Qualcomm Technologies, Inc.

Linux Android Audio Customization and Debugging Guide

80-NA157-193 F

October 30, 2017

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision history

Revision	Date	Description
A	September 2013	Initial release
B	October 2013	Updated the document title
C	February 2014	Update based on KitKat upgrade
D	March 2014	Updated Table 1-1 and Chapter 20
E	May 2014	Updated chapters on Wi-Fi display, 24-bit playback, and DRE and codec sidetone; updated document title; added chapter on hands-free Profile (HFP) Client
F	October 2017	Updated title from *MSM8974 Audio Customization and Debugging* to *Linux Android Audio Customization and Debugging Guide*

Contents

Figures

Tables

1 Introduction

1.1 Purpose

This document provides information on audio debugging and customization for Linux Android. This document is intended for engineers to debug audio issues and customize the Linux Android software.

NOTE: This document describes the dtsi and other files for the MSM89xx and MSM8x26 chipsets. The dtsi and other files for other chipsets follow similar naming conventions.

1.2 Conventions

Function declarations, function names, type declarations, attributes, and code samples appear in a different font, for example, `#include`.

Code variables appear in angle brackets, for example, `<number>`.

Commands to be entered appear in a different font, for example, **`copy a:*.* b:`**.

Button and key names appear in bold font, for example, click **Save** or press **Enter**.

If you are viewing this document using a color monitor, or if you print this document to a color printer, **red boldface** indicates code that is to be **added**, and ~~blue strikethrough~~ indicates code that is to be replaced or removed. Green typeface is either used to emphasize the code or indicates that changes were made in the code.

Shading indicates content that has been added or changed in this revision of the document.

1.3 Technical assistance

For assistance or clarification on information in this document, submit a case to Qualcomm Technologies, Inc. (QTI) at https://createpoint.qti.qualcomm.com/.

If you do not have access to the CDMATech Support website, register for access or send email to support.cdmatech@qti.qualcomm.com.

2 Overview

To debug audio issues, the correct logon is enabled in various modules. Depending on the issue, logs are enabled in one or more of the following areas:

- Kernel logs: These logs are related to audio in files present in <APSS_BUILD>/LINUX/android/kernel.
- User space logs: These logs are related to audio in files that are not present in <APSS_BUILD>/LINUX/android/kernel.
- QXDM Professional™ logging: Using QXDM Professional logging, PCM data in the Hexagon™ processor and log messages related to LPASS are captured. Isolates an issue in Hexagon or in APSS, or can provide other useful information for the analysis of audio issues.

2.1 Audio code structure

2.1.1 User space audio code

The following is a list of directory paths in the user space where the code related to audio is located for the debugging and customization

- <APSS_BUILD>//hardware/qcom/audio/hal/msm8974 – Contains the audio Hardware Abstraction Layer (HAL)-related code
- <APSS_BUILD>//external/tinyalsa/ – Contains the code related to tinymix, tinyplay, and tinycap
- <APSS_BUILD>//hardware/qcom/audio/mm-audio – Contains the implementation of QTI OMX components for the audio encoder and decoders
- <APSS_BUILD>/frameworks/av/media/libstagefright/ – Contains the source code of Stagefright implementation from Google.
- <APSS_BUILD>/frameworks/av/services/audioflinger/ – Contains the source code for AudioFlinger that manages audio streams from the user space
- <APSS_BUILD>/vendor/qcom/proprietary/mm-audio/ – Contains the code related to the Audio Calibration Database (ACDB) driver, parsers for DTS and AC3, surround sound, and so on.
- <APSS_BUILD>/external/bluetooth/bluedroid/ – Contains the code related to Bluetooth A2DP used in a QTI platform; the android_audio_hw.c file contains the A2DP audio HAL implementation
- <APSS_BUILD>/hardware/libhardware/modules/usbaudio/ – Contains the USB HAL implementation for a USB dock use case

- <APSS_BUILD>/vendor/qcom/proprietary/wfd/mm/source/framework/src/ – Contains the Wi-Fi Display (WFD) frameworks-related code; WFDMMSourceAudioSource.cpp configures the RT Proxy port via ALSA APIs and gets the PCM data from the audio layer
- <APSS_BUILD>/system/core/include/system/ – Contains audio.h and audio_policy.h that contain enum definitions and inline functions used all over the code for audio in the user space
- <APSS_BUILD>/frameworks/base/media/java/android/media/ – Contains .java files for audio that expose APIs by Android applications written in Java

2.1.2 Kernel space audio code

The following is a list of directory paths in the kernel where the code related to audio is located for debugging and customization:

- <APSS_BUILD>/kernel/sound/soc/msm/ – Contains the msm8974.c machine driver
- <APSS_BUILD>/kernel/sound/soc/msm/qdsp6v2 – Contains the source code for the platform drivers, front end (FE), and back-end (BE) DAI driver, Hexagon DSP drivers for AFE, ADM, and ASM, voice driver, and so on.
- <APSS_BUILD>/kernel/sound/soc/soc-*.c – All the SoC-*.c files provide information on the ALSA SoCs framework
- <APSS_BUILD>/kernel/drivers/slimbus/ – Contains the source for the SLIMbus driver
- <APSS_BUILD>/kernel/arch/arm/mach-msm/ – Contains some files such as acpuclock-8974.c, board-8974-gpiomux.c, board-8974.c, and clock-8974.c related to the GPIO, clock, and board-specific information on the MSM8974
- <APSS_BUILD>/kernel/arch/arm/mach-msm/qdsp6v2/ – Contains the drivers for DSP-based encoders and decoders, code for the aDSP loader, APR driver, Ion memory driver, and other utility files
- <APSS_BUILD>//LINUX/android/kernel/arch/arm/boot/dts – Contains MSM8974-*.its and MSM8974-*.Dtsi files that contain MSM8974-specific information; audio-related customization is available in files such as MSM8974.dtsi, msm8974-mtp.dtsi, and msm8974-cdp.dtsi
- <APSS_BUILD>/LINUX/android//kernel/sound/soc/codecs/ – Contains the source code for the codec driver for WCD9320; codec driver-related source files are wcd9320.c, wcd9xxx-mbhc.c, wcd9xxx-resmgr.c, wcd9xxx-common.c, and so on.
- <APSS_BUILD>//LINUX/android/kernel/drivers/mfd/ – Contains the source code for the codec driver; wcd9xxx-core.c, wcd9xxx-slimslave.c, and wcd9xxx-irq.c are the codec driver-related files

2.2 Customization guidelines

Table 2-1 Customization guidelines for audio modules

Module	Description	Customization guidelines
Audio policy manager	Manages various input and output device interfaces. Chooses, and defines appropriate routing strategy based on the Stream mode and method. Manages volume/mute settings per stream (as they become active or inactive).	Licensees can modify the device priority and matrix to suit the use case of their choice. ▪ Use the default QTI device configuration and do not deviate from Android Google framework designs. ▪ To reduce the development and testing effort, follow the QTI implementation guidelines.
Audio HAL	The hardware abstraction layer that maps AudioFlinger calls to ASoC drivers	Licensees can modify the audio HAL code when a new device is added.
XML	File contains route- and device-based mixer controls, which are used for setting the audio route and device while starting playback/ recording	Add new route and devices
Machine driver	Specific to the machine implementation or customer board	Machine-specific code is added into this file, for example, adding the external speaker enable/disable sequence or adding the frontend/backend DAI links
Device tree file	Specific to the machine implementation or customer board	Mic bias settings, CAP setting, mic bias voltage settings, MCLK frequency and so on depends on the customer device configuration

2.3 Enabling logs

This section provides information on the log messages used in different file types in the APSS build.

2.3.1 Java file logs

In the .java files, TAG definitions are as follows:

```
private final static String TAG = "Tagname";
```

When a log from the .java file appears in the user space logs, it is prefixed with the tag name mentioned in the file.

The following are various log message types available in the .java files and related examples.

Log message types

```
Log.w(TAG, "message string");   /* Warning  Message*/
Log.e(TAG, "message string");    /* Error Message  */
Log.d(TAG, " message string);    /* Debug Message */
Log.i(TAG, " message string");   /* Information Message */
```

Example log messages

```
Log.w(TAG, "setScreenOnWhilePlaying(true) is ineffective for Surface");
Log.e(TAG, "Error (" + msg.arg1 + "," + msg.arg2 + ")")
Log.d(TAG, "create failed:", ex);
Log.i(TAG, "Info (" + msg.arg1 + "," + msg.arg2 + ")");
```

2.3.2 C/C++ source file logs

Define the following for inclusion in the C++ source files to enable logging:

```
#include <utils/Log.h>
```

The file *system/core/include/cutils/log.h* has the definitions of macros used for logging.

A LOG_TAG is defined in every C++ file that has log messages defined. The LOG_TAG helps in identifying the module that prints the log message, for example, the LOG_TAG in the audio_hw.c file is defined as:

```
#define LOG_TAG "audio_hw_primary"
```

To enable logs in the file, define the following lines:

```
#define LOG_NDEBUG 0
#define LOG_NDDEBUG 0
```

In some files, #define LOG_NDDEBUG 0 is not defined.

To disable logs on the file, the code to enable logs is commented as follows.

```
// #define LOG_NDEBUG 0
// #define LOG_NDDEBUG 0
```

Log message types

```
ALOGE("message");     /* Error Message */
ALOGD("message");     /* Debug Message */
ALOGI("message");     /* Information message */
ALOGV("message");     /* Verbose Message */
ALOGW("message");     /* Warning Message */
```

2.3.3 C files log in kernel

To enable dynamic logging in some audio-related files in the kernel, the following commands may be used:

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo -n "file FILENAME +p" > /sys/kernel/debug/dynamic_debug/control
```

Examples for filenames include msm-pcm-q6.c, q6afe.c, and so on.

To disable the logs, use the **–p** option. The following command describes the usage:

```
echo -n "file FILENAME -p" > /sys/kernel/debug/dynamic_debug/control
```

The following are various log message types available in the C files for audio in the kernel and related examples.

Log message types

```
pr_debug("message%d … \n",var1,…);   /* Debug Message  */
pr_info("message%d … \n",var1,…);    /* Info Message */
pr_err("message%d … \n",var1,…);    /* Error Message */
```

Example log messages

```
pr_debug("Buffer Consumed = 0x%08x\n", *ptrmem);
pr_info("%s: CMD Format block failed\n", __func__);
pr_err("Failed to allocate memory for msm_audio\n");
```

3 Audio bringup

Audio bringup is the verification of audio paths, such as:

- Playback of audio over the earpiece, headset, and speaker
- stereo lineout and differential lineout
- Recording using handset and headset mics
- Analog and digital mic
- Recording via ANC mics, if used for the platform
- Loopbacks – Codec, AFE, and ALSA
- HDMI playback
- Headset insertion/removal and button press/release
- MI2S interface
- PCM interface

3.1 Log collection

Enable logs on the following files at compile time by defining #define DEBUG at the beginning of the file before any #include <...>:

- wcd9320.c
- soc-dapm.c
- msm8974.c
- msm-pcm-routing-v2.c

The logs must enable the following files based on the use case scenario:

- For debugging codec-related issues during audio bringup:
  - wcd9xxx-core.c
  - wcd9xxx-irq.c
  - wcd9xxx-mbhc.c
  - wcd9xxx-resmgr.c
- For debugging HDMI-related issues during audio bringup:
  - msm-dai-q6-hdmi-v2.c
  - q6afe.c

- For debugging AUX PCM-related issues during audio bringup:
  - msm-dai-q6-v2.c
- For debugging USB audio and WFD-related issues during audio bringup:
  - msm-pcm-afe-v2.c

3.2 Debugging checkpoints

During initial audio bringup, the following is verified:

- Check whether DSP is up
- Check whether SLIMbus is up
- Check whether the sound card is registered
- Verify playback and recording using tinymix, tinycap, and tinyplay commands

3.2.1 Verify DSP

In the bootup logs, the following message is displayed when the DSP has crashed or an aDSP image is loaded.

```
adsp_loader_probe: Q6/ADSP image is loaded
```

The aDSP image is loaded successfully by the Peripheral Image Loader (PIL).

Check whether the module adsp-loader.ko is present in the /system/lib/modules folder.

Frequently, it is observed that aDSP images are not present in the device file system at /firmware/image/adsp*. In this case, the message is not displayed.

On MSM8974 KitKat, the aDSP image is loaded statically. aDSP images require proper configuration in the source code to load to RAM.

The OEM must add the following line to their init.target.rc:

```
write /sys/kernel/boot_adsp/boot 1
```

The config macro CONFIG_MSM_ADSP_LOADER should be present in /kernel/arch/arm/configs/msm8974_defconfig.

```
 ..
CONFIG_MSM_ADSP_LOADER=y
```

Also, the following code should be present in the arch/arm/mach-msm/Kconfig file:

```
config MSM_ADSP_LOADER
        tristate "ADSP loader support"
        select SND_SOC_MSM_APRV2_INTF
   depends on MSM_AUDIO_QDSP6V2
        help
          Enable ADSP image loader.
     The ADSP loader brings ADSP out of reset
     for the platforms that use APRv2.
     Say M if you want to enable this module.
```

The following message indicates that the PIL has successfully loaded the DSP image.

```
<6>[    6.410851] apr_tal:Q6 Is Up
```

The SMD driver finds services in the aDSP so that the APR driver is notified of DSP service availability. The SLIMbus driver should detect the aDSP state as ready.

When the DSP image is loaded, the kernel should call the probe function of the SLIMbus driver again, and the SLIMbus driver should detect the ADSP state as ready.

NOTE: See solution# 00027452 Analyzing No Sound Card issue in MSM8x74 and MSM8x26 Android builds at https://createpoint.qti.qualcomm.com/.

If the DSP is not up, check whether the PIL is loading the DSP firmware image.

If the DSP crashes, collect the RAM dumps and contact the Customer Engineering (CE) team to analyze.

3.2.2 Verify SLIMbus

The following message indicates that the APSS is informed about the SLIMbus master capability of the aDSP. If this message is not seen, there may be an issue with the SLIMbus driver on the aDSP or APSS.

```
<6>[    6.554372] SLIM SAT: Received master capability
```

QXDM Professional logs are collected to help debug the issue further.

```
<6>[    6.653368] taiko-slim taiko-slim-pgd: wcd9xxx_check_codec_type:
detected taiko_codec, major 0x102, minor 0x1, ver 0x2
```

This message indicates that the codec has reported its presence and that it was correctly reset to perform normal operation. If the message is not displayed, there could be an issue with the codec not powered up properly.

3.2.3 Verify sound card registration

Once the DAI links are mapped, the sound card gets enumerated.

```
<6>[    7.213775] asoc: msm-stub-rx <-> msm-dai-q6.4106 mapping ok
<6>[    7.219424] asoc: msm-stub-tx <-> msm-dai-q6.4107 mapping ok
<6>[    7.224997] asoc: msm-stub-rx <-> msm-dai-q6.4108 mapping ok
<6>[    7.230725] asoc: msm-stub-tx <-> msm-dai-q6.4109 mapping ok
<6>[    7.236235] msm_audrx_init(), dev_namemsm-dai-q6-dev.16384
```

The sound card is available at /proc/asound/cards.

3.2.4 Verify playback and recording using tinymix commands

Different audio loopback verification methods on MSM8974 Linux Android platform are referred for tinymix commands to verify the playback and recording.

If there are any playback and recording issues, the following kernel logs are enabled:

```
echo -n "file wcd9320.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm8974.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file soc-dapm.c +p" > /sys/kernel/debug/dynamic_debug/control
```

To obtain more information, messages in the following files are enabled.

