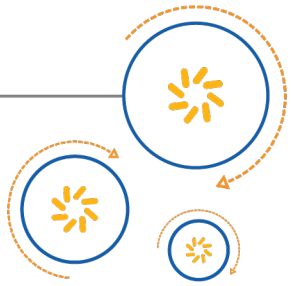
Qualcomm Technologies International, Ltd.



Audio Sink Application

User Guide

80-CT439-1 Rev. AR

October 30, 2017

Qualcomm Technologies International, Ltd. (formerly known as Cambridge Silicon Radio Limited) is a company registered in England and Wales with a registered office at: Churchill House, Cambridge Business Park, Cowley Road, Cambridge, CB4 0WZ, United Kingdom.
Registered Number: 3665875 | VAT number: GB787433096

Revision history

Revision	Date	Description
1	FEB 2012	Original publication of this document. Alternative document number CS-00236868-UG.
2	DEC 2013	Updated for ADK 3 and to the new CSR™ style
3	JAN 2014	Minor editorial corrections
4	JUN 2014	Updated for ADK 3.5
5	AUG 2014	Correction to Document References
6	AUG 2014	Guidance added for external audio amplifier power management feature using PIOs.
7	AUG 2014	Section 11.4 updated
8	JUN 2015	Updated and refactored for ADK 4.0
9	OCT 2015	Updated for ADK 4.0.1
10	FEB 2016	Editorial corrections
11	SEP 2016	Updated for ADK 4.1. Updated to conform to QTIL style
12	SEP 2016	Minor editorial correction
13	APR 2017	Updated for ADK 4.2
14	MAY 2017	Updated to remove aptX Specific tone
AR	OCT 2017	Added to the Content Management System. DRN updated to use Agile Number. No change to technical content.

Contents

Tables

Figures

1 Overview of the Audio Sink application

The Audio Sink application is a modifiable reference software package implementation of an audio rendering device supporting Bluetooth technologies. It is an on-chip virtual machine application designed to run explicitly on QTIL hardware platforms.

1.1 Audio Bluetooth profiles supported by the Audio Sink application

The Bluetooth profiles the reference Sink application implements are:

Traditional Bluetooth profiles

- Headset Profile v1.2
- Hands free Profile v1.7
- Advanced Audio Distribution Profile v1.3.1
- Audio/Video Remote Control Profile v1.6
- Message Access Profile v1.1
- Phone Book Access Profile v1.1.1

GATT-based Bluetooth low energy technology profiles

- Generic Attribute Profile
- Generic Access Profile
- Battery Service v1.0
- Device Information Service v1.1
- Immediate Alert Service v1.0
- Apple Notification Centre Service
- HID Over GATT Profile v1.0
- Transmit Power Service v1.0
- Link Loss Service v1.0.1
- Heart Rate Service v1.0

QTIL proprietary Bluetooth profiles

- Subwoofer Audio Transfer
- Generic Application Interface Architecture
- Bluetooth Low Energy technology Broadcast Service

1.2 Non Bluetooth interfaces supported by the Audio Sink application

Interfaces supported by the Audio Sink appliction that are not explicitly related to Bluetooth Profiles are:

- PCM Audio (Analog)
- I^2S Audio
- SPDIF Audio (Wired and Optical)
- USB (Audio and HID enumeration)
- FM receive
- Infra-red receive

2 Hardware

The Audio Sink application software package is designed to run on QTIL CSR8670 and CSR8675 Qualcomm® BlueCore™ ICs. These are available as QTIL Hardware Development platforms.

2.1 CNS10001V4 reference hardware design for CSR8670

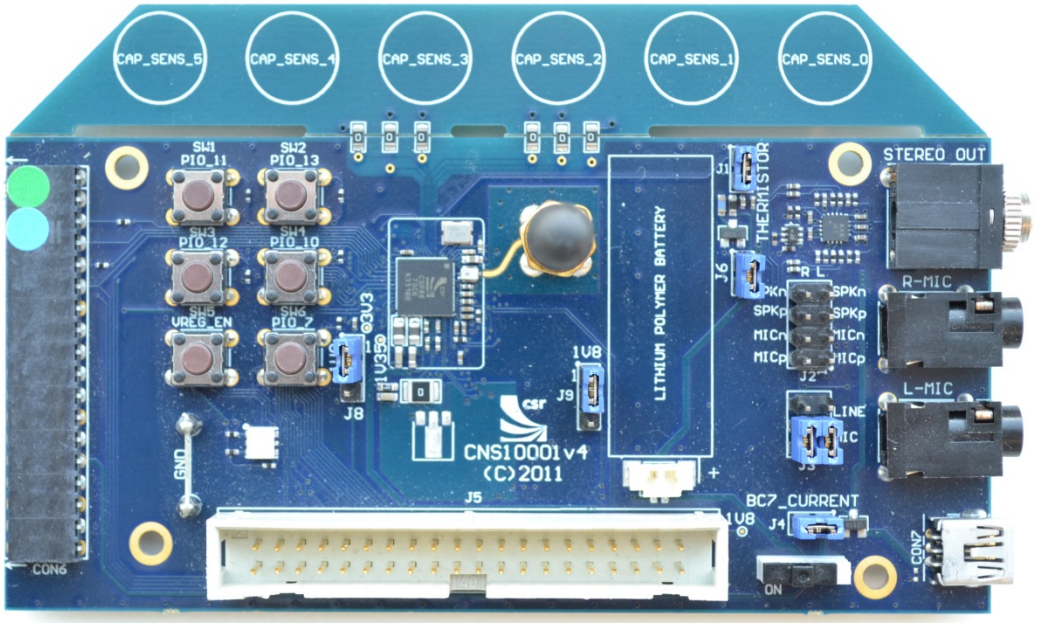


Figure 2-1 CNS1001v4 development board

2.2 H13179 reference design board

The H13179 reference design board paired with the H13478 module for CSR8675.

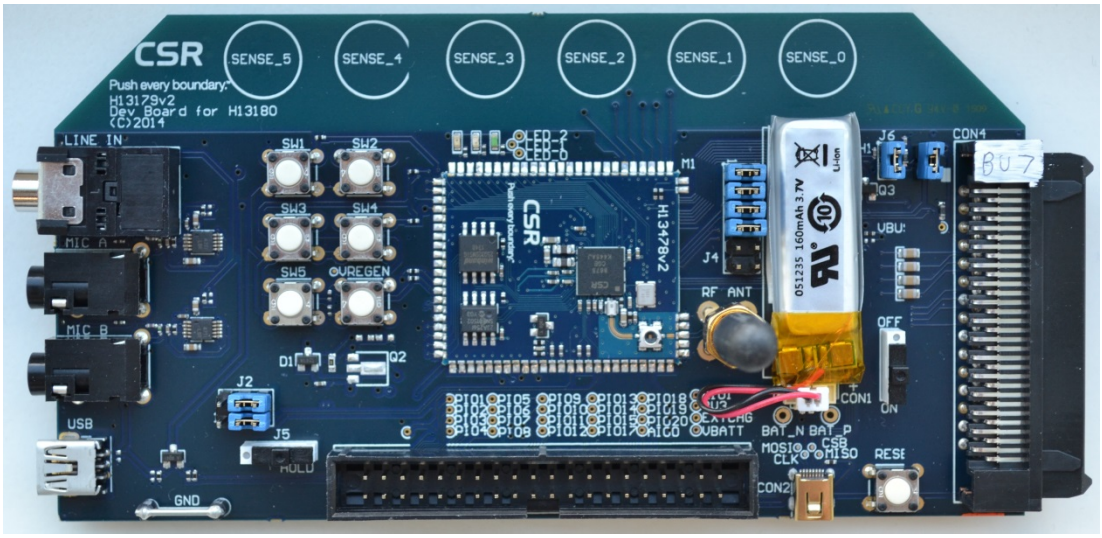


Figure 2-2 H13179v2 development board with the H13478v2 module

2.3 Supported plugin boards (for H13179 Carrier)

There are a number of QTIL plugin boards that provide reference hardware designs with support for additional hardware features, see:

- H13223 analog headphone amplifier
- Digital amplifier
- H13596 multichannel amplifier
- CNS11007 PCM audio
- H13191 SPDIF interface and infra-red receive
- CNS10010 display and FM receive

2.3.1 H13223 analog headphone amplifier

This plugin board provides an amplified headphone output using an analog stereo headphone amplifier.

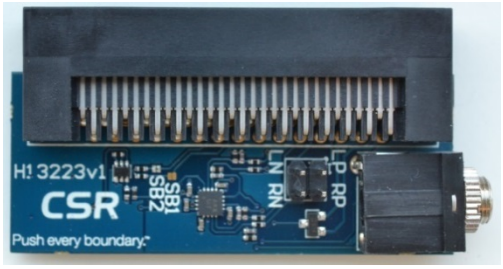


Figure 2-3 H13223v1 development board

2.3.2 Digital amplifier

This plugin board provides an amplified stereo output using a digital I^2S amplifier.

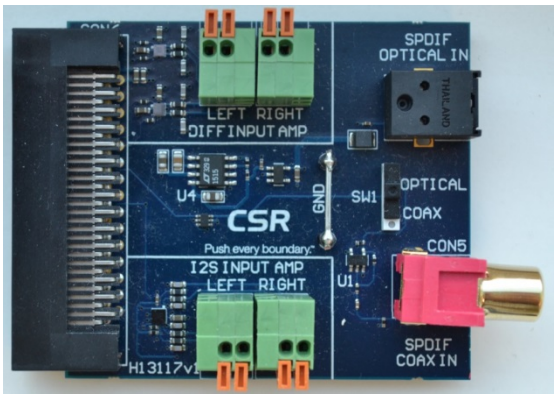


Figure 2-4 H13117v1 development board

2.3.3 H13596 multichannel amplifier

This plugin board provides multi-channel amplified audio outputs, using both analog and I^2S amplifiers. It has two TAS5715 I2S amplifiers, each of which has a main output, labeled **I2S1/2** and a headphones output, labeled **I2S 1/2 HEADPHONES**. Each amplifier is able to output either on the main output or the headphones output.

The **SPKR** output of **CON4** is routed to the **CODEC** and the **CODEC HEADPHONES** output terminals on the multi-channel amplifier. The **CODEC HEADPHONES** output has a **JACK_DET** line connected

to the **AMP_MISO** input of **CON4**, which is connected to PIO[3] on the H13179v2 board. **JACK_DET** is high when the headphones jack is inserted and low when the headphones jack is removed.

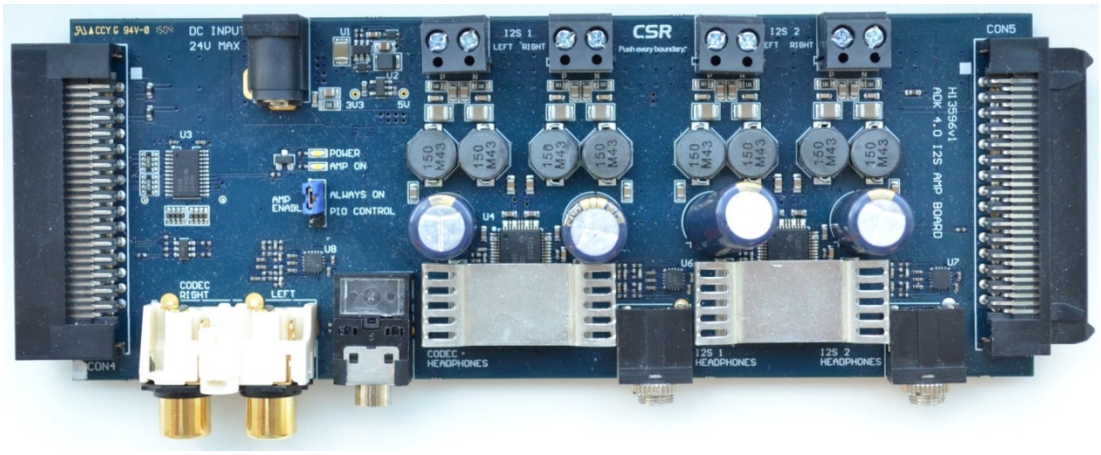


Figure 2-5 H13596v1 development board

2.3.4 CNS11007 PCM audio

This plugin board provides a PCM Audio input source.

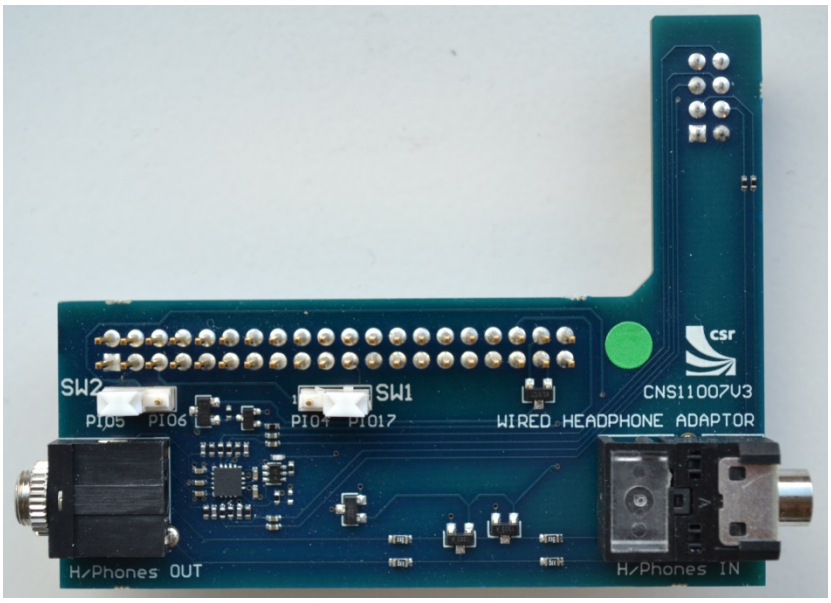


Figure 2-6 CNS11007v3 development board

2.3.5 H13191 SPDIF interface and infra-red receive

This plugin board provides a SPDIF interface (input and output) along with infra-red receive.

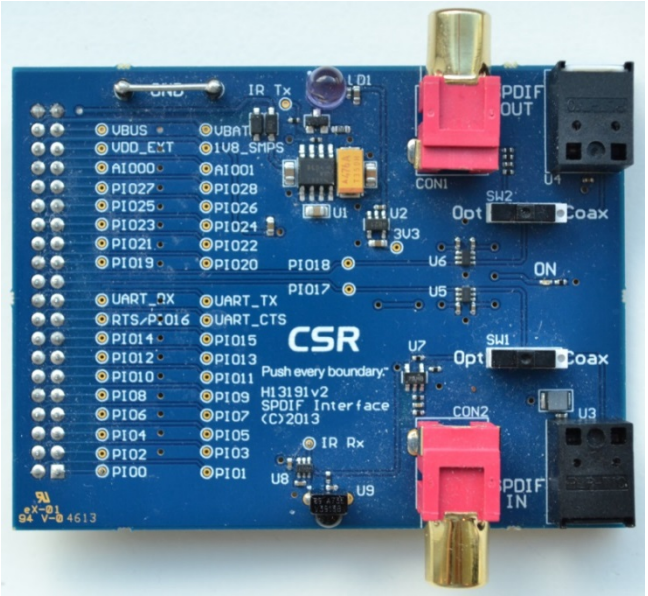


Figure 2-7 H13191v2 development board

The application does not have to support both SPDIF and infra-red to use this board, just one or the other.

