Wi-Fi communication

Fabrice MULLER

Fabrice.Muller@univ-cotedazur.fr

2023 - 2024

Wi-Fi communication

Lab 1 : WiFi connection & HTTP procotol

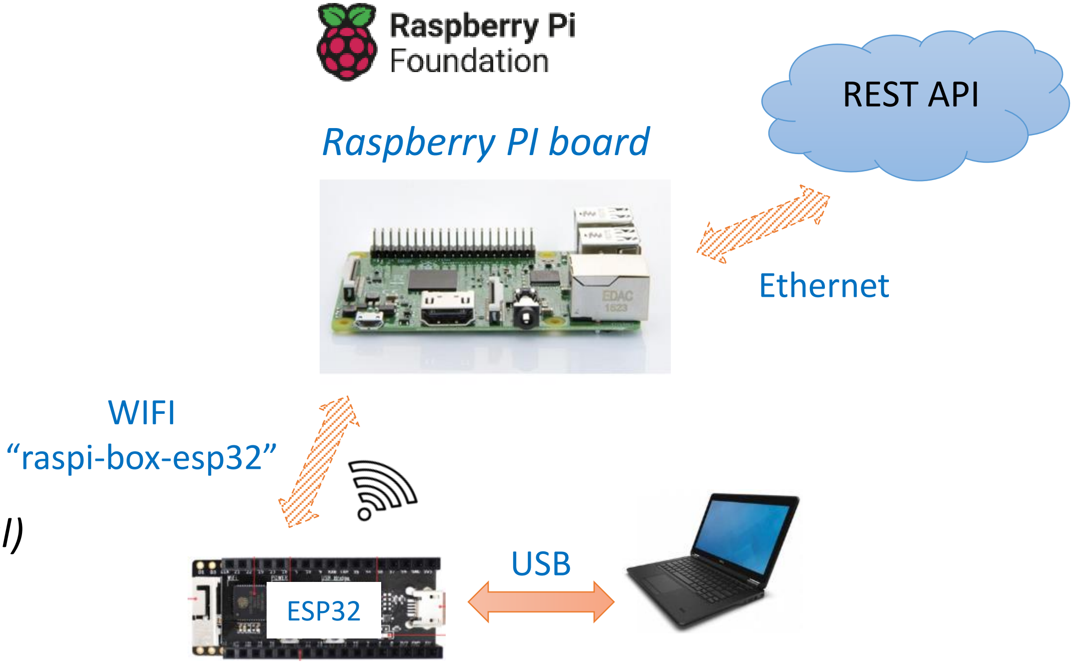
- Simple/Advanced connection
- HTTP data & memory allocation
- Update date/time with NTP server

Lab 2 : REST Client – GET method

- Get data from REST API
- JSON format & parsing
- Weather report API REST

Lab 3 : REST Client – POST method *(optional)*

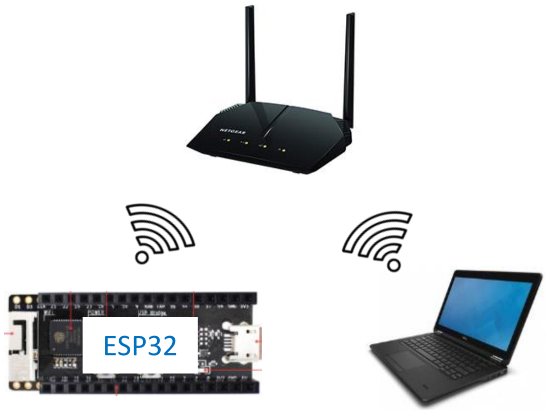
- Post data to RESP API
- JSON format
- Email API REST



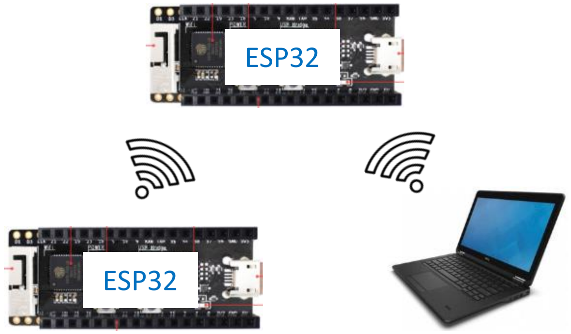
Wi-Fi Modes

- Station (STA)
  - A device uses Station Mode to join a network that already exists
  - Home box
  - Raspberry Pi (Lab)
- Access Point (AP)
  - The device is the Access Point and becomes an entity that every client device can connect to it
  - ESP32 board (Lab)
- IoT Context scenario
  - Step 1. Access Point mode is generally for set up the IoT device
  - Step 2. Once configured the IoT device will exit AP mode
  - Step 3. Run in Station mode for the rest of the IoT application

