



# Airoha IoT SDK Ultra Low Latency V2 Developer's Guide

Version: 1.5

Release date: 20 July 2023

---

© 2022 Airoha Technology Corp.

This document contains information that is proprietary to Airoha Technology Corp. ("Airoha") and/or its licensor(s). Airoha cannot grant you permission for any material that is owned by third parties. You may only use or reproduce this document if you have agreed to and been bound by the applicable license agreement with Airoha ("License Agreement") and been granted explicit permission within the License Agreement ("Permitted User"). If you are not a Permitted User, please cease any access or use of this document immediately. Any unauthorized use, reproduction or disclosure of this document in whole or in part is strictly prohibited. THIS DOCUMENT IS PROVIDED ON AN "AS-IS" BASIS ONLY. AIROHA EXPRESSLY DISCLAIMS ANY AND ALL WARRANTIES OF ANY KIND AND SHALL IN NO EVENT BE LIABLE FOR ANY CLAIMS RELATING TO OR ARISING OUT OF THIS DOCUMENT OR ANY USE OR INABILITY TO USE THEREOF. Specifications contained herein are subject to change without notice.

## Document Revision History

---

Revision	Date	Description
1.0	30 June 2022	<ul style="list-style-type: none"><li>ULL V2 Initial release</li></ul>
1.1	25 August 2022	<ul style="list-style-type: none"><li>Add Switch Dongle mode</li></ul>
1.2	25 October 2022	<ul style="list-style-type: none"><li>Add the UI behavior of ULL V2</li></ul>
1.3	02 March 2023	<ul style="list-style-type: none"><li>Add the Supported Codec Information of ULL V2</li></ul>
1.4	24 March 2023	<ul style="list-style-type: none"><li>Add the speaker mode</li></ul>
1.5	20 July 2023	<ul style="list-style-type: none"><li>Add the ABR feature</li></ul>

## Table of Contents

---

<b>1.</b>	<b>Introduction.....</b>	<b>1</b>
1.1.	Profile Overview .....	3
1.1.1.	ULL Service.....	4
1.2.	Usage Scenario .....	4
1.3.	Related SDK Library Requested .....	4
<b>2.</b>	<b>The ULL V2 Service.....</b>	<b>6</b>
2.1.	The ULL Message Sequence .....	6
2.1.1.	Connection Establishment .....	6
2.1.2.	Connection Release .....	7
2.1.3.	Set 2-RX Mixing Ratio.....	8
2.1.4.	Set Downlink Latency.....	9
2.1.5.	Critical data transmit-receive .....	9
2.1.6.	User data transmit-receive .....	10
2.2.	Using the ULL APIs .....	10
<b>3.</b>	<b>The UI behavior of ULL V2.....</b>	<b>13</b>
3.1.	Switch Dongle Mode.....	13
3.2.	Multi-link Mode and Single Link Mode.....	13
3.2.1.	Multi-link Mode .....	13
3.2.2.	Single Link Mode.....	13
3.2.3.	Reconnection Rule .....	13
3.3.	ABR Feature .....	14
3.3.1.	Adaptive bitrate mode.....	14
3.4.	Speaker Feature .....	15
3.4.1.	Speaker Mode.....	15
3.5.	Wired USB Audio and Aux In .....	15
3.6.	State machine diagram.....	15
3.7.	ULL Profile Event.....	16
3.7.1.	Events of ULL .....	16
3.8.	Key Actions .....	16
3.8.1.	ULL Key Actions.....	16
3.8.2.	Audio Key Actions .....	16
3.8.3.	Media Key Actions .....	17
3.9.	AWS Data.....	17
3.10.	FOTA .....	17

## Lists of Tables and Figures

---

Table 1. Technical parameter support for ULL.....	2
Table 2. Airoha IoT SDK library support for ULL.....	4
Table 3. Configuration of A2DP_STANDBY_ENABLE .....	13
Table 4. Audio qualities of ABR mode.....	14
Figure 1. ULL roles and link .....	2
Figure 2. ULL v2 Supported Codec .....	3
Figure 3. Protocol Model .....	3
Figure 4. ULL usage scenario .....	4
Figure 5. ULL connection establishment message sequence .....	7
Figure 6. Disconnect ULL profile .....	7
Figure 7. Set 2-RX mixing ratio on Client.....	8
Figure 8. Set 2-RX mixing ratio on Serve .....	8
Figure 9. Set downlink latency .....	9
Figure 10. Critical data transmit-receive.....	9
Figure 11. User data transmit-receive.....	10
Figure 12. Audio quality switch flow .....	14
Figure 13. ULL state machine .....	15
Figure 14. Mix ratio .....	17

