Telink B91m BLE Audio SDK

用户指南

AN-22063005-C3

Telink Semiconductor

Ver2.0.0

2023.07.24

Keyword

BLE, Audio

Brief

本文档为泰凌微电子 B91m BLE Audio 典型应用场景的用户指南，适用于 B91 (TLSR951x/921x) 和 B92(TLSR952x/922x) 系列芯片。

Published by
Telink Semiconductor

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

© Telink Semiconductor
All Rights Reserved

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.

Copyright © 2023 Telink Semiconductor (Shanghai) Co., Ltd.

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

Telink Semiconductor

更改历史

版本	修改内容
V1.0.0	初次发布
V1.1.0	新增第二章和第三章
v2.0.0	内容更新，新增第四章和第五章

Telink Semiconductor

Contents

Telink Semiconductor

Telink Semiconductor

List of Figures

Telink Semiconductor

List of Tables

Telink Semiconductor

1 硬件和软件准备

1.1 硬件准备

1.1.1 Telink 单播音频套件

Figure 1.1: Telink B91 Unicast 音频套件

配套硬件：

Table 1.1: Telink 单播音频配套开发板一览表

类别	芯片料号	开发板外部名称	个数	配套附件
Unicast Server	9517C	B91 音频开发板	2	USB 线 (供电) 2 根，天线 2 个
Unicast Client	9518A	B91 Dongle	1	-

可以通过 淘宝 购买该套件，如果还未上架，请联系 Telink FAE 购买 B91 蓝牙低功耗音频套件。

1.1.2 Telink 广播音频套件

Figure 1.2: Telink B91 Broadcast 音频套件

配套硬件：

Table 1.2: Telink 广播音频配套开发板一览表

类别	芯片料号	开发板外部名称	个数	配套附件
Broadcast sink	9517C	B91 音频开发板	2	USB 线 (供电) 2 根，天线 2 个
Broadcast source	9518A	B91 Dongle	2	-
Broadcast assistant	9518A	B91 EVK	1	USB 线 (供电) 1 根，天线 1 个

可以通过 淘宝 购买该套件，如果还未上架，请联系 Telink FAE 购买 B91 蓝牙低功耗音频套件。

1.1.3 硬件设计参考

- Telink wiki 页面下 Development Kit and Application Boards 章节

1.2 软件准备

- 下载工具
- SDK(此链接为 telink 内部链接，客户需要联系 FAE 获取)

Telink Semiconductor

2 Unicast Audio Demo

Unicast Audio Demo 分为两个角色，分别为 Unicast Client 和 Unicast Server。

B91m BLE Audio SDK v1.1.0.0 扩展了 Unicast Demo 的使用范围，用户可以通过简单配置，实现

- 一对一 (Client 对 Server)，典型场景如 Headset
- 一对二 (Client 对 Server)，典型场景如 TWS
- 音频上下行采样率 16kHz/24kHz/32kHz/48kHz 自由配置
- 音频数据的多路复用，把多通道音频数据合成一路传输

2.1 Unicast Demo Scenario 1

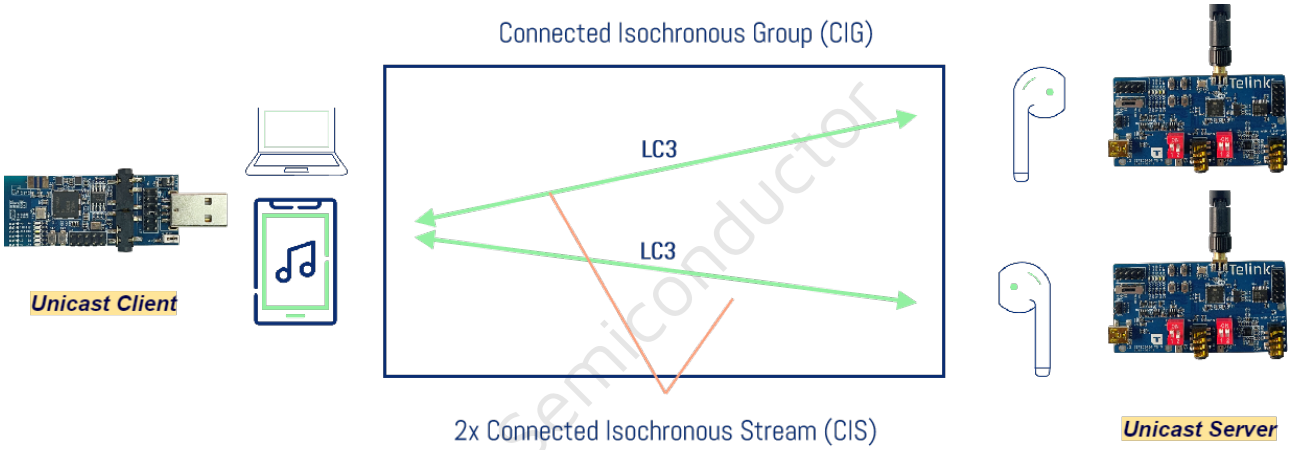

Figure 2.1: Unicast scenario 1

Unicast Client 与 Unicast Server 一对二，上行双声道 16kHz 采样率，下行双声道 48kHz 采样率，如上图。

配套开发板：

Table 2.1: Unicast Demo Scenario 1 配套开发板

类别	芯片料号	开发板外部名称	个数	配套附件
Unicast Server	9517C	B91 音频开发板	2	USB 线 (供电) 2 根，天线 2 个
Unicast Client	9518A	B91 Dongle	1	-

2.1.1 场景介绍

B91 Dongle 作为 Unicast Client，分别与两个 Unicast Server 建立音频通道。

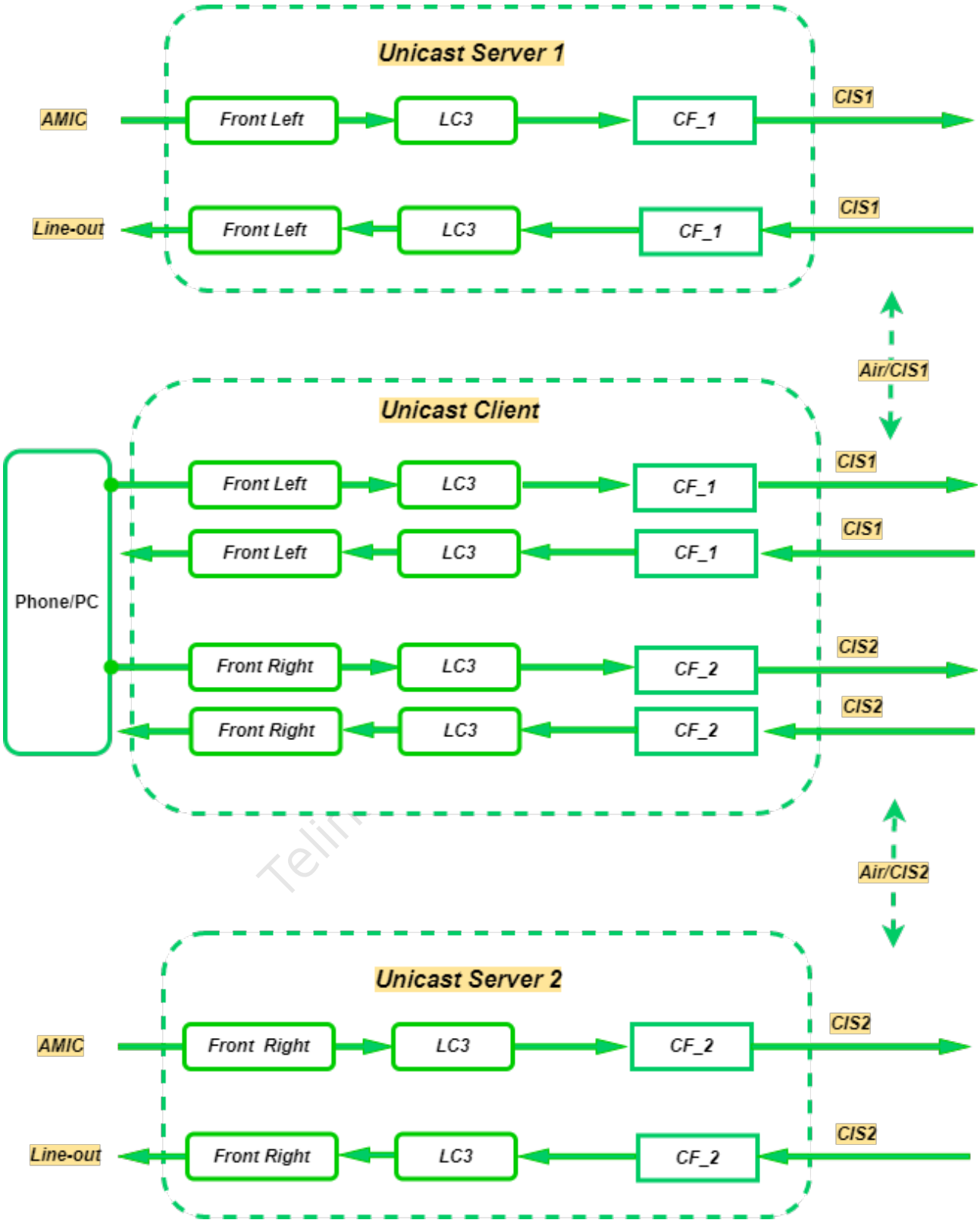

Figure 2.2: Unicast Scenario 1 结构框图

- B91 Dongle 作为 USB Device，从 USB Host(Dongle 连接的手机或电脑) 处获取 48kHz 双声道音频数据，经过 lc3 压缩后分别把左右声道发送给两个 Unicast Server。

- 两个 Unicast Server 从与 Unicast Client 建立的音频通道中获取音频数据之后，将单声道数据通过本地 Line-out 播放。
- 两个 Unicast Server 在本地通过 AMIC 采集 16kHz 环境声数据，通过与 Unicast Client 建立的音频通道发送给 Unicast Client。
- B91 Dongle 接收到两个 Unicast Server 传输的 16kHz 单声道数据之后，在本地合成 16kHz 双声道数据，作为 USB Mic 设备，将 16kHz 发送给 USB Host。

注意：

> Telink B91 Dongle 作为 USB Device，将 16kHz 双声道音频数据传给 USB Host(手机或电脑)，有些 USB Host 可能取一个声道，有些 USB Host 可能取两个声道。

2.1.2 固件编译-Unicast Client

如果已有固件可以跳过这一步。

- **Step1**：Hardware 选择 TLSR9518DONGLE。

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

- **Step2**：模式选择 TWS。

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

- **Step3**：选择音频场景。

```
#define APP_AUDIO_CONFIGURATION_PREFER
↪  BLC_AUDIO_11II_SVR_2_SINK_2_CHN_1_SRC_2_CHN_1_CISES_2_STREAMS_4
```

