Low Energy Audio – Incoming Call Over Media (Multi-Device)

Bluetooth® Informational Publication

- **Version:** v1
- **Version Date:** 2023-02-14
- **Prepared By:** Generic Audio Working Group

If you have any questions or comments on this informational publication or Bluetooth LE Audio specifications, please submit a ticket:
https://www.bluetooth.com/leaudio-spec-feedback/

If you are a SIG member and have any specific requested changes to this document, please submit an erratum in Jira:
https://bluetooth.atlassian.net/jira/software/c/projects/ES/issues

Abstract:

This document uses a Message Sequence Chart (MSC) to explain the Bluetooth LE Audio use case where a user listening to music from their PC on their earbuds is interrupted by an incoming call from their phone.

Version History

Version Number	Date (yyyy-mm-dd)	Comments
v1	2023-02-14	Approved by the Bluetooth SIG Board of Directors.

Acknowledgments

Name	Company
Andrew Estrada	Sony
Nick Hunn	GN Hearing A/S
Riccardo Cavallari	WSA
Georg Dickmann	Sonova AG
Frank Yerrace	Microsoft
Jonathan Tanner	Qualcomm Technologies International, Ltd

This document, regardless of its title or content, is not a Bluetooth Specification as defined in the Bluetooth Patent/Copyright License Agreement (“PCLA”) and Bluetooth Trademark License Agreement. Use of this document by members of Bluetooth SIG is governed by the membership and other related agreements between Bluetooth SIG Inc. (“Bluetooth SIG”) and its members, including the PCLA and other agreements posted on Bluetooth SIG’s website located at www.bluetooth.com.

THIS DOCUMENT IS PROVIDED “AS IS” AND BLUETOOTH SIG, ITS MEMBERS, AND THEIR AFFILIATES MAKE NO REPRESENTATIONS OR WARRANTIES AND DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING ANY WARRANTY OF MERCHANTABILITY, TITLE, NON-INFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, THAT THE CONTENT OF THIS DOCUMENT IS FREE OF ERRORS.

TO THE EXTENT NOT PROHIBITED BY LAW, BLUETOOTH SIG, ITS MEMBERS, AND THEIR AFFILIATES DISCLAIM ALL LIABILITY ARISING OUT OF OR RELATING TO USE OF THIS DOCUMENT AND ANY INFORMATION CONTAINED IN THIS DOCUMENT, INCLUDING LOST REVENUE, PROFITS, DATA OR PROGRAMS, OR BUSINESS INTERRUPTION, OR FOR SPECIAL, INDIRECT, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, AND EVEN IF BLUETOOTH SIG, ITS MEMBERS, OR THEIR AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This document is proprietary to Bluetooth SIG. This document may contain or cover subject matter that is intellectual property of Bluetooth SIG and its members. The furnishing of this document does not grant any license to any intellectual property of Bluetooth SIG or its members.

This document is subject to change without notice.

Contents

1 Introduction

This document helps developers understand how to use Bluetooth Low Energy (LE) Audio specifications to fulfil a common use case. This document uses a Message Sequence Chart (MSC), subdivided into 12 parts, shown in Figure 2.1 through Figure 2.12, to explain the steps for the Bluetooth LE Audio use case in which a user is listening to music from a PC on their earbuds and is interrupted by an incoming call from their phone. While each of the LE Audio specifications addresses only individual aspects needed to fulfill the given use case, this document shows how the concepts from multiple LE Audio specifications are combined.

This is not a Bluetooth specification, therefore, the established Bluetooth SIG specification language conventions for use of the words **shall, shall not, must, should, should not, may,** and **can** do not apply to this document.

1.1 Scope

The MSC in this document shows one possible implementation using the features within the Bluetooth LE Audio specifications. It illustrates many of the concepts of LE Audio specifications, such as the use of Context Types, set up of data paths, configuration of Audio Stream Endpoints, use of BAP Announcements, etc. The solution illustrated is not the only possible solution, and developers will use alternative approaches based on other considerations. The purpose of this document is to illustrate one possible approach and to share the experience of the members of the Generic Audio Working Group.

1.2 Conventions

The conventions and notations listed in Sections 1.2.1 and 1.2.2 are used in this document.

1.2.1 Audio Stream notation

In this document, <<X>> Audio Stream represents a unicast Audio Stream in which the value of Streaming_Audio_Contexts is set to <<X>>. For example, in a <<Ringtone>> Audio Stream, the Streaming_Audio_Contexts value is <<Ringtone>>.

1.2.2 MSCs

The MSCs in this document use the following conventions:

- Audio Stream Endpoint (ASE) states are shown in rounded grey boxes. A tag in the upper left corner of each box shows the client to which the server exposes the state.
- Content Control Service states are shown in rounded white boxes. A tag in the upper left corner of each box shows the type of service: Media Control Service (MCS) [1] or Telephone Bearer Service (TBS) [2].

- Descriptions that refer to the in-progress process are shown in yellow boxes that span across the relevant devices.
- Advertisements are shown as dashed arrows starting from the transmitters. The direction of the arrow does not imply reception by a specific device.
- CAP procedures that encapsulate multiple steps are surrounded by large light-grey boxes. A tag in the upper left corner of each box calls out the name of the CAP procedure.

2 Incoming call over media (multi-device)

The MSC in Figure 2.1 through Figure 2.12 shows an example in which a PC is sending music to a pair of earbuds when a phone call arrives through a phone that is bonded with the earbuds but currently not connected.

The device entities called out in the MSC list the roles supported by each of the entities. The roles supported by the PC correspond to the Telephony and Media Audio Profile (TMAP) [8] role of Unicast Media Sender. The roles supported by the earbuds correspond to the TMAP roles of Unicast Media Receiver and Call Terminal. The roles supported by the phone correspond to the TMAP Call Gateway.

When the call arrives, the phone establishes a connection with the earbuds and sends an in-band ringtone to both earbuds. Because the earbuds in this example can establish isochronous channels with only one Initiator at a time, the earbuds of this example decide to release the music audio from the PC to free the necessary resources to support the in-band ringtone from the phone.

After the user accepts the call using the earbuds' user interface (UI), the phone changes the Streaming Audio Contexts values of the <<Ringtone>> Audio Streams (which carry the in-band ringtone). The phone switches the value from <<Ringtone>> to <<Conversational>>. In addition, the phone starts a <<Conversational>> Audio Stream from one earbud to the phone to carry the voice signal from the earbud's microphone.

When the user ends the call, the PC restarts the original <<Media>> Audio Streams and resumes sending music.

2.1 Initial conditions

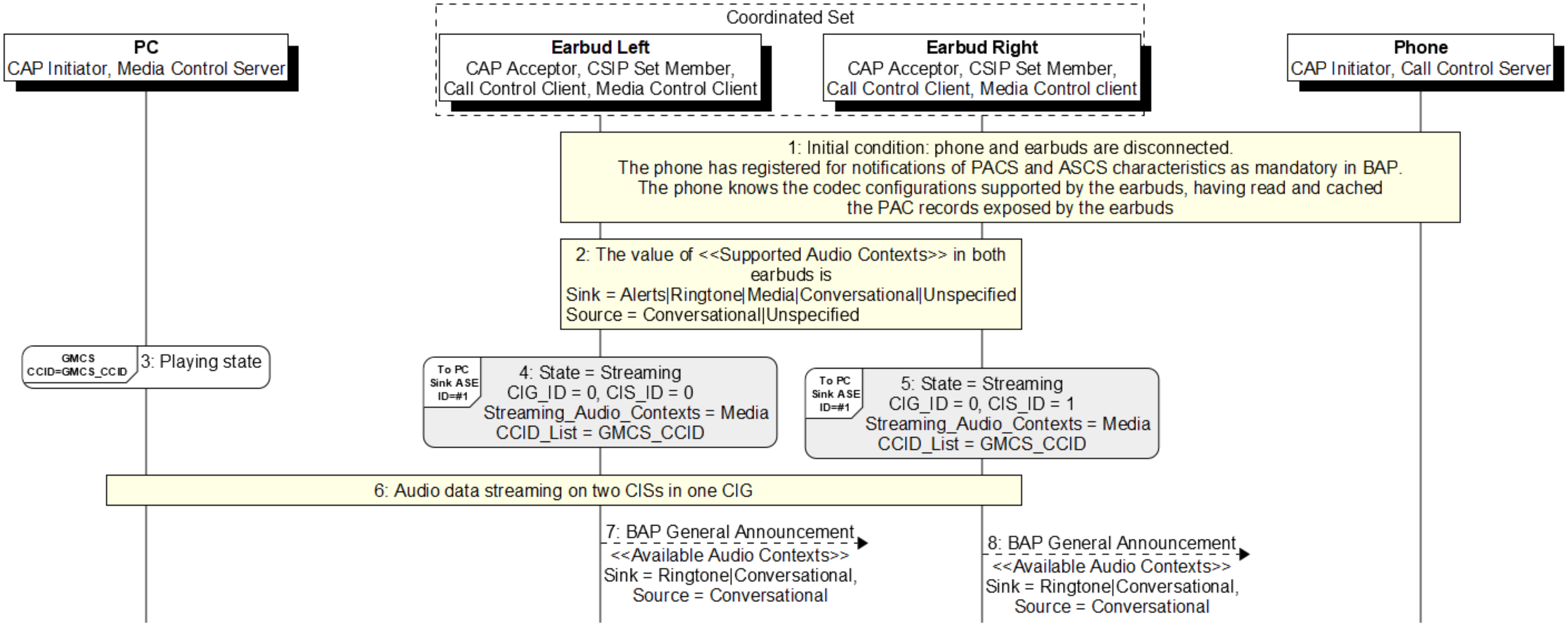
Coordinated Set
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
1: Initial condition: phone and earbuds are disconnected.
The phone has registered for notifications of PACS and ASCS characteristics as mandatory in BAP.
The phone knows the codec configurations supported by the earbuds, having read and cached
the PAC records exposed by the earbuds
2: The value of <<Supported Audio Contexts>> in both
earbuds is
Sink = Alerts|Ringtone|Media|Conversational|Unspecified
Source = Conversational|Unspecified
GMCS
CCID=GMCS_CCID
3: Playing state
To PC
Sink ASE
ID=#1
4: State = Streaming
CIG_ID = 0, CIS_ID = 0
Streaming_Audio_Contexts = Media
CCID_List = GMCS_CCID
To PC
Sink ASE
ID=#1
5: State = Streaming
CIG_ID = 0, CIS_ID = 1
Streaming_Audio_Contexts = Media
CCID_List = GMCS_CCID
6: Audio data streaming on two CISs in one CIG
7: BAP General Announcement
<<Available Audio Contexts>>
Sink = Ringtone|Conversational,
Source = Conversational
8: BAP General Announcement
<<Available Audio Contexts>>
Sink = Ringtone|Conversational,
Source = Conversational

Figure 2.1: Incoming call while playing media, part 1 of 12

Figure 2.1 shows the initial conditions:

- The values chosen for Supported Audio Contexts are implementation specific.
- The earbuds are rendering <<Media>> Audio Streams transmitted by the PC, using Audio Configuration 6(ii) (see Section 4.4 in BAP [3]).
- The availability values signaled through Basic Audio Profile (BAP) General Announcements are implementation specific [3]. In this case, they have been chosen to allow Initiators other than the PC to start <<Ringtone>> or <<Conversational>> Audio Streams, which will replace the ongoing <<Media>> Audio Stream. Note that in this example both earbuds continue to transmit BAP General Announcements throughout the scenario.

