Telink B91m BLE Audio SDK

User Guide

AN-22063004-E3

Ver2.0.0

2023.07.24

Keyword

BLE, Audio

Brief

This document provides user guide for B91m BLE audio typical application scenarios, suitable for Telink B91 (TLSR951x/921x) and B92 (TLSR952x/922x) series chip.

Published by
Telink Semiconductor

Bldg 3, 1500 Zuchongzhi Rd,
Zhangjiang Hi-Tech Park, Shanghai, China

Legal Disclaimer

This document is provided as-is. Telink Semiconductor reserves the right to make improvements without further notice to this document or any products herein. This document may contain technical inaccuracies or typographical errors. Telink Semiconductor disclaims any and all liability for any errors, inaccuracies or incompleteness contained herein.

Information

For further information on the technology, product and business term, please contact Telink Semiconductor Company www.telink-semi.com

For sales or technical support, please send email to the address of:

telinksales@telink-semi.com

telinksupport@telink-semi.com

Revision History

Version	Change Description
V1.0.0	Initial release.
V1.1.0	Added chapter 2 and 3.
v2.0.0	Updated content, added chapters 4 and 5.

Contents

List of Figures

List of Tables

1 Hardware and Software Requirement

1.1 Hardware Requirement

1.1.1 Telink Unicast Audio Kit

Figure 1.1: Telink B91 unicast audio kit

Hardware for the kit:

Table 1.1: List of Telink unicast audio kit development boards

Category	IC part number	Development board external name	Quantity	Kit accessories
Unicast Server	9517C	B91 audio development board	2	USB cable (power) x2, Antenna x2
Unicast Client	9518A	B91 Dongle	1	-

> Purchase on taobao, if it is not already on the shelve, please contact Telink FAE to buy B91 Bluetooth Low Energy (BLE) audio kit.

1.1.2 Telink Broadcast Audio Kit

Figure 1.2: Telink B91 broadcast audio kit

Hardware for the kit:

Table 1.2: List of Telink broadcast audio kit development boards

Category	IC part number	Development board external name	Quantity	Kit accessories
Broadcast sink	9517C	B91 audio development board	2	USB cable (power) x2, Antenna x2
Broadcast source	9518A	B91 Dongle	2	-
Broadcast assistant	9518A	B91 EVK	1	USB cable (power) x1, Antenna x1

> Purchase on taobao, if it is not already on the shelve, please contact Telink FAE to buy B91 BLE audio kit.

1.1.3 Hardware Reference Design

- See the section Development Kit and Application Boards under page Telink wiki

1.2 Software Requirement

- Download tool
- SDK (Telink internal link, contact FAE to get it)

2 Unicast Audio Demo

The Unicast Audio Demo is divided into two roles: Unicast Client and Unicast Server.

The B91m BLE Audio SDK v1.1.0.0 extends the usage range of unicast demo, users can simply configure it to achieve:

- one-to-one (client to server), typical scenarios include headset.
- one to two (client to server), typical scenarios include TWS.
- Freely configurable audio uplink and downlink sampling rates of 16kHz/24kHz/32kHz/48kHz.
- Mulplixing of audio data is that multiple channels of audio data are transferred on a single channel.

2.1 Unicast Demo Scenario 1

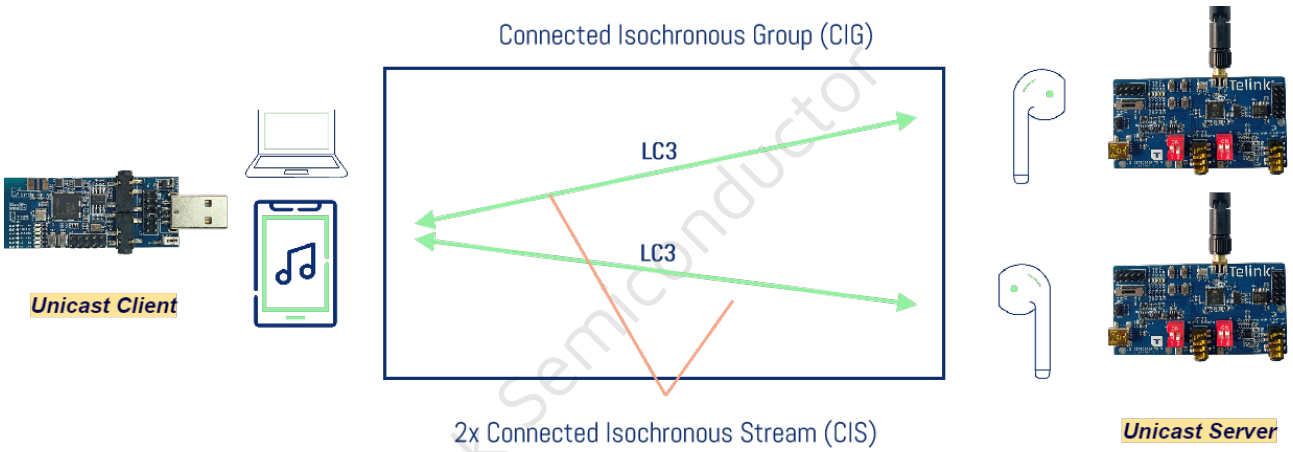

Figure 2.1: Unicast scenario 1

One unicast client to two unicast servers, with an uplink dual channel 16kHz sampling rate and a downlink dual channel 48kHz sampling rate, as shown in the above figure.

Development board for the kit:

Table 2.1: Unicast demo scenario 1 kit development boards

Category	IC part number	Development board external name	Quantity	Kit accessories
Unicast Server	9517C	B91 audio development board	2	USB cable (power) x2, Antenna x2
Unicast Client	9518A	B91 Dongle	1	-

2.1.1 Scenario Introduction

As a unicast client, B91 dongle establishes audio channels with two unicast servers respectively.

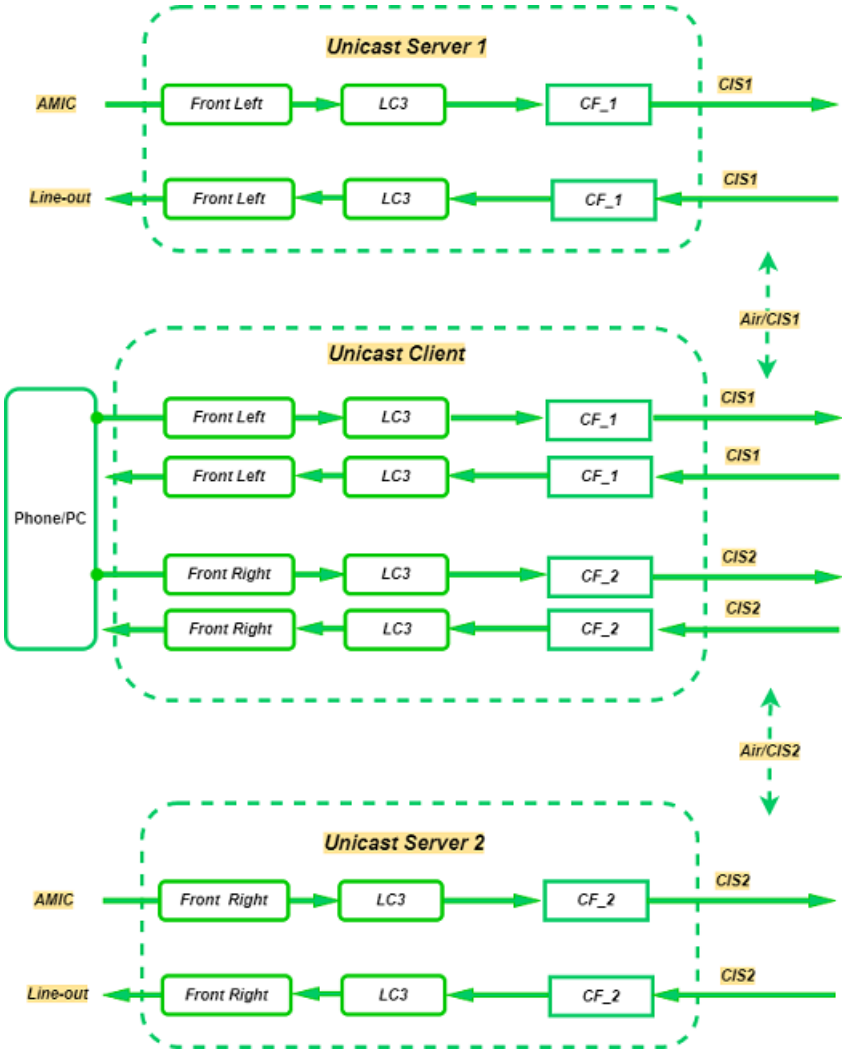

Figure 2.2: Unicast scenario 1 structure

- As the USB device, the B91 dongle acquires 48kHz dual channel audio data from the USB host (phone/PC connected to dongle), and sends the left and right channels to the two unicast servers respectively after LC3 compression.
- After the two unicast servers obtain the audio data from the audio channel established with the unicast client, the mono data is played through the local Line-out.
- The two unicast servers capture the 16kHz ambient sound data locally via AMIC and send it to the unicast client via the audio channel established with the unicast client.
- After the B91 dongle receives the 16kHz mono channel data transmitted by the two unicast servers, it synthesizes the 16kHz dual channel data locally, and it acts as a USB Mic device to send the data to the USB host.

Note:

> The Telink B91 dongle acts as a USB device that transmits 16kHz dual channel audio data to the USB host (phone/PC), some USB hosts may use one channel, and some USB hosts may use two channels.

2.1.2 Firmware Compilation - Unicast Client

You can skip this step if there is already firmware.

- **Step1**: Select TLSR9518DONGLE for hardware.

```
#define TLSR9518EVK                             1
#define TLSR9518DONGLE                          2
#define HARDWARE_TYPE                           TLSR9518DONGLE
```

- **Step2**: Select TWS for mode.

```
#define APP_SCENE_TWS                           0
#define APP_SCENE_HEADSET                       1
#define APP_AUDIO_SCENE                         APP_SCENE_TWS
```

- **Step3**: Select audio scenario.

```
#define APP_AUDIO_CONFIGURATION_PREFER
↪  BLC_AUDIO_11II_SVR_2_SINK_2_CHN_1_SRC_2_CHN_1_CISES_2_STREAMS_4
```

For details of the audio scenarios, please refer to **BAP_v1.0.1, Table4.1: Unicast LC3 Audio Configurations**.

- **Step4**: Select audio parameters, downlink 48kHz sampling rate, uplink 16kHz sampling rate.

```
#define APP_AUDIO_CODEC_INPUT_PARAMETER_PREFER
↪  BLC_AUDIO_STD_FREQ_48K_DURATION_10MS_FRAME_100BYTES
#define APP_AUDIO_CODEC_OUTPUT_PARAMETER_PREFER
↪  BLC_AUDIO_STD_FREQ_16K_DURATION_10MS_FRAME_40BYTES
```

- **Step5**: If using USB mode, you need to modify the USB audio enumeration parameters, speaker downlink dual channel 48kHz, Mic uplink dual channel 16kHz.

```
#define USB_SPEAKER_ENABLE      1
#define SPK_RESOLUTION_BIT      16
#define SPK_SAMPLE_RATE         48000
#define SPK_CHANNEL_COUNT       2

#define USB_MIC_ENABLE          1
#define MIC_RESOLUTION_BIT      16
#define MIC_SAMPLE_RATE         16000
#define MIC_CHANNEL_COUNT       2
```

If using codec mode, the codec parameters need to be modified.

```
tlk_codec_config(TLK_CODEC_OUTPUT,TLK_CODEC_FREQ_16000,TLK_CODEC_2_CHANNEL,...);
tlk_codec_config(TLK_CODEC_INPUT,TLK_CODEC_FREQ_48000,TLK_CODEC_2_CHANNEL,...);
```

- **Step6**: After the parameter configuration is completed, click compile to generate the target firmware.

2.1.3 Firmware Compilation - Unicast Server

You can skip this step if there is already firmware.

- **Step1**: Select TLSR9517CDK56D for hardware.

```
#define TLSR9517CDK56D                          1
#define TLSR9518ADK80D                          2
#define HARDWARE_TYPE                           TLSR9517CDK56D
```

- **Step2**: Select TWS for mode.

```
#define APP_SCENE_TWS                           0
#define APP_SCENE_HEADSET_EP1_MULTIPLEXING      1
#define APP_SCENE                               APP_SCENE_TWS
```

- **Step3**: For TWS, select left or right ears, 1 for the left ear and 2 for the right ear.

```
#define CISP_SET_MEMBER_RANK_ID  1 // 1/L OR 2/R
```

- **Step4**: After the parameter configuration is completed, click compile to generate the target firmware.

Note:

> The unicast server for TWS scenarios supports most audio parameters, such as uplink and downlink sampling rates of 16kHz/24kHz/32kHz/48kHz, 7.5ms frames and 10ms frames. Unicast server determines the audio parameters during protocol interaction and dynamically configures the local codec according to the audio parameters.