2.3.6 CNS10010 display and FM receive

This plugin board provides an external display and an FM receive and transmit functionality. The FM modules are Si4704/05-D50 FM Radio Receiver and Si4710-B30-GM FM Radio Transmitter. The display is a TMGC16265I module providing 16x2 characters. All modules are controlled by the CSR8670 or CSR8675 by the I^2S interface. An SD Card slot is also provided.

See *Display and FM Expansion Board – Quick Start Guide* for information how to connect to the development board.

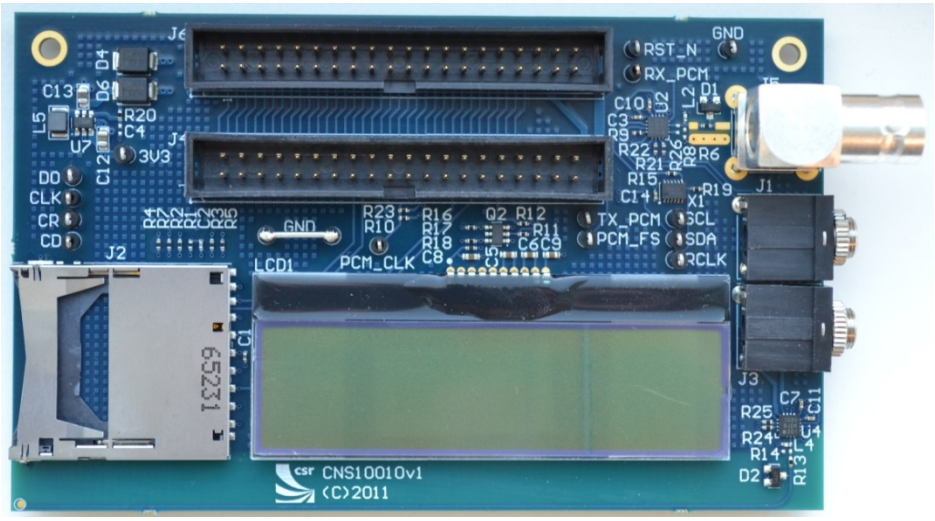


Figure 2-8 CNS10010v1 development board

NOTE The Sink application does not support the FM transmit or SD Card features of the CNS10010v1.
The application does not have to support FM receive and display to use this board, just one or the other.

3 Development tools

This section describes the common development tools supported by QTIL that help software development of the Audio Sink application.

xIDE

The QTIL xIDE software development environment can be used to:

- View
- Build
- Debug
- Run
- Reconfigure
- Modify

NOTE If unfamiliar with the xIDE environment, QTIL recommends reading the *xIDE User Guide*.

Universal Front End

The Universal Front End (UFE) tool is used to configure the DSP, there are many components of the UFE tool covered by various user guides available as part of the ADK documentation package.

PSTool

PSTool is used to read, set, or clear each of the PS Keys of a BlueCore device. See *Qualcomm® BlueLab™ PSTool User Guide* for further details.

BlueFlash

BlueFlash is a development tool providing an interface to manage the internal storage of the target hardware. It provides the following functions:

- Start/Stop target processor
- Erase Flash (erase full chip, erase stack, erase DFU/Loader, erase stack, erase persistent store, invalidate file system)
- Identify the firmware ID of target hardware
- Dump the entire contents of the target hardware to a file on the host machine
- Flash a pre-built image to the target hardware

Sink Configuration tool

The Sink Configuration tool can be used to configure the Audio Sink application, see the *ADK Configuration Tool User Guide* for further details.

4 System architecture

This section gives an overview of the system architecture adopted for the Audio Sink application.

Figure 4-1 outlines the key individual components and their internal layers, split into two main sections with the Application Sandbox sitting above Firmware and the hardware layer at the bottom layer.

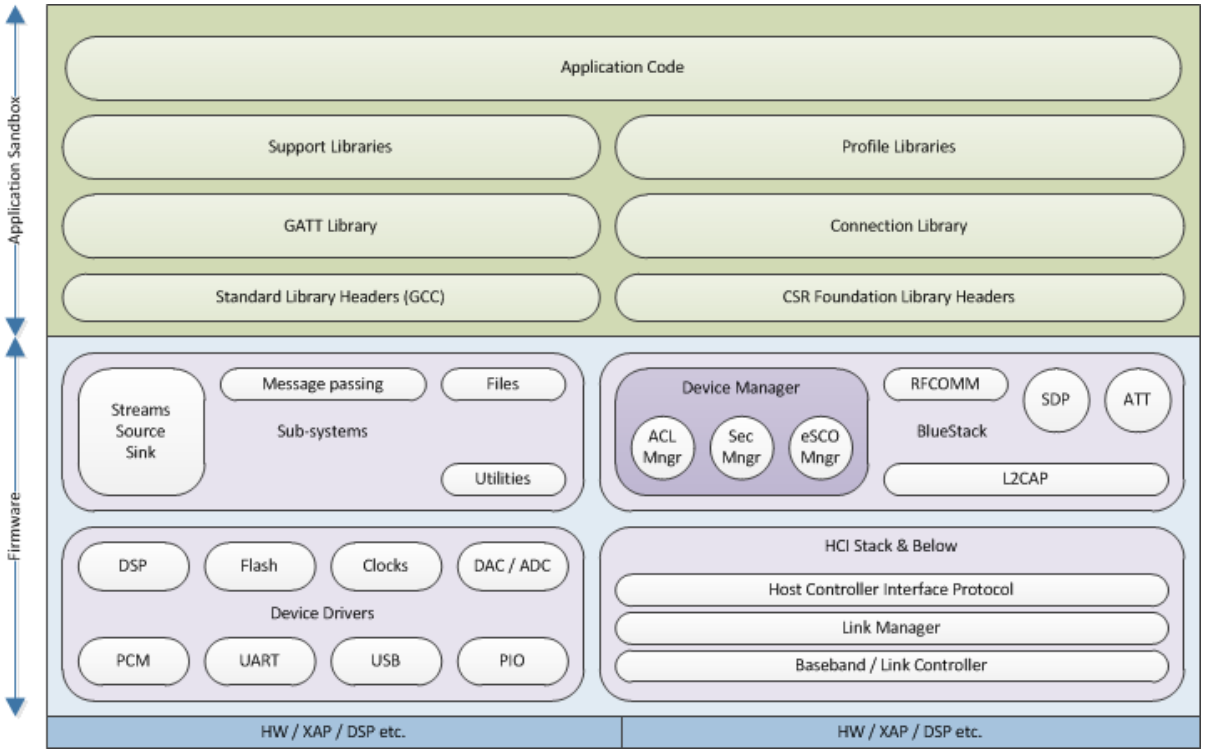


Figure 4-1 QTIL system architecture diagram

4.1 Application sandbox

This is native code protected by a hardware memory management unit providing a protected program space where applications can execute without affecting the BlueCore's Bluetooth radio performance. The sandbox is assigned a low priority, while the radio and Bluetooth Stack are given highest priority.

When the sandbox code is executed, triggered by a message or event from the Firmware, it gets 2 milliseconds before it is rescheduled (unless an interrupt occurs before then). However, calls into Firmware run to completion.

4.2 Connection library

The Connection library wraps around the Bluestack API to manage Bluetooth connections with remote devices. It provides a simple API allowing upper layers to create, update, and destroy connections with Bluetooth devices. The connection library supports Basic Rate, Enhanced Rate, and Bluetooth low energy technology connections.

4.3 Support libraries

Support libraries provide various functions that make a peripheral library easier to use e.g. the Battery library. They also provide simple helper functions e.g. for managing Bluetooth addresses using a data structure, as in the `bdaddr` library. Table 4-1 lists the support libraries used by the Audio Sink application.

Table 4-1 Support libraries used by the audio sink application

Library	Description
`audio`	Library to manage the audio interfaces
`audio_plugin_common`	Common functionality for audio interfaces
`audio_plugin_if`	Provides a list of messages the audio sub system can manage
`audio_output`	Audio plugin for multichannel audio output
`audio_ports`	Library to implement an external interface to audio library ports
`bdaddr`	Provides Bluetooth address functionality such as compare address.
`codec`	Provides functionality to configure the BlueCore DACs and ADCs
`config`	Provides interfaces to retrieve PS Key related information
`csr_a2dp_decoder_common_plugin`	Audio plugin used for A2DP audio connections.
`Csr_common_example_plugin`	Example CVC Plugin
`csr_cvc_common_plugin`	Audio plugin common to all Qualcomm® cVc™ algorithms
`csr_dut_audio_plugin`	Audio plugin used when Device Under Test mode is active
`csr_i2s_audio_plugin`	Audio plugin to control I^2S audio input/output
`csr_tone_plugin`	Audio plugin to control playing tones
`csr_voice_prompts_plugin`	Audio plugin to control playing voice prompts
`display`	Provides functionality to use the display
`display_example_plugin`	Example display plugin
`display_plugin_cns10010`	Display plugin specific to the example display hardware
`display_plugin_if`	Display plugin interface
`fm_plugin_if`	FM plugin interface
`fm_rx_api`	Provides FM Receiver functionality
`fm_rx_plugin`	FM Receiver Plugin

Table 4-1 Support libraries used by the audio sink application (cont.)

Library	Description
kalimba_standard_messages	Defines standard Qualcomm® Kalimba™ messages that can be sent between the DSP and VM
library	Provides unique message ID ranges for each library
md5	Implementation of the MD5 message-digest
pblock	Implements a persistence system allowing keyed blocks of data to be persisted to/from a PS Key.
Pio_common	Provides common support functionality for PIO interfaces
power	Provides support when using the charger and battery
print	Provides debug support for printing output messages
region	Processes regions of uint8 memory
sdp_parse	Provides support for parsing SDP records
service	Parse and search service records
upgrade	Library to manage software upgrade procedures
usb_device_class	Library to manage USB enumeration
vmtypes	Defines typedefs used by the application
wbs	Library to support Wide Band Speech implementations
broadcast	Library to manage Broadcast Audio
scm	Library to manage sub channel messages for Broadcast Audio

4.4 Profile libraries

Each Profile library implements the complex functionality of a single Bluetooth profile and manages connections for that profile with remote devices.

The on-chip profile libraries use the Connection Library for lower protocol interfacing, however bearing in mind the time restricted nature of the Sink application, short cuts bypassing the Connection library are not uncommon and indeed necessary.

The Profile libraries provide a simple API to the application layer allowing complex Bluetooth profiles to be used in a simple way. Table 4-2 lists the Profile libraries used by the Audio Sink application.

Table 4-2 Profile libraries used by the audio sink application

Library	Description
a2dp	Library to manage A2DP connections
Avrcp	Library to manage AVRCP connections
Connection	Library to manage low level Bluetooth connections, RFCOMM, L2CAP, Bluetooth low energy technology etc…
Gaia	Library to manage GAIA connections
Gatt	Library to manage GATT connections and the registered GATT database for the local device
gatt_manager	Library to manage GATT connections

Table 4-2 Profile libraries used by the audio sink application (cont.)

Library	Description
gatt_apple_notification_client	Library to manage the Apple Notification Center Client Service on a remote device
gatt_battery_client	Library to manage the Battery Service on a remote device
gatt_battery_server	Library to manage the Battery Service of the local device
gatt_device_info_client	Library to manage the Device Information Service on a remote device
gatt_device_info_server	Library to manage the Device Information Service of the local device
gatt_gap_server	Library to manage the Generic Attribute Profile Service of the local device
gatt_hid_client	Library to manage the HID Service on a remote device
gatt_imm_alert_client	Library to manage the Immediate Alert service on a remote device
gatt_imm_alert_server	Library to manage the Immediate Alert service of the local device
gatt_link_loss_server	Library to manage the Link Loss service of the local device
gatt_server	Library to manage the GATT Service of the local device
gatt_transmit_power_server	Library to manage the Transmit Power Service of the local device
gatt_heart_rate_client	Library to manage the Heart Rate Service on a remote device
gatt_heart_rate_server	Library to manage the Heart Rate Service of the local device
Hfp	Library to manage HFP connections
Hid	Library to manage HID connections
Mapc	Library to manage MAPC connections
Obex	Library to manage OBEX connections
obex_parse	Library to parse profile data for OBEX connections
pbap_common	Library providing generic PBAP support
pbapc	Library to manage PBAP connections
spp_common	Library to manage SPP connections
Swat	Library to manage SWAT connections

4.5 Audio sink application events

The Audio Sink application uses events for user interaction, are divided into two types:

- **Control events**
  Control Events allow the user to control the application, For example:
  - Power On
  - Power Off
  - Volume Up

- Volume Down
- Mute Toggle

Available control events are covered in relevant sub-sections of this document.

- **Indication events**
  Indication Events allow the application to indicate activity to the user. Typically when these events are generated, an audio tone or audio prompt is started, for indication including for example:
  - Low Battery
  - Remote Device Connected
  - Remote Device Disconnected
  - Maximum Volume Limit

Available indication events are covered in relevant sub-sections of this document.

4.6 Audio sink application states

The Audio Sink application has a central state machine controlling the main state of the application, Table 4-3 lists these states.

Table 4-3 Audio sink application states

State	Description
Limbo	The device is logically off but physically on
Connectable	The device is connectable - page scanning
Connectable Discoverable	The device is connectable and discoverable - page and inquiry scanning
Connected	The device is connected to an AG
Outgoing Call Establish	The connected AG has an outgoing call in progress
Incoming Call Establish	The connected AG has an incoming call in progress
Active Call SCO	The connected AG has an active call in progress and the audio is in the device
Active Call No SCO	The connected AG has an active call and the audio is in the handset
Three Way Call Waiting	The connected AG has an active call and a second incoming call
Three Way Call On Hold	The connected AG has an active call and a second call on hold
Three Way Conference Call	The connected AG has more than one active call
Incoming Call On Hold	The connected AG has an incoming call on hold
A2DP Streaming	The device is streaming A2DP audio
Device Under Test Mode	The device is in Test mode

5 Operational modes

The Audio Sink application supports three independent operational modes. These are essentially default sink applications (products) provided with ADK with different set of features.

- **Headset mode**
  Headset mode allows by default automatic switching of the audio sources based on a priority list. When an audio source becomes available for routing, its priority is checked against any currently routed source and becomes the routed source if it has a higher priority. Otherwise, the incoming audio is disposed of silently.
- **Soundbar mode**
  Soundbar mode allows by default manual switching of the audio sources. See for sink events for manual audio switching.
- **Subwoofer link**
  When the Audio Sink application has been configured to build as a soundbar, the Subwoofer Link can be supported. The Subwoofer Link is used to relay a low bandwidth audio signal from the soundbar to the subwoofer so the subwoofer can render an audio signal that only contains bass frequencies and has minimal latency with the soundbar.
- **Peer Device (speaker) mode**
  Peer Device mode is covered within the *ADK Sink Application Peer Device Mode User Guide*, see this document for further details.

