María José Gamba Santibañez
Luis Pablo López Iracheta
Diego Mares Rodriguez

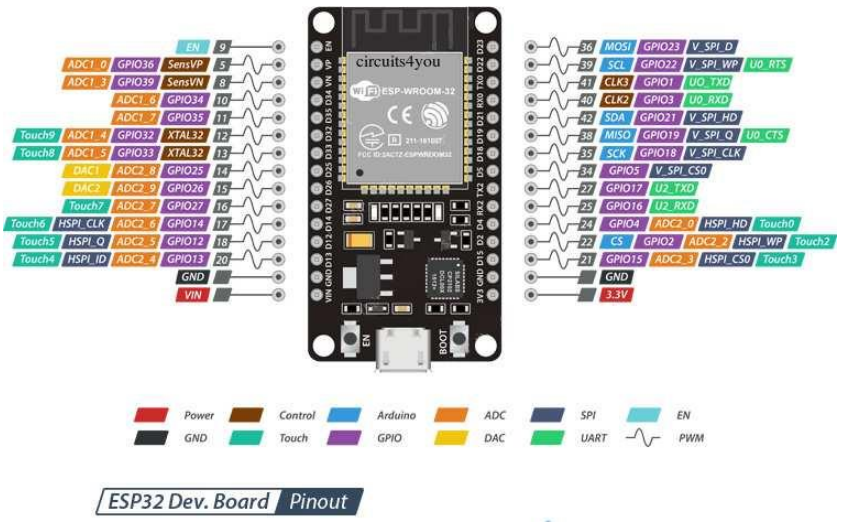
Practica #1

“Conexión wifi ESP32”

12/Septiembre/2024

Internet de las Cosas

Myriam Cristina Jiménez Mares



María José Gamba Santibañez
Luis Pablo López Iracheta
Diego Mares Rodriguez

a) Código con comentarios

Añadir las bibliotecas en el código para habilitar el acceso a internet, configurar los pines para los LEDs y el botón, e ingresar los parámetros de la conexión Wi-Fi, como el nombre de la red y la contraseña.

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

// Parámetros de acceso a red WiFi
const char* ssid = "GalaxyMyriam";
const char* password = "IberoIoT";

// Pines para los LEDs
int pinLed[4] = {13, 12, 14, 27};  // Pines para los LEDs

int contadorPresiones = 0;
int numeroDeLEDs = 4;
bool interrumpido = false;

String html;
WebServer server(80);   // El servidor se conecta por el puerto 80
```

1. Configuración de LEDs, iniciar la conexión a wifi automática, establecer mensajes de progreso para la conexión de internet

```
void setup() {
  Serial.begin (9600);

  // Configurar los LEDs como salidas
  for (int i = 0; i < numeroDeLEDs; i++) {
    pinMode(pinLed[i], OUTPUT);
  }

  // Conexión WiFi
  WiFi.begin(ssid, password);
   while (WiFi.status() != WL_CONNECTED) {  //Espera hasta conectarse a la red WiFi
    delay(1000);
    Serial.println("Conectando al WiFi...");
  }

  datosWiFi();  // Función que configura los datos de red
```

María José Gamba Santibañez
Luis Pablo López Iracheta
Diego Mares Rodriguez

2. Configuración del servidor, iniciar parámetros de animación, crear nuevas rutas para la URL al presionar el botón de “Siguiente Animacion”, mantener conexión entre la computadora y el servidor, iniciar animaciones.

```
  // Configuración de las rutas del servidor
  server.on("/", respuesta);    // Ruta principal
  server.on("/NEXT", siguienteAnimacion);  // Cambia a la siguiente animación
  server.on("/APAGAR", apagarLeds);  // Apagar leds
  server.begin();   // Inicializa el servidor
  Serial.println("Servidor iniciado");
}

void loop() {
  server.handleClient();
  // Ejecutar la animación actual
  ejecutarAnimacion(contadorPresiones % 4);
}

void respuesta() {
  code_html();
  server.send(200, "text/html", html);
}

// Función para cambiar a la siguiente animación
void siguienteAnimacion() {
  code_html();
  server.send(200, "text/html", html);
  contadorPresiones++;  // Cambia a la siguiente animación
}
void apagarLeds () {
  code_html();
  server.send(200, "text/html", html);
  apagarLeds;
}
```

3. Configurar ip’s, del Gateway, para poder establecer la conexión y que se cree la página web privada de nuestra red para poder cambiar de animacion.