2.1.4 Operating Steps

Figure 2.3: Unicast scenario 1 environment setup

- **Step1:** Burn the unicast client firmware into a TLSR9518 dongle and connect it to the phone/PC via USB.
- **Step2:** Burn unicast server firmware into two TLSR9517C audio boards (select left and right ear respectively), and power them via USB.
- **Step3:** Press SW2 button on the dongle to pair, the dongle connects to one unicast server device, then it will automatically search and connect to another unicast server via the coordinated set CSIP protocol.
- **Step4:** Play music or open the recording on the phone/PC to test the audio uplink and downlink channel.

Note:

> Before the firmware is burned, please erase the whole flash (for B91 chip, it can be selected to erase 2048KB).

2.1.5 TLSR9518 Dongle UI

- Press SW2 button to actively search for unicast server device and pair it.
- Press SW7 button to unpair (need to be in connected state).

- LED red breathing light, flashing once every second.
- LED blue indicates that dongle has established a Bluetooth ACL connection with the server.
- LED green indicates that dongle has established a Bluetooth CIS audio channel with server.

2.1.6 TLSR9517C Audio Board UI

- SW8, SW9, SW10 are customized by users.
- SW11 indicates chip reset.
- LED green breathing light, flashing once every second.
- LED red indicates that server has established a Bluetooth ACL connection with the client.
- LED white indicates that server has established a Bluetooth CIS audio channel with client.

2.2 Unicast Demo Scenario 2

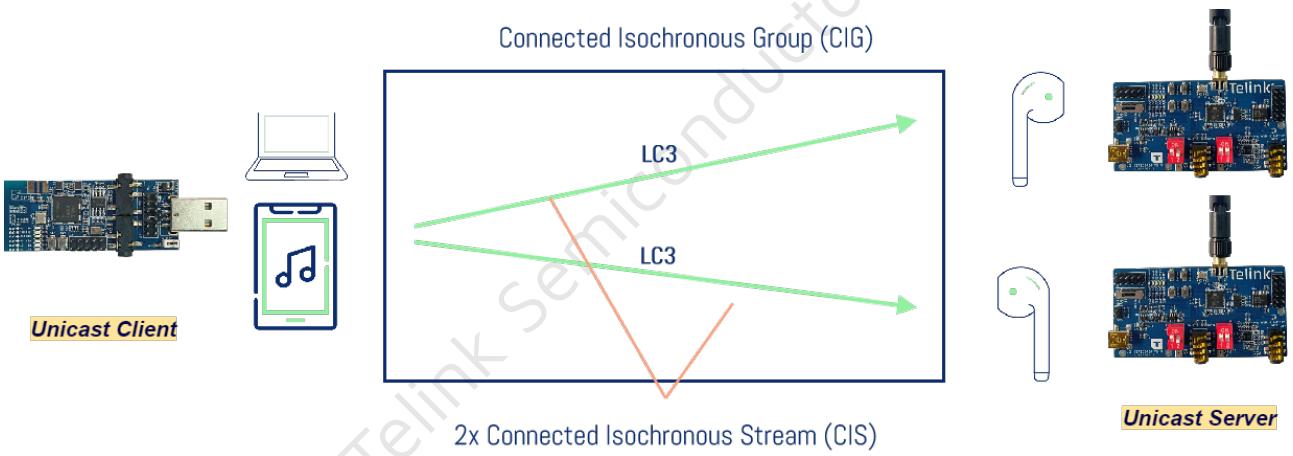

Figure 2.4: Unicast scenario 2

One unicast client to two unicast servers, single downlink, dual channel 48kHz, as shown above.

Development board for the kit:

Table 2.2: Unicast Demo Scenario 2 kit development boards

Category	IC part number	Development board external name	Quantity	Kit accessories
Unicast Server	9517C	B91 audio development board	2	USB cable (power) x2, Antenna x2
Unicast Client	9518A	B91 Dongle	1	-

2.2.1 Scenario Introduction

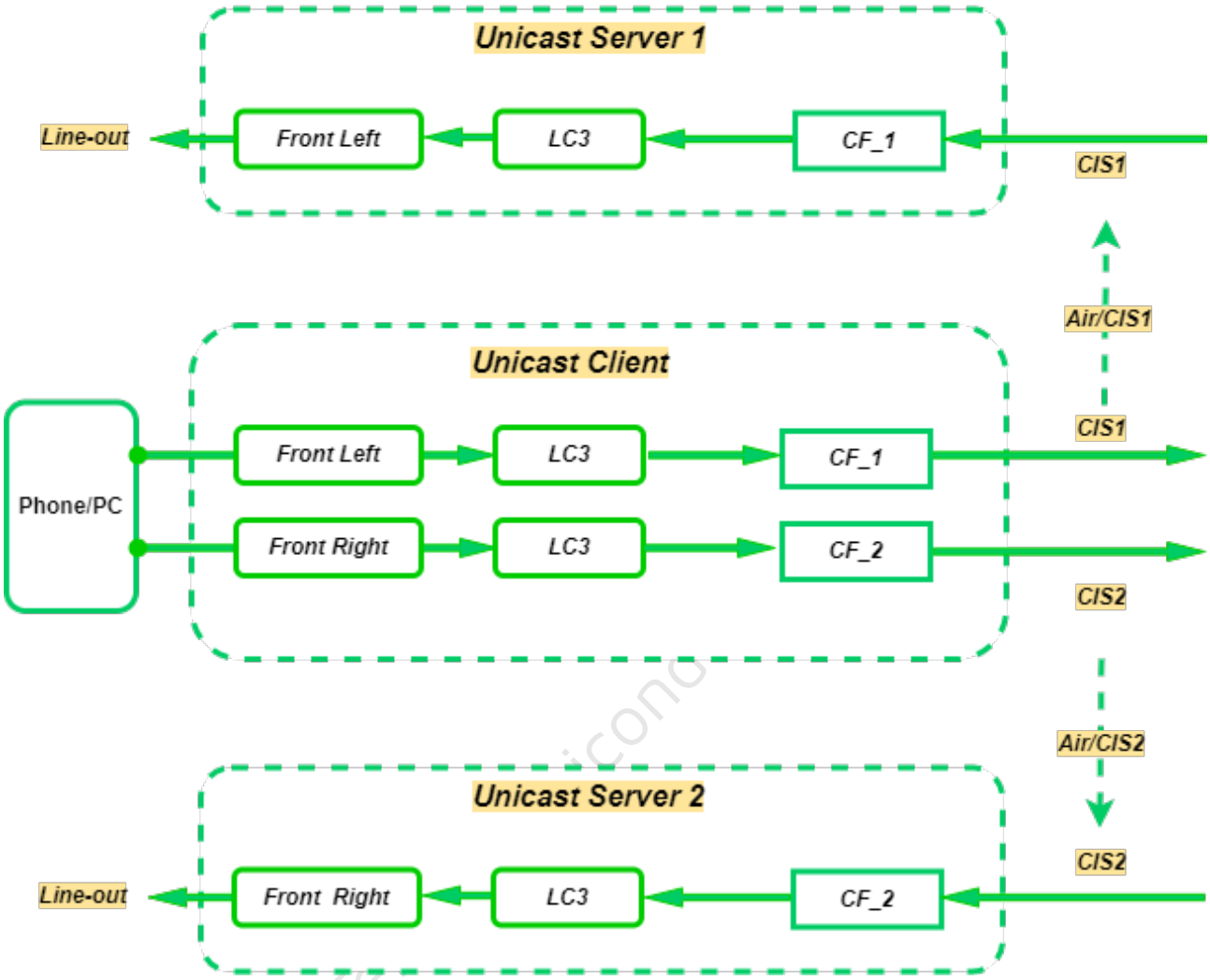

Figure 2.5: Unicast scenario 2 structure

As a unicast client, B91 dongle establishes audio channels with two unicast servers respectively.

- As the USB device, the B91 dongle acquires 48kHz dual channel audio data from the USB host (phone/PC connected to dongle), and sends the left and right channels to the two unicast servers respectively after LC3 compression.
- After the two unicast servers obtain the audio data from the audio channel established with the unicast client, the mono data is played through the local Line-out.

2.2.2 Firmware Compilation - Unicast Client

You can skip this step if there is already firmware.

- **Step1:** Select TLSR9518DONGLE for hardware.

```
#define TLSR9518EVK                             1
#define TLSR9518DONGLE                          2
#define HARDWARE_TYPE                           TLSR9518DONGLE
```

- **Step2**: Select TWS for mode.

```
#define APP_SCENE_TWS                           0
#define APP_SCENE_HEADSET                       1
#define APP_AUDIO_SCENE                         APP_SCENE_TWS
```

- **Step3**: Select audio scenario.

```
#define APP_AUDIO_CONFIGURATION_PREFER
↪  BLC_AUDIO_6II_SVR_2_SINK_2_CHN_1_SRC_N_CHN_N_CISES_2_STREAMS_2
```

For details of the audio scenarios, please refer to **BAP_v1.0.1, Table4.1: Unicast LC3 Audio Configurations**.

- **Step4**: Select audio parameters, downlink 48kHz.

```
#define APP_AUDIO_CODEC_INPUT_PARAMETER_PREFER
↪  BLC_AUDIO_STD_FREQ_48K_DURATION_10MS_FRAME_100BYTES
```

- **Step5**: If using USB mode, you need to modify the USB audio enumeration parameters, speaker downlink dual channel 48kHz.

```
#define USB_SPEAKER_ENABLE      1
#define SPK_RESOLUTION_BIT      16
#define SPK_SAMPLE_RATE         48000
#define SPK_CHANNEL_COUNT       2
#define USB_MIC_ENABLE          0
```

If using codec mode, the codec parameters need to be modified.

```
tlk_codec_config(TLK_CODEC_INPUT,TLK_CODEC_FREQ_48000,TLK_CODEC_2_CHANNEL,...);
```

- **Step6**: After the parameter configuration is completed, click compile to generate the target firmware.

2.2.3 Firmware Compilation - Unicast Server

You can skip this step if there is already firmware.

- **Step1**: Select TLSR9517CDK56D for hardware.

```
#define TLSR9517CDK56D                          1
#define TLSR9518ADK80D                          2
#define HARDWARE_TYPE                           TLSR9517CDK56D
```

- **Step2**: Select TWS for mode.

```
#define APP_SCENE_TWS                           0
#define APP_SCENE_HEADSET_EP1_MULTIPLEXING      1
#define APP_SCENE                               APP_SCENE_TWS
```

- **Step3**: For TWS, select left or right ears, 1 for the left ear and 2 for the right ear.

```
#define CISP_SET_MEMBER_RANK_ID         1 //1/L OR 2/R
```

- **Step4**: After the parameter configuration is completed, click compile to generate the target firmware.

Note:

> The unicast server for TWS scenarios supports most audio parameters, such as uplink and downlink sampling rates of 16kHz/24kHz/32kHz/48kHz, 7.5ms frames and 10ms frames. Unicast server determines the audio parameters during protocol interaction and dynamically configures the local codec according to the audio parameters.

2.2.4 Operating Steps

Figure 2.6: Unicast scenario 2 environment setup

- **Step1**: Burn the unicast client firmware into a TLSR9518 dongle and connect it to the phone/PC via USB.
- **Step2**: Burn unicast server firmware into two TLSR9517C audio boards (select left and right ear respectively), and power them via USB.
- **Step3**: Press SW2 button on the dongle to pair, the dongle connects to one unicast server device, then it will automatically search and connect to another unicast server via the coordinated set CSIP protocol.
- **Step4**: Play music or open the recording on the phone/PC to test the audio uplink and downlink channel.

Note:

> Before the firmware is burned, please erase the whole flash (for B91 chip, it can be selected to erase 2048KB).

2.2.5 TLSR9518 Dongle UI

- Press SW2 button to actively search for unicast server device and pair it.
- Press SW7 button to unpair (need to be in connected state).
- LED red breathing light, flashing once every second.
- LED blue indicates that dongle has established a Bluetooth ACL connection with the server.
- LED green indicates that dongle has established a Bluetooth CIS audio channel with server.

2.2.6 TLSR9517C Audio Board UI

- SW8, SW9, SW10 are customized by users.
- SW11 indicates chip reset.
- LED green breathing light, flashing once every second.
- LED red indicates that server has established a Bluetooth ACL connection with the client.
- LED white indicates that server has established a Bluetooth CIS audio channel with client.