NOTE Manual and Auto audio source selection are available for all the three modes. See for details.

6 Build configuration

The Audio Sink application supports a number of Bluetooth profiles and non-Bluetooth interfaces, see Generic configurable build options. Build configuration options have been provided to easily configure or disable each feature and interface.

Due to constraints it is not possible to support some build options simultaneously, for this reason, these options have been made explicitly specific to a particular operational mode.

This section lists all the available build options listing the specific build options that have been limited to a particular operational mode.

NOTE Many of the build options covered in this section are configured by the project files of example products. For more detail, see *Audio Sink Application Project Resources*.

6.1 Headset mode configurable build options

Table 6-1 lists the build options explicitly available when Headset mode is being used.

Table 6-1 Build options for Headset mode

Build Option	Description
Party mode	Enable support for Party mode
Selfie mode	Enable support for HID Selfie mode

When using the Headset project, Party mode and Selfie mode are disabled by default.

6.2 Soundbar mode configurable build options

Table 6-2 lists the build options explicitly available when Soundbar mode is being used.

Table 6-2 Build options for Soundbar mode

Build Option	Description
Subwoofer Link	Enable support for connections to a subwoofer device

When using the soundbar project, these features are enabled by default.

6.3 Peer Device (speaker) mode configurable build options

Table 6-3 lists the build options explicitly available when Peer Device mode is being used.

Table 6-3 Build options for Peer Device mode

Build Option	Description
Peer Device Support	Enable support for connections to peer devices
Broadcast Audio	Enable support for broadcasting audio to multiple receivers

When using the speaker project the Peer Device mode, TrueWireless is enabled by default.

6.4 Generic configurable build options

This section describes the generic configurable build options. Generic options are those not explicitly restricted to an operational mode.

6.4.1 Call features

Table 6-4 lists the available Call Features build options for the Audio Sink application.

Table 6-4 Build options for call features

Build Option	Description
Speech Recognition	Enable speech recognition
Use DSP for SCO	Configures the DSP to be used for SCO connections
cVc	Include or Exclude building cVc code
PBAP	Enable Phone Book Access Profile (PBAP) features
MAP	Enable Message Access Profile (MAP) features
Three Way Calling	Enable three way calling

6.4.2 Audio codecs

Table 6-5 lists the available Audio Codecs build options for the Audio Sink application.

Table 6-5 Build options for audio codecs

Build Option	Description
Wired Audio	Enable Wired Audio support
Display	Enable Display, e.g. CNS10010v1 add on board
Extra Codec	Enable support for the A2DP codecs MP3, AAC and Qualcomm® aptX™
aptX Sprint	Enable support for the A2DP codec aptX Low Latency variant
FastStream	Enable support for the A2DP FastStream variant
aptX-HD	Enable support for the aptX-HD variant

6.4.3 Bluetooth low energy technology

Table 6-6 lists the available Bluetooth low energy technology build options for the Audio Sink application.

Table 6-6 Build options for Bluetooth low energy technology

Build Option	Description
Enable Bluetooth low energy technology/GATT	Enable Bluetooth low energy technology/GATT support in the application. GATT is only used with a Bluetooth low energy link
BLE when powered off	Specifies the Bluetooth Low energy technology behavior when the application is powered off
Enable GATT ANCS client	If GATT support is enabled, this option will also enable the GATT ANCS Client functionality
Enable GATT Find Me Client	If GATT support is enabled, this option will also enable the GATT Find Me Client functionality
Enable GATT HID Client	If GATT support is enabled, this option also enables the GATT HID Client functionality
Enable GATT Battery Server	If GATT support is enabled, this option also enables the GATT Battery Server functionality
Enable GATT Find Me Server	If GATT support is enabled, this option will also enable the GATT Find Me Server functionality
Enable GATT GAIA Server	If GATT support and GAIA features are enabled, this option also enables the GATT GAIA Server functionality
Enable GATT Proximity Server	If GATT support is enabled, this option also enables the GATT Link Loss and Transmit Power Server functionality
Enable Heart Rate Client	If GATT support is enabled, this option also enables the GATT Heart Rate Client functionality
Enable Heart Rate Server	If GATT support is enabled, this option also enables the GATT Heart Rate Server functionality

6.4.4 AVRCP

Table 6-7 lists the available AVRCP build options for the Audio Sink application.

Table 6-7 Build options for AVRCP

Build Option	Description
AVRCP	Enable Audio/Video Remote Control Profile (AVRCP)
AVRCP Now Playing	Enable Now Playing functionality of AVRCP 1.5
AVRCP Player App	Enable Player Application Settings functionality of AVRCP 1.5
AVRCP Browsing	Enable Browsing functionality of AVRCP 1.5

6.4.5 USB

Table 6-8 lists the available USB build options for the Audio Sink application.

Table 6-8 Build options for USB

Build Option	Description
USB	Enable USB functionality
USB Audio	Enable USB audio functionality
USB MS Readme	`Headset.mak` directive that copies the example USB mass storage files into the filesystem

6.4.6 GAIA

Table 6-9 lists the available GAIA build options for the Audio Sink application.

Table 6-9 Build options for GAIA

Build Option	Description
GAIA	Enable Generic Application Interface Architecture (GAIA)
GAIA SPP	Use SPP instead of RFCOMM for the GAIA service
GAIA Persistent EQ	Store User Defined EQ Bank 1 Settings in Persistent Storage for the GAIA service. Compiler flags for GAIA service must be enabled alongside this flag for this feature to be enabled

6.4.7 FM

Table 6-10 lists the available FM build options for the Audio Sink application.

Table 6-10 Build options for FM

Build Option	Description
FM Audio	Enable FM Audio support
FM Audio RDS	Enable FM Audio RDS support

6.4.8 Other

Table 6-11 lists the available build options for the Audio Sink application that do not fall into any other logical grouping.

Table 6-11 Other build options

Build Option	Description
Battery Operation	Enable support for the Battery operation…..
Config Type	Configures where the static configuration settings are stored: ■ File system (default): Files containing static configuration are stored within the on-chip file system. In this mode more PS Keys are free for other use. PS Keys are still used for configuration updated by the application. ■ PS Keys: The configuration is stored entirely in PS Keys. ■ File system and PS Keys: Both methods of configuration storage are enabled. A configuration in PS takes priority over that stored in the File system.
Device ID PS Key	Enable Device ID Profile (DIP). Allows for PS configurable Device ID information, from PSKEY_DEVICE_ID, but requires permanent use of a memory pool to hold the SDP record.
Infrared Remote Support	Enable support for Infrared remote controllers. CSR8670 only.
CapacitiveSensor Buttons	Enable Capacitive touch sensors. CSR8670 only.
Device Upgrade	Enable support for device upgrade.
Hi Res Audio	Enable 24-bit and Hi Res Implementation (Interface resolution of 24-bit and a rate of 96 Khz/88.2 Khz). CSR8675 only.
BR/EDR Secure Connection	Enable support for BR/EDR Secure Connection. CSR8675 only.

7 Define symbols

Define symbols are used to conditionally compile certain features of the code, this section details the common define symbols used by the Audio Sink application.

7.1 Debug

Adding the `DEBUG_PRINT_ENABLED` define symbol enables debug output when running the Audio Sink application. Debug for particular features can be enabled and disabled by editing the relevant define symbols in the `sink_debug.h` file or by adding the particular define symbol.

NOTE When debug is enabled in the application or libraries the application does not run without a debugger attached.

Table 7-1 lists the debug defines available.

Table 7-1 Debug defines for the Audio Sink application

Symbol	Description
DEBUG_MAIN	The main system messages
DEBUG_INQ	RSSI Pairing
DEBUG_BUT_MAN	Button Manager
DEBUG_BUTTONS	Low level button parsing
DEBUG_CALL_MAN	The Call Manager
DEBUG_MULTI_MAN	The Multipoint Manager
DEBUG_AUDIO	Audio connections
DEBUG_SLC	SLC connections
DEBUG_DEV	Device Manager
DEBUG_LP	Link Policy
DEBUG_CONFIG	Config Manager
DEBUG_LM	LED Manager
DEBUG_LEDS	Lower level LED driver
DEBUG_PIO	Lower Level PIO driver
DEBUG_POWER	The Power Manager
DEBUG_TONES	Tones
DEBUG_VOLUME	Volume
DEBUG_STATES	State Manager

Table 7-1 Debug defines for the Audio Sink application (cont.)

Symbol	Description
DEBUG_AUTH	Authentication
DEBUG_DIM	Dimmable LEDs
DEBUG_A2DP	A2DP Debug
DEBUG_PEER	Peer Device
DEBUG_PEER_SM	Peer Device Security
DEBUG_INIT	Initialization
DEBUG_AVRCP	AVRCP
DEBUG_AUDIO_PROMPTS	Voice prompts
DEBUG_USB	USB
DEBUG_MALLOC	Memory Management
DEBUG_PBAP	PBAP
DEBUG_MAPC	MAPC
DEBUG_GAIA	GAIA
DEBUG_SPEECH_REC	Speech Recognition
DEBUG_WIRED	Wired audio
DEBUG_AT_COMMANDS	Custom AT Commands
DEBUG_SC	BR/EDR Secure connection
DEBUG_BLE_SC	Bluetooth low energy technology Secure connection
DEBUG_GATT	GATT
DEBUG_GATT_MANAGER	GATT Manager
DEBUG_BLE	Bluetooth low energy technology transport and messages
DEBUG_GATT_HID_CLIENT	HID Over GATT
DEBUG_HID	BR/EDR HID
DEBUG_DUT	Device Under Test
DEBUG_DI	Device ID
DEBUG_DISPLAY	Display
DEBUG_SWAT	Subwoofer
DEBUG_FM	FM Receive
DEBUG_INPUT_MANAGER	Input Manager
DEBUG_IR_RC	Infra-Red Remote Controller
DEBUG_BAT_REP	Battery Reporting
DEBUG_GATT_HRS_CLIENT	Heart rate client
DEBUG_GATT_HRS_SERVER	Heart rate server
DEBUG_BA_COMMON	Broadcast Audio
DEBUG_BA_PLUGIN	Broadcast Audio plugin

Table 7-1 Debug defines for the Audio Sink application (cont.)

Symbol	Description
DEBUG_BA_BROADCASTER	Broadcaster mode
DEBUG_BA_RECEIVER	Receiver mode

8 Device configuration

Before use, the target hardware must be configured with a valid configuration. This section describes the configuration files required to run the Audio Sink application on a target device.

8.1 Configuration files

The default configuration files for the Audio Sink application can be found in the ADK install directory:

<ADK>\apps\sink\configurations

To configure a system, two configuration files are required.

- **System configuration**
  The target hardware dictates the system configuration that should be used. Table 8-1 describes the system configuration files available for the Audio Sink application.

  Table 8-1 System configuration files

Configuration File	Description
`sink_system_csr8675.psr`	To be used with the CSR8675 hardware platform
`sink_system_csr8670.psr`	To be used with the CSR8670 hardware platform

- **Application configuration**
  The Audio Sink application is capable of being configured in many different and unique ways to support the specific requirements of an end product. For this reason, QTIL have specified a number of example end products for which configurations have been supplied. For details on each product and its configuration, see the *Audio Sink Application Project Resources*.

  NOTE It does not matter which order the configuration files are applied when merging the configuration files to the device.

9 Audio Sources

The Audio Sink application supports a variety of audio sources:

- Wired audio
- USB audio
- FM receive
- Bluetooth SCO
- Bluetooth A2DP
- Speech recognition
- SPDIF input

NOTE Selection of 16-bit/24-bit resolutions for input and output can be configured using the Sink Configuration tool.

9.1 Wired audio

Wired Audio input provides a wired input source, Table 9-1 lists the wired audio sources supported by the Audio Sink application.

Table 9-1 Wired audio sources

Input	Description
PCM Audio	The Audio Sink application is capable of accepting a raw PCM data stream for audio input
I^2S Audio	Various I^2S data input rates can be supported by the Audio Sink application as well as supporting master or slave I^2S operation

NOTE (1) When using the CSR8670 platform, PCM, and I^2S audio sources are not supported in the same project.

(2) For PCM, 24-bit supported with 16-bit audio quality. See Section 10.3 for limitations.

(3) For I2S, 24-bit and Hi Res supported.

9.2 USB audio

When connected by USB to a host machine that supports USB audio output the Audio Sink application is capable of rendering the audio source, see Section USB for more details.

9.3 FM receive

The Audio Sink application can support an FM receiver, to use FM audio the FM receiver hardware is required, see . Audio from the FM receiver can be used as the input source and routed to the active output audio interface, see .

9.4 Bluetooth SCO

Table 9-2 lists the Bluetooth SCO audio supported by theAudio Sink application.

Table 9-2 SCO audio sources

Input	Description
Bluetooth SCO (Call Audio)	SCO used to transfer HFP call audio to the Audio Sink application
Bluetooth SCO (Audio)	SCO used to transfer audio such as tones and beeps (not commonly used)

SCO call audio

SCO and ESCO packet types are supported.

SCO audio

SCO and ESCO packet types are supported.

9.5 Bluetooth A2DP

Table 9-3 lists the audio from the A2DP audio codecs that the Audio Sink application is capable of.

NOTE The Configuration tool can be used to map a specific LED pattern to identify to the user, which codec is in use.

Table 9-3 Supported A2DP codecs

Codec	Description
SBC	Low Complexity Subband Coding
MP3	MPEG-1 or MPEG-2 Audio Layer III
AAC	Advanced Audio Coding
aptX	QTIL audio codec
aptX Sprint	QTIL low latency audio codec
aptX-HD	QTIL audio codec with HD support

9.6 Speech recognition

The Audio Sink application supports a Simple Speech Recognition algorithm that allows a user to control behavior, see .

9.7 SPDIF input

This section describes the SPDIF audio source support provided by the Audio Sink application. H13191 SPDIF interface and infra-red receive shows the hardware required to support the SPDIF audio source.

NOTE (1) SPDIF Input is only available in Soundbar mode.

(2) 24-bit and Hi Res supported.

IEC 61937

By default, the Audio Sink application supports compressed digital audio over SPDIF as defined by the IEC61937 standard and supports the standard SPDIF sample rates:

- 48 kHz
- 44.1 kHz
- 32 kHz

The incoming data contains a sample rate that is automatically detected. Any change in sample rate is smooth and glitch-free.

Hardware requirements

QTIL provide hardware reference designs for coaxial (RCA Connectors) and optical (TOSLINK) SPDIF connections, contact QTIL support for further details if this hardware is required.

H13191 SPDIF interface and infra-red receive shows the hardware that is required to support a SPDIF audio source.