2.2 Phone signals incoming call

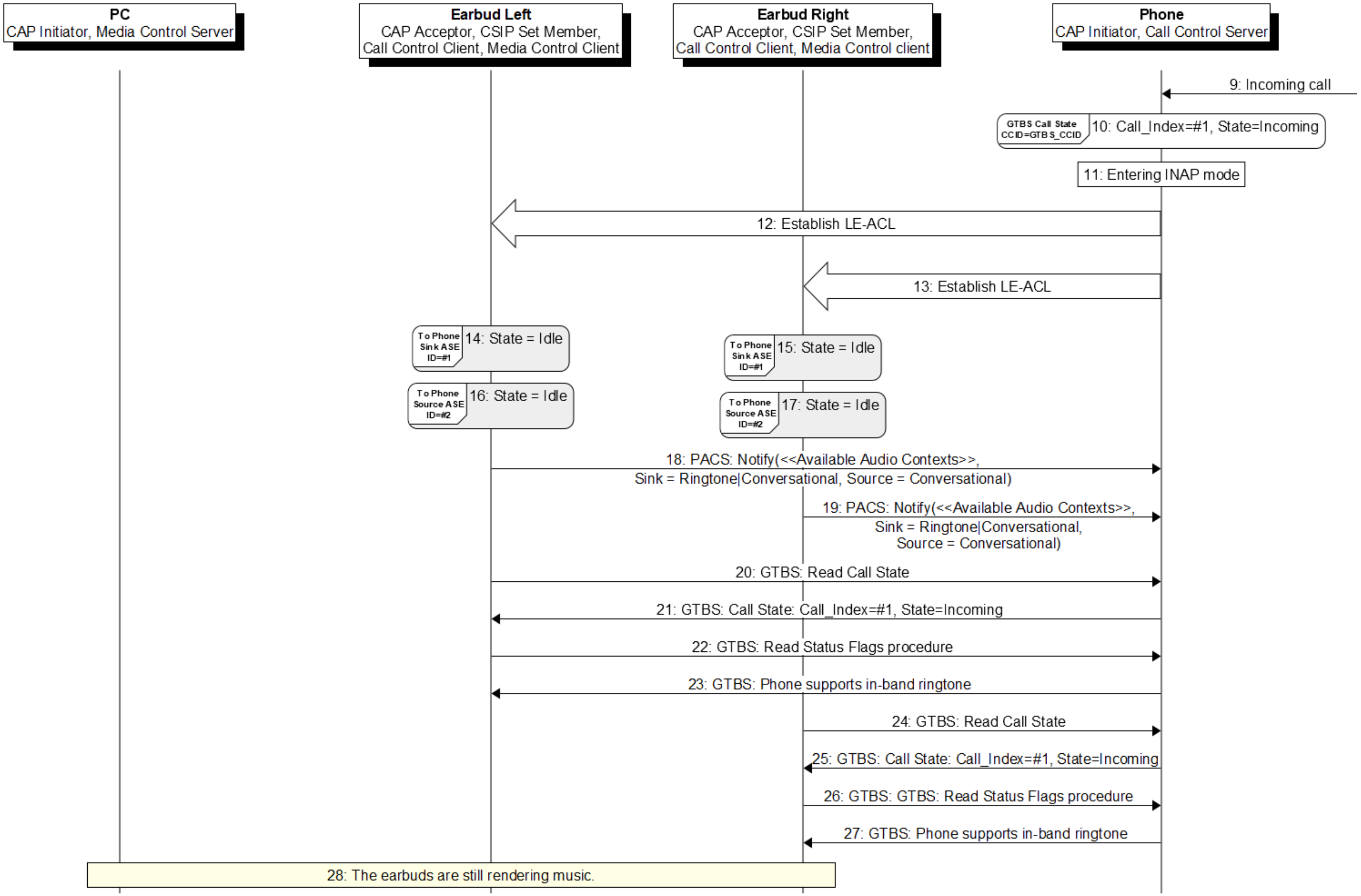
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
9: Incoming call
GTBS Call State
CCID=GTBS_CCID
10: Call_Index=#1, State=Incoming
11: Entering INAP mode
12: Establish LE-ACL
13: Establish LE-ACL
To Phone
Sink ASE
ID=#1
14: State = Idle
To Phone
Sink ASE
ID=#1
15: State = Idle
To Phone
Source ASE
ID=#2
16: State = Idle
To Phone
Source ASE
ID=#2
17: State = Idle
18: PACS: Notify(<<Available Audio Contexts>>,
Sink = Ringtone|Conversational, Source = Conversational)
19: PACS: Notify(<<Available Audio Contexts>>,
Sink = Ringtone|Conversational,
Source = Conversational)
20: GTBS: Read Call State
21: GTBS: Call State: Call_Index=#1, State=Incoming
22: GTBS: Read Status Flags procedure
23: GTBS: Phone supports in-band ringtone
24: GTBS: Read Call State
25: GTBS: Call State: Call_Index=#1, State=Incoming
26: GTBS: GTBS: Read Status Flags procedure
27: GTBS: Phone supports in-band ringtone
28: The earbuds are still rendering music.

Figure 2.2: Incoming call while playing media, part 2 of 12

Figure 2.2 shows the following steps:

- An incoming call causes the phone to enter Common Audio Profile (CAP) Immediate Need for Audio related Peripheral (INAP) mode [4]. The phone receives BAP General Announcements from the earbuds, which signal availability for <<Ringtone>> as needed to send an in-band ringtone. The phone establishes asynchronous connections (ACL) to send a ringtone to both earbuds.
- In this case, the ASEs exposed by the earbuds to the phone were already in the Idle state when the phone was last connected to the earbuds. Therefore, the earbuds do not notify the state of their ASEs to the phone. Note that the earbuds expose ASE characteristics for each client. In this example, the earbuds use the ASE_ID 1 for both a Sink ASE exposed to the PC and for a Sink ASE exposed to the phone.
- In this example, the values of the Available Audio Contexts characteristics (see Published Audio Capabilities Service—PACS [5]) on both earbuds have changed since the last connection with the phone because the PC started Media Audio Streams. Therefore, the earbuds limited their availability to include fewer Context Type values.
- Because the phone has no obligation to notify a call state change in the time since the earbuds were last connected, the earbuds read the TBS call state and the TBS status flags [2] from the phone. Because the phone supports in-band ringtone and the earbuds are available for <<Ringtone>> Audio Streams, the earbuds do not need to generate out-of-band ringtones. The phone will send <<Ringtone>> Audio Streams for the incoming call as shown in later sections.
- The earbuds' availability as signaled through both the <<Available Audio Contexts>> characteristics and BAP General Announcements reflect that fact that the earbuds will accept only Audio Streams with specific values for Streaming_Audio_Contexts such that the ongoing Media Audio Stream will not be interrupted by arbitrary Audio Streams from devices other than the PC. Note that if an incoming call was arriving on an Initiator which, due to limitations of its audio architecture, associates any audio with the Context Type value <<Unspecified>>, the unavailability of the earbuds for <<Unspecified>> informs the Initiator that it should not send a ringtone or call audio to the earbuds.