音频场景的具体内容请参考**BAP_v1.0.1, Table4.1：Unicast LC3 Audio Configurations**。

- **Step4**：选择音频参数，下行 48kHz 采样率，上行 16kHz 采样率。

```
#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**：如果使用 USB 模式，则需要修改 USB Audio 枚举参数，Speaker 下行双声道 48kHz，Mic 上行双声道 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
```

如果使用 Codec 模式，则需要修改 Codec 参数。

```
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**：参数配置完成，点击编译，生成目标固件。

2.1.3 固件编译-Unicast Server

如果已有固件可以跳过这一步。

- **Step1**：Hardware 选择 TLSR9517CDK56D。

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

- **Step2**：模式选择 TWS。

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

- **Step3**：TWS 选择左耳或右耳，1 为左耳，2 为右耳。

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

- **Step4**：参数配置完成，点击编译，生成目标固件。

注意：

> TWS 场景的 Unicast Server 支持大多数音频参数，如 16kHz,24kHz,32kHz,48kHz 上下行以及 7.5ms 帧，10ms 帧。Unicast Server 会在协议交互过程中确定音频参数，根据音频参数动态配置本地 codec。

2.1.4 操作步骤

Figure 2.3: Unicast scenario 1 环境搭建

- **Step1**: 将 Unicast Client 固件烧录到一块 TLSR9518 Dongle 中，通过 USB 连接到手机或电脑。
- **Step2**: 将 Unicast Server 固件烧录到两块 TLSR9517C 音频板中 (分别选择左耳和右耳), 通过 USB 供电。
- **Step3**: Dongle 端按键 SW2 配对，Dongle 连接到一个 Unicast Server 设备，会通过协调集 CSIP 协议自动搜索并连接到另一个 Unicast Server。
- **Step4**: 手机或电脑端播放音乐或打开录音，测试音频上下行通路。

注意：

> 固件烧录之前，请擦除整块 flash(B91 可以选择直接擦除 2048KB)。

2.1.5 TLSR9518 Dongle UI

- 按键 SW2 主动搜索 Unicast Server 设备并配对。
- 按键 SW7 解配对 (需要处于连接状态)。
- LED Red 呼吸灯，每一秒闪烁一次。
- LED Blue,Dongle 与 Server 建立了蓝牙 ACL 连接。
- LED Green,Dongle 与 Server 建立了蓝牙 CIS 音频通路。

2.1.6 TLSR9517C 音频板 UI

- SW8,SW9,SW10 用户定制。
- SW11 芯片 Reset。
- LED Green 呼吸灯，每一秒闪烁一次。
- LED Red,Server 与 Client 建立了蓝牙 ACL 连接。
- LED White,Server 与 Client 建立了蓝牙 CIS 音频通路。

2.2 Unicast Demo Scenario 2

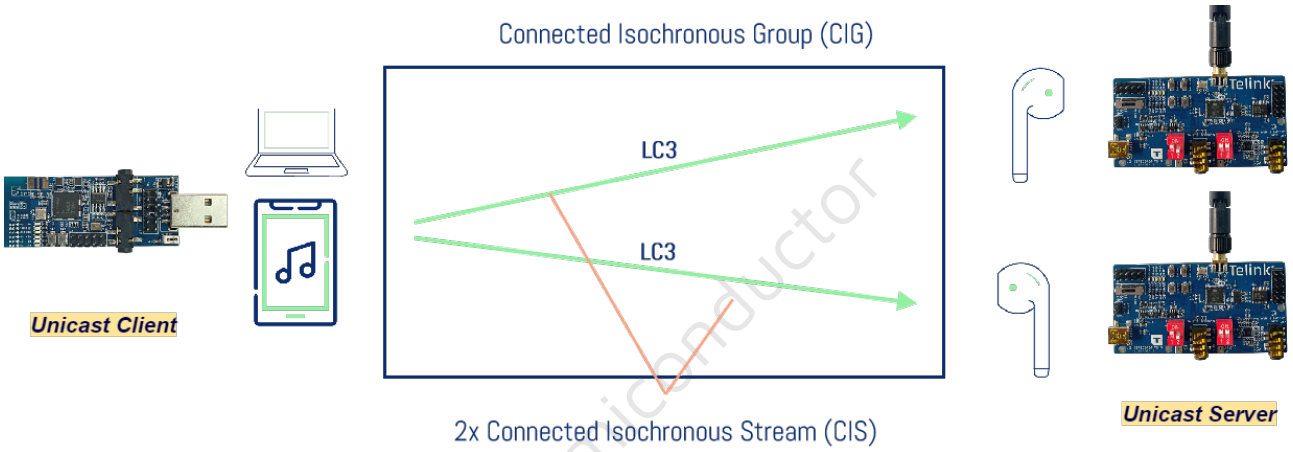

Figure 2.4: Unicast scenario 2

Unicast Client 与 Unicast Server 一对二，单下行，双声道 48kHz，如上图。

配套开发板：

Table 2.2: Unicast Demo Scenario 2 配套开发板

类别	芯片料号	开发板外部名称	个数	配套附件
Unicast Server	9517C	B91 音频开发板	2	USB 线 (供电) 2 根，天线 2 个
Unicast Client	9518A	B91 Dongle	1	-

Telink Semiconductor

2.2.1 场景介绍

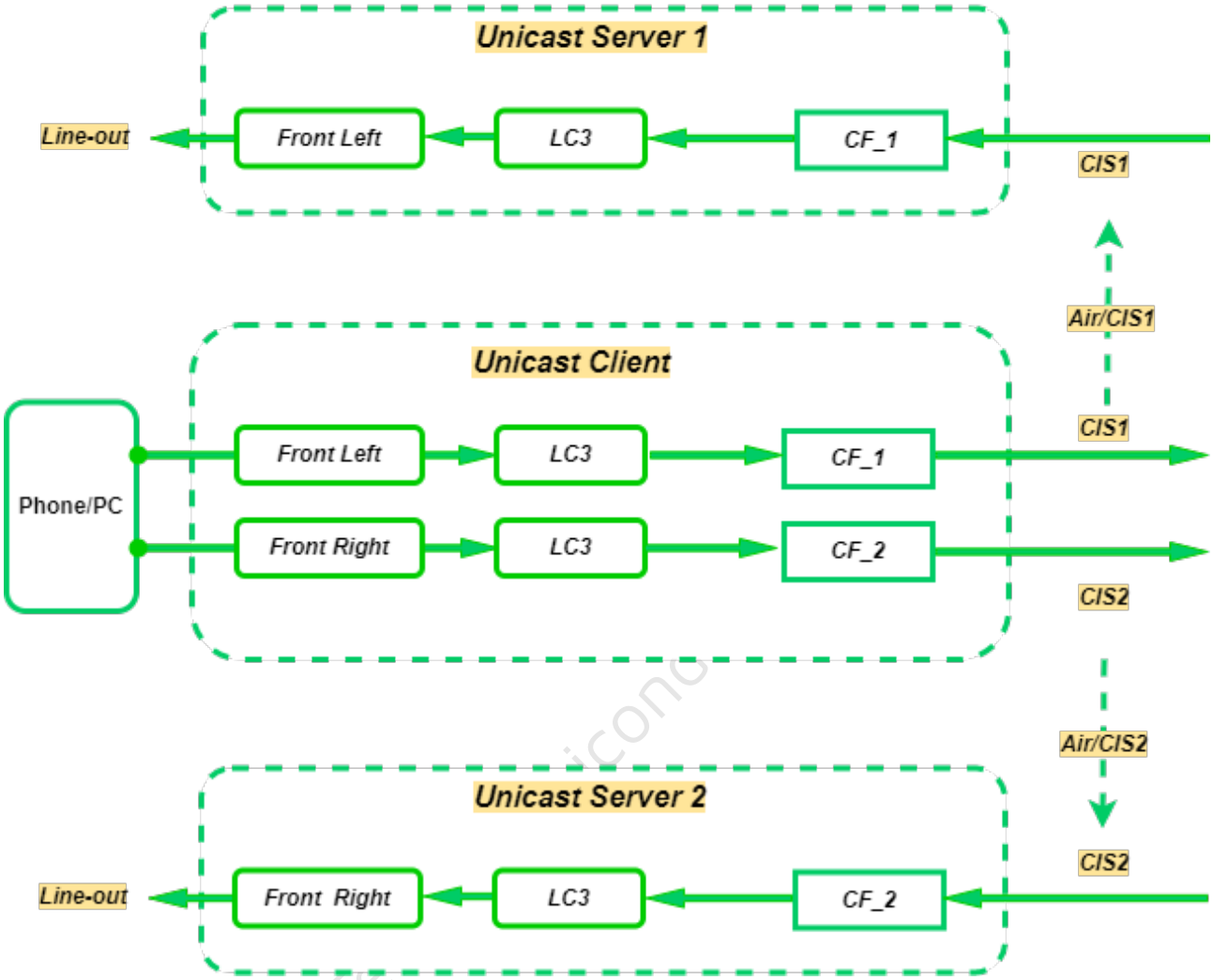

Figure 2.5: Unicast Scenario 2 结构框图

B91 Dongle 作为 Unicast Client，分别与两个 Unicast Server 建立音频通道。

- B91 Dongle 作为 USB Device，从 USB Host(Dongle 连接的手机或电脑) 处获取 48kHz 双声道音频数据，经过 lc3 编码后分别把左右声道发送给两个 Unicast Server。
- 两个 Unicast Server 从与 Unicast Client 建立的音频通道中获取音频数据之后，将单声道数据通过本地 Line-out 播放。

2.2.2 固件编译-Unicast Client

如果已有固件可以跳过这一步。

- **Step1**: Hardware 选择 TLSR9518DONGLE。

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

- **Step2**：模式选择 TWS。

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

- **Step3**：选择音频场景。

```
#define APP_AUDIO_CONFIGURATION_PREFER
↪  BLC_AUDIO_6II_SVR_2_SINK_2_CHN_1_SRC_N_CHN_N_CISES_2_STREAMS_2
```

音频场景的具体内容请参考**BAP_v1.0.1, Table4.1：Unicast LC3 Audio Configurations**。

- **Step4**：选择音频参数，下行 48kHz。

```
#define APP_AUDIO_CODEC_INPUT_PARAMETER_PREFER
↪  BLC_AUDIO_STD_FREQ_48K_DURATION_10MS_FRAME_100BYTES
```

- **Step5**：如果使用 USB 模式，则需要修改 USB Audio 枚举参数，Speaker 下行双声道 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
```

如果使用 Codec 模式，则需要修改 Codec 参数。

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

- **Step6**：参数配置完成，点击编译，生成目标固件。

2.2.3 固件编译-Unicast Server

如果已有固件可以跳过这一步。

- **Step1**：Hardware 选择 TLSR9517CDK56D。

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

- **Step2**：模式选择 TWS。

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

- **Step3**：TWS 选择左耳或右耳，1 为左耳，2 为右耳。

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

- **Step4**：参数配置完成，点击编译，生成目标固件。

注意：

> TWS 场景的 Unicast Server 支持大多数音频参数，如 16kHz,24kHz,32kHz,48kHz 上下行以及 7.5ms 帧，10ms 帧。Unicast Server 会在协议交互过程中确定音频参数，根据音频参数动态配置本地 codec。

