

TECNICATURA SUPERIOR EN

Telecomunicaciones

Electrónica Microcontrolada



Optimización 2: Comunicación y Preprocesamiento

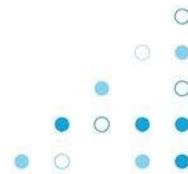
Optimización 2: Comunicación y Preprocesamiento

TP#20. Trabajo Práctico

Objetivos:

- Familiarizarse con las tecnologías de los módulos de comunicación incorporados en el ESP32.
- Conocer las librerías asociadas a cada modulo de comunicación para el ESP32 incluidas en el framework Arduino.

Realizar los siguientes ejercicios según se solicita a continuación, dejar los registros de desarrollo en el repositorio personal de la materia, con la estructura ABP de trabajo habitual.



Optimización 2: Comunicación y Preprocesamiento

Serie de Prácticas Wi-Fi con el ESP32

- **Práctica 1: Conexión Básica a una Red Wi-Fi y Monitorización de Estado**
- **Objetivo**
- Aprender a conectar el ESP32 a una red Wi-Fi en modo Estación (STA), monitorear el estado de la conexión y manejar la reconexión automática en caso de pérdida de la red.
- **Librerías Necesarias**
- **WiFi.h:** Librería estándar de Arduino para manejo de Wi-Fi.



Optimización 2: Comunicación y Preprocesamiento

Pasos a Seguir

1. Configuración del Entorno de Desarrollo

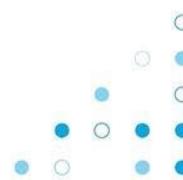
- Configura un nuevo proyecto en PlatformIO para el ESP32 utilizando el framework Arduino.

2. Implementar Conexión Wi-Fi

- Define las credenciales de la red Wi-Fi (SSID y contraseña).
- Implementa la lógica para conectar el ESP32 a la red Wi-Fi.
- Agrega un bucle que monitorice la conexión Wi-Fi.

3. Monitorizar Estado de Conexión

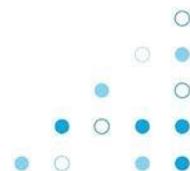
- Imprime el estado de la conexión en el Monitor Serie.
- Implementa la reconexión automática si se pierde la conexión.



Optimización 2: Comunicación y Preprocesamiento

4. Prueba de Funcionamiento

- Prueba el programa con credenciales correctas e incorrectas.
- Deshabilita el Wi-Fi del router para comprobar la reconexión automática.
- **Herramientas Externas**
 - Un router Wi-Fi.
 - Monitor Serie de VSCode.
- **Recomendaciones**
 - Implementa un tiempo de espera razonable para la reconexión para evitar bloqueos.



Optimización 2: Comunicación y Preprocesamiento

Práctica 2: Configurar el ESP32 como Punto de Acceso (AP)

- **Objetivo**
- Configurar el ESP32 como un punto de acceso (AP) para que otros dispositivos se conecten a él.
- **Librerías Necesarias**
- **WiFi.h:** Librería estándar para manejar la conexión Wi-Fi en modo AP.



Optimización 2: Comunicación y Preprocesamiento

Pasos a Seguir

1. Configuración del Modo AP

- Configura el ESP32 para que funcione en modo Punto de Acceso.
- Define el SSID y la contraseña de la red que creará el ESP32.

2. Iniciar el Punto de Acceso

- Implementa la lógica para iniciar el punto de acceso y mostrar la dirección IP asignada en el Monitor Serie.

3. Monitorear Conexiones

- Implementa una función que muestre la cantidad de dispositivos conectados al punto de acceso.



Optimización 2: Comunicación y Preprocesamiento

4. Prueba de Funcionamiento

- Conéctate a la red creada por el ESP32 desde un smartphone o PC.
- Verifica en el Monitor Serie el número de dispositivos conectados.
- **Herramientas Externas**
 - Un smartphone o PC con Wi-Fi.
- **Recomendaciones**
 - Usa una contraseña segura para evitar conexiones no autorizadas.



Optimización 2: Comunicación y Preprocesamiento

Práctica 3: Servidor Web Simple con Wi-Fi

- **Objetivo**
- Crear un servidor web en el ESP32 para controlar un LED desde un navegador web.
- **Librerías Necesarias**
- **WiFi.h:** Para la conexión Wi-Fi.
- **WebServer.h:** Para implementar el servidor web.



Optimización 2: Comunicación y Preprocesamiento

Pasos a Seguir

1. Configurar Conexión Wi-Fi

- Conecta el ESP32 a una red Wi-Fi.

2. Implementar el Servidor Web

- Configura un servidor web que escuche en el puerto 80.
- Crea una página web simple con dos botones para encender y apagar un LED.

3. Controlar el LED

- Define rutas para encender y apagar el LED.
- Implementa la lógica para cambiar el estado del LED según las peticiones recibidas.