2.3 Phone transitions earbud's ASEs into Codec Configured state

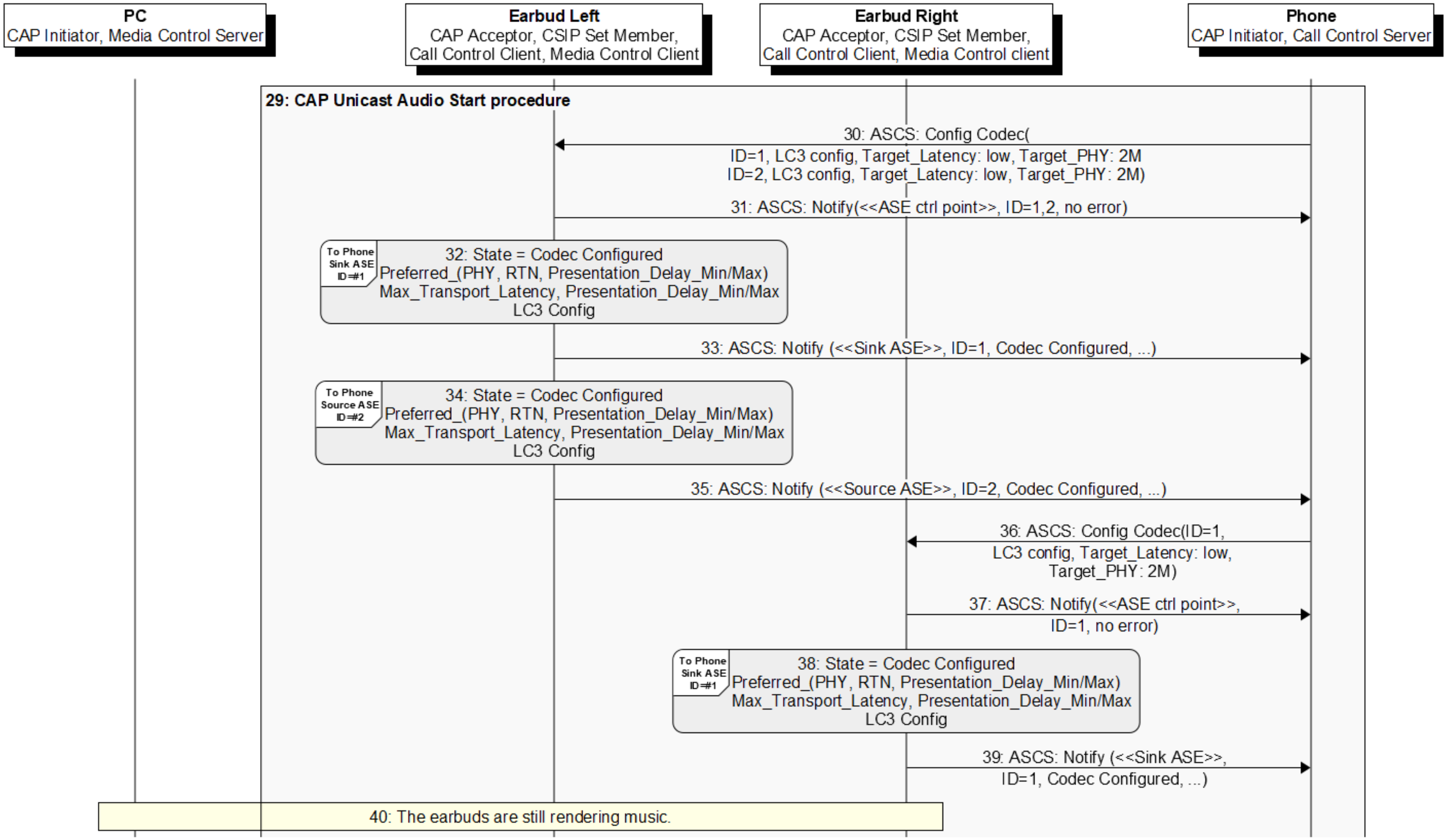
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member, Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member, Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
29: CAP Unicast Audio Start procedure
30: ASCS: Config Codec(ID=1, LC3 config, Target_Latency: low, Target_PHY: 2M ID=2, LC3 config, Target_Latency: low, Target_PHY: 2M)
31: ASCS: Notify(<<ASE ctrl point>>, ID=1,2, no error)
To Phone Sink ASE ID=#1
32: State = Codec Configured Preferred_(PHY, RTN, Presentation_Delay_Min/Max) Max_Transport_Latency, Presentation_Delay_Min/Max LC3 Config
33: ASCS: Notify (<<Sink ASE>>, ID=1, Codec Configured, ...)
To Phone Source ASE ID=#2
34: State = Codec Configured Preferred_(PHY, RTN, Presentation_Delay_Min/Max) Max_Transport_Latency, Presentation_Delay_Min/Max LC3 Config
35: ASCS: Notify (<<Source ASE>>, ID=2, Codec Configured, ...)
36: ASCS: Config Codec(ID=1, LC3 config, Target_Latency: low, Target_PHY: 2M)
37: ASCS: Notify(<<ASE ctrl point>>, ID=1, no error)
To Phone Sink ASE ID=#1
38: State = Codec Configured Preferred_(PHY, RTN, Presentation_Delay_Min/Max) Max_Transport_Latency, Presentation_Delay_Min/Max LC3 Config
39: ASCS: Notify (<<Sink ASE>>, ID=1, Codec Configured, ...)
40: The earbuds are still rendering music.

Figure 2.3: Incoming call while playing media, part 3 of 12

Figure 2.3 shows the following steps:

- The phone sets the ASEs on the earbuds to the Codec Configured state, targeting Audio Configuration 8(ii) (see Section 4.4 in BAP [3]).
- In this example, the phone immediately configures the Source ASE of the left earbud, anticipating the need for a <<Conversational>> Audio Stream from the earbud to the phone when the call is accepted. A single Config Codec operation is used to simultaneously configure multiple ASEs, which saves time compared to later configuration in a separate Config Codec operation.
  - Alternatively, the phone would perform configuration after the call is accepted.
  - In other alternatives, the phone would also have configured the Source ASE of the right earbud or the Source ASEs on both earbuds,
  - or the phone would have picked up the user’s voice through the microphone of the phone and not configured a unicast Audio Stream from the earbud at all.
- In the Config Codec operation, the phone sets the Target_Latency argument to “low", reflecting the low-latency needs of a phone call. While accepting the request, the earbuds return, as Additional_ASE_Parameters, their preferences on the setting of PHY, the number of retransmissions, and the minimum and maximum presentation delay. However, the client will potentially use different values in the subsequent Config QoS operation. The earbuds also return values for Max_Transport_Latency as well as Presentation_Delay_Min and Presentation_Delay_Max. These parameters represent upper (Max_Transport_Latency and Presentation_Delay_Max) and lower bounds (Presentation_Delay_Min) for the corresponding parameters that the client will potentially use in a subsequent Config QoS operation. The phone uses a max transport latency value that is no greater than the lowest value of Max_Transport_Latency that the earbuds have respectively exposed in the Codec Configured state (see Section 7.2.1 in BAP [3]). The phone will typically use a smaller value in the subsequent Config QoS operation to achieve low latency for the upcoming phone call.

2.4 Phone transitions earbud's ASEs into QoS Configured state

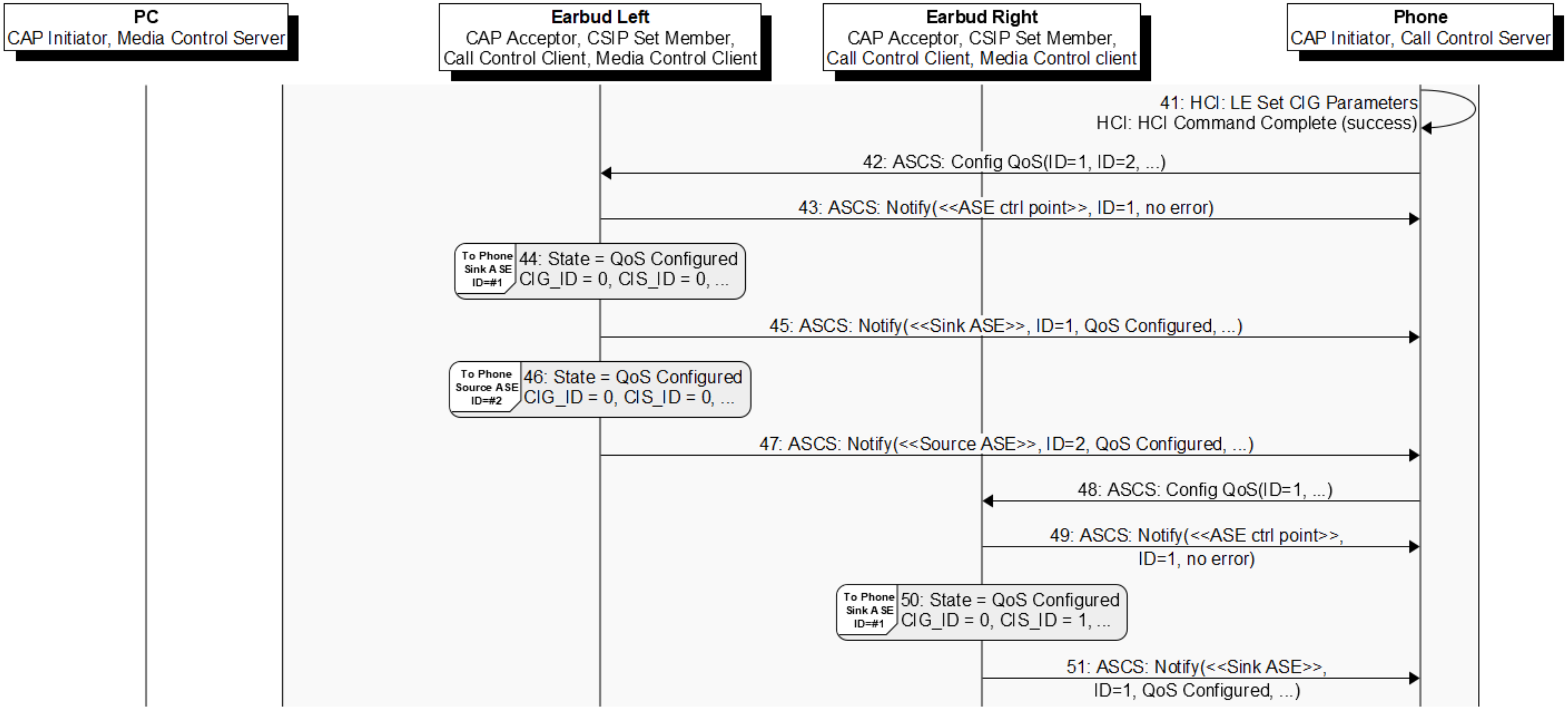
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
41: HCI: LE Set CIG Parameters
HCI: HCI Command Complete (success)
42: ASCS: Config QoS(ID=1, ID=2, ...)
43: ASCS: Notify(<<ASE ctrl point>>, ID=1, no error)
To Phone Sink ASE ID=#1
44: State = QoS Configured
CIG_ID = 0, CIS_ID = 0, ...
45: ASCS: Notify(<<Sink ASE>>, ID=1, QoS Configured, ...)
To Phone Source ASE ID=#2
46: State = QoS Configured
CIG_ID = 0, CIS_ID = 0, ...
47: ASCS: Notify(<<Source ASE>>, ID=2, QoS Configured, ...)
48: ASCS: Config QoS(ID=1, ...)
49: ASCS: Notify(<<ASE ctrl point>>,
ID=1, no error)
To Phone Sink ASE ID=#1
50: State = QoS Configured
CIG_ID = 0, CIS_ID = 1, ...
51: ASCS: Notify(<<Sink ASE>>,
ID=1, QoS Configured, ...)