2.2.4 操作步骤

Figure 2.6: Unicast scenario 2 环境搭建

- **Step1**：将 Unicast Client 固件烧录到一块 TLSR9518 Dongle 中，通过 USB 连接到手机或电脑。
- **Step2**：将 Unicast Server 固件烧录到两块 TLSR9517C 音频板中 (分别选择左耳和右耳)，通过 USB 供电。

- **Step3**：Dongle 端按键 SW2 配对，Dongle 连接到一个 Unicast Server 设备后，会通过协调集 CSIP 协议自动搜索并连接到另一个 Unicast Server。
- **Step4**：手机或电脑端播放音乐或打开录音，测试音频上下行通路。

注意：

> 固件烧录之前，请擦除整块 flash(B91 可以选择直接擦除 2048KB)。

2.2.5 TLSR9518 Dongle UI

- 按键 SW2 主动搜索 Unicast Server 设备并配对。
- 按键 SW7 解配对 (需要处于连接状态)。
- LED Red 呼吸灯，每一秒闪烁一次。
- LED Blue,Dongle 与 Server 建立了蓝牙 ACL 连接。
- LED Green,Dongle 与 Server 建立了蓝牙 CIS 音频通路。

2.2.6 TLSR9517C 音频板 UI

- SW8,SW9,SW10 用户定制。
- SW11 芯片 Reset。
- LED Green 呼吸灯，每一秒闪烁一次。
- LED Red,Server 与 Client 建立了蓝牙 ACL 连接。
- LED White,Server 与 Client 建立了蓝牙 CIS 音频通路。

2.3 Unicast Demo Scenario 3

Telink Semiconductor

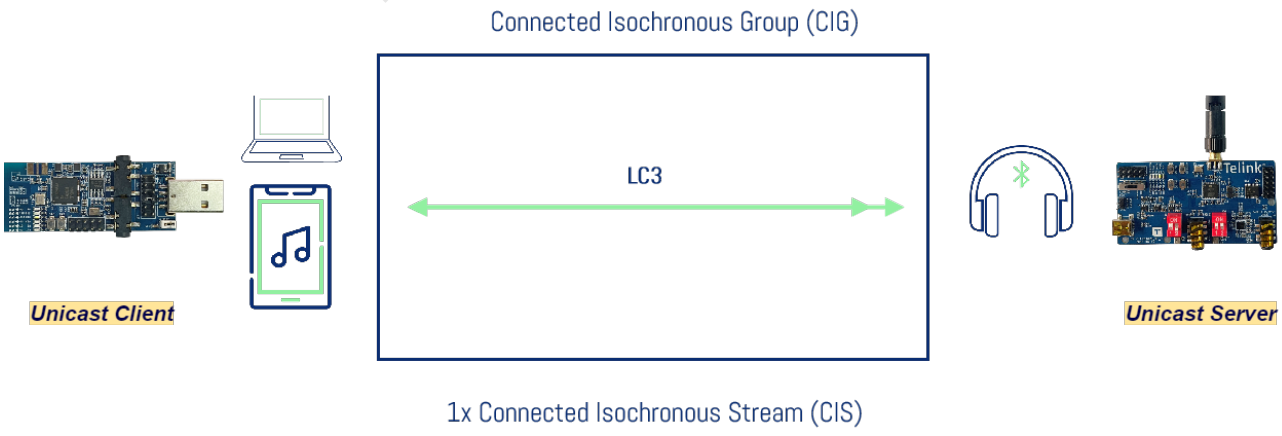

Figure 2.7: Unicast scenario 3

Unicast Client 与 Unicast Server 一对一，下行 48kHz 双声道，采用多路复用技术，一个 CIS 通路传输两路音频数据，上行 16kHz 单声道，模拟 Headset 场景，如上图。

配套开发板：

Table 2.3: Unicast Demo Scenario 3 配套开发板

类别	芯片料号	开发板外部名称	个数	配套附件
Unicast Server	9517C	B91 音频开发板	1	USB 线 (供电) 1 根，天线 1 个
Unicast Client	9518A	B91 Dongle	1	-

2.3.1 场景介绍

B91 Dongle 作为 Unicast Client，与一个 Unicast Server 建立音频通道。

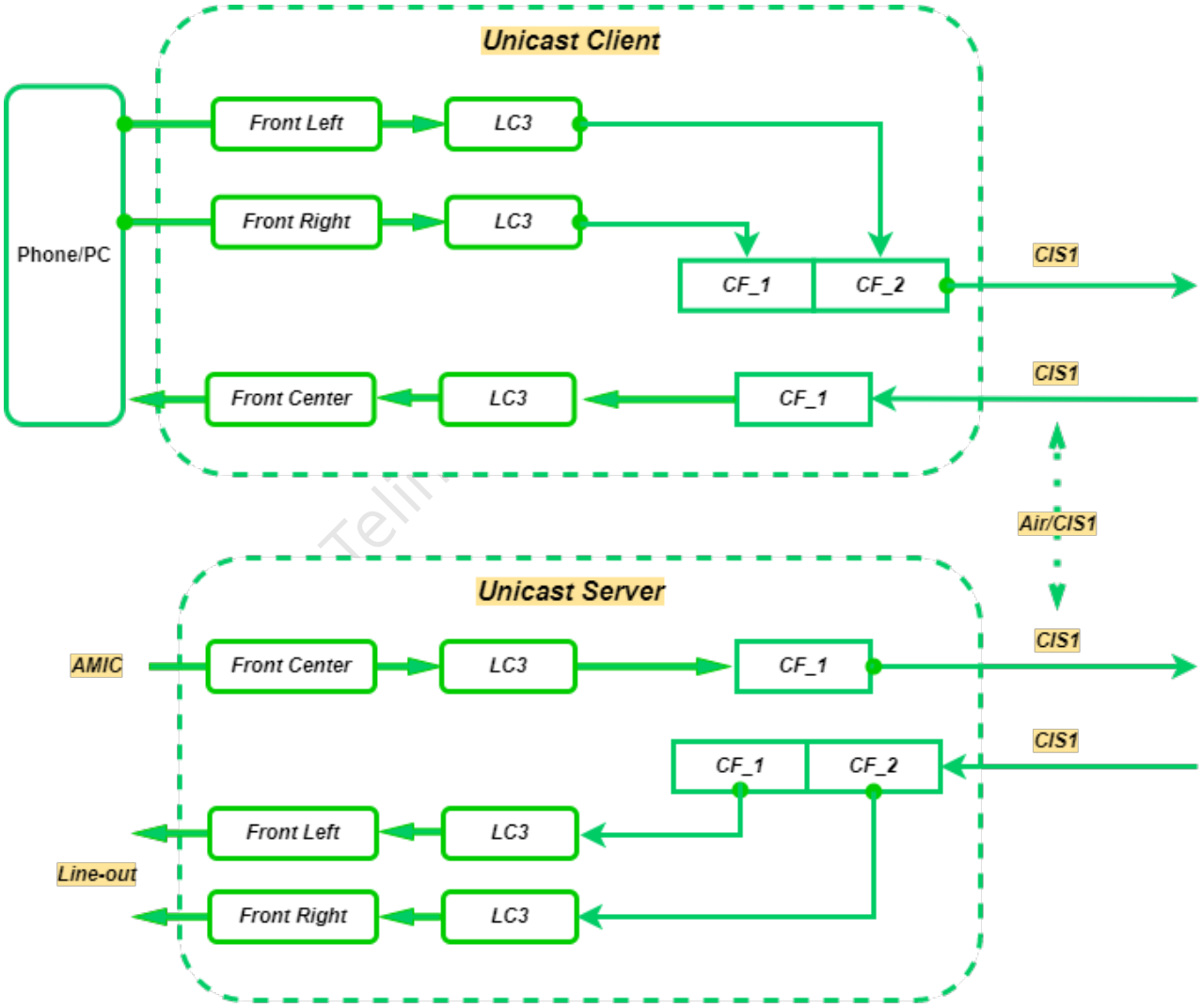

Figure 2.8: Unicast Scenario 3 结构框图

- B91 Dongle 作为 USB Device，从 USB Host(Dongle 连接的手机或电脑) 处获取 48kHz 双声道音频数据，

经过 lc3 压缩后把左右声道合并为一个 SDU 发送给 Unicast Server。

- Unicast Server 从与 Unicast Client 建立的音频通道中获取两个声道的音频数据之后，分别通过 LC3 解码通过本地 Line-out 通路播放。
- Unicast Server 在本地通过 AMIC 采集 16kHz 单声道环境声数据，通过与 Unicast Client 建立的音频通道发送给 Unicast Client。
- B91 Dongle 作为 USB Mic 设备，接收到 Unicast Server 传输的 16kHz 单声道数据之后，发送给 USB Host。

2.3.2 固件编译-Unicast Client

如果已有固件可以跳过这一步。

- **Step1**：Hardware 选择 TLSR9518DONGLE。

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

- **Step2**：模式选择 TWS。

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

- **Step3**：选择音频场景。

```
#define APP_AUDIO_CONFIGURATION_PREFER
↪  BLC_AUDIO_5_SVR_1_SINK_1_CHN_2_SRC_1_CHN_1_CISES_1_STREAMS_2
```