## 1. Introduction

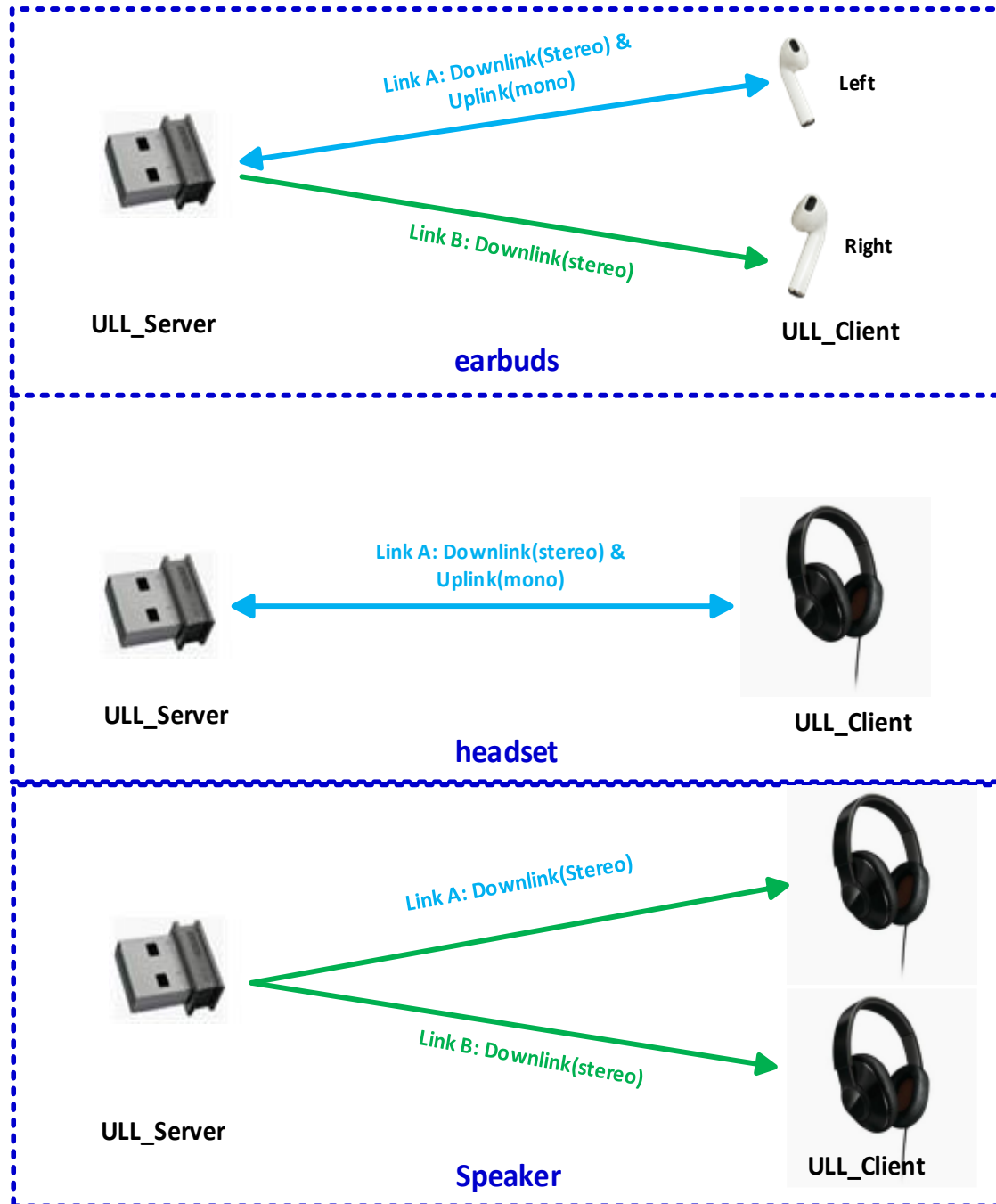
---

Ultra-Low Latency version 2(ULL V2) is an Airoha proprietary technology to support less than 20ms downlink voice/audio latency for headset/earbuds over Bluetooth LE with a well-matched Bluetooth-Dongle.

There are two roles in the ULL V2 profile:

- **ULL\_Server** — a device that usually has a capability with USB-in/line-in/i2S-in Audio Sound and encodes PCM audio data as LC3Plus or Vendor format. It can relay the data to remote device via wireless communication. It provides the following functions:
  - 2-Tx (PC→Device) & 1-RX (Device→PC)
  - 2-RX audio mixing ratio control
  - Firmware update via USB
  - Streaming start/stop state notification
- **ULL\_Client** — a device that acts as the remote audio input and output for **ULL\_Server**. There are two types of common device: Headset and Earbuds. It provides the following functions:
  - Headset
    - LE Connection with ULL\_Server
    - Firmware update via air
    - 3.5mm line-in
    - Latency switch
    - Multi-link (Dongle with ULL\_Server Feature + Smartphone's HFP)
    - USB Audio
  - Earbuds
    - LE Connection with ULL\_Server
    - Firmware update via air
    - Latency switch
    - Multi-link (Dongle with ULL\_Server Feature + Smartphone's HFP)
  - Speaker
    - Support two speakers at most with 1 ULL V2 Dongle
    - Only Support Downlink (music)
    - LE Connection with ULL\_Server
    - Firmware update via air

Figure 1 show that how ULL\_Server (dongle) connects to ULL\_Client (headset, earbuds or speakers) for transporting the audio streaming. Earbuds have two links (link A and link B) connected with ULL\_Server to transmit audio data, while the headset has only one link (link A). Link A is a bidirectional data link includes stereo downlink and mono uplink and link B is a unidirectional data link only includes a stereo downlink.



**Figure 1. ULL roles and link**


Ultra-Low Latency V2 (ULL V2) supports lower latency and higher bitrates than Ultra-Low Latency V1 (ULL V1). The downlink speed of ULL V2 supports up to 304 Kbps. The uplink speed of ULL V2 supports 64 Kbps. Table 1 shows more information.

**Table 1. Technical parameter support for ULL**

Category	ULL V1	ULL V2
Latency	22.5 ms	< 20 ms
Codec	Airoha in-house	LC3plus and Airoha in-house

Category	ULL V1	ULL V2
Downlink speed	256 kbps	172~304 kbps
Uplink speed	32 kbps	64 kbps

Figure 2 show that the detail codec information of ULL V2.

	Sinks	DL <sup>F</sup>	DL <sup>B</sup>	DL <sup>CH</sup>	DL Bit Rate	UL <sup>F</sup>	UL <sup>B</sup>	UL <sup>CH</sup>	UL Bit Rate	Urgent(5ms) Share <sup>*1</sup>	Latency
ULL 1.0 Opus	2	48k	16b	2	256Kbps	16k	16b	1	64Kbps	20B/2.5ms	~23ms
ULL 2.0 Opus <sup>*3</sup>		48k	16b		320Kbps	16k(156x) 32k(158x/157x) 			64Kbps	100B	~20ms
LC3plus (96K Hi-Res, 48K HD)		96K <sup>*2</sup>	24b		172.8Kbps				64Kbps	100B	~20ms
					304Kbps				64Kbps	100B	~20ms
					48k				200Kbps	64Kbps	100B

\*1: 65 support 16K UL only. If want to support 32K need EC and iGO support 32K and that may have RAM insufficient issue.

\*2: 65 disable all of WVE (AMA/Gsound) to support LC3plus 96K DL because insufficient IIRAM/DRAM.

\*3: Only AB156X/AB157X support

\*4: Urgent channel data is kind of ISO data, it share BW with audio packet, 100B is maximum throughput w/o audio packet retransmission.

Figure 2. ULL v2 Supported Codec

This document guides you through:

- Support for Bluetooth with the library description and supported reference examples.
- Detailed descriptions of the ULL V2 profiles.
- Custom application development and debugging logs.

## 1.1. Profile Overview

Figure 3 shows the protocols and entities used in this profile.

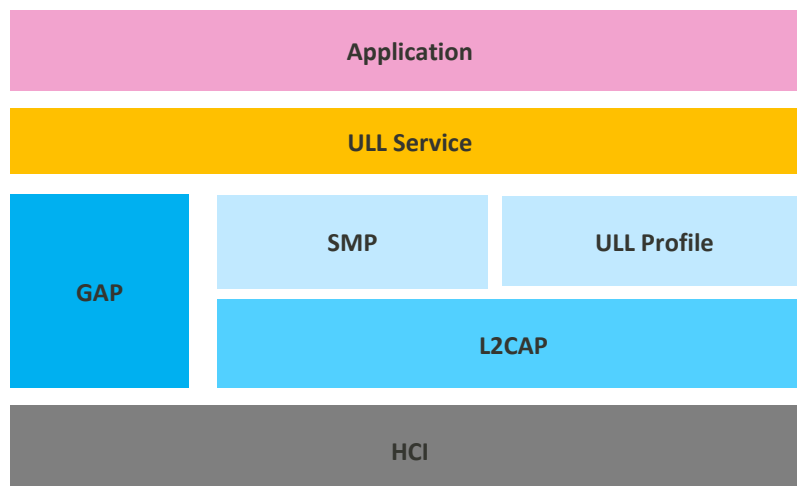


Figure 3. Protocol Model

The HCI, L2CAP, GAP, SMP, and ULL\_Profile protocols are described in the *Airoha IoT SDK Bluetooth Developers Guide.pdf* document under the <SDK\_root>/mcu/doc folder. The ULL Service is described in this document.

### 1.1.1. ULL Service

ULL Service is a service of ULL\_Profile to manage the ULL LE connections, and the configuration of audio data and the state machine of streaming transport.