2.3 Unicast Demo Scenario 3

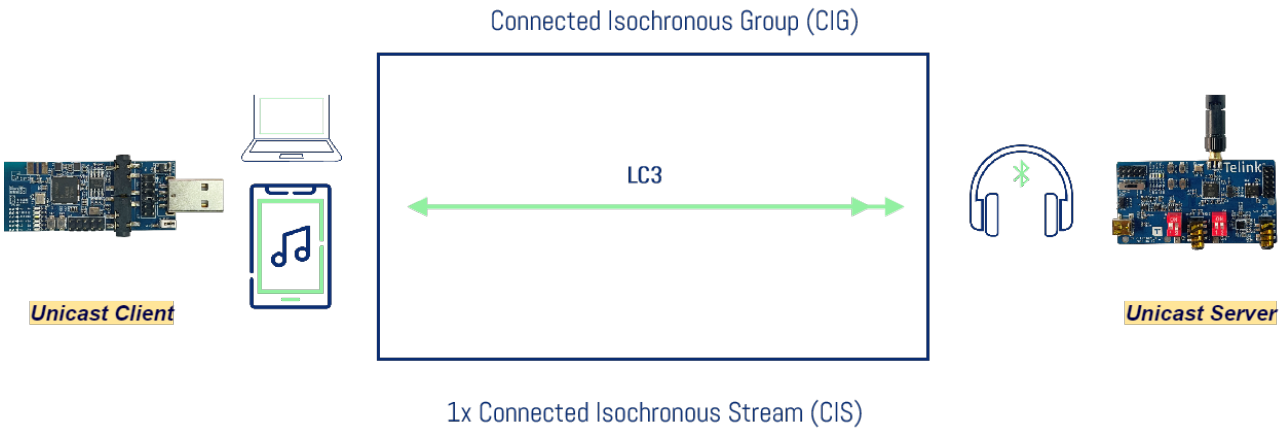

Figure 2.7: Unicast scenario 3

One unicast client to one unicast server, with a downlink 48kHz dual channel and multiplexing technology. One CIS channel transmits two audio data channels, and uplink 16kHz mono channel, simulating headset scenarios, as shown above.

Development board for the kit:

Table 2.3: Unicast Demo Scenario 3 kit development boards

Category	IC part number	Development board external name	Quantity	Kit accessories
Unicast Server	9517C	B91 audio development board	1	USB cable (power) x1, Antenna x1
Unicast Client	9518A	B91 Dongle	1	-

2.3.1 Scenario Introduction

As a unicast client, B91 dongle establishes audio channels with one unicast server.

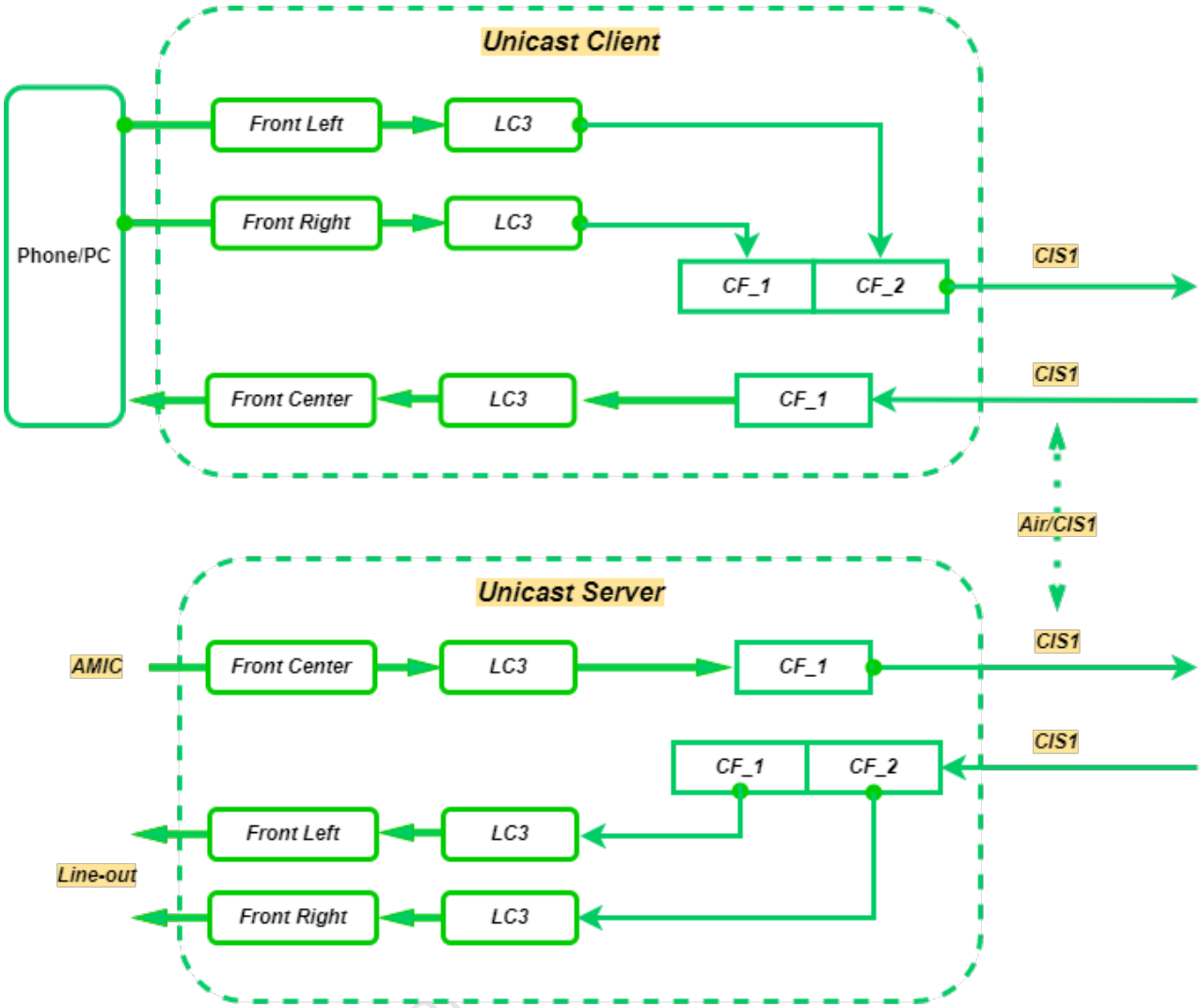

Figure 2.8: Unicast Scenario 3 structure

- As the USB device, the B91 dongle acquires 48kHz dual channel audio data from the USB host (phone/PC connected to dongle). After LC3 compression, the left and right channels are merged into one SDU and sent to unicast server.
- After the unicast server obtains the audio data of two channels from the audio channel established with the unicast client, and then decodes them through LC3 and played through the local Line-out channel.
- The unicast server capture the 16kHz mono channel ambient sound data locally via AMIC and send it to the unicast client via the audio channel established with the unicast client.
- As a USB Mic device, B91 dongle receives the 16kHz mono channel data transmitted by unicast server and sends it to USB host.

2.3.2 Firmware Compilation - Unicast Client

You can skip this step if there is already firmware.

- **Step1**: Select TLSR9518DONGLE for hardware.

```
#define TLSR9518EVK                             1
#define TLSR9518DONGLE                          2
#define HARDWARE_TYPE                           TLSR9518DONGLE
```

- **Step2**: Select TWS for mode.

```
#define APP_SCENE_TWS                           0
#define APP_SCENE_HEADSET                       1
#define APP_AUDIO_SCENE                         APP_SCENE_HEADSET
```

- **Step3**: Select audio scenario.

```
#define APP_AUDIO_CONFIGURATION_PREFER
↪  BLC_AUDIO_5_SVR_1_SINK_1_CHN_2_SRC_1_CHN_1_CISES_1_STREAMS_2
```

For details of the audio scenarios, please refer to **BAP_v1.0.1, Table4.1: Unicast LC3 Audio Configurations**.

- **Step4**: Select audio parameters, downlink 48kHz.

```
#define APP_AUDIO_CODEC_INPUT_PARAMETER_PREFER
↪  BLC_AUDIO_STD_FREQ_48K_DURATION_10MS_FRAME_100BYTES
#define APP_AUDIO_CODEC_OUTPUT_PARAMETER_PREFER
↪  BLC_AUDIO_STD_FREQ_16K_DURATION_10MS_FRAME_40BYTES
```

- **Step5**: If using USB mode, you need to modify the USB audio enumeration parameters, speaker downlink dual channel 48kHz, Mic uplink mono channel 16kHz.

```
#define USB_SPEAKER_ENABLE      1
#define SPK_RESOLUTION_BIT      16
#define SPK_SAMPLE_RATE         48000
#define SPK_CHANNEL_COUNT       2

#define USB_MIC_ENABLE          1
#define MIC_RESOLUTION_BIT      16
#define MIC_SAMPLE_RATE         16000
#define MIC_CHANNEL_COUNT       1
```

If using codec mode, the codec parameters need to be modified, input dual channel 48kHz, output mono channel 16kHz.

```
tlk_codec_config(TLK_CODEC_INPUT,TLK_CODEC_FREQ_48000,TLK_CODEC_2_CHANNEL,...);
tlk_codec_config(TLK_CODEC_OUTPUT,TLK_CODEC_FREQ_16000,TLK_CODEC_1_CHANNEL,...);
```

- **Step6**: After the parameter configuration is completed, click compile to generate the target firmware.

2.3.3 Firmware Compilation - Unicast Server

You can skip this step if there is already firmware.

- **Step1**: Select TLSR9517CDK56D for hardware.

```
#define TLSR9517CDK56D                          1
#define TLSR9518ADK80D                          2
#define HARDWARE_TYPE                           TLSR9517CDK56D
```

- **Step2**: Select headset for mode.

```
#define APP_SCENE_TWS                           0
#define APP_SCENE_HEADSET_EP1_MULTIPLEXING      1
#define APP_SCENE                               APP_SCENE_HEADSET_EP1_MULTIPLEXING
```

- **Step3**: After the parameter configuration is completed, click compile to generate the target firmware.

Note:

> The headset scenario has only one device, and there is no distinction between the left ear and the right ear.

2.3.4 Operating Steps

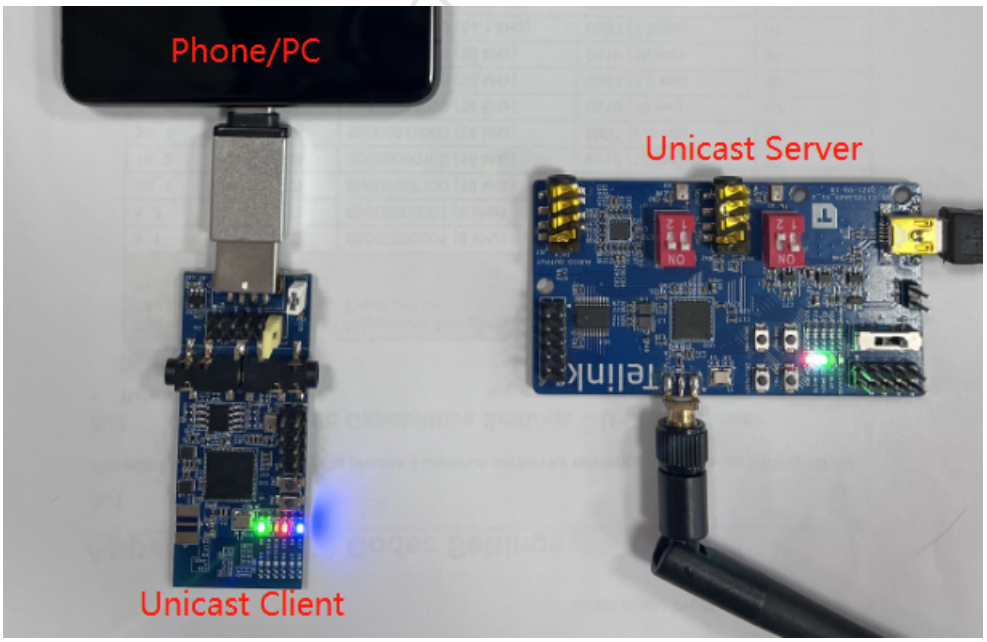

Figure 2.9: Unicast scenario 3 environment setup

- **Step1**: Burn the unicast client firmware into a TLSR9518 dongle and connect it to the phone/PC via USB.

- **Step2:** Burn unicast server firmware into a TLSR9517C audio board, and power it via USB.
- **Step3:** Press SW2 button on the dongle to pair, and the dongle connects to the target unicast server.
- **Step4:** Play music or open the recording on the phone/PC to test the audio uplink and downlink channel.

Note:

> Before the firmware is burned, please erase the whole flash (for B91 chip, it can be selected to erase 2048KB).

2.3.5 TLSR9518 Dongle UI

- Press SW2 button to actively search for unicast server device and pair it.
- Press SW7 button to unpair (need to be in connected state).
- LED red breathing light, flashing once every second.
- LED blue indicates that dongle has established a Bluetooth ACL connection with the server.
- LED green indicates that dongle has established a Bluetooth CIS audio channel with server.