音频场景的具体内容请参考**BAP_v1.0.1, Table4.1：Unicast LC3 Audio Configurations**。

- **Step4**：选择音频参数，下行 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**：如果使用 USB 模式，则需要修改 USB Audio 枚举参数，Speaker 下行双声道 48kHz，Mic 上行单声道 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
```

如果使用 Codec 模式，则需要修改 Codec 参数, 输入双声道 48kHz，输出单声道 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**：参数配置完成，点击编译，生成目标固件。

2.3.3 固件编译-Unicast Server

如果已有固件可以跳过这一步。

- **Step1**：Hardware 选择 TLSR9517CDK56D。

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

- **Step2**：模式选择 Headset。

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

- **Step3**：参数配置完成，点击编译，生成目标固件。

注意：

> Headset 场景只有一个设备，没有左耳和右耳的区分。

2.3.4 操作步骤

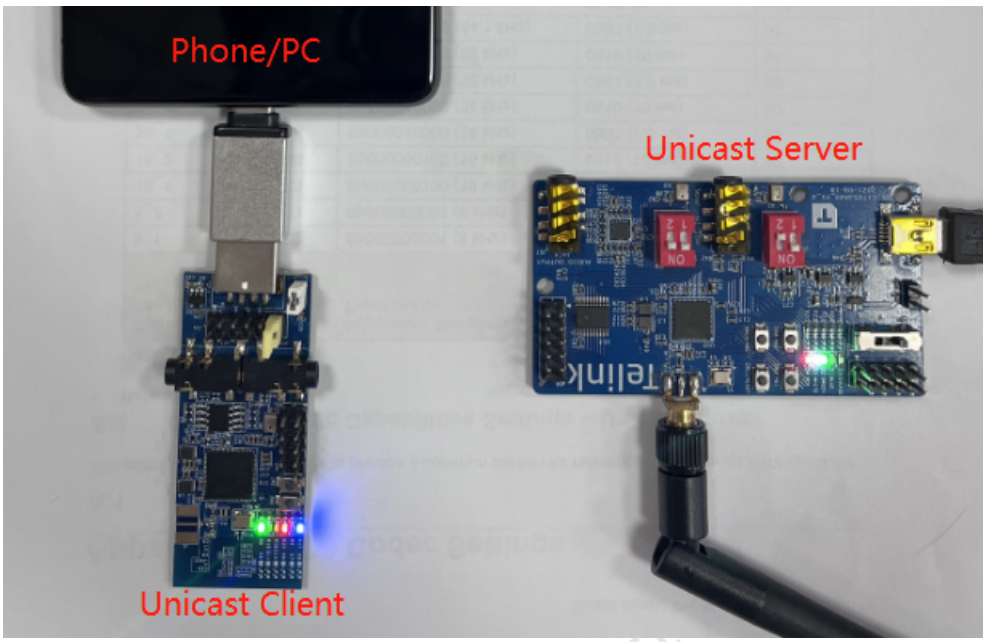

Figure 2.9: Unicast scenario 3 环境搭建

- **Step1**: 将 Unicast Client 固件烧录到一块 TLSR9518 Dongle 中，通过 USB 连接到手机或电脑。
- **Step2**: 将 Unicast Server 固件烧录到一块 TLSR9517C 音频板中, 通过 USB 供电。
- **Step3**: Dongle 端按键 SW2 配对，Dongle 连接到目标 Unicast Server。
- **Step4**: 手机或电脑端播放音乐或打开录音，测试音频上下行通路。

注意:

> 固件烧录之前，请擦除整块 flash(B91 可以选择直接擦除 2048KB)。

2.3.5 TLSR9518 Dongle UI

- 按键 SW2 主动搜索 Unicast Server 设备并配对。
- 按键 SW7 解配对 (需要处于连接状态)。
- LED Red 呼吸灯，每一秒闪烁一次。
- LED Blue,Dongle 与 Server 建立了蓝牙 ACL 连接。
- LED Green,Dongle 与 Server 建立了蓝牙 CIS 音频通路。

2.3.6 TLSR9517C 音频板 UI

- SW8,SW9,SW10 用户定制。
- SW11 芯片 Reset。

- LED Green 呼吸灯，每一秒闪烁一次。
- LED Red,Server 与 Client 建立了蓝牙 ACL 连接。
- LED White,Server 与 Client 建立了蓝牙 CIS 音频通路。

Telink Semiconductor

3 Broadcast Audio Demo

低功耗音频为蓝牙技术带来了广播音频功能，这项技术让音频通讯不再局限于点对点通讯，可以让音频源设备能够向附近不限数量的蓝牙音频接收设备播放音频流。广播音频是符合一套规定配置的蓝牙广播音频，能通过蓝牙技术实现一对多的音频广播和分享功能，为消费者提供新的、全球可互操作性的音频体验。

Bluetooth SIG 为蓝牙广播音频功能发布新商标，将音频分享功能 (低功耗蓝牙音频 Bluetooth LE Audio 技术的一部分) 正式命名为广播音频 (broadcast audio)。SIG 也给出了商标指导文件《Brand Guide for Bluetooth Trademarks_Jun2022》，要求蓝牙会员满足该技术要求的产品必须通过《公共广播配置文件（Public Broadcast Profile, PBP）规范》的测试，完成蓝牙资格认证后，才能使用商标，这进一步保证了统一的互操作性。

广播音频一个典型场景：

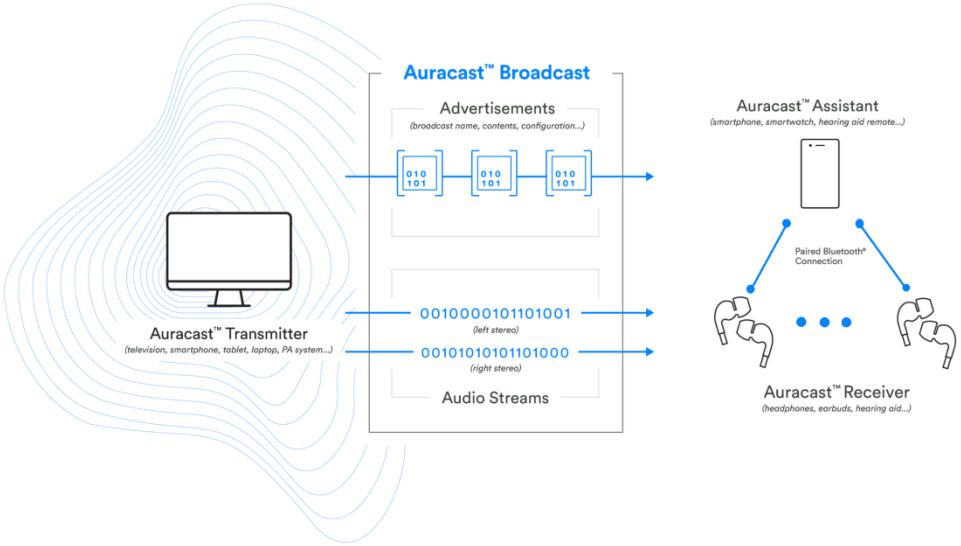

Figure 3.1: Broadcast 工作原理图

广播源（如智能手机、电视等）广播音频流以及相关接收信息，包括传输是否需要安全加密，广播名称，音频编码方式等信息。

接收器（如耳机、耳塞、助听器等）等可以搜索各种广播源发送音频流以及接收信息，并接收传输。

控制助手会搜索 Auracast 广播，并给用户提供一个可以选择连接 Auracast 广播的用户界面（UI）。在实际应用中广播控制助手可以是类似智能手表这样的设备，也可能是单独的遥控器，目的就是为了让最终用户在搜索或者选择广播音频的时候更便捷快速。在控制助手上选择传输后，控制助手会通知接收器（如耳机、耳塞、助听器等）如何接入传输，这样可以省去接收器搜索或者选择的处理。这对于控制界面比较简单的接收器设备（例如耳塞，助听器等）非常重要。

3.1 Broadcast Demo Scenario 1

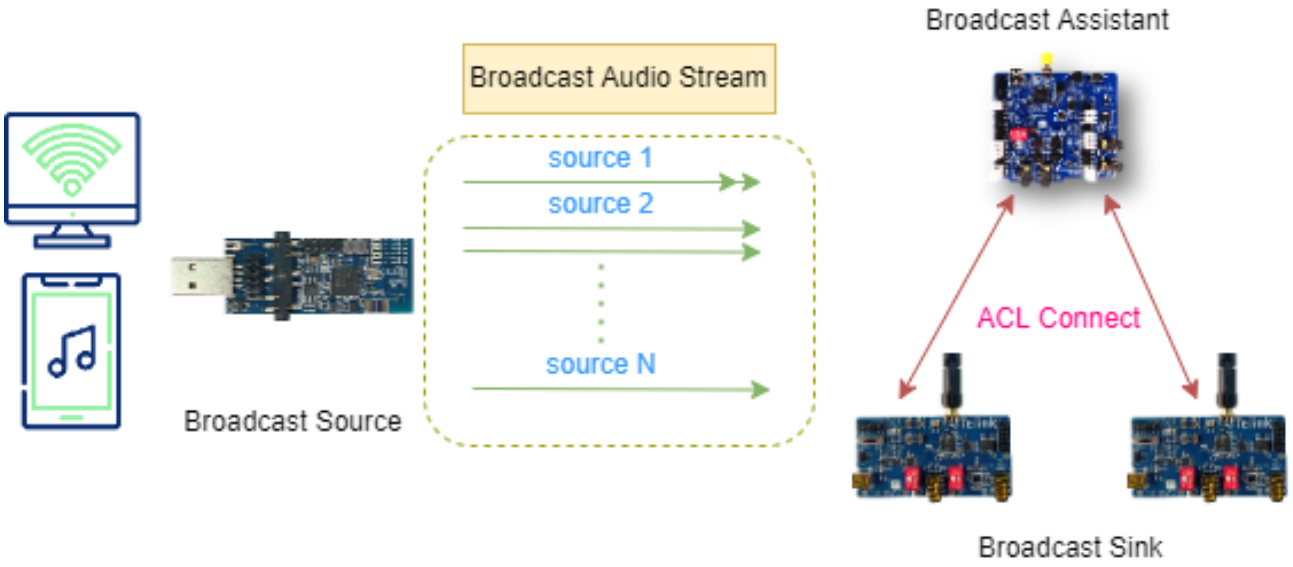

Figure 3.2: 广播音频场景 1

Broadcast Sink 通过 Broadcast Assistant 的异步连接，来选择同步 Source 信息。

配套开发板：

Table 3.1: Broadcast Demo Scenario 1 配套开发板

类别	芯片料号	开发板外部名称	个数	配套附件
Broadcast Sink	9517C	B91 音频开发板	2	USB 线 (供电) 2 根，天线 2 个
Broadcast Source	9518A	B91 Dongle	2	-
Broadcast Assistant	9218A	B91 EVK	1	USB 线 (供电) 1 根，天线 1 个

3.1.1 场景介绍

B91 Dongle 作为 Broadcast Source，向空中广播 BIG（Broadcaster Isochronous Group，广播同步组）音频。空中存在 N 个 Source 广播的 BIG 信息。Source 1 有一个 BIS（Broadcaster Isochronous Stream，广播同步流），传输两路音频；Source 2 有两个 BIS，每个 BIS 分别传输一路音频；Source N 有一个 BIS，只传输一路音频。

Broadcast Assistant 和 Broadcast Sink 之间存在一个异步连接，assistant 通过异步连接来替 sink 选择同步哪路 Source。该场景下，Assistant 提供了 USB-CDC 接口用来配置指令。

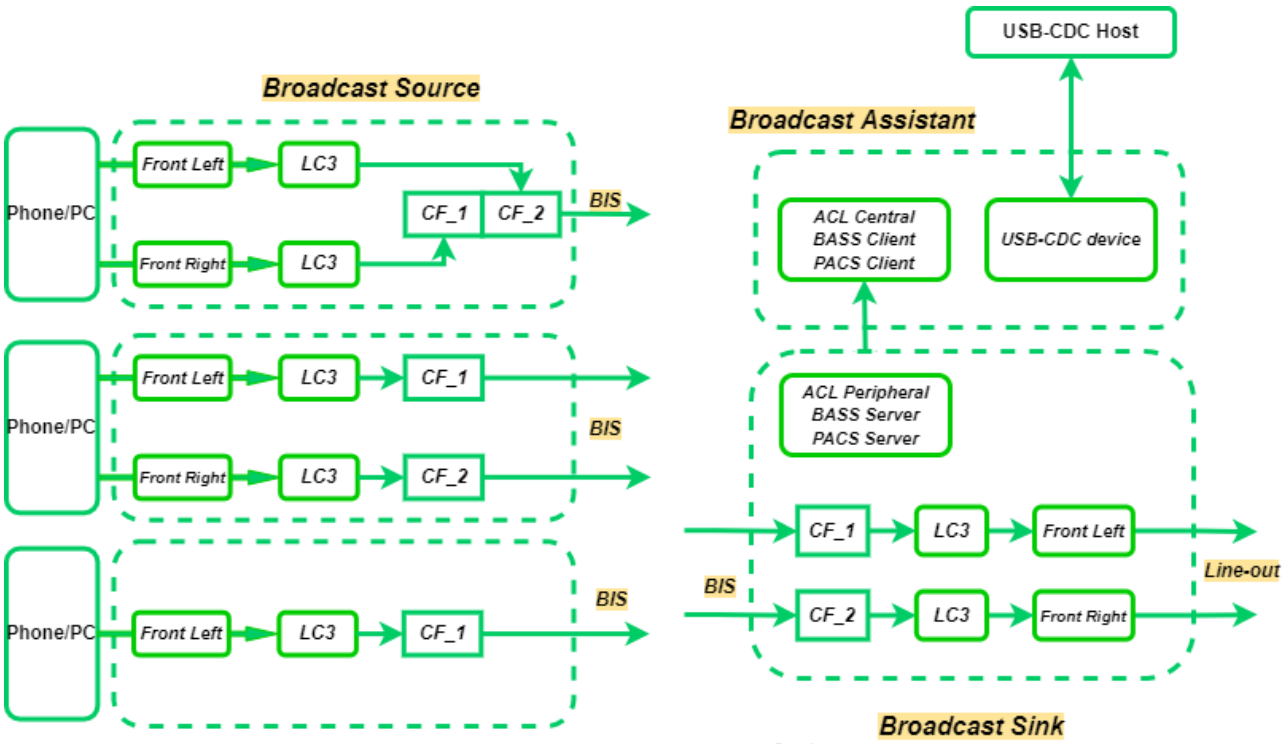

Figure 3.3: 广播音频场景 1 结构框图

- B91 Dongle 作为广播源设备，配置为 USB Device，从 USB Host(Dongle 连接的手机或电脑) 处获取 48kHz 双声道音频数据，经过 lc3 编码后，根据不同的配置将音频数据发射到空中。
- B91 EVK 作为广播辅助器设备，配置为 USB-CDC Device，可以和 USB-CDC Host(电脑使用串口工具软件) 通讯，接收到 USB 指令，来执行不同的操作。
- B91 音频开发板作为广播接收器，与广播辅助器建立异步连接，接收辅助器的不同命令，来同步空中的广播源。同步建立完成后，将获取到的音频数据，LC3 解码后，同步本地 Line-out 播放。

3.1.2 固件编译-Broadcast Source

如果已有固件可以跳过这一步。

- **Step1**: Source 版本选择 SOURCE_ONLY_VERSION。

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

- **step2**: 根据需求配置广播名称，广播 ID，设备名称等基本信息。

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

- **step3**: 根据需求选择音频输入模式，USB 输入、Line-in、AMIC 和测试模式四种方式。