Station STA



Access Point (AP)



Network
IP address & DHCP

- IP address
  - IP address is a string of digits
  - Define the location of a device on a network
  - The address comprises of 4 groups of numbers
  - IP addresses can be static or dynamic
  - IP addresses have banded designations: a numerical range of addresses have been reserved for specific uses

192.168.1.25

↑

dot

- MAC address (Media Access Control)
  - Assigned by device manufacturers
  - Unique identifier assigned to a Network Interface Controller (NIC)
  - 6 groups of two hexadecimal digits

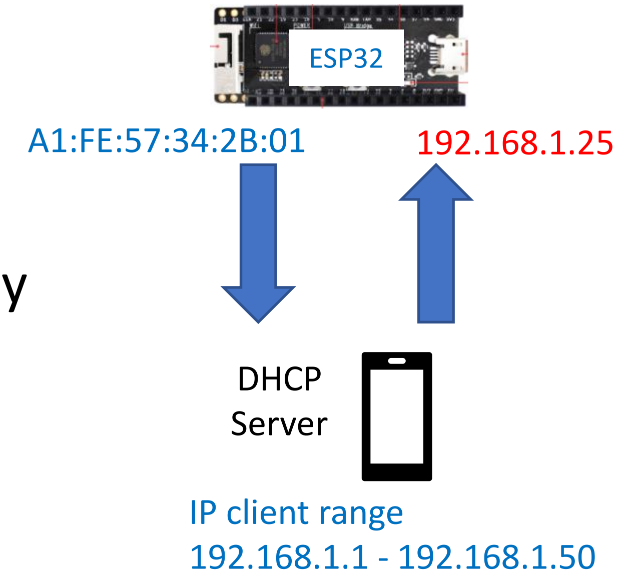
A1:FE:57:34:2B:01

↑

colon

Network
DHCP & Port number

- DHCP (Dynamic Host Configuration Protocol)
  - Network protocol allows a server to automatically assign an IP address to a device
  - The device can keep the same IP address every time from the MAC address identified by the DHCP server
- Port number
  - Different ports = different types of communications
  - Port numbers range from 1 to 65535
  - Examples: HTTP (port 80), MQTT (port 1883)
  - List of TCP and UDP port numbers
    - https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers



HTTP google site:

http://www.google.com:80/

Port number

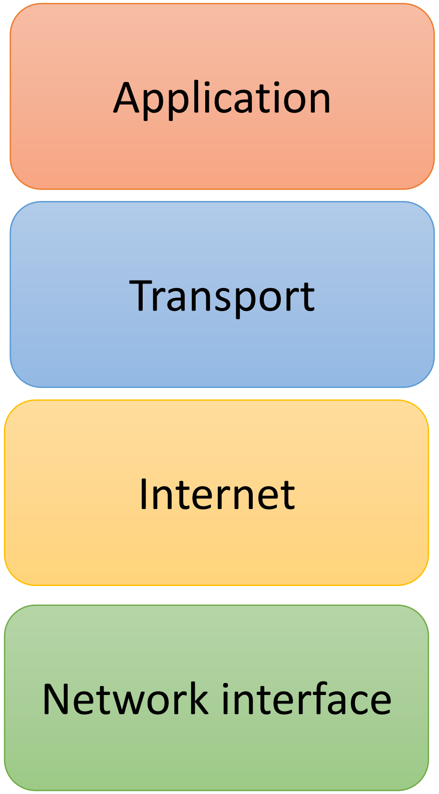
TCP/IP Model
Goals

- TCP/IP = Transmission Control Protocol/ Internet Protocol
- To help you to determine how a specific computer should be connected to the internet
- How data should be transmitted between them
- To create a virtual network when multiple computer networks are connected together
- To allow communication over large distances
- TCP/IP Stack is designed as a model to offer highly reliable and end-to-end byte stream over an unreliable inter-network