10 Audio outputs

The Audio Sink application supports the audio outputs: described in: Sections Wired audio and SPDIF output.

10.1 Wired audio

PCM Output

Audio streams decoded by the Audio Sink application can be decoded to a PCM audio stream and where required, this stream is connected to the BlueCore DAC and outputted as analog audio.

NOTE 24-bit supported with 16-bit audio quality. See 24-bit and Hi-res (88.2/96 kHz) support for limitations.

I^2S Output

Audio streams decoded by the Audio Sink application can be decoded to an I^2S audio stream and where required, this stream is connected to an external I^2S digital amplifier.

For more information about the I^2S audio interface, read *ADK I^2S User Guide.*

NOTE (1) When the subwoofer link is enabled in the Audio Sink application, the I^2S output rate is always set to 48 kHz. This is true regardless of what the user specifies as the output rate in the config tool, and it is also true regardless of whether the subwoofer connects during run-time.

(2) I^2S output peripherals need to be slave to the BlueCore if a wireless subwoofer is going to be attached.

(3) 24-bit and Hi Res supported

10.2 SPDIF output

By default, the Sink application supports SPDIF output; supported audio data format when using SPDIF output is stereo PCM, the following sample rates are supported:

- 48 kHz
- 44.1 kHz
- 32 kHz

Section H13191 SPDIF interface and infra-red receive lists the hardware required for SPDIF Audio Output.

NOTE (1) The IEC61937 standard is not supported for SPDIF Output.

(2) 24-bit supported but Hi Res (96 Khz/88.2 Khz) not supported.

11 24-bit and Hi-res (88.2/96 kHz) support

Support and limitations

- 24-bit operation is supported on the CSR8675 chip only (not on CSR8670).
- Hi-res (88.2/96 kHz) sampling rates are supported on the CSR8675 chip only (not on CSR8670).
- Internal ADCs and DACs do not support sampling rates @88.2 kHz.
- 24-bit operation delivers a processed audio quality of ~20-bits to 22-bits for wired modes and digital interfaces. Desired to achieve 120 dB SNR (20-bit linear equivalent).
- The tone/prompt input does not support 24-bit PCM audio.
- 24-bit operation for multi-channel (output) configurations is supported subject to MIPs and BW restrictions.
- 24-bit operation is not supported for USB input/output.
- 16-bit SPDIF output @88.2/96 kHz is supported.
- 24-bit SPDIF output @88.2/96 kHz is not supported.
- The SPDIF output may become inactive as DSP apps are loaded/unloaded.
- Configurations with I^2S inputs and I^2S outputs at different rates (or modes) can only be achieved by using separate physical interfaces. In these configurations multi-channel operation is limited since only a single output I^2S interface (2 audio channels) is available.SWAT, TWS and SHAREME features are not supported for Hi-rate (88.2/96 kHz) rates.
- If a resampling rate is explicitly set for the I^2S output it overrides the set output sampling rate set for wired sources.
- When subwoofer functionality is enabled all the output playback is at a fixed rate of 48 Khz there by overriding the set output sampling rate for wired sources.
- When streaming audio over Bluetooth A2DP, 24-bit output compatibility is limited to the negotiated codec, that is, only APTX-HD provides support for 24-bit capability.

Table 11-1 Wired audio support for 24-bit/ Hi-res

Wired audio							
Bit resolution	Sampling rate	Input			Output		
		ADC	SPDIF	I^2S	DAC	SPDIF	I^2S
16-bit	44.1	Yes	Yes	Yes	Yes	Yes	Yes
	48	Yes	Yes	Yes	Yes	Yes	Yes
	88.2	No	Yes	Yes	No	Yes	Yes

Table 11-1 Wired audio support for 24-bit/ Hi-res (cont.)

Wired audio							
		Yes	Yes	Yes	Yes	Yes	Yes
24-bit	44.1	Yes	Yes	Yes	Yes	Yes	Yes
	48	Yes	Yes	Yes	Yes	Yes	Yes
	88.2	No	Yes	Yes	No	No	Yes
	96	Yes	Yes	Yes	Yes	No	Yes

The audio performance when streaming audio in 24-bit resolution mode may be limited by the codec negotiated.

Native processing at the 88.2/96 kHz rate may not be possible for some processing configurations due to MIPs limitations. Under these circumstances it may be necessary to either:

1. Use a lower processing/output sampling rate
   or
2. Disable non-essential DSP functions to reduce MIPs.

11.1 Multi-channel output

The Audio Sink application can split an incoming stereo audio source by frequency bands and have multiple audio outputs that can be sent to supported output sources.

There is full support for multi-channel output on CSR8675, which in addition to the codec output has two I^2S outputs which can be used simultaneously with SPDIF input.

There is limited support for multi-channel output on CSR8670, which in addition to the codec output has only a single I^2S output which cannot be used simultaneously with SPDIF input.

11.1.1 Routing

The DSP provides several channels of output, which can be routed to either the CODEC or I^2S outputs:

- Incoming stereo audio is split across the primary and secondary DSP outputs. To ensure proper synchronization of audio the primary and secondary outputs should always be connected the same hardware, so if the primary audio is connected to **I2S1** the secondary audio should be connected to **I2S2**.
- The Aux DSP output is intended for use as a headphones output and may be connected to any hardware output.
- The Wired Subwoofer output provides a local subwoofer output channel and may be connected to any hardware output.

11.1.2 Hardware setup

Multi-channel output uses the multi-channel amplifier described in H13596 multichannel amplifier.

11.1.3 Build options

Multi-channel support is built in to all projects by default. Build options need only be modified if SPDIF input is to be used, in which case the **spdif_sink** project must be built using the **Multi_Channel_Soundbar** configuration.

11.1.4 Configuration

This section describes how to configure the target hardware for multi-channel output.

Multi-channel VM Configuration Files

Table 11-2 describes configuration files included with the ADK that configure the Audio Sink application for multi-channel output.

Table 11-2 Multi-channel VM configuration files

PSR file	Description
`multi_channel_i2s_H13596v1.psr`	Configures the primary DSP output to the **I2S1** and secondary DSP output to **I2S2**. The Aux DSP output is routed to the **CODEC** and **CODEC HEADPHONES** outputs. The hardware gain for I^2S amplifiers is set to 0dB in the I^2S start-up commands and digital gain is used to adjust volume at run-time. To use the **I2S HEADPHONES** outputs or change the hardware gain for the I^2S amplifiers, the I^2S start-up commands must be modified.
`multi_channel_i2s_H13596v1_hw_gain.psr`	Configures the DSP outputs as described for `multi_channel_i2s_H13596v1.psr`, additionally it configures hardware control of the I^2S amplifiers for run-time volume changes.

Multi-channel DSP configuration files

Table 11-3 describes the configuration files included with the ADK that configure the DSP for multi-channel output.

Table 11-3 Multi-channel DSP configuration files

PSR File	Description
`crossover_two_way_stereo.psr`	Configures default values for splitting of frequencies between primary and secondary DSP outputs. These can be modified using the Universal Front End tool.

Device configuration

After basic configuration of the target hardware as described in Device configuration has been completed, the multi-channel configuration can be applied by following the procedure outlined below:

1. Using PSTool, see *Bluelab PSTool User Guide*:
   a. One of the VM configuration files described in Table 11-2 must be merged to the target hardware.
   b. One of the DSP configuration files described in Table 11-3 must be merged to the target hardware. Some manual configuration must be done using the Sink Configuration Tool.
2. Using the Audio Sink Application Configuration Tool, see *Audio Sink Application Configuration Tool User Guide*:
   a. The **Force Re-Sampling of Tones** audio plugin feature should be enabled. This routes tone playback through the DSP so all tones are played on the Primary and Aux outputs.
   b. The **Amplifier Power Down by PIO** audio plugin feature should be enabled if using the soundbar project.
   c. **SPDIF Output PIO** must be set to **N/A** or the **I2S1** output does not function.

NOTE (1) CVC does not support multi-channel output, call audio is only be heard on the primary outputs and tones only play on the primary outputs when cVc is in use.

(2) QTIL recommends that no more than four local DSP outputs and one wireless DSP output (that is, Peer Device mode, see or Subwoofer Link, see) be enabled if configuration allows four or more Audio Sources, see , to be connected and Remote Software Upgrade, see , is enabled.

11.2 Audio routing

Manual and auto audio source selection are available for all three application variants, that is, headset, soundbar, and peer device mode.

The user can configure the audio source availability (by enabling or disabling an audio source) and the priorities associated with it.

The ADK provides events for audio source connect and disconnect. These can be mapped to LED patterns using the Configuration tool to indicate to the user when a specific events occurs

Table 11-4 List of audio sources available

Audio source selection
AG1
AG2
USB Audio
Analog (Wired)
FM Receive
SPDIF Audio
I2S
No Active Source

11.2.1 Auto mode

Using the Sink Configuration tool the user can customize the list of available audio sources (enable/disable) and also prioritize the enabled audio sources based on user's preference.

Sources are routed as soon as they become available and when their priority is higher than the currently routed source.

11.2.2 Manual mode

In the Manual mode user must explicitly choose an audio source to route. A particular audio source can be selected at any time by generating the particular Sink application events. The priority based audio switching is available for events `EventUsrSelectAudioSourceNext` and `EventUsrSelectAudioSourceNextRoutable`.

Table 11-5 describes the events used to control audio routing when Manual mode is active.

Table 11-5 Sink application events to control audio routing in Manual mode

Event	Description
`EventUsrSelectAudioSourceNext`	Select the next audio source (follows the priority list for cycling through sources). The switch will take place regardless of next audio source is routable or not.
`EventUsrSelectAudioSourceAnalog`	Select Wired Audio source (PCM or I^2S)
`EventUsrSelectAudioSourceUSB`	Select USB Audio source
`EventUsrSelectAudioSourceAG1`	Select AG1 Bluetooth A2DP Audio source
`EventUsrSelectAudioSourceAG2`	Select AG2 Bluetooth A2DP Audio source
`EventUsrSelectAudioSourceFM`	Select FM Receive Audio source
`EventUsrSelectAudioSourceSpdif`	Select SPDIF Audio source
`EventUsrSelectAudioSourceI2S`	Select I^2S Audio source
`EventUsrSelectAudioSourceNextRoutable`	Select next routable Audio source (follows the priority list for cycling through sources).
`EventUsrSelectAudioSourceNone`	Do not route any audio

11.3 Volume control

The Audio Sink application supports a variety of methods of volume control to control the output level of the active audio source:

NOTE Volume control is not specific to particular operational mode

- **DAC volume control**
  This legacy mode of volume control is available when using the BlueCore internal DACs and is supported across voice and music applications. The on-chip DACs are used to control the

amplitude of the audio output signal using a range of 16 steps with a minimum volume level of -45 dB up to a maximum volume level of 0 dB.

- **Digital volume control**
  This mode of volume control is only available to DSP music applications.
  The digital volume control works by setting the chip DAC to a fixed level and applying digital attenuation to the audio signal within the DSP itself.
  This method of volume control provides a much higher granularity of the volume changes as well as an increased range and an overall lower output level than that achieved using DAC control only.
  The digital attenuation gives a usable range of 0 dB maximum down to -80 dB minimum output level.

- **Hybrid volume control**
  For the best signal to noise ratio volume control is obtained using the Hybrid Volume Control mechanism, this is a combination of DAC and DSP volume control.
  The configuration is the same as for the digital volume control, except for setting the volume control type to hybrid.
  Hybrid volume control uses the DAC control to give steps of 3 dB in combination with the digital DSP control to give increased resolution of 0 to 3 dB.
  As with the digital volume control the resolution can be whatever is required for the product but attention must be given to the step size to smooth operation with USB and AVRCP volume level changes.

11.4 Audio enhancements

The Audio Sink application supports several audio enhancements designed to allow the user to fine tune the audio output to their own preference.

Bass-Boost/Bass-Plus

The Audio Sink application supports Bass-Boost/Bass-Plus enhancement designed to boost the lower frequencies of the audio output signal. The Bass-Boost/Bass-Plus enhancement can be controlled by the user events listed in Table 11-6.

Table 11-6 User events controlling Bass-Boost/Bass-Plus

User Event	Description
`EventUsrBassEnhanceOn`	Enables the Bass-Boost/Bass-Plus audio enhancement
`EventUsrBassEnhanceOff`	Disables the Bass-Boost/Bass-Plus audio enhancement
`EventUsrBassEnhanceEnableDisableToggle`	Toggle between enable/disable Bass-Boost/ Bass-Plus audio enhancement

NOTE The parameters used by the Bass-Boost/Bass-Plus enhancement can be configured using the UFE tool, see the *Qualcomm BlueCore Music Manager User Guide (E00F)* for details on tuning the Bass Boost enhancement.

3D/3DV enhancement

The Audio Sink application supports 3D/3DV enhancement. This is an enhanced shuffler stereo widening algorithm with an independent stereo phase shifter.

It is designed to widen the stereo output providing the perception of surround sound. 3D/3DV enhancement can be controlled by the user events listed in Table 11-7.

Table 11-7 User events controlling 3D/3DV enhancement

User Event	Description
`EventUsr3DEnhancementOn`	Enables 3D/3DV audio enhancement
`EventUsr3DEnhancementOff`	Disables 3D/3DV audio enhancement
`EventUsr3DEnhancementEnableDisableToggle`	Toggle between enable/disable 3D/3DV audio enhancement

NOTE The parameters used by the 3D/3DV Enhancement can be configured using the UFE tool, see the *Music Manager User Guide* for details on tuning the 3D enhancement.

When enabled/disabled, these audio enhancements can be indicated to the user byu an LED pattern.

12 Bluetooth

The Audio Sink application supports the following Bluetooth transports:

- Bluetooth BR/EDR
- Bluetooth low energy technology
- Dual mode

12.1 Bluetooth BR/EDR

Bluetooth BR/EDR support in the Audio Sink application includes:

- Headset profile
- Handsfree profile
- Advanced Audio Distribution Profile
- Audio/Video Remote Control Profile
- Message Access Profile
- Phone Book Access Profile
- Human Interface Device Profile

12.1.1 Headset profile

The Audio Sink application supports the Bluetooth *Headset Profile v1.2*.

The Audio Sink application supports the HSP Headset role (HS) acting as the Audio Gateway's remote audio input and output mechanism:

- CVSD support for audio connections over SCO is provided by the firmware.
- HSP Remote Volume Control is supported by the Audio Sink application.

12.1.2 Handsfree profile

The Audio Sink application supports the Bluetooth *Handsfree Profile v1.7*.

The Audio Sink application supports the HFP Headset role (HS) acting as the Audio Gateway's remote audio input and output mechanism and:

- CVSD support for audio connections over SCO is provided by the firmware.
- The DSP can be used to suppress background noise and for echo cancellation, various cVc algorithms are available and supported by the Audio Sink application, if these are required, contact QTIL for more information.
- Three Way Calling is supported by the Audio Sink application, configured using the build options described in Headset mode configurable build options.

