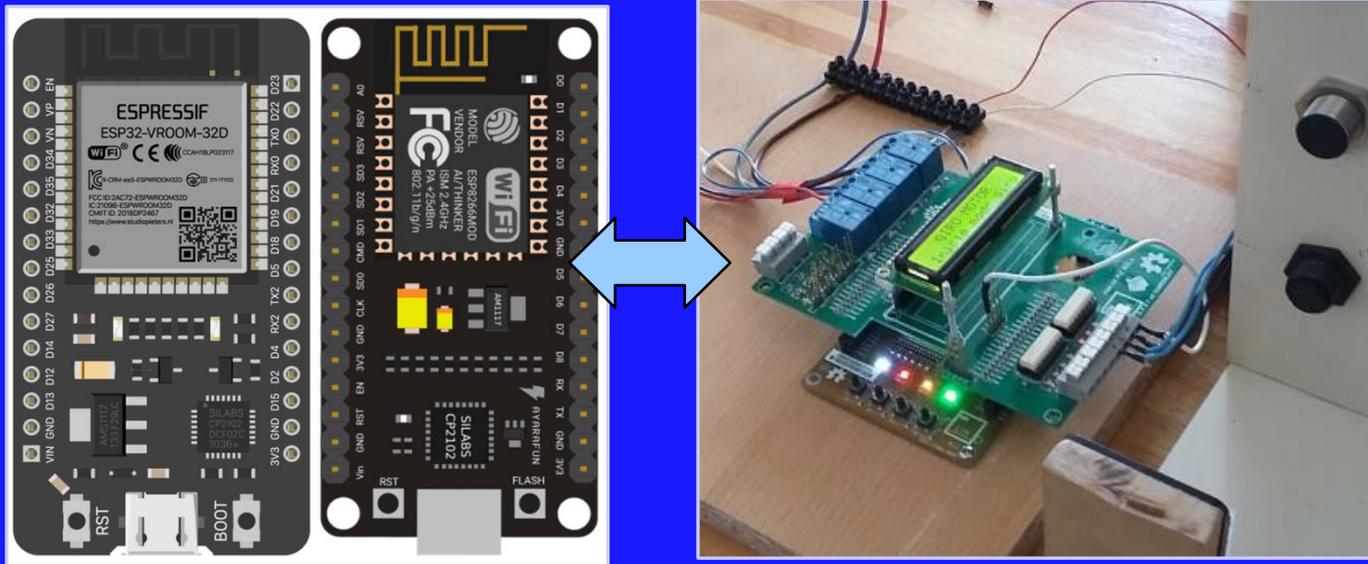


Avances en interfaces web para sistemas embebidos utilizando tecnología ESP



Parte I (Introducción) - Mg. Ing. Rafael Oliva, Ing. Esp. Nestor Cortez
Instituto de Tecnología Aplicada (ITA-UARG) Area Energías Alternativas

XIX JORNADAS DE INFORMÁTICA 2022 – UNPA /UARG
04 de noviembre 2022

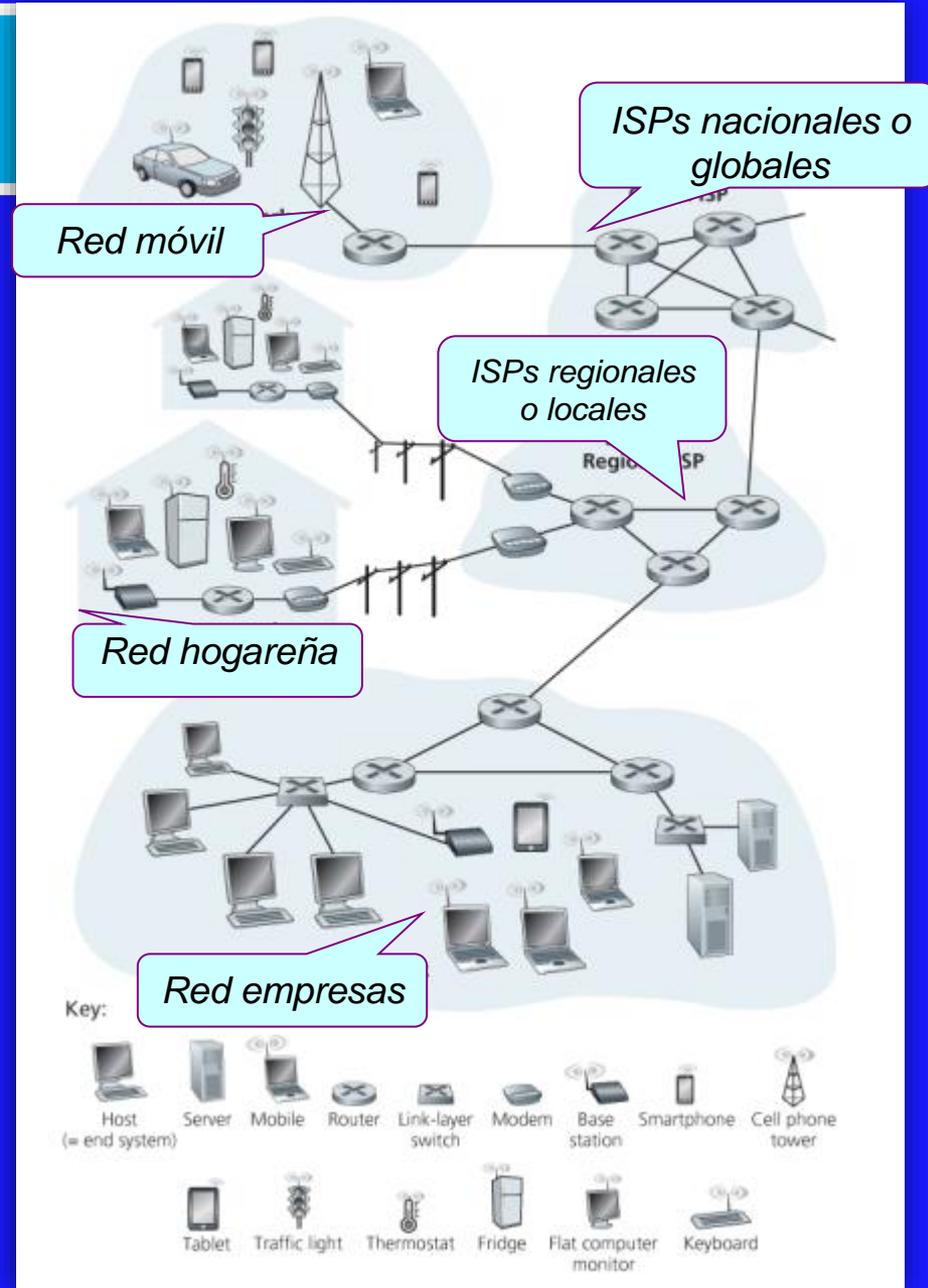
CONTENIDOS (1ra Parte):

- Interfaz Web en Internet – conceptos básicos y modelo de capas
- Uso de módulos ESP
- Sistemas embebidos – antecedentes
- Configuraciones posibles de conexión ESP-Sistema Embebido
- Caso demostración – Conexión a SISMED/SJ24 con módulo ESP
- Otros modelos y casos prácticos

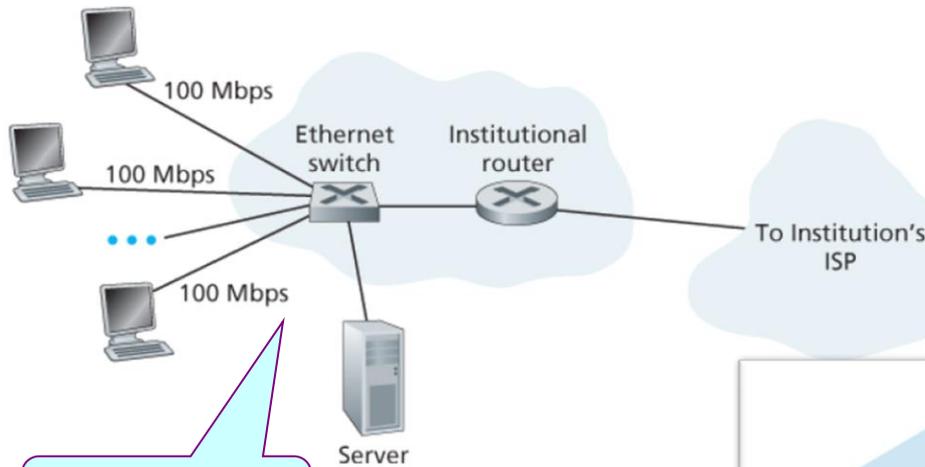
El uso de Internet para la conexión de dispositivos.. y casi todo lo demás

*ISP: Proveedor de Servicios
de Internet*

Gráficos de:
Kurose, Ross "Computer networking:
a top-down approach"
7ma Edición 2017 Ed. Pearson ISBN
0133594149

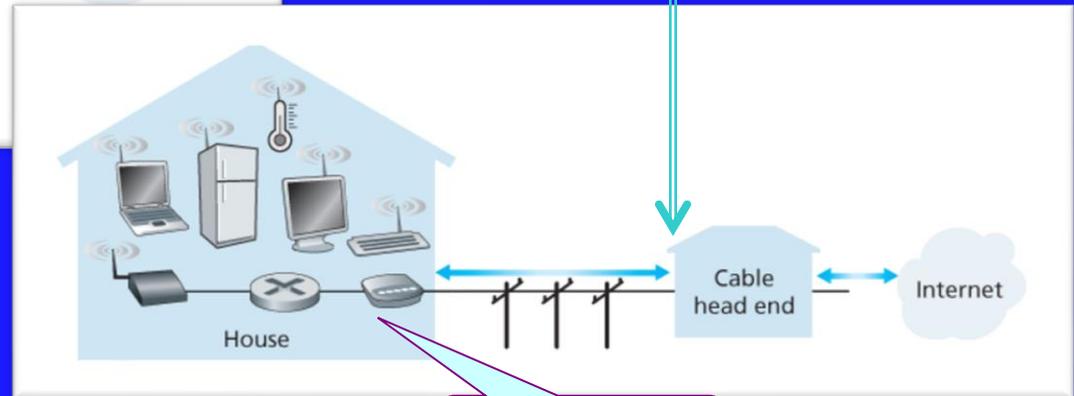


Accesos cableados (Ethernet) y WiFi



Red institucional

ISP: Proveedor de Servicios de Internet



Red hogareña típica

Gráficos de:

Kurose, Ross "Computer networking: a top-down approach"
7ma Edición 2017 Ed. Pearson ISBN
0133594149

“Stack” o pila de protocolos de Internet

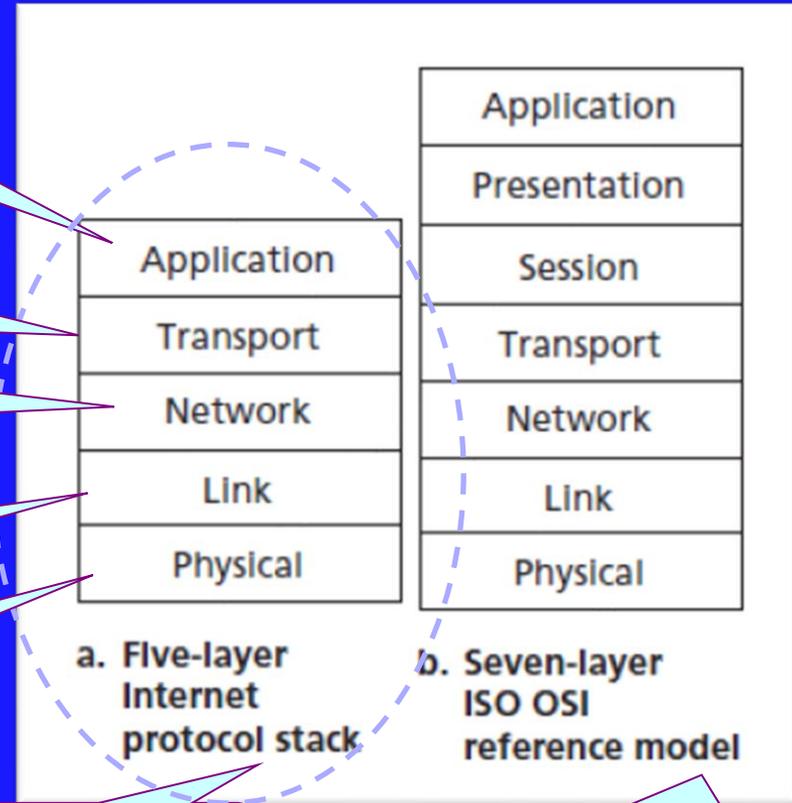
Capa de **aplicación** del usuario y los **protocolos asociados**: HTTP (Web), FTP, SMTP (mail)...

Capa de **transporte**: protocolos asociados: TCP (confiable, mas complejo), UDP (simple)

Capa de **red o network**: protocolo IP (v4 o v6) para enrutamiento de los paquetes de transporte

Capa de **enlace** para movimiento de los datagramas (Ethernet, WiFi, PPP..)

Capa **física** para movimiento de los bits individuales (cable UTP, fibra, RF)



Stack de 5 capas usado en Internet

OSI 7 capas completo, raramente usado

Gráficos de:

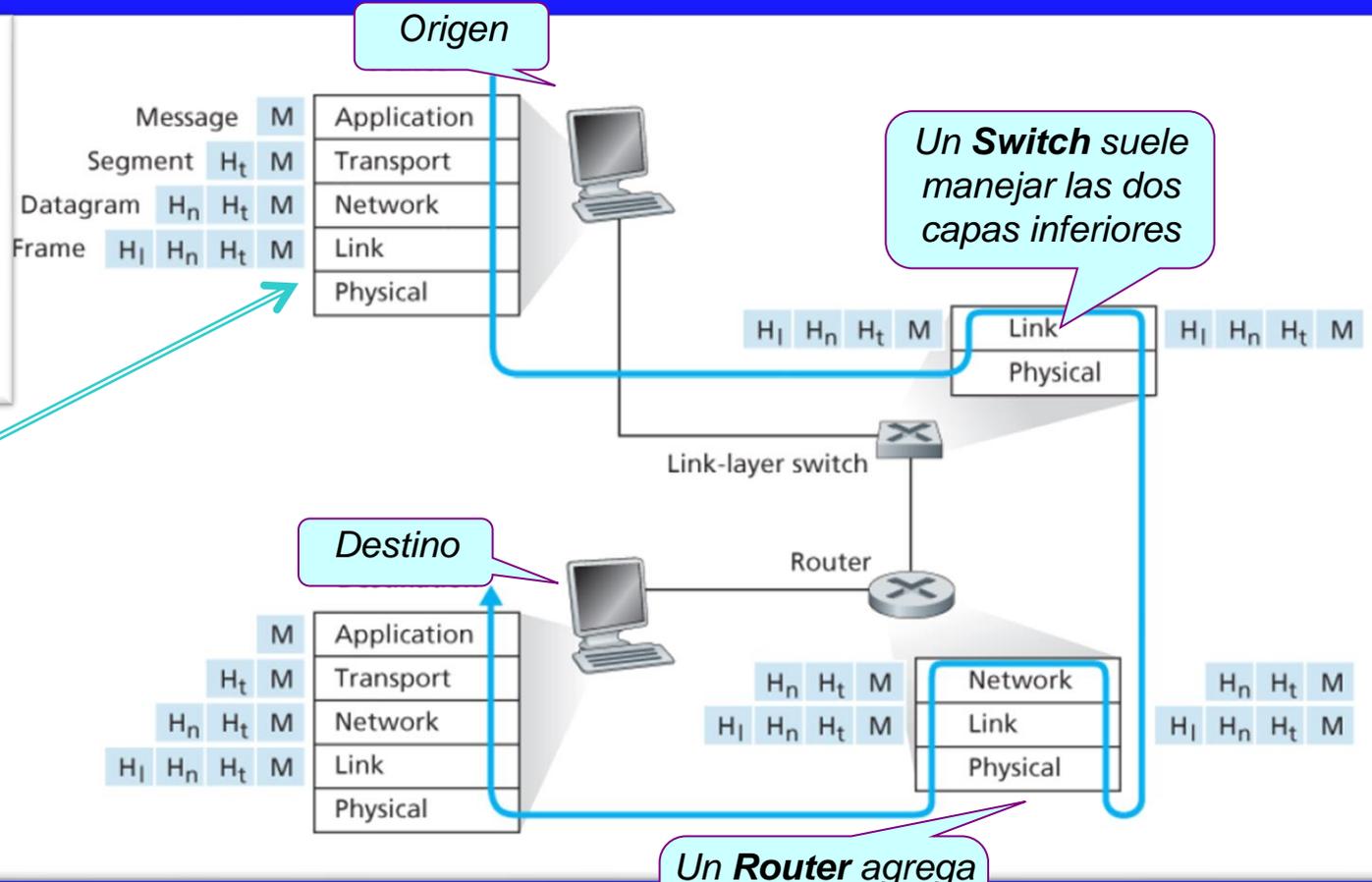
Kurose, Ross “Computer networking: a top-down approach”
7ma Edición 2017 Ed. Pearson ISBN 0133594149