```
// Función que configura los datos de red del AP
void datosWiFi() {
  IPAddress local_ip(192, 168, 229, 243);
  IPAddress gateway(192, 168, 229, 1);
  IPAddress subnet(255, 255, 255, 0);
  WiFi.config(local_ip, gateway, subnet);
}
```

María José Gamba Santibañez
Luis Pablo López Iracheta
Diego Mares Rodriguez

4. Modificar el código HTML, de la página para editar la página inicial y predetermina, para cambiar el encabezado de la página, y solo añadir un botón de “Siguiente Animacion” en la pagina.

```
void code_html() {
  // Código HTML para el control web
  html = "<align=left><font color=\"red\">Control de Animaciones</font><br>";
  html += "<hr/>"; // Línea horizontal
  // Crea el botón para cambiar la animación
  html += "<font color='green'>Animaciones: </font> <button
onclick=\"location.href='/NEXT'\">Siguiente Animacion</button><br>";
html += "<font color='red'>Animaciones: </font> <button
onclick=\"location.href='/APAGAR\">Apagar leds</button><br>";
  html += "<hr/>"; // Línea horizontal
  html += "<p> </p>"; // Línea de espacio
  html += "<p> </p>"; // Línea de espacio
  html += "</body></html>";
}
```

5. Crear nuestra función switch, para hacer el cambio de animacion según se vaya presionando el botón siendo cada animacion una función case diferente.

