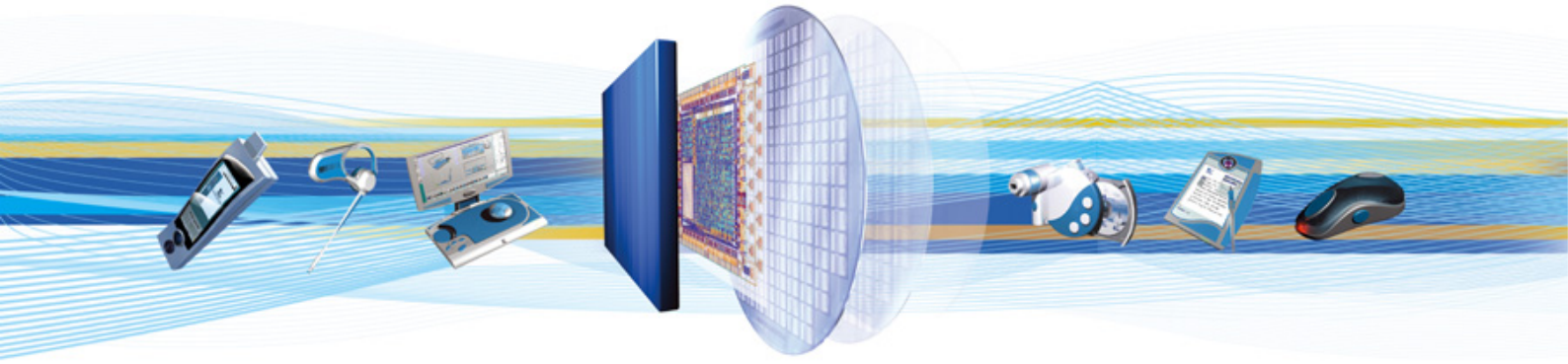
CSR Synergy Bluetooth 18.2.2

AV – Audio Video Profile

API Description

January 2012



Cambridge Silicon Radio Limited

Churchill House
Cambridge Business Park
Cowley Road
Cambridge CB4 0WZ
United Kingdom

Registered in England and Wales 3665875

Tel: +44 (0)1223 692000
Fax: +44 (0)1223 692001
www.csr.com

Contents

List of Figures

List of Tables

1 Introduction

1.1 Introduction and Scope

This document describes the message interface provided by the Audio/Video profile component, henceforward called AV. The AV provides the services specified by the Advanced Audio Distribution Profile [A2DP], the Video Distribution Profile [VDP], the Audio/Video Distribution Transport Protocol [AVDTP] and the Generic Audio/Video Distribution Profile [GAVDP].

1.2 Assumptions

The following assumption is made:

- Only one instance of the AV may be active at any time

2 Description

2.1 Introduction

The AV provides the interface for an application that could conform as source and/or sink, as specified in the Audio/Video Distribution Transport Protocol Specification, see ref. [AVDTP]. The AV supports all mandatory requirements for the Generic Audio/Video, Advanced Audio and Video Distribution profiles [GAVDP, A2DP, VDP]. The basic service as given by [AVDTP] and all signaling procedures are supported.

The AV supplies functionality for:

- Establishment and release of connection
- Configuring streams (discover, get capabilities, get/set configuration)
- Opening and starting streams
- Suspend, resume and re-configuration of streams
- Streaming of data (audio and video)

The AV implementation does not provide any support for encoding/decoding of stream data but serves merely as a transport layer for the stream data. Encoding/decoding must be performed by the application. The SBC codec, mandatory for the Advanced Audio Distribution Profile, A2DP is provided by CSR Synergy Bluetooth as example implementation. The H.263 baseline codec, mandatory for the Video Distribution Profile [VDP], is not provided by CSR Synergy Bluetooth.

Similarly, if content protection is to be used it must be performed at application level. Content protection algorithms are not provided by CSR Synergy Bluetooth.

Note also that AV does not support remote control functions. CSR Synergy Bluetooth supports remote control functions through AVRCP.

The following terms are used throughout the document to indicate the role of the AV profile:

- Initiator (INT): Device that initiates a procedure is considered to be the INT for this procedure
- Acceptor (ACP): The device addressed is considered to be the ACP for this procedure
- Source (SRC): A device is the SRC when it transmits data (audio) in a stream
- Sink (SNK): A device is the SNK when it receives data (audio) from a stream

The role of INT/ACP of a stream is independent of a device's role as SRC/SNK. A device can maintain multiple connections simultaneously, able to assume a role independently of the role with which it participates in the other connections. The device has only been tested with two simultaneous connections. For each connection the profile maintains a state machine, and in the following the states referenced are the states of the specific connection to which the event or procedure applies.

2.2 Profile Fundamentals

This profile is based on Bluetooth v.1.2 specification. Here is a brief summary of the interactions that take place when the INT wishes to send messages to the ACP.

- A link shall be established before a signaling session starts
- There are no fixed master/slave roles
- Use of security features in link level such as authorization, authentication and encryption are optional. Support for authentication and encryption is mandatory, such that the device can take part in the corresponding procedures if requested from a peer device

2.3 Reference Model

The application(s) must interface to the AV profile.

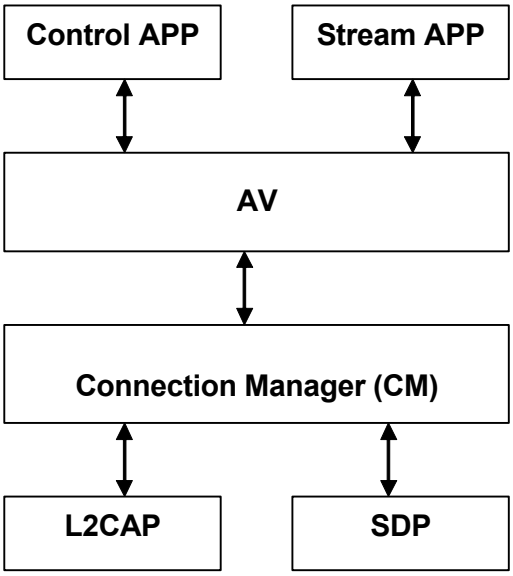


Figure 1: Reference model

As can be seen, the AV profile supports an application split: One for control (such as activation/deactivation, connection setup, discovery etc.), and another for handling the AV stream (stream configuration, establishment/release and the actual stream data). To enable splitting the control application from the stream application the stream application must register itself in the AV, see section

This design allows the AV profile to be integrated in operating systems such as GNU/Linux and Microsoft Windows more easily.

2.4 Sequence Overview

For video streaming a sink (SNK) could be any device capable of rendering or storing a stream and a source (SRC) could be any device capable of capturing or reading a stream. Typically for audio, the SNK will be a headset or speakers and the SRC a mobile phone, computer or an adapter connected to a stereo system. For video a likely application will be the distribution and presentation of low resolution video content captured or presentable by cellular phones.

It is possible for both SRC and SNK to be the initiator (INT) of a connection to another device, which is then the acceptor (ACP) device. The initiator (INT) will pair, and hence create a bond between the two devices. However, since the AV profile does not require authentication or encryption, a connection may be established without any pre-pairing of the devices in advance before initiating the connection. If, on the other hand, security is required, the bonding procedure as described in [SC] must be executed; this may be as dedicated bonding or bonding as part of connection establishment.

The AV profile should be seen as the component that establishes and maintains audio and video streaming. This includes flushing data if the throughput is reduced.

It is the responsibility of the application to encode/decode media data, encrypt/decrypt content protected streams, synchronize streams, and control jitter buffers, if such are used.

Figure 2 outlines the basic states and procedures, which the AV will go through for a stream 'life-cycle'.

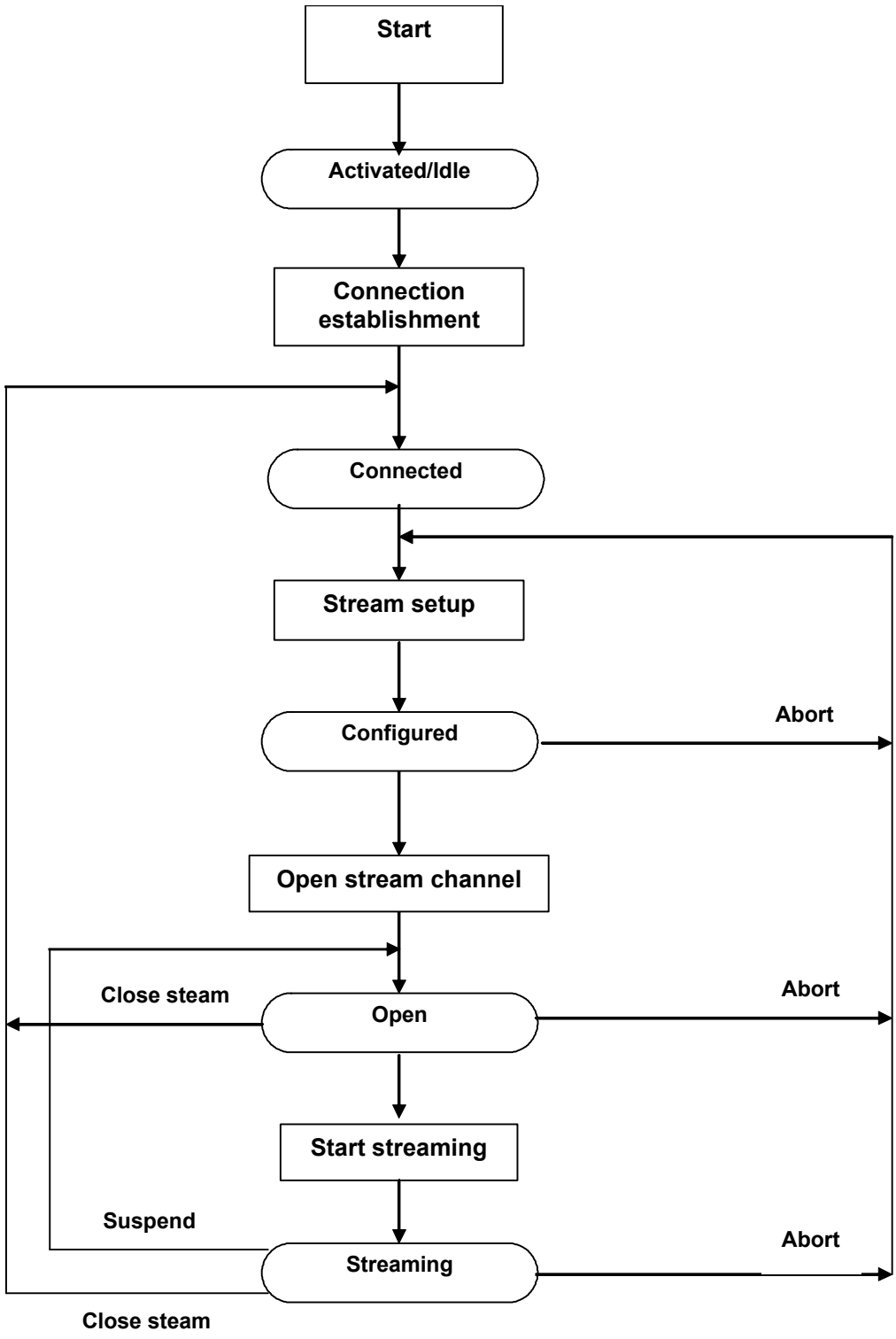


Figure 2: Sequence overview

3 Interface Description

3.1 Stream Application Registration

At any time, an application can register itself at the AV profile as the *stream handler*. All AV primitives related to stream management will be directed to the stream application/handler after the registration. Primitives related to activation/deactivation, connection establishment and release will still be sent to the AV control application. If a separate stream application exists, it will receive CSR_BT_AV_STATUS_IND primitives as a notification of control related changes, i.e. upon connection establishment/release and upon activation/deactivation.

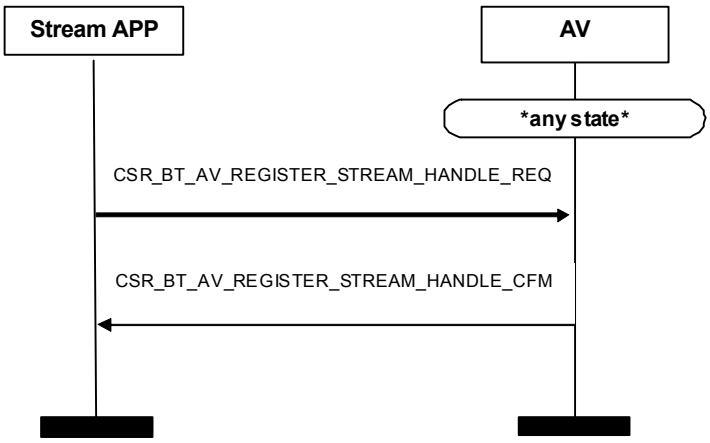


Figure 3: Stream handle registration

If no stream application registers itself in the AV profile, primitives will be sent to the control application. I.e. in the following, 'stream application' should be understood as the 'control application' in the situation that no stream handler application is registered.

Note that the control and stream application can be combined either by registering the control handle as the stream handle, or simply by not registering the stream handler at all.

3.2 Service Activation/Deactivation

3.2.1 Activation

In order to allow other devices to connect to the AV device, the application must activate its AV service. Activation will make the AV device discoverable and connectable for other devices.