It involves multiple C source files (e.g., bt\_ull\_le\_service.c, bt\_ull\_le\_conn\_service.c, bt\_ull\_le\_audio\_transmitter.c, bt\_ull\_le\_audio\_manager.c and bt\_ull\_le\_utility.c) located in mcu/middleware/airoha/bt\_ultra\_low\_latency folder.

The bt\_ull\_le\_audio\_transmitter.c file is only used to manage play/stop audio data for **ULL\_Server**, while bt\_ull\_le\_audio\_manager.c file is only for **ULL\_Client**.

### 1.2. Usage Scenario

**ULL\_Server** is a device that supports USB-in/line-in/i2S-in Audio Sound capability. It encodes the PCM streaming to LC3Plus or Vendor format and transmits to **ULL\_Client** via Bluetooth LE technology.

**ULL\_Client** can supports multilink connections (Dongle with ULL\_Server Feature + Smartphone's HFP).

The multi-link usage scenario is shown in Figure 4.

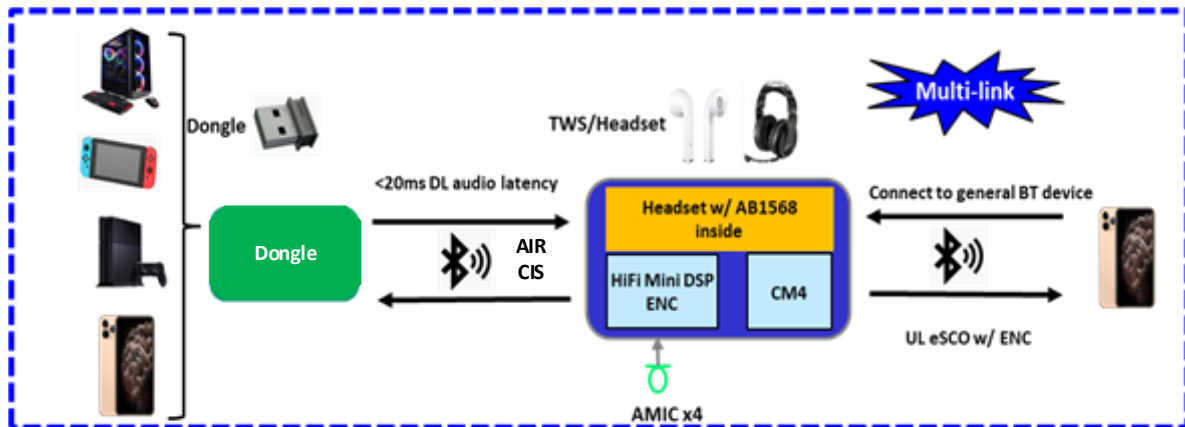


Figure 4. ULL usage scenario

### 1.3. Related SDK Library Requested

The ULL feature can only be run on Airoha IoT SDK for BT-Audio platform with the requested library files to interface the Bluetooth with C source and header files related to the platform, as shown in Table 2.

Table 2. Airoha IoT SDK library support for ULL

Module	Location	File Name	Function
Bluetooth	mcu/prebuilt/middleware/airoha/bluetooth/lib/	libbt.a	BR/EDR and Bluetooth LE stack library
		libbt_driver_[chip].a	Bluetooth driver library
		libbt_aws_mce.a	MCSync library, including MCSync implementation
		libbt_ull.a	ULL library
		libpka_dongle.a	Bluetooth controller library for ULL_Server (with ULL v2 functions)



Module	Location	File Name	Function
	mcu/prebuilt/middlew are/airoha/le_audio/ lib/	libpka_ull_g2_aws_le audio_emp.a	Bluetooth controller library for ULL_Client (with ULL v2 functions)
	mcu/prebuilt/middlew are/airoha/bluetooth /inc/	bt_platform.h	Interface for Bluetooth tasks
		bt_type.h	Common data types
		bt_system.h	Interface for the system, such as power on or off, memory initiation, and callback APIs for event handling
		bt_uuid.h	Interface for the UUID
		bt_codec.h	Interface for the codec
		bt_aws_mce.h	Interface for the MCSync
		bt_gap_le.h	Interface for the GAP LE
		bt_os_layer_api.h	Wrapper APIs for RTOS, memory, advanced encryption standard (AES), and rand
		bt_debug.h	Encapsulated debugging interface
		bt_hci_log.h	Encapsulated interface for the HCI logging
		bt_ull_le.h	Interface for the ULL profile
	/mcu/middleware/airo ha/bt_ultra_low_late ncy/inc	bt_ull_service.h	Common API for ULL service
		bt_ull_le_service.h	Interface for ULL V2 service

## 2. The ULL V2 Service

---

### 2.1. The ULL Message Sequence

The ULL V2 procedure can be established using the message sequence. The message sequence for each process is described below:

- 1) Connection establishment
- 2) Connection release
- 3) Set 2-RX mixing ratio
- 4) Set downlink latency
- 5) Critical data transmit-receive
- 6) User data transmit-receive

#### 2.1.1. Connection Establishment

Use the connection establishment operation to establish an LE connection between ULL Server and ULL Client.

Set Identity Resolving Key (SIRK) is associated with the Coordinated Set. All ULL clients that are part of the same Coordinated Set shall use the same SIRK. The SIRK is a 128-bit long random number. For example, earbuds have two devices named “agent” and “partner”. The agent and partner belong to a coordinated set so that they should have the same SIRK. It is also used to generate the RSI information that is included in the data of advertising.

On the ULL Client side, ‘**bt\_ull\_le\_srv\_set\_device\_info ()**’ is used to set SIRK before starting the advertising. User can reset SIRK by AT CMD ‘AT+LEULL=SIRK,SET,xx,xx,xx,xx,xx,xx,xx,xx,xx,xx,xx,xx,xx,xx,xx,xx\0d\0a’.

Also, ‘**bt\_ull\_le\_srv\_get\_uuid ()**’ and ‘**bt\_ull\_le\_srv\_get\_rsi ()**’ are used to combine advertising data before starting the advertising.

While, on the ULL Server side, the function ‘**bt\_ull\_le\_srv\_verify\_rsi ()**’ is used to verify the RSI information using the same SIRK with the ULL Client devices.

For more details, please refer to

<SDK\_root>/mcu/middleware/airoha/bt\_ultra\_low\_latency/inc/bt\_ull\_le\_service.h