TCP/IP Layers

- Four Layers of TCP/IP model
- Each layer
  - Includes specific protocols
  - Defines a specific function to perform
- 4 layers
  - Application Layer
  - Transport Layer
  - Internet Layer
  - Network Interface

TCP/IP Layers



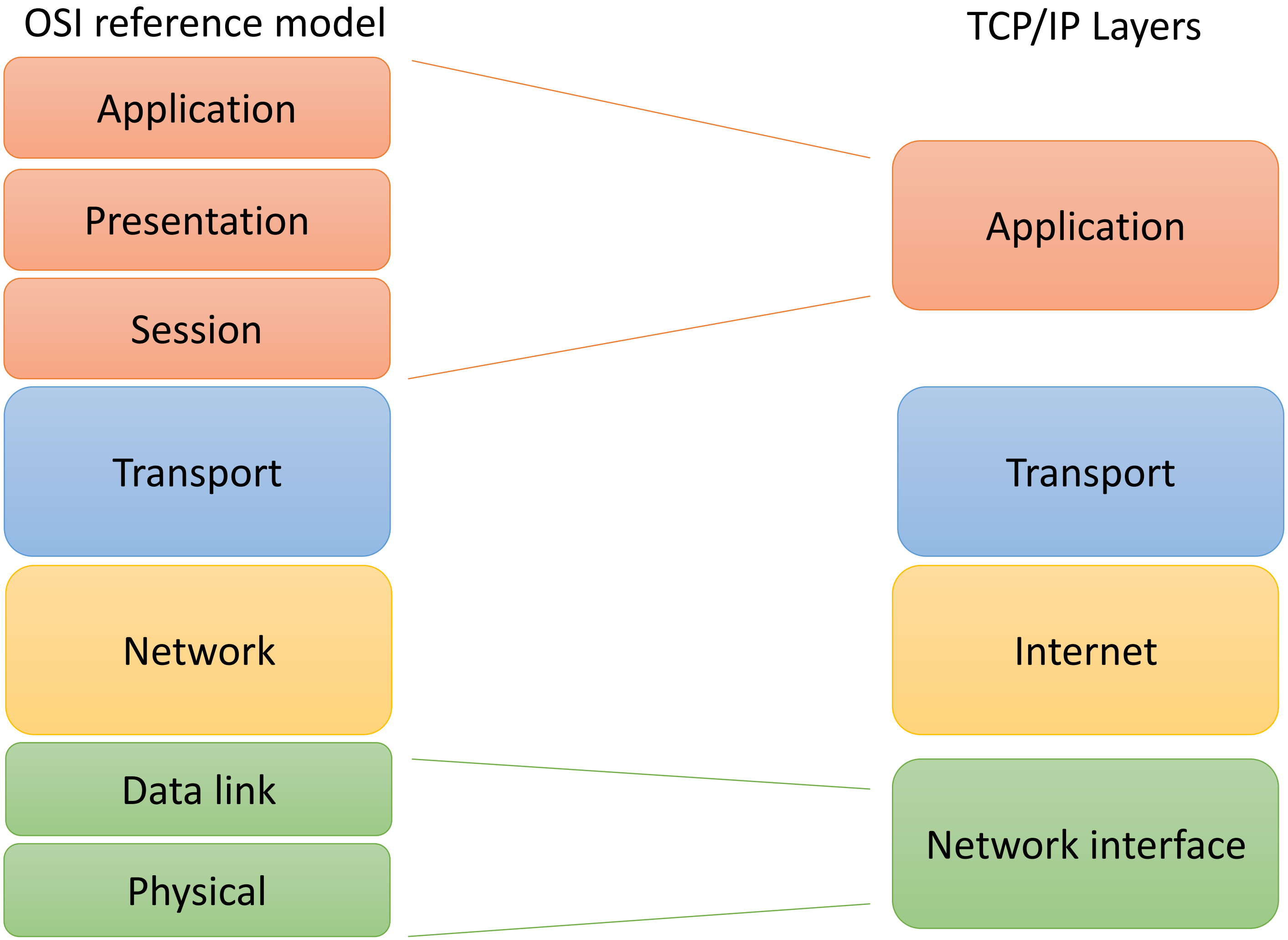
TCP/IP Layers
Application & Transport

- Application Layer
  - Highest level of OSI model
  - Interacts with an application program
  - Allows users to interact with other software application
  - Examples
    - Allows users to log on to a remote host
    - Provides various e-mail services
    - Offers distributed database sources and access for global information about various objects and IoT services
- Transport Layer
  - Divides the message received from the upper layer into segments and numbers them to make a sequence
  - Maintains the quality of service (QoS) functions
  - Determines how much data should be sent where and at what rate
  - Helps to control the reliability of a link through flow control and error control
  - offers an acknowledgment of the successful data transmission
  - Example: TCP protocol

