HANDS ON SESSION 09: WI-FI ESP32

ESP 32 Wifi

The ESP32, supports WiFi on the 2.4 GHz band.
It supports WiFi protocols 802.11 b/g/n with a maximum data transfer rate of 150 Mbps.
The device has an adjustable transmit power of up to 20.5 dBm, using lower power will decrease the current requirements as the WiFi radio can consume a fair amount of current.

WiFi Modes

The ESP32 can be used in two different WiFi modes.

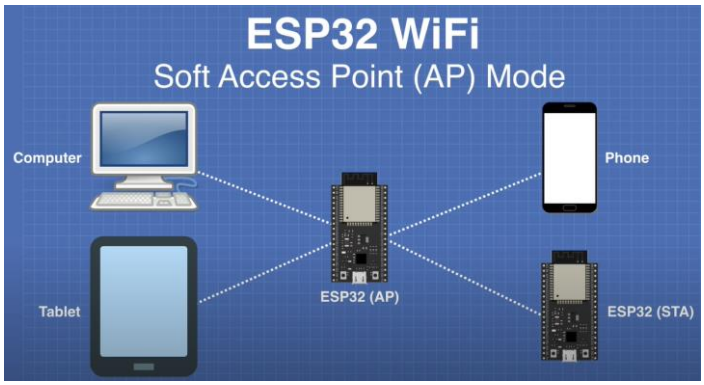
Station (STA) Mode

In STA, or Station, mode the ESP32 acts as a WiFi station or client.

In this mode the ESP32 needs to know the SSID and Password to access the WiFi network (on an unsecured network there is no password requirement).
The ESP32 is provided with a network IP address using the routers internal DHCP server. It can then be accessed using that address.
It is also possible to assign a fixed IP address to the ESP32, which is useful if you are using it as a web server where a changing IP address would cause difficulties for other clients.

Access Point (AP) Mode

In Soft Access Point, or AP, mode, the ESP32 provides a WiFi connection for external devices. These devices can be computers, phones, tablets, IoT devices or even other ESP32's configured in STA mode.



Wifi.h

To expose the Wifi functionality of ESP32 we will use Wifi.h library

To set the Wi-Fi mode, use `WiFi.mode()`

`WiFi.mode(WIFI_STA)`	station mode: the ESP32 connects to an access point
`WiFi.mode(WIFI_AP)`	access point mode: stations can connect to the ESP32

Scan WiFi

`WiFi.scanNetworks()` returns the number of networks found.

`WiFi.SSID()` prints the SSID for a specific network:

`WiFi.RSSI()` returns the RSSI of that network. RSSI stands for **R**eceived **S**ignal **S**trength **I**ndicator. It is an estimated measure of power level that an RF client device is receiving from an access point or router.

Finally, `WiFi.encryptionType()` returns the network encryption type. That specific example puts a * in the case of open networks. However, that function can return one of the following options (not just open networks):

- WIFI_AUTH_OPEN
- WIFI_AUTH_WEP
- WIFI_AUTH_WPA_PSK
- WIFI_AUTH_WPA2_PSK
- WIFI_AUTH_WPA_WPA2_PSK
- WIFI_AUTH_WPA2_ENTERPRISE

Connecting Wifi STA

WiFi.begin() to connect to a network. You must pass as arguments the network SSID and its password

WiFi.status(). When the connection is successfully established, it returns WL_CONNECTED

- WL_CONNECTED: assigned when connected to a WiFi network;
- WL_NO_SHIELD: assigned when no WiFi shield is present;
- WL_IDLE_STATUS: it is a temporary status assigned when WiFi.begin() is called and remains active until the number of attempts expires (resulting in WL_CONNECT_FAILED) or a connection is established (resulting in WL_CONNECTED);
- WL_NO_SSID_AVAIL: assigned when no SSID are available;
- WL_SCAN_COMPLETED: assigned when the scan networks is completed;
- WL_CONNECT_FAILED: assigned when the connection fails for all the attempts;
- WL_CONNECTION_LOST: assigned when the connection is lost;
- WL_DISCONNECTED: assigned when disconnected from a network;

Static IP

WiFi.config() method to assign the configurations

```
WiFi.config(local_IP, gateway, subnet, primaryDNS, secondaryDNS
```

AP

```
WiFi.softAP(const char* ssid, const char* password, int channel, int ssid_hidden, int max_connection)
```

- `ssid`: name for the access point – maximum of 63 characters;
- `password`: **minimum of 8 characters**; set to `NULL` if you want the access point to be open;
- `channel`: Wi-Fi channel number (1-13)
- `ssid_hidden`: (0 = broadcast SSID, 1 = hide SSID)

- `max_connection`: maximum simultaneous connected clients (1-4)