

ESP32 Bluetooth Networking

User Guide



Version 1.2
Espressif Systems
Copyright © 2019

About This Guide

This document provides a guide with examples to using ESP32 to establish Bluetooth® networks for IoT devices.

Release Note

Date	Version	Release notes
2017.03	V1.0	First release.
2019.05	V1.1	Updated figures.
2019.11	V1.2	Updated the download link of EspBlufi.

Documentation Change Notification

Espressif provides email notifications to keep customers updated on changes to technical documentation. Please subscribe at <https://www.espressif.com/en/subscribe>.

Certification

Download certificates for Espressif products from <https://www.espressif.com/en/certificates>.

Content

1. Introduction.....	1
1.1. Overview	1
1.2. EspBlufi	1
2. APIs for Networking Development	3
2.1. BluFi Protocol.....	3
2.2. APIs on ESP32 End.....	3
2.3. APIs on the EspBlufi APK End	4
3. ESP32 Bluetooth Networking Examples	5
3.1. Hardware and Software Preparation	5
3.2. Setting ESP32 to Station Mode	5
3.3. Setting ESP32 as a SoftAP	9



1. Introduction

1.1. Overview


The ESP32, as a single 2.4 GHz Wi-Fi and Bluetooth combo chip, supports Wi-Fi setups via both SmartConfig and Bluetooth. Users can use ESP32 for secure configuration of Wi-Fi networking for IoT devices.

Using Bluetooth for configuring Wi-Fi network presents the following advantages:

- The Bluetooth protocol is open and scalable.
- By using Bluetooth protocol, users can easily discover nearby devices via Bluetooth beacons.
- The Bluetooth protocol is secure, because the authentication of the device is done over a secure Bluetooth connection before the password is sent to the device.
- Users can also transmit data over Bluetooth to a smartphone, even when the router is not working. The phone can then upload the data to the Internet.
- The phone can now also connect to the Bluetooth device and send commands directly to control the device if the Wi-Fi network is down.

1.2. EspBlufi

ESP32 is compliant with Bluetooth v4.2 BR/EDR and BLE specifications. Espressif has developed the app EspBlufi specifically for Bluetooth networking of IoT devices. The app is supported on Android 4.3 and above. For smartphones of Android 6.0 or above, because of Google's modification to the Android API, users must give their permission for access to their current location and enable the location information module to start the Bluetooth scanning.

- EspBlufi can be downloaded via: <https://github.com/EspressifApp/EspBlufiForAndroid/releases>
- Open EspBlufi after downloading it. In the user interface as shown in Figure 1-1, click on the  icon in the upper right corner.

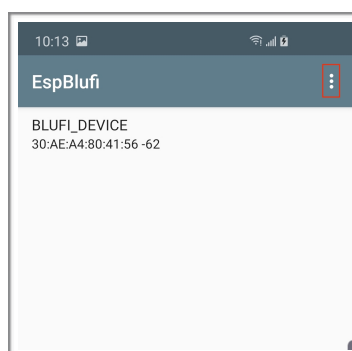




Figure 1-1. EspBlufi User Interface

- The **Configure** button will be shown in the interface and click this button.



Figure 1-2. CONFIGURE Button

- Configure the length of BLE mtu, BLE device filtering, and check the version of the app and Blufi repository.



Figure 1-3. CONFIGURE Interface



2. APIs for Networking Development

2.1. BluFi Protocol

Protocol documentation: <https://docs.espressif.com/projects/esp-idf/en/latest/api-guides/blufi.html>

2.2. APIs on ESP32 End

In the program of ESP32, security-related processes such as key establishment are determined and developed by the users. The app sends the “negotiate data” to ESP32 and the data packets will be sent to the application layer. If the data is not handled by the application layer, DH algorithm provided by BluFi can be used to exchange the key. The application layer needs to register relevant callback functions to BluFi. Table 2-1 provides description of those functions.