```
#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**: 根据需求配置扩展广播和周期性广播间隔。

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

- **step5**: 根据需求配置 presentation delay，BIS 数量 (默认最大为 2)，LC3 编码参数，每路 BIS 广播的音频通道。

```
// 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，影响 sink 端播放的延迟，单位是 us。

其中 USB 音频输入模式下，目前只支持 LC3_48_2、LC3_48_4、LC3_48_6 三种配置，更多的配置后续 demo 会支持。Line-in、AMIC 音频输入的方式，支持 codec 配置 LC3_8_2、LC3_16_2、LC3_32_2、LC3_48_2、LC3_48_4、LC3_48_6 六种配置，更多的配置后续 demo 会支持。

BIS_INDEX_1_CHANNEL 和 BIS_INDEX_2_CHANNEL 分别是 BIS 1 和 BIS 2 的音频通道，目前只支持 Left 和 Right。

BLC_AUDIO_LOCATION_FLAG_FL 表示只传输左声道；BLC_AUDIO_LOCATION_FLAG_FR 表示只传输右声道；

BLC_AUDIO_LOCATION_FLAG_FL | BLC_AUDIO_LOCATION_FLAG_FR 表示 BIS 中传输立体声包含左声道和右声道。

- **step6**: 根据需求配置 ISO 间隔，传输延迟，BIG 加密参数。

```
// 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"
```

BIG_INFO_ENC_FLAG 和 BIG_INFO_BROADCAST_CODE 两个宏定义可以控制 BIG 加密。只有 BIG_INFO_ENC_FLAG 等于 1 时，BIG 的 Broadcast code 设置为 BIG_INFO_BROADCAST_CODE。

BIG_INFO_ISO_INTERVAL 宏定义是表示 BIG 是 ISO interval，单位是 SDU interval。通常只能设置 1，2，3。

BIG_INFO_TRANSPORT_LATENCY 宏定义表示音频的传输延迟，单位是 ms。主要用来控制 PTO 的值。

- **step7**：参数配置完成，点击编译，生成目标固件。

3.1.3 固件编译-Broadcast Assistant

如果已有固件可以跳过这一步。

- **Step1**：Assistant 版本选择 UNIVERSAL_VERSION。

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

- **Step2**：如果配置 UI 接口为 APP_AUDIO_UI_USB_CDC，跳过 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**：配置 UART 的 TX 和 RX 引脚。

```
#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**：参数配置完成，点击编译，生成目标固件。

3.1.4 固件编译-Broadcast sink

如果已有固件可以跳过这一步。

- **Step1**：sink 版本选择 SINK_ONLY_VERSION。

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

- **Step2**：参数配置完成，点击编译，生成目标固件。

3.1.5 操作步骤

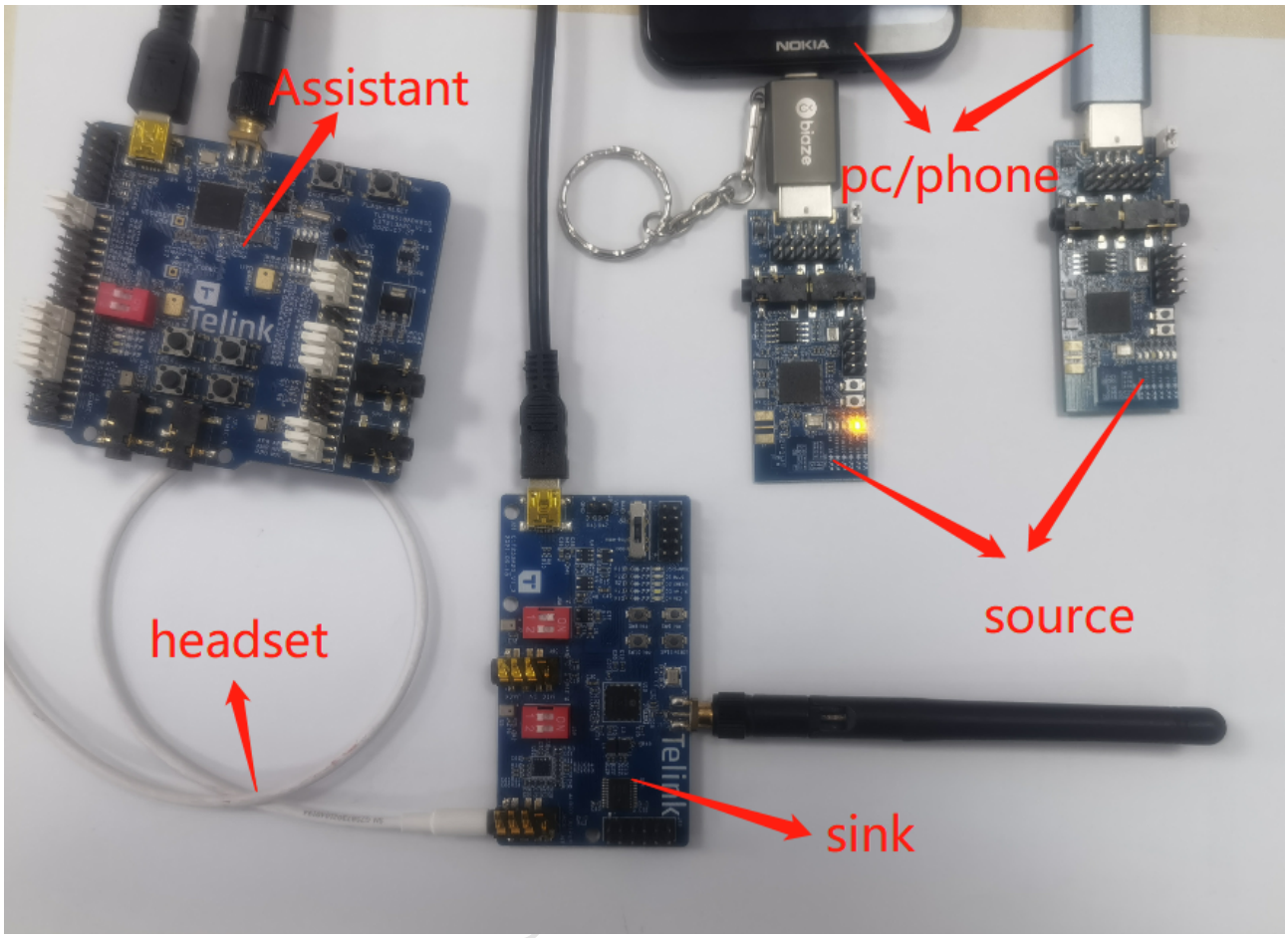

Figure 3.4: Broadcast 场景 1 环境搭建

- **Step1**: 将 Broadcast Source 固件烧录到两块 B91 Dongle 中，通过 USB 连接到手机或电脑。
- **Step2**: 将 Broadcast Assistant 固件烧录到一块 B91 EVK 中，通过 USB-CDC 连接到电脑。
- **Step3**: 将 Broadcast Sink 固件烧录到一块 B91 音频板中，通过 USB 供电。
- **Step4**: 按照 Assistant 的操作步骤，进行配对连接、广播同步、音量调节等功能。
- **Step5**: 手机或电脑端播放音乐，Sink 端能听到正在播放的音乐。

3.1.6 Broadcast Source UI

- 黄灯闪烁，每秒闪烁一次。
- 蓝灯常亮指示 USB 有音频传输。

3.1.7 Broadcast Assistant UI

- 蓝灯闪烁，每秒闪烁一次。

- 红灯指示与 Sink 建立了异步连接。
- 支持 USB-CDC 和 UART 命令控制。

3.1.8 Broadcast Sink UI

- 按键 SW9，音量加
- 按键 SW8，音量减。
- 按键 SW10，静音和取消静音。
- 白灯闪烁，每秒闪烁一次。
- 绿灯指示与 Source 建立了 BIG 同步。
- 蓝灯指示与 Assistant 建立了异步连接。

