Universidad de Guadalajara

Centro Universitario de Ciencias Exactas e Inegenierías

Título

Nombres:
Violeta Acosta Mora
Mariana Isabel Cayeros Parra
Jose Fernando Delgadillo Nolasco

Carrera:
Ingeniería Fotónica

Actividad:
Título

31 de agosto de 2024

Índice

1. Objetivo General

Objetivo General

Desarrollar un sistema que permita el control de dos LEDs mediante una interfaz.

2. Descripción General

Descripción General

Este sistema permite controlar un par de LEDs mediante una interfaz gráfica a través de conexión Wi-Fi.

3. Requerimientos del Sistema

Requerimientos del Sistema

1. El sistema debe de contar con 2 LEDs de salida.
2. Cada LED estará controlado por una parte de la interfaz.
3. Las partes de la interfaz no deben interferir entre sí.

4. Funcionamiento

La conexión del circuito consta de cables simples entre los pines 22 y 23 del microcontrolador ESP32 y los LEDs, que están protegidos por una resistencia a tierra cada uno. La selección de los pines se debe a la cercanía entre ellos. El código utilizado en la tarjeta le indica funcionar comom un punto de acceso Wi-Fi y un servidor web básico. Al iniciar, la ESP32 se configura para crear su propia red Wi-Fi con el nombre .^{Eq}uipo 5ƨ la contraseña .^{eq}uipocinco". Esto permite que otros dispositivos se conecten a él sin utilizar una red Wi-Fi externa. Primero, se apagan los LED para garantizar que comiencen a funcionar en el estado apagado. Inicia la comunicación en serie para la depuración e inicia el punto de acceso Wi-Fi. La dirección IP del punto de acceso se muestra en el monitor serial y el servidor HTTP se inicia en el puerto 80. Cuando el cliente se conecta, el servidor lee los datos recibidos y los muestra en el monitor serial. Cuando el servidor detecta una solicitud HTTP, envía una respuesta con formato HTML que contiene enlaces para controlar los LEDs. Si el cliente hace clic en el enlace para encender el LED1, el servidor recibirá la solicitud y encenderá el primer LED. Si elige desactivar el enlace al LED1, el servidor apagará el primer LED. También hay enlaces para encender y apagar el segundo LED respectivamente. Una vez procesada la solicitud y enviada la respuesta

al cliente, la conexión se cierra y el servidor está listo para aceptar nuevas conexiones. El monitor serial muestra mensajes sobre el estado de la conexión del cliente.

5. Diagrama Esquemático

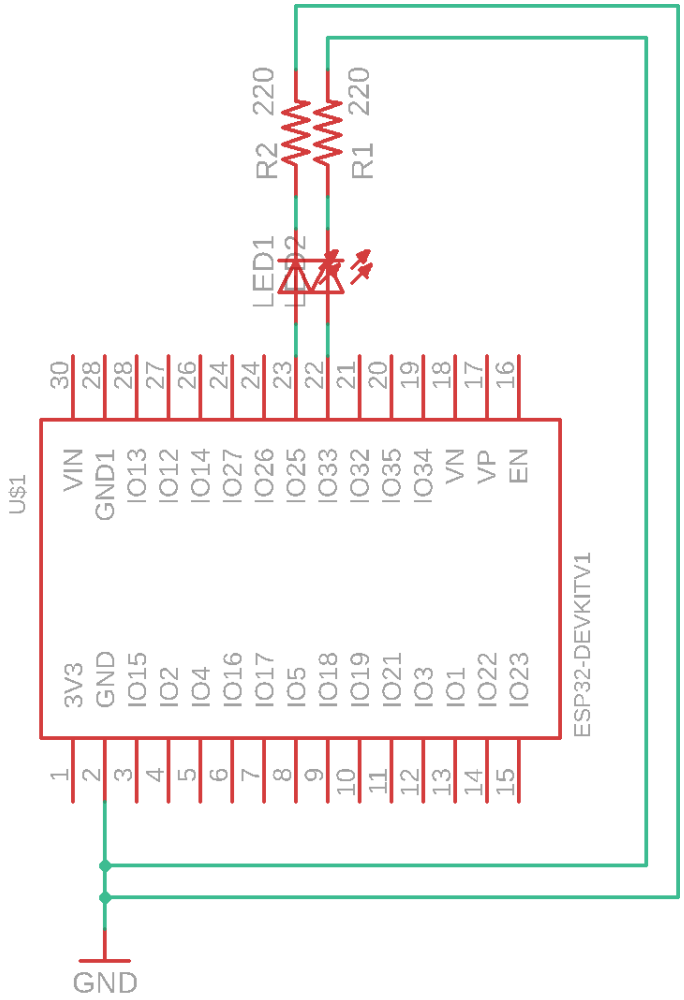

Figura 1: Diagrama esquemático del sistema.