Figure 2.4: Incoming call while playing media, part 4 of 12

Figure 2.4 shows the following steps:

- The phone has collected the Quality of Service (QoS) preferences of the earbuds (as exposed by the ASEs) and selects a set of Connected Isochronous Group (CIG)/Connected Isochronous Stream (CIS) [6] parameters to submit to the phone's controller. If the submission fails, the phone will retry with different parameters.
- In this example, the phone submits a configuration that includes a bidirectional CIS, such that the CIS configuration supports a unicast Audio Stream from an earbud to the phone as an audio return channel.
- The phone sets the ASEs on the earbuds to the QoS Configured state. The phone already includes the Source ASE on the left earbud even though the ASE is not needed for the upcoming in-band ringtone. Similar to the earlier transition to the Codec Configured state, the phone only proceeds to operate on the Source ASE after establishing the call.
- Note that a phone is allowed to perform all these operations on the earbuds simply to be ready when a call arrives, even if no incoming call was signaled. This early action potentially provides lower latency between detection of an incoming call and playing a ringtone. However, this early action also uses up radio bandwidth for both the phone and the earbuds because the ACL connection between the phone and the earbuds needs to be permanently maintained; otherwise the ASEs will not remain in QoS Configured state (see Audio Stream Control Service—ASCS, Section 3.2 [7]).

2.5 Phone requests transition of earbud's ASEs to Enabling state

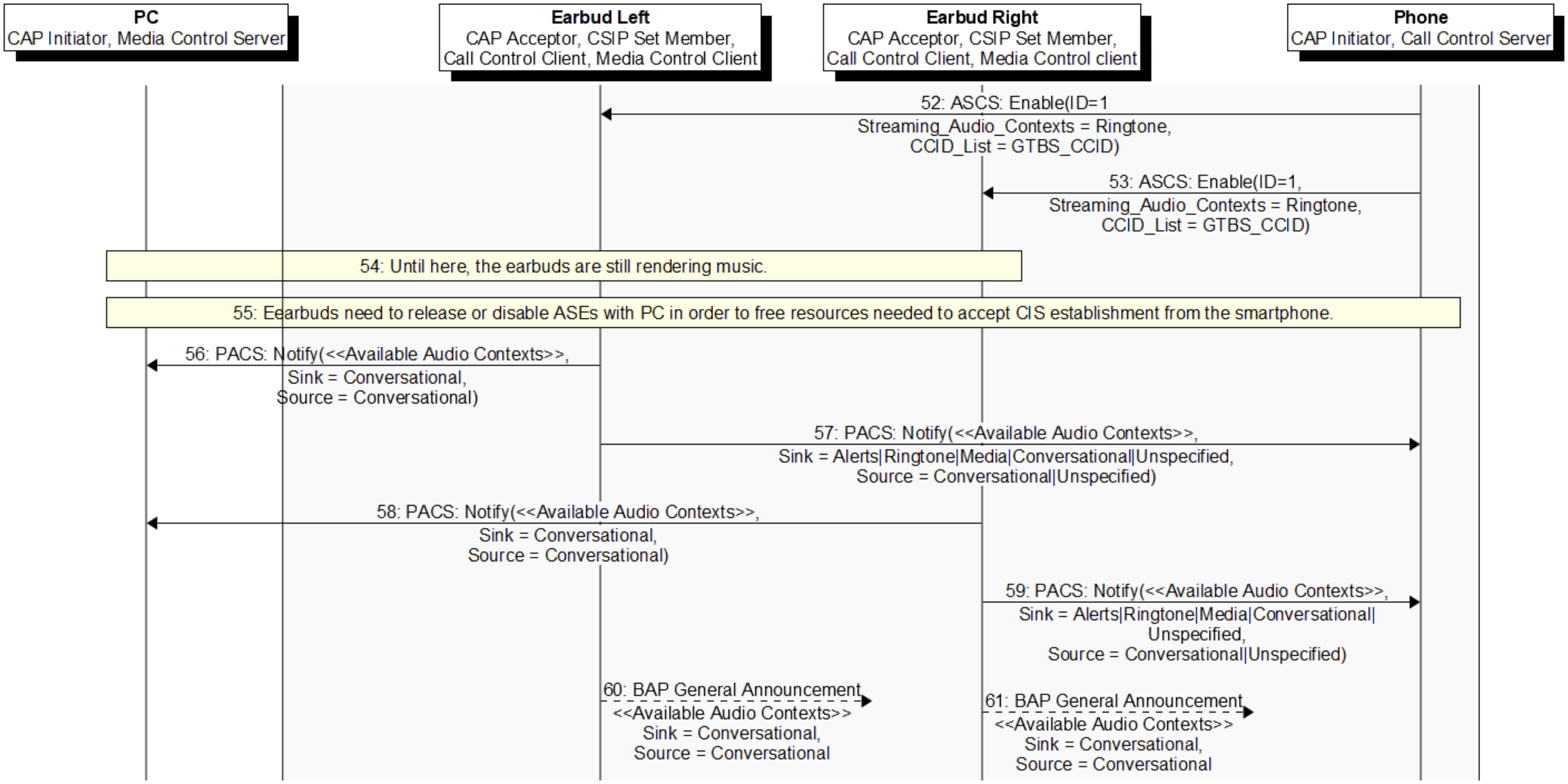
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
52: ASCS: Enable(ID=1
Streaming_Audio_Contexts = Ringtone,
CCID_List = GTBS_CCID)
53: ASCS: Enable(ID=1,
Streaming_Audio_Contexts = Ringtone,
CCID_List = GTBS_CCID)
54: Until here, the earbuds are still rendering music.
55: Earbuds need to release or disable ASEs with PC in order to free resources needed to accept CIS establishment from the smartphone.
56: PACS: Notify(<<Available Audio Contexts>>,
Sink = Conversational,
Source = Conversational)
57: PACS: Notify(<<Available Audio Contexts>>,
Sink = Alerts|Ringtone|Media|Conversational|Unspecified,
Source = Conversational|Unspecified)
58: PACS: Notify(<<Available Audio Contexts>>,
Sink = Conversational,
Source = Conversational)
59: PACS: Notify(<<Available Audio Contexts>>,
Sink = Alerts|Ringtone|Media|Conversational|
Unspecified,
Source = Conversational|Unspecified)
60: BAP General Announcement
<<Available Audio Contexts>>
Sink = Conversational,
Source = Conversational
61: BAP General Announcement
<<Available Audio Contexts>>
Sink = Conversational,
Source = Conversational

Figure 2.5: Incoming call while playing media, part 5 of 12

Figure 2.5 shows the following steps:

- The phone sets the Sink ASEs on the earbuds to the Enabling state. This procedure informs the earbuds about the purpose (Streaming_Context_Types = <<Ringtone>>) of the upcoming Audio Streams and about the content control service (CCID_List = TBS_CCID) through which the earbuds control the call. Note that the phone does not include the Source ASE on the left earbud in this transition because there is not yet a need for an audio return channel.
- Based on the purpose of the upcoming <<Ringtone>> Audio Streams, the earbuds update their availability. They limit availability to the PC to <<Conversational>> only, while signaling availability for all Context Type values that they support to the phone. The limitation of the earbuds' availability to the PC is an implementation-specific configuration which reflects the desire of the earbuds that the upcoming <<Ringtone>> Audio Stream from the phone can only be interrupted by audio associated with an active call. It is also an implementation-specific configuration to signal different availabilities to different clients (see Section 3.5.1 in PACS [5]) and to be available for all supported Context Type values to the client with which Audio Streams are established or about to be established. While the earbuds update their availability to the PC, the earbuds also update the availability that they signal through BAP General Announcements. While the phone receives different availability information from the <<Available Audio Contexts>> characteristics and from BAP General Announcements, it ignores the availability information from the BAP General Announcements, giving precedence to the values exposed through <<Available Audio Contexts>> knowing that the earbuds can signal different availability to different clients through the <<Available Audio Contexts>> characteristics while the availability information propagated through undirected BAP General Announcements is the same for all recipients of the BAP General Announcements.
- The PC does not react to the updated availability of the earbuds (see Section 7.1.1 in CAP [4]).