“Stack” o pila de protocolos

Al mensaje **M** a enviar, se le agregan “Headers” o encabezados de cada capa:

H_t = header transporte
 H_n = header de red
 H_l = header de enlace

- Aplicación
- Transporte
- Red
- Enlace
- Físico



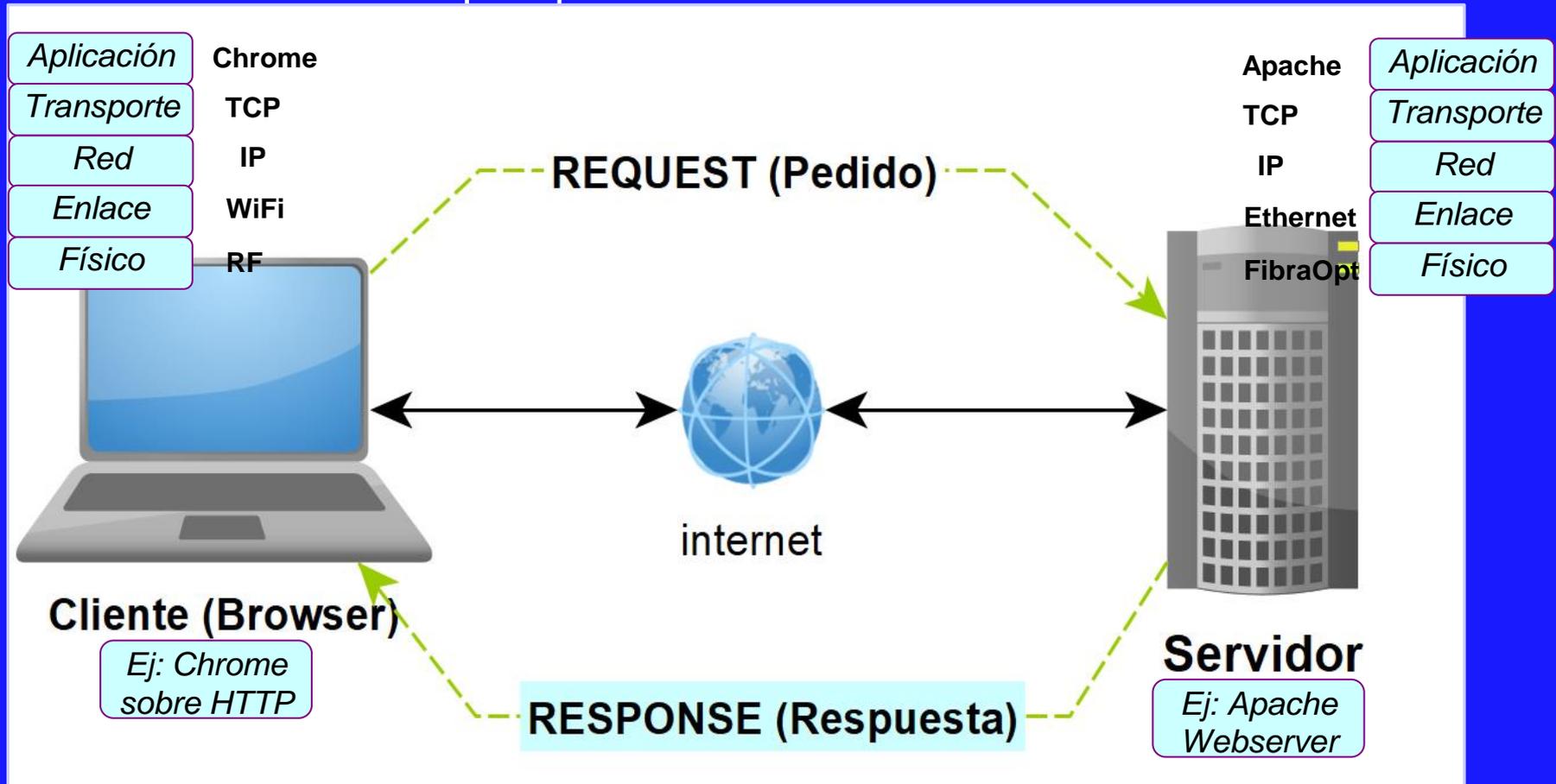
Gráficos de:

Kurose, Ross “Computer networking: a top-down approach”
 7ma Edición 2017 Ed. Pearson ISBN 0133594149

Un Router agrega capacidad de enrutamiento

Interfaz Web elemental – a Servidor convencional

Protocolo HTTP en capa Aplicación + TCP/IP



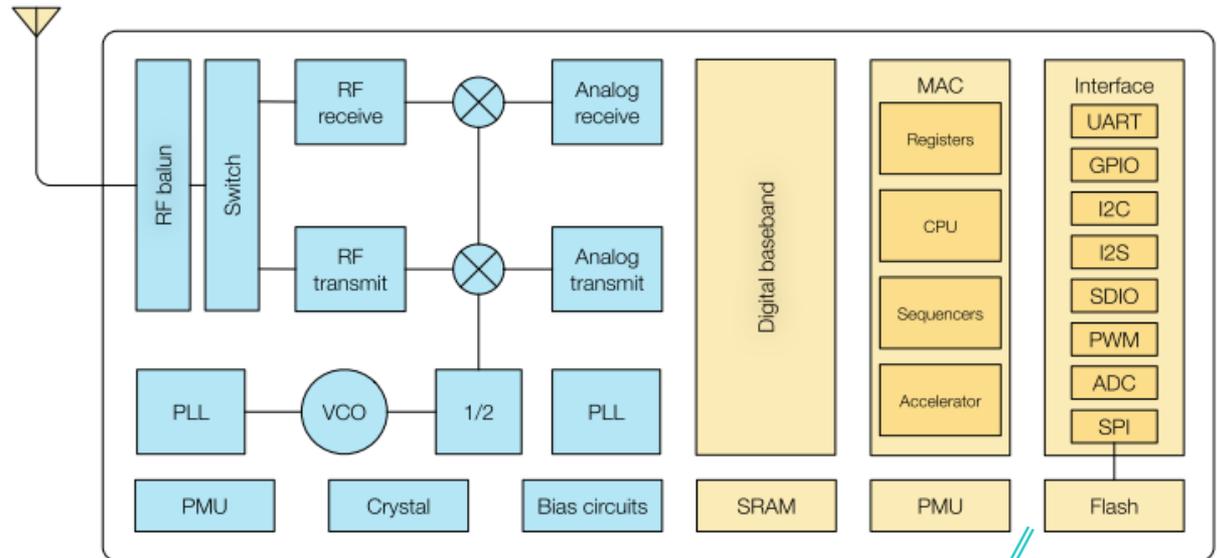
Espressif – Módulos ESP8266 y ESP32

Familia ESP

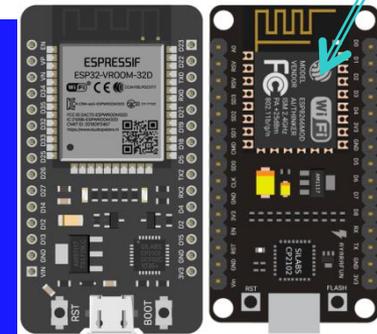
Wi-Fi SoC (System on Chip) para IoT

- Capacidades de WiFi Integradas, para funcionar sólo o como esclavo en comunicación con otros sistemas embebidos
- Contienen RAM y Memoria Flash interna, y se los puede programar con **Arduino IDE (similar a C++)** o con **Visual Studio Code / PlatformIO** y otras herramientas (C++)
- En el arranque bootea desde su flash interna. Puede funcionar como adaptador WiFi via UART o SPI
- ESPxxx integra antena, switches, RF balun, amplificador, receptor, filtros.

The functional diagram of ESP8266EX is shown as in Figure 3-1.



<https://www.espressif.com/>



Espressif – Módulos ESP32

<https://www.espressif.com/>

Familia ESP

Wi-Fi SoC (System on Chip)
para IoT



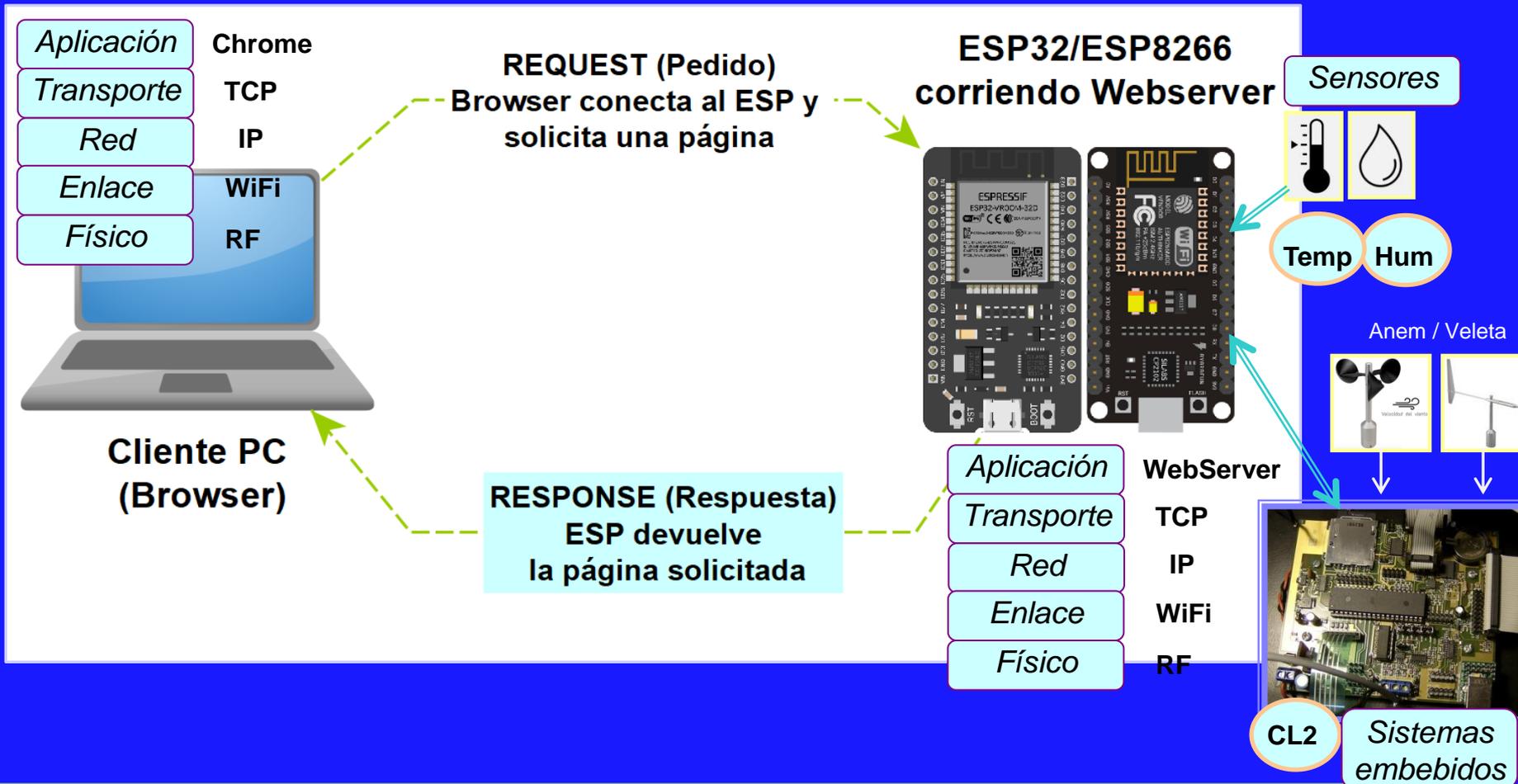
ESP32-DevKitC

ESP32-DevKitC is a low-footprint and entry-level development board that is part of the ESP32 series. This board has a rich peripheral set. The built-in ESP32 pinout is optimized for hassle-free prototyping!



Interfaz Web – reemplazo por Servidores ESP

Conexión directa a sensores u a otros sistemas embebidos

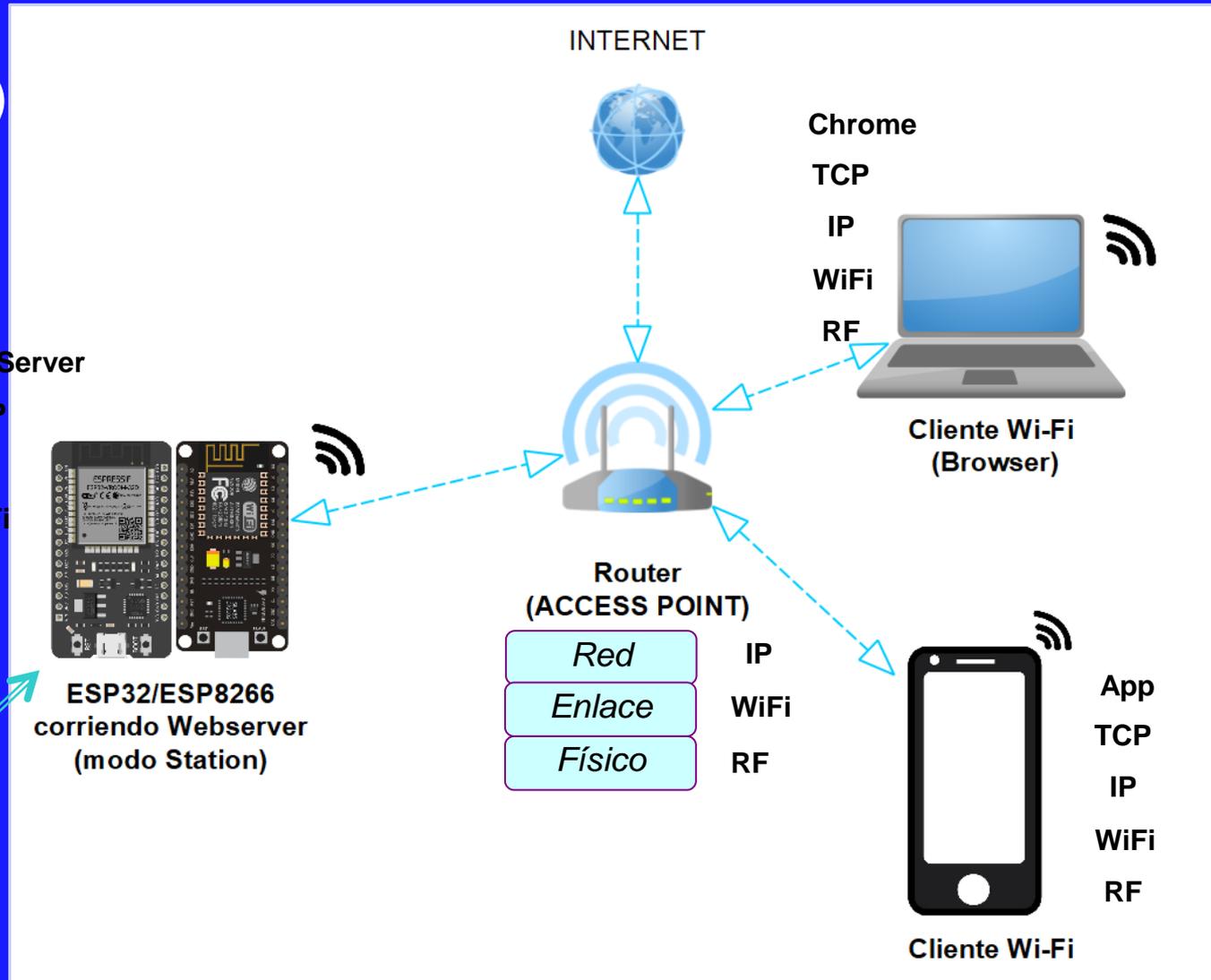


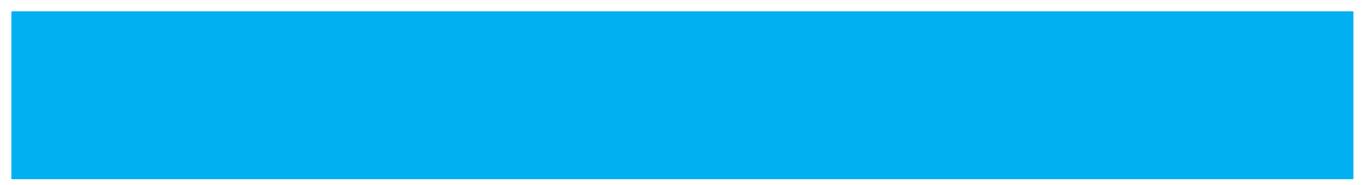
Configuración (i) – ESPs como Estación WiFi

