**ESP32 Communication Guide: Wi-Fi and Bluetooth Testing**

**Learning Objective:** Upon completing this guide, you will be able to successfully program the ESP32 to establish and test both Wi-Fi and Bluetooth Low Energy (BLE) communication links using the Arduino framework.

This guide provides the steps required to set up and test the fundamental Wi-Fi and Bluetooth (BLE) capabilities of your ESP32 board using the Arduino IDE.

**1. Prerequisites and Setup**

Before starting, ensure you have the following in place:

1. **Arduino IDE:** Installed and updated.
2. **ESP32 Board Package:** Installed in the Arduino IDE (File > Preferences > Board Manager URLs, then Tools > Board > Board Manager).
3. **USB Cable:** To connect your ESP32 to your computer.

**2. Wi-Fi Connectivity Test (Station Mode)**

In **Station mode**, the ESP32 connects to an existing Wi-Fi router (like your phone or home router).

**Code: ESP32\_WiFi**

This example attempts to connect to a Wi-Fi network and then starts a small web server.

**Testing Procedure**

1. **Update Credentials:** Open the ESP32\_WiFi file and update the ssid and password variables to match your local Wi-Fi network.
2. **Upload:** Select your board and port, then upload the code to the ESP32.
3. **Monitor Serial Output:** Open the **Serial Monitor** (set baud rate to 115200).
4. **Find IP:** Wait for the ESP32 to connect. It will print its assigned local IP address (e.g., 192.168.1.100).
5. **Access Web Server:** Open a web browser on a device connected to the *same* network (phone or PC).
6. **Verify:** Type the printed IP address into the browser's address bar (e.g., http://192.168.1.100/). You should see a simple webpage confirming the connection and a link to toggle the built-in LED (if available). This confirms both Wi-Fi connectivity and basic two-way network communication.

**3. Bluetooth Low Energy (BLE) Test (Server Role)**

We will use BLE as it is more power-efficient and the preferred modern standard. The ESP32 will act as a BLE **Server**, exposing a service and a characteristic that a phone (the Client) can interact with.

**Code: ESP32\_BT**

This example creates a BLE server named ESP32 BLE Demo with a custom characteristic for reading/writing simple text data.

**Testing Procedure**

1. **Install App:** Install a BLE scanning/testing app on your smartphone (e.g., **nRF Connect** or **BLE Scanner**).
2. **Upload Code:** Upload the ESP32\_BT code to your ESP32.
3. **Monitor Serial Output:** Open the Serial Monitor (115200 baud). It will print status messages.
4. **Scan on Phone:** Open the BLE app on your phone and start scanning.
5. **Connect:** You should see a device named ESP32 BLE Demo. Connect to it.
6. **Interact with Characteristic:**
   * Once connected, navigate to the services list. You should see a custom service (UUID 4fafc201-1fb5-459e-8fcc-c5c9c331914b).
   * Inside the service, you'll find the characteristic (UUID beb5483e-36e1-4688-b7f5-ea07361b26a8).
   * **Read Test:** Use the "Read" function in the app. The characteristic should return the string "Hello from ESP32!".
   * **Write Test:** Use the "Write" function to send a short text string (e.g., "Test Complete"). The ESP32's Serial Monitor should display the data received.

This confirms your ESP32's BLE advertising, connection handling, and GATT profile implementation are functional.