```
// Función para ejecutar la animación correspondiente
void ejecutarAnimacion(int numeroDeAnimacion) {
  switch (numeroDeAnimacion) {
    case 0:
      animacion1();
      break;
    case 1:
      animacion2();
      break;
    case 2:
      animacion3();
      break;
    case 3:
      animacion4();
      break;
  }
}
```

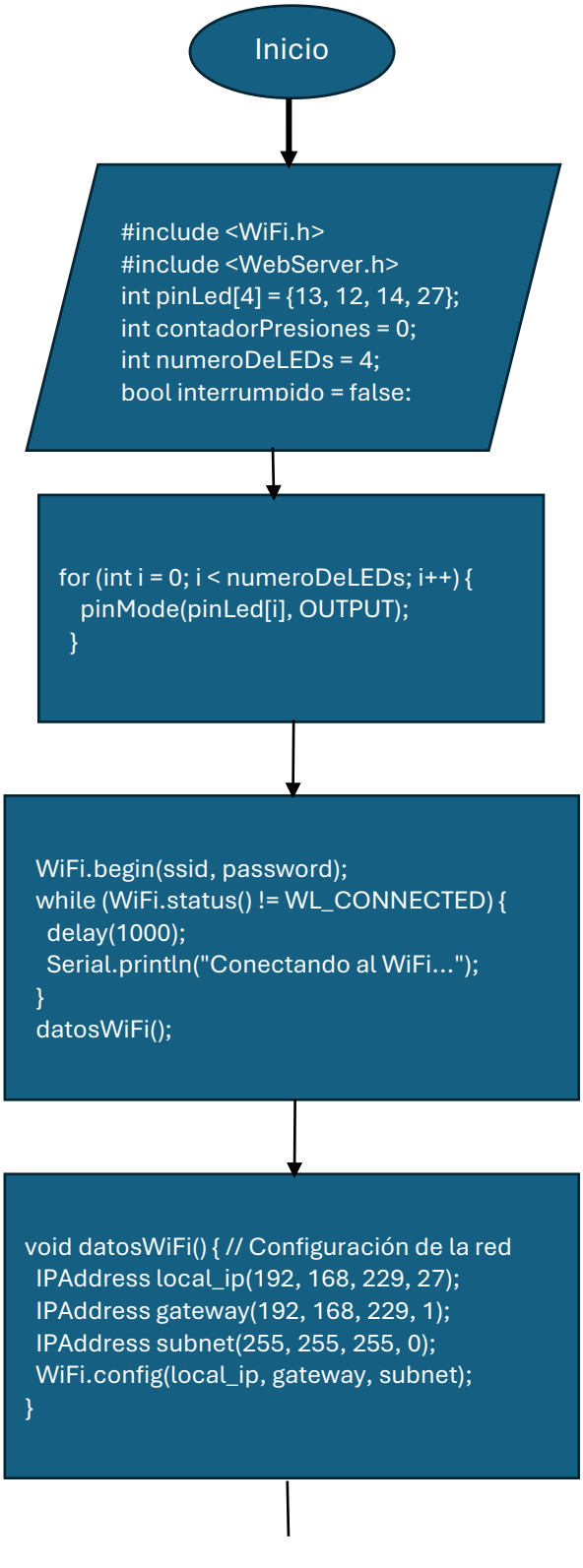
6. Insertar en el código las animaciones creadas en clases anteriores, modificarlo un poco para hacerlo compatible con el código actual, las animaciones de los leds se hacen “jugando” con el delay de prender y apagar los LEDs, para finalizar crear una función específica para apagar los LEDs interrumpiendo las animaciones.

María José Gamba Santibañez
Luis Pablo López Iracheta
Diego Mares Rodriguez

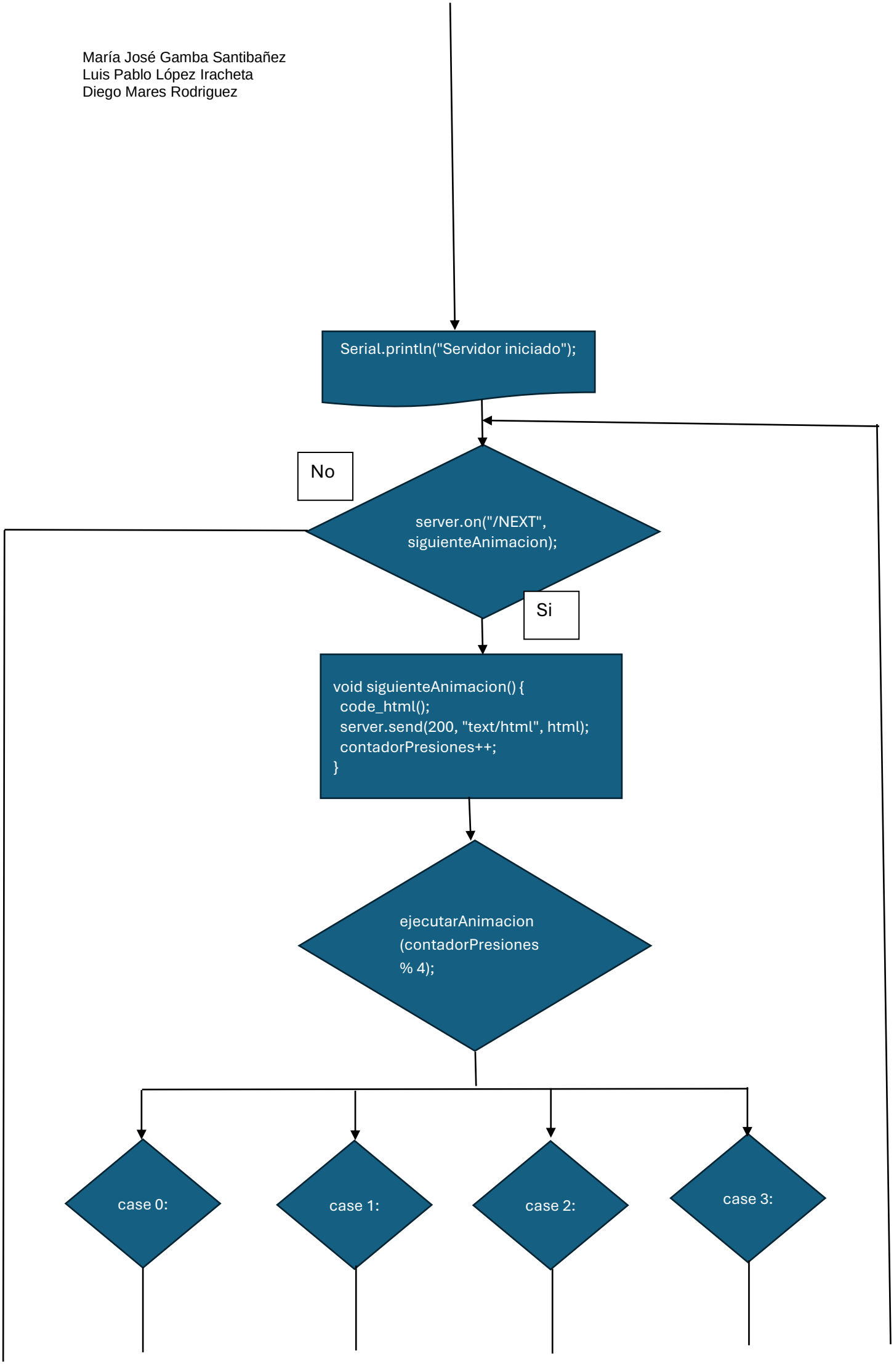
```
void animacion1() {
  // Encender un LED a la vez en secuencia
  for (int i = 0; i < numeroDeLEDs; i++) {
    digitalWrite(pinLed[i], HIGH);
    delay(100);
    digitalWrite(pinLed[i], LOW);
    delay(100);
  }
}