Optimización 2: Comunicación y Preprocesamiento

4. Prueba de Funcionamiento

- Accede a la página web alojada por el ESP32 desde un navegador.
- Interactúa con los botones y observa el comportamiento del LED.
- **Herramientas Externas**
 - Un navegador web.
 - Un LED conectado al ESP32 (o usar el LED integrado).
- **Recomendaciones**
 - Mejora la interfaz de la página web para obtener una experiencia más interactiva.



Optimización 2: Comunicación y Preprocesamiento

Práctica 4: Cliente HTTP para Enviar Datos a un Servidor

- **Objetivo**
- Configurar el ESP32 como cliente HTTP para enviar datos a un servidor remoto.
- **Librerías Necesarias**
- **WiFi.h:** Para la conexión Wi-Fi.
- **HTTPClient.h:** Para realizar solicitudes HTTP.



Optimización 2: Comunicación y Preprocesamiento

Pasos a Seguir

1. Configurar Conexión Wi-Fi

- Conecta el ESP32 a una red Wi-Fi.

2. Configurar Cliente HTTP

- Configura el ESP32 para que realice solicitudes HTTP POST a un servidor.
- Envía un conjunto de datos (por ejemplo, valores de sensores) al servidor.

3. Manejar Respuestas del Servidor

- Implementa la lógica para recibir y mostrar la respuesta del servidor en el Monitor Serie.



Optimización 2: Comunicación y Preprocesamiento

4. Prueba de Funcionamiento

- Configura un servidor web o usa una API pública para recibir los datos.
- Observa la respuesta del servidor y verifica que los datos se envíen correctamente.
- **Herramientas Externas**
 - Un servidor web o servicio API REST.
 - Alternativamente, usa herramientas como httpbin.org para pruebas
- **Recomendaciones**
 - Asegúrate de manejar los errores de conexión y respuesta del servidor.
 - Considera el uso de HTTPS para mayor seguridad si el servidor lo soporta.



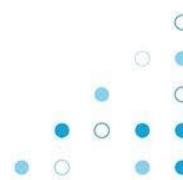
Optimización 2: Comunicación y Preprocesamiento

Serie de Prácticas Bluetooth con el ESP32

Práctica 1: Comunicación Serial con Bluetooth Clásico (SPP)

- **Objetivo**
- Configurar el ESP32 para que funcione como un dispositivo Bluetooth Serial (SPP).
- Permitir la comunicación bidireccional entre el ESP32 y un dispositivo externo (como un smartphone o PC) usando Bluetooth Clásico.

- **Librerías Necesarias**
- **BluetoothSerial.h** Librería de Arduino para manejar la comunicación Bluetooth Clásica en el ESP32.



Optimización 2: Comunicación y Preprocesamiento

Pasos a Seguir

1. Configurar Bluetooth Clásico

- Configura el ESP32 para que funcione en modo Bluetooth Clásico con el perfil de puerto serie (SPP).
- Define un nombre de dispositivo Bluetooth para el ESP32.

2. Implementar Comunicación Serial

- Configura el ESP32 para que pueda enviar y recibir datos a través del puerto serie Bluetooth.
- Envía mensajes recibidos por Bluetooth al Monitor Serie y viceversa.

3. Emparejamiento

- Haz que el ESP32 sea detectable para otros dispositivos Bluetooth para permitir el emparejamiento.



Optimización 2: Comunicación y Preprocesamiento

4. Prueba de Funcionamiento

- Desde un smartphone o PC, empareja el dispositivo con el ESP32.
 - Utiliza una aplicación de terminal Bluetooth (como "Serial Bluetooth Terminal" en Android) para enviar y recibir datos.
 - Observa el intercambio de datos en tiempo real.
-
- **Herramientas Externas**
 - Una aplicación de terminal Bluetooth en un smartphone o PC.
 - Monitor Serie de VSCode para observar los datos recibidos.
 - **Recomendaciones**
 - Asegúrate de manejar correctamente la desconexión y reconexión de dispositivos Bluetooth.



Optimización 2: Comunicación y Preprocesamiento

Práctica 2: Comunicación BLE Básica Anunciado y Conexión

- **Objetivo**
 - Configurar el ESP32 como un dispositivo Bluetooth Low Energy (BLE).
 - Implementar el anunciado de un dispositivo BLE y permitir que otros dispositivos se conecten a él.
- **Librerías Necesarias**
 - **BLEDevice.h, BLEServer.h, BLEUtils.h** Librerías de Arduino para manejar BLE en el ESP32.



Optimización 2: Comunicación y Preprocesamiento

Pasos a Seguir

1. Configurar BLE

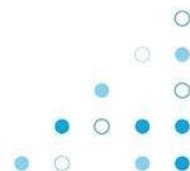
- Configura el ESP32 como un periférico BLE.
- Define un nombre de dispositivo BLE para el ESP32.

2. Anunciado

- Implementa el anunciado para que el ESP32 sea visible para otros dispositivos BLE cercanos.

3. Prueba de Conexión

- Permite que un dispositivo externo, como un smartphone, se conecte al ESP32.