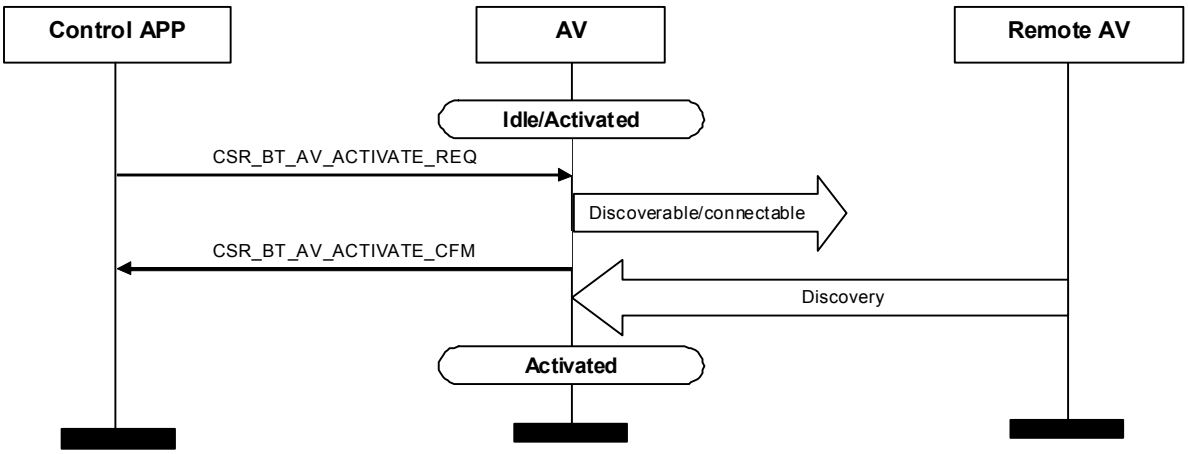


Figure 4: Service activation

If the AV application supports multiple AV roles (audio source, audio sink, video source, video sink) each of the roles should be activated. In addition, if it should be allowed for multiple devices to connect to certain role, this role should be activated multiple times.

At the event of an incoming connection from a remote device, AV will temporarily disable one service activation. If this is the only activation present, the AV device will no longer be discoverable and connectable (unless another non-AV service profile is also activated). When the connection that was initiated by the remote device is released the activation will again be enabled.

3.2.2 Deactivation

When the AV application wants to stop providing an AV service, it must perform a deactivation of the service. The application must send a CSR_BT_AV_DEACTIVATE_REQ specifying the role to deactivate. Upon completion it receives a CSR_BT_AV_DEACTIVATE_CFM. For each activation there must be a deactivation, i.e. if a role has been activated twice, then it must also be deactivated twice in order to completely stop the service. When the last service is deactivated the AV device will no longer be discoverable and connectable (unless another non-AV service profile is also activated).

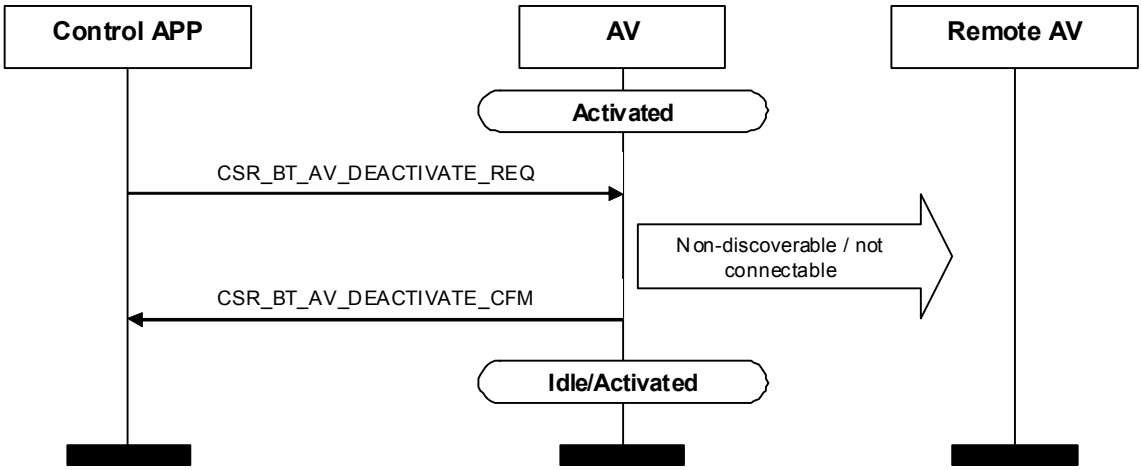


Figure 5: Deactivate service

If the service is in use by an active connection, the deactivation will take effect at the time the connection is released.

3.3 Connection Establishment

3.3.1 Incoming Connection Establishment

An incoming connection establishment from a remote AV device can occur when the AV has been activated. The arrival of an incoming connection is indicated to the application by a CSR_BT_AV_CONNECT_IND primitive.

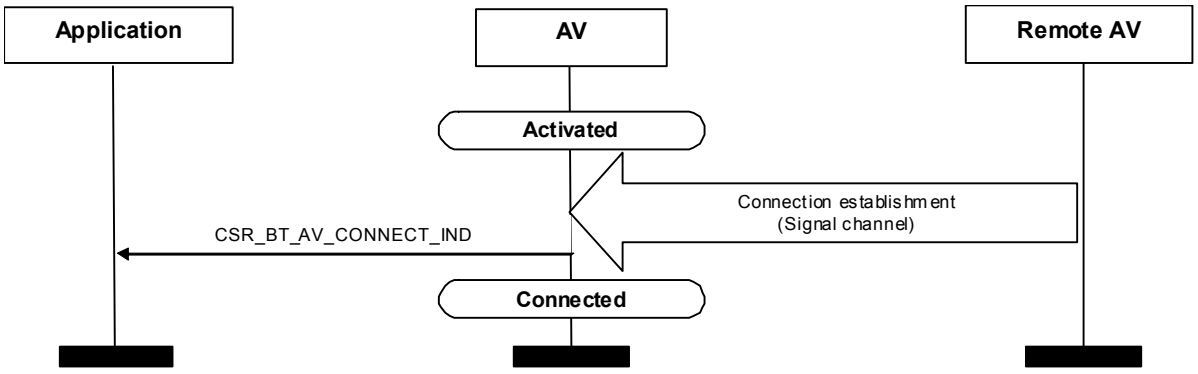


Figure 6: Remote initiated connection establishment

3.3.2 Outgoing Connection Establishment

To initiate an outgoing connection to a remote AV the control application must issue a CSR_BT_AV_CONNECT_REQ to the AV, which will then take the INT role of the connection establishment.

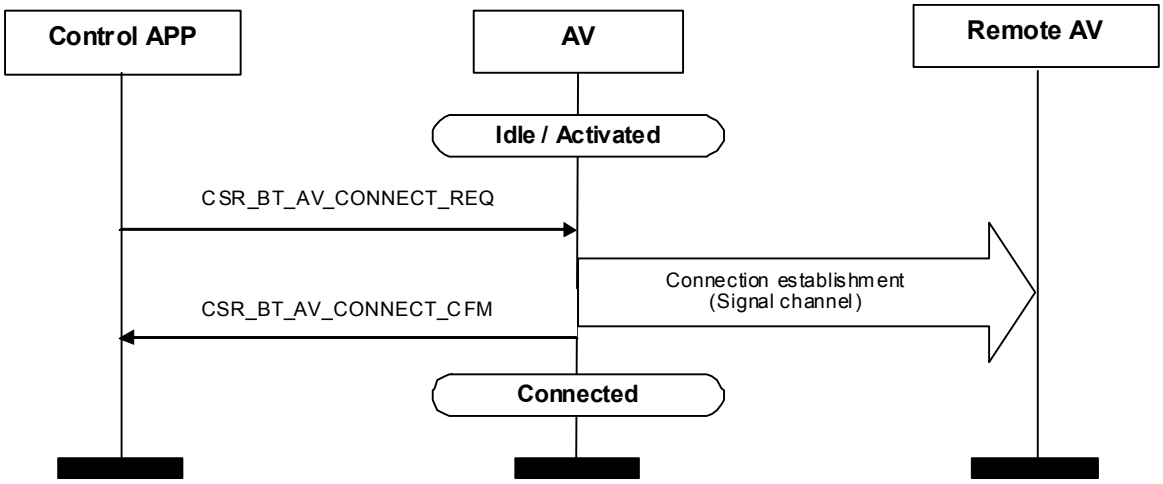


Figure 7: Locally initiated connection establishment

The result of the connection establishment attempt is returned in a CSR_BT_AV_CONNECT_CFM message.

3.3.3 Cancel Connection Establishment

A connection establishment can be cancelled if the application regrets having initiated a connection establishment. A cancellation of a connection establishment may be performed before the AV returns a CSR_BT_AV_CONNECT_CFM primitive with success.

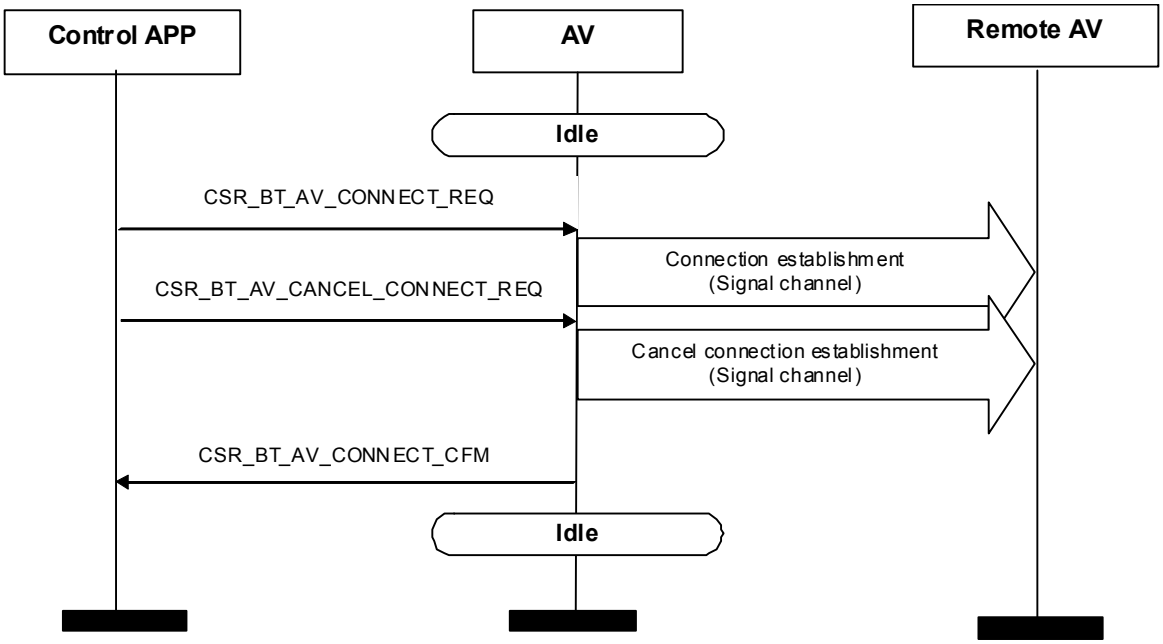


Figure 8: Cancel a locally initiated connection establishment

A successful cancellation of a connection establishment will result in the return of a CSR_BT_AV_CONNECT_CFM stating the result as a cancel connection attempt. In case the connection was established successfully immediately before the connection was cancelled (due to signals crossing) the connection will be disconnected and a CSR_BT_AV_DISCONNECT_IND will be returned to the application.

3.4 Streaming Set Up

3.4.1 Discover Procedure

An AV stream can be regarded as a transportation link with specific content settings joined together by an abstract Stream End Point (SEP) in one AV device to another SEP in the other AV device. A stream end-point has attributes assigned to it that tells about its type (source or sink), its media type (audio or video) and its availability (in use or not). The result of this procedure provides a list of SEPs that the INT can use to figure out what stream possibilities are available from the remote device.

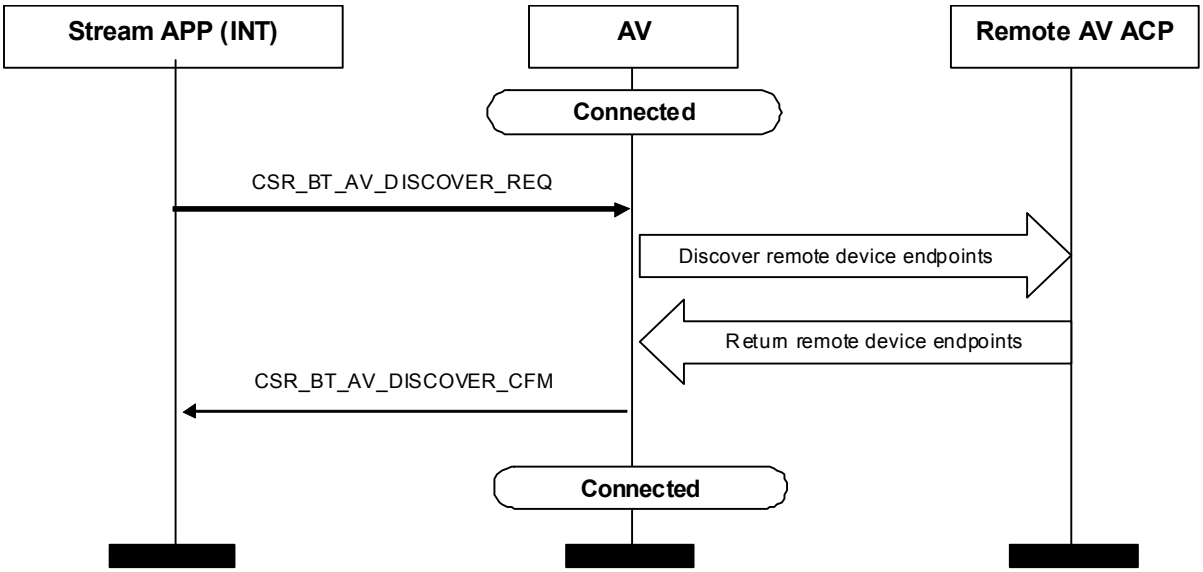


Figure 9: Discover remote device end points

3.4.2 Get Capabilities Procedure

Once a SEP has been discovered the INT can request the capabilities of the remote devices SEP by issuing a CSR_BT_AV_GET_CAPABILITIES_REQ. The ACP will return the capabilities being available such as codec requirement, sampling frequency and bit rate, etc. If the capabilities as described before can be supported by both sides and accepted by the user, if requested, a streaming connection (second L2CAP connection) can be established.