2.6 Earbuds release ASEs as exposed to PC

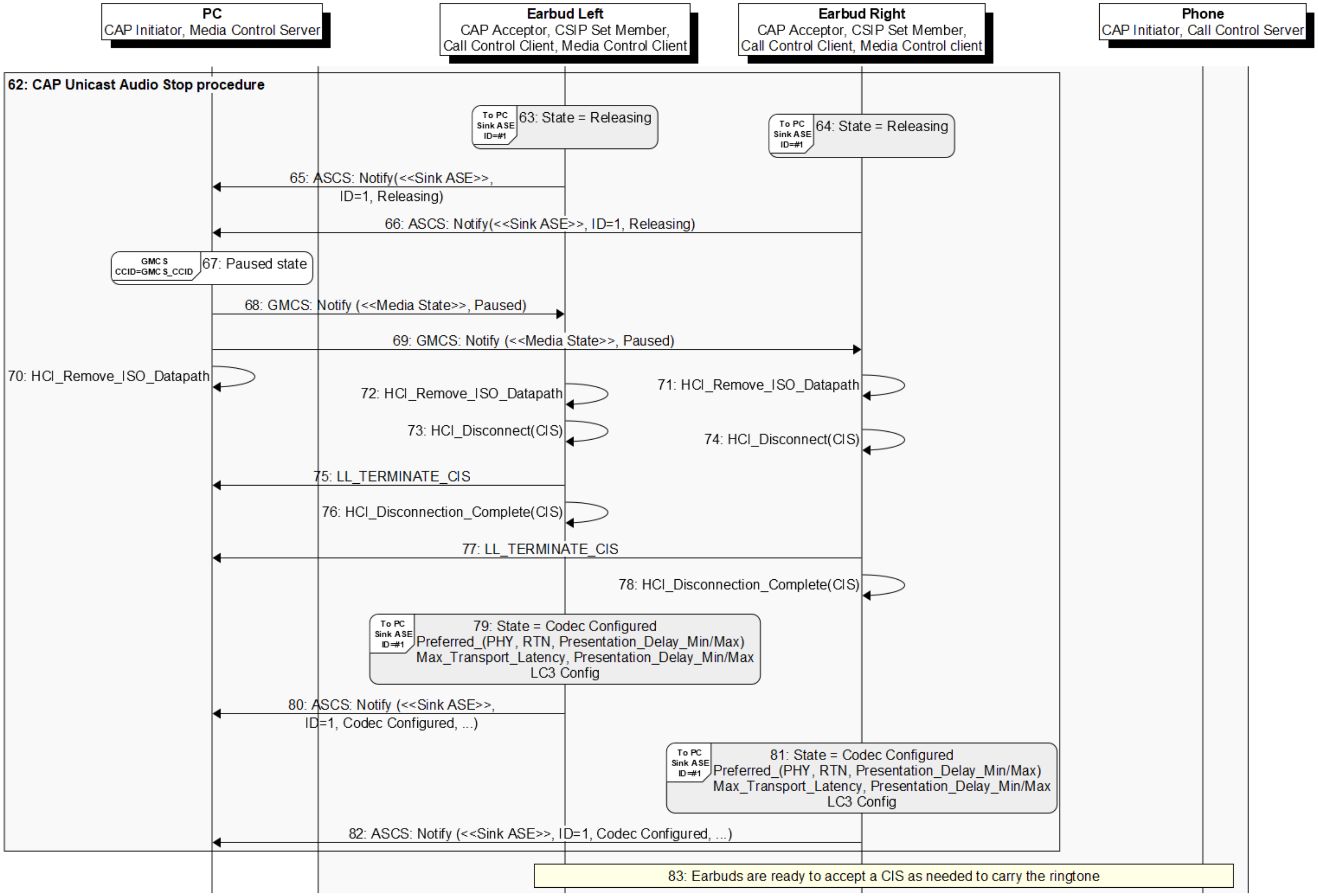
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member, Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member, Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
62: CAP Unicast Audio Stop procedure
To PC Sink ASE ID=#1
63: State = Releasing
To PC Sink ASE ID=#1
64: State = Releasing
65: ASCS: Notify(<<Sink ASE>>, ID=1, Releasing)
66: ASCS: Notify(<<Sink ASE>>, ID=1, Releasing)
GMCS CCID=GMCS_CCID
67: Paused state
68: GMCS: Notify (<<Media State>>, Paused)
69: GMCS: Notify (<<Media State>>, Paused)
70: HCI_Remove_ISO_Datapath
71: HCI_Remove_ISO_Datapath
72: HCI_Remove_ISO_Datapath
73: HCI_Disconnect(CIS)
74: HCI_Disconnect(CIS)
75: LL_TERMINATE_CIS
76: HCI_Disconnection_Complete(CIS)
77: LL_TERMINATE_CIS
78: HCI_Disconnection_Complete(CIS)
To PC Sink ASE ID=#1
79: State = Codec Configured
Preferred_(PHY, RTN, Presentation_Delay_Min/Max)
Max_Transport_Latency, Presentation_Delay_Min/Max
LC3 Config
80: ASCS: Notify (<<Sink ASE>>, ID=1, Codec Configured, ...)
To PC Sink ASE ID=#1
81: State = Codec Configured
Preferred_(PHY, RTN, Presentation_Delay_Min/Max)
Max_Transport_Latency, Presentation_Delay_Min/Max
LC3 Config
82: ASCS: Notify (<<Sink ASE>>, ID=1, Codec Configured, ...)
83: Earbuds are ready to accept a CIS as needed to carry the ringtone

Figure 2.6: Incoming call while playing media, part 6 of 12

Figure 2.6 shows the following steps:

- Because the earbuds of this example are not capable of simultaneously maintaining CISs with more than one device, they set the ASEs exposed to the PC to the Releasing state.
  - An alternative is for the earbuds to transition to the QoS configured state, which would allow for faster resumption of Audio Streams from the PC after the call ends.
- In this example, the release of the ASEs in steps 61 to 64 of the MSC causes the media player on the PC to transition to the Paused state. This is an implementation-specific choice, which is entirely under the control of the Initiator. The decision made by the Initiator will depend on its current application and content. Different decisions will be made depending on whether the media stream was interruptible or shared with other Acceptors which are still connected.
- It is possible for the earbud to send a Pause opcode to the PC's Media Control Service. However, in this example, the earbuds do not send a Pause opcode to the PC's Media Control Service because the user did not use the earbuds' UI to explicitly request the media player to be paused. If the earbuds were capable of simultaneously maintaining Audio Streams with both the PC and the phone, they would probably send a Pause opcode to the PC unless they wanted to mix in the Media Audio Stream into the upcoming Ringtone Audio Stream.
- The earbuds disconnect the CISs established with the PC to free up the necessary resources to establish CISs with the phone. After disconnecting, the earbuds set the ASEs that are exposed to the PC to the Codec Configured state. The earbuds also cache the codec configuration that was used for the <<Media>> Audio Streams.
- Note that both the PC and the earbuds maintain the established ACL connections. This is an implementation-specific choice which allows them to resume the <<Media>> Audio Stream after the end of the call without the need to reconnect.

2.7 Earbuds start rendering the ringtone transmitted by the phone

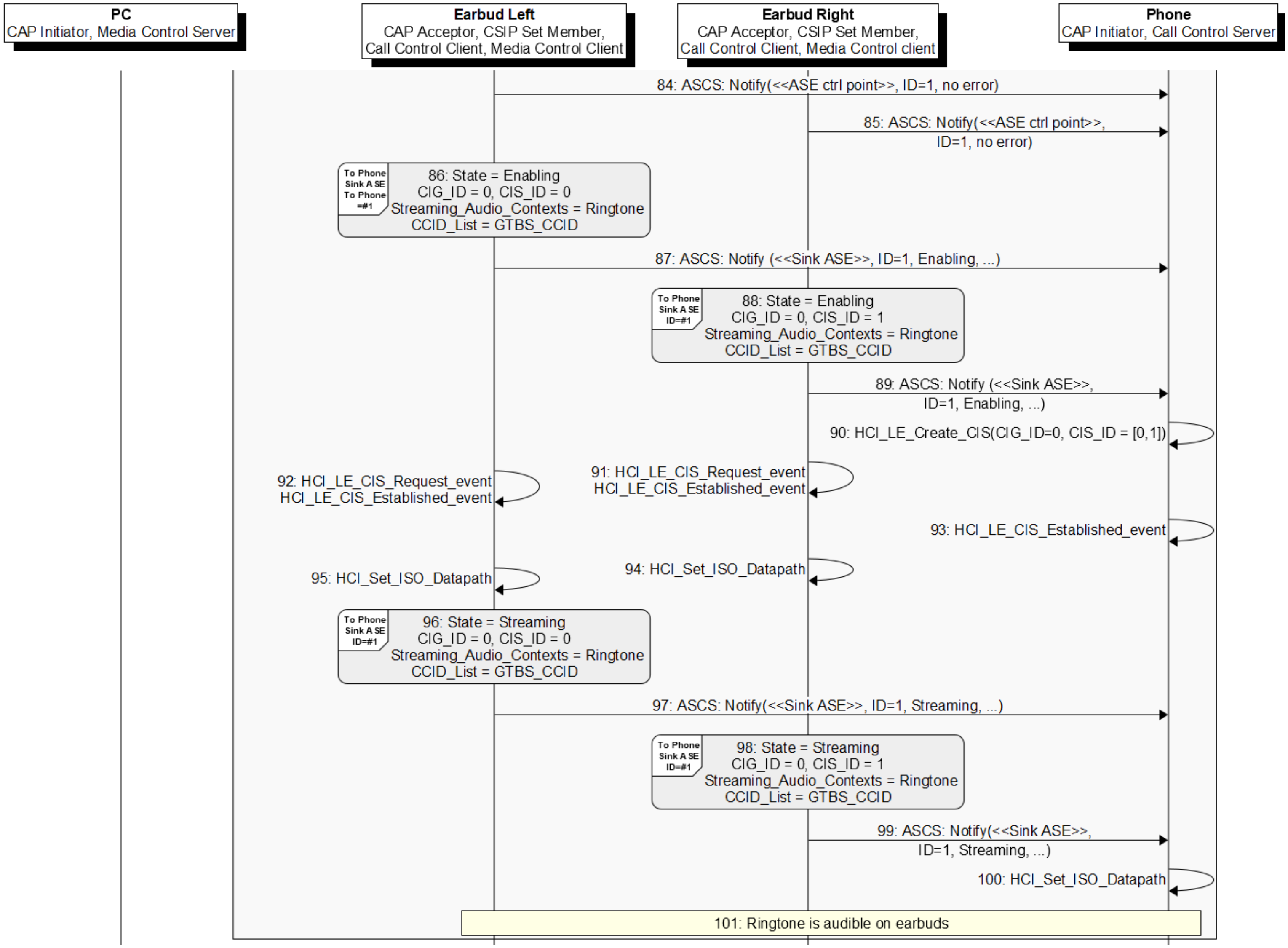
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
84: ASCS: Notify(<<ASE ctrl point>>, ID=1, no error)
85: ASCS: Notify(<<ASE ctrl point>>,
ID=1, no error)
To Phone
Sink ASE
To Phone
=#1
86: State = Enabling
CIG_ID = 0, CIS_ID = 0
Streaming_Audio_Contexts = Ringtone
CCID_List = GTBS_CCID
87: ASCS: Notify (<<Sink ASE>>, ID=1, Enabling, ...)
To Phone
Sink ASE
ID=#1
88: State = Enabling
CIG_ID = 0, CIS_ID = 1
Streaming_Audio_Contexts = Ringtone
CCID_List = GTBS_CCID
89: ASCS: Notify (<<Sink ASE>>,
ID=1, Enabling, ...)
90: HCI_LE_Create_CIS(CIG_ID=0, CIS_ID = [0,1])
91: HCI_LE_CIS_Request_event
HCI_LE_CIS_Established_event
92: HCI_LE_CIS_Request_event
HCI_LE_CIS_Established_event
93: HCI_LE_CIS_Established_event
94: HCI_Set_ISO_Datapath
95: HCI_Set_ISO_Datapath
To Phone
Sink ASE
ID=#1
96: State = Streaming
CIG_ID = 0, CIS_ID = 0
Streaming_Audio_Contexts = Ringtone
CCID_List = GTBS_CCID
97: ASCS: Notify(<<Sink ASE>>, ID=1, Streaming, ...)
To Phone
Sink ASE
ID=#1
98: State = Streaming
CIG_ID = 0, CIS_ID = 1
Streaming_Audio_Contexts = Ringtone
CCID_List = GTBS_CCID
99: ASCS: Notify(<<Sink ASE>>,
ID=1, Streaming, ...)
100: HCI_Set_ISO_Datapath
101: Ringtone is audible on earbuds

Figure 2.7: Incoming call while playing media, part 7 of 12

Figure 2.7 shows the following steps:

- The phone creates the necessary CISs to transport the <<Ringtone>> Audio Streams. Note that the phone creates a bidirectional CIS with the left earbud because the phone anticipates the need for an audio return channel after the call is accepted.
- The earbuds set their Sink ASEs to the Streaming state through an autonomously initiated Receiver Start Ready operation. This state indicates that both earbuds have previously configured their local audio paths and are ready to receive audio over the newly established CISs.