2.3.6 TLSR9517C Audio Board UI

- SW8, SW9, SW10 are customized by users.
- SW11 indicates chip reset.
- LED green breathing light, flashing once every second.
- LED red indicates that server has established a Bluetooth ACL connection with the client.
- LED white indicates that server has established a Bluetooth CIS audio channel with client.

3 Broadcast Audio Demo

LE Audio introduces broadcast audio to Bluetooth technology, a technology that takes audio communication beyond peer-to-peer communication and enables an audio source device to broadcast audio stream to an unlimited number of nearby Bluetooth audio receiving devices. Broadcast audio is Bluetooth broadcast audio that conforms to a set of prescribed configurations that enable one-to-many audio broadcasting and sharing via Bluetooth technology, providing consumers with a new, globally interoperable audio experience.

The Bluetooth SIG has issued a new trademark for the Bluetooth broadcast audio feature, officially naming the audio sharing feature (part of the Bluetooth LE Audio technology) broadcast audio. The SIG has also issued a trademark guideline document, "Brand Guide for Bluetooth_Jun2022", which requires Bluetooth members to meet the technical requirements of the product must pass the "Public Broadcast Profile (PBP) specification" test, complete the Bluetooth qualification before the use of trademarks, which further ensures the unified interoperability.

A typical scenario of broadcast audio:

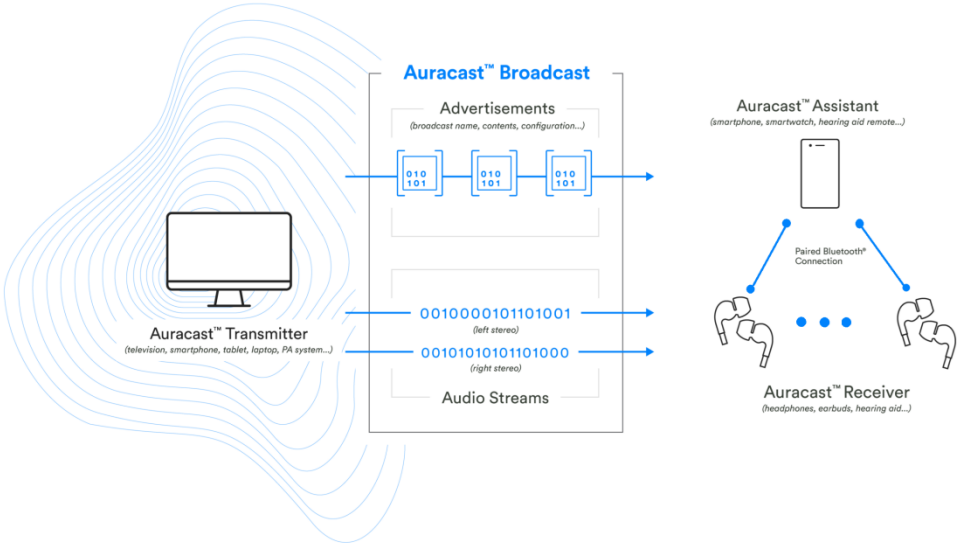

Figure 3.1: Working principle of broadcast

The broadcast audio stream from the broadcast source (e.g., smartphone, TV, etc.) and related receiving information include whether the transmission requires secure encryption, the name of the broadcast, the audio encoding method, and other information.

The receivers (e.g., headphones, earbuds, hearing aids, etc.) can search for various broadcast sources to send audio streams as well as receive information and receive transmissions.

The control assistant searches for Auracast broadcast and provides the user a user interface (UI) where they can choose to connect to Auracast broadcast. In practice, the broadcast control assistant can be a device like a smartwatch or a separate remote control, the purpose of which is to make it easier and faster for the end user to search for or select broadcast audio. After selecting the transmission on the control assistant, the control assistant will notify the receiver (e.g., headphones, earbuds, hearing aids, etc.) how to access the transmission, thus eliminating the process for the receiver to search or select. This is important for receiver devices (e.g. earbuds, hearing aids, etc.) that have a simple control interface.

3.1 Broadcast Demo Scenario 1

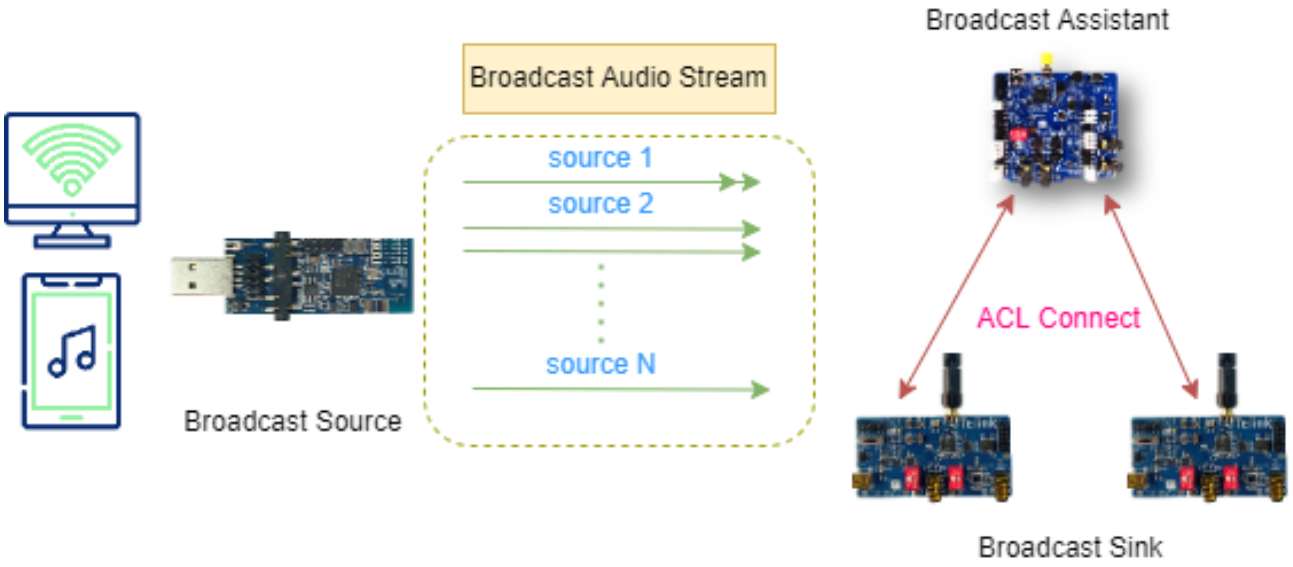

Figure 3.2: Broadcast audio scenario 1

The broadcast sink synchronizes the source information through the asynchronous connection of the broadcast assistant.

Development board for the kit:

Table 3.1: Broadcast demo scenario 1 kit development boards

Category	IC part number	Development board external name	Quantity	Kit accessories
Broadcast Sink	9517C	B91 audio development board	2	USB cable (power) x2, Antenna x2
Broadcast Source	9518A	B91 Dongle	2	-
Broadcast Assistant	9218A	B91 EVK	1	USB cable (power) x1, Antenna x1

3.1.1 Scenario Introduction

As a broadcast source, the B91 dongle broadcasts BIG (Broadcaster Isochronous Group) audio to the air. There are N sources broadcasting BIG information in the air. Source 1 has one BIS (Broadcaster Isochronous Stream) that transmits two audio channels; source 2 has two BIS, each transmitting one audio channel; and source N has one BIS that only transmits one audio channel.

There is an asynchronous connection between broadcast assistant and broadcast sink, and assistant selects which source to synchronize for the sink through the asynchronous connection. In this scenario, the assistant

provides the USB-CDC interface for configuration instructions.

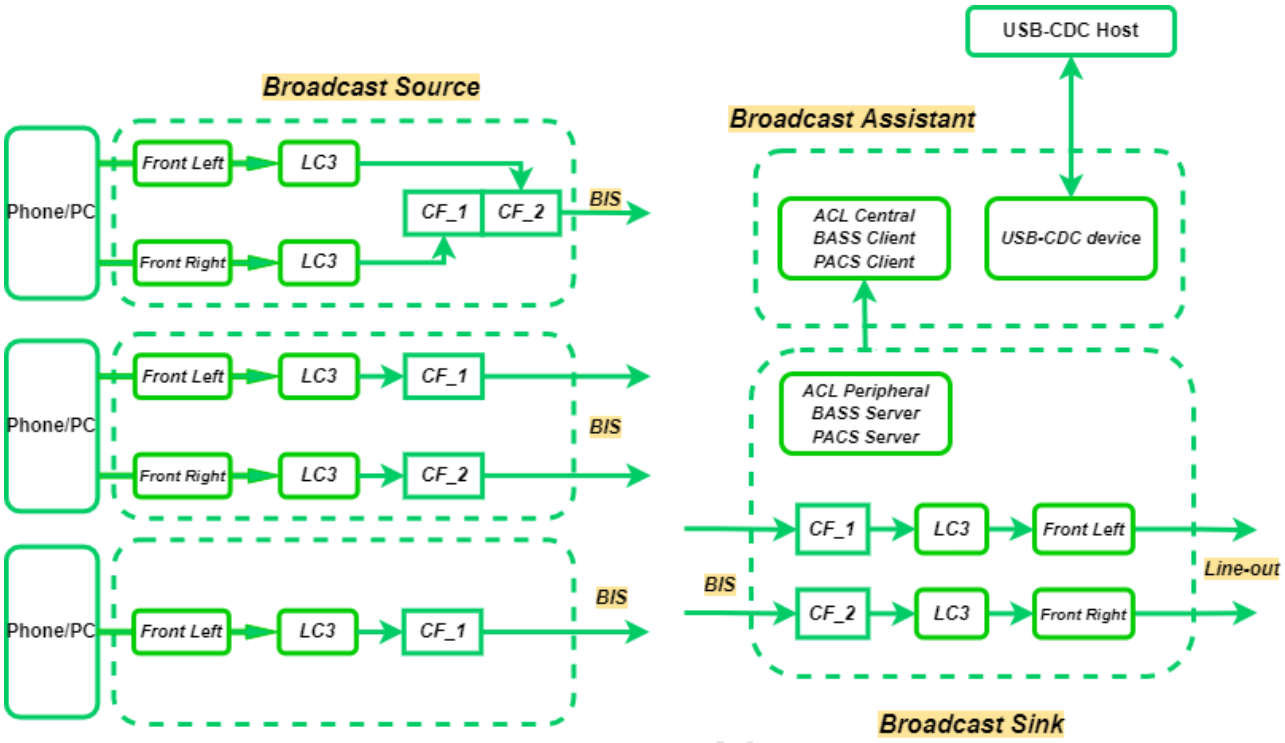

Figure 3.3: Broadcast audio scenario 1 structure

- B91 dongle, as a broadcast source device, is configured as a USB device to obtain 48kHz dual channel audio data from the USB host (phone/PC connected to dongle), and after LC3 encoding, the audio data is transmitted into the air according to different configurations.
- As a broadcast assistant device, the B91 EVK is configured as a USB-CDC device, which can communicate with the USB-CDC host (PC using serial port tool software) and receive USB instructions to perform different operations.
- The B91 audio development board acts as a broadcast sink, establishing an asynchronous connection with the broadcast assistant device, receiving different instructions from the assistant to synchronize the broadcast source in the air. After the synchronization is established, the obtained audio data will be decoded by LC3 and synchronized to the local Line-out for playback.

3.1.2 Firmware Compilation - Broadcast Source

You can skip this step if there is already firmware.

- **Step1:** Select SOURCE_ONLY_VERSION for source version.

```
#define SOURCE_ONLY_VERSION                 1
#define SOURCE_VERSION                      SOURCE_ONLY_VERSION
```

- **step2:** Configure basic information such as broadcast name, broadcast ID, device name.

```
//broadcast source extend advertising parameter
#define DEFAULT_DEV_NAME                        "Telink-BIS-SOURCE"
#define DEFAULT_BROADCAST_NAME                  "Broad-source"
#define DEFAULT_BROADCAST_ID                    0x010305
```

- **step3:** Select the audio input mode, there are four ways: USB input, Line-in, AMIC and test mode.

```
#define APP_AUDIO_INPUT_AMIC                        1
#define APP_AUDIO_INPUT_LINEIN                      2
#define APP_AUDIO_INPUT_IISIN                       4
#define APP_AUDIO_INPUT_CODEC_ENDING                4
#define APP_AUDIO_INPUT_USB_MIC                     5
#define APP_AUDIO_INPUT_NONE                        6

#define APP_AUDIO_INPUT_MODE                        APP_AUDIO_INPUT_USB_MIC
```

- **step4:** Configure the intervals for extended advertising, periodic advertising.

```
//extend advertise
#define EXT_ADV_INTERVAL                ADV_INTERVAL_400MS
//periodic advertise
#define PER_ADV_INTERVAL                PERADV_INTERVAL_800MS
```

- **step5:** Configure presentation delay, number of BIS (by default, the maximum is 2), LC3 encoding parameters, each BIS broadcast audio channel.

```
// BASE control
#define SOURCE_PRESENTATION_DELAY       10000
#define BIG_INFO_BIS_NUM                2
#define AUDIO_PARAM_LC3_CFG             LC3_CFG_48_2
#define BIS_INDEX_1_CHANNEL             BLC_AUDIO_LOCATION_FLAG_FL
#define BIS_INDEX_2_CHANNEL             BLC_AUDIO_LOCATION_FLAG_FR
```

Presentation delay is the delay that affects the playback of the sink, in units of us.