Call Manager

The Call Manager provides the user with control of incoming and outgoing voice calls it also manages three way calling (if enabled). Table 12-1 lists the events available to manage voice calls.

Table 12-1 User events controlling call handling

User Event	Description
`EventUsrInitateVoiceDial`	Starts HFP Voice Dial procedure on the primary AG
`EventUsrInitateVoiceDial_AG2`	Starts HFP Voice Dial procedure on the secondary AG if MultiPoint has been enabled
`EventUsrLastNumberRedial`	Starts HFP Last Number Redial procedure on the primary AG
`EventUsrLastNumberRedial_AG2`	Starts HFP Last Number Redial procedure on the secondary AG if MultiPoint has been enabled
`EventUsrAnswer`	Answer an incoming call
`EventUsrReject`	Reject an incoming call
`EventUsrCancelEnd`	Terminate an active call
`EventUsrTransferToggle`	Transfer call audio to or from the Audio Sink Application device depending on whether call audio is at the AG or not
`EventUsrThreeWayReleaseAllHeld`	Terminate all held calls
`EventUsrThreeWayAcceptWaitingReleaseActive`	Accept the incoming call and terminate the active call
`EventUsrThreeWayAcceptWaitingHoldActive`	Accept the incoming call and hold the active call
`EventUsrThreeWayAddHeldTo3Way`	Add the held call to the active call to create a conference call
`EventUsrThreeWayConnect2Disconnect`	When a conference call is active, this event causes the primary user to disconnect from the call and connect the second and third parties together.
`EventUsrPlaceIncomingCallOnHold`	Place the incoming call on hold before answering
`EventUsrAcceptHeldIncomingCall`	Accept the held incoming call (make the held incoming call active)

Table 12-1 User events controlling call handling (cont.)

User Event	Description
`EventUsrRejectHeldIncomingCall`	Reject the held incoming call
`EventUsrDialStoredNumber`	Dial the stored number
`EventUsrUpdateStoredNumber`	Update the stored number

Custom AT commands

AT commands are used as part of the HFP specification to allow an HF device to exchange information with an AG device.

Custom AT commands can be used to transfer additional information between devices in an active HFP connection. The Audio Sink application supports sending and receiving of custom AT commands to and from a connected AG device.

For more information on custom AT commands in the Audio Sink Application, see the *Custom AT Commands in the Audio Sink Application Note*.

12.1.3 Advanced Audio Distribution Profile

The Audio Sink application supports the Bluetooth *Advanced Audio Distribution Profile, v1.3.1*.

A2DP roles

The Audio Sink application is capable of supporting both A2DP sink and A2DP source roles:

- When the A2DP feature is enabled, the Audio Sink application will always support the A2DP Sink role.
- When Peer Device Mode is being used, the Audio Sink application also supports the A2DP Source role. It operates as an A2DP Source to be able to relay audio from the active audio source. See the *Audio Sink Application Peer Device User Guide* for more information about Peer Device mode.

A2DP codecs

The Audio Sink application supports the following A2DP codecs:

- SBC
- MP3
- AAC
- aptX
- aptX Low Latency
- FastStream
- aptX-HD

The list of supported A2DP codecs is configurable, refer to the *Audio Sink Application Configuration User Guide* for information on how to configure the list of supported A2DP codecs.

12.1.4 Audio/Video Remote Control Profile

The Audio Sink application supports the Bluetooth *Audio/Video Remote Control Profile v1.6*.

By default Audio Sink application supports Category-1 Controller and Category-2 Target. However if there is support for Peer device, then it also supports Category-2 Controller and Category-1 Target.

Table 12-2 User events for AVRCP controller

User Event	Description
`EventUsrAvrcpPlayPause`	Sends either `AVRCP_PLAY/AVRCP_PAUSE` based on play status
`EventUsrAvrcpStop`	Sends `AVRCP_STOP` to the current active AVRCP connection
`EventUsrAvrcpSkipForward`	Sends `AVRCP_FORWARD`
`EventUsrAvrcpSkipBackward`	Sends `AVRCP_BACKWARD`
`EventUsrAvrcpFastForwardPress`	Sends `AVRCP_FAST_FORWARD` press `PASSTHROUGH` command
`EventUsrAvrcpFastForwardRelease`	Sends `AVRCP_FAST_FORWARD` release `PASSTHROUGH` command
`EventUsrAvrcpRewindPress`	Sends `AVRCP_REWIND` press `PASSTHROUGH` command
`EventUsrAvrcpRewindRelease`	Sends `AVRCP_REWIND` release `PASSTHROUGH` command
`EventUsrAvrcpShuffleOff`	Sends the command to the Target player application to disabled shuffle of tracks
`EventUsrAvrcpShuffleAllTrack`	Sends the command to the Target player application to shuffle all tracks
`EventUsrAvrcpShuffleGroup`	Sends the command to the Target player application to shuffle tracks among groups
`EventUsrAvrcpRepeatOff`	Sends the command to the Target player application to disabled repeat of tracks
`EventUsrAvrcpRepeatSingleTrack`	Sends the command to the Target player application to repeat single track
`EventUsrAvrcpRepeatAllTrack`	Sends the command to the Target player application to repeat all the tracks
`EventUsrAvrcpRepeatGroup`	Sends the command to the Target player application to repeat tracks in the group
`EventUsrAvrcpPlay`	Sends `AVRCP_PLAY`
`EventUsrAvrcpPause`	Sends `AVRCP_PAUSE`
`EventUsrAvrcpToggleActive`	Used to toggle between two AVRCP connections
`EventUsrAvrcpNextGroupPress`	Sends the basic group navigation, next group press `PASSTHROUGH` command
`EventUsrAvrcpPreviousGroupPress`	Sends the basic group navigation, previous group press `PASSTHROUGH` command

Table 12-2 User events for AVRCP controller (cont.)

User Event	Description
`EventUsrAvrcpNextGroupRelease`	Sends the basic group navigation, next group release `PASSTHROUGH` command
`EventUsrAvrcpPreviousGroupRelease`	Sends the basic group navigation, previous group release `PASSTHROUGH` command

12.1.5 Message Access Profile

The Audio Sink application supports the *Bluetooth Message Access Profile v1.1.*

System Events

Table 12-3 System events used by the MAP feature

System Event	Description
`EventSysMapcMnsSuccess`	Indicates a successful MAP connection (MAP is now connected)
`EventSysMapcMnsFailed`	Indicates a failed MAP connection attempt
`EventSysMapcMsgNotification`	A new message has been received at the AG

Connection

When an AG supporting MAP is connected, the Audio Sink application tries to connect MAP events, this can be used to indicate whether the MAP connection was successful or failed, see Table 12-3.

Notifications

When MAP is connected, the Audio Sink application can register for notifications of personal messages on the connected AG. When a new personal message arrives on the AG, an event is generated to indicate that a new message has been received at the AG see Table 12-3.

12.1.6 Phone Book Access Profile

The Audio Sink application supports the Bluetooth *Phone Book Access Profile v1.1.1*.

User events

Table 12-4 User events used by the PBAP feature

User Event	Description
`EventUsrPbapDialIch`	Dial most recent entry from incoming call list
`EventUsrPbapDialMch`	Dial most recent entry from missed call list
`EventUsrPbapSetPhonebook`	Change to a different phonebook in the same repository
`EventUsrPbapBrowseEntry`	Download the first vCard object from the current phonebook
`EventUsrPbapBrowseList`	Download a list of vCards from the current phonebook
`EventUsrPbapDownloadPhonebook`	Download the current phonebook

Table 12-4 User events used by the PBAP feature (cont.)

User Event	Description
EventUsrPbapSelectPhonebookObject	Select the next phonebook from the same repository
EventUsrPbapBrowseComplete	Stop the current PBAP browsing operation
EventUsrPbapGetPhonebookSize	Retrieve the size of the current phonebook

System events

Table 12-5 System events used by the PBAP feature

System Events	Description
EventSysEstablishPbap	Connect PBAP to the connected HFP device
EventSysPbapDialFail	Indicates the requested PBAP operation failed

Connection

PBAP connections can be requested by generating the EventSysEstablishPbap system event. PBAP can be connected to HFP devices that also support the PBAP profile.

Implementation

When PBAP is connected, it can be used for advanced phonenumber dialing functionality. The advanced dialing functionality provided is:

- Dial most recent entry from the incoming call list
- Dial most recent entry from the missed call list

NOTE Further PBAP functionality is provided by the PBAP library so extra features can be developed by ADK users.

12.1.7 Human Interface Device Profile

The Audio Sink application supports the Bluetooth *Human Interface Device Profile v1.1.1*.

User events

Table 12-6 lists the user events provided to control HID connections with AGs.

Table 12-6 User events used by the HID Selfie feature

User Event	Description
EventUsrHidSelfieRelease	Selfie release
EventUsrHidSelfiePress	Selfie press
EventHidQualificationConnect	Used for HID qualification
EventHidQualificationLimitedDiscoverable	Used for HID qualification

Connection

When an AG supporting HID is connected, the Audio Sink application tries to connect HID.

Implementation

When ADK is connected over HID to a Bluetooth enabled device that is capable of capturing photos.

- Prerequisite: Camera application should be launched in the capturing device.
- On pressing the above designated key on the ADK, an image can be captured by the device.
- When an image has to be captured, an input report with 'Volume Up' has to be sent over HID interrupt channel.
- **Single Shot mode:** Press and Release takes a single shot.
- **Burst Mode:** The ADK also supports Burst mode and if the Selfie button is made Press and hold, the phone software takes it as a burst mode command and multiple shots are captured till the button is released on ADK.

NOTE When not in camera application, sending selfie press and release will be interpreted as volume up command on the smartphone.

12.2 Bluetooth low energy technology

The Audio Sink application supports Bluetooth low energy technology, an array of Generic Attribute Profile (GATT) services using Bluetooth low energy technology transport are supported. The GATT services that are supported by default are:

- GATT manager
- Supported GATT services (server role)
- Supported GATT services (client role)
- GATT events

NOTE For more information on GATT implementation in the Audio Sink application, see the *Audio Sink Application GATT User Guide*

12.2.1 GATT manager

The GATT Manager is used by the Audio Sink application as a support library for GATT connections, it directs incoming GATT requests to the correct sub-task (library) and directs outgoing GATT requests to the correct client device.

12.2.2 Supported GATT services (server role)

GATT Services (Server Role) are supported through individual VM libraries on top the GATT Manager layer. These libraries are designed to look after a single local GATT service and offer the functionality of that service to a remote GATT client.

These libraries rely on a local GATT database that must be registered by the Audio Sink application, remote devices can then access the services though the GATT database.

Table 12-7 Supported GATT services for GATT role server

GATT Service	Description
Generic Attribute Profile Service	Implements handling of the GATT service
Generic Access Profile Server	Implements handling of the GAP service
Battery Service	Implements handling of the BAS service
Device Information Service	Implements handling of the DIS service
Immediate Alert Service	Implements handling of the IAS service
Transmit Power Service	Implements handling of the TPS service
Link Loss Service	Implements handling of the LLS service
Heart Rate Service	Implements handling of the HRS service
GAIA	Implements handling of the GAIA service

12.2.3 Supported GATT services (client role)

GATT Services (Client role) are supported through individual VM libraries that sit on top the GATT Manager layer.

These libraries are designed to look after a single GATT service defined on a remote device. The Audio Sink application must discover which services are supported when a remote device is connected, after which, any services the Audio Sink application knows can be managed must be passed down to the appropriate library so that library can manage the service.

Table 12-8 Supported GATT services for GATT role client

GATT Service	Description
Generic Attribute Profile Service	Manages GATT services on remote GATT clients
Battery Service	Manages BAS services on remote GATT clients
Device Information Service	Manages DIS services on remote GATT clients
Immediate Alert Service	Manages IAS services on remote GATT clients
Apple Notification Service	Manages ANCS services on remote GATT clients
HID Over GATT Profile	Manages HID services on remote GATT clients
Heart Rate Service	Manages HRS services on remote GATT clients

12.2.4 GATT events

Table 12-9 GATT events for heart rate service

Event	Description
`EventUsrBleSimulateHrNotifications`	Event to simulate Heart rate notifications when in HR server role.
`EventSysBleHrSensorNotInContact`	Event indicates HR sensor not in contact and server can stop sending Heart Rate measurements.

Table 12-9 GATT events for heart rate service (cont.)

Event	Description
`EventSysBleHrSensorInContact`	Event indicates HR Sensor in contact and server can start sending Heart Rate measurements.
`EventSysHeartRateThresholdReached`	Event notified when Heart rate threshold is reached when in HR client role. The threshold is configurable using the Sink Configuration tool.

12.3 Dual mode

The Audio Sink application supports Bluetooth low energy technology and Bluetooth BR/EDR simultaneously. This is referred to as Dual mode support. All devices supporting Dual mode must be qualified to the *Bluetooth Core Specification v4.1*.

12.3.1 Supported Bluetooth low energy technology GAP roles

The Audio Sink application supports simultaneous Central and Peripheral Generic Access Profile (GAP) roles.

The Sink App starts advertising and scanning for new devices and no application event is required to trigger the same.

To enable the Bluetooth low energy technology scatternet, user has to configure the number of peripheral and central connections, the sum of which must not exceed 2. Implementation supports at the maximum of 2 Bluetooth low energy technology links to be active at any point.

NOTE For more information on switching GAP roles in the Audio Sink application, see *Audio Sink Application GATT User Guide*

12.3.2 Use case guidelines

Table 12-10 lists the use case guidelines for the use cases in one of the GAP roles described in Peripheral GAP role and Central GAP role. For more information on simultaneous topologies, see the *Audio Sink Application GATT User Guide*.

Table 12-10 Guidelines for Dual mode use case diagrams

Key	Description
	BR/EDR link, arrow points towards slave.
	BR/EDR link, either device can be master/slave.
	Bluetooth low energy technology link, arrow points towards slave (Peripheral) and away from master (Central).
SINK	Audio Sink application (speaker, headset, soundbar).
SUB	Sub-woofer speaker.

Table 12-10 Guidelines for Dual mode use case diagrams (cont.)

Key	Description
REMOTE	Bluetooth low energy technology remote control.
PHONE	Audio source, typically a mobile phone, laptop, or tablet. A Central device will be running the QTIL application for Bluetooth low energy technology.

12.3.3 Peripheral GAP role

When operating in Peripheral mode, the Audio Sink application is able to advertise and be connected to by Central devices.

The Dual mode support also allows a Bluetooth BR/EDR connection to exist between the Central and Peripheral devices. For example, a common use case is for a mobile phone to be the Bluetooth low energy technology Central device, and A2DP or HFP audio to be active between the mobile phone and Audio Sink application.

The use cases described in this section have been tested by QTIL to verify that audio streaming from the BR/EDR link is not affected by the Bluetooth low energy technology link.