```
echo -n "file wcd9320.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm8974.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file soc-dapm.c +p" > /sys/kernel/debug/dynamic_debug/control
```

Depending on the audio hardware interface used, different CLKs is checked on the oscilloscope to confirm the following:

- Whether MCLK is present and is set to the correct frequency
- For the SLIMbus interface, check whether the SLIMbus clock is enabled
- For the MI2S interface, check whether the MI2S bit clock and Word select clock are running at the correct frequency and the data line has data
- For the PCM interface, check whether the PCM clock and PCM Sync are running at the correct frequency and the data in and data out are changing

3.3 Device tree customization

Device tree files are found in the kernel/arch/arm/boot/dts/msm8974.dtsi file. The OEM modifies the device tree files as per the audio hardware configurations.

3.3.1 Mic configurations

The following code represents audio routing configurations in the device tree file.

```
sound {
   compatible = "qcom,msm8974-audio-taiko";
   qcom,model = "msm8974-taiko-snd-card";

   qcom,audio-routing =
        "RX_BIAS", "MCLK",
        "LDO_H", "MCLK",
        "AIF4 MAD", "MCLK",
        "AMIC1", "MIC BIAS1 Internal1",
        "MIC BIAS1 Internal1", "Handset Mic",
        "AMIC2", "MIC BIAS2 External",
        "MIC BIAS2 External", "Headset Mic",
        "AMIC3", "MIC BIAS2 External",
        "MIC BIAS2 External", "ANCRight Headset Mic",
        "AMIC4", "MIC BIAS2 External",
        "MIC BIAS2 External", "ANCLeft Headset Mic",
        "DMIC1", "MIC BIAS1 External",
        "MIC BIAS1 External", "Digital Mic1",
        "DMIC2", "MIC BIAS1 External",
        "MIC BIAS1 External", "Digital Mic2",
        "DMIC3", "MIC BIAS3 External",
        "MIC BIAS3 External", "Digital Mic3",
        "DMIC4", "MIC BIAS3 External",
        "MIC BIAS3 External", "Digital Mic4",
        "DMIC5", "MIC BIAS4 External",
        "MIC BIAS4 External", "Digital Mic5",
        "DMIC6", "MIC BIAS4 External",
        "MIC BIAS4 External", "Digital Mic6";
```

If the OEM reuses the QTI .dtsi file in kernel/arch/arm/boot/dts/msm8974.dtsi, the OEM may have access to the .dtsi file. Thus, the following changes are made in the appropriate .dtsi file. If you are viewing this document using a color monitor, or if you print this document to a color printer, **red boldface** indicates code that is to be **added**.

```
sound {
   compatible = "qcom,msm8974-audio-taiko";
   qcom,model = "msm8974-taiko-snd-card";
```

```
qcom,audio-routing =
     "RX_BIAS", "MCLK",
     "LDO_H", "MCLK",
     "AIF4 MAD", "MCLK",
     "AMIC1", "MIC BIAS1 External",
     "MIC BIAS1 External", "Primary Mic",
     "AMIC2", "MIC BIAS2 External",
     "MIC BIAS2 External", "Headset Mic",
     "AMIC3", "MIC BIAS1 External",
     "MIC BIAS1 External", "Secondary Mic",
     "DMIC1", "MIC BIAS1 External",
     "MIC BIAS1 External", "Digital Mic1",
     "DMIC2", "MIC BIAS1 External",
     "MIC BIAS1 External", "Digital Mic2",
     "DMIC3", "MIC BIAS3 External",
     "MIC BIAS3 External", "Digital Mic3",
     "DMIC4", "MIC BIAS3 External",
     "MIC BIAS3 External", "Digital Mic4",
     "DMIC5", "MIC BIAS4 External",
     "MIC BIAS4 External", "Digital Mic5",
     "DMIC6", "MIC BIAS4 External",
     "MIC BIAS4 External", "Digital Mic6";
```

The following changes mentioned are applied in the /kernel/sound/soc/msm/msm8974.c machine driver file if the OEM uses the QTI reference machine driver or the OEM owns the machine driver file.

```
static const struct snd_soc_dapm_widget msm8974_dapm_widgets[] = {

   SND_SOC_DAPM_SUPPLY("MCLK",  SND_SOC_NOPM, 0, 0,
   msm8974_mclk_event, SND_SOC_DAPM_PRE_PMU | SND_SOC_DAPM_POST_PMD),

   SND_SOC_DAPM_SPK("Lineout_1 amp", msm_ext_spkramp_event),
   SND_SOC_DAPM_SPK("Lineout_3 amp", msm_ext_spkramp_event),

   SND_SOC_DAPM_SPK("Lineout_2 amp", msm_ext_spkramp_event),
   SND_SOC_DAPM_SPK("Lineout_4 amp", msm_ext_spkramp_event),
   SND_SOC_DAPM_SPK("SPK_ultrasound amp",
                    msm_ext_spkramp_ultrasound_event),

   SND_SOC_DAPM_MIC("Primary Mic", NULL),
   SND_SOC_DAPM_MIC("Headset Mic", NULL),
   SND_SOC_DAPM_MIC("Secondary Mic", NULL),
   SND_SOC_DAPM_MIC("Analog Mic4", NULL),
   SND_SOC_DAPM_MIC("Analog Mic6", NULL),
   SND_SOC_DAPM_MIC("Analog Mic7", NULL),
     . . .
```

3.3.2 MCLK clock

The MSM/PMIC drives the master clock. This clock is needed for the codec (WCD9320).

MCLK clock frequency and clock source are defined in the device tree.

```
qcom,cdc-mclk-gpios = <&pm8941_gpios 15 0>;
qcom,taiko-mclk-clk-freq = <9600000>;
```

3.3.3 Codec reset GPIO

The WCD9320 GPIO is used for codec SoC reset. It is configured as per the hardware design of the OEMs. In the following example, the GPIO pin is set to 63.

```
qcom,cdc-reset-gpio = <&msmgpio 63 0>;
```

3.3.4 Mic bias settings

The OEM can change the voltage of each mic bias by selecting the mic bias cfilt settings.

LDOH output in volts should be 1.95 V and 3.00 V:

```
qcom,cdc-micbias-ldoh-v = <0x3>;
```

cfilt voltage is set to a maximum of qcom,cdc-micbias-ldoh-v – 0.15 V:

```
qcom,cdc-micbias-cfilt1-mv = <1800>;
qcom,cdc-micbias-cfilt2-mv = <2700>;
qcom,cdc-micbias-cfilt3-mv = <1800>;
```

Use the following cfilt for each mic bias:

```
qcom,cdc-micbias1-cfilt-sel = <0x0>;
qcom,cdc-micbias2-cfilt-sel = <0x1>;
qcom,cdc-micbias3-cfilt-sel = <0x2>;
qcom,cdc-micbias4-cfilt-sel = <0x2>;
```

3.3.5 Codec IRQ settings

The WCD9320 codec interrupt pin is configured as per the hardware design of the OEM. In the following example, the interrupt pin is configured to GPIO pin 72.

```
   wcd9xxx_intc: wcd9xxx-irq {
        compatible = "qcom,wcd9xxx-irq";
        interrupt-controller;
        #interrupt-cells = <1>;
```

```
                interrupt-parent = <&msmgpio>;
                interrupts = <72 0>;
                interrupt-names = "cdc-int";
        };
```

4 Audio-related system properties

There are many system properties that are configured by changing the system.prop file located in the /device/qcom/<chipset>/ directory. Run the following commands to change the properties.

```
adb shell setprop  <system_property_name> <value>
```

Run the command to check the value of the property.

```
adb shell getprop  <system_property_name>
```

The audio-related system properties are given in Table 4-1.

Table 4-1 Audio-related system properties

Property	Definition
ro.qc.sdk.audio.ssr	If true, the surround sound recording feature is supported; used in the function audio_extn_ssr_update_enabled() in the file hardware/qcom/audio/hal/audio_extn/ssr.c
ro.qc.sdk.audio.fluencetype=none	▪ None: Qualcomm® Fluence™ noise cancellation technology is not supported ▪ Fluence Pro: Qualcomm® Fluence™ Pro noise cancellation technology, Fluence Quad-Mic, and Dual-Mic are supported ▪ Fluence: Fluence Dual-Mic is supported
persist.audio.fluence.voicecall=true	If true, use Fluence in voice call
persist.audio.fluence.voicerec=false	If true, use Fluence during voice recording
persist.audio.fluence.speaker=true	If true, use Fluence in Speaker mode
tunnel.audio.encode=true	If true, enables Tunnel mode encoding; currently only AMR_WB format is supported; used in the function audio_extn_compr_cap_enabled() in the file /hardware/qcom/audio/hal/audio_extn/compress_capture.c

Property	Definition

af.resampler.quality=4	Used in the function void AudioResampler::init_routine() in the file frameworks/av/services/audioflinger/AudioResampler.cpp, values 0, 1, 2, 3, or 4; the meaning of the values is based on an enum defined in /frameworks/av/services/audioflinger/ AudioResampler.h.
	
	`    // Determines quality of SRC.`
	`    //  LOW_QUALITY: linear interpolator (first order)`
	`    //  MED_QUALITY: cubic interpolator (third order)`
	`    //  HIGH_QUALITY: fixed multitap FIR (for example, 48 kHz->44.1 kHz)`
	`    // NOTE: high-quality SRC will only be supported for`
	`    // certain fixed rate conversions. Sample rate cannot be`
	`    // changed dynamically.`
	`    enum src_quality {`
	`        DEFAULT_QUALITY=0,`
	`        LOW_QUALITY=1,`
	`        MED_QUALITY=2,`
	`        HIGH_QUALITY=3,`
	`        VERY_HIGH_QUALITY=4,`
	`    };`

audio.offload.buffer.size.kb=32	Value is an integer that defines the size (in kB) of the buffers used in compress offload playback; used in the function static uint32_t get_offload_buffer_size() defined in the file /hardware/qcom/audio/ hal/audio_hw.c
av.offload.enable	If true, the compress offload driver is used for supported audio formats in audio/video playback. It is used in the function AudioPolicyManagerBase::isOffloadSupported() in the file */hardware/libhardware_legacy/audio/AudioPolicyManagerBase.cpp*
use.voice.path.for.pcm.voip=true	If true, use the voice path in DSP for VoIP call-in PCM format. It is used in the following functions: ▪ voice_extn_compress_voip_pcm_prop_check() in the file hardware/qcom/audio/hal/ voice_extn/ compress_voip.c ▪ status_t AudioTrack::set() in the file frameworks/av/media/libmedia/AudioTrack.cpp
audio.offload.gapless.enabled=false	If true, sets the gapless mode in the kernel. It is used in the function static int set_gapless_mode(struct audio_device *adev) in the file /hardware/qcom/audio/hal/audio_hw.c
persist.speaker.prot.enable=false	If true, speaker protection is enabled. It is used in the function audio_extn_spkr_prot_init(void *adev) in the file /hardware/qcom/audio/hal/audio_extn/spkr_protection.c
qcom.hw.aac.encoder	If true, use QCOM DSP-based AAC encoder for encoding. It is used in the function bool ExtendedUtils::UseQCHWAACEncoder() in the file /frameworks/av/media/libstagefright/ExtendedUtils.cpp
audio.offload.disable	If true, Compress Offload mode is disabled. It is used in the function bool AudioPolicyManagerBase::isOffloadSupported() in the file /hardware/libhardware_legacy/audio/AudioPolicyManagerBase.cpp

Property	Definition
persist.headset.anc.type	▪ Feedforward – Use Feedforward ANC headset ▪ Feedback – Use feedback ANC headset Used in the function bool audio_extn_should_use_fb_anc(void) in the file /hardware/qcom/audio/hal/audio_extn/audio_extn.c
ssr.pcmdump	If true, enable PCM dump collection during surround sound recording; depending on the number of channels, the following files are created: ▪ For four channels, “/data/4ch.pcm” is created ▪ For six channels, “/data/6ch.pcm” is created This property is used in the function int32_t audio_extn_ssr_init() in the file /hardware/qcom/audio/hal/audio_extn/ssr.c
persist.debug.sf.noaudio	If true, audio is disabled in audio/video recording or playback; used in the function bool ExtendedUtils::ShellProp::isAudioDisabled() in the file /frameworks/av/media/libstagefright/ExtendedUtils.cpp
media.aac_51_output_enabled	If true or “1,” then downmixing of multichannel AAC to stereo is enabled; used in the function void SoftAAC2::maybeConfigureDownmix() in the file /frameworks/av/media/libstagefright/codecs/aacdec/SoftAAC2.cpp
media.wfd.use-pcm-audio	If true or “1,” then use PCM format in WFD audio playback; used in the function WifiDisplaySource::onReceiveM3Response() in the file frameworks/av/media/libstagefright/wifi-display/source/ WifiDisplaySource.cpp

5 Audio device calibration

To add device with new calibration data:

1. Add the device entry in the ACDB file and update the calibration data for the device.
2. If the new device is a Tx or an Rx that is chosen in a voice call, VoIP, or a VoLTE call, there must be an appropriate device pair added in the ACDB file.
3. Add the mixer control definition and path definition to the mixer path XML file with the required sequence.
4. Add an entry to device_table in hardware/qcom/audio/hal/msm8974/platform.h and platform.c.
5. Include the ACDB device ID for the device in acdb_device_table in hardware/qcom/audio/hal/msm8974/platform.c.
6. The device name mentioned in the device_table is mapped to the Android device. A change in the platform_get_output_snd_device () function in the hardware/qcom/audio/hal/msm8974/platform.c file is required.

5.1 Device table

In the hardware/qcom/audio/hal/msm8974/platform.h file, the sound device enum contains the list of sound input/output device names mapped to corresponding mixer paths.

```
/* Sound devices specific to the platform
 * The DEVICE_OUT_* and DEVICE_IN_* should be mapped to these sound
 * devices to enable corresponding mixer paths
 */
enum {
    SND_DEVICE_NONE = 0,

    /* Playback devices */
    SND_DEVICE_MIN,
    SND_DEVICE_OUT_BEGIN = SND_DEVICE_MIN,
    SND_DEVICE_OUT_HANDSET = SND_DEVICE_OUT_BEGIN,
    SND_DEVICE_OUT_SPEAKER,
    SND_DEVICE_OUT_SPEAKER_REVERSE,
    SND_DEVICE_OUT_HEADPHONES,
    SND_DEVICE_OUT_SPEAKER_AND_HEADPHONES,
    SND_DEVICE_OUT_VOICE_HANDSET,
    SND_DEVICE_OUT_VOICE_SPEAKER,
    SND_DEVICE_OUT_VOICE_HEADPHONES,
```

In the hardware/qcom/audio/hal/msm8974/platform.c file, device_table defines each device string name.

```
static const char * const device_table[SND_DEVICE_MAX] = {
    [SND_DEVICE_NONE] = "none",
    /* Playback sound devices */
    [SND_DEVICE_OUT_HANDSET] = "handset",
    [SND_DEVICE_OUT_SPEAKER] = "speaker",
    [SND_DEVICE_OUT_SPEAKER_REVERSE] = "speaker-reverse",
    [SND_DEVICE_OUT_HEADPHONES] = "headphones",
    [SND_DEVICE_OUT_SPEAKER_AND_HEADPHONES] = "speaker-and-headphones",
    [SND_DEVICE_OUT_VOICE_HANDSET] = "voice-handset",
    [SND_DEVICE_OUT_VOICE_SPEAKER] = "voice-speaker",
    [SND_DEVICE_OUT_VOICE_HEADPHONES] = "voice-headphones",
    [SND_DEVICE_OUT_HDMI] = "hdmi",
    [SND_DEVICE_OUT_SPEAKER_AND_HDMI] = "speaker-and-hdmi",
    [SND_DEVICE_OUT_BT_SCO] = "bt-sco-headset",
    [SND_DEVICE_OUT_BT_SCO_WB] = "bt-sco-headset-wb",
```

…

5.2 Mixer path XML

The mixer path XML contains the list mixer control definition and the mixer control path.

Based on the mixer path XML setting, the libaudioroute library module from Google gets up the route for each stream mixer controls.

Mixer path XML is located in the device/qcom/msm8974/mixer_paths.xml.