void animacion2() {
  // Encender los LEDs de afuera hacia adentro
  for (int i = 0; i < numeroDeLEDs / 2; i++) {
    digitalWrite(pinLed[i], HIGH);
    digitalWrite(pinLed[numeroDeLEDs - 1 - i], HIGH);
    delay(200);
    digitalWrite(pinLed[i], LOW);
    digitalWrite(pinLed[numeroDeLEDs - 1 - i], LOW);
    delay(200);
  }
}
void animacion3() {
  // Encender todos los LEDs al mismo tiempo
  for (int i = 0; i < numeroDeLEDs; i++) {
    digitalWrite(pinLed[i], HIGH);
  }
  delay(500);
  for (int i = 0; i < numeroDeLEDs; i++) {
    digitalWrite(pinLed[i], LOW);
  }
  delay(500);
}
void animacion4() {
  // Encender LEDs de adentro hacia afuera
  for (int i = numeroDeLEDs / 2; i >= 0; i--) {
    digitalWrite(pinLed[i], HIGH);
    digitalWrite(pinLed[numeroDeLEDs - 1 - i], HIGH);
    delay(200);
    digitalWrite(pinLed[i], LOW);
    digitalWrite(pinLed[numeroDeLEDs - 1 - i], LOW);
    delay(200);
  }
}
void apagarLeds() {
  for (int i = 0; i < numeroDeLEDs; i++) {
    digitalWrite(pinLed[i], LOW);
  }
}
```

María José Gamba Santibañez
Luis Pablo López Iracheta
Diego Mares Rodriguez

b) Diagrama de flujo



María José Gamba Santibañez
Luis Pablo López Iracheta
Diego Mares Rodriguez



María José Gamba Santibañez
Luis Pablo López Iracheta
Diego Mares Rodriguez

```
void animacion1() {
  for (int i = 0; i < numeroDeLEDs; i++) {
    digitalWrite(pinLed[i], HIGH);
    delay(100);
    digitalWrite(pinLed[i], LOW);
    delay(100);
}
```

```
void animacion2() {
for (int i = 0; i < numeroDeLEDs / 2; i++) {

digitalWrite(pinLed[i], HIGH);

digitalWrite(pinLed[numeroDeLEDs - 1 - i], HIGH);
    delay(200);

digitalWrite(pinLed[i], LOW);

digitalWrite(pinLed[numeroDeLEDs - 1 - i], LOW);
    delay(200);
  }
}
```

```
void animacion3() {
  // Encender todos los LEDs al mismo tiempo
  for (int i = 0; i < numeroDeLEDs; i++) {

digitalWrite(pinLed[i], HIGH);
  }
  delay(500);
  for (int i = 0; i < numeroDeLEDs; i++) {

digitalWrite(pinLed[i], LOW);
  }
  delay(500);
}
```

```
void animacion4() {
  // Encender LEDs de adentro hacia afuera
  for (int i = numeroDeLEDs / 2; i >= 0; i--) {
    digitalWrite(pinLed[i], HIGH);

digitalWrite(pinLed[numeroDeLEDs - 1 - i], HIGH);
    delay(200);
    digitalWrite(pinLed[i], LOW);

digitalWrite(pinLed[numeroDeLEDs - 1 - i], LOW);
    delay(200);
  }
}
```

server.on("/APAGAR", apagarLeds);

no

Si

```
void apagarLeds() {
  for (int i = 0; i < numeroDeLEDs; i++) {
    digitalWrite(pinLed[i], LOW);
  }
}
```

María José Gamba Santibañez
Luis Pablo López Iracheta
Diego Mares Rodriguez

c) Comentarios

Los problemas que encontramos se debían principalmente a la conexión de nuestro ESP32 a la red del celular de la maestra, ya que el dispositivo mostraba un estado constante de "Conectando", pero nunca lograba establecer la conexión. Tras revisar detalladamente las direcciones IP en la computadora utilizando el comando `whoami` y ajustarlas en el código para asegurar que coincidieran correctamente, finalmente logramos establecer la conexión. Posteriormente, al ingresar la dirección IP en el navegador, pudimos conectar exitosamente el botón digital.

d) Conclusiones

En esta práctica, logramos conectar exitosamente el ESP32 a una red inalámbrica y controlar un botón digital a través de dicha conexión. Aunque inicialmente enfrentamos dificultades para establecer la conexión debido a problemas con las direcciones IP y la configuración de red, la experiencia nos permitió profundizar en la importancia de una correcta configuración de los parámetros de red y la verificación de comandos en el sistema. Al superar estos desafíos, adquirimos habilidades clave en el manejo de dispositivos IoT, así como en la solución de problemas relacionados con la comunicación de hardware y redes. Este ejercicio refuerza el valor de la precisión y la paciencia en el proceso de desarrollo con plataformas de microcontroladores como el ESP32.