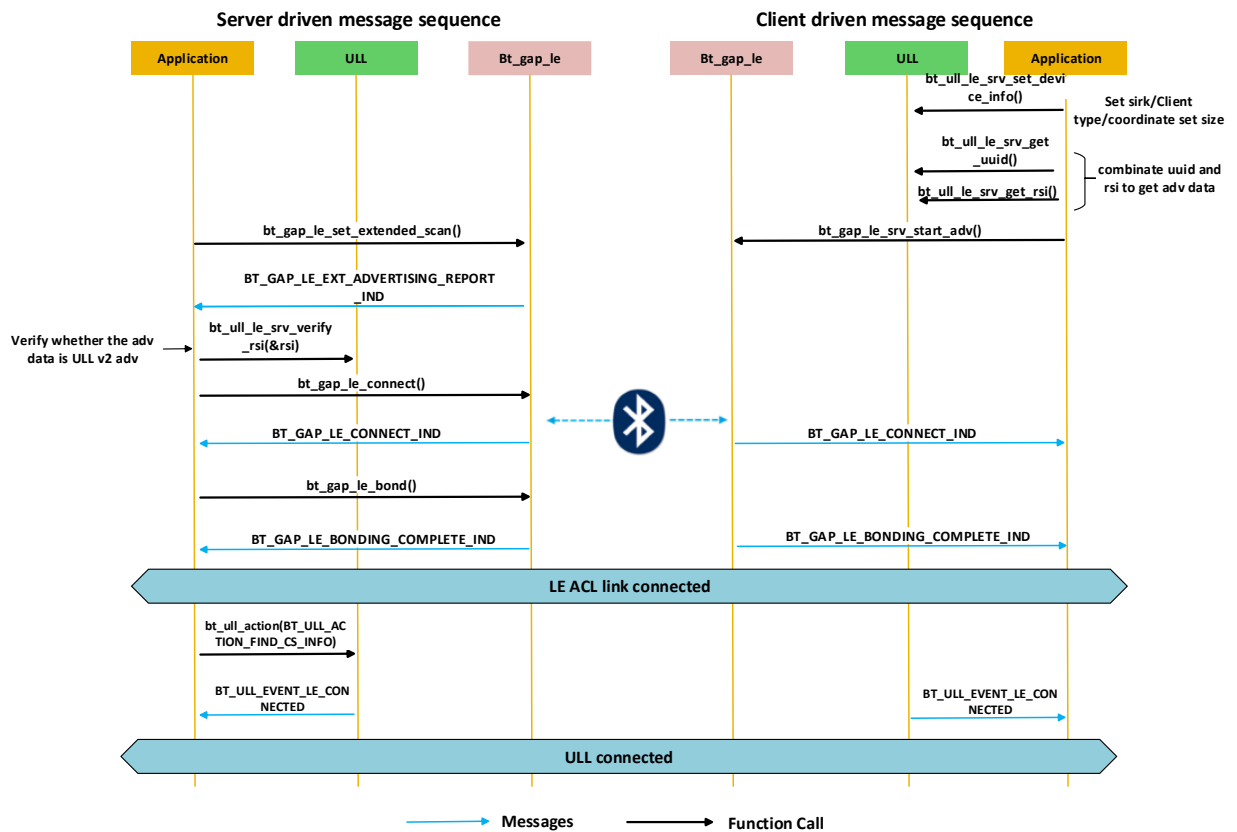


Figure 5. ULL connection establishment message sequence

### 2.1.2. Connection Release

The connection release procedure is used to disconnect the LE ACL link between the ULL server and ULL client. Both Server and Client can initiate the disconnection procedure.

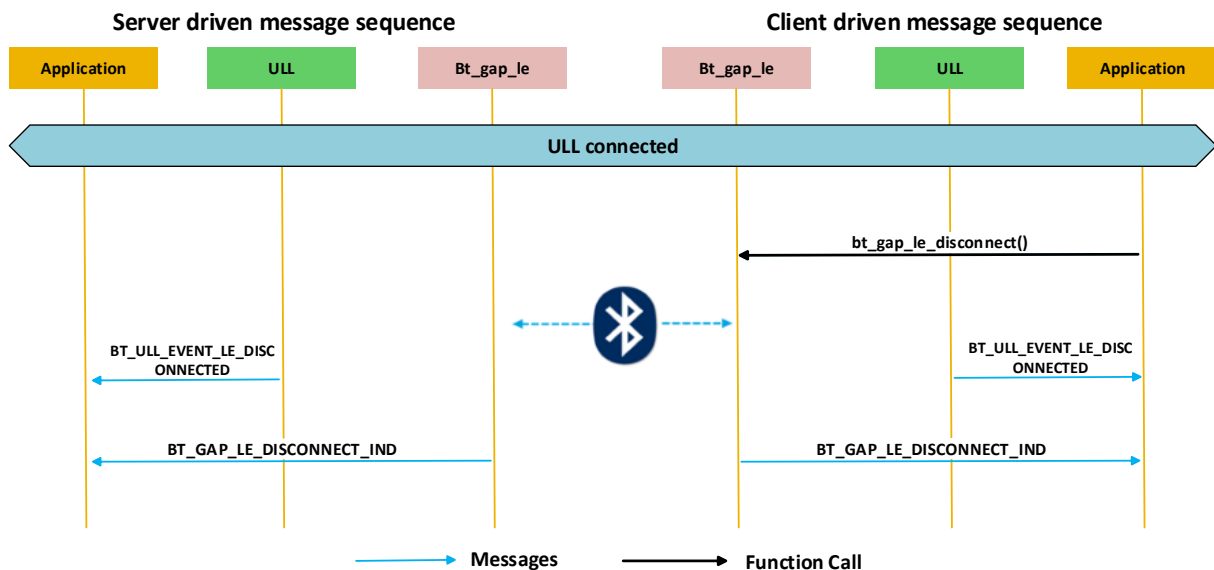


Figure 6. Disconnect ULL profile

#### 2.1.3. Set 2-RX Mixing Ratio

Use the set mixing ratio operation to control USB Host (Ex. PC) so that you can combine two audio streams together. It supports 0% ~ 100% ratio adjustment for two audio streams independently. For more details, refer to `<SDK_root>/mcu/middleware/airoha/bt_ultra_low_latency/inc/bt_ull_service.h`.