Aplicación	WebServer
Transporte	TCP
Red	IP
Enlace	WiFi
Físico	RF

Sensores

Temp Hum

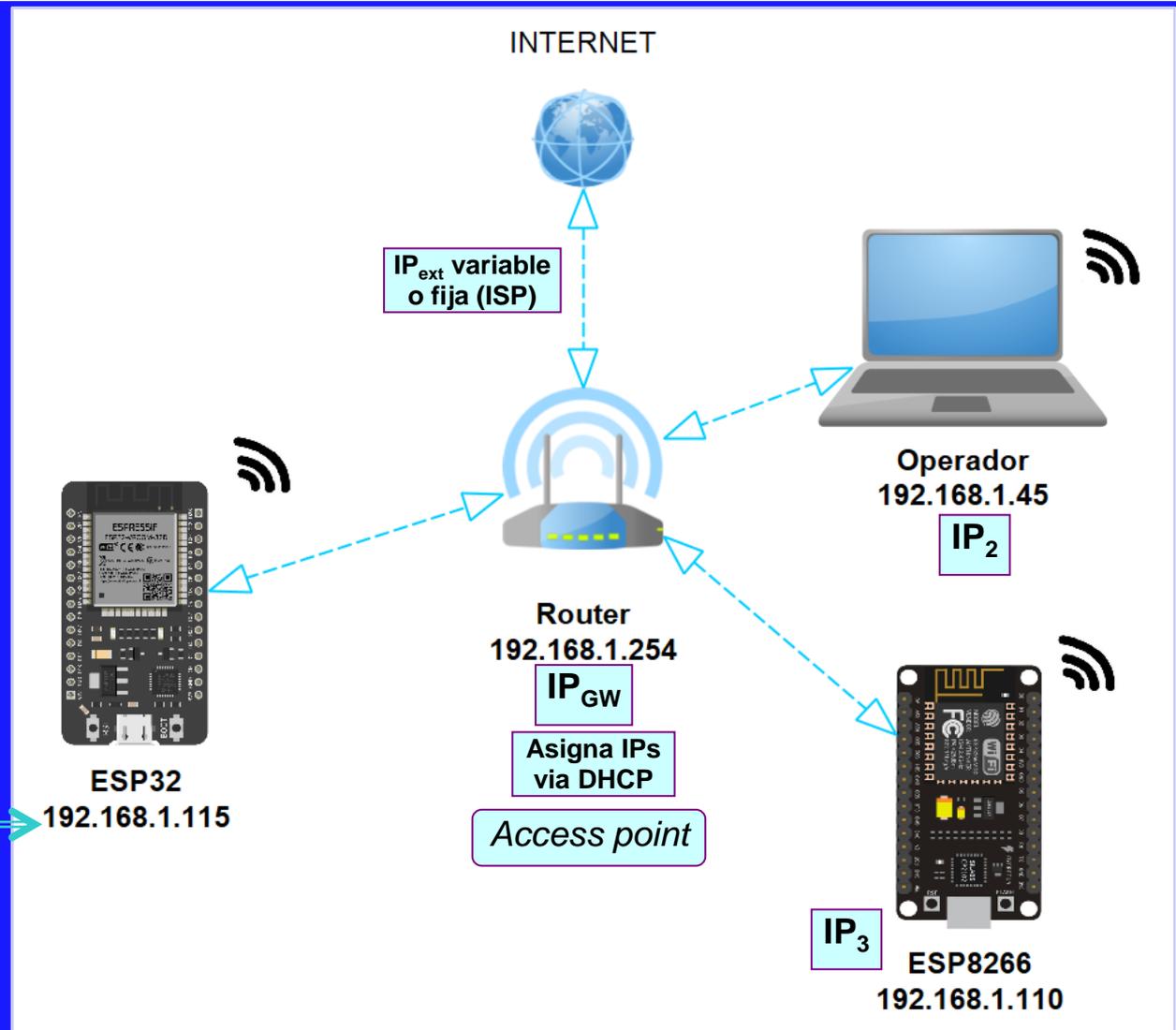


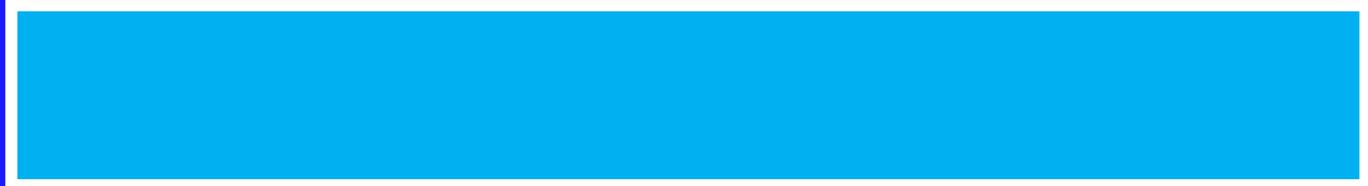


Configuración (ib) – ESPs como Estación WiFi

Direcciones típicas en una red interna

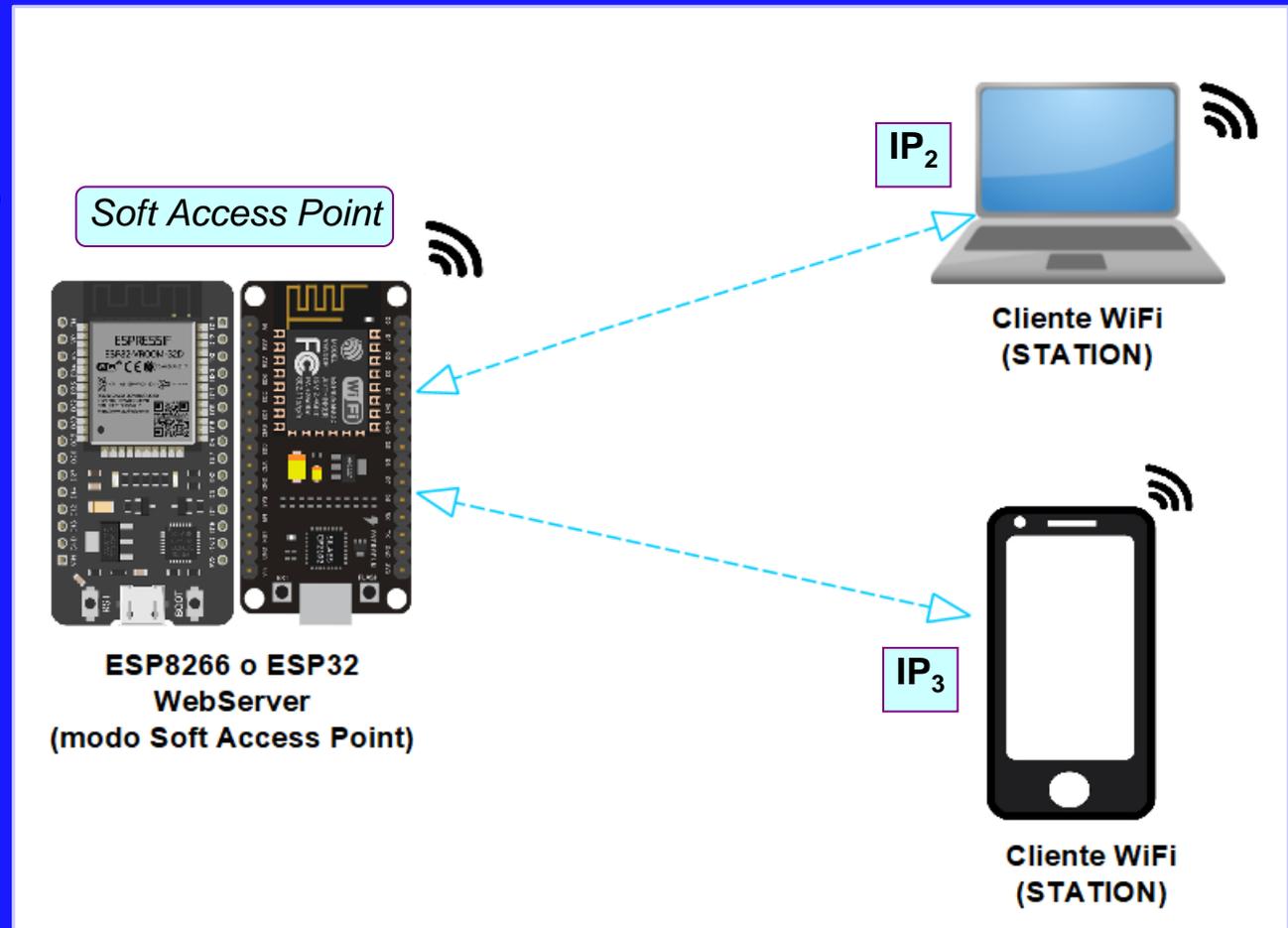
Aplicación	
Transporte	TCP
Red	IP ₁
Enlace	WiFi
Físico	RF





Configuración (ii) – ESPs como “Soft Access Point”

(sin vinculación
a Internet)



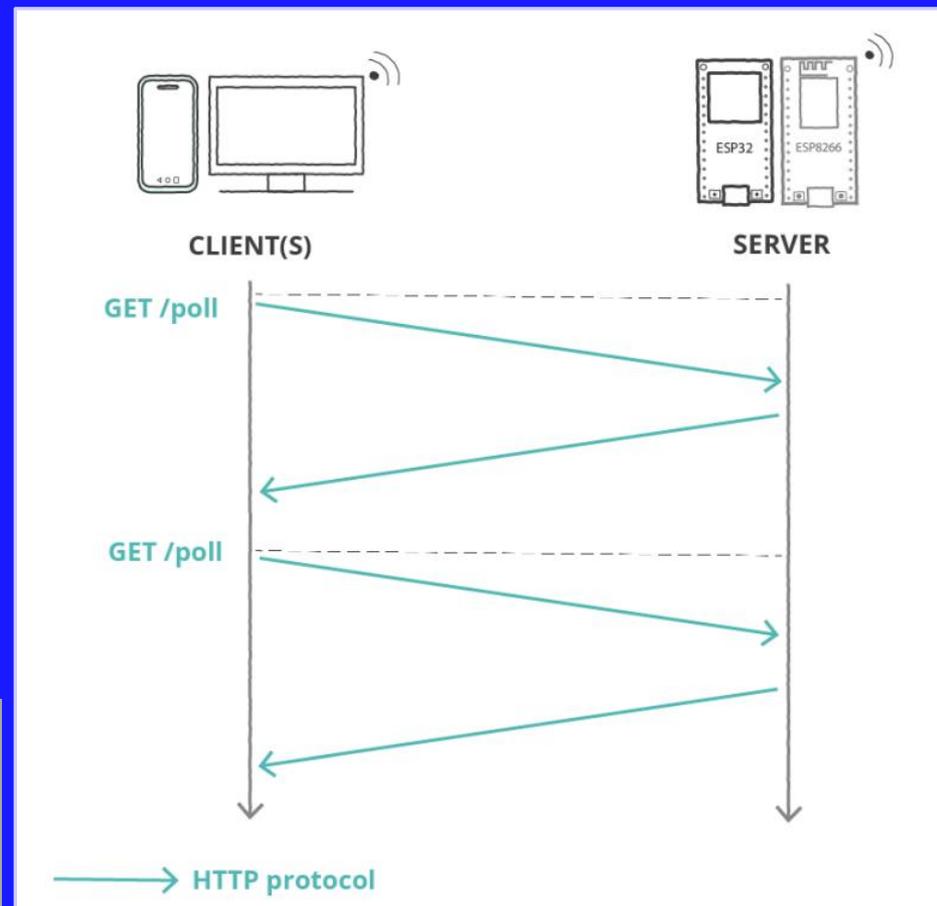
Comunicación Cliente-Servidor (i) – HTTP Polling

El cliente “consulta” o “encuesta” (*polls*) al servidor periódicamente utilizando comandos GET/poll. El Servidor ESP solamente responde ante consultas del cliente (usuario)

Gráficos de:

R.Santos, S.Santos “Building Web Servers with ESP8266/ESP32”
2ndEd 2021 ebook

<https://randomnerdtutorials.com/courses>



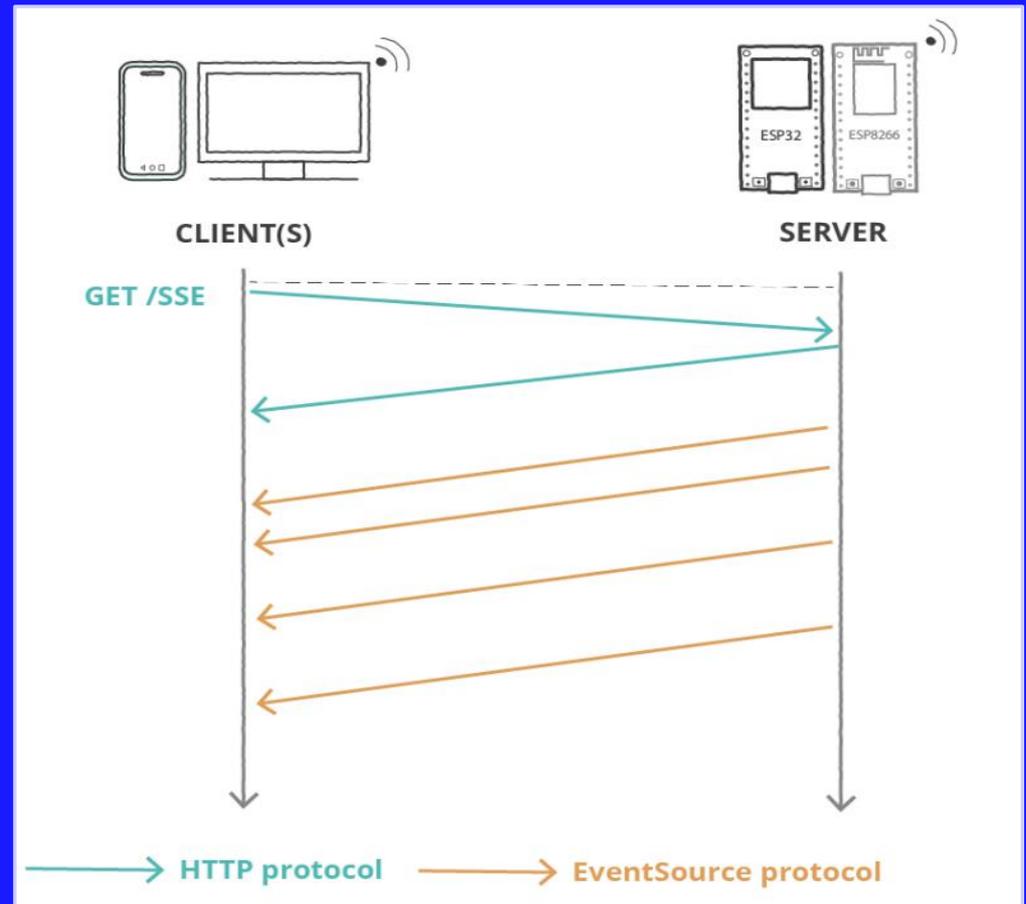
Comunicación Cliente-Servidor (ii) – Eventos del Lado Servidor (SSE)

El cliente inicia el modo SSE, después de lo cual el Servidor ESP envía automáticamente respuestas a eventos (por ejemplo lecturas de sensor, pulsaciones de botones, etc.) El cliente no puede enviar datos hacia el servidor después del protocolo inicial GET/SSE, salvo que se lo termine.