In USB audio input mode, currently only supports LC3_48_2, LC3_48_4, LC3_48_6 three configurations, more configurations will be supported in the subsequent demo. In Line-in, AMIC audio input mode, support for codec configuration LC3_8_2, LC3_16_2, LC3_32_2, LC3_48_2, LC3_48_4, LC3_48_6 six configurations, more configurations will be supported by subsequent demos.

BIS_INDEX_1_CHANNEL and BIS_INDEX_2_CHANNEL are the audio channels for BIS 1 and BIS 2, respectively, and currently only left and right are supported.

BLC_AUDIO_LOCATION_FLAG_FL indicates that only the left channel is transmitted.

BLC_AUDIO_LOCATION_FLAG_FR indicates that only the right channel is transmitted.

BLC_AUDIO_LOCATION_FLAG_FL | BLC_AUDIO_LOCATION_FLAG_FR indicates that the stereo transmitted in BIS includes both left and right channels.

- **step6:** Configure ISO interval, transport delay, and BIG encryption parameters.

```
// BIS Parameter set
#define BIG_INFO_ISO_INTERVAL           1       //iso interval = sdu interval*n
#define BIG_INFO_TRANSPORT_LATENCY      20      //unit ms
#define BIG_INFO_ENC_FLAG               0
#define BIG_INFO_BROADCAST_CODE         "Telink"
```

The macro definitions BIG_INFO_ENC_FLAG and BIG_INFO_BROADCAST_CODE can control BIG encryption. Only when BIG_INFO_ENC_FLAG is 1, the broadcast code of BIG is set to BIG_INFO_BROADCAST_CODE.

The macro definition BIG_INFO_ISO_INTERVAL indicates that the BIG is the ISO interval and the unit is the SDU interval. usually only 1, 2, 3 can be set.

The macro definition BIG_INFO_TRANSPORT_LATENCY indicates the transport delay of audio with the unit is ms, It is mainly used to control the value of the PTO.

- **step7:** After the parameter configuration is completed, click compile to generate the target firmware.

3.1.3 Firmware Compilation - Broadcast Assistant

You can skip this step if there is already firmware.

- **Step1:** Select UNIVERSAL_VERSION for assistant version.

```
#define UNIVERSAL_VERSION                   1
#define ASSISTANT_VERSION                   UNIVERSAL_VERSION
```

- **Step2:** If the UI interface is configured as APP_AUDIO_UI_USB_CDC, skip step3.

```
#define APP_AUDIO_UI_UART                       1
#define APP_AUDIO_UI_USB_CDC                    2
#define APP_AUDIO_UI_IFACE                      APP_AUDIO_UI_USB_CDC
```

- **Step3:** Configure the TX and RX pins of the UART.

```
#ifndef PARSE_CHAR_UART_TX_PIN
#define PARSE_CHAR_UART_TX_PIN                  UART1_TX_PC6
#endif

#ifndef PARSE_CHAR_UART_RX_PIN
#define PARSE_CHAR_UART_RX_PIN                  UART1_RX_PC7
#endif
```

- **Step4:** After the parameter configuration is completed, click compile to generate the target firmware.

3.1.4 Firmware Compilation - Broadcast Sink

You can skip this step if there is already firmware.

- **Step1:** Select SINK_ONLY_VERSION for sink version.

```
#define SINK_ONLY_VERSION                   1
#define SINK_VERSION                        SINK_ONLY_VERSION
```

- **Step2:** After the parameter configuration is completed, click compile to generate the target firmware.

3.1.5 Operating Steps

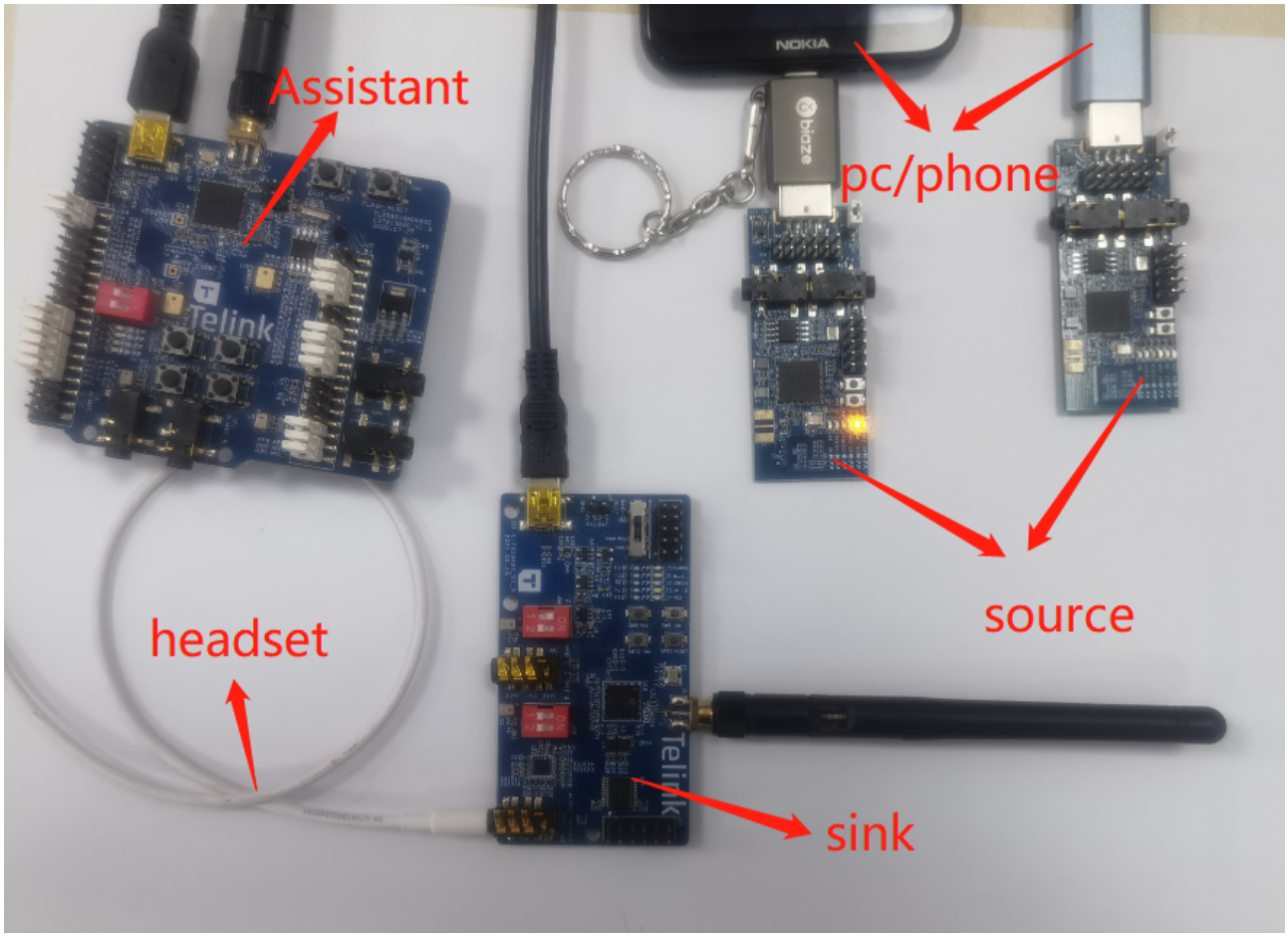

Figure 3.4: Broadcast scenario 1 environment setup

- **Step1:** Burn broadcast source firmware into two B91 dongle and connect to the phone or PC via USB.
- **Step2:** Burn the broadcast assistant firmware into a B91 EVK and connect it to the PC via USB-CDC.
- **Step3:** Burn the broadcast sink firmware into a B91 audio board, power it via USB.
- **Step4:** Follow the assistant's operation steps for pairing and connecting, broadcast synchronization, and volume adjustment.

- **Step5**: Music is played on the phone or PC, and the sink can hear the music being played.

3.1.6 Broadcast Source UI

- The yellow light flashes, once per second.
- The blue light is always on to indicate that there is audio transmission from the USB.

3.1.7 Broadcast Assistant UI

- The blue light flashes, once per second.
- The red light indicates that an asynchronous connection has been established with sink.
- Supports USB-CDC and UART command contr

3.1.8 Broadcast Sink UI

- Button SW9, volume up.
- Button SW8, volume down.
- Button SW10, mute and unmute.
- The white light flashes, once per second.
- The green light indicates that BIG synchronization has been established with the source.
- The blue light indicates that an asynchronous connection has been established with assistant.

3.2 Broadcast Demo Scenario 2

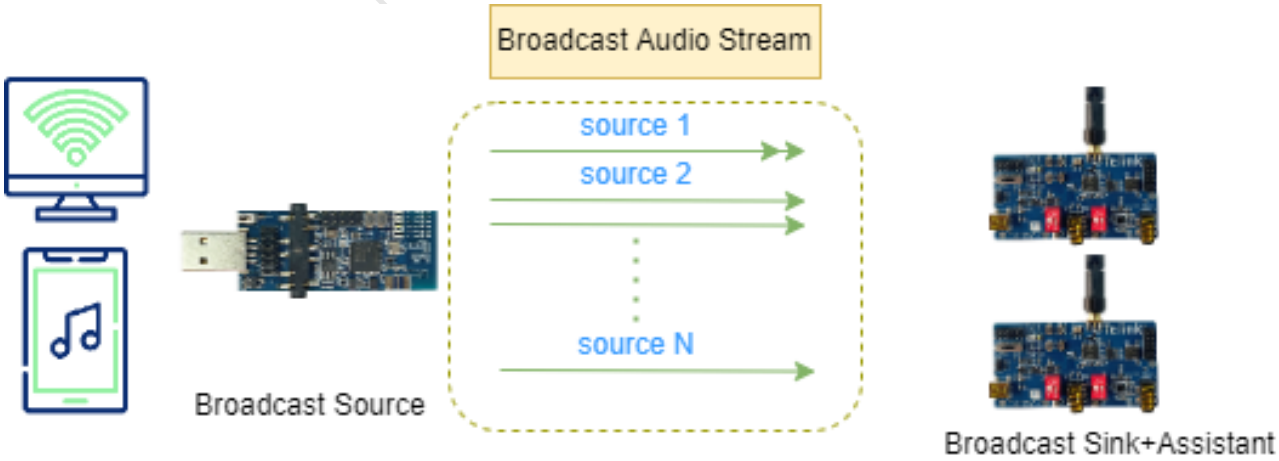

Figure 3.5: Broadcast audio scenario 2

The scenario integrates the sink and assistant functions in one demo, and the user can select to synchronize the source information directly through the buttons on the B91 audio development board.

Development board for the kit:

Table 3.2: Broadcast Demo Scenario 2 kit development boards

Category	IC part number	Development board external name	Quantity	Kit accessories
Broadcast Sink+Assistant	9517C	B91 audio development board	2	USB cable (power) x2, Antenna x2
Broadcast Source	9518A	B91 Dongle	2	-

3.2.1 Scenario Introduction

As a broadcast source, the B91 dongle broadcasts BIG audio to the air. There are N sources broadcasting BIG information in the air. Source 1 has one BIS that transmits two audio channels; source 2 has two BIS, each transmitting one audio channel; and source N has one BIS that only transmits one audio channel.

Broadcast assistant and broadcast sink are integrated in the B91 audio development board. When powered on, it will automatically scan the corresponding broadcast source information in the air, and the assistant UI is simplified to using buttons to trigger synchronization of the new source.