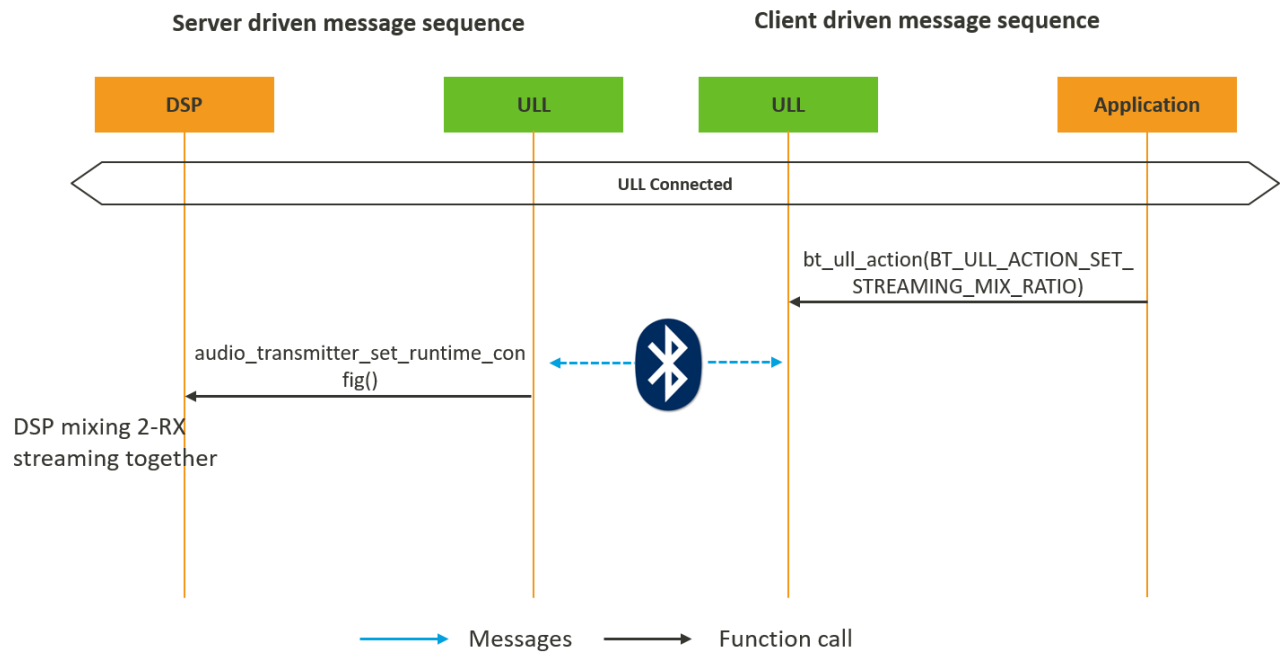


Figure 7. Set 2-RX mixing ratio on Client

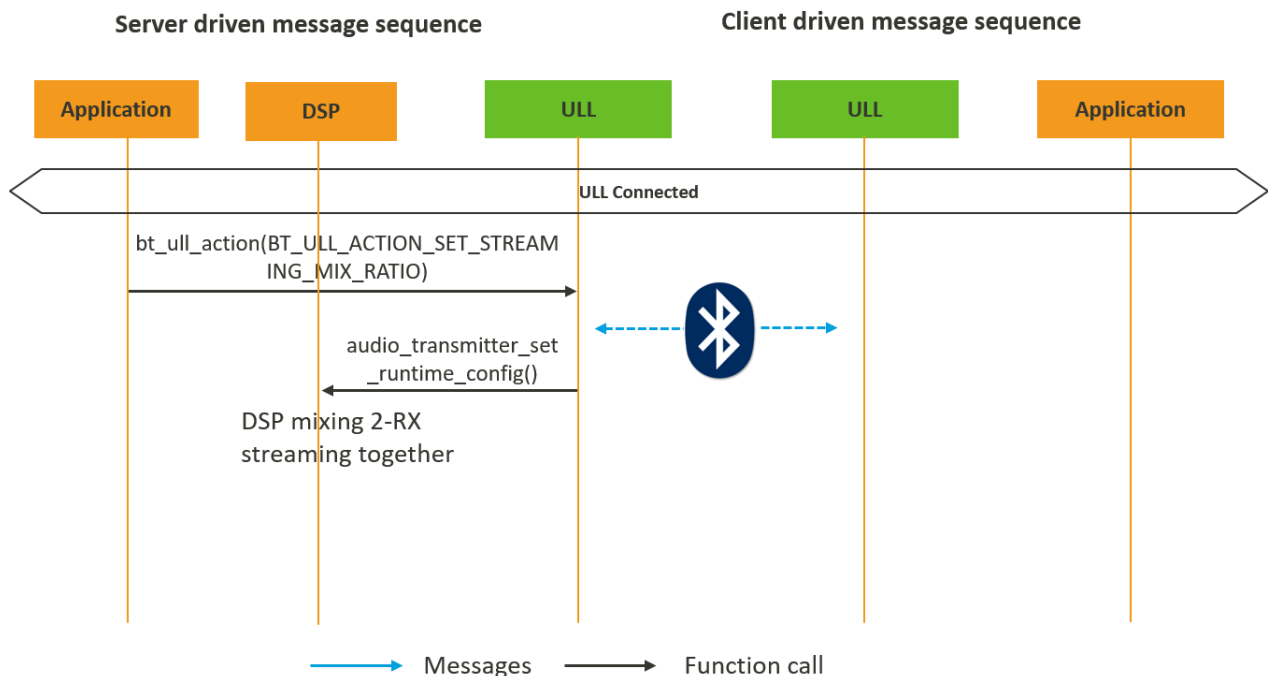


Figure 8. Set 2-RX mixing ratio on Serve

#### 2.1.4. Set Downlink Latency

In a multilink scenario, to support more than one link, we must sometimes change the ULL downlink streaming latency to a different value (Ex. 20ms, 30ms, or 40ms) on ULL\_Client due to the limitation of Bluetooth bandwidth. For more details, refer to <SDK\_root>/mcu/middleware/airoha/bt\_ultra\_low\_latency/inc/bt\_ull\_le\_service.h.

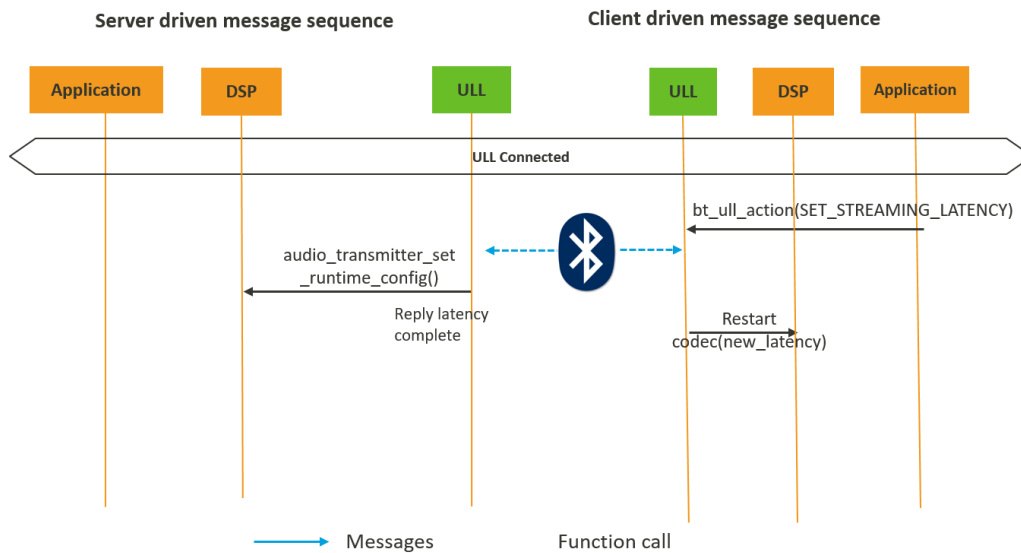


Figure 9. Set downlink latency

#### 2.1.5. Critical data transmit-receive

Use the critical data transmit-receive to exchange some unreliably continuous data (such as sensor data) with a flush timeout between Server and Client. The maximum length of critical data is 100 bytes. There is currently only support for Client to send critical data to Server when ULL is streaming. For more details, refer to <SDK\_root>/mcu/middleware/airoha/bt\_ultra\_low\_latency/inc/bt\_ull\_service.h.