3.2 Broadcast Demo Scenario 2

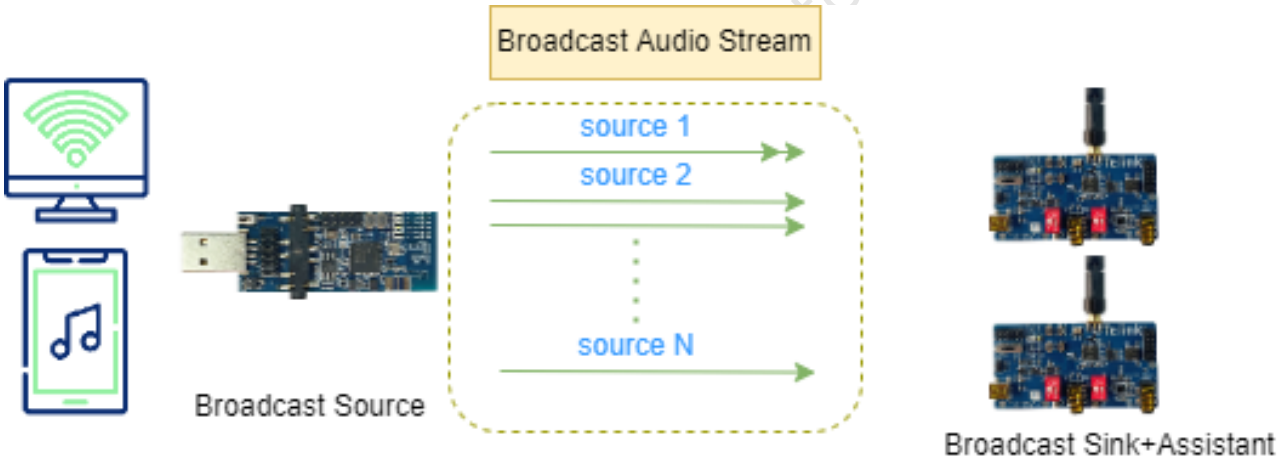

Figure 3.5: 广播音频场景 2

该场景在一个 Demo 中集成了 Sink 和 Assistant 的功能，user 可以直接通过 B91 音频开发板上的按键选择同步 Source 信息。

配套开发板：

Table 3.2: Broadcast Demo Scenario 2 配套开发板

类别	芯片料号	开发板外部名称	个数	配套附件
Broadcast Sink+Assistant	9517C	B91 音频开发板	2	USB 线 (供电) 2 根，天线 2 个
Broadcast Source	9518A	B91 Dongle	2	-

3.2.1 场景介绍

B91 Dongle 作为 Broadcast Source，向空中广播 BIG 音频。空中存在 N 个 Source 广播的 BIG 信息。Source 1 有一个 BIS，传输两路音频；Source 2 有两个 BIS，每个 BIS 分别传输一路音频；Source N 有一个 BIS，只传输一路音频。

Broadcast Assistant 和 Broadcast Sink 集成在 B91 音频开发板中，上电会自动扫描空中符合的 Broadcast Source 信息，Assistant UI 简化为使用按键触发同步新 Source。

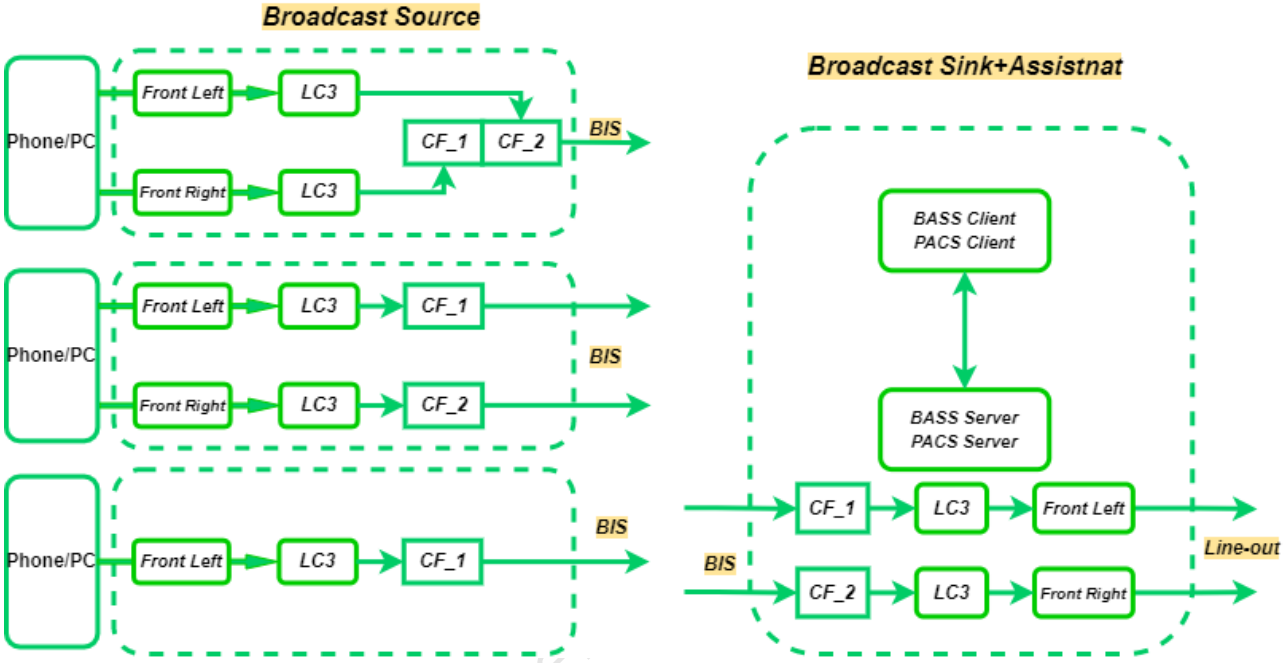

Figure 3.6: 广播音频场景 2 结构框图

- B91 Dongle 作为广播源设备，配置为 USB Device，从 USB Host(Dongle 连接的手机或电脑) 处获取 48kHz 双声道音频数据，经过 lc3 编码后，根据不同的配置将音频数据发射到空中。
- B91 音频开发板作为广播接收器和广播辅助器，上电后广播辅助器功能自动扫描空中的广播源，当用户按键触发广播源同步后，开启广播接收器的广播源同步功能，同步建立完成后，将获取到的音频数据，LC3 解码后，同步本地 Line-out 播放。

3.2.2 固件编译-Broadcast Source

参考 Broadcast Demo Scenario 1 章节下的 Source 编译流程。

3.2.3 固件编译-Broadcast Sink+Assistant

如果已有固件可以跳过这一步。

- **Step1**：sink 版本选择 SINK_WITH_ASSISTANT_VERSION.

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

- **Step2**：参数配置完成，点击编译，生成目标固件。

3.2.4 操作步骤

- **Step1**：将 Broadcast Source 固件烧录到两块 B91 Dongle 中，通过 USB 连接到手机或电脑。
- **Step2**：将 Broadcast Sink+Assistant 固件烧录到一块 B91 音频板中，通过 USB 供电。
- **Step3**：按下 B91 音频板上的按键，SW9 和 SW8 均能同步到新的 Source 信息，如果空中只有一个 Source，则反复同步该 Source。
- **step4**：手机或电脑端播放音乐，Sink 端能听到正在播放的音乐。

3.2.5 Broadcast Source UI

- 黄灯闪烁，每秒闪烁一次。
- 蓝灯常亮指示 USB 有音频传输。

3.2.6 Broadcast Sink+Assistant UI

- 按键 SW9 同步下一个 Source。
- 按键 SW8 同步上一个 Source。
- 白灯闪烁，每秒闪烁一次。
- 绿灯指示与 Source 建立了 BIG 同步。
- 蓝灯指示有扫描到合法的 Source 信息。

Telink Semiconductor

3.3 Broadcast Demo Scenario 3

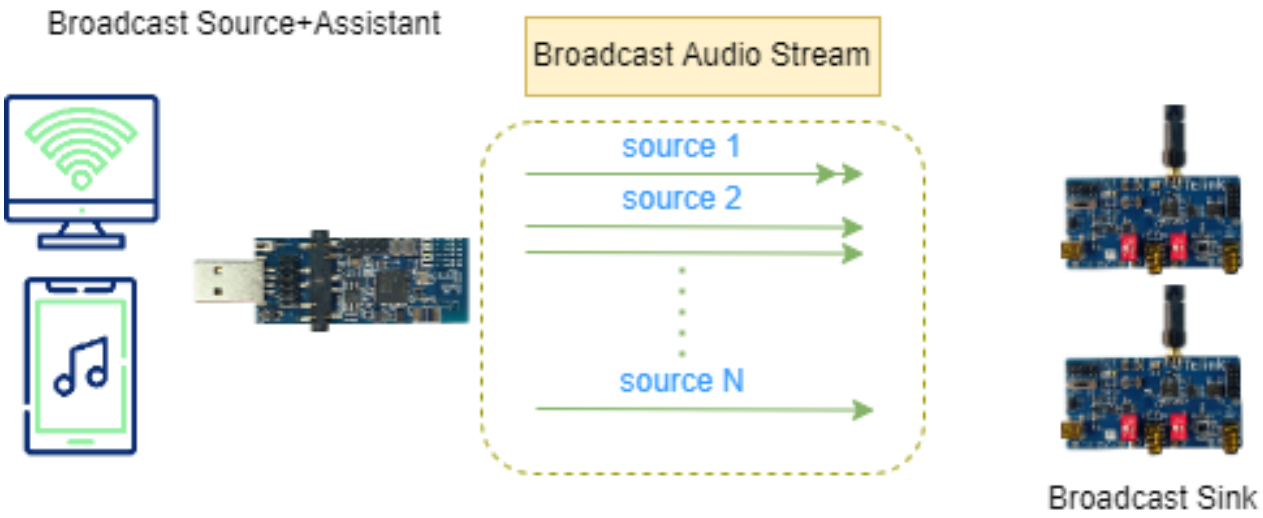

Figure 3.7: 广播音频场景 3

该场景在一个 Demo 中集成了 Source 和 Assistant 的功能，user 可以通过 UART 接口，选择将音频同步给哪个 Sink 设备。

配套开发板：

Table 3.3: Broadcast Demo Scenario 3 配套开发板

类别	芯片料号	开发板外部名称	个数	配套附件
Broadcast Sink	9517C	B91 音频开发板	2	USB 线 (供电) 2 根，天线 2 个
Broadcast Source+Assistant	9518A	B91 Dongle	2	-