Gráficos de:

R.Santos, S.Santos "Building Web Servers with ESP8266/ESP32"
2ndEd 2021 ebook

<https://randomnerdtutorials.com/courses>



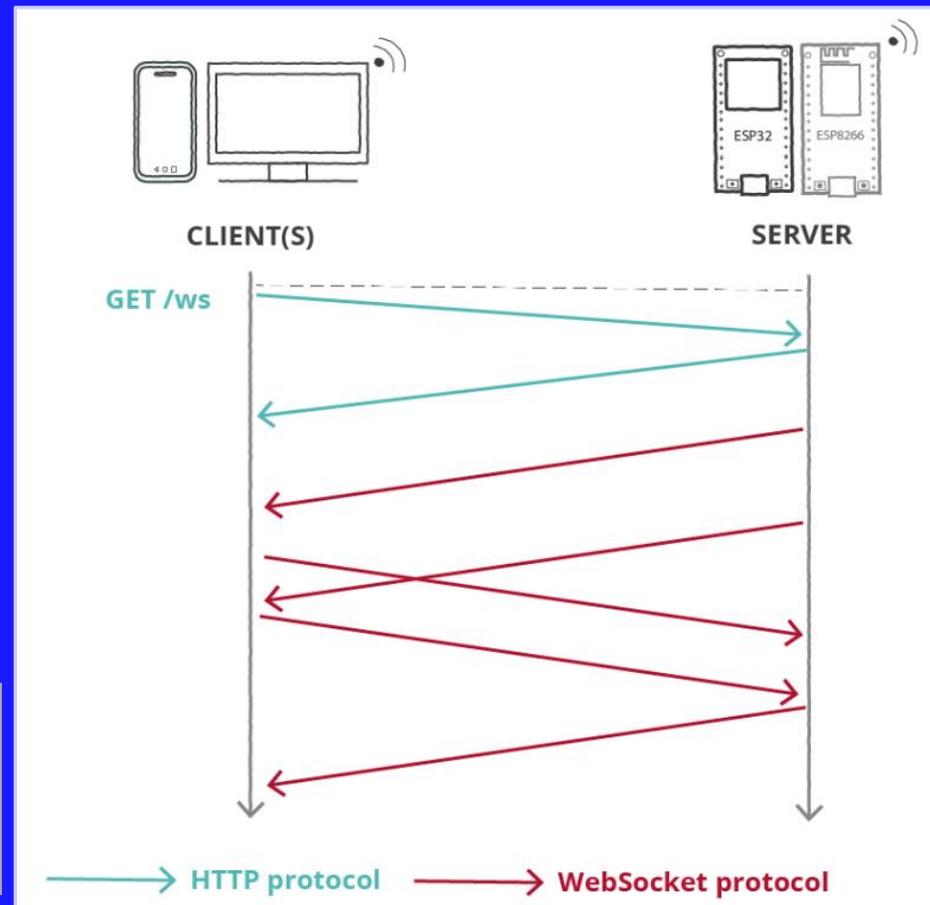
Comunicación Cliente-Servidor (iii) – WebSocket

Un WebSocket es una conexión persistente bidireccional, que inicia el cliente con el Servidor ESP, después de lo cual ambas partes pueden enviar y recibir datos usando una conexión TCP.

Gráficos de:

R.Santos, S.Santos "Building Web Servers with ESP8266/ESP32"
2ndEd 2021 ebook

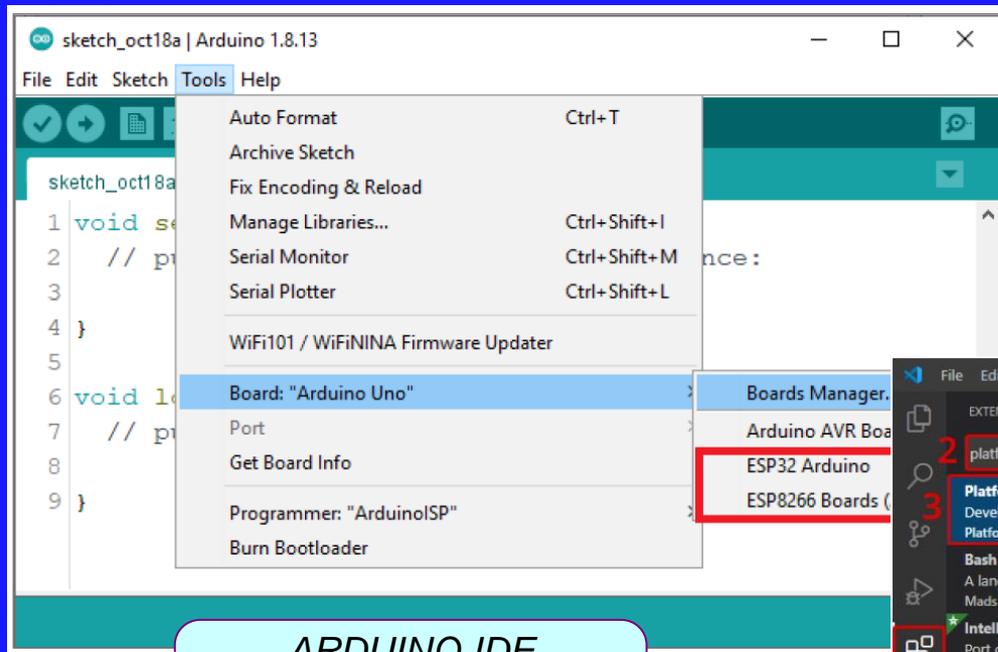
<https://randomnerdtutorials.com/courses>



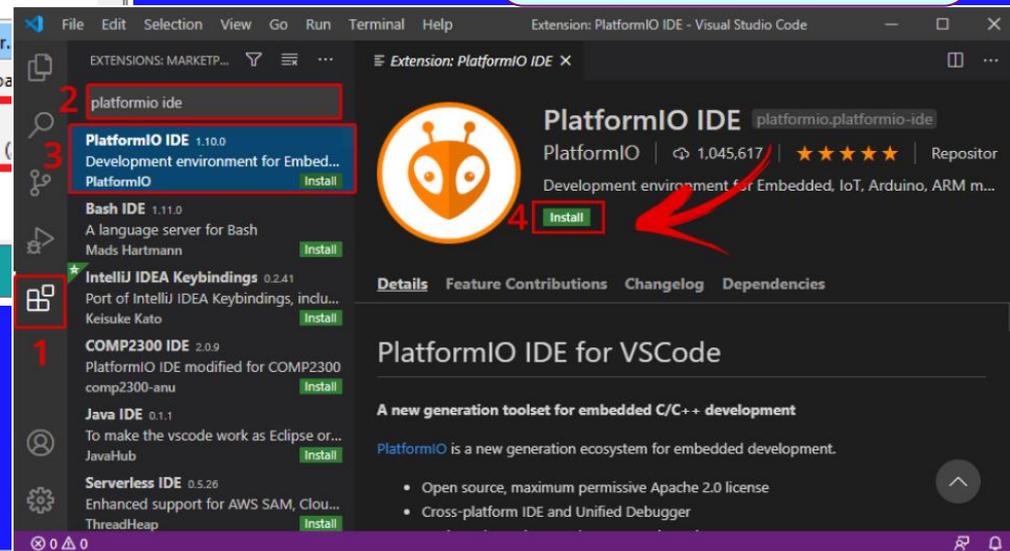
Primer Proyecto con ESP8266 / ESP32 “Hola Mundo” programable desde Arduino IDE o VSC / PIO

*Ambos GRATUITOS y
disponibles para Windows,
Ubuntu o MacOS*

VISUAL STUDIO CODE
<https://code.visualstudio.com/download>



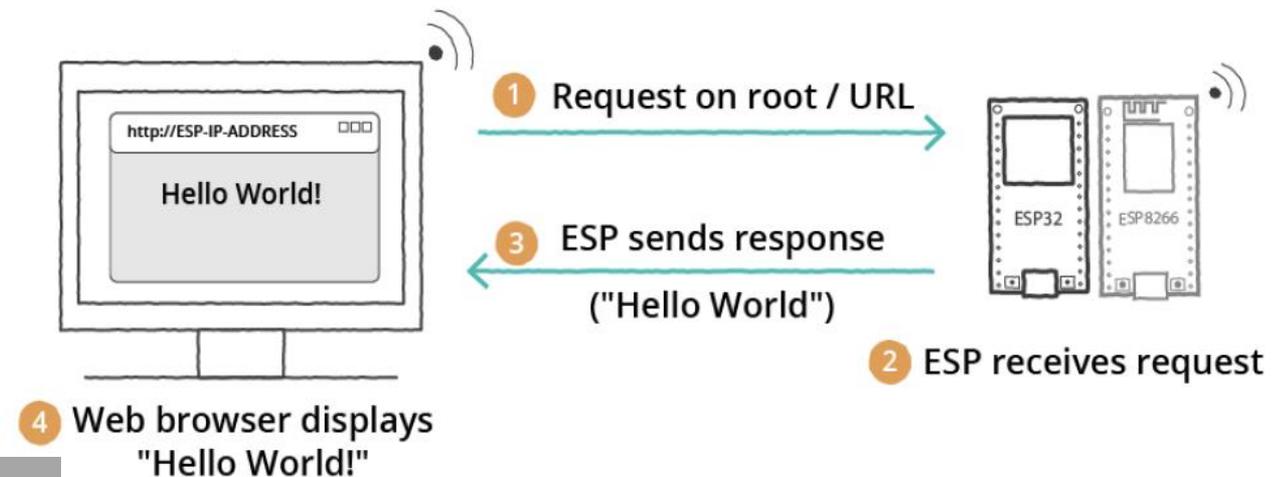
ARDUINO IDE
<https://www.arduino.cc/en/software>



Primer Proyecto con ESP8266 / ESP32 “Hola Mundo”

(i) HTML en código .cpp

Project Overview

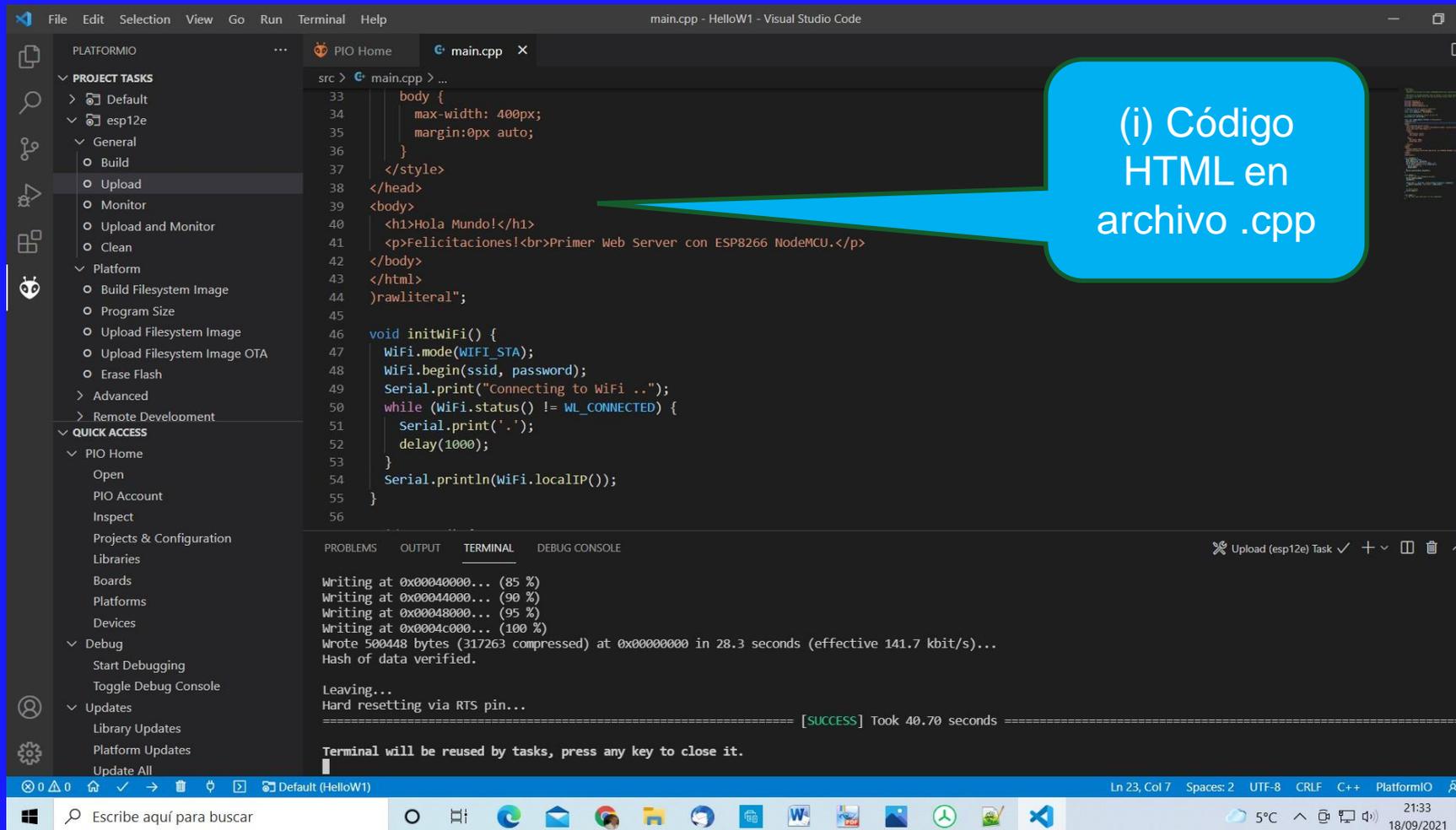


Gráficos de:

R.Santos, S.Santos "Building Web Servers with ESP8266/ESP32"
2ndEd 2021 ebook

<https://randomnerdtutorials.com/courses>

Primer Proyecto con ESP8266 / ESP32 “Hola Mundo” (alt-i)



The screenshot shows the Visual Studio Code interface with the following elements:

- Editor:** Displays the content of `main.cpp`, which is HTML code:


```

src > main.cpp > ...
33     body {
34         max-width: 400px;
35         margin:0px auto;
36     }
37     </style>
38 </head>
39 <body>
40     <h1>Hola Mundo!</h1>
41     <p>Felicitaciones!<br>Primer Web Server con ESP8266 NodeMCU.</p>
42 </body>
43 </html>
44 )rawliteral";
45
46 void initWiFi() {
47     WiFi.mode(WIFI_STA);
48     WiFi.begin(ssid, password);
49     Serial.print("connecting to WiFi ..");
50     while (WiFi.status() != WL_CONNECTED) {
51         Serial.print('.');
52         delay(1000);
53     }
54     Serial.println(WiFi.localIP());
55 }
56
      
```
- Terminal:** Shows the upload process:


```

Writing at 0x00040000... (85 %)
Writing at 0x00044000... (90 %)
Writing at 0x00048000... (95 %)
Writing at 0x0004c000... (100 %)
Wrote 500448 bytes (317263 compressed) at 0x00000000 in 28.3 seconds (effective 141.7 kbit/s)...
Hash of data verified.

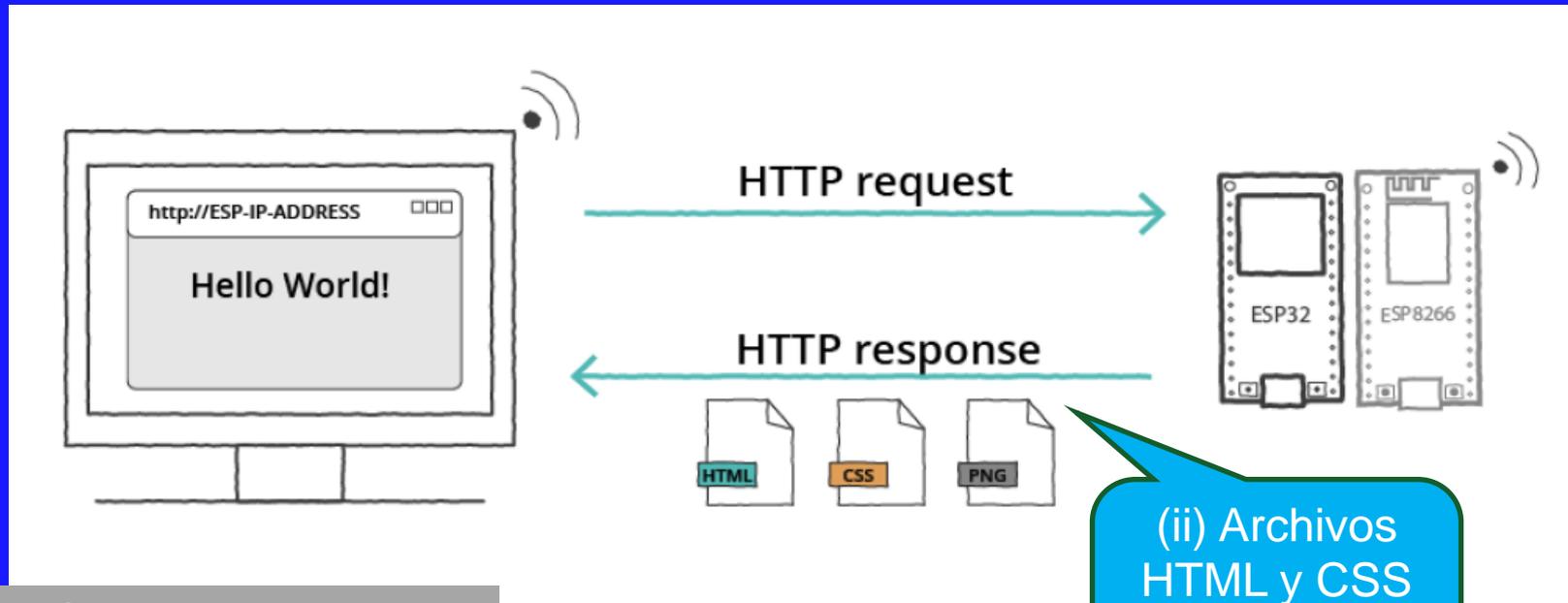
Leaving...
Hard resetting via RTS pin...

===== [SUCCESS] Took 40.70 seconds =====

Terminal will be reused by tasks, press any key to close it.
      
```
- Callout:** A blue speech bubble points to the HTML code in the editor, containing the text: (i) Código HTML en archivo .cpp