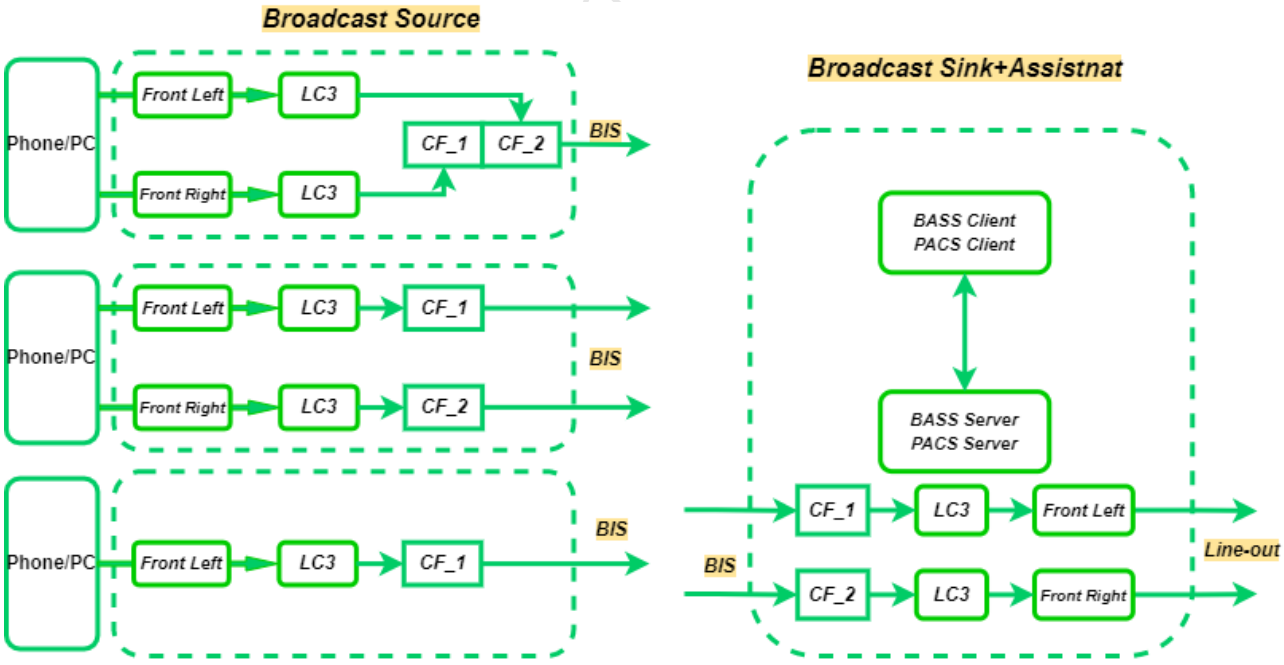

Figure 3.6: Broadcast audio scenario 2 structure

- B91 dongle, as a broadcast source device, is configured as a USB device to obtain 48kHz dual channel audio data from the USB host (phone/PC connected to dongle), and after LC3 encoding, the audio data is transmitted into the air according to different configurations.

- The B91 audio development board acts as a broadcast sink and broadcast assistant, after power on, the broadcast assistant function will automatically scan the broadcast source in the air. When the user uses the button to trigger the broadcast source synchronization, the broadcast source synchronization function of the broadcast sink is enabled. After the synchronization is established, the obtained audio data will be decoded by LC3 and synchronized to the local Line-out for playback.

3.2.2 Firmware Compilation - Broadcast Source

Refer to the broadcast source compilation process under the Broadcast Demo Scenario 1 section.

3.2.3 Firmware Compilation - Broadcast Sink+Assistant

You can skip this step if there is already firmware.

- **Step1:** Select SINK_WITH_ASSISTANT_VERSION for sink version.

```
#define SINK_WITH_ASSISTANT_VERSION          2
#define SINK_VERSION                         SINK_WITH_ASSISTANT_VERSION
```

- **Step2:** After the parameter configuration is completed, click compile to generate the target firmware.

3.2.4 Operating Steps

- **Step1:** Burn broadcast source firmware into two B91 dongle and connect to the phone or PC via USB.
- **Step2:** Burn the broadcast sink+assistant firmware into a B91 audio board, power it via USB.
- **Step3:** After pressing the button on the B91 audio board, both SW9 and SW8 can synchronize to the new source information. If there is only one source in the air, the source is repeatedly synchronized.
- **step4:** Music is played on the phone or PC, and the sink can hear the music being played.

3.2.5 Broadcast Source UI

- The yellow light flashes, once per second.
- The blue light is always on to indicate that there is audio transmission from the USB.

3.2.6 Broadcast Sink+Assistant UI

- The button SW9 to synchronize the next source.
- The button SW8 to synchronize the previous source.
- The white light flashes, once per second.
- The green light indicates that BIG synchronization has been established with the source.
- The blue light indicates that legitimate source information has been scanned.

3.3 Broadcast Demo Scenario 3

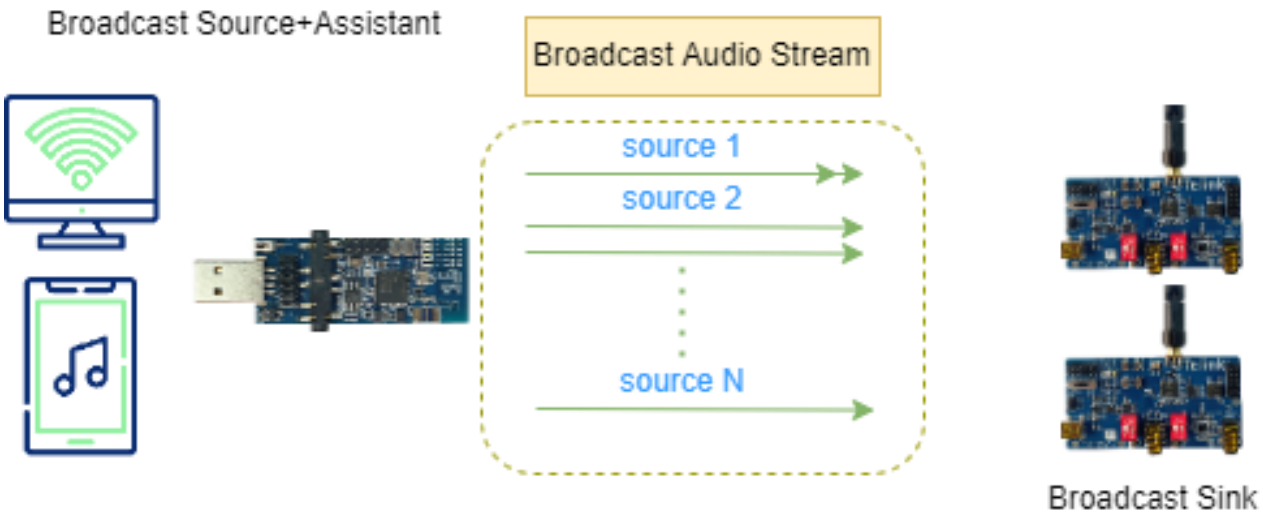

Figure 3.7: Broadcast audio scenario 3

The scenario integrates the source and assistant functions in one demo, and the user can choose which sink device to synchronize the audio through the UART interface.

Development board for the kit:

Table 3.3: Broadcast Demo Scenario 3 kit development boards

Category	IC part number	Development board external name	Quantity	Kit accessories
Broadcast Sink	9517C	B91 audio development board	2	USB cable (power) x2, Antenna x2
Broadcast Source+Assistant	9518A	B91 Dongle	2	-

3.3.1 Scenario Introduction

The B91 dongle acts as a collection of broadcast source+assistant, broadcasts BIG audio to the air. There are N sources broadcasting BIG information in the air. Source 1 has one BIS that transmits two audio channels; source 2 has two BIS, each transmitting one audio channel; and source N has one BIS that only transmits one audio channel.

There is an asynchronous connection between broadcast assistant and broadcast sink, and assistant selects which source to synchronize for the sink through the asynchronous connection. In this scenario, the assistant provides the UART interface to configure instructions.

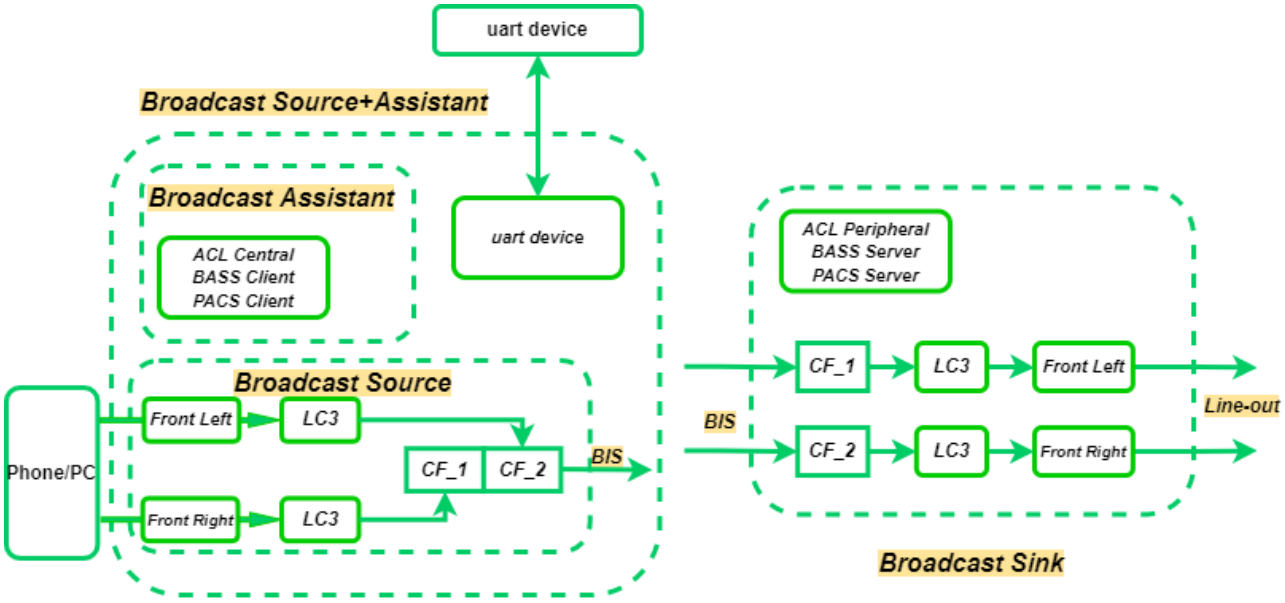

Figure 3.8: Broadcast audio scenario 3 structure

3.3.2 Firmware Compilation

This scenario is currently not supported by the SDK.

4 System Delay

Compared to Classic Bluetooth audio, BLE audio has the protocol layer advantage in terms of delay. The system delay source and composition of BLE audio typical scenarios are analyzed below.

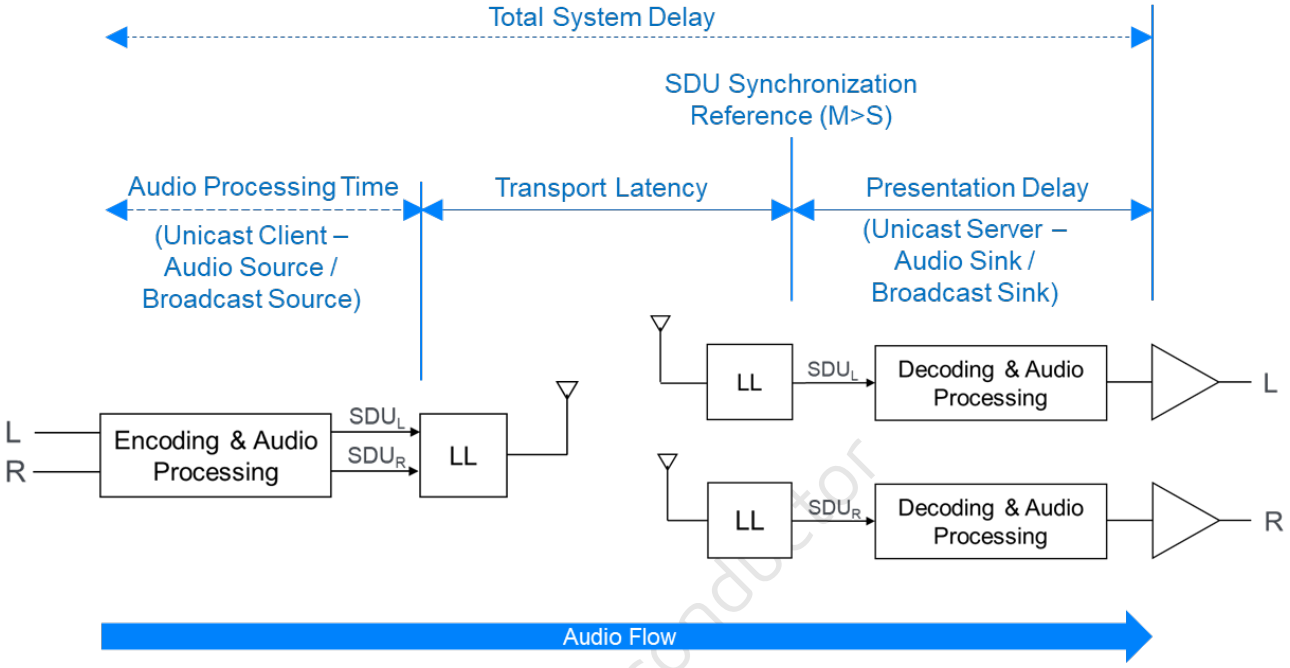

Figure 4.1: Total System Delay

It can be seen from the above figure that the total delay of the system consists of three parts:

- Audio Processing Time
- Transport Latency
- Presentation Delay

4.1 Audio Processing Time

Audio Processing Time is referred to as T_proc.