```
<mixer>
    <!-- These are the initial mixer settings -->
    <ctl name="Voice Rx Device Mute" id="0" value="0" />
    <ctl name="Voice Rx Device Mute" id="1" value="-1" />
    <ctl name="Voice Rx Device Mute" id="2" value="20" />
    <ctl name="Voice Tx Mute" id="0" value="0" />
    <ctl name="Voice Tx Mute" id="1" value="-1" />
    <ctl name="Voice Tx Mute" id="2" value="500" />
    <ctl name="Voice Rx Gain" id="0" value="0" />
    <ctl name="Voice Rx Gain" id="1" value="-1" />
    <ctl name="Voice Rx Gain" id="2" value="20" />
    <ctl name="Voip Tx Mute" id="0" value="0" />
    <ctl name="Voip Tx Mute" id="1" value="500" />
    <ctl name="Voip Rx Gain" id="0" value="0" />
    <ctl name="Voip Rx Gain" id="1" value="20" />
    <ctl name="Voip Mode Config" value="12" />
    <ctl name="Voip Rate Config" value="0" />
    <ctl name="Voip Evrc Min Max Rate Config" id="0" value="1" />
    <ctl name="Voip Evrc Min Max Rate Config" id="1" value="4" />
    <ctl name="Voip Dtx Mode" value="0" />
    <ctl name="TTY Mode" value="Off" />
    <ctl name="LINEOUT1 Volume" value="13" />
    <ctl name="LINEOUT2 Volume" value="13" />
    <ctl name="LINEOUT3 Volume" value="13" />
    <ctl name="LINEOUT4 Volume" value="13" />
```

…

5.2.1 Mixer path definition

5.2.1.1 Audio route (FE to BE)-specific mixer settings

```
<!-- These are audio route (FE to BE) specific mixer settings -->
    <path name="deep-buffer-playback">
        <ctl name="SLIMBUS_0_RX Audio Mixer MultiMedia1" value="1" />
    </path>

    <path name="deep-buffer-playback hdmi">
        <ctl name="HDMI Mixer MultiMedia1" value="1" />
    </path>

    <path name="deep-buffer-playback speaker-and-hdmi">
        <path name="deep-buffer-playback hdmi" />
        <path name="deep-buffer-playback" />
    </path>

    <path name="deep-buffer-playback bt-sco">
        <ctl name="INTERNAL_BT_SCO_RX Audio Mixer MultiMedia1" value="1" />
    </path>

    <path name="deep-buffer-playback bt-sco-wb">
        <ctl name="Internal BTSCO SampleRate" value="16000" />
        <path name="deep-buffer-playback bt-sco" />
    </path>

    <path name="deep-buffer-playback afe-proxy">
        <ctl name="AFE_PCM_RX Audio Mixer MultiMedia1" value="1" />
    </path>

    <path name="low-latency-playback">
        <ctl name="SLIMBUS_0_RX Audio Mixer MultiMedia5" value="1" />
    </path>
```

5.2.1.2 Sound device-specific mixer settings

```
<!-- These are actual sound device specific mixer settings -->
    <path name="adc1">
        <ctl name="AIF1_CAP Mixer SLIM TX7" value="1"/>
        <ctl name="SLIM_0_TX Channels" value="One" />
        <ctl name="SLIM TX7 MUX" value="DEC6" />
        <ctl name="DEC6 MUX" value="ADC1" />
        <ctl name="IIR1 INP1 MUX" value="DEC6" />
    </path>

    <path name="adc2">
        <ctl name="AIF1_CAP Mixer SLIM TX7" value="1"/>
        <ctl name="SLIM_0_TX Channels" value="One" />
        <ctl name="SLIM TX7 MUX" value="DEC5" />
        <ctl name="DEC5 MUX" value="ADC2" />
        <ctl name="IIR1 INP1 MUX" value="DEC5" />
    </path>

    <path name="dmic1">
        <ctl name="AIF1_CAP Mixer SLIM TX7" value="1"/>
        <ctl name="SLIM_0_TX Channels" value="One" />
        <ctl name="SLIM TX7 MUX" value="DEC7" />
        <ctl name="DEC7 MUX" value="DMIC1" />
        <ctl name="IIR1 INP1 MUX" value="DEC7" />
    </path>

    <path name="speaker">
        <ctl name="SLIM RX1 MUX" value="AIF1_PB" />
        <ctl name="RX1 MIX1 INP1" value="RX1" />
        <ctl name="CLASS_H_DSM MUX" value="DSM_HPHL_RX1" />
        <ctl name="DAC1 Switch" value="1" />
    </path>
```

5.2.1.3 Volume control through mixer_path.xml

1. WCD codec digital gain settings:
   - RXx Digital Volume – RXx_VOL_CTL_B2_CTL x = [1 -7]
   - IIRn INPx Volume – IIR1_GAIN_Bx_CTL x = [1 -4]

   The gains are set from 0 to 124 as "value" as shown in the following command, where "0" means -84 dB, "84" means 0 dB, and "124" means +40 dB.

```
<ctl name="DEC1 Volume" value="84" />
```

2. WCD codec analog gain settings:
   - ADCx Volume – TX_x_GAIN x = [1,2]
   - LINEOUTx Volume – RX_LINE_x_GAIN x = [1-7]
   - HPHL Volume – RX_HPH_L_GAIN
   - HPHR Volume – RX_HPH_R_GAIN
   - EAR PA Gain – RX_EAR_GAIN
   - SPK DRV Volume – SPKR_DRV_GAIN

   For the register value to reflect the proper gain steps, see *WCD9320 Audio Codec Hardware Register Description for OEMs* (80-NA556-2).

   For example, use the following mixer control to set the ADC1 Volume to +28.5 dB, which maps to value 19:

```
<ctl name="ADC1 Volume" value="19" />
```

   For example, the codec register value [6:2] is set to 0x13 (19 in decimal) as shown in Table 5-1.

Table 5-1 TX_1_GAIN

Bits	Name	Description
7	CH1_EN	Tx path 1 enable ▪ 0 – Disable ▪ 1 – Enable
6:2	CH1_GAIN	Tx path 1 analog gain ▪ 0x0 – 0 dB ▪ 0x1 – +1.5 dB ▪ 0x2 – +3 dB ▪ 0x3 – +4.5 dB ▪ 0x4 – +6 dB ▪ 0x5 – +7.5 dB ▪ 0x6 – +9 dB ▪ 0x7 – +10.5 dB ▪ 0x8 – +12 dB ▪ 0x9 – +13.5 dB ▪ 0xA – +15 dB ▪ 0xB – +16.5 dB ▪ 0xC – +18 dB ▪ 0xD – +19.5 dB ▪ 0xE – +21 dB ▪ 0xF – +22.5 dB ▪ 0x10 – +24 dB ▪ 0x11 – +25.5 dB ▪ 0x12 – +27 dB ▪ 0x13 – +28.5 dB Other settings not supported

Bits	Name	Description
1	CH1_IB_SEL	Ch1 current control ▪ 0x0 – For all gain settings ▪ 0x1 – For gains setting from 24 dB to 28.5 dB
0	RESERVED	Reserved

Use the following mixer control to set RX_LINE_1_GAIN to 0 dB, which maps to value 20.

```
<ctl name="RX_LINE_1_GAIN" value="20" />
```

Reverse mapping occurs when 0 dB is the maximum gain that is set to RX_LINE_1_GAIN, with the minimum gain set to -30 dB.

For this example, the codec register value [4:0] is set to 0x14 (20 in decimal) as shown in Table 5-2.

Table 5-2 RX_LINE_1_GAIN

Bits	Name	Description
7:6	RESERVED	Reserved
5	GAIN_SOURCE_SEL	▪ 0 – Compander ▪ 1 – Register
4:0	PA_GAIN	▪ 0x0 – NEG_0P0_DB ▪ 0x1 – NEG_1P5_DB ▪ 0x2 – NEG_3P0_DB ▪ 0x3 – NEG_4P5_DB ▪ 0x4 – NEG_6P0_DB ▪ 0x5 – NEG_7P5_DB ▪ 0x6 – NEG_9P0_DB ▪ 0x7 – NEG_10P5_DB ▪ 0x8 – NEG_12P0_DB ▪ 0x9 – NEG_13P5_DB ▪ 0xA – NEG_15P0_DB ▪ 0xB – NEG_16P5_DB ▪ 0xC – NEG_18P0_DB ▪ 0xD – NEG_19P5_DB ▪ 0xE – NEG_21P0_DB ▪ 0xF – NEG_22P5_DB ▪ 0x10 – NEG_24P0_DB ▪ 0x11 – NEG_25P5_DB ▪ 0x12 – NEG_27P0_DB ▪ 0x13 – NEG_28P5_DB ▪ 0x14 – NEG_30P0_DB ▪ Others – Reserved

5.3 ACDB device ID table

In Android KitKat, ACDB device ID information no longer includes the UCM replacement mixer path XML. The tinyALSA audio HAL maintains the ACDB ID information.

acdb_device_table is located in the hardware/qcom/audio/hal/msm8974/platform.c and defines the ACDB ID for each device.

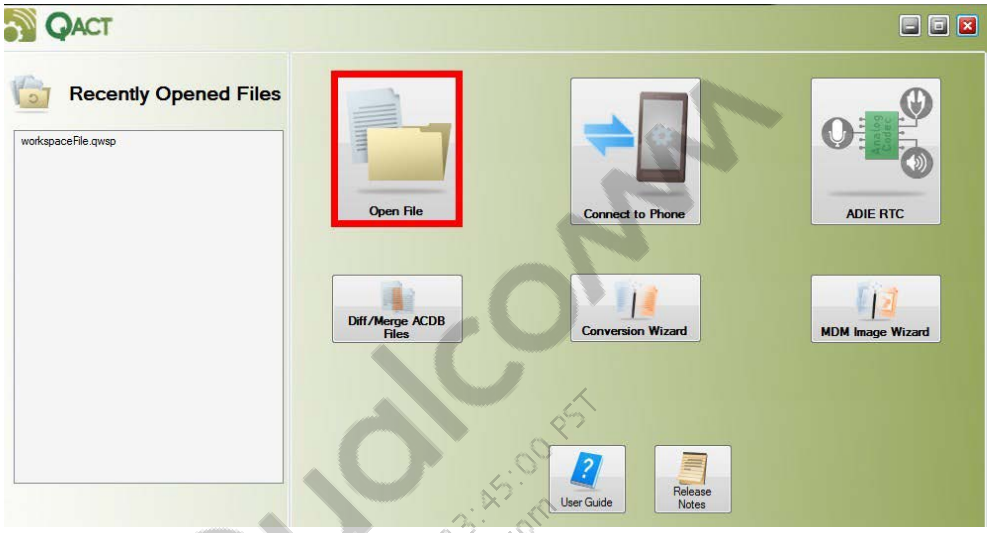
```
/* ACDB IDs (audio DSP path configuration IDs) for each sound device */
static const int acdb_device_table[SND_DEVICE_MAX] = {
    [SND_DEVICE_NONE] = -1,
    [SND_DEVICE_OUT_HANDSET] = 7,
    [SND_DEVICE_OUT_SPEAKER] = 14,
    [SND_DEVICE_OUT_SPEAKER_REVERSE] = 14,
    [SND_DEVICE_OUT_HEADPHONES] = 10,
    [SND_DEVICE_OUT_SPEAKER_AND_HEADPHONES] = 10,
    [SND_DEVICE_OUT_VOICE_HANDSET] = 7,
    [SND_DEVICE_OUT_VOICE_SPEAKER] = 14,
    [SND_DEVICE_OUT_VOICE_HEADPHONES] = 10,
    [SND_DEVICE_OUT_HDMI] = 18,
    [SND_DEVICE_OUT_SPEAKER_AND_HDMI] = 14,
    [SND_DEVICE_OUT_BT_SCO] = 22,
    [SND_DEVICE_OUT_BT_SCO_WB] = 39,
    [SND_DEVICE_OUT_VOICE_TTY_FULL_HEADPHONES] = 17,
    [SND_DEVICE_OUT_VOICE_TTY_VCO_HEADPHONES] = 17,
    [SND_DEVICE_OUT_VOICE_TTY_HCO_HANDSET] = 37,
    [SND_DEVICE_OUT_AFE_PROXY] = 0,
    [SND_DEVICE_OUT_USB_HEADSET] = 0,
    [SND_DEVICE_OUT_SPEAKER_AND_USB_HEADSET] = 14,
    [SND_DEVICE_OUT_TRANSMISSION_FM] = 0,
    [SND_DEVICE_OUT_ANC_HEADSET] = 26,
    [SND_DEVICE_OUT_ANC_FB_HEADSET] = 27,
    [SND_DEVICE_OUT_VOICE_ANC_HEADSET] = 26,
    [SND_DEVICE_OUT_VOICE_ANC_FB_HEADSET] = 27,
    [SND_DEVICE_OUT_SPEAKER_AND_ANC_HEADSET] = 26,
    [SND_DEVICE_OUT_ANC_HANDSET] = 103,
    [SND_DEVICE_OUT_SPEAKER_PROTECTED] = 101,
```

5.4 Add a device

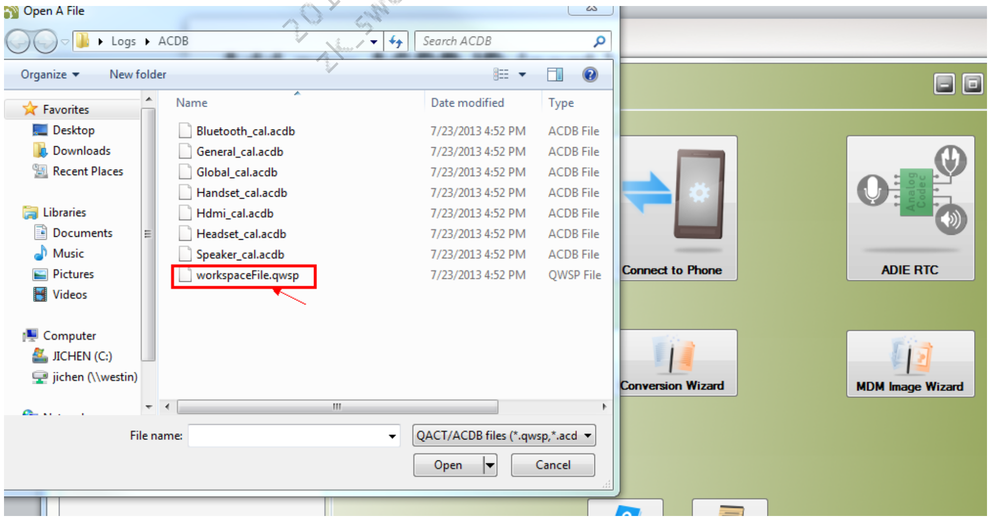
5.4.1 Create ACDB device ID and calibration data

To add a device file to the ACDB in QACT™ platform Offline mode:

1. Open QACT and click **Open File**.

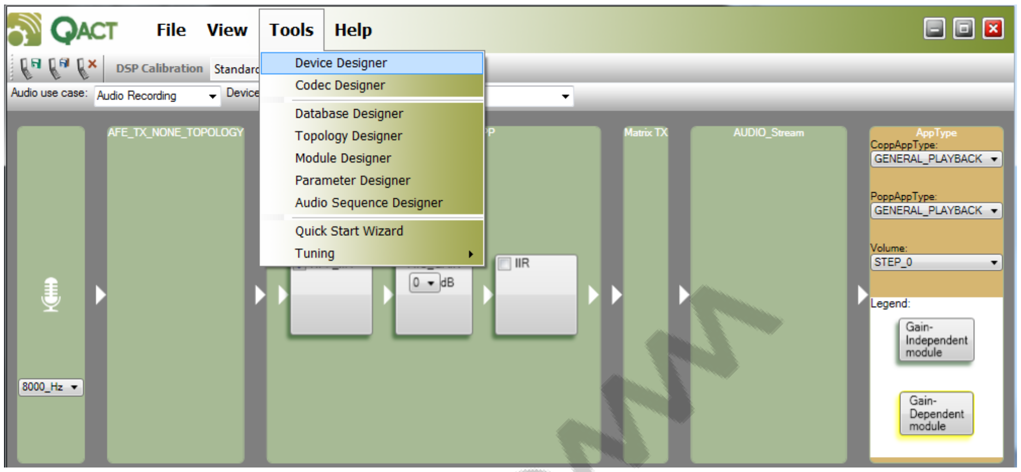


2. Select **workspaceFile.qwsp** previously stored under vendor\qcom\proprietary\mm-audio\audcal\family-b\acdbdata\8974 and open the file.

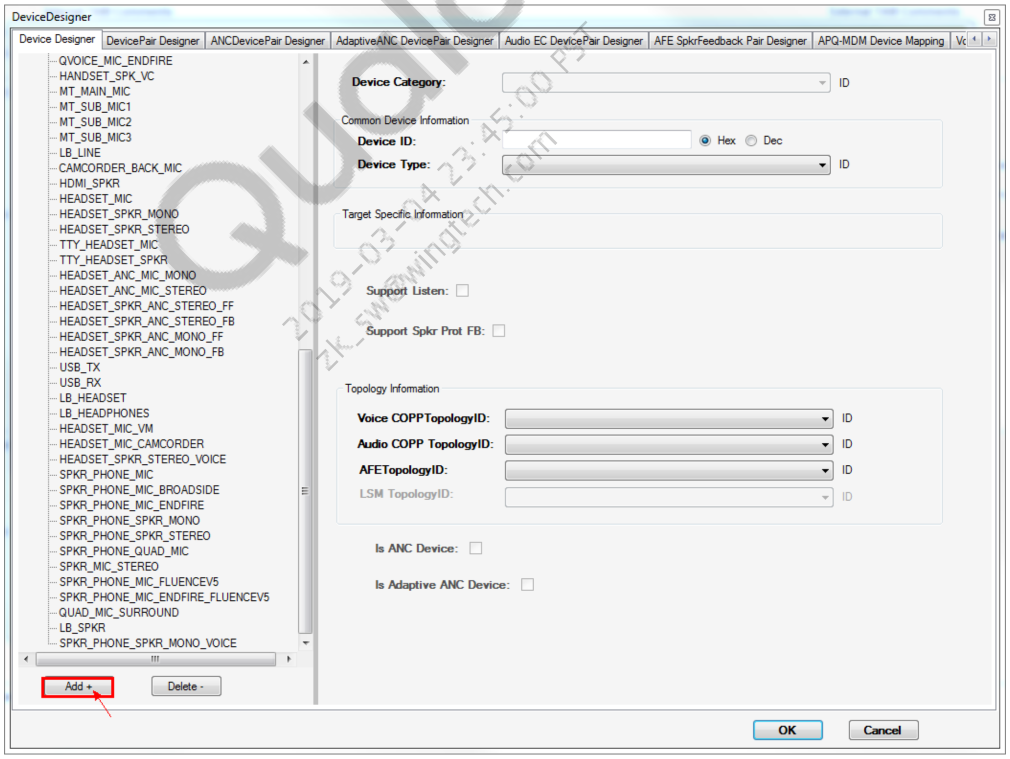


Name	Date modified	Type
Bluetooth_cal.acdb	7/23/2013 4:52 PM	ACDB File
General_cal.acdb	7/23/2013 4:52 PM	ACDB File
Global_cal.acdb	7/23/2013 4:52 PM	ACDB File
Handset_cal.acdb	7/23/2013 4:52 PM	ACDB File
Hdmi_cal.acdb	7/23/2013 4:52 PM	ACDB File
Headset_cal.acdb	7/23/2013 4:52 PM	ACDB File
Speaker_cal.acdb	7/23/2013 4:52 PM	ACDB File
workspaceFile.qwsp	7/23/2013 4:52 PM	QWSP File

3. Click **Tools>Device Designer**.



4. When the Device Designer window opens, click **Add+** to add the new ACDB device.


DeviceDesigner

Device Designer | DevicePair Designer | ANCDevicePair Designer | AdaptiveANC DevicePair Designer | Audio EC DevicePair Designer | AFE SpkrFeedback Pair Designer | APQ-MDM Device Mapping | Vc

QVOICE_MIC_ENDFIRE
HANDSET_SPK_VC
MT_MAIN_MIC
MT_SUB_MIC1
MT_SUB_MIC2
MT_SUB_MIC3
LB_LINE
CAMCORDER_BACK_MIC
HDMI_SPKR
HEADSET_MIC
HEADSET_SPKR_MONO
HEADSET_SPKR_STEREO
TTY_HEADSET_MIC
TTY_HEADSET_SPKR
HEADSET_ANC_MIC_MONO
HEADSET_ANC_MIC_STEREO
HEADSET_SPKR_ANC_STEREO_FF
HEADSET_SPKR_ANC_STEREO_FB
HEADSET_SPKR_ANC_MONO_FF
HEADSET_SPKR_ANC_MONO_FB
USB_TX
USB_RX
LB_HEADSET
LB_HEADPHONES
HEADSET_MIC_VM
HEADSET_MIC_CAMCORDER
HEADSET_SPKR_STEREO_VOICE
SPKR_PHONE_MIC
SPKR_PHONE_MIC_BROADSIDE
SPKR_PHONE_MIC_ENDFIRE
SPKR_PHONE_SPKR_MONO
SPKR_PHONE_SPKR_STEREO
SPKR_PHONE_QUAD_MIC
SPKR_MIC_STEREO
SPKR_PHONE_MIC_FLUENCEV5
SPKR_PHONE_MIC_ENDFIRE_FLUENCEV5
QUAD_MIC_SURROUND
LB_SPKR
SPKR_PHONE_SPKR_MONO_VOICE

Add + Delete -

Device Category: ID

Common Device Information

Device ID: [✓ Hex] Dec

Device Type: ID

Target Specific Information

Support Listen:

Support Spkr Prot FB:

Topology Information

Voice COPPTopologyID: ID

Audio COPP TopologyID: ID

AFETopologyID: ID

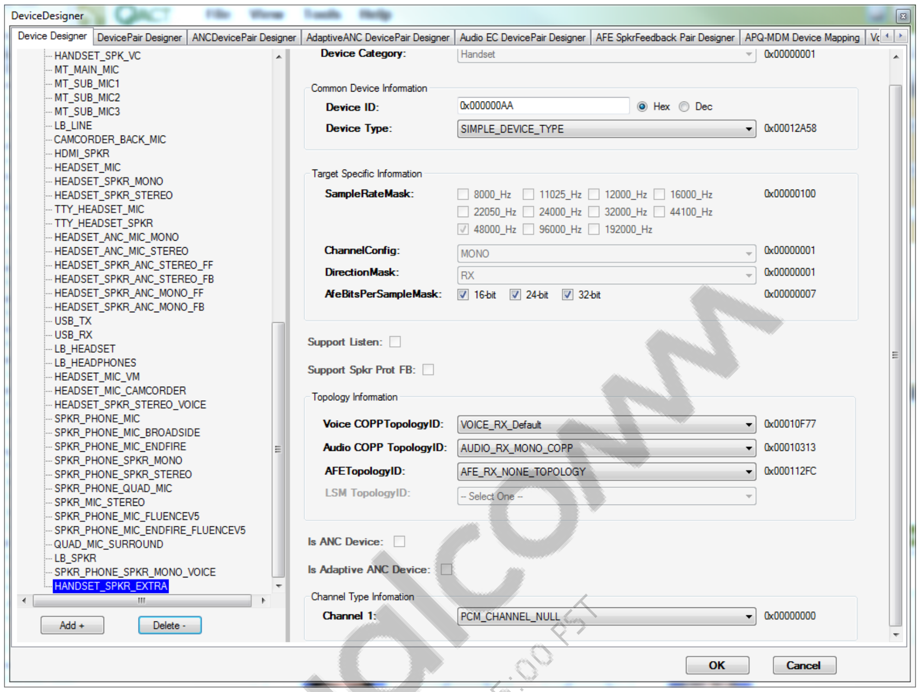
LSM TopologyID: ID

Is ANC Device:

Is Adaptive ANC Device:

OK Cancel

5. Enter information for the new device and click **OK**.