Primer Proyecto con ESP8266 / ESP32 “Hola Mundo”

(ii) HTML almacenado en Flash File System (SPIFFS)

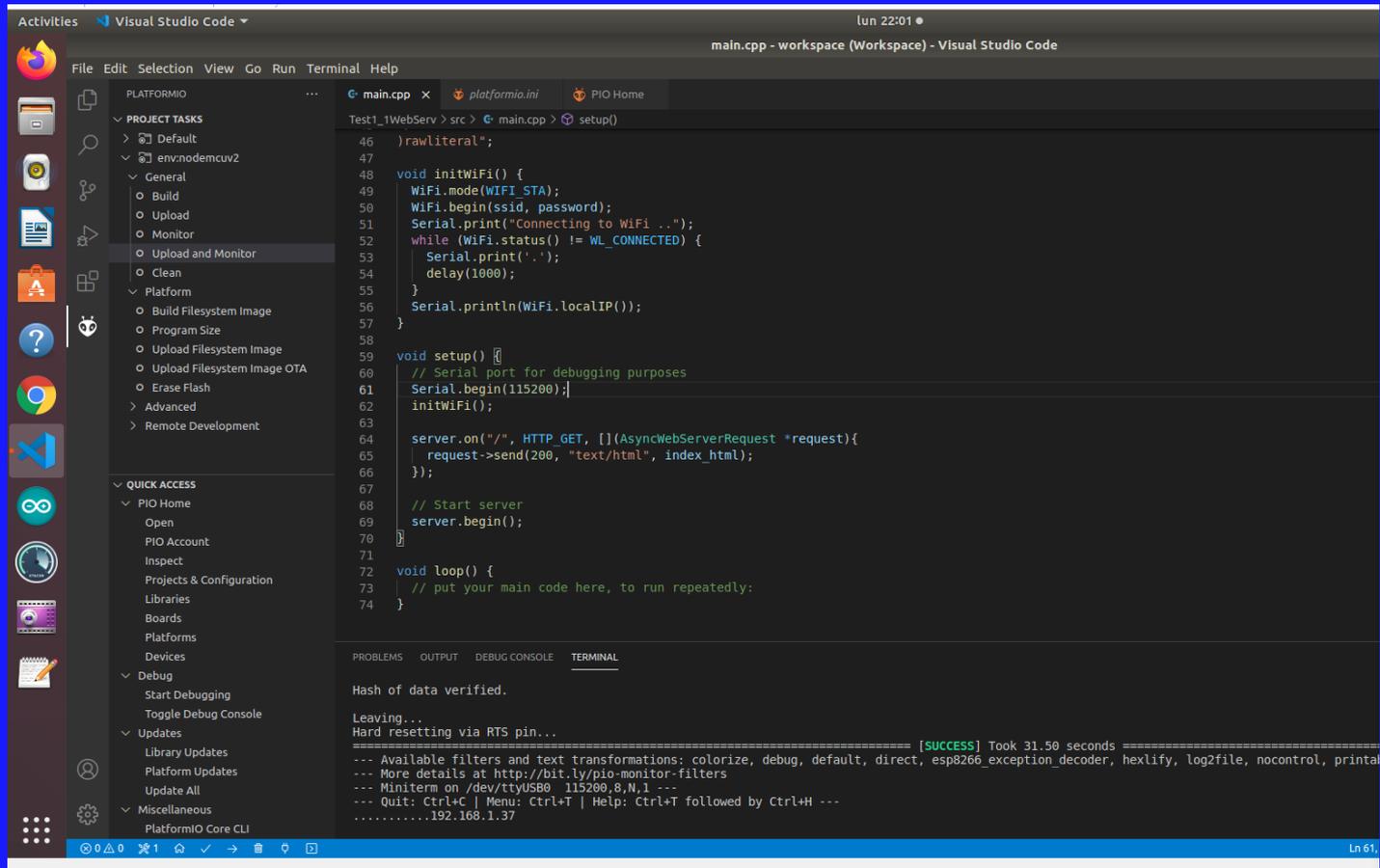


Gráficos de:

R.Santos, S.Santos “Building Web Servers with ESP8266/ESP32”
2ndEd 2021 ebook

<https://randomnerdtutorials.com/courses>

Interfaz Web a Sistemas Embebidos Existentes (VSC / PIO bajo Ubuntu 18.04 LTS)

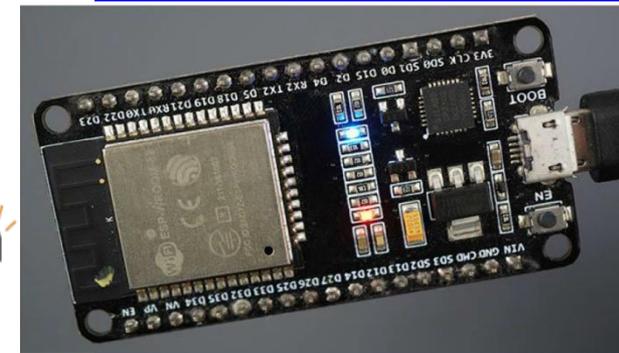
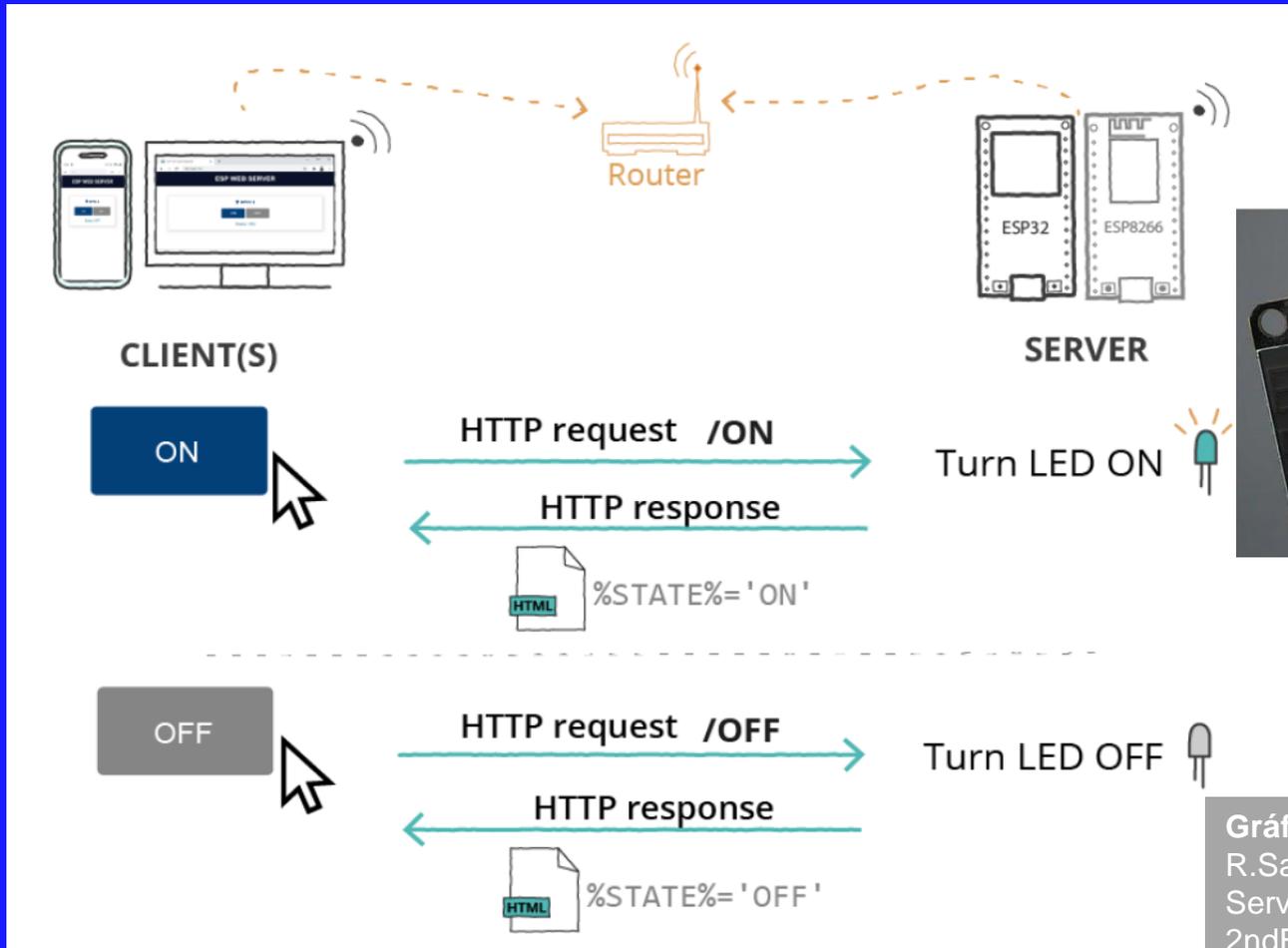


```
main.cpp - workspace (Workspace) - Visual Studio Code
File Edit Selection View Go Run Terminal Help
PLATFORMIO
PROJECT TASKS
  Default
  env:nodemcu2
  General
    Build
    Upload
    Monitor
    Upload and Monitor
    Clean
  Platform
    Build Filesystem Image
    Program Size
    Upload Filesystem Image
    Upload Filesystem Image OTA
    Erase Flash
  Advanced
  Remote Development
QUICK ACCESS
  PIO Home
  Open
  PIO Account
  Inspect
  Projects & Configuration
  Libraries
  Boards
  Platforms
  Devices
  Debug
  Start Debugging
  Toggle Debug Console
  Updates
  Library Updates
  Platform Updates
  Update All
  Miscellaneous
  PlatformIO Core CLI

Test1_1WebServ > src > main.cpp > setup()
46 )rawliteral";
47
48 void initWiFi() {
49   WiFi.mode(WIFI_STA);
50   WiFi.begin(ssid, password);
51   Serial.print("Connecting to WIFI ..");
52   while (WiFi.status() != WL_CONNECTED) {
53     Serial.print('.');
54     delay(1000);
55   }
56   Serial.println(WiFi.localIP());
57 }
58
59 void setup() {
60   // Serial port for debugging purposes
61   Serial.begin(115200);
62   initWiFi();
63
64   server.on("/", HTTP_GET, [](AsyncWebServerRequest *request){
65     request->send(200, "text/html", index_html);
66   });
67
68   // Start server
69   server.begin();
70
71 }
72
73 void loop() {
74   // put your main code here, to run repeatedly:
75 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Hash of data verified.
Leaving...
Hard resetting via RTS pin...
===== [SUCCESS] Took 31.50 seconds =====
--- Available filters and text transformations: colorize, debug, default, direct, esp8266_exception_decoder, hexlify, log2file, nocontrol, print
--- More details at http://bit.ly/pio-monitor-filters
--- Miniterm on /dev/ttyUSB0 115200,8,N,1 ---
--- Quit: Ctrl+C | Menu: Ctrl+T | Help: Ctrl+T followed by Ctrl+H ---
.....192.168.1.37
```

Control de Entrada / Salida vía Webserver ESP



Gráficos de:
 R.Santos, S.Santos "Building Web Servers with ESP8266/ESP32"
 2ndEd 2021 ebook
<https://randomnerdtutorials.com/courses>