Table 2-1. Security-Related Functions

Function	Description
<pre>typedef void (*esp_blufi_negotiate_data_handler_t) (uint8_t *data, int len, uint8_t **output_data, int *output_len, bool *need_free);</pre>	<ul style="list-style-type: none"> This function is used to receive the “negotiate data” during key exchange. The data to be sent should be output by passing the parameters <code>output_data</code> and <code>output_len</code> to BluFi. After BluFi calls <code>negotiate_data_handler</code>, <code>output_data</code> output by <code>negotiate_data_handler</code> is sent. Since the length of data to be sent is not definite, <code>**output_data</code> has to <code>malloc</code> a part of memory or use global variables to release memory via <code>need_free</code>.
<pre>typedef int (* esp_blufi_encrypt_func_t)(uint8_t iv8, uint8_t *crypt_data, int cyprt_len);</pre>	The length of the encoded and decoded data must be the same. <code>iv8</code> in this function means the 8-bit sequence of the packets. It can be used for eight bits of the initialization vector.
<pre>typedef int (* esp_blufi_decrypt_func_t)(uint8_t iv8, uint8_t *crypt_data, int crypt_len);</pre>	The length of the encoded and decoded data must be the same. <code>iv8</code> in this function means the 8-bit sequence of the packets. It can be used for eight bits of the initialization vector.
<pre>typedef uint16_t (*esp_blufi_checksum_func_t)(uint8_t iv8, uint8_t *data, int len);</pre>	The function is used to calculate the checksum. Calling this function returns the value of checksum. The returned value will be compared with the checksum at the end of the packet by BluFi.



2.3. APIs on the EspBlufi APK End

- Source code of EspBlufi for Android: <https://github.com/EspressifApp/EspBlufiForAndroid>
- Documentation of EspBlufi for Android: [https://github.com/EspressifApp/EspBlufiForAndroid/blob/master/doc/Introduction to the EspBlufi API Interface for Android_cn.md](https://github.com/EspressifApp/EspBlufiForAndroid/blob/master/doc/Introduction%20to%20the%20EspBlufi%20API%20Interface%20for%20Android_cn.md)
- Source code of EspBlufi for iOS: <https://github.com/EspressifApp/EspBlufiForiOS>
- Documentation of EspBlufi for iOS: https://github.com/EspressifApp/EspBlufiForiOS/blob/master/doc/Blufi_APP_API_for_iOS_CN.md



3. ESP32 Bluetooth Networking Examples

3.1. Hardware and Software Preparation

- 1 × ESP32 module
- 1 × PC, connected to the module to supply power and print log for ESP32.
- 1 × smartphone (Android 4.3 or above)
- Download and install EspBlufi on the smartphone. Turn on the Wi-Fi and Bluetooth capabilities on the phone. The link for downloading EspBlufi is <https://github.com/EspressifApp/EspBlufiForAndroid/releases>.

3.2. Setting ESP32 to Station Mode

1. Power on the module. The following log will be output to the serial port tool:

```
E (3155) BT: Startup BTU
E (3165) BT: GATTS_CreateService: handle of service handle1
E (3175) BT: bta_dm_co_ble_load_local_keys: func not ported
I (3185) BLUFI_DEMO: BD ADDR: 24:0a:c4:01:4d:be
I (3185) BLUFI_DEMO: BLUFI VERSION 01.00
I (3185) BLUFI_DEMO: BLUFI init finish
```

2. Open the EspBlufi app, and refresh the interface by swiping down from the top of the screen. The nearby Bluetooth-enabled devices will be shown on the screen.

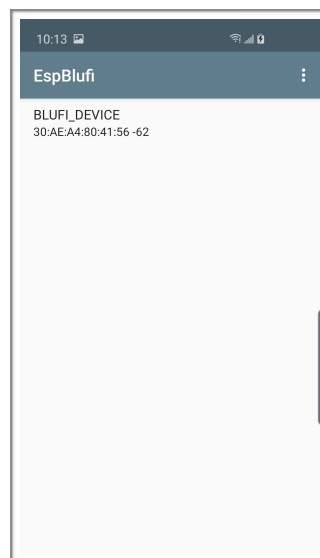


Figure 3-1. EspBlufi Interface



- Click on the ESP32 device in the interface as Figure 3-1 shows, and click **Connect** button to establish Bluetooth connection to it. Successful connection will yield the interface shown in Figure 3-2.