```
HANDSET_SPK_VC
MT_MAIN_MIC
MT_SUB_MIC1
MT_SUB_MIC2
MT_SUB_MIC3
LB_LINE
CAMCORDER_BACK_MIC
HDMI_SPKR
HEADSET_MIC
HEADSET_SPKR_MONO
HEADSET_SPKR_STEREO
TTY_HEADSET_MIC
TTY_HEADSET_SPKR
HEADSET_ANC_MIC_MONO
HEADSET_ANC_MIC_STEREO
HEADSET_SPKR_ANC_STEREO_FF
HEADSET_SPKR_ANC_STEREO_FB
HEADSET_SPKR_ANC_MONO_FF
HEADSET_SPKR_ANC_MONO_FB
USB_TX
USB_RX
LB_HEADSET
LB_HEADPHONES
HEADSET_MIC_VM
HEADSET_MIC_CAMCORDER
HEADSET_SPKR_STEREO_VOICE
SPKR_PHONE_MIC
SPKR_PHONE_MIC_BROADSIDE
SPKR_PHONE_MIC_ENDFIRE
SPKR_PHONE_SPKR_MONO
SPKR_PHONE_SPKR_STEREO
SPKR_PHONE_QUAD_MIC
SPKR_MIC_STEREO
SPKR_PHONE_MIC_FLUENCEV5
SPKR_PHONE_MIC_ENDFIRE_FLUENCEV5
QUAD_MIC_SURROUND
LB_SPKR
SPKR_PHONE_SPKR_MONO_VOICE
HANDSET_SPKR_EXTRA
```

Device Category:	Handset	0x00000001
Common Device Information		
Device ID:	0x000000AA [✓ Hex] Dec	
Device Type:	SIMPLE_DEVICE_TYPE	0x00012A58
Target Specific Information		
SampleRateMask:	8000_Hz, 11025_Hz, 12000_Hz, 16000_Hz, 22050_Hz, 24000_Hz, 32000_Hz, 44100_Hz, [✓ 48000_Hz], 96000_Hz, 192000_Hz	0x00000100
ChannelConfig:	MONO	0x00000001
DirectionMask:	RX	0x00000001
AfeBitsPerSampleMask:	[✓ 16-bit] [✓ 24-bit] [✓ 32-bit]	0x00000007
Support Listen:	☐	
Support Spkr Prot FB:	☐	
Topology Information		
Voice COPPTopologyID:	VOICE_RX_Default	0x00010F77
Audio COPP TopologyID:	AUDIO_RX_MONO_COPP	0x00010313
AFETopologyID:	AFE_RX_NONE_TOPOLOGY	0x000112FC
LSM TopologyID:	-- Select One --	
Is ANC Device:	☐	
Is Adaptive ANC Device:	☐	
Channel Type Infomation		
Channel 1:	PCM_CHANNEL_NULL	0x00000000

The following naming convention rules apply to both voice and audio devices:

- A mic device is referred to as a Tx device or a mic device. The mic device represents a signal path from a mic to the codec to the aDSP. Such devices are clearly identifiable by the presence of the string MIC in the device name, for example, HANDSET_MIC.
- A speakerphone device is referred to as an Rx device or an SPKR device. The speakerphone device represents a signal path from the aDSP in the MSM™ chipset to the codec to the output transducer (speaker, earpiece, headset, and so on). Such devices are clearly identifiable by the presence of the string SPKR in the device name, for example, HANDSET_SPKR.
- Audio has separate devices for playback and recording; whereas, the voice call has the Tx and Rx devices paired together. However, each device that exists as a voice device pair is individually defined in the Device Designer. For example, the voice device HANDSET_MIC&HANDSET_SPKR has two components, HANDSET_MIC and HANDSET_SPKR, each of which is individually defined. To create custom use cases for voice calls, a device pair is required.

In this example, the new device record is named as HANDSET_SPKR_EXTRA.

- Device Category – The category of the devices that are added.
- Device ID – Unique value. To show, randomly pick a number (0x000000AA).
- Device Type – SIMPLE_DEVICE_TYPE is chosen.
- SampleRateMask – Fill in the sampling rates that the device supports.
- DirectionMask – Determines if the device is used for Tx or Rx.
- AfeBitsPerSampleMask – The number of bits per audio frame that this device profile supports.

- Topology Information
  - Voice COPP topology ID
  - Audio COPP topology ID
  - AFE topology ID
- Is ANC device
- Is Adaptive ANC device
- Channel type information

5.4.2 Create mixer path XML element for new device

Create an XML element for the new device device/qcom/msm8974/mixer_paths.xml.

```
    <path name="speaker_extra">
        <ctl name="SLIM RX1 MUX" value="AIF1_PB" />
        <ctl name="RX1 MIX1 INP1" value="RX1" />
        <ctl name="CLASS_H_DSM MUX" value="DSM_HPHL_RX1" />
        <ctl name="DAC1 Switch" value="1" />
    </path>
```

By default, the given mixer control is set to value 0, which is called when the mixer path is torn down.

```
<ctl name="RX1 MIX1 INP1" value="ZERO" />
<ctl name="CLASS_H_DSM MUX" value="ZERO" />
<ctl name="DAC1 Switch" value="0" />
```

5.4.3 Audio HAL changes

5.4.3.1 Define the new device entry in device_table

Update the hardware/qcom/audio/hal/msm8974/platform.h and platform.c files to add the entry in device_table for the new device name.

A meaningful string is used, for example, “Speaker Extra.” This string must match the mixer path XML name in Section 5.4.2.

In platform.h, add the new enum SND_DEVICE_OUT_SPEAKER_Extra to represent the newly added device.

If you are viewing this document using a color monitor, or if you print this document to a color printer, **red boldface** indicates code that is to be **added**.

```
    SND_DEVICE_OUT_VOICE_ANC_HEADSET,
    SND_DEVICE_OUT_VOICE_ANC_FB_HEADSET,
    SND_DEVICE_OUT_SPEAKER_AND_ANC_HEADSET,
    SND_DEVICE_OUT_ANC_HANDSET,
    SND_DEVICE_OUT_SPEAKER_PROTECTED,
    SND_DEVICE_OUT_SPEAKER_EXTRA,
    SND_DEVICE_OUT_END,
```

Add the speaker-extra entry to device_table in platform.c.

```
[SND_DEVICE_OUT_VOICE_ANC_FB_HEADSET] = "voice-anc-fb-headphones",
[SND_DEVICE_OUT_SPEAKER_AND_ANC_HEADSET] = "speaker-and-anc-headphones",
[SND_DEVICE_OUT_ANC_HANDSET] = "anc-handset",
[SND_DEVICE_OUT_SPEAKER_PROTECTED] = "speaker-protected",
[SND_DEVICE_OUT_SPEAKER_EXTRA] = "speaker-extra",
```

5.4.3.2 Add ACDB ID mapping

Add the ACDB device ID to acdb_device_table in hardware/qcom/audio/hal/msm8974/platform.c.

```
[SND_DEVICE_OUT_VOICE_ANC_FB_HEADSET] = 27,
[SND_DEVICE_OUT_SPEAKER_AND_ANC_HEADSET] = 26,
[SND_DEVICE_OUT_ANC_HANDSET] = 103,
[SND_DEVICE_OUT_SPEAKER_PROTECTED] = 101,
[SND_DEVICE_OUT_SPEAKER_EXTRA] = 170,
```

5.4.3.3 Add Android device to SND device mapping

In the platform_get_input_snd_device ()/platform_get_output_snd_device() function in the platform.c file, add the new logic to map the Android device to the newly added device.

In the following example, the Android device ID from Android framework is DEVICE_OUT_SPEAKER (0x2). The ID mapping is at system/core/include/system/audio.h.

In this example, the existing hi-fi playback on the handset speaker is rerouted to the newly added handset speaker named speaker-extra.