Taking the unicast scenario as an example (broadcast and unicast scenarios are basically the same), the audio processing time is mainly composed of the following parts, as shown below:

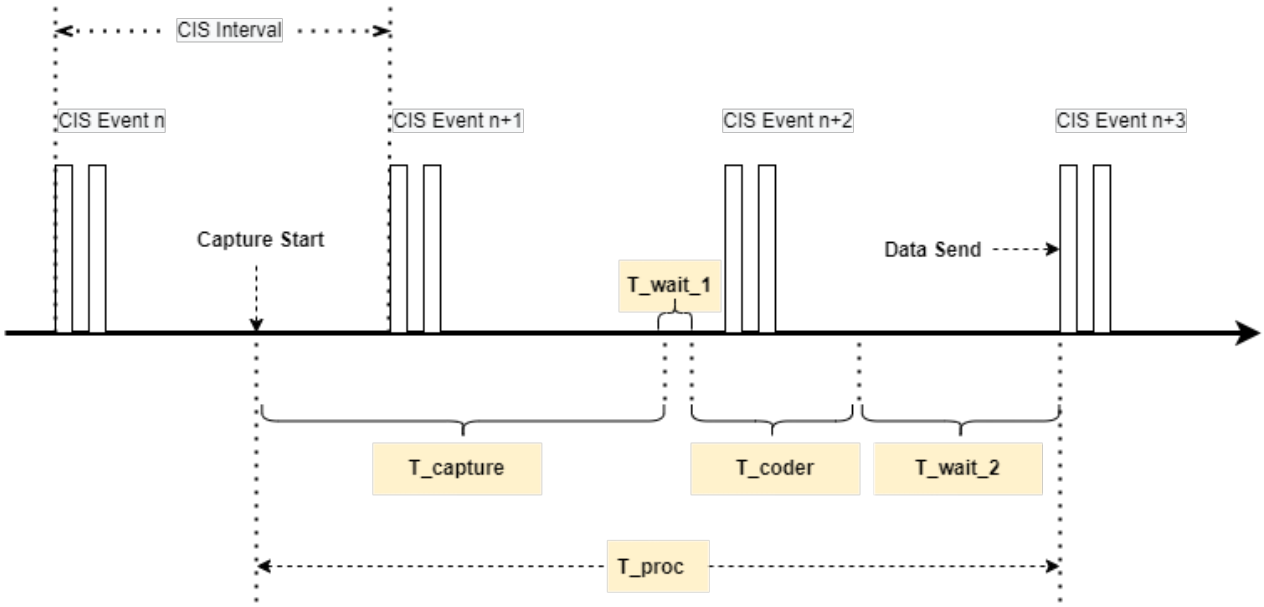

Figure 4.2: Audio_Processing_Time

- **Audio capture time: T_capture**

The audio capture time is a fixed time, if the SDU interval is 10ms, the capture time is fixed at 10ms, if the SDU interval is 7.5ms, the capture time is fixed at 7.5ms.

- **Audio codec time: T_coder**

The codec time of audio is mainly affected by two factors:

(1) The amount of data per frame

The amount of data per frame is affected by factors such as sampling rate, bit width, frame length, etc.

(2) The data processing capability of the device

Mainly reflected in the computing power of the CPU, which is affected by the device clock frequency.

- **Processing time and transmission wait time: T_wait_1 and T_wait_2**

After a frame of audio data is captured, it needs to be processed before encoding and decoding, such as left and right channels (if any) extraction, software program running time, etc., which introduces the waiting time **T_wait_1**.

After a frame of audio data is captured, it cannot be sent immediately after encoding, and it needs to wait until the next packet sending time, which introduces the waiting time **T_wait_2**.

As can be seen from the above description, the audio processing time is mainly composed of three parts:

T_proc = T_capture + T_coder + T_wait_1 + T_wait_2

- **T_capture** is the audio data capture time, fixed to SDU_Interval.
- **T_coder** is mainly affected by device performance, and strong CPU performance can shorten the time.
- **T_wait_1** and **T_wait_2** are the software running time and the logic null waiting time, with the greatest flexibility.

4.2 Transport Latency

Transport Latency is referred to as T_trans.

The audio data is sent from the link layer of the sending end to the link layer of the receiving end, and the time from sending to receiving is fixed. Taking central to peripheral and the data format is unframed as an example, the specific calculation formula is as follows:

T_trans = CIG_Sync_Delay + (FT-1)*ISO_Interval + (ISO_Interval÷SDU_Interval)×SDU_Interval

- **CIG_Sync_Delay** is a delay introduced to synchronize multiple CIS within the same CIG.
- **ISO_Interval** can be considered as a CIS Interval.
- **SDU_Interval** is the generation time of SDU, if SDU_Interval = 10ms means there is a SDU generated every 10ms.
- **FT** is a SDU that can be transmitted within a maximum of several ISO Intervals.

For the specific meanings of the above parameters, please refer to "Core_V5.4", Vol 6, Part B.

In short: Once the negotiated CIS link between CIS central and CIS peripheral is established, the transport latency between the two is determined.

4.3 Presentation Delay

After the receiver receives the SDU reported by the controller, it needs to be processed by the software, decoded by LC3, and then sent to the codec at a fixed time for playback, which introduces the presentation delay, or T_pres for short.

4.4 Audio System Delay

From the above analysis:

T_system_delay = T_proc + T_trans + T_pres

that is,

T_system_delay = T_capture + T_coder + T_wait_1 + T_wait_2 + T_trans + T_pres

- **T_capture** is the audio capture time, fixed as SDU_Interval.
- **T_coder** and **T_pres** are related to the LC3 codec time, mainly related to CPU performance.
- **T_wait_1** and **T_wait_2** are mainly related to the software architecture, a good software architecture can minimize the time.
- **T_trans** is mainly related to FT, that is, the number of retransmissions. The less ISO_Interval that one SDU is transmitted in, the shorter transport latency is.

From the above analysis, it can be seen that if the total system delay is to be reduced, the measures that can be taken are:

(1) Shortening SDU_Interval can shorten T_capture and T_trans.

(2) Using a CPU with stronger computing power can shorten T_coder and T_pres.

(3) Depending on the specific project, using an efficient software architecture can minimize T_wait_1 + T_wait_2 and T_pres.

(4) Reduce the FT, and shorten the number of retransmissions as much as possible while ensuring the quality of the wireless transmission.

5 Appendix

5.1 Assistant Command Set

All control commands for the Broadcast Assistant need to end with "\r\n" (ASCII: 0x0d 0x0A). In the following description of the commands, this will not be repeated.

5.1.1 Scan Sink

The command format for scanning sinks: scan-sink <start|stop|clear> \r\n

(1) scan-sink start

Enable the scan sink function, the assistant will first reply with "assistant start scan sink".

Then if the Sink device is scanned, it will be reported in the following format.

[dev_idx] <public/random> xx:xx:xx:xx:xx:xx name: complete name

The following is an example of the scan.

[1] public 12:34:56:00:00:00 name:Telink-BIS-SINK

Where dev_idx needs to be filled in when connecting behind.

(2) scan-sink stop

Stopping the scan sink function, the assistant will first reply with "assistant stop scan sink".

(3) scan-sink clear

Stopping the scan sink function and clearing the sink information that has been scanned, the assistant will first reply with "assistant clear sink info".

5.1.2 Connect Specific Sink Devices

The command format for connecting specific sink devices: conn-sink <dev_idx> \r \n

where dev_idx is the index value reported after the scan-sink start command.

conn-sink 1

If the index value does not exist, the helper will reply:

```
connect sink index error
show sink to view sink index.
```

If the index value exists, the assistant will create an ACL connection with the corresponding sink device. And reply with "assistant start connect sink".

During the connection process, the assistant will report some intermediate status based on the profile's state. This mainly includes the MAC address of the ACL-connected device, the ACL handle for the connection, the status of the BASS, PACS, and VCP services, and the flag indicating the end of SDP.

```
acl connected Handle:80, Addr 12:34:56:00:00:00
ConnHandle:80, PACS Found Start.
ConnHandle:80, PACS Found End.
ConnHandle:80, BASS Found Start.
Sink no any Sync Source Info
ConnHandle:80, BASS Found End.
ConnHandle:80, VCP Volume Controller Found Start.
ConnHandle:80, VCP Volume Controller Found End.
ConnHandle:80, SDP over
```

5.1.3 Scan Source Information for Connected Sinks

The command format for scanning source information for connected sinks: scan-bcast <start|stop|clear> <conn_idx> \r\n

Where conn_idx currently defaults to 1, users can query the index according to the "show conn" command.

(1) scan-bcast start <conn_idx>

Starting the scan source information operation.

If the conn_idx index is abnormal, the assistant will reply with "index error.You can run the"show conn" command to view connection information".

If the assistant has not completed the SDP process, it will reply with "wait sdp discovery ending".

If the connected Sink device does not have the BASS service, the assistant will reply with "connect device not supported BASS Server".

If the connected Sink device does not have the PACS service, the assistant will reply with "connect device not supported PACS Server".

If everything is normal, the assistant will reply with "assistant start scan broadcast source".

When the assistant scans a matching source information, it will report the information. The example below shows the format, where 1 in [1] represents the source_idx that will be used for subsequent source additions. The LC3 codec ID is 0600000000, indicating whether the source is encrypted, and providing audio sample rate and channel information for each BIS.

```
Found Source Info[1]
    public Address is 12:34:56:00:00:00
    Device Name:Telink-BIS-SOURCE
    Broadcast Name:Broad-source
        BIS Index: 1
        Codec ID: 0600000000
        Sampling Frequency: 48kHz
        Front Left: Supported
        BIS Index: 2
        Codec ID: 0600000000
        Sampling Frequency: 48kHz
```

```
        Front Right: Supported
    Source is Unencrypted
```

(2) scan-bcast stop <conn_idx>

Stopping the scan source information operation, the assistant will first reply with "assistant stop scan broadcast source".

(3) scan-bcast clear <conn_idx>

Stopping the scan source information operation and clearing the source information that has been scanned, the assistant will first reply with "assistant clear broadcast source info".

5.1.4 Add Source to Connected Sink

The command format for adding sources to connected sink: add-source <conn_idx> <source_idx> <bis_sync> [broadcast_key]\r\n

Where conn_idx currently defaults to 1, users can query the index according to the "show conn" command.

The source_idx is reported during the scan source operation, and users can also query the index using the "show source" command.

The bis_sync represents the BIS synchronization information of the source, which is reported during the scan source operation. To synchronize BIS index 1, bis_sync = BIT(0); to synchronize BIS index 2, bis_sync = BIT(1); to synchronize BIS index 1 and 2, bis_sync = BITS(0, 1).

If the counterpart already has synchronized source information, the command will first remove the existing source information before adding new source information.

An example command for adding a source is: add-source 1 1 3

The assistant will report the periodic advertising synchronization status of the sink as well as the BIS synchronization status.

```
started add source
sink PA state is Sync
BIS Synced state is 0x00000000
sink PA state is Sync
BIS Synced state is 0x00000003
```

5.1.5 Query Information Command Set

The command format for querying information: show <conn|sink|source|vcp> [conn_dev]\r\n

(1) show conn

Querying the connected sink information, the assistant replies with "connect sink info" and prints the information of the connected sink.

[1]Connect:public 12:34:56:00:00:00 name:Telink-BIS-SINK

(2) show sink

Querying the scanned sink information, the assistant replies with "scan sink info" and prints the information of the scanned sink.

[1] public 12:34:56:00:00:00 name:Telink-BIS-SINK

(3) show source

Querying the scanned source information, the assistant replies and prints the source information.

```
Connected Index:1 is scanning source info
Found Source Info[1]
    public Address is A4:C1:38:10:00:1F
    Device Name:Telink-BIS-SOURCE
    Broadcast Name:Broad-source
        BIS Index: 1
        Codec ID: 0600000000
        Sampling Frequency: 48Hz
        Front Left: Supported
        BIS Index: 2
        Codec ID: 0600000000
        Sampling Frequency: 48Hz
        Front Right: Supported
    Source is Unencrypted
```

(4) show vcp [conn_dev]

Querying the volume control protocol information of the connected sink device, including VCS and VOCS information. If the device has VCS information, volume control is possible; if it has VOCS information, volume offset control for each output is possible.

```
Remote Volume Control State, connHandle: 0x80
volume(min:0, max:255) is 20, muteSate:Unmute
VOCS index is 0
Location:Front Left
Audio Description is Telink BIS Left Output
Volume Offset(min:-255, max:255) is 0
VOCS index is 1
Location:Front Right
Audio Description is Telink BIS Rigth Output
Volume Offset(min:-255, max:255) is 0
```

5.1.6 Volume Control Commands

The volume control instructions are divided into four operations: mute/unmute, volume+, volume-, and set absolute volume.