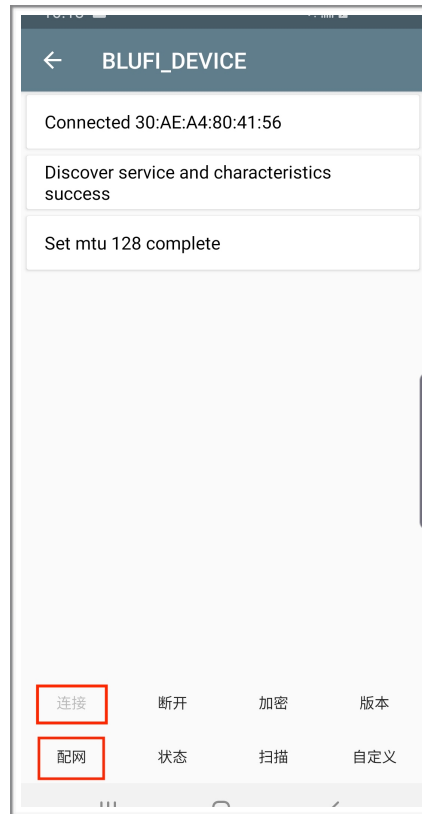


Figure 3-2. Interface Showing Successful Connection

Meanwhile, the following log will be output to the serial port tool:

```
E (37875) BT: btm_ble_resolve_random_addr_on_conn_cmpl unable to match and resolve random address
E (38145) BT:
device is connected 48:63:0f:47:72:86, server_if=4,reason=0x0,connect_id=4
E (38145) BT: smp_br_connect_callback is called on unexpected transport 2
I (38145) BLUFI_DEMO: BLUFI ble connect
E (38285) BT: MTU request PDU with MTU size 64
E (38285) BT: BTM_SetBleDataLength failed, peer does not support request
E (38285) BT: Call back not found for application conn_id=3
I (40185) BLUFI_DEMO: BLUFI get wifi status from AP
```

Note:

If the **Networking** button is not shown or can not be clicked in the interface of Figure 3-2, it means that Bluetooth connection has failed.

- Click on the **Networking** button to configure the network, get the dropdown menu for network configuration, as Figure 3-3 shows:

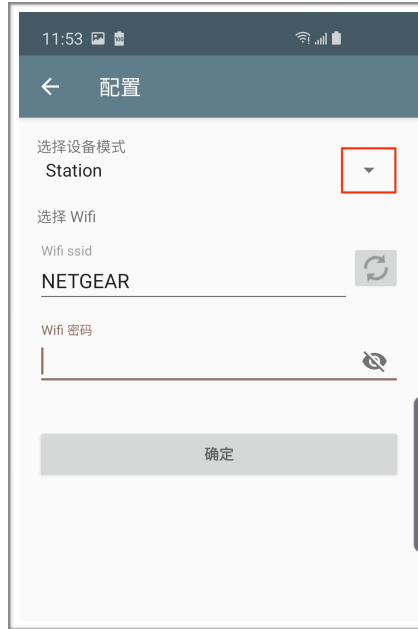


Figure 3-3. Network Configuration Interface

5. Select the device mode from the dropdown menu, as Figure 3-4 shows. (BluFi networking supports the following three modes: Station, SoftAP, and Station+SoftAP.)




Figure 3-4. Selecting Device Mode



6. Select Station mode as an instance, click on the refresh button , and choose the Wi-Fi SSID. Then, enter the password.



Figure 3-5. Configuring the Station Mode

7. Click on the  button to complete the configuration. Successful configuration will lead to the interface as Figure 3-6 shows. The information marked in red in Figure 3-6 shows the Wi-Fi mode (which is Station mode in this example) and the connection information (such as the AP's BSSID and SSID, as well as the connection status).