##### ULL Critical Data Transmit-Receive

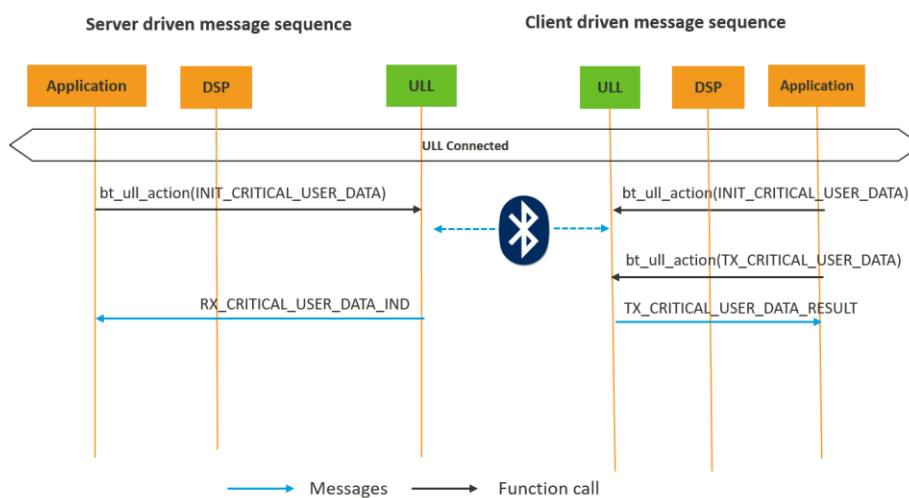


Figure 10. Critical data transmit-receive

### 2.1.6. User data transmit-receive

Use the user data transmit-receive to exchange user defined data between Server and Client. For more details, refer to <SDK\_root>/mcu/middleware/airoha/bt\_ultra\_low\_latency/inc/bt\_ull\_service.h.

ULL User Data Transmit-Receive

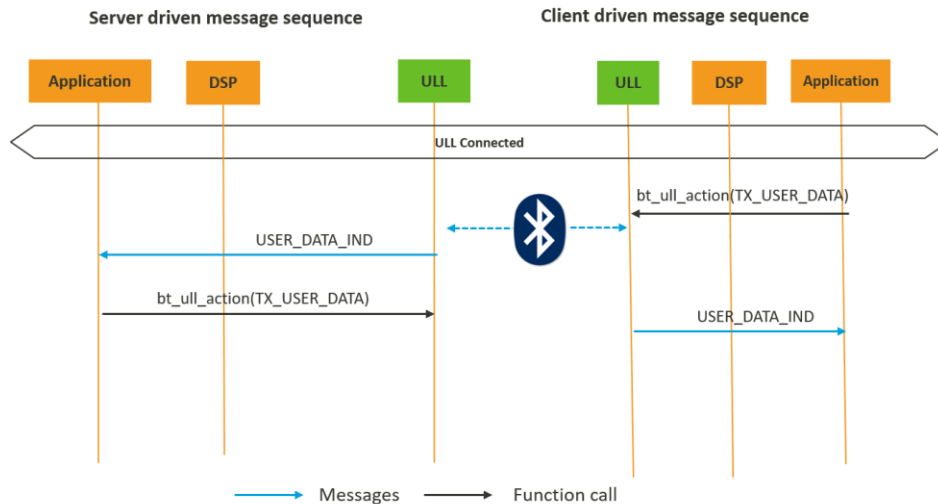


Figure 11. User data transmit-receive

## 2.2. Using the ULL APIs

This section describes how to use the ULL APIs for application development. The functionality of the ULL APIs is implemented in the module `bt_ultra_low_latency`, related APIs can be found in <SDK\_root>/mcu/middleware/airoha/bt\_ultra\_low\_latency/inc/bt\_ull\_le\_service.h, the other header files are used internally, and applications cannot use them at any time.

- 1) Call `bt_ull_le_srv_init()` to start the ULL role during the initiation process in Dongle as Server or Headset/Earbuds as Client when the system powers on.

```
bt_ull_le_srv_init(role, callback);
```

- 2) Call `bt_ull_le_srv_set_device_info()` to set the necessary device information of the ULL Client according the ULL client is a Headset or Earbuds device before the Bluetooth powers on. It is only initiated by the ULL Client.

```
bt_ull_le_device_info_t dev_info;
dev_info.client_type = BT_ULL_EARBUDS_CLIENT ;
dev_info.size = 2; /**< The size of ULL LE Coordinated set. */
dev_info.sirk = {0x00,0x01,0x02,...0x0F};
dev_info.group_device_addr = {0xC1,0xC2,...0xC6}; //for earbuds, here are the 2
earbuds's device address.
bt_ull_le_srv_set_device_info(&dev_info);
```

- 3) Call `bt_ull_le_srv_action()` to control audio stream, e.g., ULL\_Server or ULL\_Client setting the volume.

```
bt_ull_volume_t volume_param;
volume_param.streaming.streaming_interface =
BT_ULL_STREAMING_INTERFACE_SPEAKER;
volume_param.streaming.port = 0;
volume_param.action = BT_ULL_VOLUME_ACTION_SET_UP;
volume_param.channel = BT_ULL_AUDIO_CHANNEL_DUAL;
volume_param.volume = 1;
bt_ull_le_srv_action(BT_ULL_ACTION_SET_STREAMING_VOLUME,&volume_param,sizeof(vo
lume_param));
```

4) Call `bt_ull_le_srv_get_streaming_info()` to get the specified streaming information.

```
bt_ull_streaming_t streaming
streaming.streaming_interface = BT_ULL_STREAMING_INTERFACE_SPEAKER;
streaming.port = 0;
bt_ull_streaming_info_t info = {0}
bt_ull_le_srv_get_streaming_info(streaming,&info);
```

5) Call `bt_ull_le_srv_lock_streaming()` to lock or unlock the streaming. For example, the upper user can lock the streaming before the OTA procedure is started.

```
bt_ull_le_srv_lock_streaming(true);
```

6) Call `bt_ull_le_srv_get_uuid()` to get the UUID of ULL V2. The UUID is included in the advertising data. It is used to verify whether the device supports ULL V2.

```
bt_ull_le_uuid_t *uuid;
uuid = bt_ull_le_srv_get_uuid ();
```

7) Call `bt_ull_le_srv_get_rsi()` to calculate the Resolvable Set Identifier (RSI). The RSI is randomly generated by the SIRQ. An RSI can be resolved if the corresponding SIRQ is available by using the Resolvable Set Identifier resolution operation.

```
bt_ull_le_rsi_t rsi;
bt_ull_le_srv_get_rsi(&rsi);
```

8) Call `bt_ull_le_srv_verify_rsi()` to verify the RSI using the correct SIRQ.

```
bt_ull_le_rsi_t rsi;
bt_ull_le_srv_verify_rsi(&rsi);
```

9) Call `bt_ull_le_srv_get_role()` to get the role of ULL service.

```
bt_ull_role_t role;
role = bt_ull_le_srv_get_role();
```

10) Call `bt_ull_le_srv_set_access_address()` to set the vendor access address for transmission air interface packets.

```
bt_ull_le_set_adv_scan_access_addr_t access_addr = {0};
access_addr.access_addr[0] = 0x6D;
access_addr.access_addr[1] = 0xEB;
```

```
access_addr.access_addr[2] = 0x98;  
access_addr.access_addr[3] = 0xE8;  
bt_ull_le_srv_set_access_address(&access_addr);
```

11) Call `bt_ull_le_srv_enable_adaptive_bitrate_mode()` to enable or disable the adaptive bitrate mode.

```
bt_ull_le_adaptive_bitrate_params_t adaptive_bitrate_param;  
adaptive_bitrate_param.enable = true;  
adaptive_bitrate_param.crc_threshold = 9;  
adaptive_bitrate_param.flush_timeout_threshold = 3;  
adaptive_bitrate_param.report_interval = 100;  
adaptive_bitrate_param.rx_timeout_threshold = 3;  
bt_ull_le_srv_enable_adaptive_bitrate_mode(&adaptive_bitrate_param);
```



### 3. The UI behavior of ULL V2

---

#### 3.1. Switch Dongle Mode

The dongle project on SDK3.4.0 supports the coexistence of the LEA and ULL2.0 features. However, there can be only one kind of connection at the same time. Therefore, we provide some ways to allow users to switch the dongle to a specific mode, i.e. ULL 2.0 mode or LE Audio mode.