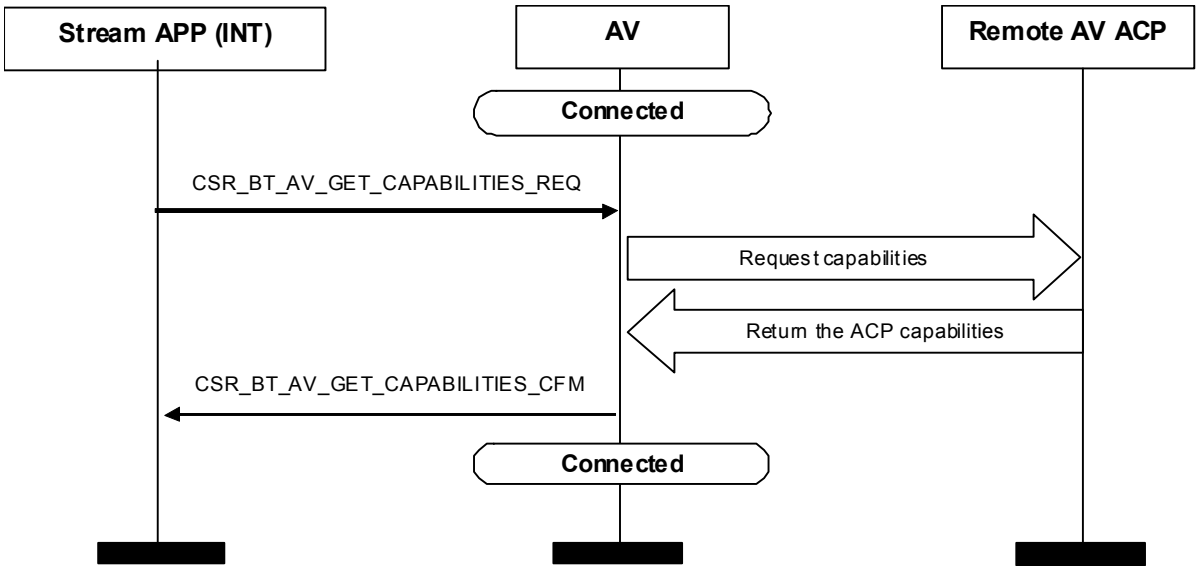


Figure 10: Capabilities request sequence

3.4.3 Set Configuration Procedure

Based on the collected end-point discovery information and service capabilities, the INT determines the most suitable audio streaming parameters (codec, content protection and transport service) for the SNK and the SRC.

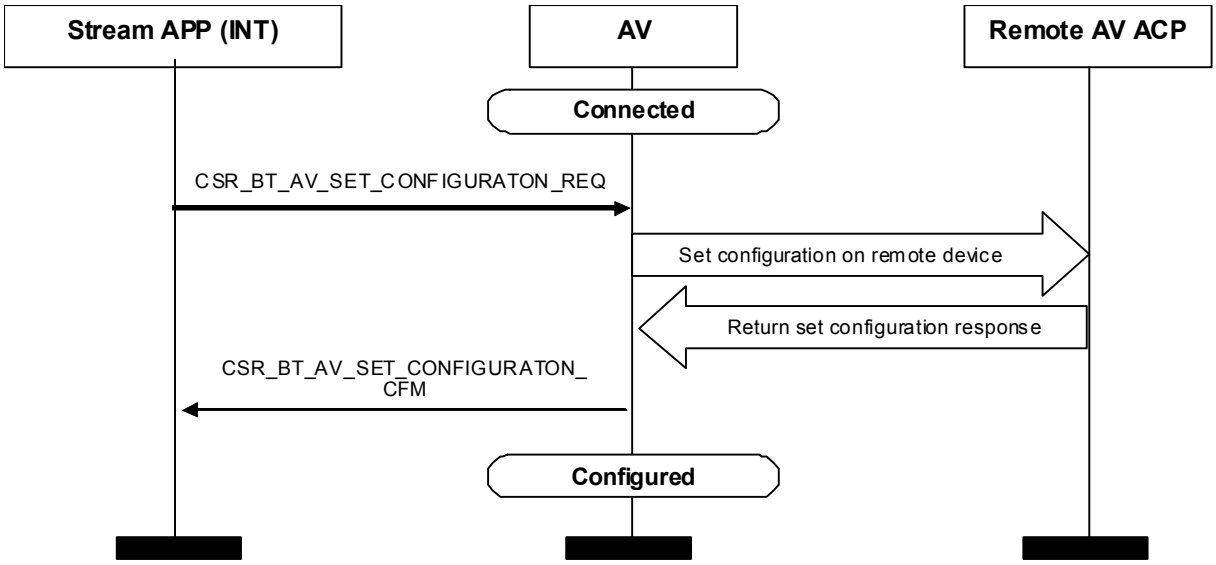


Figure 11: Capabilities request sequence

3.4.4 Open Procedure

The opening of a streaming audio connection will open a new L2CAP channel, which is then mapped to the transport session and used for the media streaming transport.

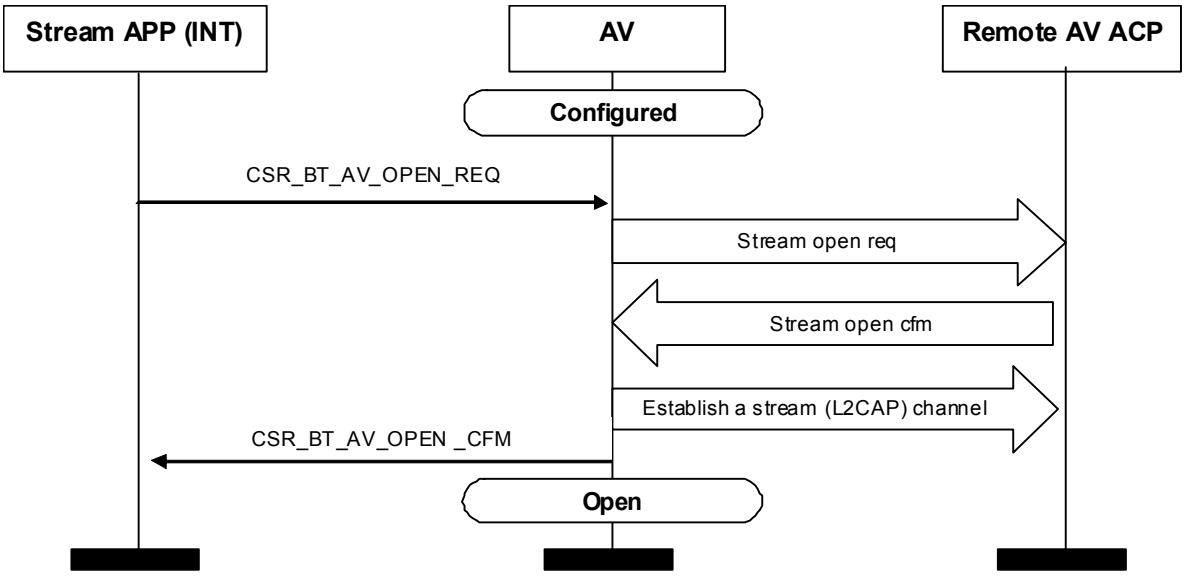


Figure 12: Streaming connection establishment sequence

The stream end point (SNK device) shall have been previously configured for streaming between the peer devices.

Only the INT of a stream configuration procedure may be the INT of the open and start procedures that normally follow to perform streaming over the configured stream end-points. After the start procedure has completed, both devices may trigger any of the stream suspend, stream reconfigure or stream close procedures.

3.4.5 Start Procedure

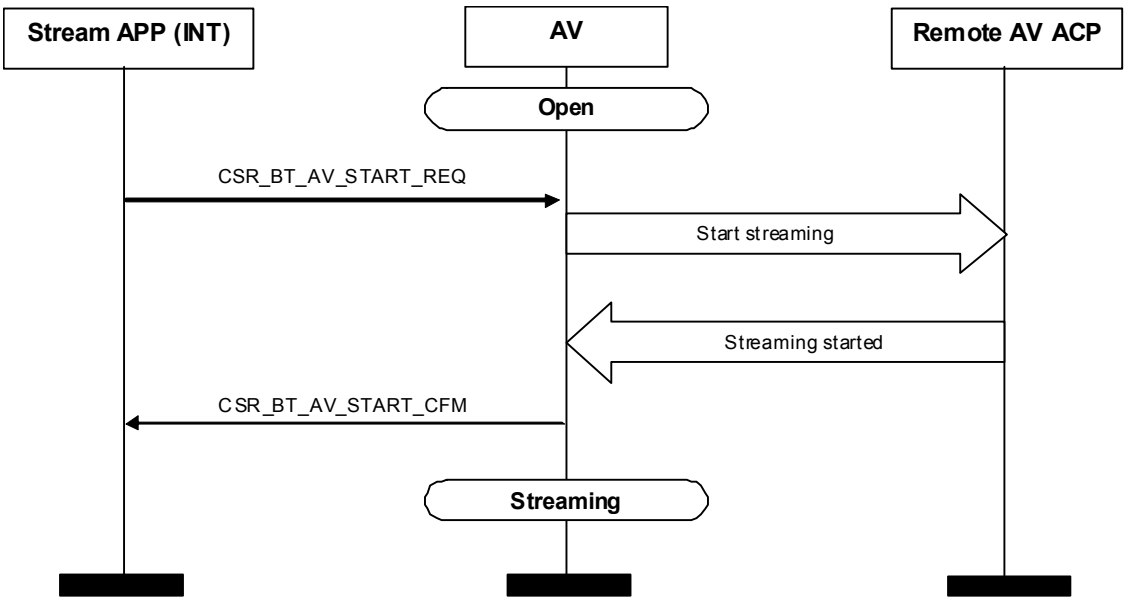


Figure 13: Start streaming sequence

After a stream establishment is completed the start streaming procedure causes the streaming to start.

3.5 Streaming Control

The device that initiates the stream release becomes the INT of the procedure. When the ACP receives the indication that a stream end-point is to be released the ACP releases all the resources allocated to the stream end-point. This includes buffer resources, etc. After the INT receives a confirmation that the stream end-point has

been released, the INT releases all the transport channels and resources allocated to the stream end-point referred to.

3.5.1 Close Procedure

Streaming release can be initiated in Open or Streaming state. Either the SRC or the SNK device can initiate the CSR_BT_AV_CLOSE_REQ.

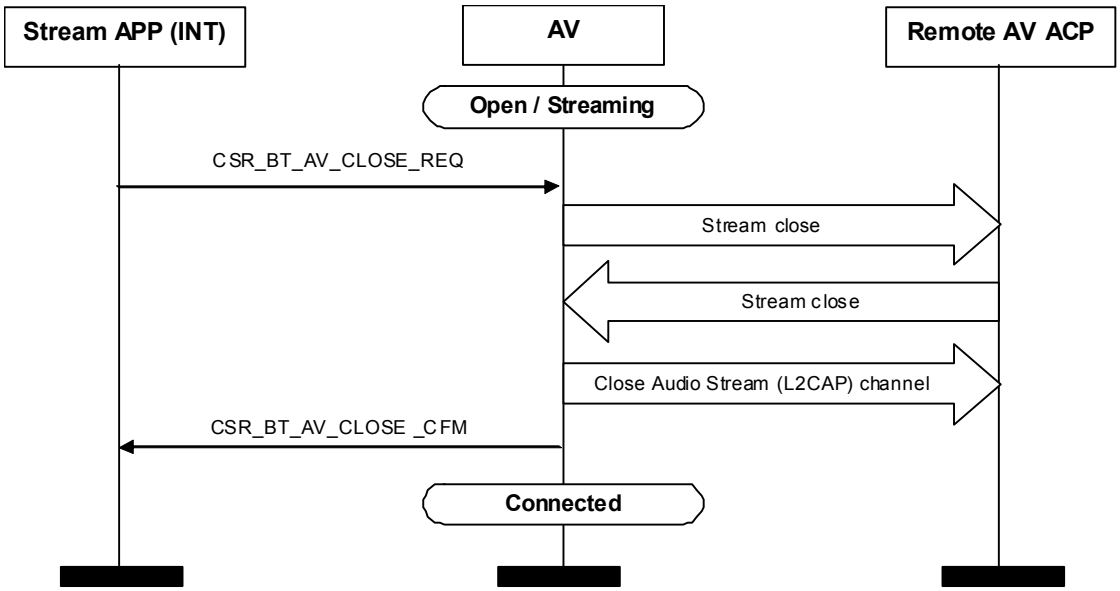


Figure 14: Close audio streaming sequence

3.5.2 Suspend Procedure

This figure shows the stream suspend procedure. The device that initiates the procedure becomes the INT. Suspend can be initialised by either the SRC or the SNK device.