```
if (devices & AUDIO_DEVICE_OUT_WIRED_HEADPHONE ||
    devices & AUDIO_DEVICE_OUT_WIRED_HEADSET) {
    if (devices & AUDIO_DEVICE_OUT_WIRED_HEADSET
        && audio_extn_get_anc_enabled()) {
        if (audio_extn_should_use_fb_anc())
            snd_device = SND_DEVICE_OUT_ANC_FB_HEADSET;
        else
            snd_device = SND_DEVICE_OUT_ANC_HEADSET;
    }
    else
        snd_device = SND_DEVICE_OUT_HEADPHONES;
} else if (devices & AUDIO_DEVICE_OUT_SPEAKER) {
    if (adev->speaker_lr_swap)
```

```
            snd_device = SND_DEVICE_OUT_SPEAKER_REVERSE;
        else

            ~~snd_device = SND_DEVICE_OUT_SPEAKER;~~
            snd_device = SND_DEVICE_OUT_SPEAKER_EXTRA;
```

5.4.4 Log collection

The following logs are required to check if the new device is added properly.

5.4.4.1 User space logs

1. Rebuild APSS and flash an updated system.img to the device.
2. Push the modified *.acdb files to the etc/ of the target through adb.
3. Capture a log while playing music through a music player.
4. Create a log after the following command:

```
adb logcat –c
adb logcat –v treadtime | tee logcat.log
```

5.4.4.2 Kernel logs

If kernel logs are required, enable them as follows:

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo -n "file q6afe.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file q6adm.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm-pcm-routing-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file audio_acdb.c +p" > /sys/kernel/debug/dynamic_debug/control
```

5.4.5 Log analysis

To verify adding a device from the logs:

```
01-02 00:01:24.849   194   590 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: routing=2

01-02 00:01:24.849   194   590 V msm8974_platform:
platform_get_output_snd_device: enter: output devices(0x2)

01-02 00:01:24.849   194   590 V msm8974_platform:
platform_get_output_snd_device: exit: snd_device(speaker-extra)
```

```
01-02 00:01:24.849   194   590 D audio_hw_primary: select_devices:
out_snd_device(27: speaker-extra) in_snd_device(0: )

01-02 00:01:24.849   194   590 V audio_hw_primary: disable_audio_route:
enter: usecase(0)

01-02 00:01:24.849   194   590 V audio_hw_primary: disable_audio_route:
reset mixer path: deep-buffer-playback

01-02 00:01:24.889   194   590 V audio_hw_primary: disable_audio_route:
exit

01-02 00:01:24.889   194   590 D hardware_info: hw_info_append_hw_type :
device_name = headphones

01-02 00:01:24.889   194   590 V audio_hw_primary: disable_snd_device:
snd_device(4: headphones)

01-02 00:01:24.889   194   590 D hardware_info: hw_info_append_hw_type :
device_name = speaker-extra

01-02 00:01:24.889   194   590 V audio_hw_primary: enable_snd_device:
snd_device(27: speaker-extra)

01-02 00:01:24.889   194   590 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 170, path =  0

01-02 00:01:24.889   194   590 D ACDB-LOADER: ACDB -> send_adm_topology

01-02 00:01:24.889   194   590 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID

01-02 00:01:24.889   194   590 D ACDB-LOADER: ACDB -> send_audtable

01-02 00:01:24.889   194   590 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE

01-02 00:01:24.889   194   590 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL

01-02 00:01:24.889   194   590 D ACDB-LOADER: ACDB -> send_audvoltable

01-02 00:01:24.889   194   590 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
```

6 Playback

The audio playback modes on Android devices supported by QTI are categorized as:

- Deep Buffer mode play back – The default music playback mode.
- Low Latency mode playback – Touch tones, alert tones, and so on, are played with minimum delay. This mode is used for game tones; basically anything that uses OpenSL APIs can go through this path.
- Compressed Offload mode play back – The APSS sends large buffers of encoded data (decoding happens in the DSP) to the DSP and then goes into a low-power state, waking up intermittently to provide more data.

6.1 Deep buffer playback

Figure 6-1 shows the paths taken by the control and data flows through the user space, kernel, and DSP components involved in media playback on Android.

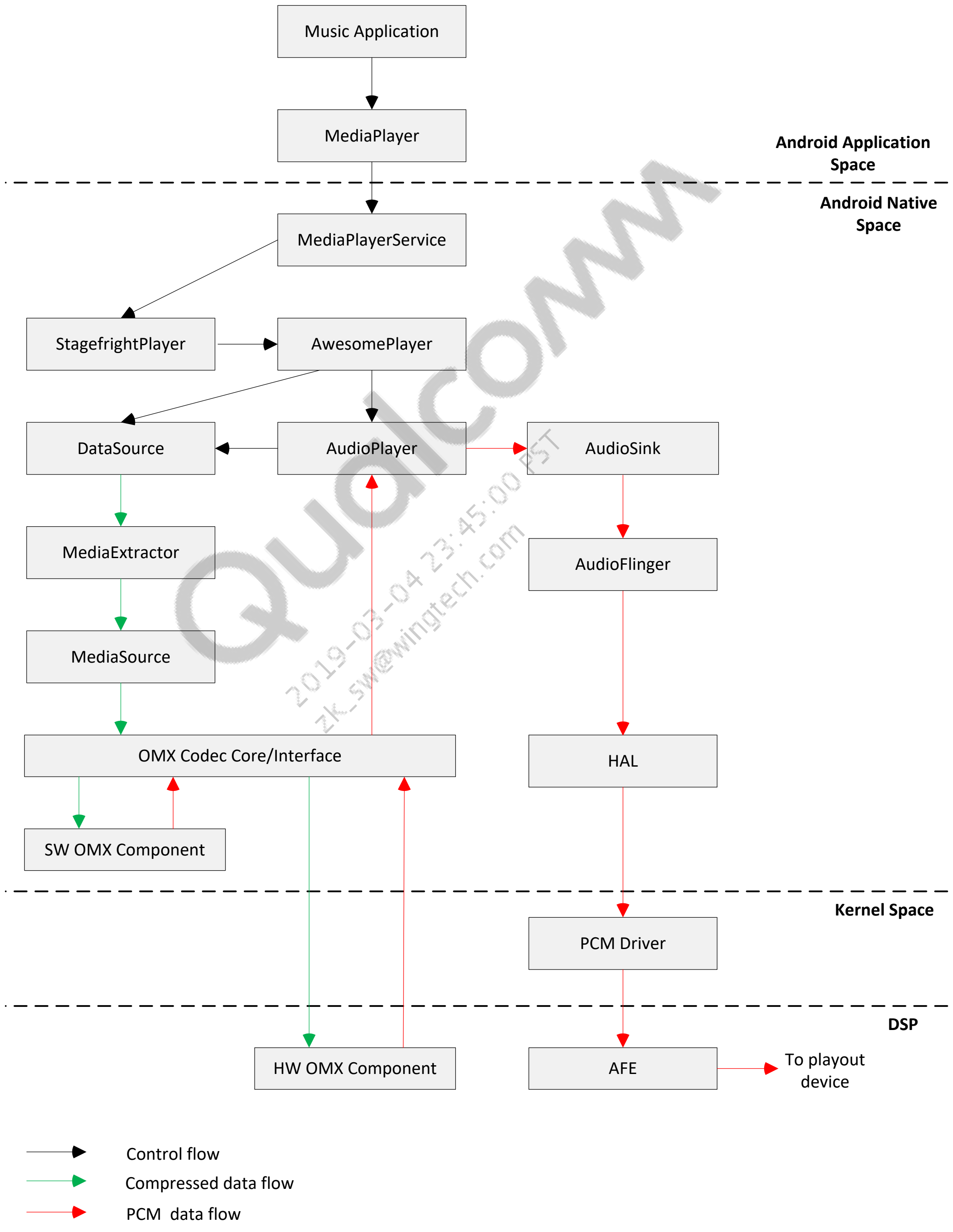


Figure 6-1 Android media playback components

MediaPlayerService is a system service that links the Android user space to the media framework. Stagefright is the native Android media framework that facilitates media playback and capture through various physical and proxy devices. Other component functions are:

- MediaExtractor is responsible for retrieving track data and the corresponding metadata from the underlying file system or http stream.
- OMX layer is used for decoding/encoding data to and from raw PCM data into appropriate audio formats as supported by the system. Software or hardware instances of codecs, service a request depending on the platform capabilities.
- If an audio track is present, the AudioPlayer is responsible for rendering audio and providing time base for AV synchronization in AwesomePlayer.
- AwesomePlayer works as the engine to coordinate the modules and is finally connected into the Android media framework through the adapter that is the StagefrightPlayer.
- AudioFlinger resamples and routes the media stream to the actual devices that are involved with the output or input operation.

6.1.1 Call flow

6.1.1.1 Initialize MP3 playback

Figure 6-2 shows the call flow involved in initializing normal audio playback.

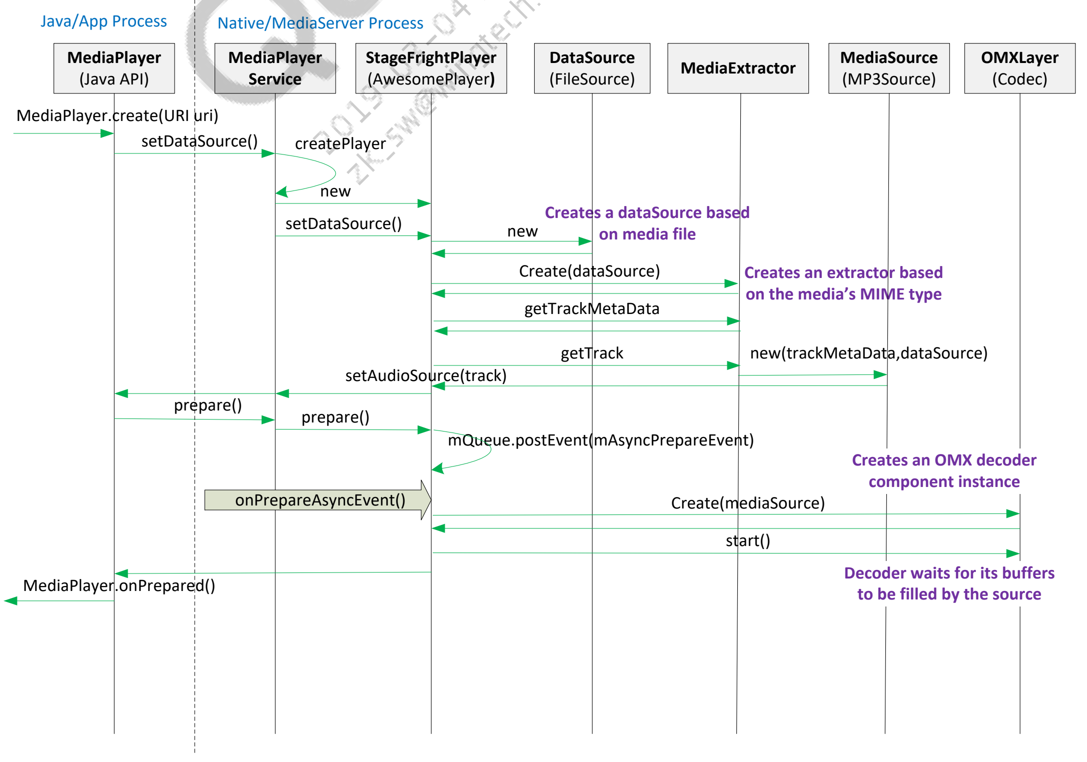
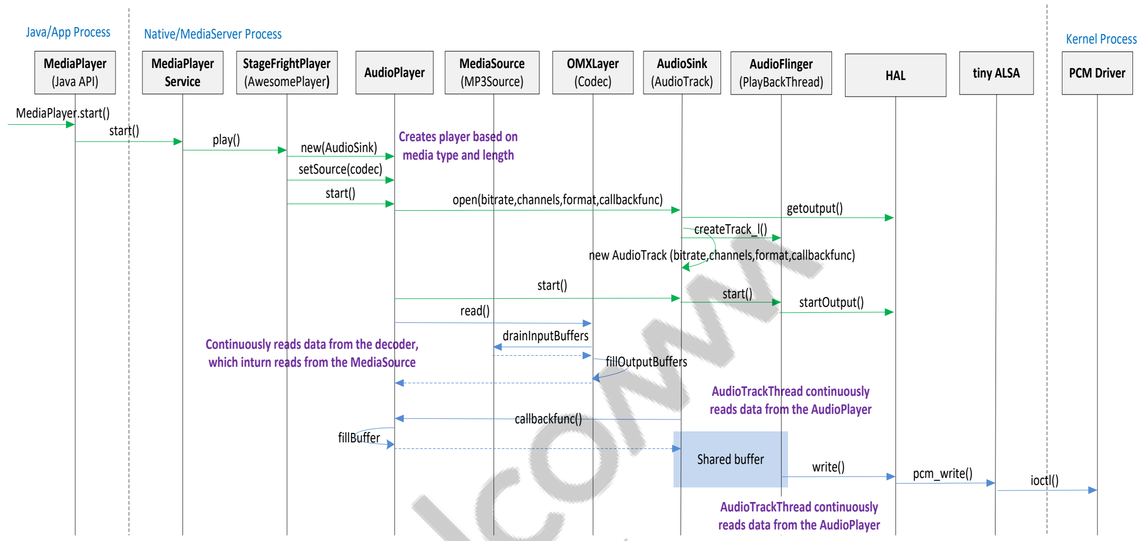

Java/App Process
Native/MediaServer Process
MediaPlayer (Java API)
MediaPlayer Service
StageFrightPlayer (AwesomePlayer)
DataSource (FileSource)
MediaExtractor
MediaSource (MP3Source)
OMXLayer (Codec)
MediaPlayer.create(URI uri)
setDataSource()
createPlayer
new
setDataSource()
new
Creates a dataSource based on media file
Create(dataSource)
Creates an extractor based on the media's MIME type
getTrackMetaData
getTrack
new(trackMetaData,dataSource)
setAudioSource(track)
prepare()
prepare()
mQueue.postEvent(mAsyncPrepareEvent)
Creates an OMX decoder component instance
onPrepareAsyncEvent()
Create(mediaSource)
start()
MediaPlayer.onPrepared()
Decoder waits for its buffers to be filled by the source

Figure 6-2 Normal playback initialization call flow

6.1.1.2 Start playback

Figure 6-3 shows the call and data flows during active audio playback.



Java/App Process
Native/MediaServer Process
Kernel Process
MediaPlayer (Java API)
MediaPlayer Service
StageFrightPlayer (AwesomePlayer)
AudioPlayer
MediaSource (MP3Source)
OMXLayer (Codec)
AudioSink (AudioTrack)
AudioFlinger (PlayBackThread)
HAL
tiny ALSA
PCM Driver
MediaPlayer.start()
start()
play()
new(AudioSink)
setSource(codec)
start()
Creates player based on media type and length
open(bitrate,channels,format,callbackfunc)
getoutput()
createTrack_l()
new AudioTrack (bitrate,channels,format,callbackfunc)
start()
start()
startOutput()
read()
drainInputBuffers
fillOutputBuffers
Continuously reads data from the decoder, which inturn reads from the MediaSource
AudioTrackThread continuously reads data from the AudioPlayer
callbackfunc()
fillBuffer
Shared buffer
write()
pcm_write()
ioctl()
AudioTrackThread continuously reads data from the AudioPlayer

Figure 6-3 Normal playback call flow

6.1.2 Log collection

This section highlights the various files and/or components in which logging is enabled to troubleshoot issues related to audio playback on Android.