6. Descripción de Puertos

El sistema utiliza los siguientes pines para la conexión:

- LED1: Pin D22
- LED2: Pin D23

7. Diagrama de Flujo

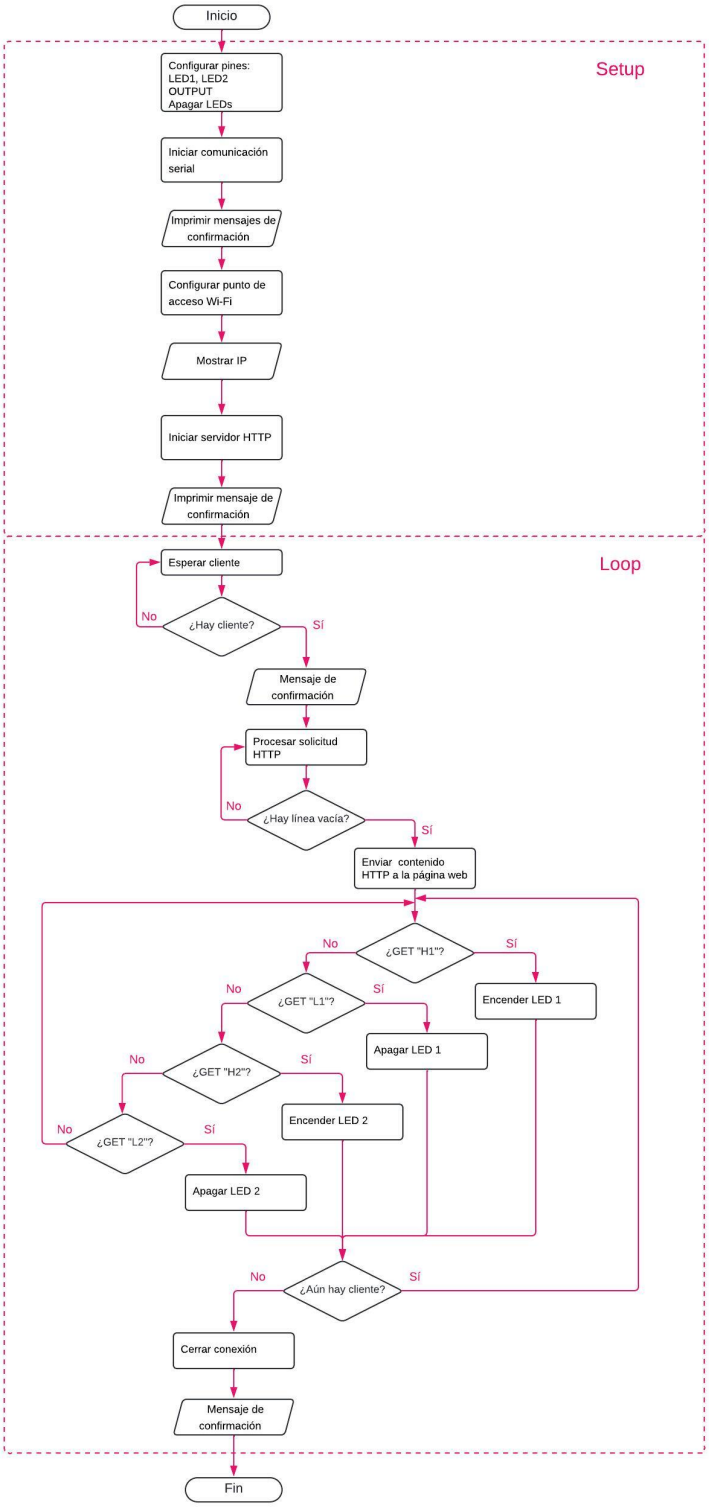

Figura 2: Diagrama de flujo del sistema.