3.3.1 场景介绍

B91 Dongle 作为 Broadcast Source+Assistant 的集合，向空中广播 BIG 音频。空中存在 N 个 Source 广播的 BIG 信息。Source 1 有一个 BIS，传输两路音频；Source 2 有两个 BIS，每个 BIS 分别传输一路音频；Source N 有一个 BIS，只传输一路音频。

Broadcast Assistant 和 Broadcast Sink 之间存在一个异步连接，assistant 通过异步连接来替 sink 选择同步哪路 Source。该场景下，Assistant 提供了 UART 接口用来配置指令。

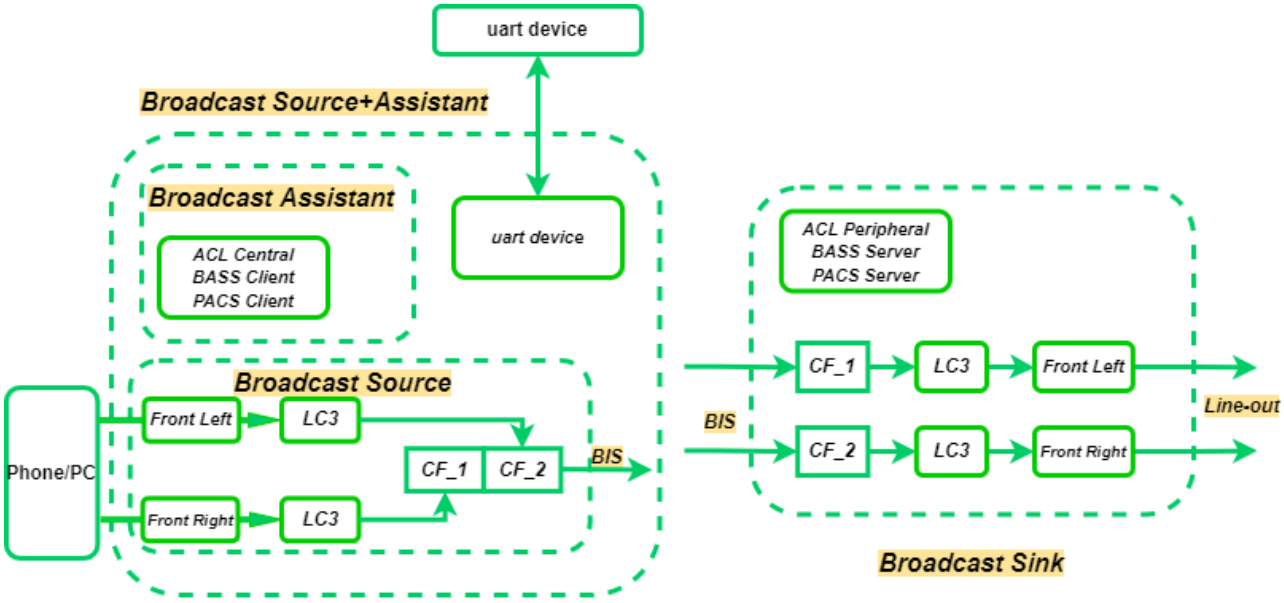

Figure 3.8: 广播音频场景 3 结构框图

3.3.2 固件编译

该场景，SDK 暂不支持。

Telink Semiconductor

4 SystemDelay

相比于经典蓝牙音频，低功耗蓝牙音频在延时方面有协议层的优势，下面分析 BLE Audio 典型场景的系统延时来源和组成。

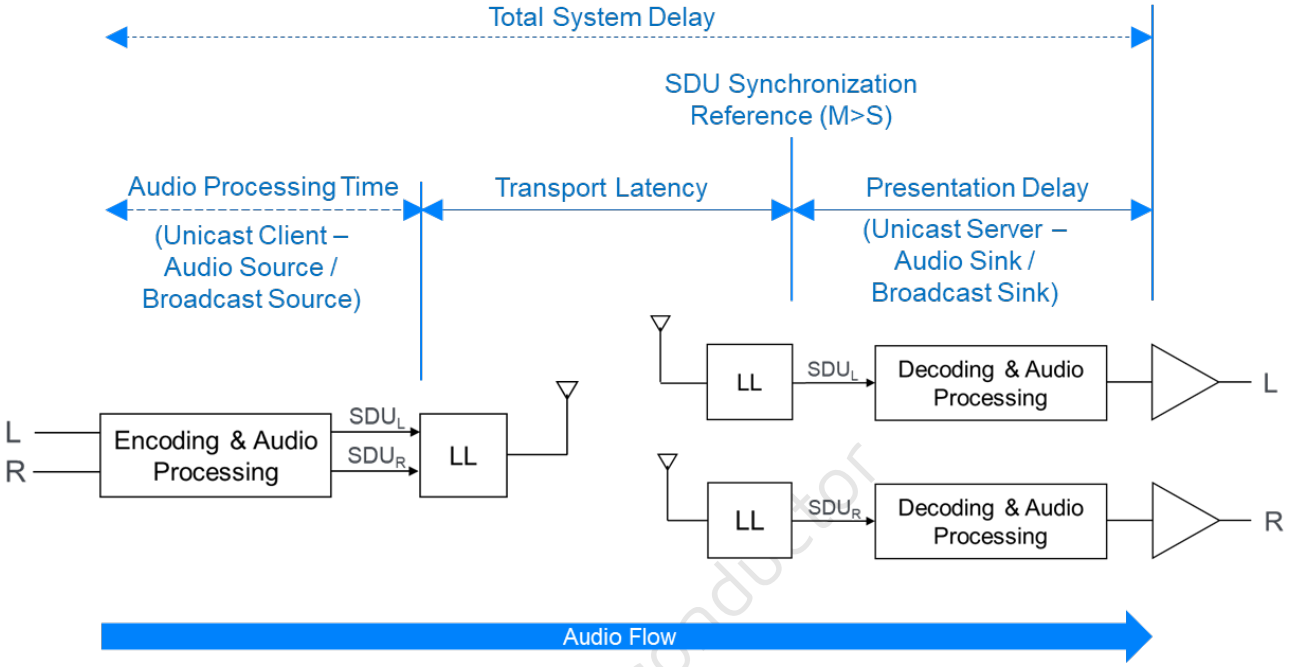

Figure 4.1: Total System Delay

由上图可以看出，系统总延时由三部分组成：

- Audio Processing Time - 音频处理时间
- Transport Latency - 音频传输延时
- Presentation Delay - 音频演示延时

4.1 Audio Processing Time

Audio Processing Time 简称为 T_proc。

以 Unicast 场景为例 (Broadcast 与 Unicast 场景基本相同)，音频处理时间主要由以下几部分时间组成，如下图：

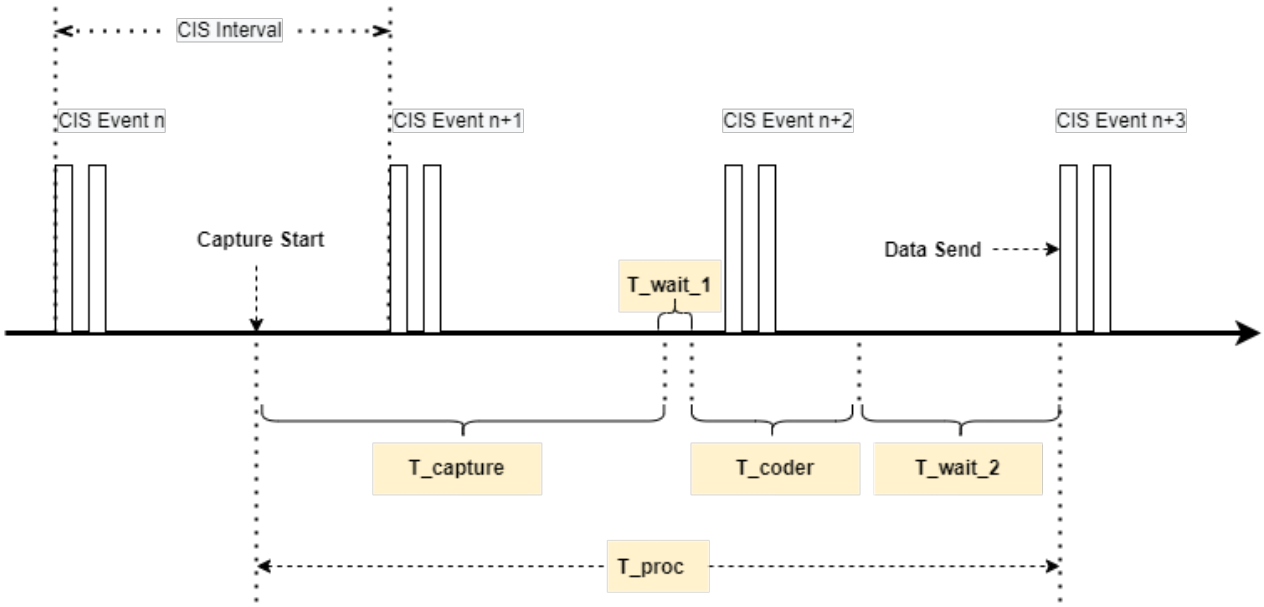

Figure 4.2: Audio_Processing_Time

- **音频采集时间 T_capture**

音频采集时间是一个固定时间，如果 SDU Interval 是 10ms，采集时间固定是 10ms，如果 SDU Interval 是 7.5ms，采集时间固定是 7.5ms。

- **音频编解码时间 T_coder**

音频的编解码时间主要受两个因素的影响：

(1) 每帧数据的数据量

每帧数据的数据量受采样率，位宽，帧长等因素的影响。

(2) 设备的数据处理能力

主要体现在 CPU 的运算能力，受设备时钟频率的影响。

- **处理时间以及传输等待时间 T_wait_1 和 T_wait_2**

一帧音频数据采集完成之后，需要经过处理才能进行编解码，例如左右声道 (如果有) 提取 (LC3 是一个单通道编解码算法)，软件程序运行耗时等，由此引入等待时间 **T_wait_1**。

一帧音频数据经过采集，编码之后并不能立即发送，需要等待直到下一次发包时间，由此引入等待时间 **T_wait_2**。

从上述描述可以看出，音频处理时间主要有三部分组成：

T_proc = T_capture + T_coder + T_wait_1 + T_wait_2

- **T_capture** 为音频数据采集时间，固定为 SDU_Interval。
- **T_coder** 主要受到设备性能的影响，CPU 性能强就可以缩短该时间。
- **T_wait_1** 和 **T_wait_2** 是软件运行耗时以及逻辑空等时间，弹性是最大的。

Telink Semiconductor

4.2 Transport Latency

Transport Latency 简称为 T_trans。

音频数据从发送端的链路层发出，到接收端的链路层接收到，其间的时间是固定的，以 Central to Peripheral 且数据格式为 unframed 为例，具体的计算公式为：