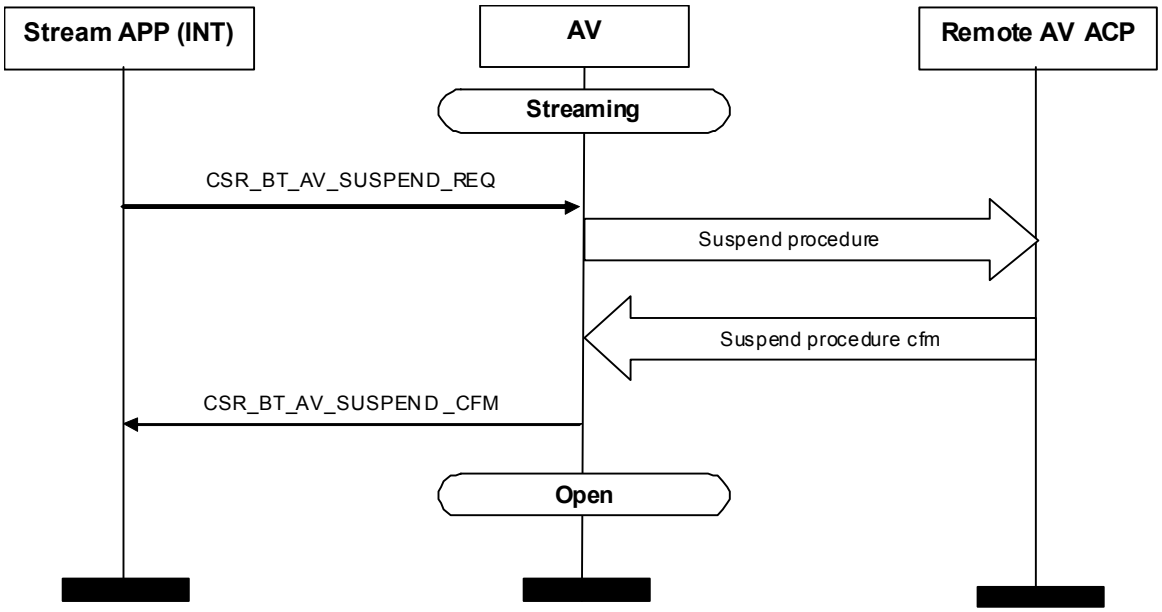


Figure 15: Suspend audio streaming sequence

3.5.3 Reconfigure Procedure

The following figure shows the procedure to reconfigure a stream. The first step is to perform a 'suspend streaming' procedure. When the stream is successfully suspended, a reconfigure command is used for changing current capability settings. The CSR_BT_AV_RECONFIGURE_REQ can be initiated from both sides after the stream is suspended.

It is only the device that initiated the suspend command that is allowed to initiate a start streaming command again.

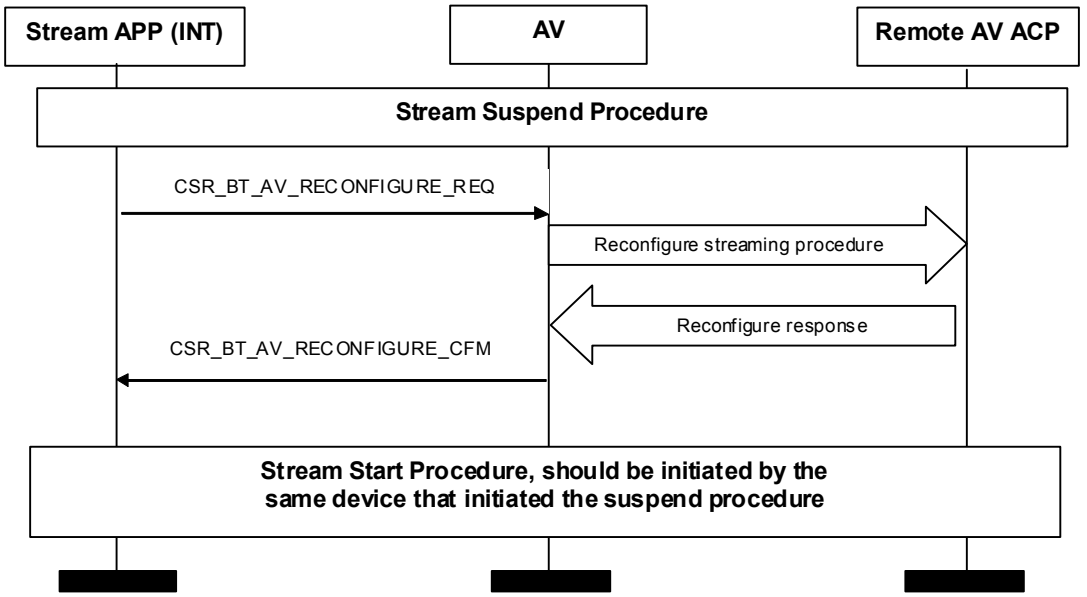


Figure 16: On-handled command from headset

3.5.4 Abort Procedure

This figure shows the stream abort procedure. The abort procedure can be initialised in streaming state and in open state. Either the SRC or the SNK device can initialise the abort procedure.

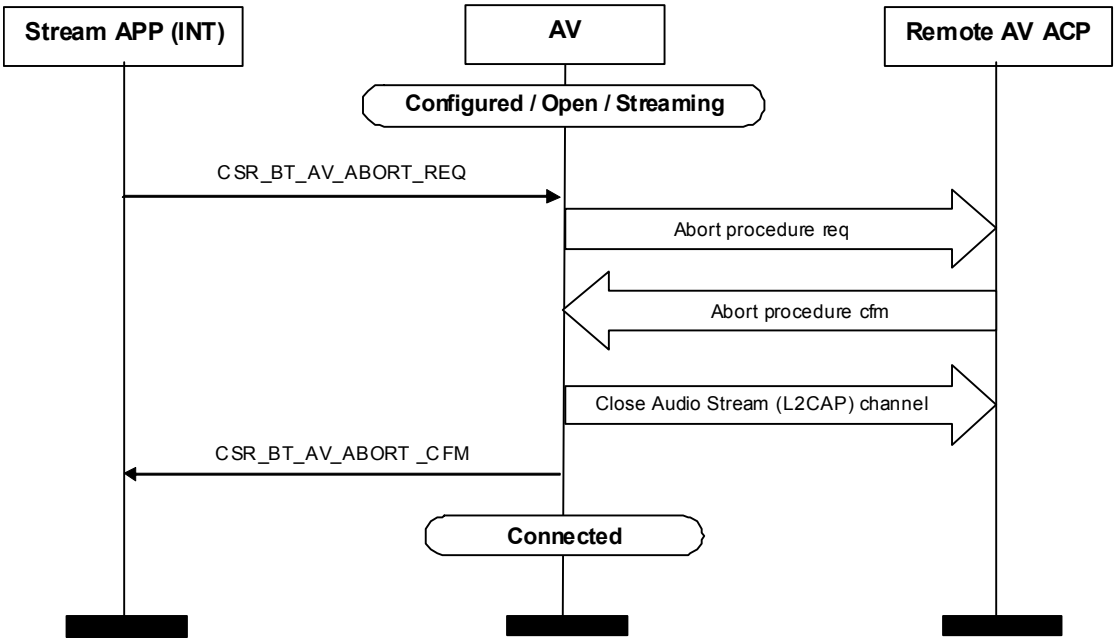


Figure 17: Abort procedure

3.6 Abnormal/remote Disconnect

If an abnormal connection release occurs, the application(s) must release all related data. The control application will receive a CSR_BT_AV_DISCONNECT_IND.

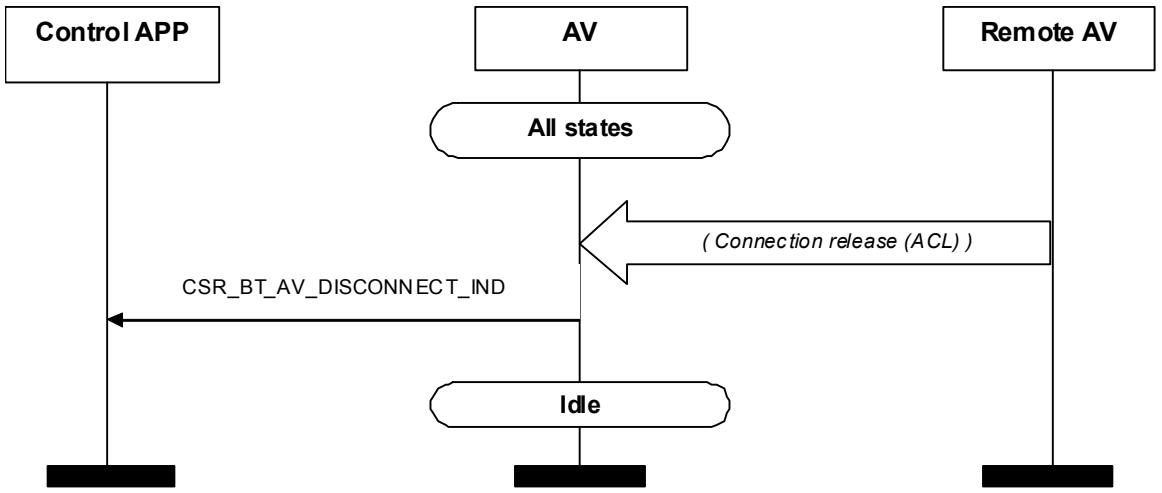


Figure 18: Abnormal connection release

3.7 Disconnect

An AV signal link is released by sending the CSR_BT_AV_DISCONNECT_REQ to the AV profile. When the AV signal link is disconnected a CSR_BT_AV_DISCONNECT_IND is sent to the application.

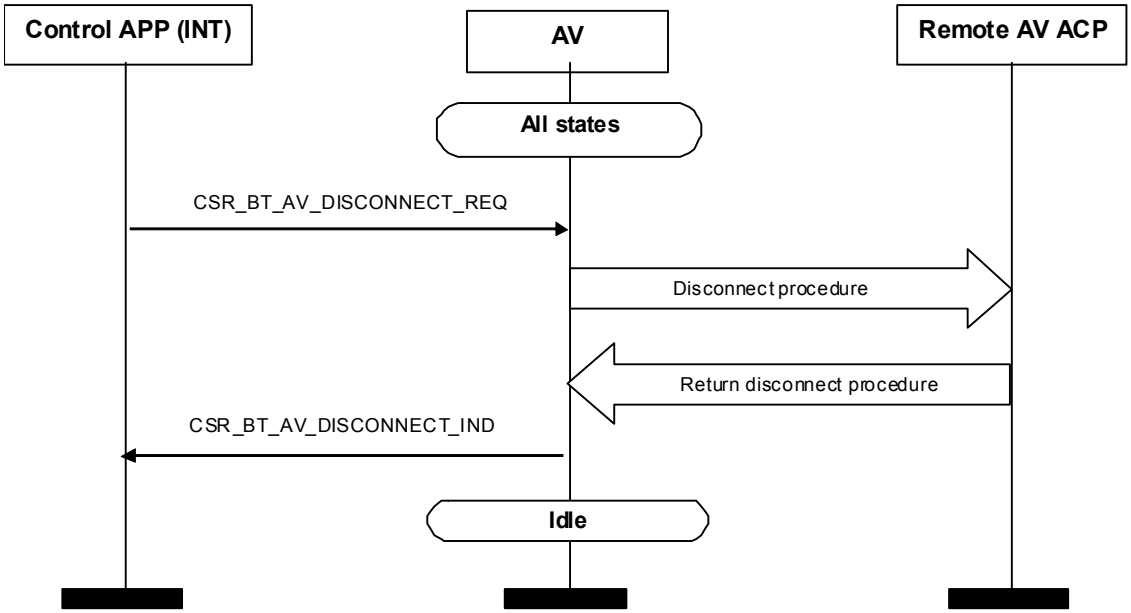


Figure 19: Disconnect request

3.8 Streaming Data

After the streaming start procedure has been started, stream data can be sent to the AV profile. If the AV cannot immediately transmit the stream data to the remote AV SINK device, the data will be buffered internal the AV for later transmission. The internal AV buffer will be flushed if the stream is closed, suspended or aborted.

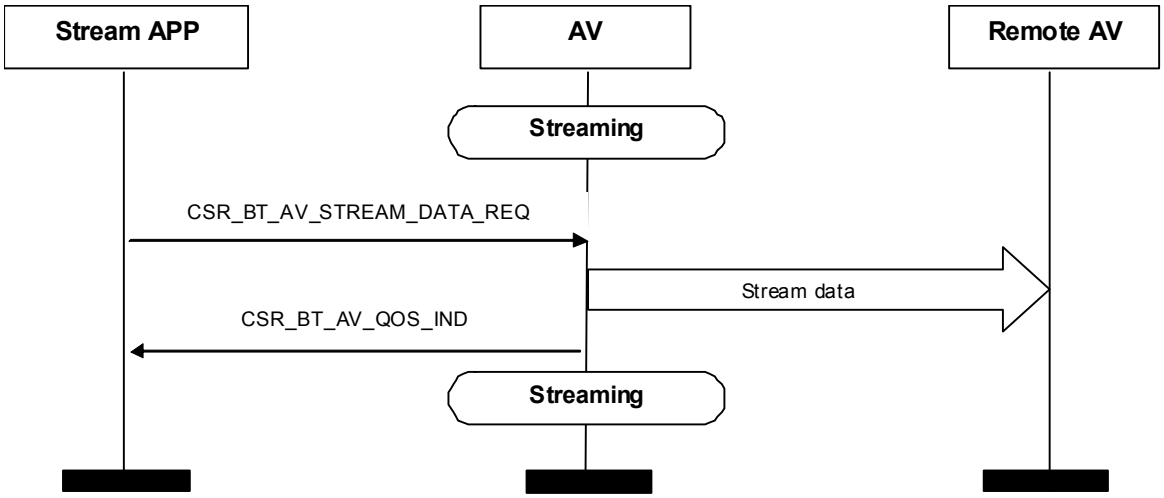


Figure 20: Data from the application to the AV profile

The stream application will receive a CSR_BT_AV_QOS_IND primitive each time a fixed number of CSR_BT_AV_STREAM_DATA_REQ primitives have been submitted to the AV. The CSR_BT_AV_QOS_IND contains information about the relative amount of data in the internal AV buffer, i.e. the amount of data waiting to be transmitted to the remote AV device. The fixed interval with which the CSR_BT_AV_QOS_IND arrives is configurable and this also applies to the size of the internal AV buffer. If the interval is configured to be 0 (zero) then a CSR_BT_AV_QOS_IND will only be received when the buffer runs full and when it subsequently again is empty.