Peripheral use case 1

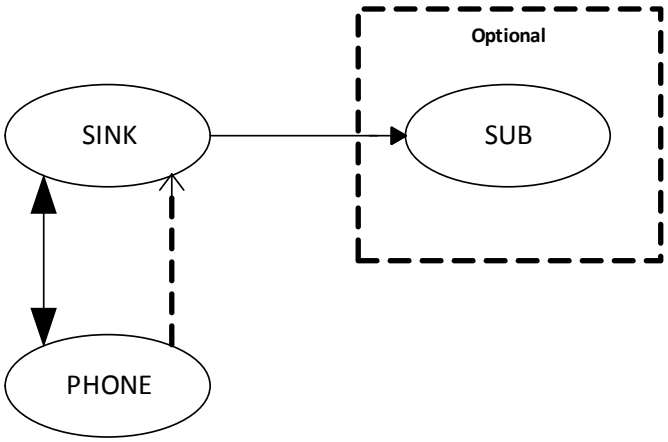


Figure 12-1 Audio Sink application in Peripheral mode with connected subwoofer device

Peripheral use case 2

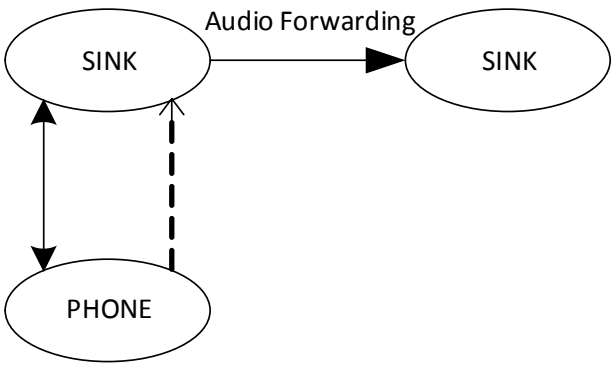


Figure 12-2 Audio Sink application in Peripheral mode with connected peer device

Protected Bluetooth BR/EDR audio

Bluetooth BR/EDR audio must be protected when Dual Mode is being used, so that audio glitches are not observed.

The following guidelines should be followed when the Audio Sink application is operating in the Peripheral role with Bluetooth BR/EDR audio:

- QTIL recommends a Bluetooth low energy technology connection interval between 90 ms and 110 ms to protect the BR/EDR audio. The asynchronous BR/EDR is scheduled around the synchronous Bluetooth low energy technology link, so this connection interval gives time for rescheduling.
- QTIL recommends a Bluetooth low energy technology undirected advertising interval of at least 20 ms if it is advertising at the same time as Bluetooth BR/EDR audio.
- Bluetooth BR/EDR (e)SCO audio will be preserved over Bluetooth low energy technology. A Bluetooth low energy technology connection may drop, or fail to be established if there is a collision with the (e)SCO audio. QTIL recommends trying to negotiate the `S3` eSCO parameter set from the HFP profile specification for CVSD coding, and the `T2` eSCO parameter set for mSBC coding. This results in a T_{esco} interval of 12 slots, which is recommended for reducing the collisions.

12.3.4 Central GAP role

When operating in Central Mode, the Audio Sink application is able to find and connect to nearby Peripheral devices.

NOTE The Audio Sink application is designed to connect to a Bluetooth low energy technology remote control in the Central role. It is not designed to work with a mobile phone or similar device operating in the Peripheral GAP role.

The use cases described in this section have been tested by QTIL to verify that audio streaming from the BR/EDR link is not affected by the Bluetooth low energy technology link.

Central use case 1

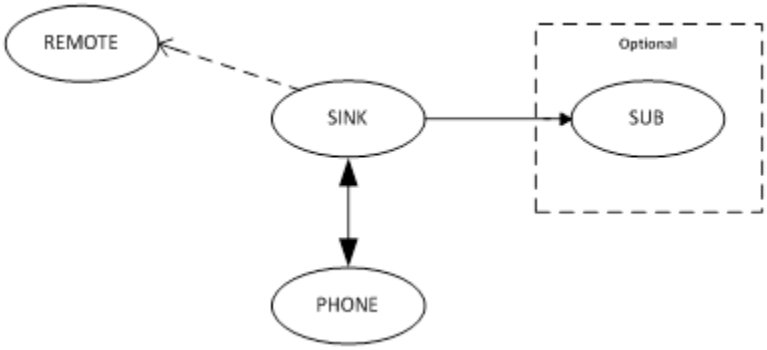


Figure 12-3 Audio Sink application in Central mode with connected subwoofer device

Central use case 2

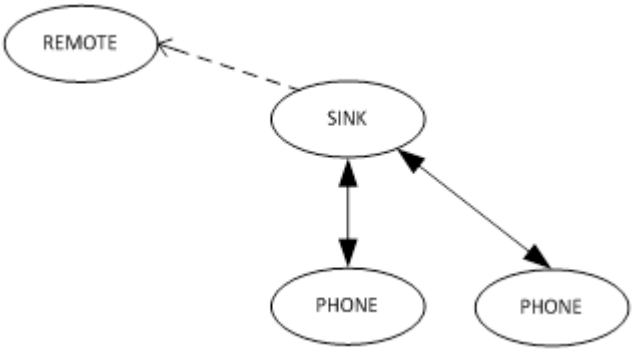


Figure 12-4 Audio Sink application in Central mode with multipoint connections

Central use case 3

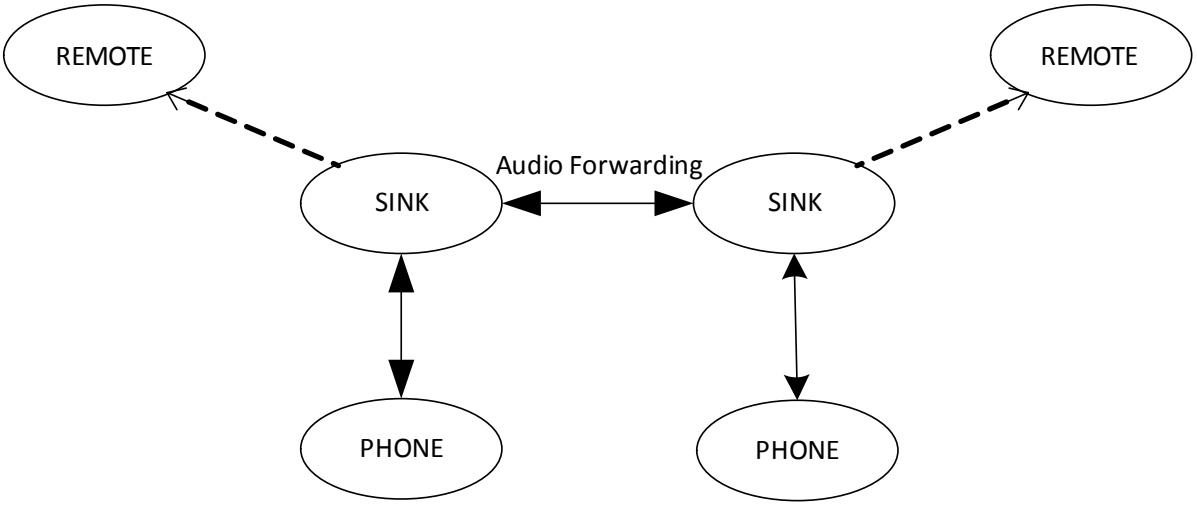


Figure 12-5 Audio Sink application in Central mode with peer device mode active

Protected Bluetooth BR/EDR audio

Bluetooth BR/EDR audio must be protected when Dual mode is being used, so that audio glitches are not observed.

The following guidelines should be followed when the Audio Sink application is operating in the Central role with Bluetooth BR/EDR audio:

- QTIL recommends a Bluetooth low energy technology connection interval between 90 ms and 110 ms to protect the BR/EDR audio. The asynchronous BR/EDR is scheduled around the synchronous Bluetooth low energy technology link, so this connection interval gives time for rescheduling.
- QTIL recommends setting a suitable Bluetooth low energy technology scanning interval and window if it is scanning at the same time as Bluetooth BR/EDR audio.
- If 100% duty is requested, 100% is attempted. It is up to the application to request sensible parameters.

For a lower duty cycle scan, a 2.5% scanning period would be suitable. For example, 320 slot interval and 8 slot window. For a higher duty cycle scan, a 10% scanning period would be suitable. For example, 80 slot interval, 8 slot window.

12.4 RSSI inquiry

The Audio Sink application supports RSSI Inquiry scanning for two modes:

1. Proximity pairing.
2. Proximity connection.

12.4.1 Inquiry search

The Audio Sink application can perform an Inquiry search to discover nearby Bluetooth devices that are discoverable.

When the inquiry search is complete,that is, either a maximum number of devices have been found or the Inquiry Timeout has fired, the list of discovered devices is sorted so the device with the strongest RSSI value is put at the top.

NOTE To configure the inquiry scanning procedure, see the *Audio Sink Application Configuration User Guide*.

12.4.2 Proximity pairing

Proximity Pairing allows the Audio Sink application device to make a pairing request to the nearest discovered remote device. There are two modes of operation for Proximity Pairing:

1. Attempt All
2. Attempt One

When a Proximity Pairing procedure is started, the Audio Sink application device starts Discoverable allowing a remote device to discover and pair in the conventional way.

Proximity pairing timeout

When a Proximity Pairing procedure is started, the Proximity Pairing Timeout is armed; this is a configurable internal timer, to configure this timer see *Audio Sink Application Configuration User Guide*. While this timer is armed, it can be considered the Proximity Pairing procedure is active.

Attempt All

With Attempt All, Proximity Pairing performs an Inquiry Search. When this is complete, an attempt to pair with the first discovered device in the list is actioned. If this pairing attempt fails, a pairing request to the next device in the list will be attempted.

This continues until a successful pairing is made, or the end of the list is reached. If the end of the list is reached, another Inquiry Search is performed and the actions repeated.

Attempt One

With Attempt One, Proximity Pairing performs an Inquiry Search. When this is complete, an attempt to pair with the first discovered device in the list is actioned. If this pairing attempt fails a new inquiry search is performed and the same procedure repeated until a successful pairing is made.

Ending proximity pairing

There are two scenarios that signal the end of the Proximity Pairing procedure, these are:

1. A successful pairing has occurred:
   - The Pairing Mode Timeout is canceled.
2. The Pairing Mode Timeout fires:
   - The Audio Sink application device remains in Discoverable mode but no further action is taken.

Limitations

For proximity pairing to work, the remote device must be in Discoverable mode.

12.4.3 Proximity connection

This feature is designed to improve reconnection times when there are numerous entries in the Audio Sink application's PDL.

After an inquiry scan has been performed, Proximity Connection allows the Audio Sink application device to make a connection request to the nearest discovered remote device.

- If the connection attempt to the nearest discovered device fails, the normal reconnection procedure is actioned.
- If the connection is successful the Audio Sink application device enters the Connected state.

NOTE Connection attempts are only actioned for discovered devices that also exist in the Audio Sink application's PDL.

12.4.4 RSSI sink application events

Table 12-11 Sink application events used to control RSSI inquiry

Event	Description
`EventUsrRssiPair`	Initiate the Proximity Pairing Procedure
`EventSysRssiResume`	When the inquiry search has completed, but no devices were discovered, this event is used to start a new inquiry search. **NOTE** The number of new searches performed can be configured, see the *Audio Sink Application Configuration User Guide.*
`EventSysRssiPairTimeout`	System event used to indicate when the Proximity Pairing timeout fires
`EventSysRssiPairReminder`	System event generated every 5 seconds during a Proximity Pairing procedure. This is used to set an indication that Proximity Pairing procedure is still active.

13 Audi Sink application user interface

The Audio Sink application provides the following controlers and indication systems used to provide implement a user interface:

- Input manager
- Bluetooth low energy technology remote controller
- Infra-red remote controller
- Buttons
- Logical inputs
- Simple speech recognition
- Tones
- Audio prompts
- LEDs
- Display
- Secure connections

13.1 Input manager

This section covers the Input Manager which is responsible for managing input events generated by Input Monitor tasks.

13.1.1 Input Monitor Tasks

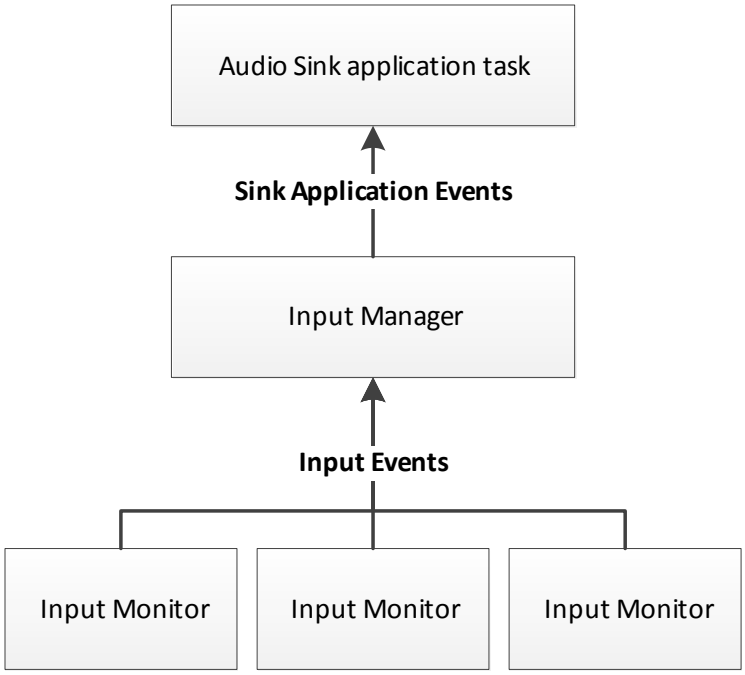
An Input Monitor task defines a module that is responsible for notifying the Input Manager of any input events based on a resource it is monitoring.

Table 13-1 Input monitor tasks of the audio sink application

Input Monitor	Description
Bluetooth low energy technology Input Monitor	See Bluetooth low energy technology remote controller
Infra-red Input Monitor	See Infra-red remote controller

13.1.2 Architecture

Architecture shows the layered architecture used by the Input Manager.



13.1.3 Input events

Each Input Monitor indicates to the Input Manager when an input event occurs.

Table 13-2 Input Manager events

Input Event	Description
`InputEventDown`	The input have been pressed down
`InputEventVShortRelease`	The input(s) have been released before the short timer fires
`InputEventShortTimer`	The short timer fires while the input(s) are still pressed down
`InputEventShortRelease`	The input(s) have been released after the short timer has fired, but before the long timer has fired
`InputEventLongTimer`	The long timer fires while the input(s) are still pressed down
`InputEventLongRelease`	The input(s) have been released after the long timer has fired, but before the very long timer has fired
`InputEventVLongTimer`	The very long timer fires while the input(s) are still pressed down
`InputEventVLongRelease`	The input(s) have been released after the very long timer has fired, but before the very very long timer has fired
`InputEventVVLongTimer`	The very very long timer fires while the input(s) are still pressed down

Table 13-2 Input Manager events (cont.)