2.8 User accepts the call

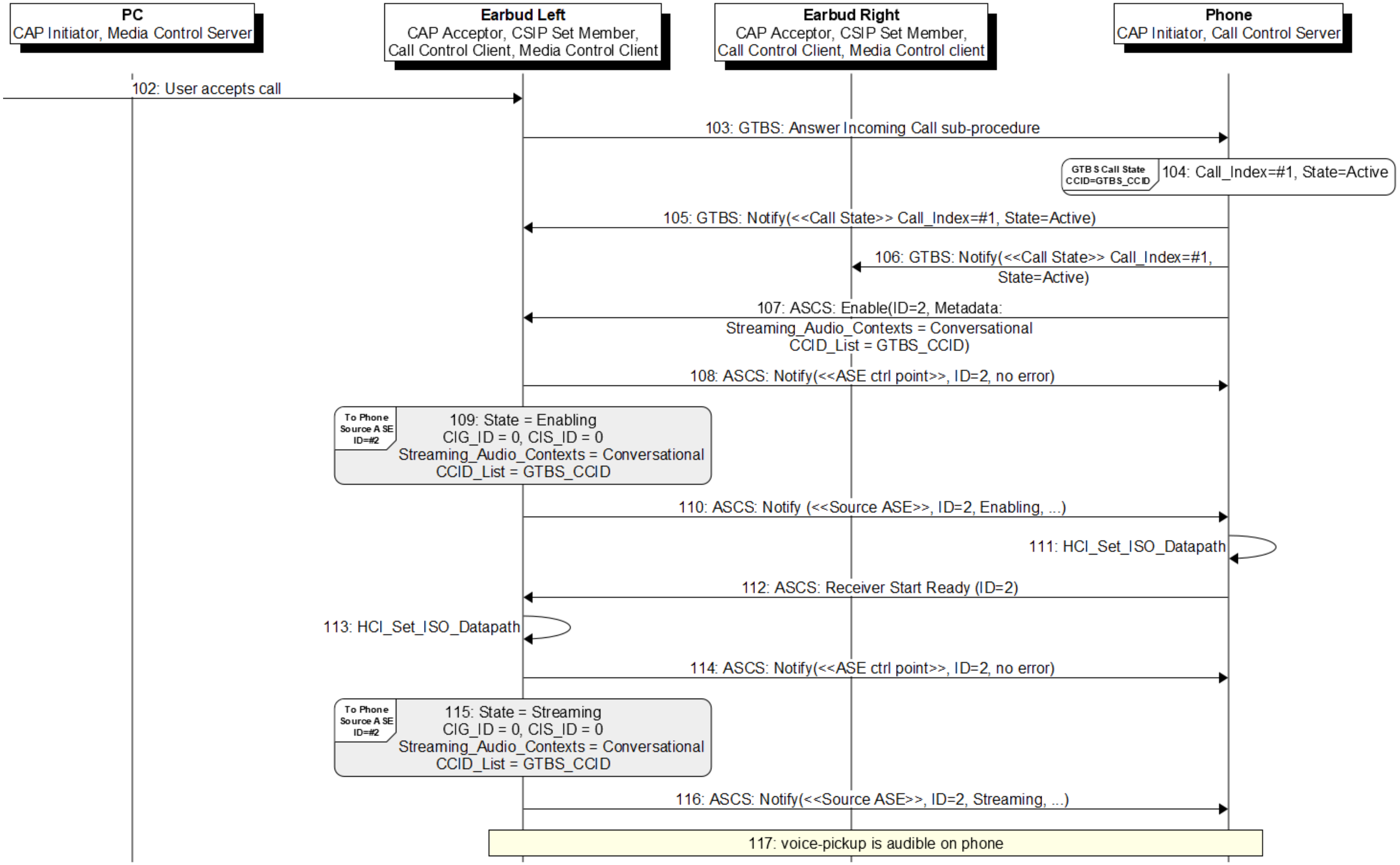
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
102: User accepts call
103: GTBS: Answer Incoming Call sub-procedure
GTBS Call State
CCID=GTBS_CCID
104: Call_Index=#1, State=Active
105: GTBS: Notify(<<Call State>> Call_Index=#1, State=Active)
106: GTBS: Notify(<<Call State>> Call_Index=#1,
State=Active)
107: ASCS: Enable(ID=2, Metadata:
Streaming_Audio_Contexts = Conversational
CCID_List = GTBS_CCID)
108: ASCS: Notify(<<ASE ctrl point>>, ID=2, no error)
To Phone
Source ASE
ID=#2
109: State = Enabling
CIG_ID = 0, CIS_ID = 0
Streaming_Audio_Contexts = Conversational
CCID_List = GTBS_CCID
110: ASCS: Notify (<<Source ASE>>, ID=2, Enabling, ...)
111: HCI_Set_ISO_Datapath
112: ASCS: Receiver Start Ready (ID=2)
113: HCI_Set_ISO_Datapath
114: ASCS: Notify(<<ASE ctrl point>>, ID=2, no error)
To Phone
Source ASE
ID=#2
115: State = Streaming
CIG_ID = 0, CIS_ID = 0
Streaming_Audio_Contexts = Conversational
CCID_List = GTBS_CCID
116: ASCS: Notify(<<Source ASE>>, ID=2, Streaming, ...)
117: voice-pickup is audible on phone

Figure 2.8: Incoming call while playing media, part 8 of 12

Figure 2.8 shows the following steps:

- The user accepts the call using the left earbud’s UI. The left earbud signals the acceptance to the phone. The call then transitions into the Active state.
- The phone sets the Source ASE on the left earbud to the Enabling state. The left earbud does not modify its availability but remains available for any supported Context Type value to the phone and for the <<Conversational>> Context Type to the PC or any other device. Therefore, the left earbud would accept <<Conversational>> Audio Streams from any device. This configuration reflects that <<Conversational>> Audio Streams are associated with active calls, which in most cases are the result of some user action. If the user accepted a different call on the PC, the PC call audio would therefore be allowed to replace the <<Conversational>> Audio Streams from the phone.
- The phone signals “Receiver Start Ready” to the left earbud, which causes the earbud to configure the audio path for the return audio and set the earbud’s Source ASE to the Streaming state.

2.9 Start of rendering of the audio received from the caller

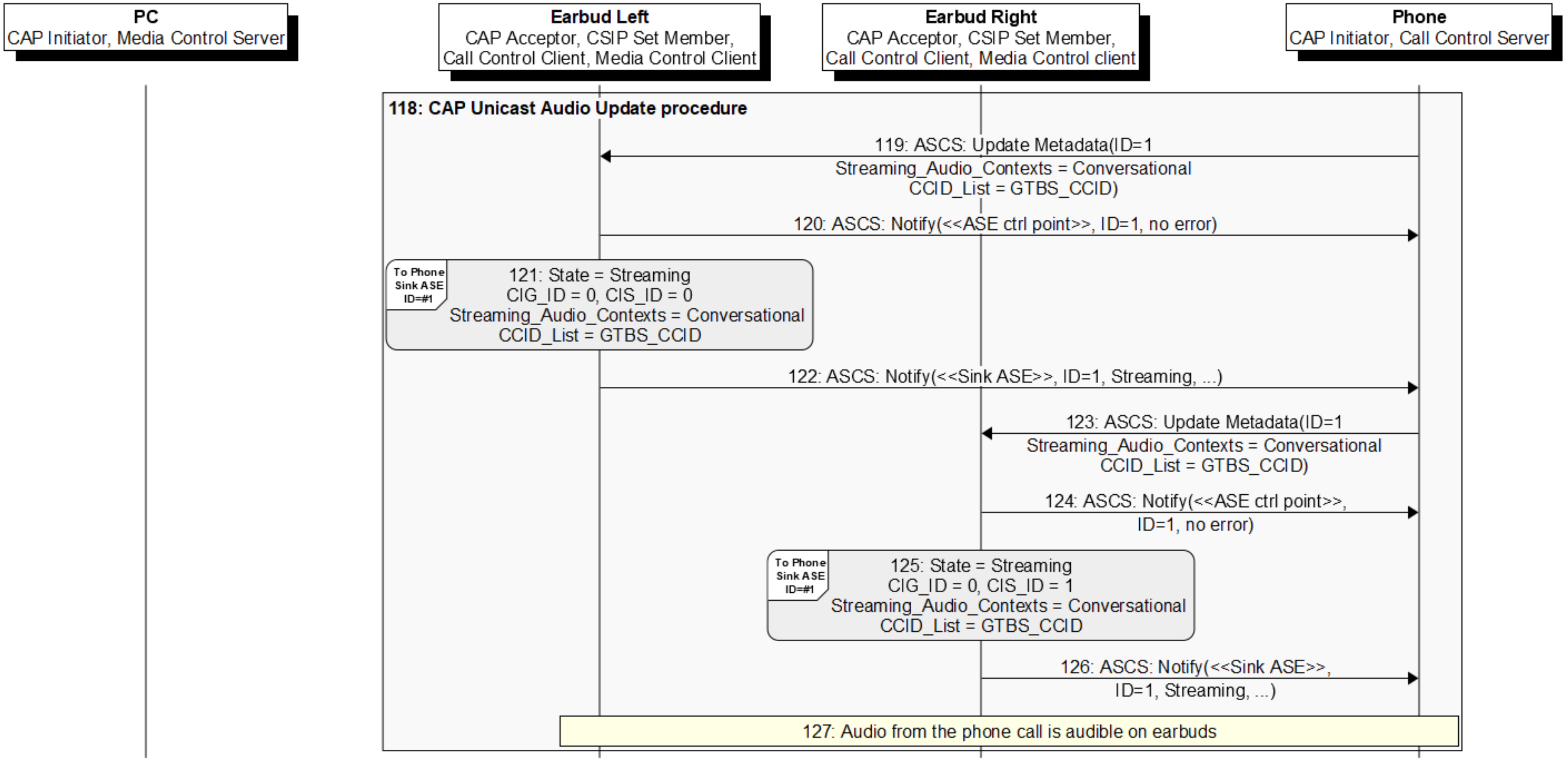
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
118: CAP Unicast Audio Update procedure
119: ASCS: Update Metadata(ID=1
Streaming_Audio_Contexts = Conversational
CCID_List = GTBS_CCID)
120: ASCS: Notify(<<ASE ctrl point>>, ID=1, no error)
To Phone Sink ASE ID=#1
121: State = Streaming
CIG_ID = 0, CIS_ID = 0
Streaming_Audio_Contexts = Conversational
CCID_List = GTBS_CCID
122: ASCS: Notify(<<Sink ASE>>, ID=1, Streaming, ...)
123: ASCS: Update Metadata(ID=1
Streaming_Audio_Contexts = Conversational
CCID_List = GTBS_CCID)
124: ASCS: Notify(<<ASE ctrl point>>,
ID=1, no error)
To Phone Sink ASE ID=#1
125: State = Streaming
CIG_ID = 0, CIS_ID = 1
Streaming_Audio_Contexts = Conversational
CCID_List = GTBS_CCID
126: ASCS: Notify(<<Sink ASE>>,
ID=1, Streaming, ...)
127: Audio from the phone call is audible on earbuds