The interval with which the CSR_BT_AV_QOS_IND is sent to the stream application is configurable (CSR_BT_AV_QOS_IND_INTERVAL in csr_bt_usr_config.h). This interval can also be change runtime by using the CSR_BT_AV_SET_QOS_INTERVAL_REQ message.

The CSR_BT_AV_QOS_IND may be used by the stream application to take action when the quality of the AV service changes, e.g. in case of congestion it could increase the encoder compression rate to lower bandwidth usage or temporarily stop the streaming.

4 Audio Video Primitives

This section gives an overview of the primitives and parameters in the interface. Detailed information can be found in the corresponding csr_bt_av_prim.h file.

4.1 List of All Primitives

Primitives:	Reference:
CSR_BT_AV_ACTIVATE_REQ	See section 4.2
CSR_BT_AV_ACTIVATE_CFM	See section 4.2
CSR_BT_AV_DEACTIVATE_REQ	See section 4.3
CSR_BT_AV_DEACTIVATE_CFM	See section 4.3
CSR_BT_AV_REGISTER_STREAM_HANDLE_REQ	See section 4.4
CSR_BT_AV_REGISTER_STREAM_HANDLE_CFM	See section 4.4
CSR_BT_AV_CONNECT_REQ	See section 4.5
CSR_BT_AV_CONNECT_CFM	See section 4.5
CSR_BT_AV_CONNECT_IND	See section 4.5
CSR_BT_AV_CANCEL_CONNECT_REQ	See section 4.6
CSR_BT_AV_DISCONNECT_REQ	See section 4.7
CSR_BT_AV_DISCONNECT_IND	See section 4.7
CSR_BT_AV_STREAM_DATA_REQ	See section 4.8
CSR_BT_AV_STREAM_DATA_IND	See section 4.8
CSR_BT_AV_QOS_IND	See section 4.9
CSR_BT_AV_STREAM_MTU_SIZE_IND	See section 4.10
CSR_BT_AV_DISCOVER_REQ	See section 4.11
CSR_BT_AV_DISCOVER_IND	See section 4.11
CSR_BT_AV_DISCOVER_RES	See section 4.11
CSR_BT_AV_DISCOVER_CFM	See section 4.11
CSR_BT_AV_GET_CAPABILITIES_REQ	See section 4.12
CSR_BT_AV_GET_CAPABILITIES_IND	See section 4.12
CSR_BT_AV_GET_CAPABILITIES_RES	See section 4.12
CSR_BT_AV_GET_CAPABILITIES_CFM	See section 4.12
CSR_BT_AV_GET_ALL_CAPABILITIES_IND	See section 4.12
CSR_BT_AV_GET_ALL_CAPABILITIES_RES	See section 4.12
CSR_BT_AV_SET_CONFIGURATION_REQ	See section 4.13
CSR_BT_AV_SET_CONFIGURATION_IND	See section 4.13
CSR_BT_AV_SET_CONFIGURATION_RES	See section 4.13
CSR_BT_AV_SET_CONFIGURATION_CFM	See section 4.13
CSR_BT_AV_GET_CONFIGURATION_REQ	See section 4.14
CSR_BT_AV_GET_CONFIGURATION_IND	See section 4.14
CSR_BT_AV_GET_CONFIGURATION_CFM	See section 4.14
CSR_BT_AV_GET_CONFIGURATION_RES	See section 4.14
CSR_BT_AV_RECONFIGURE_REQ	See section 4.15
CSR_BT_AV_RECONFIGURE_IND	See section 4.15
CSR_BT_AV_RECONFIGURE_CFM	See section 4.15
CSR_BT_AV_RECONFIGURE_RES	See section 4.15

Primitives:	Reference:
CSR_BT_AV_OPEN_REQ	See section 4.16
CSR_BT_AV_OPEN_IND	See section 4.16
CSR_BT_AV_OPEN_RES	See section 4.16
CSR_BT_AV_OPEN_CFM	See section 4.16
CSR_BT_AV_START_REQ	See section 4.17
CSR_BT_AV_START_IND	See section 4.17
CSR_BT_AV_START_RES	See section 4.17
CSR_BT_AV_START_CFM	See section 4.17
CSR_BT_AV_CLOSE_REQ	See section 4.18
CSR_BT_AV_CLOSE_IND	See section 4.18
CSR_BT_AV_CLOSE_RES	See section 4.18
CSR_BT_AV_CLOSE_CFM	See section 4.18
CSR_BT_AV_SUSPEND_REQ	See section 4.19
CSR_BT_AV_SUSPEND_IND	See section 4.19
CSR_BT_AV_SUSPEND_CFM	See section 4.19
CSR_BT_AV_SUSPEND_RES	See section 4.19
CSR_BT_AV_ABORT_REQ	See section 4.20
CSR_BT_AV_ABORT_IND	See section 4.20
CSR_BT_AV_ABORT_RES	See section 4.20
CSR_BT_AV_ABORT_CFM	See section 4.20
CSR_BT_AV_STATUS_IND	See section 4.21
CSR_BT_AV_SECURITY_CONTROL_REQ	See section 4.22
CSR_BT_AV_SECURITY_CONTROL_IND	See section 4.22
CSR_BT_AV_SECURITY_CONTROL_RES	See section 4.22
CSR_BT_AV_SECURITY_CONTROL_CFM	See section 4.22
CSR_BT_AV_SECURITY_IN_REQ	See section 4.23
CSR_BT_AV_SECURITY_IN_CFM	See section 4.23
CSR_BT_AV_SECURITY_OUT_REQ	See section 4.23
CSR_BT_AV_SECURITY_OUT_CFM	See section 4.23
CSR_BT_AV_SET_QOS_INTERVAL_REQ	See section 4.24
CSR_BT_AV_LP_NEG_CONFIG_REQ	See section 4.25
CSR_BT_AV_DELAY_REPORT_REQ	See section 4.27
CSR_BT_AV_DELAY_REPORT_IND	See section 4.27
CSR_BT_AV_GET_CHANNEL_INFO_REQ	See section 4.26
CSR_BT_AV_GET_STREAM_CHANNEL_INFO_REQ	See section 4.26
CSR_BT_AV_GET_CHANNEL_INFO_CFM	See section 4.26

Table 1: List of all primitives

4.2 CSR_BT_AV_ACTIVATE

Parameters / Primitives	type	phandle	localRole	avResultCode	avResultSupplier
CSR_BT_AV_ACTIVATE_REQ	✓	✓	✓		
CSR_BT_AV_ACTIVATE_CFM	✓			✓	✓

Table 2: CSR_BT_AV_ACTIVATE Primitives

Description

This signal is used for activating an AV service and making the device discoverable and connectable. The AV service is given by the 4 roles: audio source, audio sink, video source and video sink. If multiple incoming connections must be supported for a given service then multiple activations must be performed for this service.

The device remains activated until deactivated. The confirmation is sent to the task identified by phandle.

Parameters

type	Signal identity CSR_BT_AV_ACTIVATE_REQ/CFM.
phandle	The identity of the calling process. It is possible to initiate the procedure from any higher layer process as the response is returned to the process identified by phandle.
localRole	The service provided by the application (audio or video source/sink).
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h

4.3 CSR_BT_AV_DEACTIVATE

Parameters / Primitives	type	localRole	avResultCode	avResultSupplier
CSR_BT_AV_DEACTIVATE_REQ	✓	✓		
CSR_BT_AV_DEACTIVATE_CFM	✓		✓	✓

Table 3: CSR_BT_AV_DEACTIVATE Primitives

Description

Deactivate a service, which has previously been activated. Only the service specified will be deactivated but if a service has been activate multiple times the it also must be deactivated the same amount of times to completely deactivate the service. When the last service is deactivated the AV device will no longer be discoverable and connectable. The confirmation is sent to the task identified by phandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_DEACTIVATE_REQ/CFM.
localRole	The service to deactivate (audio or video source/sink)
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h

4.4 CSR_BT_AV_REGISTER_STREAM_HANDLE

Primitives \ Parameters	type	streamHandle
CSR_BT_AV_REGISTER_STREAM_HANDLE_REQ	✓	✓
CSR_BT_AV_REGISTER_STREAM_HANDLE_CFM	✓	

Table 4: CSR_BT_AV_REGISTER_STREAM_HANDLE Primitives

Description

Register the application which is to receive AV data messages. This message can be sent at any time, and the new will take effect immediately.

Parameters

type	Signal identity CSR_BT_AV_REGISTER_STREAM_HANDLE_REQ/CFM.
streamHandle	Handle of the task which is to receive all future AV data.

4.5 CSR_BT_AV_CONNECT

Primitives \ Parameters	type	phandle	deviceAddr	remoteRole	localRole	avResultCode	avResultSupplier	connectionId	btConnId
CSR_BT_AV_CONNECT_REQ	✓	✓	✓	✓	✓				
CSR_BT_AV_CONNECT_CFM	✓		✓			✓	✓	✓	✓
CSR_BT_AV_CONNECT_IND	✓		✓					✓	✓

Table 5: CSR_BT_AV_CONNECT Primitives

Description

A CSR_BT_AV_CONNECT_REQ initiates the establishment of a connection towards another device specified by its Bluetooth address. The result is returned to the application in CSR_BT_AV_CONNECT_CFM. If the device has been activated, the application is notified of the arrival of a connection from remote device with CSR_BT_AV_CONNECT_IND.

The confirmation and indication is sent to task identified by phandle provided in the activation procedure. If two different handles were provided a status indication is sent to the task specified by streamAppHandle when a connect indication is sent to the task specified by phandle. The same applies for the confirmation as long as the result specified in the confirmation reports success.

Parameters

type	Signal identity CSR_BT_AV_CONNECT_REQ/CFM/IND.
phandle	The identity of the calling process. It is possible to initiate the procedure from any higher layer process as the response is returned to the process identified by phandle.
deviceAddr	The Bluetooth device address to which a connection must be established
remoteRole	The wanted service/role from the remote device.
localRole	The service/role of the device itself.
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h
connectionId	Identity the established connection (if successful).
btConnId	Identifier used when moving the connection to another AMP controller, i.e. when calling the `CsrBtAmpmMoveReqSend`-function.

4.6 CSR_BT_AV_CANCEL_CONNECT

Primitives \ Parameters	type	deviceAddr
CSR_BT_AV_CANCEL_CONNECT_REQ	✓	✓

Table 6: CSR_BT_AV_CANCEL_CONNECT Primitives

Description

A CSR_BT_AV_CANCEL_CONNECT_REQ cancels a prior request for connection establishment. If the cancel request is successful a CSR_BT_AV_CONNECT_CFM is return with the result set to 'CSR_BT_AV_CANCEL_CONNECT_ATTEMPT'. If the connection setup was completed before the AV received the CSR_BT_AV_CANCEL_CONNECT_REQ the connection will be disconnected and a CSR_BT_AV_DISCONNECT_IND returned to the application.

Parameters

type	Signal identity CSR_BT_AV_CANCEL_CONNECT_REQ.
deviceAddr	The Bluetooth device address of the device to which the prior connection request was initiated.

4.7 CSR_BT_AV_DISCONNECT

Parameters / Primitives	type	connectionId	localTerminated	reasonCode	reasonSupplier
CSR_BT_AV_DISCONNECT_REQ	✓	✓			
CSR_BT_AV_ DISCONNECT_IND	✓	✓	✓	✓	✓

Table 7: CSR_BT_AV_DISCONNECT Primitives

Description

A CSR_BT_AV_DISCONNECT_REQ initiates a release of a previously established connection. When the connection has been released, or if the connection is released by the remote device, this is indicated to the application by a CSR_BT_AV_DISCONNECT_IND. If the phandle and streamAppHandle provided in the activation procedure differ, a status indication is sent to the task identified by the streamAppHandle.

Parameters

type	Signal identity CSR_BT_AV_DISCONNECT_REQ/IND.
connectionId	Identity of the connection to be released
localTerminated	TRUE if termination of connection happened on request from the local host; FALSE otherwise.
reasonCode	The reason code of the operation. Possible values depend on the value of reasonSupplier. If e.g. the reasonSupplier == CSR_BT_SUPPLIER_CM then the possible reason codes can be found in csr_bt_cm_prim.h. All values which are currently not specified in the respective prim.h files are regarded as reserved and the application should consider them as errors.
reasonSupplier	This parameter specifies the supplier of the reason given in reasonCode. Possible values can be found in csr_bt_result.h

4.8 CSR_BT_AV_STREAM_DATA

Parameters / Primitives	type	shandle	hdr_type	pad & pad2	length	*data
CSR_BT_AV_STREAM_DATA_REQ	✓	✓	✓		✓	✓
CSR_BT_AV_STREAM_DATA_IND	✓	✓		✓	✓	✓

Table 8: CSR_BT_AV_STREAM_DATA Primitives

Description

The source application uses the CSR_BT_AV_STREAM_DATA_REQ to send media stream data. The sink receives media stream data in a CSR_BT_AV_STREAM_DATA_IND.

If streaming data for a mandatory (e.g. SBC) or optional (e.g. MP3 or AAC) codec, the application must reserve space for the media packet header (CSR_BT_AV_FIXED_MEDIA_PACKET_HDR_SIZE bytes) in front of the media data. The header should also be filled out by the application or by a call to the AvStreamDataReqSend function. See csr_bt_av_lib.c/csr_bt_av_lib.h. However, the sequence number is set into the media packet header by the AV profile.

The indication is sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_STREAM_DATA_REQ/IND.
shandle	Stream handle, identifies the stream to which the data belongs.
hdr_type	Media packet header type: CSR_BT_AV_MEDIA_PACKET_HDR_TYPE_RTP (applies for all mandatory and optional codecs) or CSR_BT_AV_MEDIA_PACKET_HDR_TYPE_NONE to send raw application media packets.
pad & pad2	Message structure padding – will be removed in future release, no valid content.
length	Length of data.
*data	Pointer to a block of media packet data.