The user can switch mode by customization. For example, AT CMD, pressing a button, or via the config tools. The SDK3.4.0 provides a way to select the mode using AT CMD.

- 1) Send AT CMD: AT+DONGLE\_MODE=ULL2 to switch the Dongle mode to ULL2.0 mode.
- 2) Send AT CMD: AT+DONGLE\_MODE=BROADCAST to switch the Dongle mode to LE Audio broadcast mode.
- 3) Send AT CMD: AT+DONGLE\_MODE=UNICAST to switch the Dongle mode to LE Audio unicast mode.

#### 3.2. Multi-link Mode and Single Link Mode

##### 3.2.1. Multi-link Mode

Multi-link mode means that the DUT (headset or earbuds) can be connected to one dongle and one smartphone (HFP profile only by default) at the same time.

If you want the DUT to support the A2DP profile when both the dongle and smartphone are connected with the DUT, you can use the ATCMD: "AT+LEULL=A2DPSTANDBY,ON" to enable it. Therefore, in a different setting when both SRC are connected, the supported profiles and the latency value between the dongle and DUT is described in the following table.

**Table 3. Configuration of A2DP\_STANDBY\_ENABLE**

A2DPSTANDBY enabled	Supported profiles by smartphone	Latency value between DUT and dongle
y	A2DP and HFP	40ms
n	HFP	20ms

If the DUT is connected to only one source device, it still enables the page scan or LE advertising so another source device can connect to the DUT. Therefore, if the DUT is first connected to one ULL Dongle, the enabled page scan shares the Bluetooth bandwidth BT resource, the latency of the connection between the ULL dongle, and the DUT increases to 40ms.

##### 3.2.2. Single Link Mode

The single link mode means the DUT can be connected to only one source device at a time, either the ULL Dongle or smartphone. The latency of the connection between the dongle and the DUT is 20ms. In this mode, the user can use a key to switch the connection between the dongle and the smartphone.

If Bluetooth powers on in this mode, the device reconnects to the last connected device.

##### 3.2.3. Reconnection Rule

When the DUT has connected to one or two SRC devices, the DUT reconnects to the connected devices.

### 3.3. ABR Feature

#### 3.3.1. Adaptive bitrate mode

You can call `bt_ull_le_srv_enable_adaptive_bitrate_mode()` function to enable or disable Adaptive Bitrate (ABR) mode at the SDK3.8.0.

When ABR mode is enabled, the ULL service automatically switches to the appropriate audio quality based on the Quality of Service (QoS). When ABR mode is disabled, the ULL service works in the default audio quality. SDK3.8.0 provides three audio qualities to switch to in ABR mode.

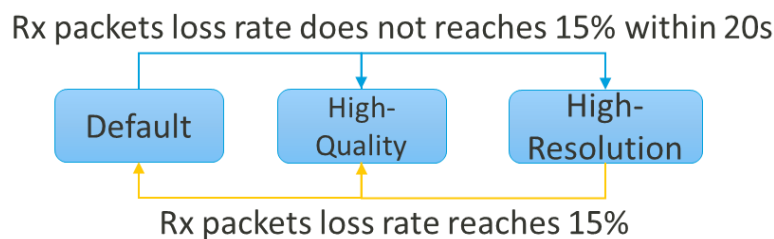
**Table 4. Audio qualities of ABR mode**

Audio quality	UL sample rate (kHz)	UL bitrate (kbps)	DL sample rate (kHz)	DL bitrate (kbps)
Default	32 kHz	64 kbps	96 kHz	304 kbps
High-Quality	32kHz	104 kbps	96 kHz	560 kbps
High-Resolution	32 kHz	104 kbps	96 kHz	940.8 kbps

There is a way to enable/disable ABR mode for testing. Send the AT command on the headset or earbuds to enable/disable ABR mode.

- 1) Send AT command: `AT+LEULL=ABR,enable,100` (100 is the QoS event report interval if Rx packet is bad, unit: ms)
- 2) Send AT command: `AT+LEULL=ABR,disable`

When ABR mode is enabled, it checks the RX packet loss rate every report interval. The audio quality uses the High-Resolution level by default. If the packet loss rate reaches 15%, the ULL service switches the audio quality to a lower level until it reaches the lowest level. If the packet loss rate does not reach 15% within 20s, the ULL service switches to a higher level of audio quality until it reaches the highest level.



**Figure 12. Audio quality switch flow**

If you want to work on one of the audio qualities shown in Table 4, You can use the following AT command while ABR mode is disabled.

- 1) Default audio quality, send AT command: `AT+LEULL=AUD_QOS,1`
- 2) High-Quality audio quality, send AT command: `AT+LEULL=AUD_QOS,4`
- 3) High-Resolution audio quality, send AT command: `AT+LEULL=AUD_QOS,5`

### 3.4. Speaker Feature

#### 3.4.1. Speaker Mode

You can switch to Speaker mode on a Headset project. The SDK3.6.0 provides a way to switch the mode using the AT CMD.

AT CMD: AT+LEULL=SPEAKERMODE,ON/OFF ON, means that it switches to Speaker mode on a Headset Project, i.e. the Headset supports the speaker feature. There is support only for Downlink (i.e. music) in this mode; OFF means it switches to the normal Headset features.

You must reset the device after running this AT CMD.

### 3.5. Wired USB Audio and Aux In

This feature is only supported on headset projects.

When the wired USB audio is enabled or Aux in is plugged in, the DUT disconnects the dongle. If the DUT is currently connected to a smartphone, it tries to reconnect the A2DP profile.

When wired USB audio is disabled or Aux in is not plugged in, the DUT tries to reconnect with the dongle.

### 3.6. State machine diagram

The state machine diagram includes connection, disconnection, AUX or USB audio in or out, and using the key switch connection.

The multi-link mode has one more state than the single link mode, i.e. connected 2 SRC, which is shown in blue in this diagram.

