DONE            < ESP32 finish scanning AP
2  ARDUINO_EVENT_WIFI_STA_START            < ESP32 station start
3  ARDUINO_EVENT_WIFI_STA_STOP             < ESP32 station stop
4  ARDUINO_EVENT_WIFI_STA_CONNECTED        < ESP32 station connected to AP
5  ARDUINO_EVENT_WIFI_STA_DISCONNECTED     < ESP32 station disconnected from AP
6  ARDUINO_EVENT_WIFI_STA_AUTHMODE_CHANGE  < the auth mode of AP connected by ESP32 station changed
7  ARDUINO_EVENT_WIFI_STA_GOT_IP           < ESP32 station got IP from connected AP
8  ARDUINO_EVENT_WIFI_STA_LOST_IP          < ESP32 station lost IP and the IP is reset to 0
9  ARDUINO_EVENT_WPS_ER_SUCCESS            < ESP32 station wps succeeds in enrollee mode
10 ARDUINO_EVENT_WPS_ER_FAILED             < ESP32 station wps fails in enrollee mode
11 ARDUINO_EVENT_WPS_ER_TIMEOUT            < ESP32 station wps timeout in enrollee mode
12 ARDUINO_EVENT_WPS_ER_PIN                < ESP32 station wps pin code in enrollee mode
13 ARDUINO_EVENT_WIFI_AP_START             < ESP32 soft-AP start
14 ARDUINO_EVENT_WIFI_AP_STOP              < ESP32 soft-AP stop
15 ARDUINO_EVENT_WIFI_AP_STACONNECTED      < a station connected to ESP32 soft-AP
16 ARDUINO_EVENT_WIFI_AP_STADISCONNECTED   < a station disconnected from ESP32 soft-AP
17 ARDUINO_EVENT_WIFI_AP_STAIPASSIGNED     < ESP32 soft-AP assign an IP to a connected station
18 ARDUINO_EVENT_WIFI_AP_PROBEREQRECVED    < Receive probe request packet in soft-AP interface
19 ARDUINO_EVENT_WIFI_AP_GOT_IP6           < ESP32 ap interface v6IP addr is preferred
19 ARDUINO_EVENT_WIFI_STA_GOT_IP6          < ESP32 station interface v6IP addr is preferred
20 ARDUINO_EVENT_ETH_START                 < ESP32 ethernet start
21 ARDUINO_EVENT_ETH_STOP                  < ESP32 ethernet stop
22 ARDUINO_EVENT_ETH_CONNECTED             < ESP32 ethernet phy link up
23 ARDUINO_EVENT_ETH_DISCONNECTED          < ESP32 ethernet phy link down
24 ARDUINO_EVENT_ETH_GOT_IP                < ESP32 ethernet got IP from connected AP
19 ARDUINO_EVENT_ETH_GOT_IP6               < ESP32 ethernet interface v6IP addr is preferred
25 ARDUINO_EVENT_MAX
```

Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

ESP 32 – WI-FI DOGAĐAJI

```
#include <WiFi.h>
const char* ssid = "REPLACE_WITH_YOUR_SSID";
const char* password = "REPLACE_WITH_YOUR_PASSWORD";
void WiFiStationConnected(WiFiEvent_t event, WiFiEventInfo_t info){
  Serial.println("Connected to AP successfully!");
}
void WiFiGotIP(WiFiEvent_t event, WiFiEventInfo_t info){
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
void WiFiStationDisconnected(WiFiEvent_t event, WiFiEventInfo_t info){
  Serial.println("Disconnected from WiFi access point");
  Serial.print("WiFi lost connection. Reason: ");
  Serial.println(info.wifi_sta_disconnected.reason);
  Serial.println("Trying to Reconnect");
  WiFi.begin(ssid, password);
}
void setup(){
  Serial.begin(115200);
  WiFi.disconnect(true);
  delay(1000);
  WiFi.onEvent(WiFiStationConnected, WiFiEvent_t::ARDUINO_EVENT_WIFI_STA_CONNECTED);
  WiFi.onEvent(WiFiGotIP, WiFiEvent_t::ARDUINO_EVENT_WIFI_STA_GOT_IP);
  WiFi.onEvent(WiFiStationDisconnected, WiFiEvent_t::ARDUINO_EVENT_WIFI_STA_DISCONNECTED);
  WiFi.begin(ssid, password);
}
void loop(){
  delay(1000);
}
```

Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/

ESP 32 – WATCHDOG

```
#include <esp_task_wdt.h>

#define WDT_TIMEOUT 10 //10 seconds WDT

extern "C" {
  #include "freertos/FreeRTOS.h"
  #include "freertos/timers.h"
}


esp_task_wdt_init (WDT_TIMEOUT, true);    //enable panic so ESP32 restarts

esp_task_wdt_add (NULL);                  //add current thread to WDT watch
esp_task_wdt_add (read_sonda_task);       //add read_sonda_task to WDT watch



esp_task_wdt_reset();
```

Preuzeto sa: https://randomnerdtutorials.com/esp32-useful-wi-fi-functions-arduino/