For all volume control instructions, after the operation is completed or the sink's own volume variable is updated, the volume value and mute flag information of the sink will be printed.

Conn:80, volume(min:0, max:255) is 40, muteSate:Unmute

(1) mute/unmute

The command format: mute <conn_idx>

The instructions by default will toggle between muting and unmuting.

(2) volume+

The command format: vol+ <conn_idx>

By default, a volume+ command will be sent. The exact amount by which the volume is increased depends on the settings of the sink.

(3) volume-

The command format: vol- <conn_idx>

By default, a volume- command will be sent. The exact amount by which the volume is reduced depends on the settings of the sink.

(4) Set absolute volume

The command format: set-vol <conn_idx> <volume value>

By default, the volume value is set to the volume of the sink, ranging from 0 to 255.

5.1.7 Volume Offset Control

If the remote Sink device has VOCS support, users can set a separate output volume offset. The command format is as follows:

set-vol-offset <conn_idx> <vocs_idx> <vol_offset>

vocs_idx can be queried using show vcp <conn_idx>.

The vol_offset offset value is returned as -255 to 255.

Both setting the volume offset and modifying the volume offset on the sink side will be printed.

Conn:80, vocs index is 1 volume offset(min:-255, max:255) is -60

5.1.8 Examples

This section takes one assistant, two sinks and two sources as an example to explain how to configure different source information for the two sinks.

Basic information of the sinks:

sink 1:

- public address: 12:34:56:00:00:00
- Device name: Telink-BIS-SINK

sink 2:

- public address: 12:34:56:00:00:01
- Device name: Telink-BIS-SINK

Basic information of the source:

source 1:

- public address: 12:34:56:00:01:00
- Device name: Telink-BIS-SOURCE

source 2:

- public address: 12:34:56:00:01:01
- Device name: Telink-BIS-SOURCE

(1) Scan the sink device in the air

The following displays the process of sink scanning. First, send the "scan-sink start" command. When the desired sink device is detected, send the "scan-sink stop" command to stop the current scan. You can use the "show sink" command to print the information of the currently scanned sinks.

```
[10:22:41.371]send: scan-sink start
[10:22:41.372]recv: assistant start scan sink
[1] public 12:34:56:00:00:00 name:Telink-BIS-SINK
[2] public 12:34:56:00:00:01 name:Telink-BIS-SINK

[10:22:44.061]send: scan-sink stop
[10:22:44.063]recv: assistant stop scan sink

[10:22:50.587]send: show sink
[10:22:50.589]recv: scan sink info
Now Stop Scan new Sink
[1] public 12:34:56:00:00:00 name:Telink-BIS-SINK
[2] public 12:34:56:00:00:01 name:Telink-BIS-SINK
```

(2) Create ACL connection with sink 1

Based on the scanned sink information above, the dev_idx of Sink 1 is 1. Therefore, by using the "conn-sink 1" command, an ACL connection can be established with sink 1.

```
[10:27:22.053]send: conn-sink 1
[10:27:22.054]recv: assistant start connect sink
acl connected Handle:80, Addr 12:34:56:00:00:00
ConnHandle:80, PACS Found Start.
ConnHandle:80, PACS Found End.
ConnHandle:80, BASS Found Start.
sink PA state is Loss
BIS Synced state is 0x00000000
ConnHandle:80, BASS Found End.
```

```
ConnHandle:80, VCP Volume Controller Found Start.
Sink no any Sync Source Info
ConnHandle:80, VCP Volume Controller Found End.
ConnHandle:80, SDP over

[10:31:37.373]send: show conn
[10:31:37.377]recv: connect sink info
[1]Connect:public 12:34:56:00:00:00 name:Telink-BIS-SINK
```

After establishing an ACL connection, the supported service information of the sink will be printed, including PACS, BASS, and VCP information. If the BASS service is included, the periodic advertising and BIS broadcast isochronous status will be printed. In the example above, the sink does not have any synchronization information.

When the assistant reports "SDP over," it means that the service discovery process has been completed, and other operations can be proceed. There may be a significant time difference between service discovery for reconnecting and pairing. Users can use packet capture to examine this further.

(3) Scan source information for sink 1

From the above logs, we can see that the "conn_idx" of Sink 1 after connection is 1. Therefore, sending the command "scan-bcast start 1" to scan for relevant source information in the air. Once scanning a matching source information, sending the command "scan-bcast stop 1" to stop scanning for sources.

```
[10:32:39.822]send: scan-bcast start 1
[10:32:39.824]recv: assistant start scan broadcast source
[10:32:39.959]recv: Found Source Info[1]
    public Address is 12:34:56:00:01:00
    Device Name:Telink-BIS-SOURCE
    Broadcast Name:Broad-source
        BIS Index: 1
        Codec ID: 0600000000
        Sampling Frequency: 48Hz
        Front Left: Supported
        BIS Index: 2
        Codec ID: 0600000000
        Sampling Frequency: 48Hz
        Front Right: Supported
    Source is Unencrypted
[10:32:40.199]recv: Found Source Info[2]
    public Address is 12:34:56:00:01:01
    Device Name:Telink-BIS-SOURCE
    Broadcast Name:Broad-source
        BIS Index: 1
        Codec ID: 0600000000
        Sampling Frequency: 48Hz
        Front Left: Supported
```

```
        BIS Index: 2
        Codec ID: 0600000000
        Sampling Frequency: 48Hz
        Front Right: Supported
    Source is Unencrypted

[10:33:59.365]send: scan-bcast stop 1
[10:33:59.375]recv: assistant stop scan broadcast source
```

The above log shows the source information for the over-the-air compliance.

(4) Add/switch source information for sink 1

According to the previous information, the source_idx of source 1 is 1 and the source_idx of source 2 is 2, and both are unencrypted.

The following are the operation steps of the log, Users can determine whether the operations were successful based on the audio heard from the sink:

- Synchronize the left and right channel audio of Source 1.
- Synchronize the left and right channel audio of Source 2.
- Synchronize the left channel audio of Source 1.
- Synchronize the left channel audio of Source 2.
- Synchronize the right channel audio of Source 1.
- Synchronize the right channel audio of Source 2.


```
//Synchronize the left and right channel audio of Source 1;
[10:38:46.205]send: add-source 1 1 3
[10:38:46.208]recv: started add source
[10:38:46.849]recv: sink PA state is Sync
BIS Synced state is 0x00000000
[10:38:46.909]recv: sink PA state is Sync
BIS Synced state is 0x00000003
//Synchronize the left and right channel audio of Source 2;
[10:38:50.197]send: add-source 1 2 3
[10:38:50.202]recv: started add source
[10:38:50.330]recv: sink PA state is Loss
BIS Synced state is 0x00000003
sink PA state is Loss
BIS Synced state is 0x00000000
[10:38:50.449]recv: Sink no any Sync Source Info
[10:38:50.751]recv: sink PA state is Sync
BIS Synced state is 0x00000000
[10:38:50.930]recv: sink PA state is Sync
BIS Synced state is 0x00000003
//Synchronize the left channel audio of Source 1;
```

```
[10:38:59.580]send: add-source 1 1 1
[10:38:59.585]recv: started add source
[10:38:59.690]recv: sink PA state is Loss
BIS Synced state is 0x00000003
sink PA state is Loss
BIS Synced state is 0x00000000
[10:38:59.810]recv: Sink no any Sync Source Info
[10:39:00.231]recv: sink PA state is Sync
BIS Synced state is 0x00000000
[10:39:00.291]recv: sink PA state is Sync
BIS Synced state is 0x00000001
//Synchronize the left channel audio of Source 2;
[10:39:03.709]send: add-source 1 2 1
[10:39:03.714]recv: started add source
[10:39:03.830]recv: sink PA state is Loss
BIS Synced state is 0x00000001
sink PA state is Loss
BIS Synced state is 0x00000000
[10:39:03.950]recv: Sink no any Sync Source Info
[10:39:04.430]recv: sink PA state is Sync
BIS Synced state is 0x00000000
[10:39:04.551]recv: sink PA state is Sync
BIS Synced state is 0x00000001
//Synchronize the right channel audio of Source 1;
[10:39:08.837]send: add-source 1 1 2
[10:39:08.841]recv: started add source
[10:39:08.990]recv: sink PA state is Loss
BIS Synced state is 0x00000001
sink PA state is Loss
BIS Synced state is 0x00000000
[10:39:09.110]recv: Sink no any Sync Source Info
[10:39:09.531]recv: sink PA state is Sync
BIS Synced state is 0x00000000
[10:39:09.650]recv: sink PA state is Sync
BIS Synced state is 0x00000002
////Synchronize the right channel audio of Source 2;
[10:39:12.597]send: add-source 1 2 2
[10:39:12.602]recv: started add source
[10:39:12.711]recv: sink PA state is Loss
BIS Synced state is 0x00000002
sink PA state is Loss
BIS Synced state is 0x00000000
[10:39:12.890]recv: Sink no any Sync Source Info
[10:39:13.250]recv: sink PA state is Sync
BIS Synced state is 0x00000000
[10:39:13.430]recv: sink PA state is Sync
```

```
BIS Synced state is 0x00000002
```

(5) Volume Control Operation

After completing the service discovery process, the command "show vcp 1" can be sent to query the volume control protocol supported by the sink. The connected sink supports volume control, mute, left channel volume gain (not implemented at the sink's application layer), and right channel volume gain (not implemented at sink's application layer).

```
[10:41:59.686]send: show vcp 1
[10:41:59.689]recv: Remote Volume Control State, connHandle: 0x80
volume(min:0, max:255) is 20, muteSate:Unmute
VOCS index is 0
Location:Front Left
Audio Description is Telink BIS Left Output
Volume Offset(min:-255, max:255) is 0
VOCS index is 1
Location:Front Right
Audio Description is Telink BIS Rigth Output
Volume Offset(min:-255, max:255) is 0
```

When the sink jumps the volume by pressing a key, the assistant will print the current volume information.

```
Conn:80, volume(min:0, max:255) is 40, muteSate:Unmute
```

The assistant can also send volume+, volume-, mute and other commands。

```
//volume+
[10:45:10.741]send: vol+ 1
[10:45:10.857]recv: Conn:80, volume(min:0, max:255) is 60, muteSate:Unmute
[10:45:11.541]send: vol+ 1
[10:45:11.637]recv: Conn:80, volume(min:0, max:255) is 80, muteSate:Unmute
//volume-
[10:45:13.382]send: vol- 1
[10:45:13.497]recv: Conn:80, volume(min:0, max:255) is 80, muteSate:Unmute
[10:45:14.141]send: vol- 1
[10:45:14.217]recv: Conn:80, volume(min:0, max:255) is 60, muteSate:Unmute
//mute/unmute
[10:45:17.541]send: mute 1
[10:45:17.637]recv: Conn:80, volume(min:0, max:255) is 60, muteSate:Mute
[10:45:18.212]send: mute 1
[10:45:18.296]recv: Conn:80, volume(min:0, max:255) is 60, muteSate:Unmute
//set volume
[10:47:38.849]send: set-vol 1 75
[10:47:38.940]recv: Conn:80, volume(min:0, max:255) is 75, muteSate:Unmute
[10:47:43.293]send: set-vol 1 160
```

```
[10:47:43.380]recv: Conn:80, volume(min:0, max:255) is 160, muteSate:Unmute
[10:47:53.501]send: set-vol 1 8
[10:47:53.580]recv: Conn:80, volume(min:0, max:255) is 8, muteSate:Unmute
```

The sink supports the VOCS service, so the VOCS value can also be set.

```
//Set the left channel volume offset value
[10:49:12.158]send: set-vol-offset 1 0 27
[10:49:12.242]recv: Conn:80, vocs index is 0 volume offset(min:-255, max:255) is 27,
write volume offset value is success
[10:49:18.621]send: set-vol-offset 1 0 -7
[10:49:18.722]recv: Conn:80, vocs index is 0 volume offset(min:-255, max:255) is -7,
write volume offset value is success
//Set the right channel volume offset value
[10:49:31.772]send: set-vol-offset 1 1 -60
[10:49:31.862]recv: Conn:80, vocs index is 1 volume offset(min:-255, max:255) is -60
write volume offset value is success
[10:49:36.269]send: set-vol-offset 1 1 120
[10:49:36.362]recv: Conn:80, vocs index is 1 volume offset(min:-255, max:255) is 120
write volume offset value is success
////Query volume parameters
[10:49:40.133]send: show vcp 1
[10:49:40.137]recv: Remote Volume Control State, connHandle: 0x80
volume(min:0, max:255) is 8, muteSate:Unmute
VOCS index is 0
Location:Front Left
Audio Description is Telink BIS Left Output
Volume Offset(min:-255, max:255) is -7
VOCS index is 1
Location:Front Right
Audio Description is Telink BIS Rigth Output
Volume Offset(min:-255, max:255) is 120
```