Two library functions for sending stream data exist. AvStreamDataReqSend is used for sending media packets with RTP header (applies to all mandatory and optional codecs) and AvStreamRawDataReqSend for sending raw application specific media packets (e.g. for vendor specific codecs).

The AvStreamDataReqSend library function takes the additional four parameters related to the filling of the RTP media header fields:

timestamp	Timestamp of the first audio frame in data. Also, refer to [A2DP].
padding	TRUE if the data is padded, otherwise FALSE. In most cases use FALSE.
marker	Marker bit. Set to FALSE.
payloadType	Describes the payload type. For SBC use any value in the dynamic range ([96;127]), for MPEG-1,2 (MP3) use 14.

4.9 CSR_BT_AV_QOS

Parameters / Primitives	type	shandle	bufferStatus
CSR_BT_AV_QOS_IND	✓	✓	✓

Table 9: CSR_BT_AV_QOS Primitives

Description

AV will regularly send a CSR_BT_AV_QOS_IND to the stream application giving and indication of the amount of data buffered on the streams transmit queue. The application may choose to adjust the bit rate of the stream based on these indications in order to match the available connection bandwidth.

The interval with which the CSR_BT_AV_QOS_IND is sent to the stream application is configurable (CSR_BT_AV_QOS_IND_INTERVAL in csr_bt_usr_config.h), or can be set runtime by using the CSR_BT_AV_SET_QOS_INTERVAL_REQ message, see section 4.24.

If the interval is set to zero (0), then only when the buffer is full and when the buffer is subsequently emptied a CSR_BT_AV_QOS_IND will be sent. Note that if the stream is closed or suspended the AV buffer will automatically be flushed and in this case a CSR_BT_AV_QOS_IND indicating 'empty' is not sent to the stream application.

Parameters

type	Signal identity CSR_BT_AV_QOS_IND.
shandle	Stream handle.
bufferStatus	Current status of the stream buffer. Indicates the amount of buffered media packets in the transmit queue. A value of zero (0) means that the queue is empty and the value 10 means that it is full.

4.10 CSR_BT_AV_STREAM_MTU_SIZE

Primitives \ Parameters	type	shandle	remoteMtuSize	btConnId
CSR_BT_AV_STREAM_MTU_SIZE_IND	✓	✓	✓	✓

Table 10: CSR_BT_AV_STREAM_MTU_SIZE Primitives

Description

When the AV opens a stream, i.e. creates a L2CAP channel for the stream data, the maximum allowed transmit frame size is indicated to the application in a CSR_BT_AV_STREAM_MTU_SIZE_IND. The application is not allowed to generate media packets larger than the reported 'remoteMtuSize'.

The indication is sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_STREAM_MTU_SIZE_IND.
shandle	Stream handle.
remoteMtuSize	The maximum size media packet allowed in a single L2CAP frame by the remote device.
btConnId	Identifier used when moving the connection to another AMP controller, i.e. when calling the `CsrBtAmpmMoveReqSend`-function. The less significant 16 bits of this field contain the local connection identifier of the L2CAP connection established for the AV stream.

4.11 CSR_BT_AV_DISCOVER

Primitives \ Parameters	type	connectionId	tLabel	seidInfoCount	*seidInfo	avResponse	avResultCode	avResultSupplier
CSR_BT_AV_DISCOVER_REQ	✓	✓	✓					
CSR_BT_AV_DISCOVER_IND	✓	✓	✓					
CSR_BT_AV_DISCOVER_RES	✓	✓	✓	✓	✓	✓		
CSR_BT_AV_DISCOVER_CFM	✓	✓	✓	✓	✓		✓	✓

Table 11: CSR_BT_AV_DISCOVER Primitives

Description

A CSR_BT_AV_DISCOVER_REQ initiates the stream end-point discover procedure. When an application receives a CSR_BT_AV_DISCOVER_IND, it should respond back by sending a CSR_BT_AV_DISCOVER_RES with the stream end-points that it has. The initiator of the procedure will receive the response in a CSR_BT_AV_DISCOVER_CFM.

The confirmation and indication are sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_DISCOVER_REQ/IND/RES/CFM.
connectionId	Connection identifier.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15
seidInfoCount	Number of stream end-points returned in *seidInfo*
seidInfo	Pointer to the first structure element in a list of '*seidInfoCount*' elements containing stream end-point information (end-point identifier, media type (audio/video), SNK/SRC, and 'in use' status). Beware, that the receiver of the signal is always responsible for CsrPfree of the pointer.
avResponse	This parameter should be CSR_BT_AV_ACCEPT if the request was acceptable, and otherwise it should be one of the CSR_BT_RESULT_CODE_A2DP_ error codes found in csr_bt_av_prim.h.
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h

4.12 CSR_BT_AV_GET_CAPABILITIES

Primitives \ Parameters	type	connectionId	tLabel	acpSeid	servCapLen	*servCapData	avResponse	avResultCode	avResultSupplier
CSR_BT_AV_GET_CAPABILITIES_REQ	✓	✓	✓	✓					
CSR_BT_AV_GET_CAPABILITIES_IND	✓	✓	✓	✓					
CSR_BT_AV_GET_CAPABILITIES_RES	✓	✓	✓		✓	✓	✓		
CSR_BT_AV_GET_CAPABILITIES_CFM	✓	✓	✓		✓	✓		✓	✓
CSR_BT_AV_GET_ALL_CAPABILITIES_IND	✓	✓	✓	✓					
CSR_BT_AV_GET_ALL_CAPABILITIES_RES	✓	✓	✓		✓	✓	✓		

Table 12: CSR_BT_AV_GET_CAPABILITIES Primitives

Description

The application initiates the get capabilities procedure by sending a CSR_BT_AV_GET_CAPABILITIES_REQ in order to retrieve the capabilities of a stream identified by ‘acpSeid’.

The remote application will receive a CSR_BT_AV_GET_CAPABILITIES_IND and should respond by returning a CSR_BT_AV_GET_CAPABILITIES_RES with a list of the stream capabilities. If both devices comply with version 1.3 of the AVDTP spec, the remote application will receive a CSR_BT_AV_GET_ALL_CAPABILITIES_IND instead and shall answer with a CSR_BT_AV_GET_ALL_CAPABILITIES_RES message. The ACP application should only include application capabilities in the list, i.e. media codec and content protection capabilities (if supported). Transport capabilities will be added to the list by the AV. The INT application will receive the response in a CSR_BT_AV_GET_CAPABILITIES_CFM containing the complete list of stream capabilities (both application and transport capabilities).

The difference between the CSR_BT_AV_GET_CAPABILITIES_IND and the CSR_BT_GET_ALL_CAPABILITIES_IND is that the answer to the former shall not include “delay report“ information among its capabilities, while the answer to the latter shall contain delay reporting as a capability.

The confirmation and indication are sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_GET_CAPABILITIES_REQ/IND/RES/CFM.
connectionId	Connection identifier.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15
acpSeid	Stream endpoint identifier of the acceptor.
servCapLen	Length of service capability list (in bytes).
*servCapData	Pointer to service capability list.). Beware, that the receiver of the signal is always responsible for CsrPfree of the pointer

avResponse	This parameter should be CSR_BT_AV_ACCEPT if the request was acceptable, and otherwise it should be one of the CSR_BT_RESULT_CODE_A2DP_ error codes found in csr_bt_av_prim.h.
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h

4.13 CSR_BT_AV_SET_CONFIGURATION

Primitives \ Parameters	type	connectionId	tLabel	acpSeid	intSeid	appServCapLen	*appServCapData	shandle	servCapLen	*servCapData	servCategory	avResponse	avResultCode	avResultSupplier
CSR_BT_AV_SET_CONFIGURATION_REQ	✓	✓	✓	✓	✓	✓	✓							
CSR_BT_AV_SET_CONFIGURATION_IND	✓	✓	✓	✓	✓			✓	✓	✓				
CSR_BT_AV_SET_CONFIGURATION_RES	✓		✓					✓			✓	✓		
CSR_BT_AV_SET_CONFIGURATION_CFM	✓	✓	✓					✓			✓		✓	✓

Table 13: CSR_BT_AV_SET_CONFIGURATION Primitives

Description

The sending of a CSR_BT_AV_SET_CONFIGURATION_REQ initiates the configuration of a stream identified by the pair of INT and ACP stream end-point identifiers. The configuration pointed to by 'appServCapData' should only contain the application service categories, i.e. media codec and content protection service categories (if supported). The 'remoteServCapData' sent in the request should point to the service capability list returned in a previous CSR_BT_AV_GET_CAPABILITIES_CFM and is used by the AV to select a suitable configuration of the transport service categories.

The ACP application will receive the configuration in a CSR_BT_AV_SET_CONFIGURATION_IND and should attempt to configure the stream using the configuration pointed to by 'servCapData'. The ACP application returns a response by sending a CSR_BT_AV_SET_CONFIGURATION_RES. The INT application will receive the response in a CSR_BT_AV_SET_CONFIGURATION_CFM.

The confirmation and indication are sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_GET_CAPABILITIES_REQ/IND/RES/CFM.
connectionId	Connection identifier.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15
acpSeid	Stream end-point identifier of the acceptor.
intSeid	Stream end-point identifier of the initiator.
appServCapLen	Length of application service capabilities configuration (in bytes).
*appServCapData	Pointer to application service capabilities configuration.). Beware, that the receiver of the signal is always responsible for CsrPfree of the pointer

shandle	Stream handle, identifies the successfully configured stream.
servCapLen	Length of service capabilities configuration (in bytes).
*servCapData	Pointer to the complete list of service capabilities configuration (in bytes).). Beware, that the receiver of the signal is always responsible for CsrPfree of the pointer
servCategory	In case the configuration is rejected, this holds the value of service category element that caused the reject.
avResponse	This parameter should be CSR_BT_AV_ACCEPT if the request was acceptable, and otherwise it should be one of the CSR_BT_RESULT_CODE_A2DP_ error codes found in csr_bt_av_prim.h.
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h

4.14 CSR_BT_AV_GET_CONFIGURATION

Primitives \ Parameters	type	shandle	tLabel	servCapLen	*servCapData	avResultCode	avResultSupplier	avResponse
CSR_BT_AV_GET_CONFIGURATION_REQ	✓	✓	✓					
CSR_BT_AV_GET_CONFIGURATION_IND	✓	✓	✓					
CSR_BT_AV_GET_CONFIGURATION_CFM	✓	✓	✓	✓	✓	✓	✓	
CSR_BT_AV_GET_CONFIGURATION_RES	✓	✓	✓	✓	✓			✓

Table 14: CSR_BT_AV_GET_CONFIGURATION Primitives

Description

The application initiates the retrieval of a current stream configuration from the remote device by sending a CSR_BT_AV_GET_CONFIGURATION_REQ. When the ACP application receives a CSR_BT_AV_GET_CONFIGURATION_IND it should retrieve and return the configuration in a CSR_BT_AV_GET_CONFIGURATION_RES. The INT application will receive the response in a CSR_BT_AV_GET_CONFIGURATION_CFM.

The confirmation and indication are sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_GET_CONFIGURATION_REQ/IND/CFM/RES.
shandle	Stream handle.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15
servCapLen	Length of service capabilities configuration (in bytes).
*servCapData	Pointer to the complete list of service capabilities configuration (in bytes).). Beware, that the receiver of the signal is always responsible for CsrPfree of the pointer
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h
avResponse	This parameter should be CSR_BT_AV_ACCEPT if the request was acceptable, and otherwise it should be one of the CSR_BT_RESULT_CODE_A2DP_ error codes found in csr_bt_av_prim.h.

4.15 CSR_BT_AV_RECONFIGURE

Primitives \ Parameters	type	shandle	tLabel	servCapLen	*servCapData	servCategory	avResultCode	avResultSupplier	avResponse
CSR_BT_AV_RECONFIGURE_REQ	✓	✓	✓	✓	✓				
CSR_BT_AV_RECONFIGURE_IND	✓	✓	✓	✓	✓				
CSR_BT_AV_RECONFIGURE_CFM	✓	✓	✓			✓	✓	✓	
CSR_BT_AV_RECONFIGURE_RES	✓	✓	✓			✓			✓

Table 15: CSR_BT_AV_RECONFIGURE Primitives

Description

A stream reconfiguration is initiated by the sending of a CSR_BT_AV_RECONFIGURE_REQ. The reconfiguration procedure may only be initiated when the stream has previously been suspended. The configuration pointed to by 'servCapData' is only allowed to contain application service categories, i.e. media codec and content protection categories, if supported.

The ACP application should attempt to reconfigure and respond to a CSR_BT_AV_RECONFIGURE_IND by returning a CSR_BT_AV_RECONFIGURE_RES, which by the INT application is received as a CSR_BT_AV_RECONFIGURE_CFM.

The confirmation and indication are sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_RECONFIGURE_REQ/IND/CFM/RES.
shandle	Stream handle.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15
servCapLen	Length of application service capabilities configuration (in bytes).
*servCapData	Pointer to application service capabilities configuration.). Beware, that the receiver of the signal is always responsible for CsrPfree of the pointer
servCategory	In case the configuration is rejected, this holds the value of service category element that caused the reject.
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h

avResponse	This parameter should be CSR_BT_AV_ACCEPT if the request was acceptable, and otherwise it should be one of the CSR_BT_RESULT_CODE_A2DP_ error codes found in csr_bt_av_prim.h.

4.16 CSR_BT_AV_OPEN

Parameters / Primitives	type	shandle	tLabel	avResponse	avResultCode	avResultSupplier
CSR_BT_AV_OPEN_REQ	✓	✓	✓			
CSR_BT_AV_OPEN_IND	✓	✓	✓			
CSR_BT_AV_OPEN_CFM	✓	✓	✓		✓	✓
CSR_BT_AV_OPEN_RES	✓	✓	✓	✓		

Table 16: CSR_BT_AV_OPEN Primitives

Description

The stream is opened by the application sending CSR_BT_AV_OPEN_REQ to AV. The stream must previously have been configured before this procedure is initiated. The ACP application should respond to a CSR_BT_AV_OPEN_IND by returning a CSR_BT_AV_OPEN_RES, which by the INT application is received as a CSR_BT_AV_OPEN_CFM.