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

- **CIG_Sync_Delay** 是为了使同一个 CIG 内的多个 CIS 同步 Sync 而引入的延时。
- **ISO_Interval** 可以理解为 CIS Interval。
- **SDU_Interval** 是 SDU 的产生时间，如果 SDU_Interval = 10ms 意味着每隔 10ms 就有一个 SDU 产生。
- **FT** 可以理解为一个 SDU 最多可以在几个 ISO Interval 内传输。

以上参数的具体含义请参考《Core_V5.4》, Vol 6, Part B。

简言之：一旦 CIS Central 和 CIS Peripheral 之间协商的 CIS 链路建立起来，两者之间的 Transport Latency 就是确定的。

4.3 Presentation Delay

接收端收到 Controller 上报的 SDU 之后，需要经过软件处理，LC3 解码，然后在固定时间送到 codec 中播放，由此引入演示延时 Presentation Delay, 简称 T_pres。

4.4 Audio System Delay

由以上分析可知

T_system_delay = T_proc + T_trans + T_pres

即

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

- **T_capture** 是音频采集时间，固定为 SDU_Interval。
- **T_coder** 和 **T_pres** 主要和 LC3 编解码时间相关，即主要和 CPU 性能有关。
- **T_wait_1** 和 **T_wait_2** 主要和软件架构有关，好的软件架构可以最大的缩短该时间。
- **T_trans** 主要和 FT 即重传次数有关，一个 SDU 在越少的 ISO_Interval 中传输，Transport Latency 越短。

由以上分析可知，如果要缩短系统总延时，可以采取的措施为：

(1) 缩短 SDU_Interval，可以缩短 T_capture 和 T_trans。

(2) 使用运算能力更强的 CPU，可以缩短 T_coder 和 T_pres。

(3) 根据特定的项目，使用高效的软件架构，可以在最大程度上缩短 T_wait_1 + T_wait_2 以及 T_pres。

(4) 减小 FT，在保证无线传输质量的情况下尽可能的缩短重传次数。

Telink Semiconductor

5 附录

5.1 Assistant 指令集

广播辅助器所有的控制指令，都需要以\r\n(ASCII: 0x0d 0x0A) 结尾，下文介绍指令时，不再重复。

5.1.1 扫描 sink

扫描 sink 的指令格式: scan-sink <start|stop|clear> \r\n

(1) scan-sink start

开启扫描 sink 功能，辅助器首先会回复 assistant start scan sink。

之后如果扫描到符合的 Sink 设备，会按照以下格式上报

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

如下是扫描到的示例。

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

其中 dev_idx 后面连接时候，需要填入。

(2) scan-sink stop

停止扫描 sink 功能，辅助器首先会回复 assistant stop scan sink。

(3) scan-sink clear

停止扫描 sink 功能，并清空已经扫描到的 sink 信息，辅助器首先会回复 assistant clear sink info。

5.1.2 连接特定 sink 设备

连接特定 sink 的指令格式: conn-sink <dev_idx> \r \n

其中 dev_idx 是在 scan-sink start 指令后上报的索引值。

conn-sink 1

如果索引值不存在，辅助器会回复：

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

如果索引值存在，辅助器会和对应是 sink 设备创建 ACL 连接。并回复 assistant start connect sink。

连接过程中，辅助器会根据 profile 的状态，上报一些中间状态。主要是 ACL 连接设备的 MAC 地址，连接的 ACL 句柄，BASS、PACS、VCP 三个服务的状态，SDP 结束的标记。

Telink Semiconductor

```
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 为已连接的 Sink 扫描源信息

为 sink 扫描源信息的指令格式：scan-bcast <start|stop|clear> <conn_idx> \r\n

其中 conn_idx 目前默认是 1，user 可以根据 show conn 指令查询索引。

(1) scan-bcast start <conn_idx>

开始扫描源信息操作.

如果 conn_idx 索引不正常，辅助器会回复 index error. You can run the "show conn" command to view connection information.

如果辅助器为完成 SDP 的流程，辅助器会回复 wait sdp discovery ending.

如果连接的 Sink 设备，没有 BASS 服务，辅助器会回复 connect device not supported BASS Server.

如果连接的 Sink 设备，没有 PACS 服务，辅助器会回复 connect device not supported PACS Server.

如果一切正常，辅助器会回复 assistant start scan broadcast source.

当辅助器扫描到符合要求的源信息时，会上报信息，如下是一个示例。其中 [1] 中 1 是 source_idx，在后续源的添加中会使用到。LC3 编解码器 ID 是 0600000000，源加密与否，每个 BIS 音频采样率和声道信息等。

```
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>

停止扫描源信息操作，辅助器回复 assistant stop scan broadcast source.

Telink Semiconductor

(3) scan-bcast clear <conn_idx>

停止扫描源信息操作，并且清空已经扫描到的所有源信息，辅助器回复 assistant clear broadcast source info.

5.1.4 对已连接的 sink 添加源

为已连接的 sink 添加源的指令格式: add-source <conn_idx> <source_idx> <bis_sync> [broadcast_key]\r\n

其中 conn_idx 目前默认是 1，user 可以根据 show conn 指令查询索引。

source_idx 是在扫描源操作时上报的，user 也可以根据 show source 指令查询索引。

bis_sync 是表示源信息的 BIS 同步信息，这些是在扫描源操作时上报的。如想要同步 BIS index 1，bis_sync=BIT(0); 想要同步 BIS index 2, bis_sync=BIT(1); 想要同步 BIS index1 和 2，bis_sync=BITS(0, 1) 即可。

该指令如果对方已经有同步的源信息，会先进行移除源信息后重新添加新的源信息。

添加源的示例指令如: add-source 1 1 3

辅助器会上报，sink 的周期性广播同步和 BIS 同步状态都会上报。

```
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 查询信息指令集合

查询信息指令格式: show <conn|sink|source|vcp> [conn_dev]\r\n

(1) show conn

查询已连接 sink 信息，辅助器回复 connect sink info. 打印已连接 sink 的信息。

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

(2) show sink

查询已扫描到的 sink 信息，辅助器回复 scan sink info. 打印已扫描到的 sink 信息。

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

(3) show source

查询已扫描到的 source 信息，辅助器回复并且打印 source 信息。

```
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]

查询已连接的 sink 设备的音量控制协议信息，包括 VCS 和 VOCS 信息。如果设备有 VCS 信息，可以控制音量，如果有 VOCS 信息，可以控制每个输出的音量偏移。

```
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 音量控制指令

音量控制指令，分为静音/取消静音、音量加、音量减、设置绝对音量四个操作。

所有的音量控制指令，操作完成/sink 端自身音量变量，都会打印 sink 的音量值、静音标记信息。

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

(1) 静音/取消静音

指令格式：mute <conn_idx>

指令默认会静音和取消静音翻转设置。

(2) 音量加

指令格式：vol+ <conn_idx>

默认会发送音量加，具体 sink 端音量增加多少，需要看 sink 端的设置。

(3) 音量减

指令格式：vol- <conn_idx>

默认会发送音量减，具体 sink 端音量减少多少，需要看 sink 端的设置。

(4) 设置绝对音量

指令格式：set-vol <conn_idx> <volume value>

默认会将 volume value 值，设置为 sink 的音量，范围是 0-255.

5.1.7 音量偏移控制

如果对端 Sink 设备拥有 VOCS 服务，可以设置单独设置输出音量偏移值。指令格式：set-vol-offset <conn_idx> <vocs_idx> <vol_offset>

vocs_idx 可以使用 show vcp <conn_idx> 查询。

vol_offset 偏移值返回为-255 到 255.

设置音量偏移或者 sink 端自行修改音量偏移值，都会打印。

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

5.1.8 示例

本小节以一个 assistant，两个 sink，两个 source 为例，讲述如何为两个 sink 配置不同的 source 信息。

sink 的基本信息：

sink 1, public 地址：12:34:56:00:00:00 设备名：Telink-BIS-SINK

sink 2, public 地址：12:34:56:00:00:01 设备名：Telink-BIS-SINK

source 的基本信息：

source 1, public 地址：12:34:56:00:01:00 设备名：Telink-BIS-SOURCE

source 2, public 地址：12:34:56:00:01:01 设备名：Telink-BIS-SOURCE

(1) 扫描空中的 sink 设备

下面显示了 sink 扫描的流程，首先发送 scan-sink start 指令，当扫描到期望的 sink 设备时，发送 scan-sink stop 指令停止当前扫描，可以通过 show sink 指令打印当前扫描到的 sink 信息。

```
[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
```

Telink Semiconductor

(2) 与 sink 1 创建 ACL 连接

根据上面扫描到的 sink 信息，可以看到 sink 1 的 dev_idx 为 1，所以发现 conn-sink 1 指令，可以与 sink 1 建立 ACL 连接。

```
[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
```

在创建 ACL 连接后，会打印 sink 支持的服务信息，包括 PACS、BASS、VCP 信息。如果包括 BASS 服务，会打印周期性广播和 BIS 广播同步情况。上面示例 sink 没有任何同步信息。

当 assistant 上报 SDP over 时，表示完成了服务发现的流程，可以进行其他操作。回连和配对的服务发现时间差异较大，user 可以通过抓包自行查看。

(3) 为 sink 1 扫描 source 信息

上述日志能看到 sink 1 连接后的 conn_idx 为 1，所以发送 scan-bcast start 1 命令扫描空中符合的 source 信息，当扫描到符合要求的 source 信息后，发送 scan-bcast stop1 命令，停止扫描 source。

```
[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
```

上述日志显示了空中符合要求的 source 信息。

(4) 为 sink 1 添加/切换 source 信息

根据前面的信息，source 1 的 source_idx 为 1，source 2 的 source_idx 为 2，并且两者都是不加密的。

下面日志的操作步骤，user 可以根据 sink 端听到的声音来判断有无操作成功。

- 同步 source 1 左右声道音频；
- 同步 source 2 左右声道音频；
- 同步 source 1 左声道音频；
- 同步 source 2 左声道音频；
- 同步 source 1 右声道音频；
- 同步 source 2 右声道音频；

```
//同步 source 1 左右声道音频;
[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
//同步 source 2 左右声道音频;
[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
```

Telink Semiconductor

```
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
//同步 source 1 左声道音频;
[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
//同步 source 2 左声道音频;
[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
//同步 source 1 右声道音频;
[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
////同步 source 2 右声道音频;
[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) 音量控制的操作

当完成服务发现流程后，可以发送 show vcp 1 指令，查询 sink 支持的音量控制协议。连接的 sink 是支持音量控制、静音、左声道音量增益 (sink 端应用层未实现)，右声道音量增益 (sink 端应用层未实现)。

```
[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
```

当 sink 端按键跳转音量时，assistant 端会打印当前的音量信息。

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

assistant 也可以发送音量加、音量减，静音等指令。

```
//音量加
[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
//音量减
[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
//静音/取消静音
[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
//设置音量
[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
```

sink 端支持 VOCS 服务，所以也可以设置 VOCS 值。

```
//设置左声道音量偏移值
[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
//设置右声道音量偏移值
[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
////查询音量参数
[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
```