TCP/IP Layers
Internet & Network

- Internet Layer
  - Also known as a network layer
  - To send the packets from any network and any computer so that they reach the destination whatever the route they take
- Network Interface Layer
  - Also called a network access layer
  - To define how the data should be sent physically through the network
  - Responsible for the transmission of the data between two devices on the same network
  - How bits should be signaled by hardware devices which directly interfaces with a network medium (coaxial fiber, twisted-pair cables …)

OSI model versus TCP/IP model



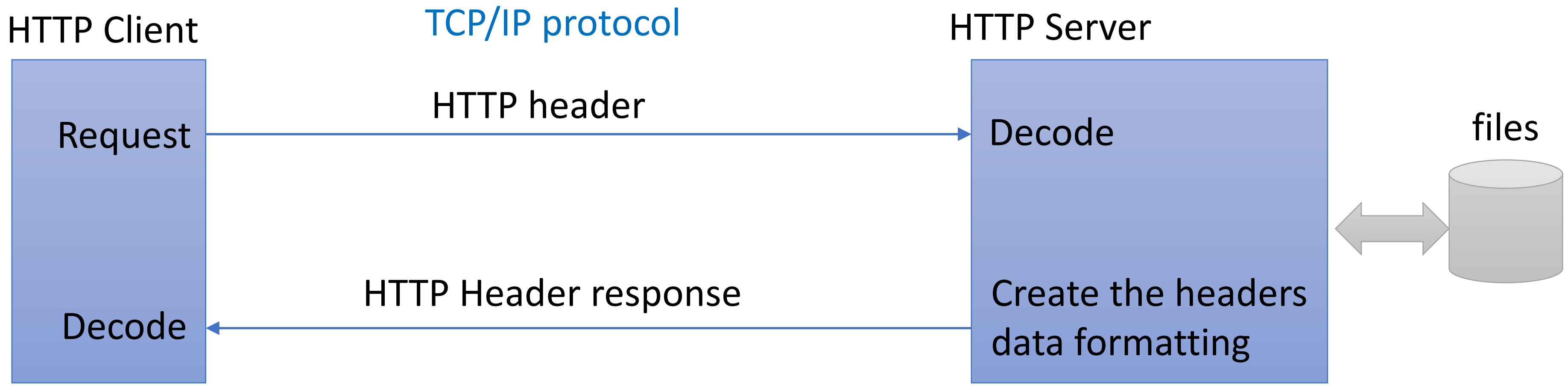
Application layer example
HTTP protocol

- HTTP = HyperText Transfer Protocol
- Using the TCP/IP protocol

https://www.espressif.com/en/products/socs/esp32

Host name

Path name

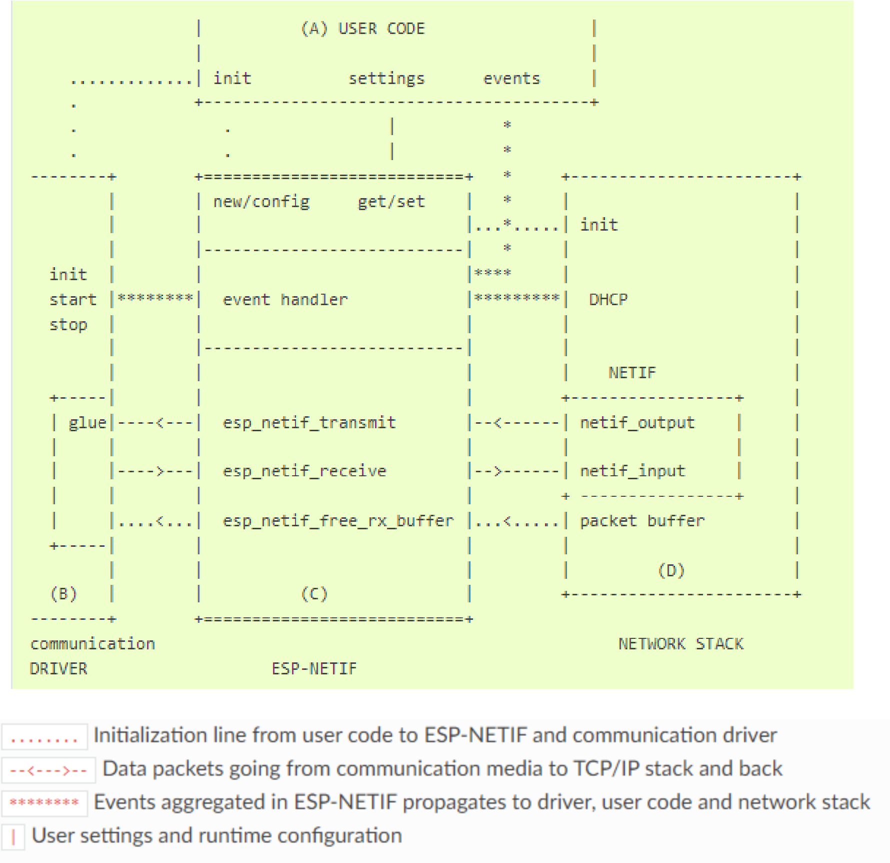


ESP-NETIF Introduction

- Abstraction layer for the application on top of the TCP/IP stack
- ESP-NETIF APIs are thread safe even if the underlying TCP/IP stack APIs are not.
- To manage DHCP, IP addresses, and other attributes of a physical network interface
- To make easy for applications that have multiple network interfaces to switch network interfaces or select a default network interface for internet access
- Do not need to call ESP-NETIF APIs directly
  - they are called from the default network event handlers

ESP-NETIF Architecture

- User code
  - Interaction with a specific IO driver for communication media and configured TCP/IP network stack
- Communication driver, IO driver, media driver
  - Event handlers
    - Define behaviour patterns of interaction with ESP-NETIF
  - Glue IO layer
    - Adapts the input/output functions to use ESP-NETIF
- ESP-NETIF
  - Intermediary between an IO driver and a network stack
- Network stack
  - no public interaction with application code
  - shall be fully abstracted by ESP-NETIF API



```
                 |          (A) USER CODE                 |
                 |                                        |
    .............| init          settings      events     |
    .            +----------------------------------------+
    .               .                |           *
    .               .                |           *
--------+        +===========================+   *     +-----------------------+