The confirmation and indication are sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_OPEN_REQ/IND/CFM/RES.
shandle	Stream handle.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15
avResponse	This parameter should be CSR_BT_AV_ACCEPT if the request was acceptable, and otherwise it should be one of the CSR_BT_RESULT_CODE_A2DP_ error codes found in csr_bt_av_prim.h.
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h

4.17 CSR_BT_AV_START

Primitives \ Parameters	type	connectionId	tLabel	listLength	list	reject_shandle	avResponse	avResultCode	avResultSupplier
CSR_BT_AV_START_REQ	✓		✓	✓	✓				
CSR_BT_AV_START_IND	✓		✓	✓	✓				
CSR_BT_AV_START_RES	✓		✓	✓	✓	✓	✓		
CSR_BT_AV_START_CFM	✓	✓	✓			✓		✓	✓

Table 17: CSR_BT_AV_START Primitives

Description

The starting of one or more streams is initiated by sending a CSR_BT_AV_START_REQ. The ACP should upon receiving a CSR_BT_AV_START_IND, attempt to start each stream in the list of stream handles (‘first_shandle’) starting from the top of the list. If the ACP application fails to start a stream a response must be returned containing the handle of the stream, which fails. Any already started streams shall remain started. If all streams are started the ACP application should return a response indicating success. The response is received by the INT application by a CSR_BT_AV_START_CFM.

The confirmation and indication are sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_START_REQ/IND/RES/CFM.
connectionId	The connection instance ID for which the stream start has happened.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15
listLength	Number of stream handles in list.
list	The first array element in a list of ‘listLength’ elements containing stream handles. The stream handles must be listed in ascending order.
reject_shandle	In case a stream fails to start, the stream handles of this stream.
avResponse	This parameter should be CSR_BT_AV_ACCEPT if the request was acceptable, and otherwise it should be one of the CSR_BT_RESULT_CODE_A2DP_ error codes found in csr_bt_av_prim.h.
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h

4.18 CSR_BT_AV_CLOSE

Primitives \ Parameters	type	shandle	tLabel	avResponse	avResultCode	avResultSupplier
CSR_BT_AV_CLOSE_REQ	✓	✓	✓			
CSR_BT_AV_CLOSE_IND	✓	✓	✓			
CSR_BT_AV_CLOSE_RES	✓	✓	✓	✓		
CSR_BT_AV_CLOSE_CFM	✓	✓	✓		✓	✓

Table 18: CSR_BT_AV_CLOSE Primitives

Description

The closing of a stream is initiated by the sending of a CSR_BT_AV_CLOSE_REQ. The ACP application should respond to a CSR_BT_AV_CLOSE_IND by returning a CSR_BT_AV_CLOSE_RES, which by the INT application is received as a CSR_BT_AV_CLOSE_CFM.

The confirmation and indication are sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_CLOSE_REQ/IND/RES/CFM.
shandle	Stream handle.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15
avResponse	This parameter should be CSR_BT_AV_ACCEPT if the request was acceptable, and otherwise it should be one of the CSR_BT_RESULT_CODE_A2DP_ error codes found in csr_bt_av_prim.h.
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h

4.19 CSR_BT_AV_SUSPEND

Primitives \ Parameters	type	connectionId	tLabel	listLength	list	reject_shandle	avResultCode	avResultSupplier	avResponse
CSR_BT_AV_SUSPEND_REQ	✓		✓	✓	✓				
CSR_BT_AV_SUSPEND_IND	✓		✓	✓	✓				
CSR_BT_AV_SUSPEND_CFM	✓	✓	✓			✓	✓	✓	
CSR_BT_AV_SUSPEND_RES	✓		✓	✓	✓	✓			✓

Table 19: CSR_BT_AV_SUSPEND Primitives

Description

The application can initiate the suspend procedure by sending a CSR_BT_AV_SUSPEND_REQ and suspend one or more streams.

The confirmation and indication are sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_SUSPEND_REQ/IND/CFM/RES.
connectionId	The connection instance ID for which the suspend has happened.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15.
listLength	Number of stream handles in list.
list	Array of stream handles.
reject_shandle	In case a stream fails to start, the stream handles of this stream.
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h
avResponse	This parameter should be CSR_BT_AV_ACCEPT if the request was acceptable, and otherwise it should be one of the CSR_BT_RESULT_CODE_A2DP_ error codes found in csr_bt_av_prim.h.

4.20 CSR_BT_AV_ABORT

Parameters / Primitives	type	shandle	tLabel
CSR_BT_AV_ABORT_REQ	✓	✓	✓
CSR_BT_AV_ABORT_IND	✓	✓	✓
CSR_BT_AV_ABORT_RES	✓	✓	✓
CSR_BT_AV_ABORT_CFM	✓	✓	✓

Table 20: CSR_BT_AV_ABORT Primitives

Description

A stream can be aborted by the application by sending a CSR_BT_AV_ABORT_REQ. If the ACP application accepts the CSR_BT_AV_ABORT_IND that it receives it should respond by returning a CSR_BT_AV_ABORT_RES, which by the INT application is received as a CSR_BT_AV_ABORT_CFM.

The confirmation and indication are sent to the task identified by streamAppHandle provided in the activation procedure.

Parameters

type	Signal identity CSR_BT_AV_ABORT_REQ/IND/RES/CFM.
shandle	Stream handle.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15.

4.21 CSR_BT_AV_STATUS

Primitives \ Parameters	type	statusType	roleType	connectionId	appHandle
CSR_BT_AV_STATUS_IND	✓	✓	✓	✓	✓

Table 21: CSR_BT_AV_STATUS Primitives

Description

If two separate AV applications exist, i.e. a control application and a stream application, then the stream application will be sent a CSR_BT_AV_STATUS_IND each time a control event occurs. The CSR_BT_AV_STATUS_IND contains information about connection establishment/release and activation/deactivation events.

Parameters

type	Signal identity CSR_BT_AV_STATUS_IND.
statusType	The type of signal that spawned the indication. Possible values are: CSR_BT_AV_ACTIVATE_CFM CSR_BT_AV_DEACTIVATE_CFM CSR_BT_AV_CONNECT_CFM CSR_BT_AV_CONNECT_IND CSR_BT_AV_DISCONNECT_IND
roleType	The service/role successfully activated (valid only for statusType CSR_BT_AV_ACTIVATE_CFM and CSR_BT_AV_DEACTIVATE_CFM).
connectionId	Identifier of the connection (valid only for statusType CSR_BT_AV_CONNECT_CFM, CSR_BT_AV_CONNECT_IND and CSR_BT_AV_DISCONNECT_IND).
appHandle	Handle for the task managing connection setup.

4.22 CSR_BT_AV_SECURITY_CONTROL

Primitives \ Parameters	type	shandle	tLabel	contProtMethodLen	*contProtMehodData	avResponse	avResultCode	avResultSupplier
CSR_BT_AV_SECURITY_CONTROL_REQ	✓	✓	✓	✓	✓			
CSR_BT_AV_SECURITY_CONTROL_IND	✓	✓	✓	✓	✓			
CSR_BT_AV_SECURITY_CONTROL_RES	✓	✓	✓	✓	✓	✓		
CSR_BT_AV_SECURITY_CONTROL_CFM	✓	✓	✓	✓	✓		✓	✓

Table 22: CSR_BT_AV_SECURITY_CONTROL Primitives

Description

An AV application can use the CSR_BT_AV_SECURITY_REQ to pass data related to content protection to the remote device. An application that receives a CSR_BT_AV_SECURITY_IND should respond with a CSR_BT_AV_SECURITY_RES. The response is returned in a CSR_BT_AV_SECURITY_CFM. The data exchanged with these primitives is dependent on the used content protection method which is outside the scope of this API.

Parameters

type	Signal identity CSR_BT_AV_SECURITY_CONTROL_REQ/IND/RES/CFM
shandle	Stream handle.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15
contProtMethodLen	Length of the content protection method specific data exchanged.
*contProtMethodData	Pointer to content protection method specific data.). Beware, that the receiver of the signal is always responsible for CsrPfree of the pointer
avResponse	This parameter should be CSR_BT_AV_ACCEPT if the request was acceptable, and otherwise it should be one of the CSR_BT_RESULT_CODE_A2DP_ error codes found in csr_bt_av_prim.h.
avResultCode	The result code of the operation. Possible values depend on the value of avResultSupplier. If e.g. avResultSupplier == CSR_BT_SUPPLIER_AV or avResultSupplier == CSR_BT_SUPPLIER_A2DP then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
avResultSupplier	This parameter specifies the supplier of the result given in avResultCode. Possible values can be found in csr_bt_result.h

4.23 CSR_BT_AV_SECURITY_IN / OUT

Primitives \ Parameters	type	appHandle	secLevel	resultCode	resultSupplier
CSR_BT_AV_SECURITY_IN_REQ	✓	✓	✓		
CSR_BT_AV_SECURITY_IN_CFM	✓			✓	✓
CSR_BT_AV_SECURITY_OUT_REQ	✓	✓	✓		
CSR_BT_AV_SECURITY_OUT_CFM	✓			✓	✓

Table 23: CSR_BT_AV_SECURITY_IN and CSR_BT_AV_SECURITY_OUT Primitives

Description

Applications that wish to change the enforcement to a specific profile security level, i.e. authentication, encryption and/or authorisation, can use this API to set up the security level for *new* connections. Note that this API is for the local device only and can be used from within any state.

The *CSR_BT_SECURITY_IN_REQ* signal sets up the security level for new incoming connections. Already established or pending connections are not altered.

The *CSR_BT_SECURITY_OUT_REQ* signal sets up the security level for new outgoing connections. Already established and pending connections are not altered. Note that *authorisation* should not be used for outgoing connections as that may be confusing for the user – there is really no point in requesting an outgoing connection and afterwards having to authorise as they are both locally-only decided procedures.

Note, that any attempts to set security to a less secure level than the mandatory security level will be rejected. See csr_bt_profiles.h for mandatory security settings. The default settings used by CSR Synergy Bluetooth are set to require authentication and encryption.

Note that if MITM protection is requested and the remote device does not have the required IO capabilities, pairing/bonding will fail and connections to the remote device *cannot* be made. See [SC] for further details.

Parameters

type — Signal identity CSR_BT_AV_SECURITY_IN/OUT_REQ/CFM

appHandle — Application handle to which the confirm message is sent.

secLevel — The application must specify one of the following values:

- CSR_BT_SEC_DEFAULT : Use default security settings
- CSR_BT_SEC_MANDATORY : Use mandatory security settings
- CSR_BT_SEC_SPECIFY : Specify new security settings

If CSR_BT_SEC_SPECIFY is set the following values can be OR'ed additionally:

- CSR_BT_SEC_AUTHORISATION: Require authorisation
- CSR_BT_SEC_AUTHENTICATION: Require authentication

- CSR_BT_SEC_ SEC_ENCRYPTION: Require encryption (implies
- authentication)
- CSR_BT_SEC_MITM: Require MITM protection (implies encryption)

resultCode	The result code of the operation. Possible values depend on the value of resultSupplier. If e.g. the resultSupplier == CSR_BT_SUPPLIER_CM then the possible result codes can be found in csr_bt_cm_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
resultSupplier	This parameter specifies the supplier of the result given in resultCode. Possible values can be found in csr_bt_result.h

4.24 CSR_BT_AV_SET_QOS_INTERVAL

Primitives \ Parameters	type	qosInterval
CSR_BT_AV_SET_QOS_INTERVAL_REQ	✓	✓

Table 24: CSR_BT_AV_SET_QOS_INTERVAL Primitives

Description

An AV application can use the CSR_BT_AV_SET_QOS_INTERVAL_REQ message to change the interval in which the CSR_BT_AV_QOS_IND is sent to the stream application. If the interval is set to zero (0), then only when the buffer is full and when the buffer is subsequently emptied a CSR_BT_AV_QOS_IND will be sent. Note that if the stream is closed or suspended the AV buffer will automatically be flushed and in this case a CSR_BT_AV_QOS_IND indicating 'empty' is not sent to the stream application.

The CSR_BT_AV_QOS_IND message is describe in section 4.9.

Parameters

type	Signal identity CSR_BT_AV_SET_QOS_INTERVAL_REQ.
qosInterval	Sets the interval in which the CSR_BT_AV_QOS_IND is sent to the stream application.

4.25 CSR_BT_AV_LP_NEG_CONFIG

Primitives \ Parameters	type	enable
CSR_BT_AV_LP_NEG_CONFIG	✓	✓

Table 25: CSR_BT_AV_LP_NEG_CONFIG Primitive

Description

Per default, the AV profile will try to enter low power mode (sniff) when there is no data traffic on a connection. However, this can be a drawback in applications where the audio data streamed is sent over the air without the AV profiles intervention. Entering sniff mode in this case will result in poor audio quality. To avoid this, an AV application can use the CSR_BT_AV_LP_NEG_CONFIG_REQ message to enable or disable the low power mode negotiation process started by the local device. If the “enable” field is set to FALSE, the local device will not start the procedure to enter low power mode. Once the request has been issued, if the local device shall try to set a connection in low power mode later, the application must issue this signal again, but with the “enable” field set to TRUE.

In any case, the local device will accept attempts from the remote device to enter low power mode

Parameters

type	Signal identity CSR_BT_AV_LP_NEG_CONFIG_REQ.
enable	Boolean to determine whether the local device shall start low power mode negotiation or not.