6.1.2.1 User space logs

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

6.1.2.2 Kernel logs

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file msm-pcm-q6-v2.c +p > /sys/kernel/debug/dynamic_debug/control
echo file msm-pcm-routing-v2.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging on msm-pcm-q6-v2.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

6.1.2.3 Additional logging

Depending on the issue, additional logging is enabled in the user space and kernel space.

In the user space, enable logs on files listed at the following locations, if the facility exists:

```
\frameworks\av\media\libmediaplayerservice
MediaPlayerService.cpp
StagefrightPlayer.cpp

\frameworks\av\media\libmedia
AudioSystem.cpp
AudioTrack.cpp
mediaplayer.cpp
IOMX.cpp

\frameworks\av\media\libstagefright
AwesomePlayer.cpp
AudioPlayer.cpp
FileSource.cpp
MP3Extractor.cpp
OMXClient.cpp
OMXCodec.cpp

\frameworks\av\services\audioflinger
AudioFlinger.cpp
AudioMixer.cpp
AudioPolicyService.cpp

\harwdware\qcom\audio\hal
audio_hw.c

\harwdware\qcom\audio\hal\msm8974
platform.c
hw_info.c

\harwdware\qcom\audio\hal\policy_hal
AudioPolicyManager.cpp

\harwdware\libhardware_legacy\audio
AudioPolicyManagerBase.cpp
```

In the kernel side, enable logs on the following files to debug issues related to APSS and DSP communication:

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6afe.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6asm.c  +p > /sys/kernel/debug/dynamic_debug/control
```

For calibration-related issues, enable log messages in audio_acdb.c.

```
echo file audio_acdb.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging in q6asm.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

6.1.3 Log analysis

6.1.3.1 User space logs

Audio playback is initiated from the application layer through the MediaPlayer service. MediaPlayer in turn uses the Stagefright framework to fulfill the actual playout. StagefrightPlayer instantiates AwesomePlayer in the playback scenario.

```
12-12 17:01:50.637  6394  6394 V MediaPlayer: reset
12-12 17:01:50.647   218  1155 V MediaPlayerService: Client(8) constructor
12-12 17:01:50.647  6394  6394 V MediaPlayer: setDataSource(63, 0,
576460752303423487)

12-12 17:01:50.647   218   218 V StagefrightPlayer: StagefrightPlayer
12-12 17:01:50.647   218   218 V AudioSink: AudioOutput(37)
12-12 17:01:50.657   218   218 V StagefrightPlayer: setDataSource(24, 0,
9720383))
```

Once MediaPlayer, StagefrightPlayer, and AwesomePlayer instances are created, the input file data is extracted and a MediaSource is created from it.

```
12-12 17:01:50.657   218   218 V MP3Extractor: skipped ID3 tag, new
starting offset is 2205 (0x000000000000089d)
12-12 17:01:50.657   218   218 V MP3Extractor: found possible 1st frame at
2205 (header = 0xfffbb200)
12-12 17:01:50.657   218   218 V MP3Extractor: subsequent header is
fffbb200
12-12 17:01:50.657   218   218 V MP3Extractor: found subsequent frame #2 at
2832
12-12 17:01:50.657   218   218 V MP3Extractor: subsequent header is
fffbb200
12-12 17:01:50.657   218   218 V MP3Extractor: found subsequent frame #3 at
3459
12-12 17:01:50.657   218   218 V MP3Extractor: subsequent header is
fffbb200
12-12 17:01:50.657   218   218 V MP3Extractor: found subsequent frame #4 at
4086
```

```
12-12 17:01:50.667   218   218 V ExtendedExtractor: Sniff Failed
12-12 17:01:50.667   217   526 D PRDrmPlugIn: PRDrmPlugin::onInitialize,
uniqueId = 4764
12-12 17:01:50.667   217   526 D PRDrmPlugIn:
PRDrmPlugin::onSetOnInfoListener: uniqueId = 4764
12-12 17:01:50.667   217   526 D PRDrmPlugIn: PRDrmPlugin::onTerminate,
uniqueId = 4764
12-12 17:01:50.667   218   218 V MediaExtractor: Autodetected media content
as 'audio/mpeg' with confidence 0.20
12-12 17:01:50.667   218   218 D ExtendedUtils: extended extractor not
needed, return default
12-12 17:01:50.667   218   218 V AwesomePlayer: mBitrate = 192000 bits/sec
12-12 17:01:50.667   218   218 V MediaPlayerService:  setDataSource

12-12 17:01:50.667  6394  6394 V MediaPlayer: prepare
```

The prepare() API is then invoked, which instantiates the decoder that is used to decode the compressed media. In this case, an MP3 file is to be played out so the software MP3 decoder provided by Google is instantiated. Input and output buffers for the decoder are also allocated as part of its creation and it is started.

```
12-12 17:01:50.667   218  6481 V OMXCodec: matching
'OMX.google.mp3.decoder' quirks 0x00000000
12-12 17:01:50.667   218  6481 V OMXCodec: matching 'MP3Decoder' quirks
0x00000000
12-12 17:01:50.667   218  6481 V OMXCodec: Attempting to allocate OMX node
'OMX.google.mp3.decoder'
12-12 17:01:50.667   218  6481 V SoftOMXPlugin: makeComponentInstance
'OMX.google.mp3.decoder'
12-12 17:01:50.667   218  6481 V OMXCodec: Successfully allocated OMX node
'OMX.google.mp3.decoder'
12-12 17:01:50.667   218  6481 V OMXCodec: [OMX.google.mp3.decoder]
allocating 4 buffers of size 8192 on input port
12-12 17:01:50.667   218  6481 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb7482160 on input port
12-12 17:01:50.667   218  6481 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb747c8d8 on input port
12-12 17:01:50.667   218  6481 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb7486508 on input port
12-12 17:01:50.667   218  6481 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb7492cf8 on input port
12-12 17:01:50.667   218  6481 V OMXCodec: [OMX.google.mp3.decoder]
allocating 4 buffers of size 9216 on output port
12-12 17:01:50.667   218  6481 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb74926f0 on output port
12-12 17:01:50.667   218  6481 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb7485998 on output port
12-12 17:01:50.667   218  6481 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb747e448 on output port
```

```
12-12 17:01:50.667   218  6481 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb748d030 on output port
12-12 17:01:50.667   218  6482 V OMX     : OnEvent(0, 0, 2)
12-12 17:01:50.667   218  6483 V OMXCodec: [OMX.google.mp3.decoder]
onStateChange 2
12-12 17:01:50.667   218  6483 V OMXCodec: [OMX.google.mp3.decoder] Now
Idle.
```

The decoder moves into the executing state and waits for its buffers to be filled by the source. The prepare sequence is complete.

```
12-12 17:01:50.667   218  6482 V OMX     : OnEvent(0, 0, 3)
12-12 17:01:50.667   218  6483 V OMXCodec: [OMX.google.mp3.decoder]
onStateChange 3
12-12 17:01:50.667   218  6483 V OMXCodec: [OMX.google.mp3.decoder] Now
Executing.
12-12 17:01:50.667  6394  6406 V MediaPlayer: prepared
12-12 17:01:50.667  6394  6394 V MediaPlayer: prepare complete - status=0
```

Once initialized, the start () API is invoked, which kicks off media playback. StagefrightPlayer creates an instance of AudioPlayer and starts it to perform the actual playout of media.

```
12-12 17:01:50.737  6394  6394 V MediaPlayer: start
12-12 17:01:50.737   218   218 V StagefrightPlayer: start
```

StagefrightPlayer through the AudioPlayer provides data to and reads data from the decoder buffers.

```
12-12 17:01:50.737   218   218 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb7482160 (length 627), timestamp 0 us (0.00 secs)
12-12 17:01:50.737   218   218 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb747c8d8 (length 627), timestamp 26122 us (0.03
secs)
12-12 17:01:50.737   218  6482 V SimpleSoftOMXComponent: msgType = 1
12-12 17:01:50.737   218   218 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb7486508 (length 627), timestamp 52244 us (0.05
secs)
12-12 17:01:50.737   218  6482 V SimpleSoftOMXComponent: msgType = 1
12-12 17:01:50.737   218   218 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb7492cf8 (length 627), timestamp 78367 us (0.08
secs)

12-12 17:01:50.747   218   218 V OMXCodec: [OMX.google.mp3.decoder] Calling
fillBuffer on buffer 0xb74926f0
12-12 17:01:50.747   218   218 V OMXCodec: [OMX.google.mp3.decoder] Calling
fillBuffer on buffer 0xb7485998
```

```
12-12 17:01:50.747   218   218 V OMXCodec: [OMX.google.mp3.decoder] Calling
fillBuffer on buffer 0xb747e448
12-12 17:01:50.747   218   218 V OMXCodec: [OMX.google.mp3.decoder] Calling
fillBuffer on buffer 0xb748d030

12-12 17:01:50.747   218  6483 V OMXCodec: [OMX.google.mp3.decoder]
EMPTY_BUFFER_DONE(buffer: 0xb7482160)
12-12 17:01:50.747   218  6483 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb7482160 (length 627), timestamp 104489 us (0.10
secs)

12-12 17:01:50.747   218  6483 V OMXCodec: [OMX.google.mp3.decoder]
FILL_BUFFER_DONE(buffer: 0xb74926f0, size: 2492, flags: 0x00000000,
timestamp: 0 us (0.00 secs))
```

The AudioSink created previously is opened, which instantiates an AudioTrack instance and adds it to the mixer. Volume and optional effects are set in AudioMixer.

```
12-12 17:01:50.747   218   218 V AudioSink: open(44100, 2, 0x0, 0x1, 4, 37
0x8)
12-12 17:01:50.747   218   218 V AudioPolicyService: getOutput()
12-12 17:01:50.747   218   218 V legacy_audio_policy_hal: audio_io_handle_t
android_audio_legacy::ap_get_output(audio_policy*, audio_stream_type_t,
uint32_t, audio_format_t, audio_channel_mask_t, audio_output_flags_t, const
audio_offload_info_t*): tid 218
12-12 17:01:50.747   218   218 V AudioPolicyManagerBase: getOutput() device
8, stream 3, samplingRate 0, format 0, channelMask 3, flags 0

12-12 17:01:50.747   218   218 V AudioSystem: getOutputSamplingRate()
reading from output desc
12-12 17:01:50.747   218   218 V AudioSystem: getSamplingRate() streamType
3, output 2, sampling rate 48000
12-12 17:01:50.747   218   218 V AudioSink: no track available to recycle
12-12 17:01:50.747   218   218 V AudioSink: creating new AudioTrack
12-12 17:01:50.747   218   218 V AudioTrack: sampleRate 44100, channelMask
0x3, format 1
12-12 17:01:50.747   218   218 V AudioTrack: set() streamType 3 frameCount
3528 flags 0008
12-12 17:01:50.747   218   218 V AudioSystem: getLatency() streamType 3,
output 2, latency 160
12-12 17:01:50.747   218   218 V AudioSystem: getFrameCount() streamType 3,
output 2, frameCount 960
12-12 17:01:50.747   218   218 V AudioSystem: getSamplingRate() streamType
3, output 2, sampling rate 48000
12-12 17:01:50.747   218   218 V AudioTrack: createTrack_l() output 2
afLatency 160
12-12 17:01:50.747   218   218 V AudioTrack: afFrameCount=960,
minBufCount=8, afSampleRate=48000, afLatency=160
12-12 17:01:50.747   218   218 V AudioTrack: minFrameCount: 7056,
afFrameCount=960, minBufCount=8, sampleRate=44100, afSampleRate=48000,
afLatency=160
12-12 17:01:50.747   218   218 V AudioTrack: Minimum buffer size corrected
from 3528 to 7056
12-12 17:01:50.747   218   218 V AudioFlinger: createTrack() sessionId: 37
12-12 17:01:50.747   218   218 V AudioFlinger: createTrack() lSessionId: 37
12-12 17:01:50.747   218   218 V AudioMixer: add track (1)

12-12 17:01:50.747   218   218 V AudioFlinger: Track constructor name 4097,
calling pid 6394
12-12 17:01:50.747   218   218 V AudioFlinger: acquiring 37 from 6394
12-12 17:01:50.747   218   218 V AudioFlinger:  incremented refcount to 2
12-12 17:01:50.747   218   218 V AudioSink: setVolume
12-12 17:01:50.757   218   218 V AudioSink: open() DONE status 0
```

After the AudioSink is opened, it is started which initiates audio path setup to the actual device where playback is needed.

```
12-12 17:01:50.757   218   218 V AudioSink: start
12-12 17:01:50.757   218   218 V AudioFlinger: start(4097), calling pid
6394 session 37
12-12 17:01:50.757   218   218 V AudioFlinger: ? => ACTIVE (4097) on thread
0xb749b7b8

12-12 17:01:50.757   218   218 V AudioPolicyService: startOutput()
12-12 17:01:50.757   218   218 V AudioPolicyManagerBase: startOutput()
output 2, stream 3, session 37
12-12 17:01:50.757   218   218 V AudioPolicyManagerBase: getNewDevice()
selected device 8
12-12 17:01:50.757   218   218 V AudioPolicyManagerBase: setOutputDevice()
output 2 device 0008 delayMs 0
12-12 17:01:50.757   218   218 V AudioPolicyManager: checkAndSetVolume:
index 10 output 2 device 8
12-12 17:01:50.757   218   218 V AudioFlinger: signal playback thread
12-12 17:01:50.757   218   592 V AudioFlinger: anticipated start
```

The output path for playback is opened up through Audio HAL. Device is headphones defined in /hardware/qcom/audio/hal/msm8974/platform.c. Use case (0) is USECASE_AUDIO_PLAYBACK_DEEP_BUFFER defined in hardware/qcom/audio/hal/audio_hw.h.

```
12-12 17:01:50.757   218   592 V audio_hw_primary: start_output_stream:
enter: usecase(0: deep-buffer-playback) devices(0x8)
12-12 17:01:50.757   218   592 V msm8974_platform:
platform_get_output_snd_device: enter: output devices(0x8)
12-12 17:01:50.757   218   592 V msm8974_platform:
platform_get_output_snd_device: exit: snd_device(headphones)
12-12 17:01:50.757   218   592 D audio_hw_primary: select_devices:
out_snd_device(4: headphones) in_snd_device(0: )
12-12 17:01:50.757   218   592 D hardware_info: hw_info_append_hw_type :
```

```
device_name = headphones
12-12 17:01:50.757   218   592 V audio_hw_primary: enable_snd_device:
snd_device(4: headphones) is already active
12-12 17:01:50.757   218   592 V audio_hw_primary: enable_audio_route:
enter: usecase(0)
12-12 17:01:50.757   218   592 V audio_hw_primary: enable_audio_route:
apply mixer path: deep-buffer-playback
12-12 17:01:50.757   218   592 V audio_hw_primary: enable_audio_route: exit
```

card_id (0) SOUND_CARD and device_id (0) is DEEP_BUFFER_PCM_DEVICE defined in hardware/qcom/audio/hal/msm8974/platform.h.

```
12-12 17:01:50.757   218   592 V audio_hw_primary: start_output_stream:
Opening PCM device card_id(0) device_id(0)

12-12 17:01:50.757   218   218 V MediaPlayerService: [8] notify
(0xb747c3c0, 6, 0, 0)
12-12 17:01:50.757   218  6484 V AudioTrack: obtainBuffer(2352) returned
7056 = 2352 + 4704 err 0
12-12 17:01:50.757   218  6484 V AudioPlayer: AudioCallback
```