        |        | new/config     get/set    |   *     |                       |
        |        |                           |...*.....| init                  |
        |        |---------------------------|   *     |                       |
  init  |        |                           |****     |                       |
  start |********|  event handler            |*********|  DHCP                 |
  stop  |        |                           |         |                       |
        |        |---------------------------|         |                       |
        |        |                           |         |    NETIF              |
  +-----|        |                           |         +-----------------+     |
  | glue|----<---|  esp_netif_transmit       |--<------| netif_output    |     |
  |     |        |                           |         |                 |     |
  |     |---->---|  esp_netif_receive        |-->------| netif_input     |     |
  |     |        |                           |         + ----------------+     |
  |     |....<...|  esp_netif_free_rx_buffer |...<.....| packet buffer         |
  +-----|        |                           |         |                       |
        |        |                           |         |         (D)           |
  (B)   |        |          (C)              |         +-----------------------+
--------+        +===========================+
communication                                                NETWORK STACK
DRIVER                   ESP-NETIF
```

....	Initialization line from user code to ESP-NETIF and communication driver
--<--->--	Data packets going from communication media to TCP/IP stack and back
********	Events aggregated in ESP-NETIF propagates to driver, user code and network stack
|	User settings and runtime configuration

https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/network/esp_netif.html

References

- TCP/IP Model: What are Layers & Protocol? TCP/IP Stack, By Lawrence Williams, https://www.guru99.com/tcp-ip-model.html
- ESP-NETIF, https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/network/esp_netif.html
- lwIP, https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-guides/lwip.html
- lwIP GitHub, https://github.com/espressif/esp-lwip