4.26 CSR_BT_AV_GET_CHANNEL_INFO

Primitives \ Parameters	type	btConnId	shandle	aclHandle	remoteCid	resultCode	resultSupplier
CSR_BT_AV_GET_CHANNEL_INFO_REQ	✓	✓					
CSR_BT_AV_GET_STREAM_CHANNEL_INFO_REQ	✓		✓				
CSR_BT_AV_GET_CHANNEL_INFO_CFM	✓			✓	✓	✓	✓

Table 26: CSR_BT_AV_GET_CHANNEL_INFO Primitives

Description

The AV application may need to get information about the ACL handle and the remote channel ID used in an A2DP control or stream connection. This is needed for instance if the on-chip encoder feature is used. Beware, this feature is not available on all BlueCore versions. Refer to the chip documentation for details.

When the information requested is for an AV control connection, the application must use the CSR_BT_AV_GET_CHANNEL_INFO_REQ message. If the information requested is for an A2DP stream, the application shall use the CSR_BT_AV_GET_STREAM_CHANNEL_INFO_REQ message. In both cases, the AV profile will answer with a CSR_BT_AV_GET_CHANNEL_INFO_CFM.

Note: In order to use the on-chip encoder feature, the code must be built using the compiler flag "CSR_DSPM_ENABLE".

Parameters

type	Signal identity CSR_BT_AV_GET_CHANNEL_INFO_REQ, CSR_BT_AV_GET_STREAM_CHANNEL_INFO_REQ or CSR_BT_AV_GET_CHANNEL_INFO_CFM.
btConnId	Connection ID of the AV control connection.
Shandle	Handle ID of the stream connection
aclHandle	Handle ID of the ACL connection to the remote device
remoteCid	Destination channel ID of the connection
resultCode	The result code of the operation. Possible values depend on the value of resultSupplier. If e.g. the resultSupplier == CSR_BT_SUPPLIER_AV then the possible result codes can be found in csr_bt_av_prim.h. All values which are currently not specified in the respective prim.h file are regarded as reserved and the application should consider them as errors.
resultSupplier	This parameter specifies the supplier of the result given in resultCode. Possible values can be found in csr_bt_result.h

4.27 CSR_BT_AV_DELAY_REPORT

Primitives \ Parameters	type	tLabel	delay	connectionId
CSR_BT_AV_DELAY_REPORT_REQ	✓	✓	✓	✓
CSR_BT_AV_DELAY_REPORT_IND	✓		✓	✓

Table 27: CSR_BT_AV_DELAY_REPORT Primitive

Description

In order to allow synchronization between audio and video streaming, the sink device needs to let the source device know whether there is a delay in playing the audio or the video at the sink. Thus, the source can compensate for this delay when streaming the data. This is a feature introduced after version 1.3 of the specification. The sink device will send the CSR_BT_AV_DELAY_REPORT_REQ message and the source device will then receive the CSR_BT_AV_DELAY_REPORT_IND. No confirmation or response messages are needed. The sink device must always be the initiator of this information exchange.

Parameters

type	Signal identity CSR_BT_AV_DELAY_REPORT_REQ / IND.
tLabel	Transaction label. Tag identifying a message transaction so that a response can be matched against its command. Valid range: 0 – 15
delay	Delay in tenths of milliseconds. CsrUint16 type; max value is 6 seconds.
connectionId	Connection label.

5 Application Generated Result Codes

Reproduction of the error code table 8.18.6.2, page 81 in [AV].

ACP to INT, Signal Response Header Error Codes			
ID	**Related Signalling Command**	**Error Abbreviation**	**Error Description**
0x01	All messages	CSR_BT_AV_BAD_HEADER_FORMAT	The request packet header format error that is not specified above ERROR_CODE

ACT to INT, Signal Response Payload Format Error Codes			
ID	**Related Signalling Command**	**Error Abbreviation**	**Error Description**
0x11	All messages	CSR_BT_AV_BAD_LENGTH	The request packet length does not match the assumed length.
0x12	All messages	CSR_BT_AV_BAD_ACP_SEID	The requested command indicates an invalid ACP SEID (not addressable).
0x13	Set Configuration	CSR_BT_AV_SEP_IN_USE	The SEP is in use.
0x14	Reconfigure	CSR_BT_AV_SEP_NOT_IN_USE	The SEP is not in use.
0x17	Set Configuration Reconfigure	CSR_BT_AV_BAD_SERV_CATEGORY	The value of the Service Category in the request packet is not defined in AVDTP.
0x18	All messages	CSR_BT_AV_BAD_PAYLOAD_FORMAT	The requested command has an incorrect payload format (Format errors not specified in this ERROR CODE).
0x19	All messages	CSR_BT_AV_NOT_SUPPORTED_COMMAND	The requested command is not supported by the device.
0x1A	Reconfigure	CSR_BT_AV_INVALID_CAPABILITIES	The reconfigure command is an attempt to reconfigure transport service capabilities of the SEP. Reconfigure is only permitted for application service capabilities.

ACP to INT, Signal Response Transport Service Capabilities Error Codes			
ID	**Related Signalling Command**	**Error Abbreviation**	**Error Description**
0x22	Set Configuration	CSR_BT_AV_BAD_RECOVERY_TYPE	The requested Recovery Type is not defined in AVDTP.
0x23	Set Configuration	CSR_BT_AV_BAD_MEDIA_TRANSPORT_FORMAT	The format of Media Transport Capability is not correct.
0x25	Set Configuration	CSR_BT_AV_BAD_RECOVERY_FORMAT	The format of the Recovery Service Capability is not correct.
0x26	Set Configuration	CSR_BT_AV_BAD_ROHC_FORMAT	The format of Header Compression Service Capability is not correct.
0x27	Set Configuration	CSR_BT_AV_BAD_CP_FORMAT	The format of Content Protect Service Capability is not correct.
0x28	Set Configuration	CSR_BT_AV_BAD_MULTIPLEXING_FORMAT	The format of Multiplexing Service Capability is not correct.
0x29	Set Configuration	CSR_BT_AV_UNSUPPORTED_CONFIGURATION	Configuration not supported.

ACP to INT, Procedure Error Codes			
ID	**Related Signalling Command**	**Error Abbreviation**	**Error Description**
0x31	All messages	CSR_BT_AV_BAD_STATE	Indicates that the ACP state machine is in an invalid state in order to process the signal.

Reproduction of the error code table 5.3, page in [A2DP].

Error ID	Related Signaling Command	Related CODEC	Error Abbreviation	Error Description
0xC1	Set Configuration Reconfigure	ALL	CSR_BT_AV_INVALID_CODEC_TYPE	Media Codec Type is not valid
0xC2	Set Configuration Reconfigure	ALL	CSR_BT_AV_NOT_SUPPORTED_CODEC_TYPE	Media Codec Type is not supported
0xC3	Set Configuration Reconfigure	ALL	CSR_BT_AV_INVALID_SAMPLING_FREQUENCY	Sampling Frequency is not valid, or multiple values have been selected
0xC4	Set Configuration Reconfigure	ALL	CSR_BT_AV_NOT_SUPPORTED_SAMPLING_FREQUENCY	Sampling Frequency is not supported
0xC5	Set Configuration Reconfigure	SBC, Mpeg-1,2 Audio, ATTRAC family	CSR_BT_AV_INVALIND_CHANNEL_MODE	Channel Mode is not valid, or multiple values have been selected
0xC6	Set Configuration Reconfigure	SBC, Mpeg-1,2 Audio, ATTRAC family	CSR_BT_AV_NOT_SUPPORTED_CHANNEL_MODE	Channel Mode is not supported
0xC7	Set Configuration Reconfigure	SBC	CSR_BT_AV_INVALID_SUBBANDS	None or multiple values have been selected for Number of Subbands
0xC8	Set Configuration Reconfigure	SBC	CSR_BT_AV_NOT_SUPPORTED_SUBBANDS	Number of Subbands is not supported
0xC9	Set Configuration Reconfigure	SBC	CSR_BT_AV_INVALID_ALLOCATION_METHOD	None or multiple values have been selected for Allocation Method
0xCA	Set Configuration Reconfigure	SBC	CSR_BT_AV_NOT_SUPPORTED_ALLOCATION_METHOD	Allocation Method is not supported
0xCB	Set Configuration Reconfigure	SBC	CSR_BT_AV_INVALID_MINIMUM_BITPOOL_VALUE	Minimum Bitpool Value is not valid
0xCC	Set Configuration Reconfigure	SBC	CSR_BT_AV_NOT_SUPPORTED_MINIMUM_BITPOOL_VALUE	Minimum Bitpool Value is not supported
0xCD	Set Configuration Reconfigure	SBC	CSR_BT_AV_INVALID_MAXIMUM_BITPOOL_VALUE	Maximum Bitpool Value is invalid
0xCE	Set Configuration Reconfigure	SBC	CSR_BT_AV_NOT_SUPPORTED_MAXIMUM_BITPOOL_VALUE	Maximum Bittpool Value is not supported

Error ID	Related Signaling Command	Related CODEC	Error Abbreviation	Error Description
0xCF	Set Configuration Reconfigure	Mpeg-1,2 Audio	CSR_BT_AV_INVALID_LAYER	None or multiple values have been selected for Layer
0xD0	Set Configuration Reconfigure	Mpeg-1,2 Audio	CSR_BT_AV_NOT_SUPPORTED_LAYER	Layer is not supported
0xD1	Set Configuration Reconfigure	Mpeg-1,2 Audio	CSR_BT_AV_NOT_SUPPORTED_CRC	CRC is not supported
0xD2	Set Configuration Reconfigure	Mpeg-1,2 Audio	CSR_BT_AV_NOT_SUPPOERTED_MPF	MPF-2 is not supported
0xD3	Set Configuration Reconfigure	Mpeg-1,2 Audio, Mpeg-2,4 AAC, ATRAC family	CSR_BT_AV_NOT_SUPPORTED_VBR	VBR is not supported
0xD4	Set Configuration Reconfigure	Mpeg-1,2 Audio, ATTRAC family	CSR_BT_AV_INVALID_BIT_RATE	None or multiple values have been selected for Bit Rate
0xD5	Set Configuration Reconfigure	Mpeg-1,2 Audio, Mpeg-2,4 AAC, ATRAC family	CSR_BT_AV_NOT_SUPPORTED_BIT_RATE	Bit Rate is not supported
0xD6	Set Configuration Reconfigure	Mpeg-2,4 AAC,	CSR_BT_AV_INVALID_OBJECT_TYPE	Either 1) Object type is not valid(b3-b0) or 2) None or multiple values have been selected for Object Type
0xD7	Set Configuration Reconfigure	Mpeg-2,4 AAC,	CSR_BT_AV_NOT_SUPPORTED_OBJECT_TYPE	Object Type is not supported
0xD8	Set Configuration Reconfigure	Mpeg-2,4 AAC,	CSR_BT_AV_INVALID_CHANNELS	None or multiple values have been selected for Channels
0xD9	Set Configuration Reconfigure	Mpeg-2,4 AAC,	CSR_BT_AV_NOT_SUPPORTED_CHANNELS	Channels is not supported
0xDA	Set Configuration Reconfigure	ATRAC family	CSR_BT_AV_INVALID_VERSION	Version is not valid
0xDB	Set Configuration Reconfigure	ATRAC family	CSR_BT_AV_NOT_SUPPORTED_VERSION	Version is not supported
0xDC	Set Configuration Reconfigure	ATRAC family	CSR_BT_AV_NOT_SUPPORTED_MAX_SUL	Maximum SUL is not Acceptable for the Decoder in the SNK

Error ID	Related Signaling Command	Related CODEC	Error Abbreviation	Error Description
0xDD	Set Configuration Reconfigure	SBC	CSR_BT_AV_INVALID_BLOCK_LENGTH	None or multiple values have been selected for Block Length
0xDE-0xDF				RFD
0xE0	Set Configuration Reconfigure	ALL	CSR_BT_AV_INVALID_CP_TYPE	The requested CP Type is not supported
0xE0	Set Configuration Reconfigure	ALL	CSR_BT_AV_INVALID_CP_FORMAT	The format of Content Protection Service, Content Protection Scheme Dependent Data is not correct
0xE2-0xFF				RFD

6 Document References

Document	Reference
Advanced Audio Distribution Profile Specification Version: 1.2 Date: : 2007-04-16	[A2DP]
Audio/Video Distribution Transport Protocol Specification Version: 1.2 Date: 2007-04-16	[AV]
Video Distribution Profile Version: 1.0 Date: 2004-09-08	[VDP]
Generic Audio/Video Distribution Profile Specification Version: 1.2 Date: : 2007-04-16	[GAVDP]
CSR Synergy Bluetooth, SC – Security Controller API Description	[SC]

Terms and Definitions

AV	Audio/Video
ACP	Acceptor
BlueCore®	Group term for CSR's range of Bluetooth wireless technology chips
Bluetooth®	Set of technologies providing audio and data transfer over short-range radio connections
CSR	Cambridge Silicon Radio
INT	Initiator
SNK	Sink
SRC	Source
UniFi™	Group term for CSR's range of chips designed to meet IEEE 802.11 standards

Document History

Revision	Date	History
1	26 SEP 11	Ready for release 18.2.0
2	23 JAN 12	Ready for release 18.2.2

TradeMarks, Patents and Licences

Unless otherwise stated, words and logos marked with ™ or ® are trademarks registered or owned by CSR plc or its affiliates. Bluetooth® and the Bluetooth logos are trademarks owned by Bluetooth SIG, Inc. and licensed to CSR. Other products, services and names used in this document may have been trademarked by their respective owners.

The publication of this information does not imply that any licence is granted under any patent or other rights owned by CSR plc.

CSR reserves the right to make technical changes to its products as part of its development programme.

While every care has been taken to ensure the accuracy of the contents of this document, CSR cannot accept responsibility for any errors.

Life Support Policy and Use in Safety-critical Compliance

CSR's products are not authorised for use in life-support or safety-critical applications. Use in such applications is done at the sole discretion of the customer. CSR will not warrant the use of its devices in such applications.

Performance and Conformance

Refer to www.csrsupport.com for compliance and conformance to standards information.