Figure 2.9: Incoming call while playing media, part 9 of 12

Figure 2.9 shows the following step:

- The phone requests the earbuds to update the Streaming_Audio_Contexts Metadata values on their Sink ASEs from <<Ringtone>> to <<Conversational>>, reflecting the changed purpose of the Audio Streams. Again, the earbuds do not update their availability.

2.10 Remote caller terminates call, ASEs are released

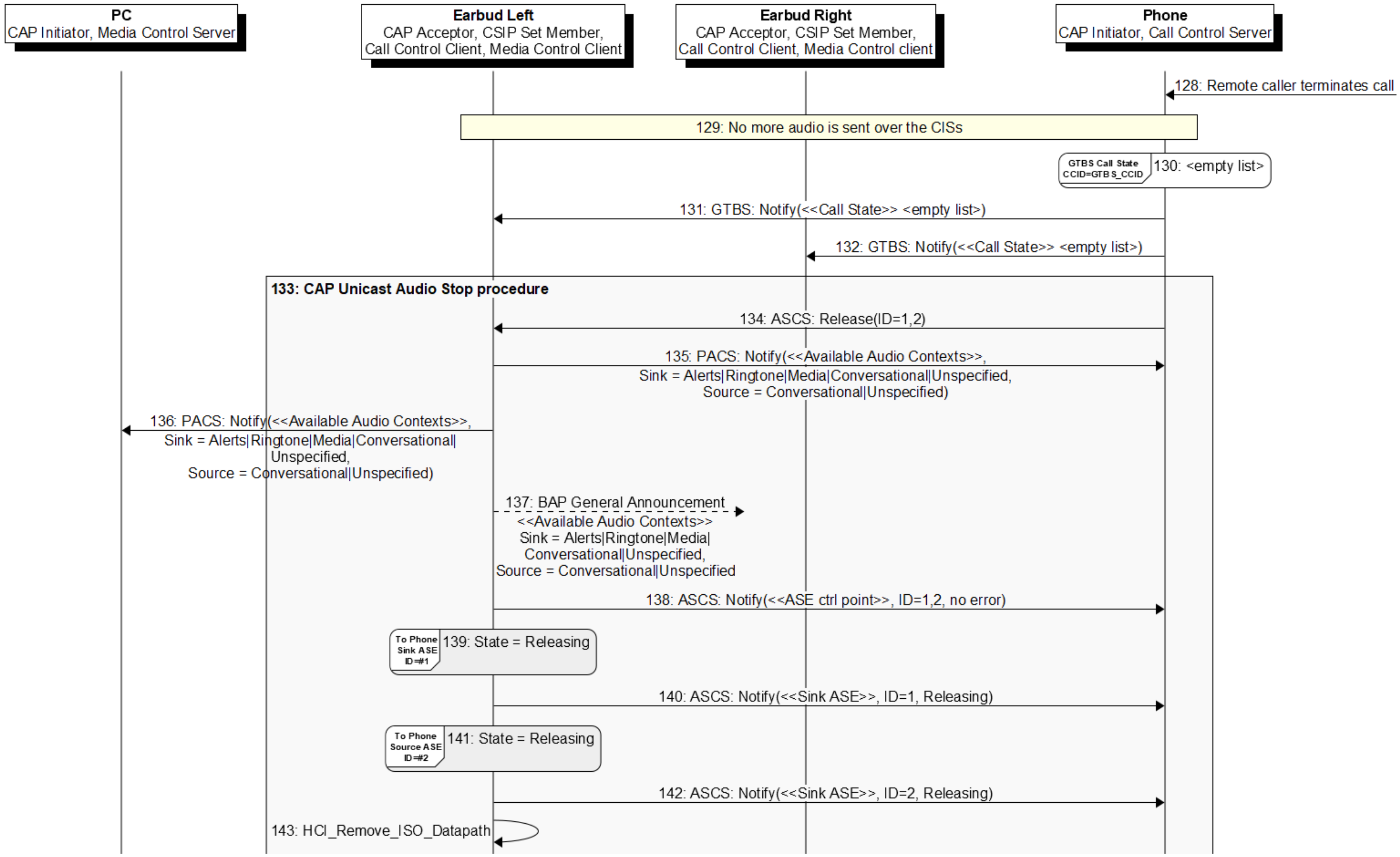
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member,
Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
128: Remote caller terminates call
129: No more audio is sent over the CISs
GTBS Call State
CCID=GTBS_CCID
130: <empty list>
131: GTBS: Notify(<<Call State>> <empty list>)
132: GTBS: Notify(<<Call State>> <empty list>)
133: CAP Unicast Audio Stop procedure
134: ASCS: Release(ID=1,2)
135: PACS: Notify(<<Available Audio Contexts>>,
Sink = Alerts|Ringtone|Media|Conversational|Unspecified,
Source = Conversational|Unspecified)
136: PACS: Notify(<<Available Audio Contexts>>,
Sink = Alerts|Ringtone|Media|Conversational|
Unspecified,
Source = Conversational|Unspecified)
137: BAP General Announcement
<<Available Audio Contexts>>
Sink = Alerts|Ringtone|Media|
Conversational|Unspecified,
Source = Conversational|Unspecified
138: ASCS: Notify(<<ASE ctrl point>>, ID=1,2, no error)
To Phone
Sink ASE
ID=#1
139: State = Releasing
140: ASCS: Notify(<<Sink ASE>>, ID=1, Releasing)
To Phone
Source ASE
ID=#2
141: State = Releasing
142: ASCS: Notify(<<Sink ASE>>, ID=2, Releasing)
143: HCI_Remove_ISO_Datapath

Figure 2.10: Incoming call while playing media, part 10 of 12

Figure 2.10 shows the following steps:

- The remote caller terminates the call. As a result, the Call State characteristic has no more call state entries. The phone notifies this change to both earbuds.
- The phone requests for both earbuds to release their ASEs.

2.11 Earbuds update their availability and terminate the CISs with the phone

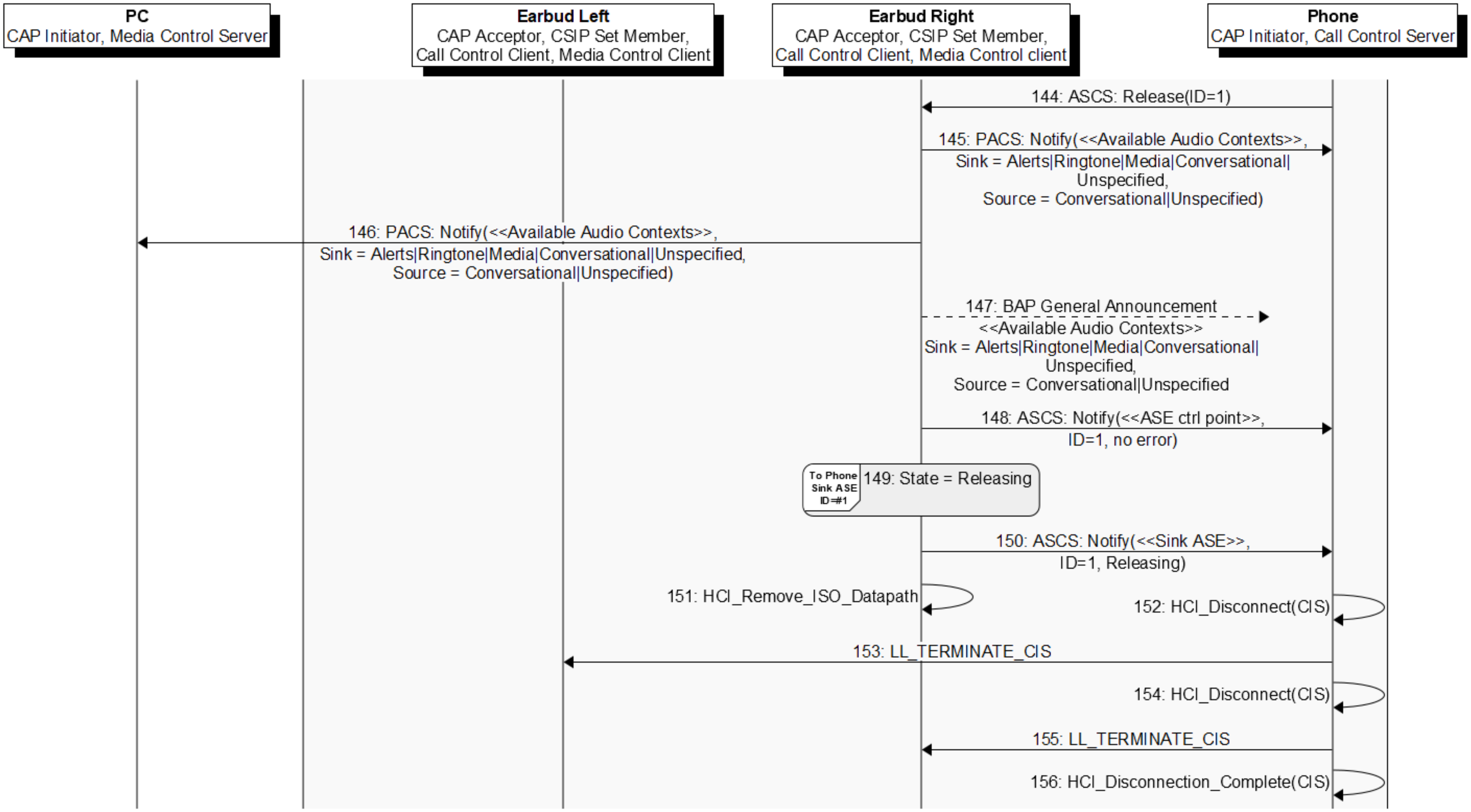
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member, Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member, Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
144: ASCS: Release(ID=1)
145: PACS: Notify(<<Available Audio Contexts>>, Sink = Alerts|Ringtone|Media|Conversational| Unspecified, Source = Conversational|Unspecified)
146: PACS: Notify(<<Available Audio Contexts>>, Sink = Alerts|Ringtone|Media|Conversational|Unspecified, Source = Conversational|Unspecified)
147: BAP General Announcement <<Available Audio Contexts>> Sink = Alerts|Ringtone|Media|Conversational| Unspecified, Source = Conversational|Unspecified
148: ASCS: Notify(<<ASE ctrl point>>, ID=1, no error)
To Phone Sink ASE ID=#1
149: State = Releasing
150: ASCS: Notify(<<Sink ASE>>, ID=1, Releasing)
151: HCI_Remove_ISO_Datapath
152: HCI_Disconnect(CIS)
153: LL_TERMINATE_CIS
154: HCI_Disconnect(CIS)
155: LL_TERMINATE_CIS
156: HCI_Disconnection_Complete(CIS)