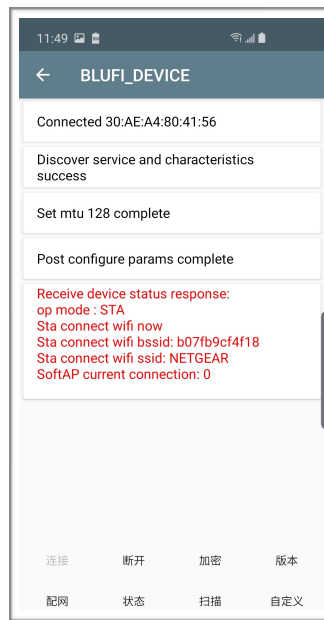


Figure 3-6. Station Connection Information



Meanwhile, the following log will be output to the serial port tool:

```
I (121745) BLUFI_DEMO: BLUFI Set WIFI opmode 1
I (121845) BLUFI_DEMO: Recv STA SSID DVES_HOME
I (121945) BLUFI_DEMO: Recv STA PASSWORD 12345678
I (122085) BLUFI_DEMO: BLUFI request wifi connect to AP
I (122955) wifi: n:6 2, o:1 0, ap:255 255, sta:6 2, prof:1
I (123615) wifi: state: init -> auth (b0)
I (123625) wifi: state: auth -> assoc (0)
I (123645) wifi: state: assoc -> run (10)
I (123675) wifi: connected with DVES_HOME, channel 6
I (124305) event: ip: 192.168.1.105, mask: 255.255.255.0, gw: 192.168.1.1
I (133645) wifi: pm start, type:0
```

3.3. Setting ESP32 as a SoftAP

1. Power up the module. Connect the EspBlufi app to ESP32 via Bluetooth. Set the device mode as SoftAP in the network configuration interface, as Figure 3-7 shows. For details of doing so, please refer to Steps 1 to 4 in **Section 3.2**.



Figure 3-7. Selecting SoftAP Mode

2. Configure the SoftAP mode, as Figure 3-8 shows. Select the security encryption mode, channel, and maximum number of Stations to be connected. Enter the SoftAP's SSID and password. Click on the **确定** button to complete the configuration.



Figure 3-8. Configuring the SoftAP Mode

3. Success of the SoftAP configuration will lead to the interface as Figure 3-9 shows. The current Wi-Fi mode and the connection status are shown in the interface.

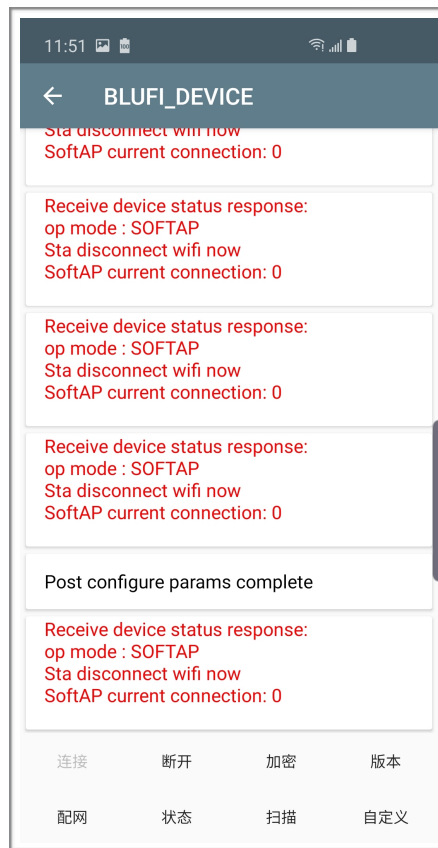


Figure 3-9. SoftAP Connection Information



Meanwhile, the following log will be output to the serial port tool:

```
I (141967) wifi: mode : softAP (24:0a:c4:01:4d:bd)
I (142067) BLUFI_DEMO: Recv SOFTAP SSID blufi_softap
, ssid len 13
I (142167) BLUFI_DEMO: Recv SOFTAP PASSWORD 12345678
I (142267) BLUFI_DEMO: Recv SOFTAP CHANNEL 10
I (142357) BLUFI_DEMO: Recv SOFTAP MAX CONN NUM 2
I (143107) BLUFI_DEMO: Recv SOFTAP AUTH MODE 4
```

- 4. Turn on the smartphone's Wi-Fi capability. The SoftAP connected can be found in the WLAN interface, as Figure 3-10 shows.