Control de Entrada / Salida vía Webserver ESP

(ii) Archivos HTML y CSS en /data

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>ESP IOT DASHBOARD /+3+3
5 <meta charset="utf-8" /+3+3
6 <link rel="stylesheet" type="text/css" href="style.css">
7 <link rel="icon" type="image/png" href="favicon.png">
8 <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.2/css/all.css">
9 </head>
10 <body>
11 <div class="topnav">
12 <h1>ESP WEB SERVER en ESP8266</h1>
13 </div>
14 <div class="content">
15 <div class="card-grid">
16 <div class="card">
17 <p class="card-title">GPIO 2</p>
18 <p>
19 <a href="on"><button class="btn btn-primary">ON</button>
20 <a href="off"><button class="btn btn-secondary">OFF</button>
21 </p>
22 <p class="state">State: OFF</p>
23 </div>
24 </div>
25 </div>
26 </body>
27 </html>
  
```

192.168.1.35/off

alpha www.ieee.org/orga... www.google.com Symbaloo EDU_CIAA LyR_3 MIoT Nueva p

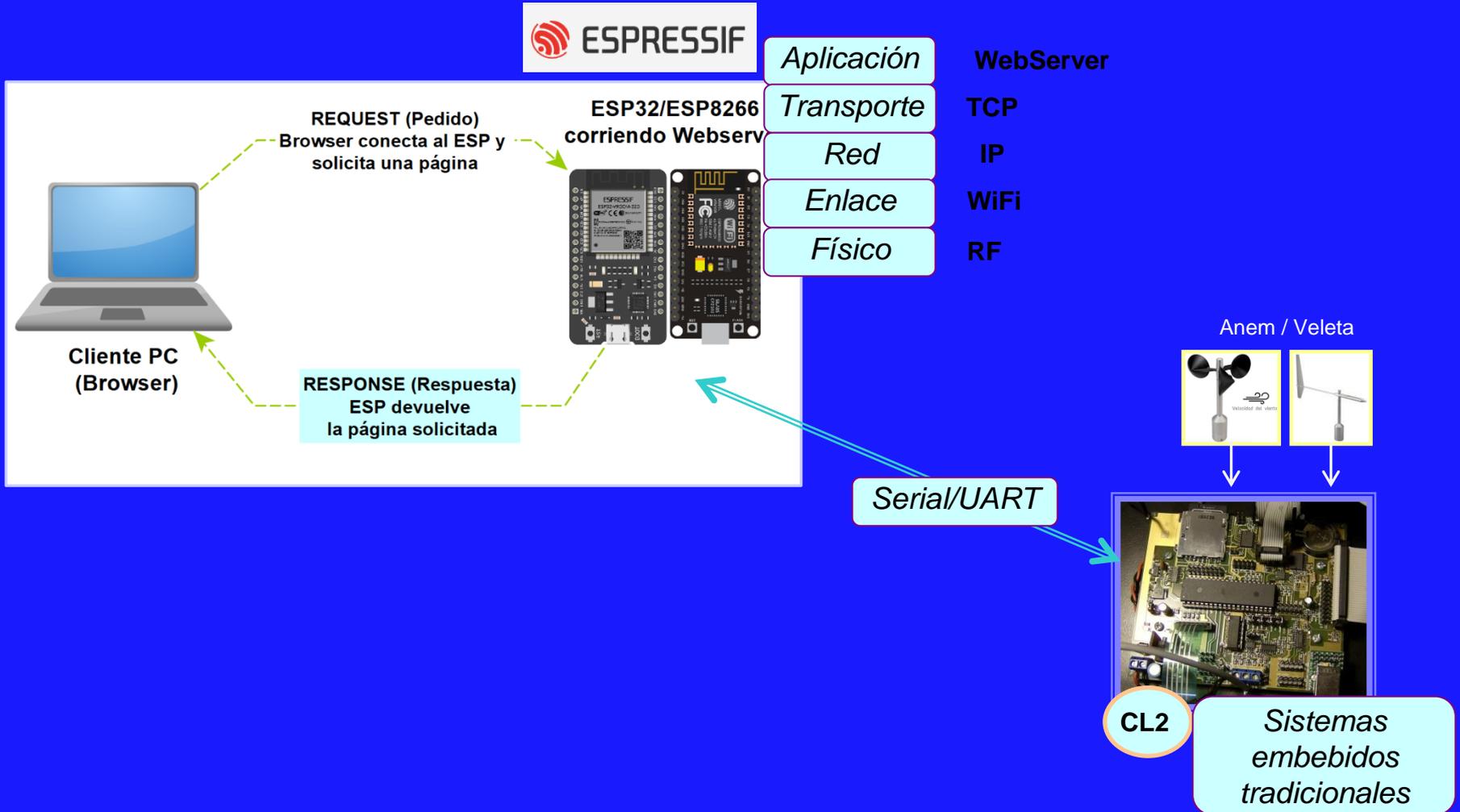
ESP WEB SERVER en ESP8266

💡 GPIO 2

ON OFF

State: OFF

Interfaz Web a sistema embebido tradicional



Que es un sistema embebido?

“Un sistema embebido (*embedded*) es cualquier aplicación en que una computadora dedicada se construye como parte del sistema”

(Jack Ganssle - The Art of Programming Embedded Systems)



Instrumental



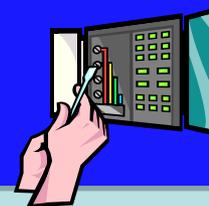
Celulares



Microondas

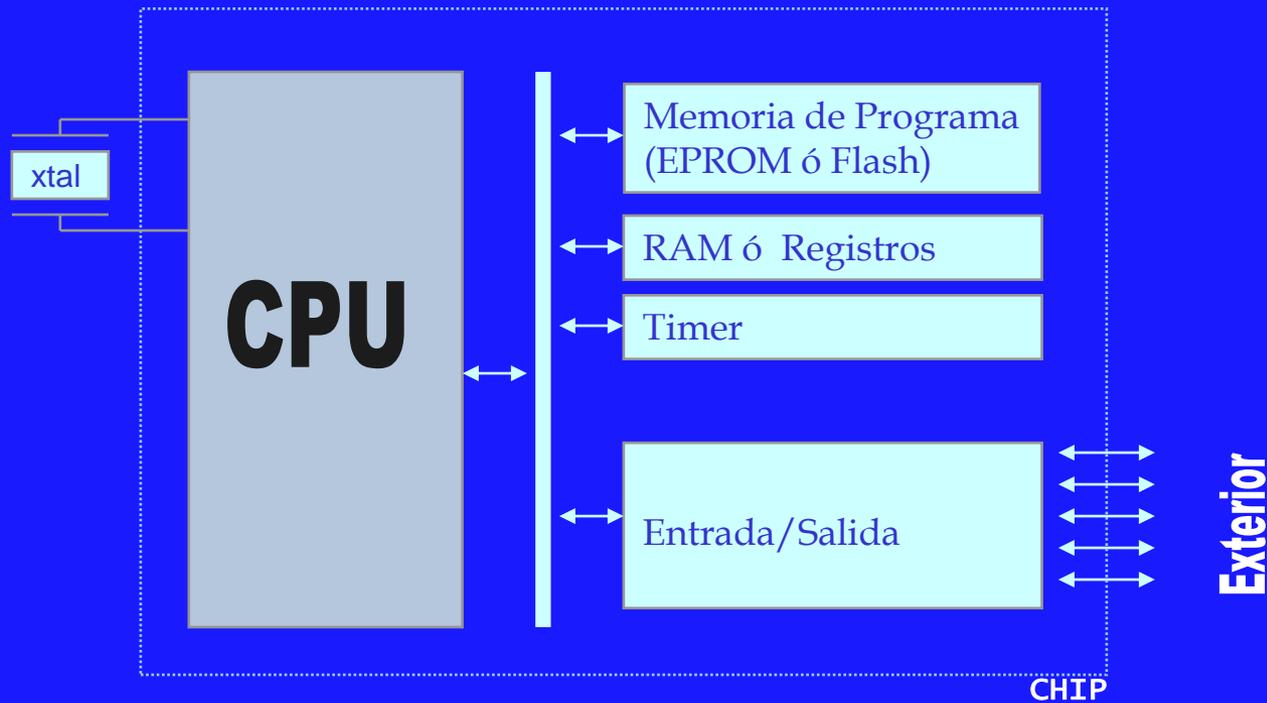


Vehículos



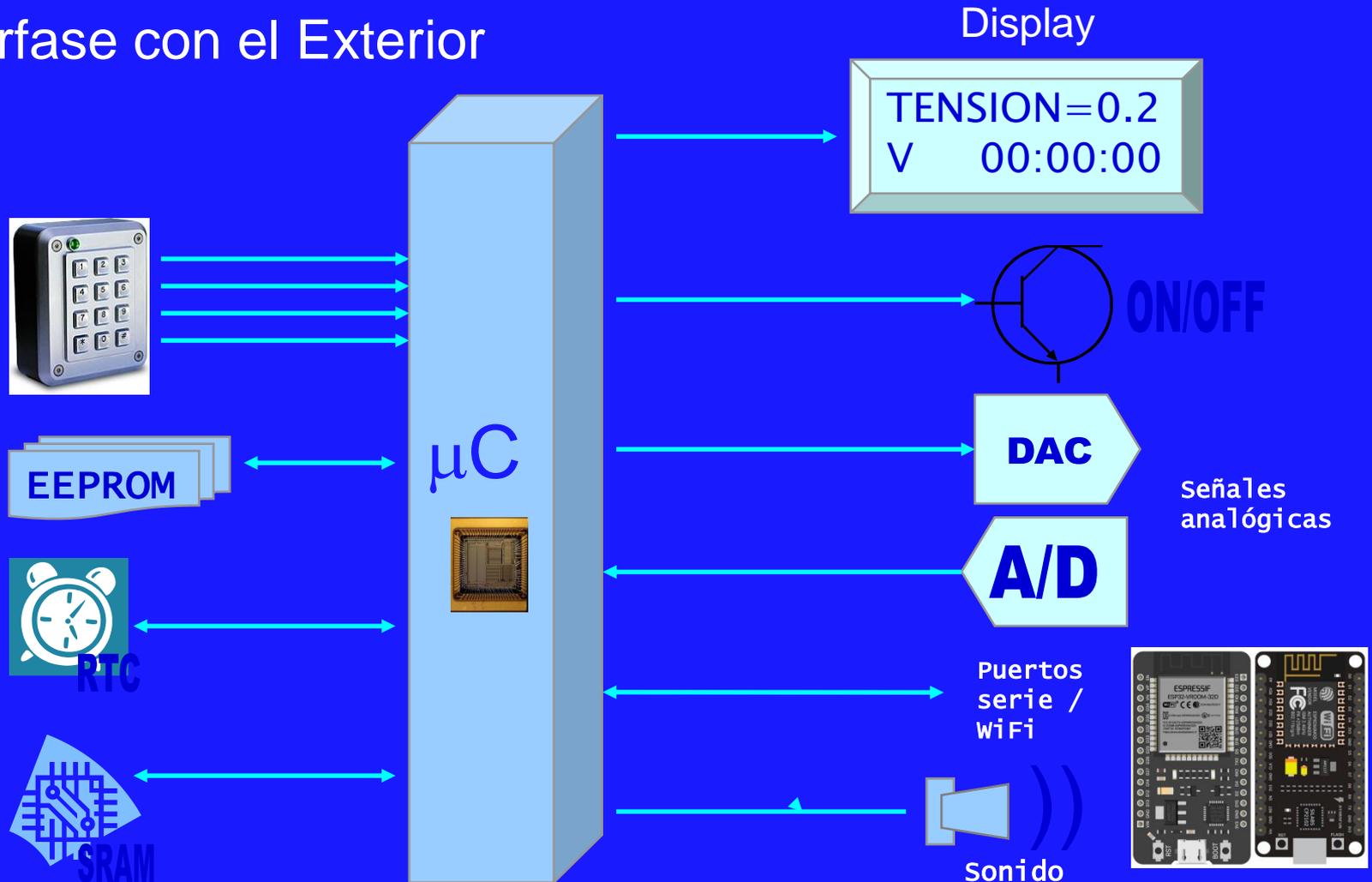
Alarmas

COMPONENTES DE UN MICROCONTROLADOR ELEMENTAL



Que es un sistema embebido?

Interfase con el Exterior