Input Event	Description
`InputEventVVLongRelease`	The input(s) have been released after the very very long timer has fired
`InputEventRepeatTimer`	The repeat timer has fired while the input is still pressed down

13.1.4 Timers

Table 13-3 lists the timers supported by the Input Manager. The values of each timer are configurable, see the *ADK Sink Application Configuration User Guide*.

Table 13-3 Input Manager timers

Timer	Description
`multipleDetectTimer`	Timer that is armed immediately after the first input is pressed, when this timer fires, no further multiple buttons are included in the pattern
`shortTimer`	Fires after a short duration of the input(s) being held down
`longTimer`	Fires after a long duration of the input(s) being held down
`vLongTimer`	Fires after a very long duration of the input(s) being held down
`vvLongTimer`	Fires after a very very long duration of the input(s) being held down
`repeatTimer`	Fires repeatedly at a configured interval while the input(s) are held down

NOTE By default, the Input Manager timers are used by every Input Monitor task to ensure each input source is based on the same user interaction.

13.1.5 Virtual buttons

The Input Manager uses the concept of virtual buttons to translate input events received from Input Monitor tasks. A virtual button is not a physical button at the Input Manager, it could be a physical button or a button on a remote controller managed by an input monitor.

13.1.6 Event translation

The Input Manager translates Input Events to Audio Sink application events. It does this using a translation table.

Each entry of the table outlines a set of rules that must be matched for a Audio Sink application event to be generated, see Table 13-4.

Table 13-4 Input Manager event translation requirements

Lookup Table Field	Description
Input event	The Input event generated
Input mask	Identifies the virtual button(s) held down when the Input event was generated

Table 13-4 Input Manager event translation requirements (cont.)

Lookup Table Field	Description
State mask	Identifies the Application state(s) that must be active when the Input event was generated
Audio Sink application event	The Audio Sink application event that is generated if the Input event was generated when the Input mask and current state match the translation requirements.

13.2 Bluetooth low energy technology remote controller

The Bluetooth low energy technology Input Monitor is a feature of the Audio Sink application that supports communication with Bluetooth low energy technology HID remote controllers supporting the HID Over GATT profile.

Remote controller devices send notifications to the Audio Sink application device containing HID consumer codes indicating user interactions.

13.2.1 Events

The Bluetooth low energy technology Input Monitor is controlled by a set of Audio Sink application user events, see Table 13-5.

Table 13-5 Sink application events for the Bluetooth low energy technology input monitor

Audio Sink Application Event	Description
`EventUsrBleEnterSuspendMode`	Put all connected remote controllers into suspend mode
`EventUsrBleExitSuspendMode`	Take all connected remote controllers out of suspend mode
`EventUsrBleEnableNotifications`	Request that all connected remote controllers send notifications to the Audio Sink application device when user interactions occur
`EventUsrBleDisableNotifications`	Request that all connected remote controllers do not send any notifications to the Audio Sink application device

13.2.2 Multiple remote controllers

The Bluetooth low energy technology Input Monitor can manage multiple concurrent connections with HID Devices. To configure the maximum number of supported concurrent connections, refer to the *ADK Sink Configuration User Guide*.

13.2.3 HID translation

Notifications received must contain one or more user HID codes, these codes are translated to a virtual button and the appropriate Input events generated.

The Input Manager is responsible for translating Input events on virtual buttons to Sink Application user events. The list of HID codes supported is configurable, refer to the *ADK Sink Configuration User Guide* for a guide on configuring the Bluetooth low energy technology Input Monitor.

13.3 Infra-red remote controller

The Infra-red Input Monitor is responsible for managing events received from infra-red remote controllers.

NOTE The Infra-red Input Monitor does not support multiple infra-red buttons being pressed.

13.3.1 Supported infra-red consumer protocols

The Audio Sink application supports two Infra-red protocols:

- NEC
- RC5

NOTE The Audio Sink application cannot support both protocols simultaneously, that is, it cannot communicate with both an NEC and an RC5 remote simultaneously.

13.3.2 Infra-red Learning mode

The infra-red Learning mode is a feature that allows the Infra-red Input Monitor to learn new Infra-red codes. To learn a new Infra-red code:

1. Infra-red Learning mode is enabled using the `EventUsrStartIRLearningMode` control event.
2. If a known button on the infra-red control is pressed to generate an Audio Sink application event, the event is not recorded.
3. If an unknown button on the infra-red remote controller is pressed:
   - ☐ The button is mapped to an Audio Sink application event for the Audio Sink application state when the unknown button is pressed.

Any future presses on this button when in the same Audio Sink application state are now known and generate the associated Audio Sink application user event.

NOTE (1) Infra-red Learning mode can be disabled at any time using the `EventUsrStopIRLearningMod` control event..

(2) Events can be learned from any connected input whose input is monitored by an Input Monitor task that sends Input events to the Input Manager.

13.3.3 Events

Control events

The Infra-red Input Monitor can be controlled by a set of Audio Sink application user events, see Table 13-6.

Table 13-6 Sink application control events for the Infra-red input monitor

Audio Sink application Event	Description
`EventUsrStartIRLearningMode`	Enable Infra-red Learning mode
`EventUsrStopIRLearningMode`	Disable Infra-red Learning mode
`EventUsrClearIRCodes`	Clear all Infra-red codes learned by the Audio Sink application

Indication events

The Infra-red Input Monitor uses indication events to indicate interactions to a user, see Table 13-7.

Table 13-7 Sink application indication events for the Infra-red input monitor

Audio Sink Application Event	Description
`EventSysIRCodeLearnSuccess`	Indicates when an infra-red event has been learned
`EventSysIRCodeLearnFail`	Indicates when the infra-red Input Monitor failed to learn an infra-red event
`EventSysIRLearningModeTimeout`	Indicates the infra-red Learning mode has timed out and is now disabled
`EventSysIRLearningModeReminder`	Indicates that Learning mode is active

13.3.4 Infra-red translation

The Audio Sink application maps infra-red codes received from an infra-red remote controller to a virtual button. The Input Manager translates Input events on virtual buttons to Sink Application user events. The default supported list of infra-red codes is configurable, see the *ADK Sink Configuration User Guide*.

13.4 Buttons

The Audio Sink application supports local button presses as a user interface input. Buttons are managed by the Button Manager sub task and can be configured, to configure the buttons, refer to the *Audio Sink Application Configuration User Guide*.

NOTE A helper application buttonparsepro provides a quick and easy method of generating the code to bind button names to PIO events and to define the message sent to the application task when the button event occurs, refer to the description of the **buttonparsepro** helper application in the *Bluetooth SDK Command Line Tools User Guide*.

13.5 Logical inputs

The Audio Sink application relies on events triggered on logical inputs rather than physical PIOs to generate user events.

13.5.1 Event generation

The Audio Sink application has an event lookup table that is used to generate user events based on:

- The current application state
- Input Events:
  - Logical input or inputs being pressed
  - Timer firing while logical input or inputs are pressed
  - Logical nput or inputs being released

This table is configurable using the Sink Configuration Tool see the *ADK Sink Configuration Tool User Guide*.

13.5.2 Button translation

Buttons on physical PIOs connected to the BlueCore device can be monitored by the Audio Sink application, these buttons must be translated to a logical input.

To configure button translation, see the the *ADK Sink Configuration Tool User Guide*.

13.5.3 Button locking

The Audio Sink application allows the buttons to be locked or unlocked, when buttons are locked any button presses are ignored and when unlocked are processed as normal.

Table 13-8 User events to lock and unlock buttons

User Event	Description
`EventUsrButtonLockingOn`	Lock the buttons
`EventUsrButtonLockingOff`	Unlock the buttons
`EventUsrButtonLockingToggle`	Toggle between locking and unlocking the buttons

13.6 Simple speech recognition

The Audio Sink application supports Simple Speech Recognition (SSR) allowing users to control the device's behavior using simple voice commands.

SSR detects when the user says “Yes” or “No” to control the user interface of the device.

13.6.1 User events

The Audio Sink application allows users to control or disable the SSR algorithm, see Table 13-9.

Table 13-9 User events controlling the SSR algorithm

User Event	Description
`EventUsrSSROn`	Enable SSR functionality
`EventUsrSSROff`	Disable SSR functionality
`EventUsrSpeechRecognitionTuningStart`	Start the SSR Tuning mode

NOTE The SSR enabled/disabled state is persistent across power cycles of the Audio Sink application device.

13.6.2 Tuning mode

A Tuning mode is provided which allows fine tuning of the SSR algorithm within the DSP, it is important that the Tuning mode is only be used while tuning the SSR algorithm and is disabled in the final implementation.

To start the SSR Tuning mode, the user event `EventUsrSSROn` must be generated.

While in Tuning mode, if either a “Yes” or “No” response is received the application restarts the speech recognition automatically. When this happens, the device connects to the Universal Front End DSP Configuration Tool and SSR can be fine-tuned.

13.7 Tones

The Audio Sink application can notify users of key events or reminders using audio tones.

Tones can be mapped to events using the ADK Sink Configuration Tool, see the *ADK Sink Configuration Tool User Guide* for more details.

13.7.1 Custom tones

The audio sink application supports custom tones, to setup and configure custom tones refer to the *Audio Sink Application Configuration User Guide*.

13.8 Audio prompts

The Audio Sink application can notify users of key events or reminders using audio prompts. These prompts are short audio files that can contain music, speech or complex tones.

Details on how to add and configure audio prompts in an application using the Audio Sink Application Configuration Tool are described in the *ADK Audio Prompts Application Note*.

13.8.1 Events

Audio Prompts can be controlled through the user events listed in Table 13-10.

Table 13-10 User events to control audio prompts

User Event	Description
`EventUsrAudioPromptsOn`	Turn audio prompts on
`EventUsrAudioPromptsOff`	Turn audio prompts off
`EventUsrSelectAudioPromptLanguageMode`	Select the next language bank

13.9 LEDs

The Audio Sink application can notify users of state information using LEDs. LED patterns for each state can be configured so the user can identify the active state of the application at any time.

13.9.1 LED indications based on state and event changes

The following state changes can be indicated to the user using mapped LED patterns

- TWS/ShareMe connection
- Party mode connection
- ANC active
- Multipoint connection

Table 13-11 System events available for LED indications

System Event	Description
`EventSysPartyModeConnected`	Party mode connected
`EventSysPartyModeDisconnected`	Party mode disconnected
`EventSysMultipointConnected`	Multipoint connected
`EventSysMultipointDisconnected`	Multipoint disconnected
`EventSysPeerConnected`	Peer device connected
`EventSysPeerDisconnected`	Peer device disconnected
`EventSysAncDisabled`	ANC disabled
`EventSysAncActiveModeEnabled`	ANC enabled in active mode
`EventSysAncLeakthroughModeEnabled`	ANC enabled in leakthrough mode

13.9.2 LED indications upon audio source selection

There are events available for audio source connection disconnection, which can be indicated to user using LED patterns.

- When the input audio source is connected a system event is generated, this can be mapped to LED pattern and indicated to the user.
- Events upon connection/disconnection of Line in, SPDIF, USB, and BT(A2DP/HFP) are available, which can be mapped using configuration tool for indication of LED pattern.

13.9.3 LED indications on codec negotiation

When audio streaming is from an A2DP audio source, the negotiated codec can be indicated to the user using the system events generated upon codec negotiation.

- Whenever streaming is started the corresponding codec is indicated through system events.
- When stream is suspended or closed the respective codec exit event is also available.

Table 13-12 shows the list of codec events available for indication

System event	Description
`EventSysCodecSbcInUse`	SBC codec is in use
`EventSysCodecSbcDisconnected`	SBC disconnected
`EventSysCodecMp3InUse`	MP3 codec is in use
`EventSysCodecMp3Disconnected`	MP3 disconnected
`EventSysCodecAacInUse`	AAC codec is in use
`EventSysCodecAacDisconnected`	AAC disconnected
`EventSysCodecAptxInUse`	aptX codec is in use
`EventSysCodecAptxDisconnected`	aptX disconnected
`EventSysCodecAptxLLInUse`	aptX-LL codec is in use
`EventSysCodecAptxLLDisconnected`	aptX-LL disconnected
`EventSysCodecAptxHDInUse`	aptX-HD codec is in use
`EventSysCodecAptxHDDisconnected`	aptX-HD disconnected
`EventSysCodecFaststreamInUse`	Faststream codec is in use
`EventSysCodecFaststreamDisconnected`	Faststream disconnected

They are configurable using the Sink Configuration Tool refer to the *ADK Sink Configuration Tool User Guide*.

13.10 Display

With the addition of the CNS10010v1 Display Module the Audio Sink application can notify users of information using a simple display, e.g. battery level, audio source, currently playing audio track (AVRCP), caller ID (HFP), and other information can be displayed to the user.

13.11 Secure connections

The Audio Sink application supports BR/EDR and Bluetooth low energy technology secure connections. See the *BR/EDR and Bluetooth low energy technology Secure Connections User Guide* for more information on secure connection feature and its configurations.

14 USB

The Audio Sink application can be configured to enumerate as a USB device, with support for different class specifications. The supported USB device classes are detailed in:

- Battery charging
- Audio device
- HID consumer
- Mass storage

14.1 Battery charging

The Audio Sink application is capable of supporting USB Battery Charging. This specification is met by the USB Charger Configuration settings available in the Audio Sink application.

14.2 Audio device

The Audio Sink application is capable of supporting USB Audio Microphone and Speaker. This allows audio to be routed to the Sink applications speaker from a USB host device. Alternatively it allows the microphone input to the Sink application to be used as a USB microphone source to the USB host device.

Audio controls are included as part of the speaker and microphone configuration so that the USB host can control the audio levels and stay synchronized with the Sink application. The microphone can be muted. The speaker can be muted and the volume can be incremented or decremented.

14.3 HID consumer

The Audio Sink application is capable of supporting HID Consumer Usage Page controls. This allows the Audio Sink application events to be translated to HID consumer control events.

Table 14-1 HID consumer control events

User Event	HID Consumer Control
`EventUsrMainOutVolumeUp`	Volume Increment **NOTE** Is only sent when USB is the active audio source
`EventUsrMainOutVolumeDown`	Volume Decrement **NOTE** Only sent when USB is the active audio source

Table 14-1 HID consumer control events (cont.)

User Event	HID Consumer Control
`EventUsrUsbMute`	Mute
`EventUsrUsbPlayPause`	Play/Pause
`EventUsrUsbStop`	Stop
`EventUsrUsbFwd`	Scan Next Track
`EventUsrUsbBack`	Scan Previous Track

14.4 Mass storage

The Audio Sink application is capable of supporting read-only USB Mass Storage, this allows for inclusion of user manuals or other documentation as part of the file system.

By default the `START.HTM` file is added to the file system when USB Mass Storage is enabled. The `usb_fat` and `usb_root` files are also added to the file system. These describe the FAT entries needed to read and display the default `START.HTM` file.

