ESP32 implementado con Wifi y Web Server

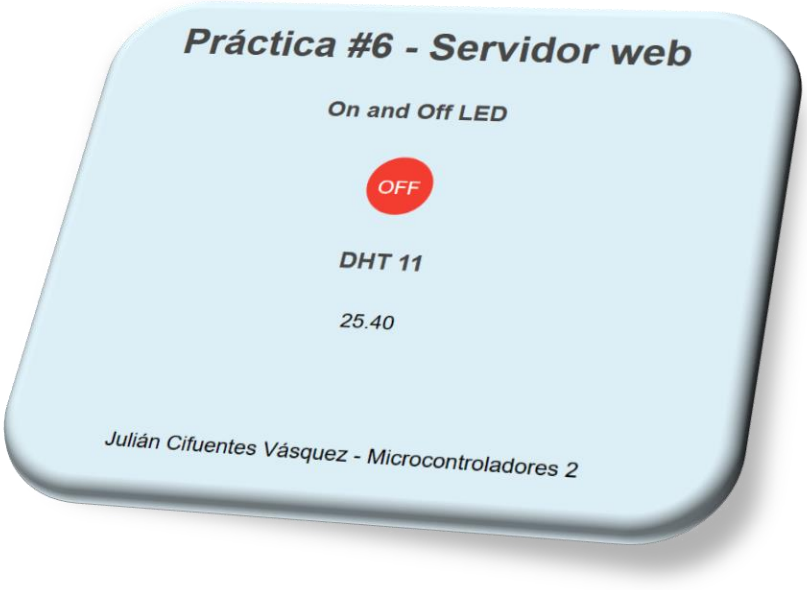
Julián Cifuentes Vásquez, julian.cifuentesv@comunidad.iush.edu.co

Medellín, 17 de mayo de 2024

IMPLEMENTACIÓN:
implementación de un servidor Web usando el ESP32 y FreeRTOS que contiene:
- 1 tarea para el manejo de las peticiones que recibe el servidor (handleServer)
- Una página web que tenga: i) Un botón de encendido y apago para el control de encendido y apagado de un led. ii) La visualización del valor de temperatura más reciente reportado.
- 1 Interrupción de software de Timer que cada 5 seg mande al servidor web e valor de temperatura censado.

Enlaces:
- ➔ Video
- ➔ Arduino

Interfaz:



Código:

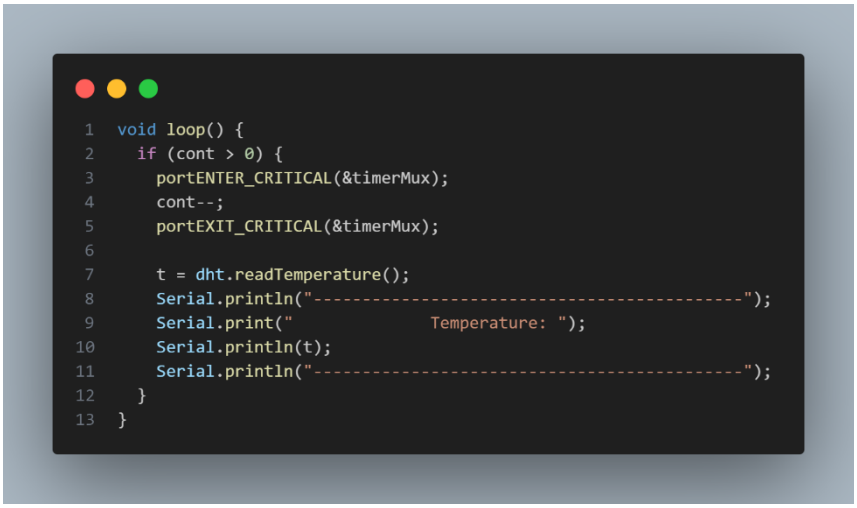
```
#include <WiFi.h>
#include <WebServer.h>
#include "DHT.h"

#define ssid "UNE_HFC_7010(2)"
#define password "AFAFF929"

#define ledPin 23
#define dhtPin 22
#define DHTTYPE DHT11

WebServer server(80);

DHT dht(dhtPin, DHTTYPE);

volatile float cont;
bool ledState = false;
float t = 0;
```

```
hw_timer_t* timer = NULL;  // Apuntador
portMUX_TYPE timerMux = portMUX_INITIALIZER_UNLOCKED;

// Maneja el servidor web
void handleServer(void* param) {
  for (;;) {
    server.handleClient();                // Maneja las solicitudes del cliente
    vTaskDelay(10 / portTICK_PERIOD_MS);  // Pequeña espera
  }
}
```

```
/Pagina web para servir al cliente
String getHTML(bool ledState, float t) {
  String html = "<!DOCTYPE html><html>";

  // <meta> viewport . Para que la pagina se vea correctamente en cualquier dispositivo
  html += "<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0, user-scalable=no\">\n";
  html += "<META HTTP-EQUIV='Refresh' CONTENT='5'>";
  html += "<meta charset=\"UTF-8\">";
  html += "<title>Control de LED </title></head>";

  html += "<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align: center;}\n";
  html += "body{margin-top: 50px;background-color: #D5E6ED;} h1 {color: #444444;margin: 50px auto 30px;} h3 {color: #444444;margin: 30px auto 30px;}\n";
  html += ".button {display: block;width: 80px;background color: #3498db;border: none;color: white;padding: 13px 30px;text decoration: none;font size: 25px;margin: 0px auto 35px;cursor: pointer;border radius: 4px;}\n";
  html += ".round-button {width: 40px;height: 40px;background-color: #4CAF50;border: none;color: white;text-align: center;text-decoration: none;display: inline-block;font-size: 16px;line-height: 40px;border-radius: 50%;cursor: pointer;transition: background-color 0.3s;}\n";
  html += ".round-button2 {width: 40px;height: 40px;background-color: #EB302F;border: none;color: white;text-align: center;text-decoration: none;display: inline-block;font-size: 16px;line-height: 40px;border-radius: 50%;cursor: pointer;transition: background-color 0.3s;}\n";
  html += "</style>\n";

  html += "<body>";
  html += "<br>";

  html += "<h1>Práctica #6 - Servidor web</h1>";

  html += "<h3>On and Off LED</h3>";

  html += "<form action =\"/\" method=\"POST\">";
  if (ledState) {
    html += "<button class = \"round-button\" type=\"submit\">ON</button>";
  } else {
    html += "<button class = \"round-button2\" type=\"submit\">OFF</button>";
  }
  html += "</form>";

  html += "<h3>DHT 11</h3>";
  html += "<p>";
  html += String(t);
  html += "</p>";

  html += "<br><br><br>";

  html += "<p>Julián Cifuentes Vasquez - Microcontroladores 2 </p>";

  html += "</body>";

  html += "</html>";
  return html;
}
```

```
void handleToggle() {
  ledState = !ledState;
  digitalWrite(ledPin, ledState);
  server.send(200, "text/html", getHTML(ledState, t));
}


void IRAM_ATTR getTemperature() {
  portENTER_CRITICAL_ISR(&timerMux);
  cont++;
  portEXIT_CRITICAL_ISR(&timerMux);
  //  server.send(200, "text/html", getHTML(ledState, t));
}
```

```
// Configuración de la conexión Wifi y el servidor web
void setup() {
  Serial.begin(115200);

  pinMode(ledPin, OUTPUT);
  dht.begin();


  // Conectar al wifi
  Serial.println("Conectando al WiFi...");
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("\nConectado al WiFi");
  Serial.print("IP: ");
  Serial.println(WiFi.localIP());

  // Configurar el servidor
  server.on("/", HTTP_GET, []() {
    server.send(200, "text/html", getHTML(ledState, t));  // Se envía el get
  });

  server.on("/", HTTP_POST, handleToggle);
  //server.on("/", HTTP_GET, getTemperature);

  server.begin();  // Inicia el servidor

  // Crea la tarea para manejar el servidor
  xTaskCreatePinnedToCore(
    handleServer,
    "ServerTask",
    10000,
    NULL,
    1,
    NULL,
    1);

  timer = timerBegin(0, 80, true);                    // timer 0, prescalar: 80, UP counting
  timerAttachInterrupt(timer, &getTemperature, true);  // Attach interrupt
  timerAlarmWrite(timer, 5000000, true);              // 1000000 for 1 sec. delay.
  timerAlarmEnable(timer);
}
```



```
void loop() {
  if (cont > 0) {
    portENTER_CRITICAL(&timerMux);
    cont--;
    portEXIT_CRITICAL(&timerMux);

    t = dht.readTemperature();
    Serial.println("-----------------------------------------");
    Serial.print("              Temperature: ");
    Serial.println(t);
    Serial.println("-----------------------------------------");
  }
}
```