Figure 2.11: Incoming call while playing media, part 11 of 12

Figure 2.11 shows the following steps:

- After the ASEs on both earbuds have transitioned to Releasing state, the phone disconnects the CISs with both earbuds.
- After the earbuds have updated their availabilities exposed towards the PC the user has the option to restart the previous audio stream using an interface option exposed by the PC, causing the PC to send a Play opcode using MCS [1]. Note that, unlike classic implementations, CAP states that ”when the Acceptor has initiated the Release operation, then the Initiator should only try to restart the Unicast Start Procedure if requested via a user interaction” (see Section 7.3.1.4 in CAP [4]).

2.12 The earbuds set their ASEs to Codec Configured state and disconnect from the phone

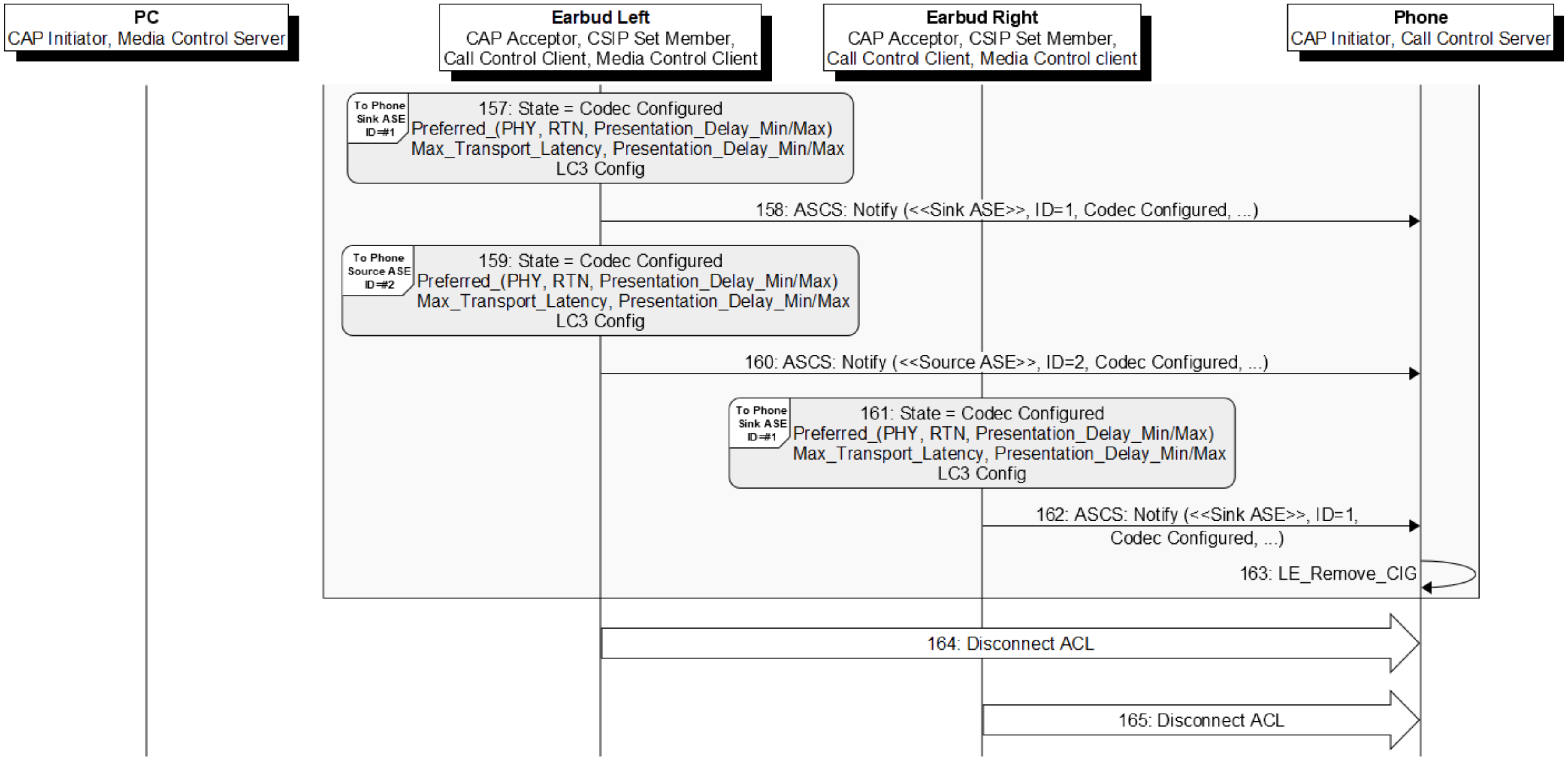
PC
CAP Initiator, Media Control Server
Earbud Left
CAP Acceptor, CSIP Set Member, Call Control Client, Media Control Client
Earbud Right
CAP Acceptor, CSIP Set Member, Call Control Client, Media Control client
Phone
CAP Initiator, Call Control Server
To Phone Sink ASE ID=#1
157: State = Codec Configured Preferred_(PHY, RTN, Presentation_Delay_Min/Max) Max_Transport_Latency, Presentation_Delay_Min/Max LC3 Config
158: ASCS: Notify (<<Sink ASE>>, ID=1, Codec Configured, ...)
To Phone Source ASE ID=#2
159: State = Codec Configured Preferred_(PHY, RTN, Presentation_Delay_Min/Max) Max_Transport_Latency, Presentation_Delay_Min/Max LC3 Config
160: ASCS: Notify (<<Source ASE>>, ID=2, Codec Configured, ...)
To Phone Sink ASE ID=#1
161: State = Codec Configured Preferred_(PHY, RTN, Presentation_Delay_Min/Max) Max_Transport_Latency, Presentation_Delay_Min/Max LC3 Config
162: ASCS: Notify (<<Sink ASE>>, ID=1, Codec Configured, ...)
163: LE_Remove_CIG
164: Disconnect ACL
165: Disconnect ACL

Figure 2.12: Incoming call while playing media, part 12 of 12

Figure 2.12 shows the following steps:

- As a result of the disconnection of the CISs, the ASEs on the earbuds transition to the Codec Configured state, caching the codec configuration that was just used for the call.
    - Alternatively, the earbuds have the option to enter the Idle state.
- The earbuds disconnect the ACL connections with the phone. (This is an implementation-specific choice.)
- Note that after the phone is aware of the terminated call, it has the option to immediately disconnect the ACL connections with the earbuds.
    - In other alternatives, the earbuds have the following additional options:
        - To disconnect the ACL connections with the phone immediately after the earbuds were informed about the terminated call
        - to disable the ASEs, transition to the QoS Configured state, and keep the ACL connection with the phone in order to allow for faster re-establishment of Audio Streams for a future call

2.13 Considerations on setup latency

The number of GATT transactions characterize the setup latency.

- After the earbuds establish a connection with the phone, it takes three GATT transactions on each earbud for the <<Ringtone>> stream to be established (Config Codec, Config QoS, and Enable).
- After the user accepts the call, it takes three GATT transactions on one earbud for the bi-directional <<Conversational>> streams to be established (Enable Source ASE, Receiver Start Ready Source ASE, and Update Metadata Sink ASE).
- Each of the above-mentioned GATT transactions consists of a control point write transaction, followed by a notification about the resulting ASE state change.

Additional interactions, like PACS notification on availability and reading of GTBS call state are performed but, depending on the implementation, may not add additional latency.

3 References

[1] Media Control Service Specification, Version 1.0

[2] Telephone Bearer Service Specification, Version 1.0

[3] Basic Audio Profile Specification, Version 1.0.1

[4] Common Audio Profile Specification, Version 1.0

[5] Published Audio Capabilities Service Specification, Version 1.0.1

[6] Bluetooth Core Specification, Version 5.3 or later

[7] Audio Stream Control Service Specification, Version 1.0

[8] Telephony and Media Audio Profile Specification, Version 1.0