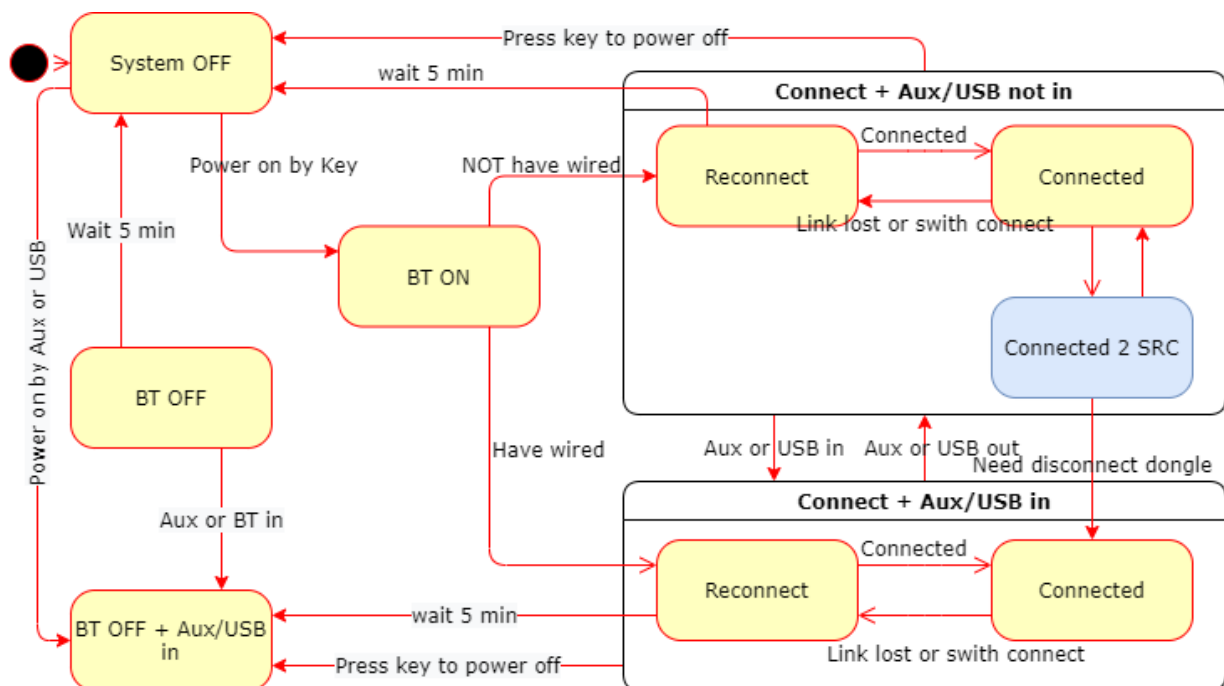


Figure 13. ULL state machine

### 3.7. ULL Profile Event

#### 3.7.1. Events of ULL

The upper user can register an event callback function to listen to some ULL events from the ULL profile.

e.g., listens BT\_ULL\_EVENT\_LE\_CONNECTED and BT\_ULL\_EVENT\_LE\_DISCONNECTED to get the result of ULL connection. And, listens BT\_ULL\_EVENT\_LE\_STREAMING\_START\_IND, BT\_ULL\_EVENT\_LE\_STREAMING\_STOP\_IND to get the streaming status.

### 3.8. Key Actions

#### 3.8.1. ULL Key Actions

There are some key actions that are specifically for the ULL project. They are:

- KEY\_DISCOVERABLE, to trigger the earbuds or headset to start the advertising.
- KEY\_ULL\_SWITCH\_LINK\_MODE, on the headset or earbuds side, to trigger the switch for the link mode between single mode and multi-link mode. The single mode means only connection to either a smartphone or dongle at the same time. The multi-link mode means the DUT can connect to both the smartphone and dongle.
- KEY\_ULL\_RECONNECT, on the headset or earbuds side, to trigger the switch for the connection between the smartphone and dongle. It is only useful under single mode.

The key mapping table is defined in <project>\src\boards\<Your board>\customerized\_key\_config.c; Customer can change the table to define the preferred table.

Customer can refer to app\_ull\_idle\_activity.c to review how to process the key events.

#### 3.8.2. Audio Key Actions

Currently, the code uses a rotary key to change the mix ratio and side tone gain. Customer can review the code and implement the feature by the key event.

- Mix ratio:  
Our dongle can implement two audio channels on the PC. One channel is for gaming and the other channel is for chatting. Use the key action KEY\_AUDIO\_MIX\_RATIO\_GAME\_ADD and KEY\_AUDIO\_MIX\_RATIO\_CHAT\_ADD to process the requirement. The default setting is 21 levels. When level is gaming max level (default is 0), the gaming ratio is 100%, and the chat ratio is 0. When the level is balanced (default is 10), the gaming ratio and chat ratio is 100%. When the level is chat max level (default is 20), the gaming ratio is 0 and the chat ratio is 100%. Customer can change the macro ULL\_MIX\_RATIO\_GAME\_MAX\_LEVEL, ULL\_MIX\_RATIO\_CHAT\_MAX\_LEVEL and ULL\_MIX\_RATIO\_BALANCED\_LEVEL.

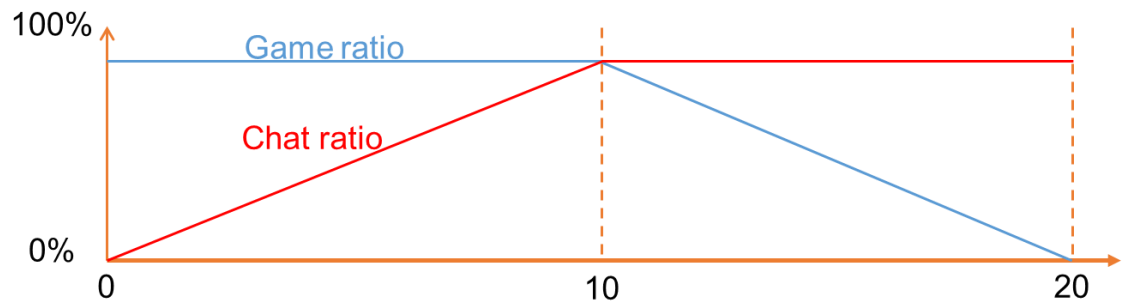


Figure 14. Mix ratio

- Side tone volume:  
Use the KEY\_AUDIO\_SIDE\_TONE\_VOLUME\_UP and KEY\_AUDIO\_SIDE\_TONE\_VOLUME\_DOWN to increase or decrease the side tone volume. The minimum value is defined as ULL\_SIDE\_TONE\_VOLUME\_MIN\_LEVEL; The maximum value is defined as ULL\_SIDE\_TONE\_VOLUME\_MAX\_LEVEL; The increasing or decreasing value when the user slides the rotary one step is defined as ULL\_SIDE\_TONE\_CHANGE\_LEVEL\_PRE\_STEP.

There is support for pressing a key to mute the microphone. The key action is KEY\_MUTE\_MIC.

### 3.8.3. Media Key Actions

Headset or earbuds can control the PC media. The PC media can be ULL connection audio or wired USB audio. The supported actions are KEY\_AVRCP\_PLAY, KEY\_AVRCP\_PAUSE, KEY\_AVRCP\_FORWARD and KEY\_AVRCP\_BACKWARD. If the headset or earbuds have connected with one smartphone and one PC, and both the smartphone and PC are not playing, the action occurs on the smartphone. If the PC is playing and the smartphone is not playing or is disconnected, the action occurs on the PC. The processing code for controlling the smartphone media or ULL media is in app\_music. The processing code for controlling USB audio media is in app\_usb\_audio which is only supported by the headset.

## 3.9. AWS Data

Some events related to the ULL must be sent from Agent to Partner or from Partner to Agent.

- Key event: Event group is EVENT\_GROUP\_UI\_SHELL\_KEY, event id is KEY\_ULL\_RECONNECT. The Partner cannot handle the reconnect key event so it sends the event to Agent for processing.

## 3.10. FOTA

When doing FOTA with a smartphone, APP calls bt\_cm\_write\_scan\_mode() to disable the page scan.