Optimización 2: Comunicación y Preprocesamiento

4. Prueba de Funcionamiento

- Utiliza una aplicación BLE en un smartphone (como "nRF Connect") para escanear dispositivos BLE y encontrar el ESP32.
 - Conéctate al ESP32 y observa el estado de la conexión.
-
- **Herramientas Externas**
 - Aplicación BLE en un smartphone, como "nRF Connect" o "BLE Scanner".
-
- **Recomendaciones**
 - Usa un identificador único (UUID) para el dispositivo BLE para identificarlo claramente entre otros dispositivos.



Optimización 2: Comunicación y Preprocesamiento

Práctica 3: Comunicación BLE - Servicios y Características

- **Objetivo**
 - Configurar el ESP32 para que actúe como un servidor GATT BLE.
 - Definir servicios y características para permitir la lectura y escritura de datos desde un dispositivo externo.
- **Librerías Necesarias**
 - **BLEDevice.h, BLEServer.h, BLEUtils.h, BLECharacteristic.h**: Librerías de Arduino para manejar BLE GATT.



Optimización 2: Comunicación y Preprocesamiento

Pasos a Seguir

1. Configurar el Servidor GATT

- Configura el ESP32 para que actúe como un servidor GATT.

2. Definir Servicios y Características

- Crea un servicio BLE y define al menos una característica que permita la lectura y la escritura de datos.

3. Implementar Funciones de Lectura y Escritura

- Implementa funciones de callback para manejar las operaciones de lectura y escritura en la característica definida.



Optimización 2: Comunicación y Preprocesamiento

4. Prueba de Funcionamiento

- Utiliza una aplicación BLE en un smartphone para escanear el ESP32.
 - Conéctate al dispositivo, encuentra el servicio y la característica, y realiza operaciones de lectura y escritura.
-
- **Herramientas Externas**
 - Aplicación BLE en un smartphone, como "nRF Connect" o "BLE Scanner".
-
- **Recomendaciones**
 - Usa UUIDs personalizados para los servicios y características.
 - Implementa la lógica para manejar correctamente las solicitudes de lectura y escritura.



Optimización 2: Comunicación y Preprocesamiento

Práctica 4: Notificaciones BLE

- **Objetivo**
- Configurar el ESP32 para enviar notificaciones BLE a dispositivos conectados cuando cambie el valor de una característica.
- **Librerías Necesarias**
- **BLEDevice.h, BLEServer.h, BLEUtils.h, BLECharacteristic.h**: Librerías de Arduino para manejar BLE y notificaciones.



Optimización 2: Comunicación y Preprocesamiento

- **Pasos a Seguir**

- 1. Configurar el Servidor GATT y Características**

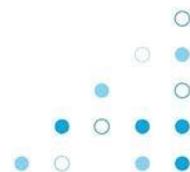
- Configura el ESP32 como un servidor GATT.
- Crea una característica que permita el envío de notificaciones.

- 2. Implementar Notificaciones**

- Implementa la lógica para actualizar el valor de la característica y enviar una notificación a los dispositivos conectados cuando el valor cambie.

- 3. Prueba de Funcionamiento**

- Utiliza una aplicación BLE en un smartphone para conectarte al ESP32.
- Activa las notificaciones en la característica y observa cómo se reciben automáticamente las actualizaciones de valor.



Optimización 2: Comunicación y Preprocesamiento

- **Herramientas Externas**
- Aplicación BLE en un smartphone, como "nRF Connect".
- **Recomendaciones**
- Asegúrate de enviar las notificaciones en intervalos razonables para evitar el agotamiento de la batería.



Optimización 2: Comunicación y Preprocesamiento

Práctica 5: Comunicación Bidireccional BLE - Control Remoto

- **Objetivo**
- Configurar el ESP32 como un servidor BLE que permita recibir comandos para controlar un LED y enviar el estado del LED de vuelta al cliente.
- **Librerías Necesarias**
- **BLEDevice.h, BLEServer.h, BLEUtils.h, BLECharacteristic.h**: Librerías de Arduino para manejar BLE.



Optimización 2: Comunicación y Preprocesamiento

Pasos a Seguir

1. Configurar el Servidor BLE

- Configura el ESP32 como un servidor BLE con dos características: una para recibir comandos de control (escritura) y otra para enviar el estado del LED (lectura).

2. Implementar Control del LED

- Implementa la lógica para cambiar el estado del LED según los comandos recibidos.

3. Implementar Lectura del Estado del LED

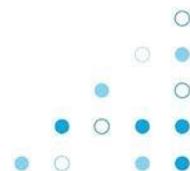
- Implementa la lógica para leer el estado del LED y enviar este estado a los dispositivos conectados.



Optimización 2: Comunicación y Preprocesamiento

4. Prueba de Funcionamiento

- Utiliza una aplicación BLE en un smartphone para enviar comandos y recibir el estado del LED.
 - Observa cómo el LED cambia de estado según los comandos enviados.
-
- **Herramientas Externas**
 - Aplicación BLE en un smartphone, como "nRF Connect".
 - **Recomendaciones**
 - Implementa mecanismos de seguridad, como la autenticación, si se usa en un entorno real.



Optimización 2: Comunicación y Preprocesamiento





¡Muchas gracias!