Caso demostración – Conexión a SISMED/SJ24 con módulo ESP Sistema de Medición SJ24 UA Puerto San Julian (Instalado 2013)



Puerto San
Julián,
Argentina



Predio
Chacra
UASJ-UNPA



Sistema de Medición SISMED/SJ24



VISTA DEL SISTEMA

Sistema de
Registro



Rectificador,
reguladores, cargador



Control
Baterías

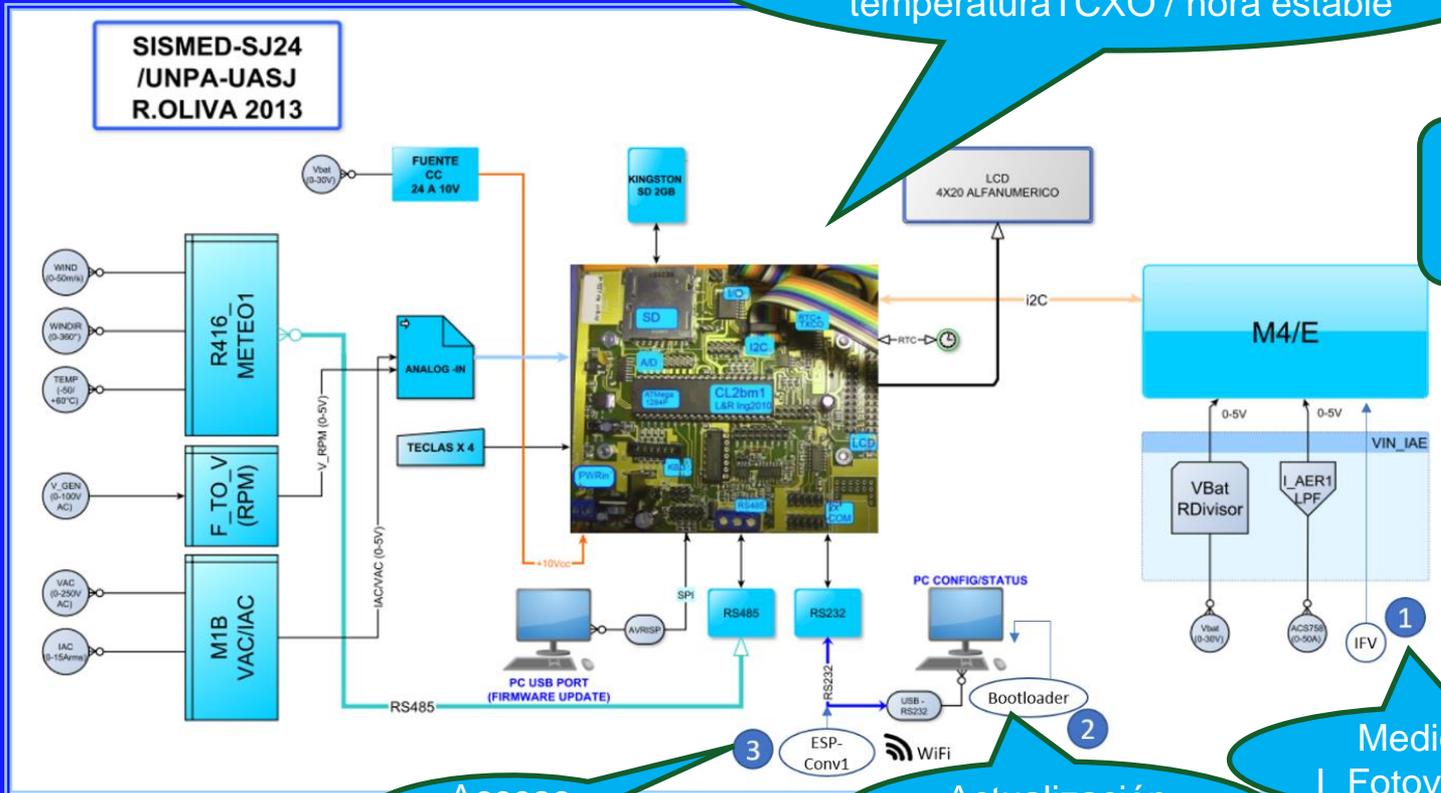


Sistema de Medición SISMED/SJ24

ESTRUCTURA DE

Reloj interno RTC con oscilador compensado por temperatura TCXO / hora estable

(1), (2), (3) agregados posteriores



Acceso WiFi

Actualización Firmware Serial

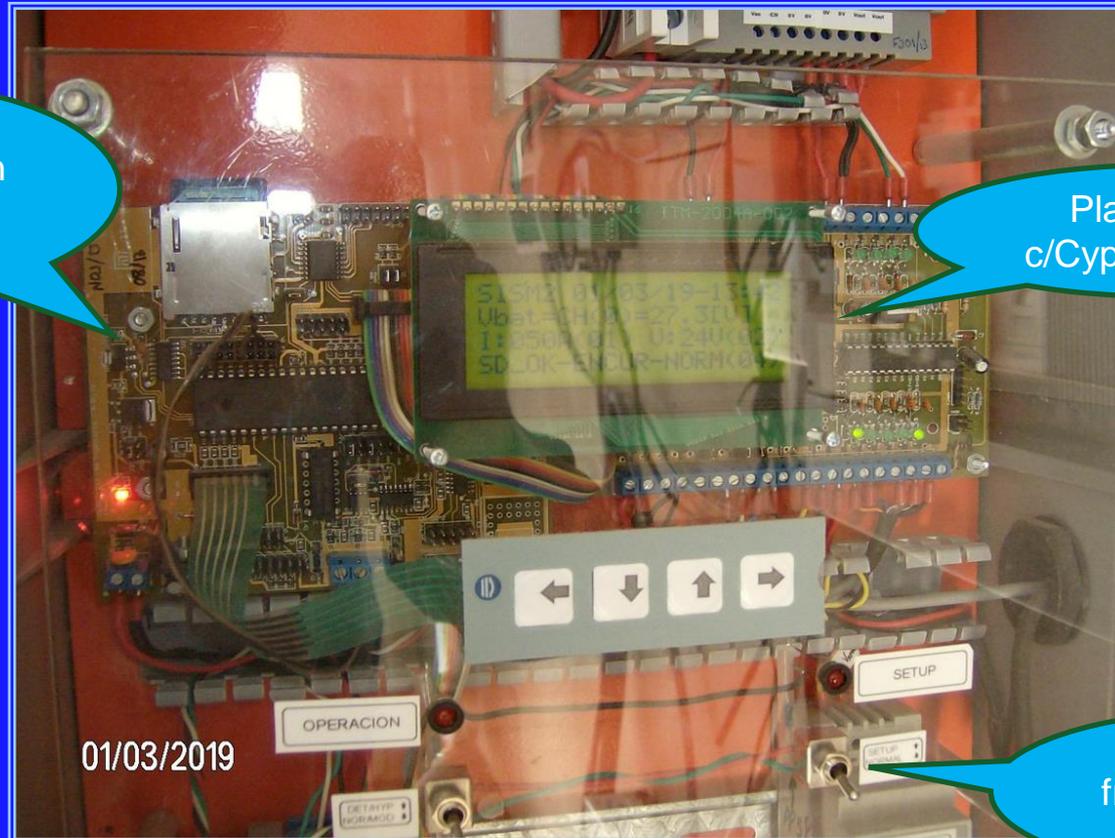
Medición I_Fotovoltaica

Sistema de Medición SISMED/SJ24

GABINETE PRINCIPAL / MÓDULOS DE HARDWARE

Placa CL2b con
ATMega1284P

Placa M4/E
c/Cypress 29466



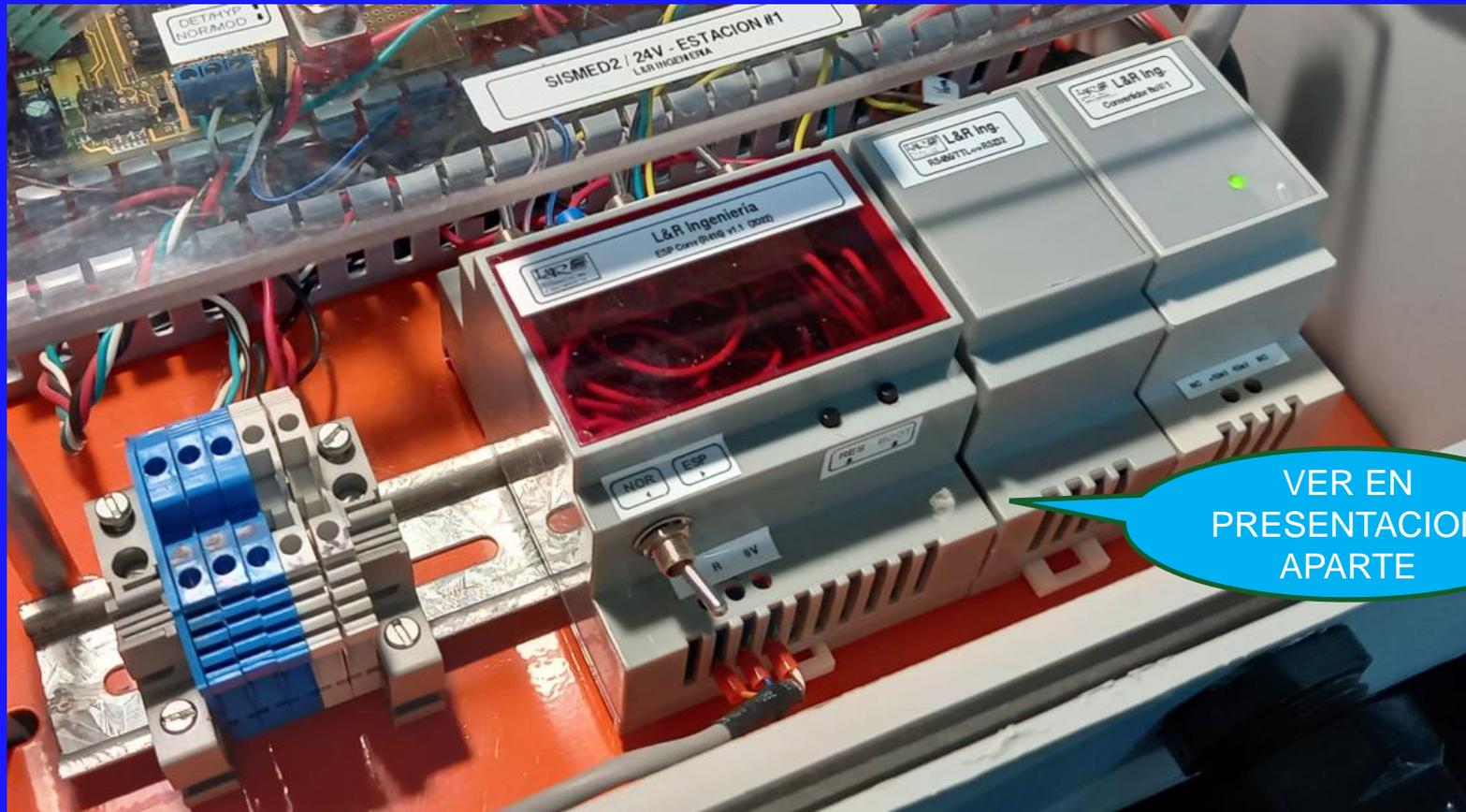
Switches
frontales: Modo
Operación



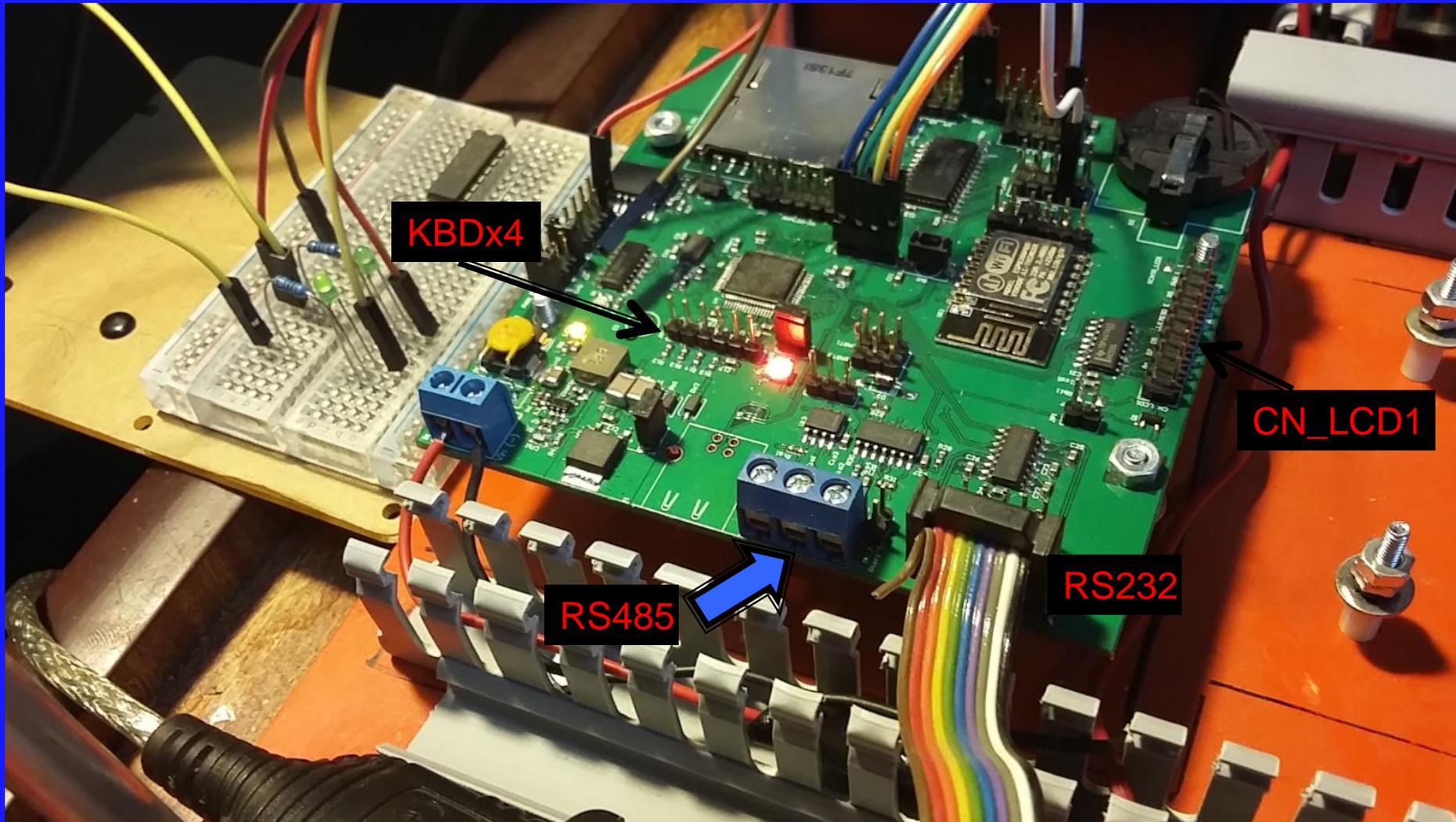
SISTEMAS
EMBEBIDOS
2022



Sistema de Medición SISMED/SJ24 AGREGADO DE MODULOS WI-FI /2022



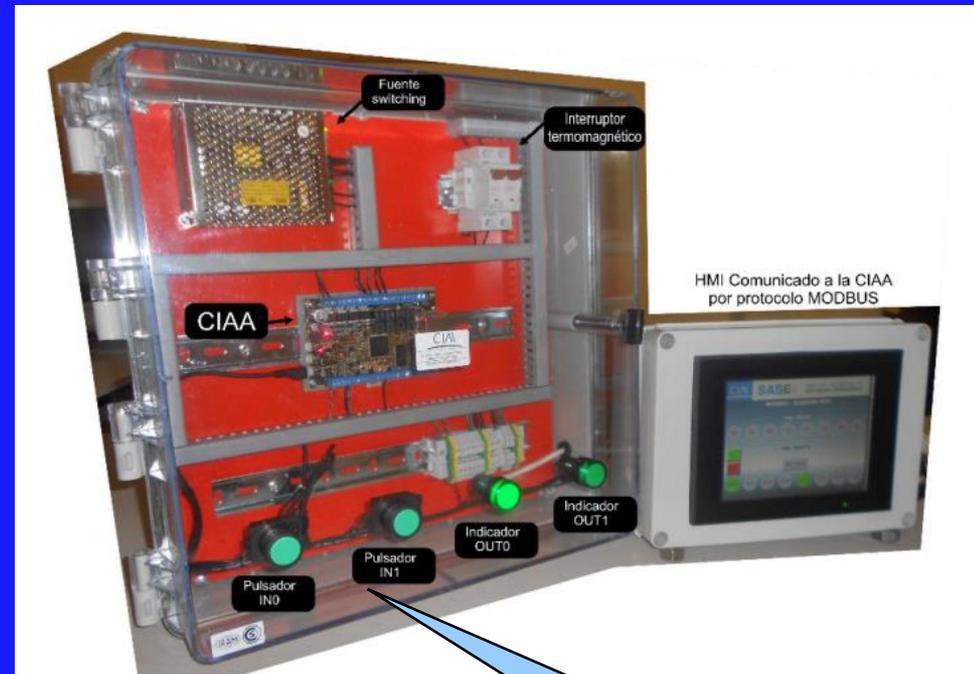
Desarrollo de nueva placa CL3 (2018)



ORIGENES DE LA CIAA

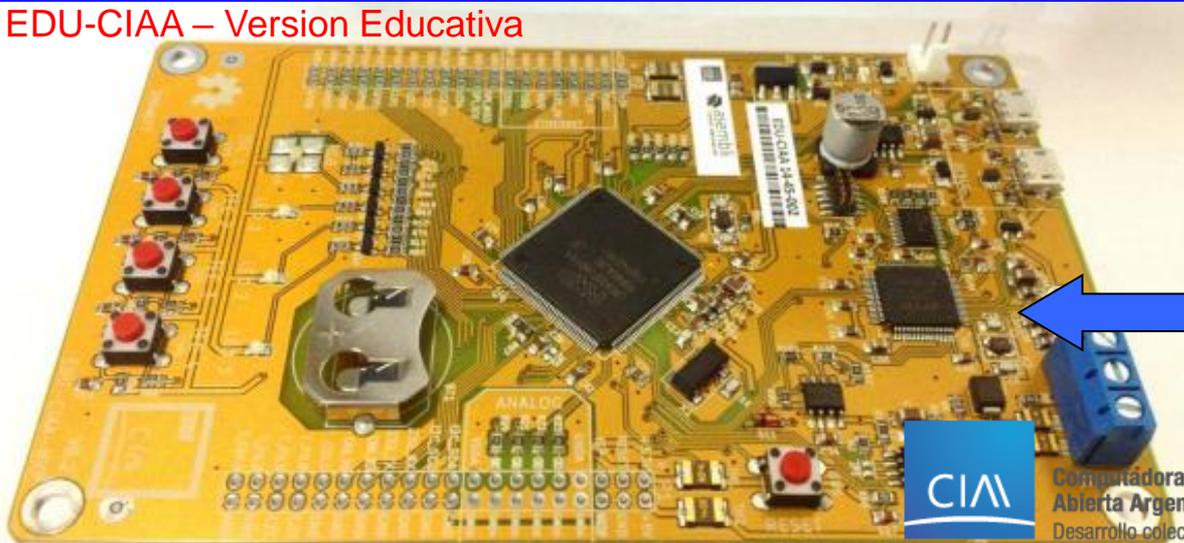
La Computadora Industrial Abierta Argentina (CIAA) se comenzó a gestar en julio de 2013, cuando el Ministerio de Industria de la Nación y la SPU convocaron a la Asociación Civil para la Investigación, Promoción y Desarrollo de los Sistemas Electrónicos Embebidos (ACSE) y a la Cámara de Industrias Electrónicas, Electromecánicas y Luminotécnicas (CADIEEL) a participar en un plan para incorporar tecnología nacional en la cadena de producción, específicamente en los puntos en que se importa casi todo (PLCs, Controladores especiales).