15 Subwoofer

The Audio Sink application manages Bluetooth connections with subwoofer devices that support the QTIL proprietary SWAT protocol.

NOTE A subwoofer connection cannot be supported when Peer Device mode is active.

15.1 Subwoofer user events

Table 15-1 lists the user events available to control the subwoofer.

Table 15-1 User events to control subwoofer behavior

User Event	Description
`EventUsrSubwooferStartInquiry`	Start searching for a new subwoofer device
`EventUsrSubwooferVolumeDown`	Lower the volume of the subwoofer
`EventUsrSubwooferVolumeUp`	Increase the volume of the subwoofer
`EventUsrSubwooferDisconnect`	Disconnect the subwoofer device
`EventUsrSubwooferDeletePairing`	Remove the paired subwoofer from the Paired Device List

15.2 Subwoofer system events

Table 15-2 lists the system events the system automatically generates when subwoofer support has been enabled.

Table 15-2 System events to control subwoofer behavior

System Event	Description
`EventSysSubwooferCheckPairing`	Automatically start searching for a subwoofer device if one has not been previously paired
`EventSysSubwooferOpenLLMedia`	Opens a low latency media channel with the subwoofer
`EventSysSubwooferOpenStdMedia`	Opens a standard latency media channel with the subwoofer
`EventSysSubwooferCloseMedia`	Close the open media channel
`EventSysSubwooferStartStreaming`	Start streaming audio data to the subwoofer
`EventSysSubwooferSuspendStreaming`	Suspend streaming of audio data to the subwoofer
`EventSysSubwooferSetVolume`	Automatically set the subwoofer volume

Table 15-2 System events to control subwoofer behavior (cont.)

System Event	Description
`EventSysTrimVolumeMax`	Notify user that the maximum subwoofer trim level has been reached
`EventSysTrimVolumeMin`	Notify user that the minimum subwoofer trim level has been reached

15.3 Subwoofer pairing

The Audio Sink application can discover, pair, and connect to a subwoofer device, to do this, a subwoofer inquiry search must be performed, this can be configured to occur automatically or so that a user can initiate this procedure.

When paired, the subwoofer device becomes protected, this means that if a user initiates a clear the paired device list procedure, the pairing information for the subwoofer device is not removed. The only way a user can remove the pairing for the subwoofer is to explicitly generate the user event `EventUsrSubwooferDeletePairing`.

15.4 Subwoofer reconnection

The subwoofer device is always responsible for connecting to the Audio Sink application, this is to ensure the audio stream (if present) at the Audio Sink application device is protected and is not interrupted by the page scanning for another device.

15.5 Subwoofer volume control

The Subwoofer volume is automatically synchronized with the Audio Sink applications volume to ensure that both devices operate at the same volume level. However, this is not always adequate due to hardware choices etc. for this reason the Audio Sink application provides a subwoofer trim volume, which can be used to tweak the volume of the subwoofer device.

This can be easily actioned by generating the user events listed in Table 15-2.

15.6 Subwoofer media transport

The Audio Sink application uses two methods of transporting media packets to the subwoofer device:

1. High- bandwidth low-latency transport (eSCO)
   The low-latency transport is used when the Audio Sink application does not have to manage audio streaming from another remote device.
2. Low-bandwidth standard-latency transport (L2CAP).
   The standard-latency transport is used when the Audio Sink application has to manage Bluetooth audio streaming from a remote device and simultaneously relay that audio to the subwoofer.

NOTE If the low-latency transport is used for transmitting audio received from a Bluetooth audio source, the Audio Sink application device cannot guarantee to transmit subwoofer media when required. This leads to buffer exhaustion on the subwoofer device and audible pops and clicks become apparent. Using the standard latency transport ensures that the media

packets are always transmitted and significantly reduces the risk of buffer exhaustion on the subwoofer device.

Table 15-3 shows the media transport used for each audio source supported by the Audio Sink application.

Table 15-3 Subwoofer media transports used for audio sources

Audio Source	Subwoofer Transport
Speech Recognition	None (audio not sent to subwoofer)
Bluetooth SCO (call audio)	None (audio not sent to subwoofer)
A2DP Audio	low-bandwidth, standard-latency
USB Audio	high-bandwidth, low-latency
Wired Audio (PCM or I^2S)	high-bandwidth, low-latency
FM Receive	high-bandwidth, low-latency
SPDIF Audio	high-bandwidth, low-latency
Bluetooth SCO (audio)	None (audio not sent to subwoofer)
No Active Source	None (audio not sent to subwoofer)

16 FM receiver control

The Audio Sink application can control an FM receiver. FM receiver hardware is required for this to work, see .

16.1 FM receiver user events

Table 16-1 lists the User events available to control and use the FM receiver.

Table 16-1 FM receiver controls

User Event	Description
`EventUsrFmRxOn`	Enable the FM receiver (turns on the FM receiver)
`EventUsrFmRxOff`	Disable the FM receiver (turns off the FM receiver)
`EventUsrFmRxTuneUp`	Initiates an FM auto tune in an increasing frequency direction
`EventUsrFmRxTuneDown`	Initiates an FM auto tune in a decreasing frequency direction
`EventUsrFmRxStore`	Store the currently tuned frequency
`EventUsrFmRxTuneToStore`	Tunes the FM receiver to the stored frequency in increasing order
`EventUsrFmRxErase`	Erases the currently tuned frequency

16.2 FM receiver tuning

FM receivers work by being tuned to a particular FM frequency. Therefore, the Audio Sink application allows users to tune the FM hardware to find a particular FM radio station. User events have been implemented to provide this support, see .

16.2.1 Storing radio stations

The Audio Sink application allows users to store their favorite radio stations, a maximum of 3 radio stations can be stored. The currently tuned frequency must be set to the radio station that is to be stored, and then a `EventUsrFmRxStore` user event must be generated.

NOTE If the maximum number of stations have already been stored, the user must delete an existing stored station using `EventUsrFmRxErase` before storing the new station.

16.2.2 Erasing stored radio stations

The Audio Sink application allows users to erase a previously stored station from the list of stored stations. To do this the FM receiver must be tuned to the station to be erased. While the tuned radio station is active, the user event to erase the stored station, `EventUsrFmRxErase` must be generated.

NOTE To easily find a stored station before erasing it, the stored stations can be cycled using the `EventUsrFmRxTuneToStore` event; this user event tunes the FM receiver to the stored frequency in increasing order.

17 NFC in the Audio Sink application

The NFC (Near Field Communication) support that the Audio Sink application can provide is described in:

- NFC setup
- NFC tag detection

17.1 NFC setup

NFC hardware can be used to detect when a remote device has been placed close to the Audio Sink application device:

- The NFC hardware must be physically connected to a PIO of the device running the Audio Sink application.
- The NFC hardware will be treated like a button and must be mapped using the Audio Sink Application Configuration Tool.

17.2 NFC tag detection

When the NFC hardware detects a device close (finds an NFC tag from another device), it must toggle the connected PIO as if it were a button. The Audio Sink application detects the PIO being toggled and automatically generates the user event signaling an NFC device has been detected, see Table 17-1.

Table 17-1 Audio Sink Application Events for NFC

Audio Sink Application Event	Description
`EventUsrNFCTagDetected`	Indicates that an NFC device has been detected nearby

17.2.1 NFC actions

The NFC action that occurs depends on the current state of the Audio Sink application, see Table 17-2.

Table 17-2 NFC actions

Audio Sink Application State	Action
Not connected to an AG	Start Discoverable mode: ■ Allow remote devices to discover and pair with the Audio Sink application
Connected to an AG	Perform Audio Transfer: ■ If audio was routed at the AG, audio is routed at the Audio Sink application ■ If audio was routed at the Audio Sink application, audio is routed at the AG

18 Software Upgrade

18.1 DFU upgrade

Device Firmware Upgrade is standardized by the USB Implementers Forum as a device class specification, the Audio Sink application is compliant with this standard and includes security checks:

- Audio Sink application devices are programmed with RSA public keys when manufactured.
- DFU files are signed with RSA private keys to make sure that only DFU files for the specific product can be used by the end user.
- The supported transport for DFU is USB.

DFU can be used to update:

- Firmware stack.
- VM application.
- Other files in the VM file system (voice prompts, language packs, and so on).
- PS Keys.
- There are 2 DFU upgrade methods, both DFU tools are available as part of the **Qualcomm® BlueSuite™** download:
  - Using QTIL driver and **DFUWizard** application. This is referred to as Traditional DFU mode.
  - DFU using USB HID interface and TestEngine **HidDfu** tools. This is referred to as Driverless DFU because the end user does not have to install any device drivers.

Table 18-1 User events controlling DFU upgrade

User Event	Description
`EventUsrEnterDFUMode`	Puts the device into Traditional DFU mode to upgrade using **DFUWizard**
`EventUsrEnterDriverlessDFUMode`	Puts the device into Driverless DFU mode to upgrade using **HidDfu** tools

These events trigger the device to restart and enumerate in DFU mode where the appropriate tools can connect to the device and carry out the upgrade.

18.1.1 Generating DFU files

Generating signed DFU files for device upgrade is detailed in the *Audio Sink Application DFU User Guide*.

18.2 Remote software upgrade

The Audio Sink application provides a remote software upgrade feature. This feature provides the ability to download and update the software running on the device. For further details, see the *Audio Sink Application Software Upgrade User Guide*.

19 Power management

The Audio Sink application contains a power management system to monitor discharge and charging of the attached battery. See the *Audio Sink Application Power Management User Guide*.

20 Device Under Test mode

The Audio Sink application supports Device Under Test (DUT) mode allowing production tests to be performed on the device.

NOTE While DUT mode is active, normal operation is not supported.

Table 20-1 Audio sink application events controlling DUT mode

Audio Sink Application Event	Description
`enterDutMode`	Makes the device connectable and discoverable so that a Bluetooth tester can connect to it and run production tests.
`enterTxContinuousTestMode`	This event is equivalent to the BlueSuite function `radiotestTxstart`, which is used to test the output power of a device.
`enterServiceMode`	Clears the paired device list and puts the device into discoverable mode.
	Puts the device into DUT mode. This happens automatically when calling other Device Under Test functions.
`enterAudioTestMode`	Start Audio Loopback mode. The audio is routed from the microphone to the speaker on the device.
`enterToneTestMode`	Route audio from the microphone to the speaker and play an audio tone.
`enterKeyTestMode`	Puts the device into a mode where button presses cycle the LED through the available colors.

Document references

Document	Reference
xIDE User Guide	80-CT405-1/CS-00101500-UG
PSTool User Guide	80-CT424-1/CS-00101505-UG
Audio Sink Application Project Resources	80-CT449-1/CS-00332640-UG
ADK Configuration Tool User Guide	80-CT426-1/CS-00309942-UG
ADK Applications Configuration Architecture Overview	80-CU111-1/CS-00400589-TO
ADK Application Configuration System	80-CT548-1/CS-00400610-UG
ADK Build Scripts XML Definitions User Guide	80-CT541-1/CS-00346862-UG
ADK Audio Sink Application Peer Device User Guide	80-CT414-1/CS-00316086-UG
Audio Sink Application Custom AT Commands User Guide	80-CT428-1/CS-00330103-UG
ADK Audio Prompts Application Note	80-CT418-1/CS-00237358-AN
ADK 4.3 Audio Sink Application Software Upgrade User Guide	80-CF420-1/CS-00406799-UG
Audio Sink Application DFU User Guide	80-CT448-1/CS-00331713-UG
Audio Sink Application GATT User Guide	80-CT429-1/CS-00329966-UG
Audio Sink Application Power Management User Guide	80-CT450-1/CS-00333471-UG
Qualcomm BlueCore Music Manager User Guide (E00F)	80-CF438-1/CS-00406876-UG
ADK 4.2 I^2S User Guide	80-CF421-1 /CS-00406801-UG
Display and FM Expansion Board - Quick Start Guide	80-CT440-1/CS-00307647-DC
BR/EDR and Bluetooth low energy technology SC User Guide	80-CF327-1/CS-00345503-UG
Headset Profile v1.2	www.bluetooth.org
Handsfree Profile v1.7	www.bluetooth.org
Human Interface Device v1.1.1	www.bluetooth.org
Advanced Audio Distribution Profile, v1.3.1	www.bluetooth.org
Audio/Video Remote Control Profile v1.6	www.bluetooth.org
Bluetooth Message Access Profile v1.1	www.bluetooth.org
Phone Book Access Profile v1.1.1	www.bluetooth.org

Terms and definitions

Term	Definition
A2DP	Advanced Audio Distribution Profile
AAC	Advanced Audio Coding
ACL	Asynchronous ConnectionLess
ADC	Analog-to-Digital Converter
ADK	Audio or Application Development Kit
AG	Audio Gateway Device
API	Application Programming Interface
AVRCP	Audio/Video Remote Control Profile
BAS	Battery Service
BLE	Bluetooth low energy, now Bluetooth low energy technology
BlueCore	Group term for the range of QTIL Bluetooth wireless technology ICs
Bluetooth low energy technology	Previously BLE
Bluetooth	Set of technologies providing audio and data transfer over short-range radio connections
BR	Basic Data Rate
cVc	Clear Voice Capture
DIP	Device ID Profile
DIS	Device Information Service
DSP	Digital Signal Processor
DUT	Device Under Test
EDR	Enhanced Data Rate
eSCO	Extended SCO
FM	Frequency Modulation
GAIA	Generic Application Interface Architecture
GAP	Generic Access Profile
GATT	Generic Attribute Profile
HF	Handsfree Device
HFP	Handsfree Profile
HID	Human Interface Device

Term	Definition
HOGP	HID Over GATT Profile
HSP	Headset Profile
IAS	Immediate Alert Service
IC	Integrated Circuit
ID	Identifier
L2CAP	Logical Link Control and Adaptation Protocol
LED	Light Emitting Diode
MAP	Message Access Profile
MP3	MPEG-1 or MPEG-2 Audio Layer III Coding
OBEX	OBject EXchange protocol
PBAP	Phone Book Access Profile
PC	Personal Computer
PCM	Pulse Code Modulation
PDL	Paired Device List (same as Trusted Device List)
PIO	Programmable Input/output
PS Key	Persistent Store Key
QTIL	Qualcomm Technologies International, Ltd.
RDS	Radio Data System
RFCOMM	Radio Frequency COMMunication
RSSI	Received Signal Strength Indication
SBC	Sub-band Coding
SCO	Synchronous Connection-Oriented
SDP	Service Discovery Protocol; element of Bluetooth
SIG	(Bluetooth) Special Interest Group
SPDIF	Sony/Philips Digital InterFace
SPI	Serial Peripheral Interface
SPP	Serial Port Profile
SWAT	Subwoofer Audio Transfer Profile
TDL	Trusted Device List (same as Paired Device List)
TPS	Transmit Power Service
TWS	TrueWireless
UART	Universal Asynchronous Receiver Transmitter
UFE	Universal Front End
USB	Universal Serial Bus
VM	Virtual Machine
xIDE	The QTIL Integrated Development Environment