The end-to-end path is set up and audio is played out of the device. AudioPlayer tracks the timestamps. The decoder continues decoding the input media as long as input data is available.

```
12-12 17:01:50.757   218  6484 V AudioPlayer: buffer->size() = 0,
mPositionTimeMediaUs=-0.00 mPositionTimeRealUs=0.00
12-12 17:01:50.757   218  6484 V OMXCodec: [OMX.google.mp3.decoder] Calling
fillBuffer on buffer 0xb74926f0
2-12 17:01:50.757   218  6482 V OMX     : OnFillBufferDone
buffer=0xb74926f0
12-12 17:01:50.757   218  6482 V OMX     : OnEmptyBufferDone
buffer=0xb7482160
12-12 17:01:50.757   218  6483 V OMXCodec: [OMX.google.mp3.decoder]
EMPTY_BUFFER_DONE(buffer: 0xb7482160)
12-12 17:01:50.757   218  6483 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb7482160 (length 627), timestamp 208979 us (0.21
secs)

12-12 17:01:50.767   218  6484 V AudioPlayer: buffer->size() = 1580,
mPositionTimeMediaUs=0.05 mPositionTimeRealUs=0.00
12-12 17:01:50.767   218  6484 V OMXCodec: [OMX.google.mp3.decoder] Calling
fillBuffer on buffer 0xb747e448
12-12 17:01:50.767   218  6482 V OMX     : OnFillBufferDone
buffer=0xb747e448
12-12 17:01:50.767   218  6483 V OMXCodec: [OMX.google.mp3.decoder]
EMPTY_BUFFER_DONE(buffer: 0xb7486508)
12-12 17:01:50.767   218  6483 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb7486508 (length 627), timestamp 261224 us (0.26
secs)
```

The audio file has reached its end and playback is complete. All components are stopped, and output device routing is reset.

```
12-12 17:02:51.077   218  6484 V OMXCodec: [OMX.google.mp3.decoder] There
is no more output data available, not calling fillOutputBuffer
12-12 17:02:51.077   218  6484 V AudioPlayer: buffer->size() = 2116,
mPositionTimeMediaUs=404.82 mPositionTimeRealUs=13.71
12-12 17:02:51.077   218  6484 V OMXCodec: [OMX.google.mp3.decoder] There
is no more output data available, not calling fillOutputBuffer

12-12 17:02:51.567   218  6481 V AwesomePlayer: MEDIA_PLAYBACK_COMPLETE
12-12 17:02:51.567  6394  6406 V MediaPlayer: playback complete
12-12 17:02:51.567   218  6481 V AudioSink: stop
12-12 17:02:51.567   218  6481 V AudioFlinger: stop(4097), calling pid 218

12-12 17:02:51.897   218   592 V AudioPolicyService: stopOutput()
12-12 17:02:51.897   218   589 V AudioPolicyManagerBase: stopOutput()
output 2, stream 3, session 37
12-12 17:02:51.897   218   589 V AudioPolicyManagerBase: changeRefCount()
stream 3, count 0
12-12 17:02:51.897   218   589 V AudioPolicyManagerBase: getNewDevice()
selected device 0
12-12 17:02:51.907   218   589 V AudioPolicyManagerBase: setOutputDevice()
output 2 device 0000 delayMs 320
```

If there is no activity after 3 seconds of playback, the audio path is torn down.

```
12-12 17:02:54.717   218   592 V AudioFlinger: Audio hardware entering
standby, mixer 0xb5c37008, suspend count 0
12-12 17:02:55.007   218   592 V audio_hw_primary: stop_output_stream:
enter: usecase(0: deep-buffer-playback)
12-12 17:02:55.007   218   592 V audio_hw_primary: disable_audio_route:
enter: usecase(0)
12-12 17:02:55.007   218   592 V audio_hw_primary: disable_audio_route:
reset mixer path: deep-buffer-playback
12-12 17:02:55.007   218   592 V audio_hw_primary: disable_audio_route:
exit
12-12 17:02:55.007   218   592 D hardware_info: hw_info_append_hw_type :
device_name = headphones
12-12 17:02:55.007   218   592 V audio_hw_primary: stop_output_stream:
exit: status(0)
12-12 17:02:55.007   218   592 V audio_hw_primary: out_standby: exit
```

6.1.3.2 Kernel logs

```
<7>[ 1893.155109] msm_pcm_routing_process_audio: reg 2 val 0 set 1
```

The PCM routing is to back-end DAI ID 2 (MSM_BACKEND_DAI_
SLIMBUS_0_RX) from front-end DAI ID 0 (MSM_FRONTEND_DAI_MULTIMEDIA1). The

back-end DAI IDs and FRONT END DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[ 1893.157750] msm_pcm_hw_params: perf: 0
<7>[ 1893.161147] event_handler:Payload = [0x10db3]stat[0x0]
<7>[ 1893.227273] msm_pcm_playback_prepare
<7>[ 1893.227711] event_handler:Payload = [0x10d98]stat[0x0]
<7>[ 1893.229604] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1893.229613] msm-pcm-routing msm-pcm-routing: reg 0 val 1
<7>[ 1893.229897] msm_pcm_playback_copy: prtd->out_count = 8
<7>[ 1893.229906] msm_pcm_playback_copy:fbytes =3840: xfer=0 size=3840
<7>[ 1893.229920] msm_pcm_playback_copy:fbytes = 0: xfer=3840
<7>[ 1893.229929] msm_pcm_trigger: Trigger start
<7>[ 1893.229961] pcm_irq_pos = 0
```

PCM playback is started by setting the hardware parameters and calling msm_pcm_playback_prepare followed by an msm_pcm_trigger() function.

PCM data is copied to DSP buffers. There are eight buffers each of size 3840 bytes.

```
<7>[ 1893.231054] msm_pcm_playback_copy:fbytes =3840: xfer=0 size=3840
<7>[ 1893.231064] msm_pcm_playback_copy:fbytes = 0: xfer=3840
<7>[ 1893.231069] msm_pcm_playback_copy:writing 3840 bytes of buffer to dsp
<7>[ 1893.231164] pcm_irq_pos = 3840
```

Once the buffer is consumed, the pcm_irq_pos increments by period size that is 3840.

```
<7>[ 1893.231494] msm_pcm_playback_copy: prtd->out_count = 6
<7>[ 1893.231502] msm_pcm_playback_copy:fbytes =3840: xfer=0 size=3840
<7>[ 1893.231513] msm_pcm_playback_copy:fbytes = 0: xfer=3840
<7>[ 1893.231518] msm_pcm_playback_copy:writing 3840 bytes of buffer to dsp
<7>[ 1893.232117] pcm_irq_pos = 7680
.
.
.
<7>[ 1893.234006] msm_pcm_playback_copy: prtd->out_count = 1
<7>[ 1893.234014] msm_pcm_playback_copy:fbytes =3840: xfer=0 size=3840
<7>[ 1893.234023] msm_pcm_playback_copy:fbytes = 0: xfer=3840
<7>[ 1893.234028] msm_pcm_playback_copy:writing 3840 bytes of buffer to dsp
<7>[ 1893.304147] pcm_irq_pos = 26880
```

Once all eight buffers are consumed, the pcm_irq_pos value wraps around to 0.

```
<7>[ 1893.324039] ASM_DATA_EVENT_WRITE_DONE_V2
<7>[ 1893.324112] msm_pcm_playback_copy:fbytes =3840: xfer=0 size=3840
<7>[ 1893.324127] msm_pcm_playback_copy:fbytes = 0: xfer=3840
```

```
<7>[ 1893.324138] msm_pcm_playback_copy:writing 3840 bytes of buffer to dsp
<7>[ 1893.324194] pcm_irq_pos = 0

<7>[ 1957.114940] SNDRV_PCM_TRIGGER_STOP
```

Playback is stopped by sending SNDRV_PCM_TRIGGER_STOP.

```
<7>[ 1957.114996] msm_pcm_playback_close: cmd_pending 0x8
<7>[ 1957.124043] ASM_DATA_EVENT_WRITE_DONE_V2
<7>[ 1957.124053] Buffer Consumed = 0x7ee74b00
<7>[ 1957.144043] ASM_DATA_EVENT_WRITE_DONE_V2
<7>[ 1957.144053] Buffer Consumed = 0x7ee75a00
.
<7>[ 1957.264094] ASM_DATA_EVENT_WRITE_DONE_V2
<7>[ 1957.264105] Buffer Consumed = 0x7ee73c00
 <7>[ 1957.344750] ASM_DATA_EVENT_RENDERED_EOS
```

All the buffers sent to DSP are consumed, and the DSP sends an EOS event.

```
<7>[ 1957.400402] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1957.405273] msm_pcm_routing_process_audio: reg 2 val 0 set 0
```

PCM routing from front-end DAI to BACK END DAI is disabled.

6.1.4 Customization

None

6.1.5 Debugging

See Section 6.3.

6.2 Low Latency mode playback

Figure 6-4 shows the paths taken by the control and data flows through the user space, kernel, and DSP components involved in Low Latency mode playback on Android.

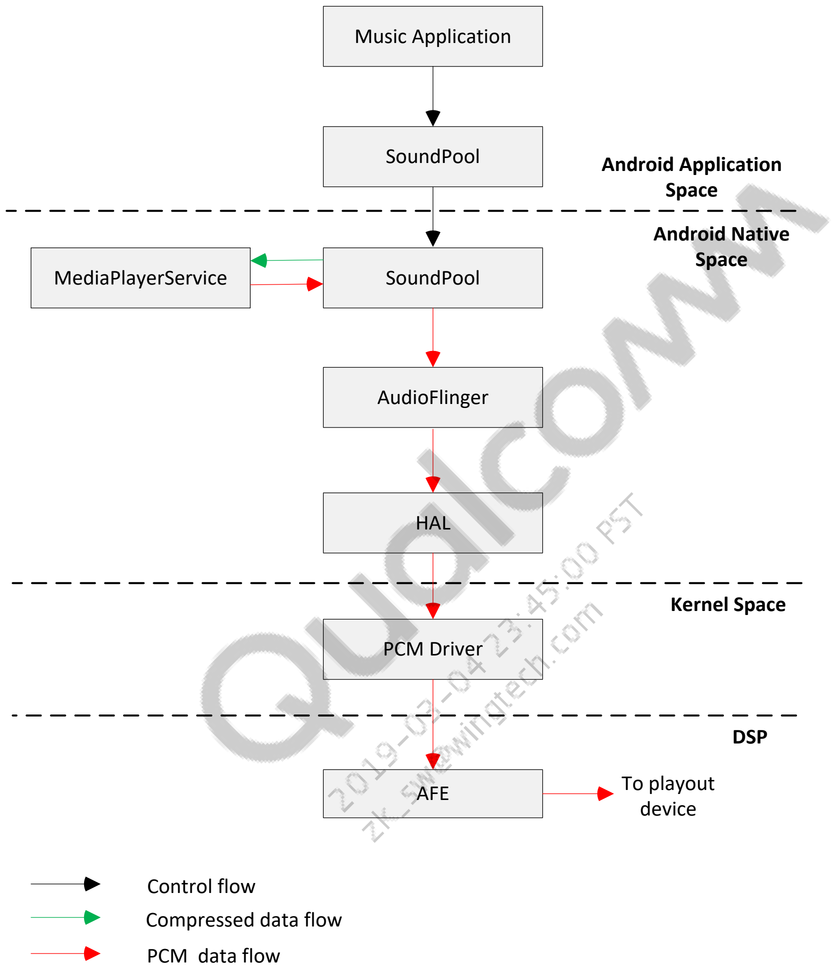


Figure 6-4 Android Low Latency mode playback components

Low Latency mode playback is used to play out system tones and various other real-time sounds in applications like games. It is preferable to play out the media as soon as possible as it is to user interface components/actions.

To realize this feature, changes were made to the following components in the playback loop:

- SoundPool is the main component that manages low latency playback. It facilitates loading media at once and at the start of an application. The MediaPlayerService decodes the media and holds the sounds as PCM in memory and reduces playback latency.

6.2.1 Call flow

Figure 6-5 shows the call flow involved in initializing and playback of low latency audio.

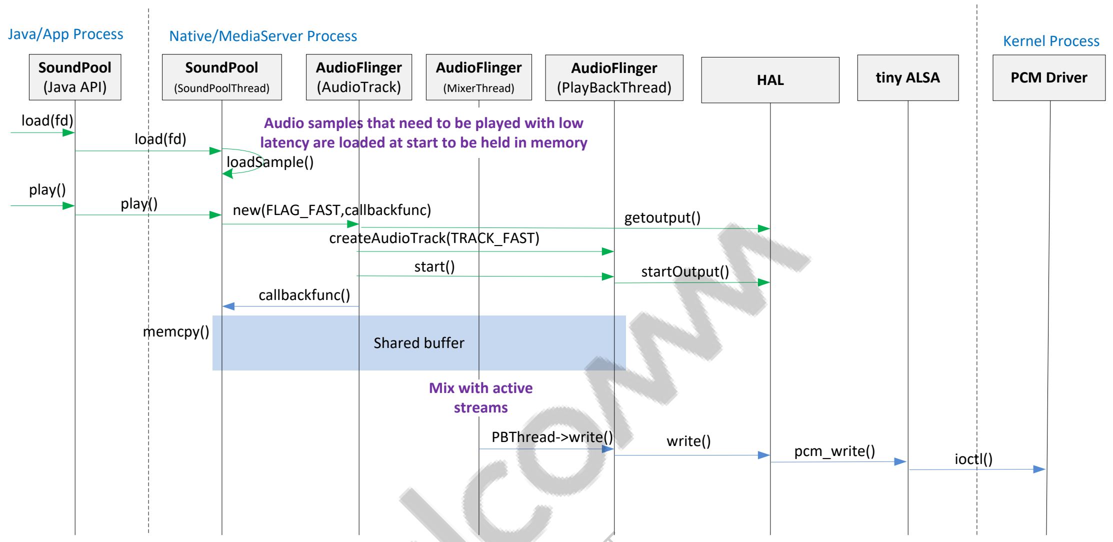

Java/App Process
Native/MediaServer Process
Kernel Process
SoundPool (Java API)
SoundPool (SoundPoolThread)
AudioFlinger (AudioTrack)
AudioFlinger (MixerThread)
AudioFlinger (PlayBackThread)
HAL
tiny ALSA
PCM Driver
load(fd)
load(fd)
Audio samples that need to be played with low latency are loaded at start to be held in memory
loadSample()
play()
play()
new(FLAG_FAST,callbackfunc)
getoutput()
createAudioTrack(TRACK_FAST)
start()
startOutput()
callbackfunc()
memcpy()
Shared buffer
Mix with active streams
PBThread->write()
write()
pcm_write()
ioctl()

Figure 6-5 Low Latency mode playback call flow

6.2.2 Log collection

This section highlights the various files and/or components in which logging is enabled to troubleshoot issues related to audio playback on Android.

6.2.2.1 User space logs

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

6.2.2.2 Kernel logs

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file msm-pcm-q6-v2.c +p > /sys/kernel/debug/dynamic_debug/control
echo file msm-pcm-routing-v2.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging in msm-pcm-q6-v2.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

6.2.2.3 Additional logging

Depending on the issue, additional logging is enabled in the user space and kernel space. In the user space, enable logs in files listed at the following locations, if the facility exists:

```
\frameworks\av\media\libmedia
AudioSystem.cpp
AudioTrack.cpp
IOMX.cpp
SoundPool.cpp

\frameworks\av\media\libstagefright
AudioSink.cpp
FileSource.cpp
MP3Extractor.cpp
OMXClient.cpp
OMXCodec.cpp

\frameworks\av\services\audioflinger
AudioFlinger.cpp
FastMixer.cpp
AudioPolicyService.cpp

\harwdware\qcom\audio\hal
audio_hw.c

\harwdware\qcom\audio\hal\msm8974
platform.c
hw_info.c
```

On the kernel side, logs on the following files are enabled to debug issues related to APSS and DSP communication:

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6afe.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6asm.c  +p > /sys/kernel/debug/dynamic_debug/control
```

For issues related to calibration, enable log messages in audio_acdb.c.

```
echo file audio_acdb.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging in q6asm.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

6.2.3 Log analysis

6.2.3.1 User space logs

A sound clip is played via SoundPool.