Características: Diseño colaborativo, Licencia abierta en hardware y software, participación de Universidades e Instituciones Técnicas de todo el país.



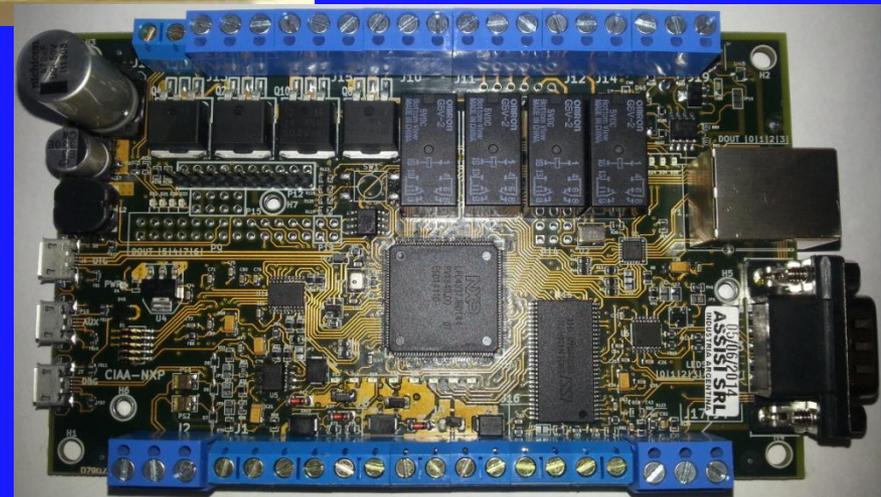
Prototipo en
funcionamiento
hacia fines de
2014

EDU-CIAA – Version Educativa

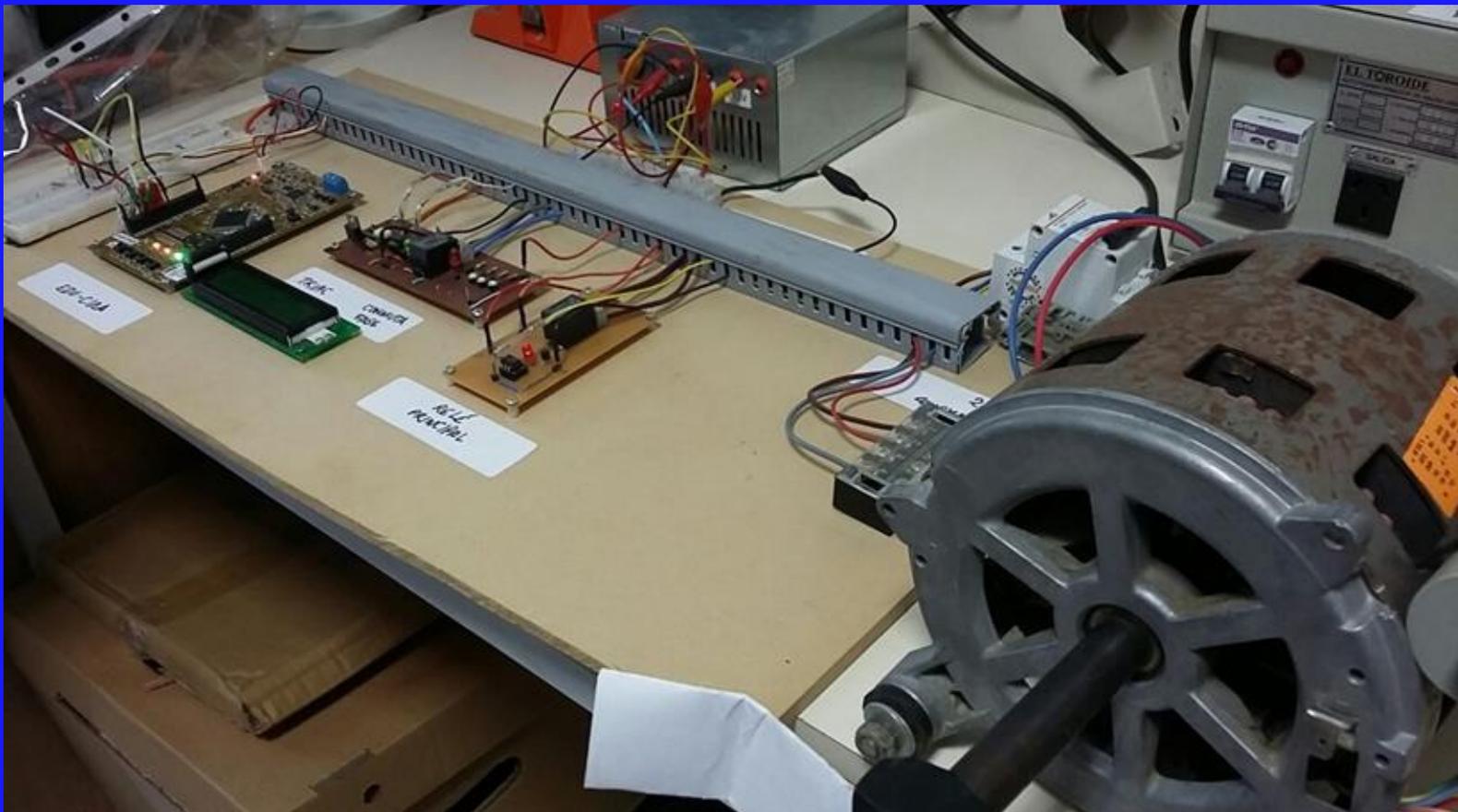


Primeras versiones:
CIAA y EDU-CIAA-NXP
(LPC4337 – NXP ARM
Cortex M4/M0)
EDU-CIAA: VERSION
EDUCATIVA DE BAJO
COSTO

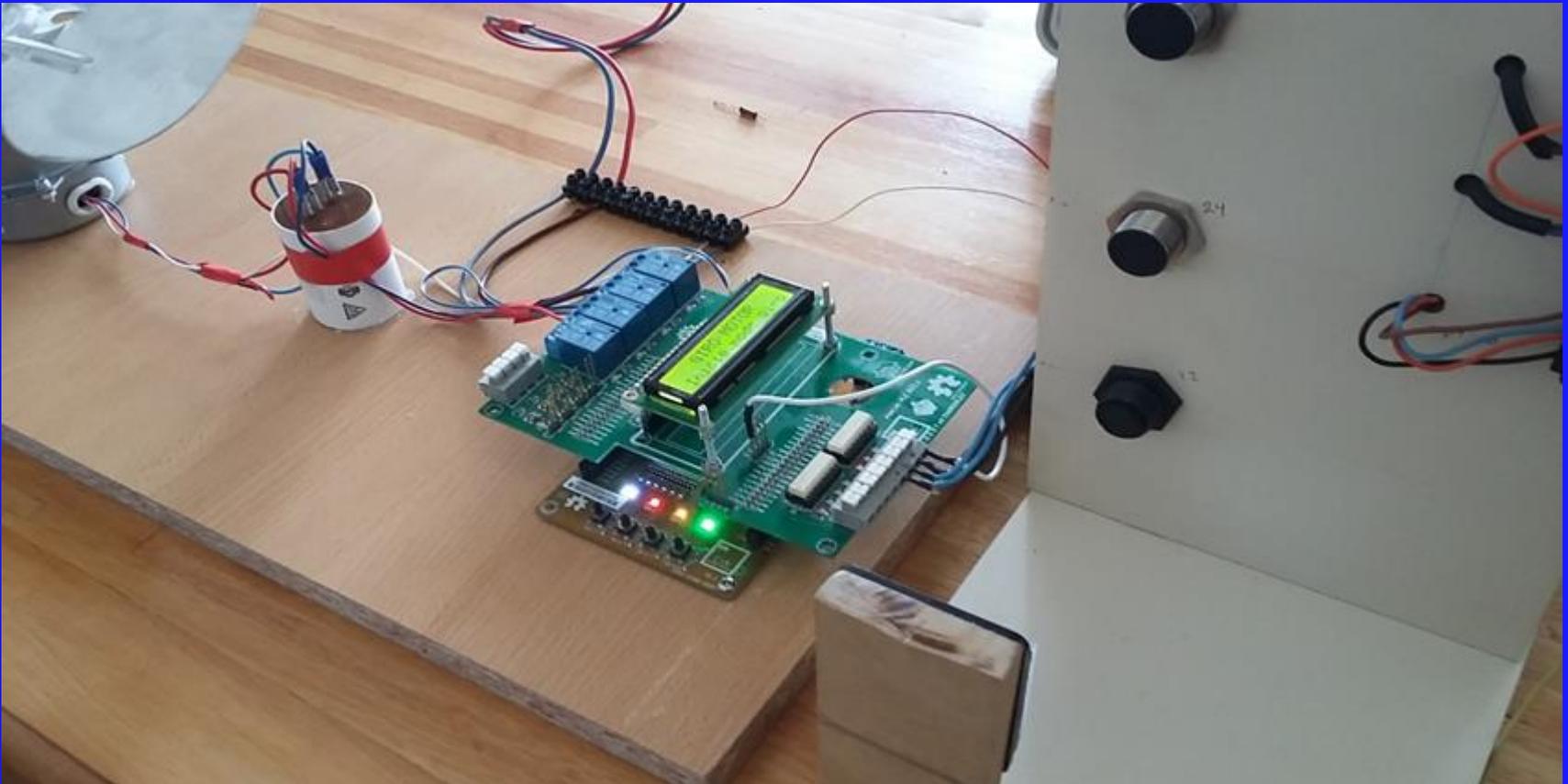
- Procesador dual core ARM-
Cortex M4/M0
- Orientado a aplicaciones de
control
- Se han adquirido varias EDU-
CIAA entre docentes e
Investigadores UNPA



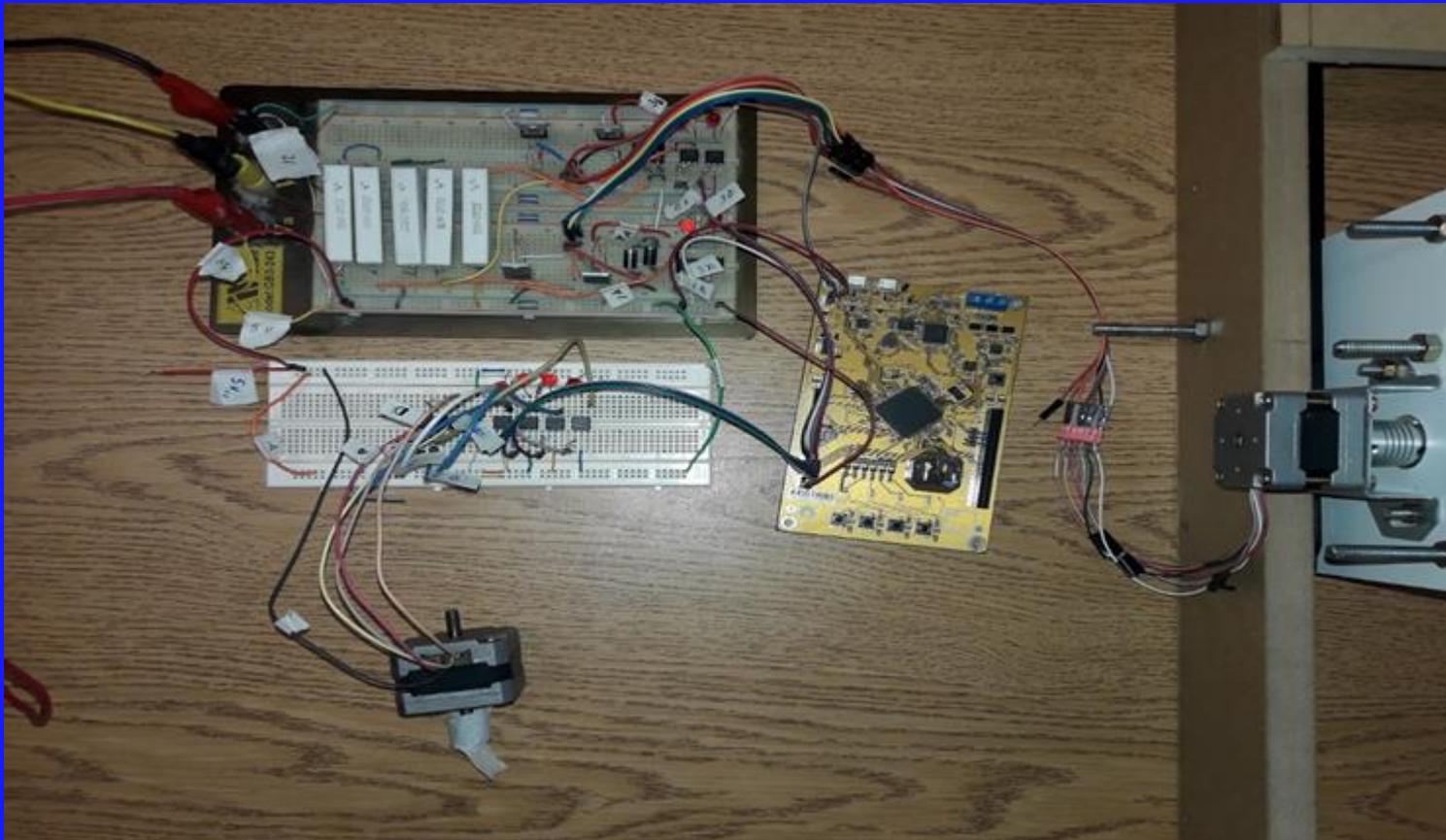
Ejemplo de aplicación 1: Inversor de giro motor monofásico (2019)



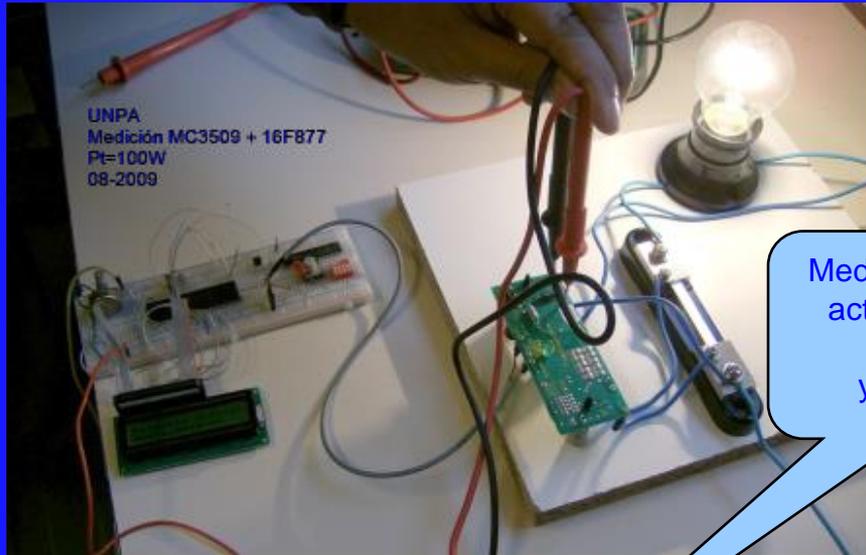
Ejemplo de aplicación 2: Control de ventilador con EDU-CIAA. sensores de proximidad y Poncho LCD



Ejemplo de aplicación 3: Control de motor PaP con EDU-CIAA.

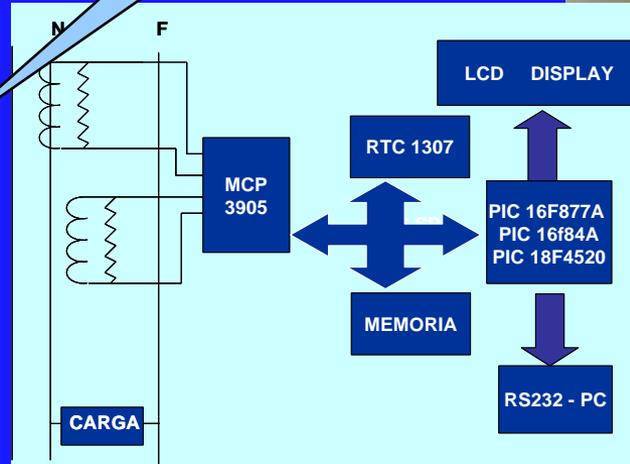
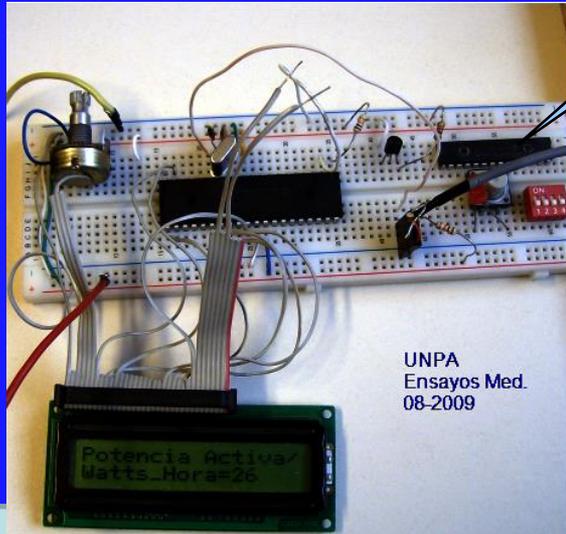
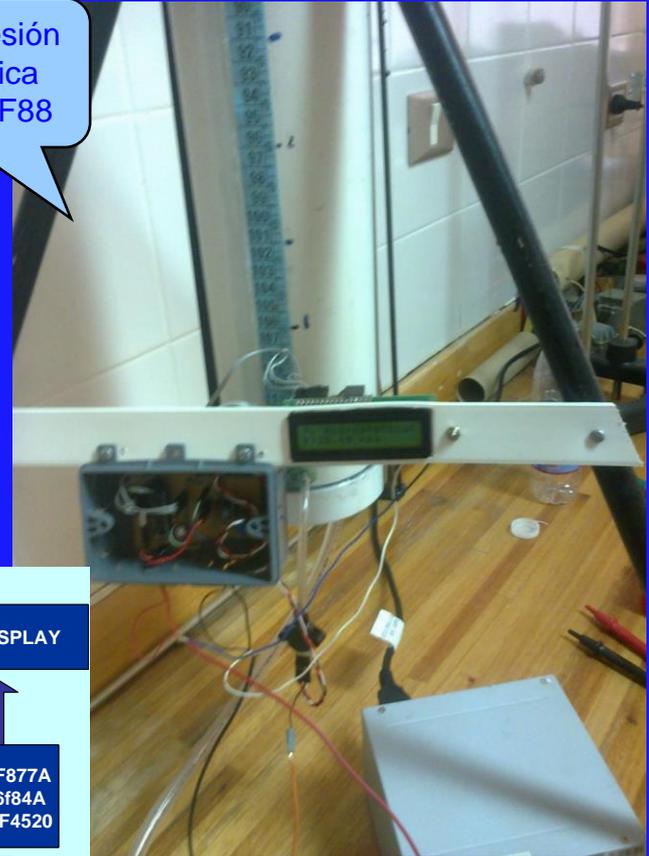


Sistemas Microchip (Ing. N.Cortez)



Medidor presión manométrica con PIC 16F88

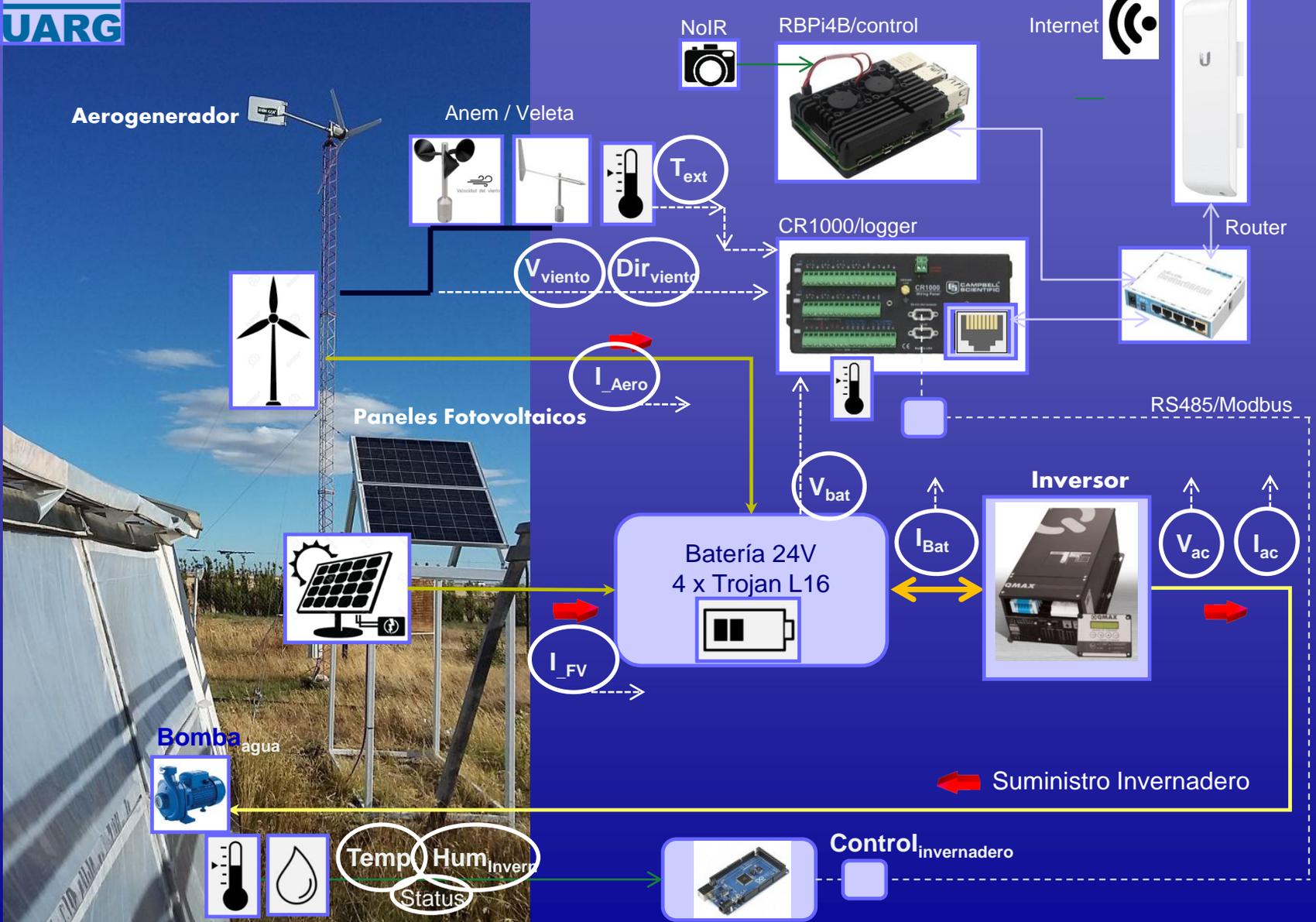
Medidor potencia activa con PIC 16F877 y MC3509





Sistema Eolico/Solar AEA/UNPA-UARG

Suministro a Invernadero



Gracias por su atención! →
Continua Ing. Nestor Cortez

<https://www.energiasalternativas-unpa.net/>

<http://ita.uargadmin.uarg.unpa.edu.ar/ita/index.php/investigacion-2/grupos-investigacion?showall=&start=1>