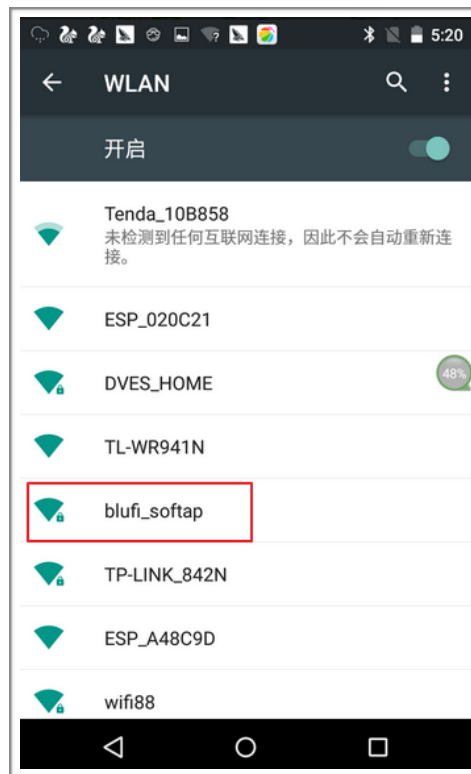


Figure 3-10. The Configured ESP32 SoftAP

- 5. Connect the smartphone to the ESP32 SoftAP. As shown in Figure 3-11, the prompt **网络已连接** will appear, indicating that the Wi-Fi is connected.

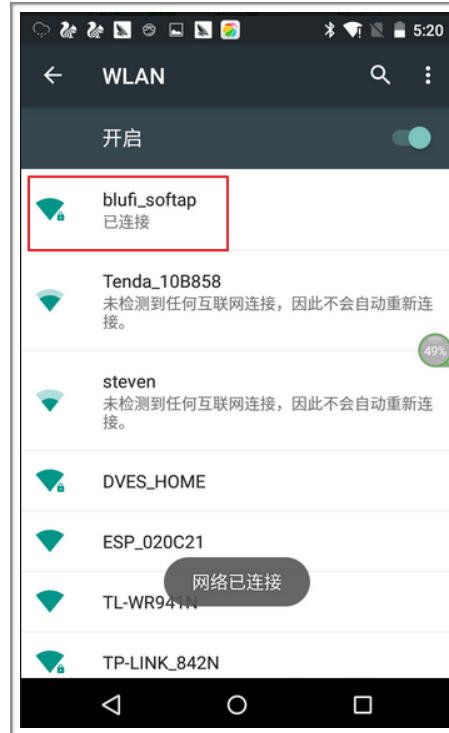


Figure 3-11. Wi Fi Connection Prompt

Meanwhile, the following log will be output to the serial port tool:

```
I (293557) wifi: n:10 o:10 0, ap:10 2, sta:255 255, prof:10  
I (293557) wifi: station: 98:d6:f7:64:13:08 join, AID=1, n, 20
```

The ESP32 device is successfully connected to Wi-Fi via Bluetooth networking.

Note:

All values of connection time shown on the EspBlufi app screenshots in this document are provided only as examples. Connection time is subject to the actual situations.



Espressif IoT Team
www.espressif.com

Disclaimer and Copyright Notice

Information in this document, including URL references, is subject to change without notice.

THIS DOCUMENT IS PROVIDED AS IS WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR SAMPLE.

All liability, including liability for infringement of any proprietary rights, relating to the use of information in this document, is disclaimed. No licenses express or implied, by estoppel or otherwise, to any intellectual property rights are granted herein.

The Wi-Fi Alliance Member logo is a trademark of the Wi-Fi Alliance. The Bluetooth logo is a registered trademark of Bluetooth SIG.

All trade names, trademarks and registered trademarks mentioned in this document are property of their respective owners, and are hereby acknowledged.

Copyright © 2019 Espressif Inc. All rights reserved.