7 DIAGRAMA DE FLUJO

```
// DECLARACIONES
#include <WiFi.h>         // Incluye la librer a para manejar el WiFi en la ESP32
#include <WiFiClient.h>   // Incluye la librer a para manejar los clientes WiFi
#include <WiFiAP.h>       // Incluye la librer a para manejar el punto de acceso WiFi

// DEFINICIONES
#define LED1 23  // Define el pin 23 como LED1 (cambio: antes solo hab a una definici n de LED para el pin 23)
#define LED2 22  // Define el pin 22 como LED2 (nuevo: se agreg esta l nea para controlar un segundo LED)

// Configuraci n de las credenciales WiFi
const char *ssid = "Equipo 5";          // Nombre de la red WiFi
const char *password = "equipocinco";   // Contrase a de la red WiFi

WiFiServer server(80);  // Inicializa un servidor HTTP en el puerto 80

// CONFIGURACIONES
void setup() {
  pinMode(LED1, OUTPUT);  // Configura el pin LED1 como salida
  pinMode(LED2, OUTPUT);  // Configura el pin LED2 como salida (nuevo: se agreg  esta l nea para el segundo LED)

  // Apagar los LEDs al inicio (nuevo: estas dos l neas se agregaron para asegurar que ambos LEDs inicien apagados)
  digitalWrite(LED1, LOW);  // Apaga LED1
  digitalWrite(LED2, LOW);  // Apaga LED2

  Serial.begin(115200);  // Inicia la comunicaci n serial a 115200 baudios
  Serial.println();      // Imprime una l nea en blanco en el monitor serial
  Serial.println("Configurando punto de acceso...");  // Mensaje para indicar que el punto de acceso se est  configurando

  // Iniciar el punto de acceso WiFi
  WiFi.softAP(ssid, password);  // Configura la ESP32 como punto de acceso usando las credenciales definidas

  IPAddress myIP = WiFi.softAPIP();  // Obtiene la direcci n IP del punto de acceso
  Serial.print("Direccion IP: ");    // Imprime el texto "Direccion IP: " en el monitor serial
  Serial.println(myIP);              // Imprime la direcci n IP obtenida

  server.begin();  // Inicia el servidor HTTP
  Serial.println("Servidor iniciado");  // Mensaje para indicar que el servidor ha iniciado
}
```

```
39
40 // BUCLE PRINCIPAL
41 void loop() {
42   WiFiClient client = server.available();  // Espera a que haya
       clientes entrantes
43
44   if (client) {  // Si se conecta un cliente,
45     Serial.println("Nuevo Cliente.");  // Imprime un mensaje en el
         monitor serial
46     String currentLine = "";            // Crea una cadena vac a
         para guardar los datos recibidos del cliente
47
48     while (client.connected()) {  // Cicla mientras el cliente
         est  conectado
49       if (client.available()) {   // Si hay bytes disponibles para
           leer del cliente,
50         char c = client.read();   // Lee un byte del cliente
51         Serial.write(c);          // Lo imprime en el monitor
             serial
52
53         // Si el byte recibido es una nueva l nea ('\n'), se
             procesa la solicitud
54         if (c == '\n') {
55           // Si la l nea actual est  vac a, es el fin de la
               petici n HTTP
56           if (currentLine.length() == 0) {
57             // Responde con un encabezado HTTP 200 OK
58             client.println("HTTP/1.1 200 OK");
59             client.println("Content-type:text/html");
60             client.println();
61
62             // Envia el contenido HTML de la p gina
63             client.print("Click <a href=\"/H1\">aqui</a> para
                 encender LED1.<br>");  // Enlace para encender LED1
64             client.print("Click <a href=\"/L1\">aqui</a> para
                 apagar LED1.<br>");   // Enlace para apagar LED1
65             client.print("Click <a href=\"/H2\">aqui</a> para
                 encender LED2.<br>");  // Enlace para encender LED2
                 (nuevo: se agreg  esta l nea para controlar LED2)
66             client.print("Click <a href=\"/L2\">aqui</a> para
                 apagar LED2.<br>");   // Enlace para apagar LED2 (
                 nuevo: se agreg  esta l nea para controlar LED2)
67
68             // La respuesta HTTP termina con otra l nea en blanco:
69             client.println();
70             break;  // Sale del bucle una vez que se ha enviado la
                 respuesta
71           } else {
72             // Limpia la l nea actual si se recibe una nueva
                 l nea
73             currentLine = "";
74           }
75         } else if (c != '\r') {
76           // A ade el car cter a la l nea actual si no es un
               retorno de carro
77           currentLine += c;
```

```
78          }
79
80          // Revisar si la petici n del cliente fue "GET /H1", "GET
                /L1", "GET /H2" o "GET /L2":
81          if (currentLine.endsWith("GET /H1")) {
82            digitalWrite(LED1, HIGH);  // GET /H1 enciende LED1
83          }
84          if (currentLine.endsWith("GET /L1")) {
85            digitalWrite(LED1, LOW);  // GET /L1 apaga LED1
86          }
87          if (currentLine.endsWith("GET /H2")) {  // (nuevo: se
                agreg  esta l nea para manejar el encendido de LED2)
88            digitalWrite(LED2, HIGH);  // GET /H2 enciende LED2
89          }
90          if (currentLine.endsWith("GET /L2")) {  // (nuevo: se
                agreg  esta l nea para manejar el apagado de LED2)
91            digitalWrite(LED2, LOW);  // GET /L2 apaga LED2
92          }
93        }
94      }
95      // Cerrar la conexi n:
96      client.stop();  // Termina la conexi n con el cliente
97      Serial.println("Cliente Desconectado.");  // Imprime un mensaje
            en el monitor serial indicando que el cliente se ha
            desconectado
98    }
99  }
```

Listing 1: Código para controlar dos LEDs mediante Wi-Fi

8. Conclusiones

Se cumplieron todos los requerimientos establecidos para el sistema: Los dos LEDs fueron controlados de manera independiente sin interferencia entre las partes de la interfaz gráfica creada y controlada a través de un servidor Wi-Fi del propio microcontrolador ESP32.