```
12-13 14:35:18.520   965  1152 V SoundPool: play sampleID=1,
leftVolume=0.501187, rightVolume=0.501187, priority=0, loop=0,
rate=1.000000
12-13 14:35:18.520   965  1152 V SoundPool: mState = 0 mChannelID=0,
mNumChannels=1, mPos = 0, mPriority=-1, mLoop=0
12-13 14:35:18.520   965  1152 V SoundPool: Allocated active channel
12-13 14:35:18.520   965  1152 V SoundPool: play channel 0x608f8738 state =
0
12-13 14:35:18.520   965  1152 V SoundPool: SoundChannel::play 0x608f8738:
sampleID=1, channelID=1, leftVolume=0.501187, rightVolume=0.501187,
priority=0, loop=0, rate=1.000000

12-13 14:35:18.520   217  4961 V AudioPolicyService: getOutput()
12-13 14:35:18.520   217  4961 V legacy_audio_policy_hal: audio_io_handle_t
android_audio_legacy::ap_get_output(audio_policy*, audio_stream_type_t,
uint32_t, audio_format_t, audio_channel_mask_t, audio_output_flags_t, const
audio_offload_info_t*): tid 4961
12-13 14:35:18.520   217  4961 V AudioPolicyManagerBase: getOutput() device
8, stream 1, samplingRate 0, format 0, channelMask 3, flags 0
12-13 14:35:18.520   217  4961 V AudioPolicyManagerBase: getOutput()
returns output 2
12-13 14:35:18.520   965  1152 V AudioSystem: getFrameCount() streamType 1,
output 2, frameCount 960
```

An AudioTrack instance is created with the AUDIO_OUTPUT_FLAG_FAST flag. Flags supported are at system/core/include/system/audio.h.

```
12-13 14:35:18.520   217  1072 V AudioPolicyManagerBase: getOutput() device
8, stream 1, samplingRate 48000, format 1, channelMask 3, flags 4
12-13 14:35:18.520   217  1072 V AudioPolicyManagerBase: selectOutput()
commonFlags for output 3, 0001
12-13 14:35:18.520   217  1072 V AudioPolicyManagerBase: getOutput()
returns output 3
12-13 14:35:18.520   965  1152 V AudioSystem: getLatency() streamType 1,
output 3, latency 50
12-13 14:35:18.520   965  1152 V AudioSystem: getFrameCount() streamType 1,
output 3, frameCount 960
12-13 14:35:18.520   965  1152 V AudioSystem: getOutputSamplingRate()
reading from output desc
12-13 14:35:18.520   965  1152 V AudioSystem: getSamplingRate() streamType
1, output 3, sampling rate 48000
12-13 14:35:18.520   965  1152 V AudioTrack: createTrack_l() output 3
afLatency 50
```

```
12-13 14:35:18.520   217   217 V AudioFlinger: createTrack() sessionId: 0
12-13 14:35:18.520   217   217 V AudioFlinger: createTrack() lSessionId: 17
12-13 14:35:18.520   217   217 V AudioFlinger: AUDIO_OUTPUT_FLAG_FAST
accepted: frameCount=4512 mFrameCount=240
12-13 14:35:18.530   965  1152 V SoundPool: setVolume 0x60afee48
```

AudioFlinger then starts the output stream, which causes the audio path to be set up and calibration to, be pushed for the output device.

```
12-13 14:35:18.530   217  1119 V AudioFlinger: start(4096), calling pid 965
session 17
12-13 14:35:18.530   217  1119 V AudioFlinger: ? => ACTIVE (4096) on thread
0xb7bdb7f8
12-13 14:35:18.530   217  1119 V AudioPolicyService: startOutput()
12-13 14:35:18.530   217  1119 V AudioPolicyManagerBase: startOutput()
output 3, stream 1, session 17
12-13 14:35:18.530   217  1119 V AudioPolicyManagerBase: getNewDevice()
selected device 8
12-13 14:35:18.530   217  1119 V AudioPolicyManagerBase: setOutputDevice()
output 3 device 0008 delayMs 0

12-13 14:35:18.650   217  1071 D audio_hw_primary: out_set_parameters:
enter: usecase(1: low-latency-playback) kvpairs: routing=8
12-13 14:35:18.650   217  1071 V audio_hw_primary: out_set_parameters:
exit: code(1)
12-13 14:35:18.650   217  1119 V AudioFlinger: signal playback thread
12-13 14:35:18.660   217   609 V AudioMixer: add track (1)
12-13 14:35:18.660   217   609 V AudioMixer:
AudioMixer::unprepareTrackForDownmix(1)
12-13 14:35:18.660   217   609 V AudioMixer:  nothing to do, no downmixer
to delete
12-13 14:35:18.660   217   609 V AudioMixer: setParameter(TRACK,
MAIN_BUFFER, 0xb7bf0450)
12-13 14:35:18.660   217   609 V AudioMixer: enable(1)
```

Device (8) is headphones defined in /hardware/qcom/audio/hal/msm8974/platform.c. Use case (0) is USECASE_AUDIO_PLAYBACK_LOW_LATENCY defined in hardware/qcom/audio/hal/audio_hw.h.

```
12-13 14:35:18.660   217   609 V audio_hw_primary: start_output_stream:
enter: usecase(1: low-latency-playback) devices(0x8)
12-13 14:35:18.660   217   609 V msm8974_platform:
platform_get_output_snd_device: enter: output devices(0x8)
12-13 14:35:18.660   217   609 V msm8974_platform:
platform_get_output_snd_device: exit: snd_device(headphones)
12-13 14:35:18.660   217   609 D audio_hw_primary: select_devices:
out_snd_device(4: headphones) in_snd_device(0: )
12-13 14:35:18.660   217   609 D hardware_info: hw_info_append_hw_type :
device_name = headphones
```

```
12-13 14:35:18.660   217   609 V audio_hw_primary: enable_snd_device:
snd_device(4: headphones)
```

The headphone device has the ACDB ID 10 associated with it. Calibration data for the headphones is sent to the DSP, as shown in the following log comment.

```
12-13 14:35:18.660   217   609 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 10, path =  0
12-13 14:35:18.660   217   609 D ACDB-LOADER: ACDB -> send_adm_topology
12-13 14:35:18.660   217   609 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
12-13 14:35:18.660   217   609 D ACDB-LOADER: ACDB -> send_audtable
12-13 14:35:18.660   217   609 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
12-13 14:35:18.660   217   609 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
12-13 14:35:18.660   217   609 D ACDB-LOADER: ACDB -> send_audvoltable
12-13 14:35:18.660   217   609 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
12-13 14:35:18.660   217   609 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
12-13 14:35:18.660   217   609 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
```

card_id (0) SOUND_CARD and device_id (15) is LOWLATENCY_PCM_DEVICE defined in hardware/qcom/audio/hal/msm8974/platform.h.

```
12-13 14:35:18.670   217   609 V audio_hw_primary: enable_audio_route:
enter: usecase(1)
12-13 14:35:18.670   217   609 V audio_hw_primary: enable_audio_route:
apply mixer path: low-latency-playback
12-13 14:35:18.670   217   609 V audio_hw_primary: enable_audio_route: exit
12-13 14:35:18.670   217   609 V audio_hw_primary: start_output_stream:
Opening PCM device card_id(0) device_id(15)
12-13 14:35:18.670   217   609 V audio_hw_primary: start_output_stream:
exit
```

Once playback is done, AudioFlinger removes the track and stops output, which causes the audio path to be torn down.

```
12-13 14:35:18.860   217  1071 V AudioFlinger: removeTracks_l removing
track on session 17
12-13 14:35:18.870   217  1071 V AudioPolicyService: stopOutput()

12-13 14:35:18.870   217   605 V AudioPolicyManagerBase: stopOutput()
output 3, stream 1, session 17
12-13 14:35:18.870   217   605 V AudioPolicyManagerBase: changeRefCount()
stream 1, count 0
12-13 14:35:18.870   217   605 V AudioPolicyManagerBase: getNewDevice()
selected device 0
```

```
12-13 14:35:18.870   217   605 V AudioPolicyManagerBase: setOutputDevice()
output 3 device 0000 delayMs 100
12-13 14:35:18.870   217   605 V AudioPolicyManagerBase: setOutputDevice()
prevDevice 0008
12-13 14:35:18.870   217   605 V AudioPolicyManagerBase: setOutputDevice()
setting same device 0000 or null device for output 3

12-13 14:35:19.650   965  1825 V SoundPool: stop
12-13 14:35:19.650   217   217 V AudioFlinger: stop(4096), calling pid 965
12-13 14:35:19.650   217   217 V AudioFlinger: not stopping/stopped =>
stopping/stopped (4096) on thread 0xb5b30008
12-13 14:35:19.650   965  1825 V SoundPool: done_l(1)
```

After 3 sec of playback completion, if there is no further activity, a standby function is called to tear down the audio path and reset routing.

```
12-13 14:35:23.550   217  1071 V AudioFlinger: Audio hardware entering
standby, mixer 0xb5b30008, suspend count 0
12-13 14:35:23.550   217  1071 V audio_hw_primary: out_standby: enter:
usecase(1: low-latency-playback)
12-13 14:35:23.600   217  1071 V audio_hw_primary: stop_output_stream:
enter: usecase(1: low-latency-playback)
12-13 14:35:23.600   217  1071 V audio_hw_primary: disable_audio_route:
enter: usecase(1)
12-13 14:35:23.610   217  1071 V audio_hw_primary: disable_audio_route:
reset mixer path: low-latency-playback
12-13 14:35:23.610   217  1071 V audio_hw_primary: disable_audio_route:
exit
12-13 14:35:23.610   217  1071 D hardware_info: hw_info_append_hw_type :
device_name = headphones
12-13 14:35:23.610   217  1071 V audio_hw_primary: disable_snd_device:
snd_device(4: headphones)
12-13 14:35:23.610   217  1071 V audio_hw_primary: stop_output_stream:
exit: status(0)
12-13 14:35:23.610   217  1071 V audio_hw_primary: out_standby: exit
```

6.2.3.2 Kernel logs

```
<7>[ 1318.140936] msm_pcm_routing_process_audio: reg 2 val 4 set 1
```

The PCM routing is to back-end DAI ID 2 (MSM_BACKEND_DAI_SLIMBUS_
0_RX,) from front-end DAI ID 4 (MSM_FRONTEND_DAI_MULTIMEDIA1\5). The back-end DAI IDs and FRONT END DAI IDs are mentioned in kernel/sound/soc/msm/
qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[ 1318.141285] msm_pcm_hw_params: perf: 1
<7>[ 1318.144630] event_handler:Payload = [0x10db3]stat[0x0]
<7>[ 1318.144652] msm_pcm_hw_params: session ID 1
```

```
<7>[ 1318.196810] msm_pcm_playback_prepare
<7>[ 1318.198020] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1318.198027] msm-pcm-routing msm-pcm-routing: reg 0 val 1
<7>[ 1318.198108] msm_pcm_playback_copy: prtd->out_count = 2
<7>[ 1318.198116] msm_pcm_playback_copy:fbytes =960: xfer=0 size=960
<7>[ 1318.198125] msm_pcm_playback_copy:fbytes = 0: xfer=960
<7>[ 1318.198134] msm_pcm_trigger: Trigger start
```

PCM data is copied to the DSP buffers. There are two buffers each of size 960 bytes.

PCM data is copied to the DSP buffers. There are two buffers each of size 960 bytes.

```
<7>[ 1318.198108] msm_pcm_playback_copy: prtd->out_count = 2
<7>[ 1318.198116] msm_pcm_playback_copy:fbytes =960: xfer=0 size=960
<7>[ 1318.198125] msm_pcm_playback_copy:fbytes = 0: xfer=960

<7>[ 1318.198188] msm_pcm_playback_copy: prtd->out_count = 1
<7>[ 1318.198195] msm_pcm_playback_copy:fbytes =960: xfer=0 size=960
<7>[ 1318.198202] msm_pcm_playback_copy:fbytes = 0: xfer=960

<7>[ 1318.198919] event_handler:writing 960 bytes of buffer to dsp
<7>[ 1318.198940] event_handler:writing 960 bytes of buffer to dsp
```

Once the buffer is consumed, the pcm_irq_pos increments by period size, which is 960.

```
<7>[ 1318.201531] ASM_DATA_EVENT_WRITE_DONE_V2
<7>[ 1318.201537] Buffer Consumed = 0x7ee70000
<7>[ 1318.201543] pcm_irq_pos = 960
<7>[ 1318.201569] msm_pcm_playback_copy: prtd->out_count = 1
<7>[ 1318.201579] msm_pcm_playback_copy:fbytes =960: xfer=0 size=960
<7>[ 1318.201587] msm_pcm_playback_copy:fbytes = 0: xfer=960
<7>[ 1318.201594] msm_pcm_playback_copy:writing 960 bytes of buffer to dsp

<7>[ 1318.205887] ASM_DATA_EVENT_WRITE_DONE_V2
<7>[ 1318.205893] Buffer Consumed = 0x7ee703c0
<7>[ 1318.205899] pcm_irq_pos = 0
<7>[ 1318.205925] msm_pcm_playback_copy: prtd->out_count = 1
<7>[ 1318.205935] msm_pcm_playback_copy:fbytes =960: xfer=0 size=960
<7>[ 1318.205942] msm_pcm_playback_copy:fbytes = 0: xfer=960
<7>[ 1318.205947] msm_pcm_playback_copy:writing 960 bytes of buffer to dsp
.
.
.
<7>[ 1323.024685] SNDRV_PCM_TRIGGER_STOP
```

Playback gets stopped by sending SNDRV_PCM_TRIGGER_STOP.

```
<7>[ 1323.024776] msm_pcm_playback_close: cmd_pending 0x8
```

```
<7>[ 1323.025915] ASM_DATA_EVENT_WRITE_DONE_V2
<7>[ 1323.025931] Buffer Consumed = 0x7ee703c0
<7>[ 1323.030913] ASM_DATA_EVENT_WRITE_DONE_V2
<7>[ 1323.030928] Buffer Consumed = 0x7ee70000
<7>[ 1323.039729] ASM_DATA_EVENT_RENDERED_EOS
```

All the buffers sent to DSP are consumed and DSP sends an EOS event.

```
<7>[ 1323.078743] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1323.079695] msm_pcm_routing_process_audio: reg 2 val 4 set 0
```

PCM routing from front-end DAI to BACK END DAI is disabled.

6.2.4 Customization

There is no customization.

6.2.5 Debugging

See Section 6.3.

6.3 Debugging

Table 8-1 shows common playback issues that were encountered during previous integration activities and some troubleshooting steps to resolve them.

Table 6-1 Debugging and troubleshooting tips

Issue type	Symptoms	Troubleshooting steps
Audio mute	No audio is heard from the output device	1. Check the software volume, DSP volume, WCD codec digital, and analog gain. 2. Check that data logged at the AFE (DSP output) is proper using QXDM Professional logging. 3. Check mixer controls to verify if the codec path and DSP routing are correct for the specific output device.
Noise	Noise at the start, end, or pause/resume of playback	Check if DSP buffers are not flushed or if the flush command is failing. Logging PCM data at the AFE using QXDM Professional can help narrow down the issue.
	PoP noise when disabling/enabling device	Check that the bringup or power-down sequence for the QCD codec is correct.
	Noise when changing audio effects	Check PCM data before and after the effects module to determine if the noise is introduced in the effects module. Such issues are seen during LPA playback.

Issue type	Symptoms	Troubleshooting steps
	Continuous background noise during playback	1. Check if PCM data at the AFE is correct using QXDM Professional. 2. Capture codec-related logs to isolate codec issues. 3. Hardware teams may need to be involved to root cause such issues.
Volume	Volume is not at the value expected or changes unexpectedly, for example, during concurrency scenarios	Check user space logs, calibration data for output device used, and audio effects applied.
Discontinuous audio	Audio playback contains gaps	Check PCM data at the user space and DSP output. If the gaps are isolated to the user space, buffer management logic, and CPU loading are analyzed.
Configuration	Audio is either slow or fast	1. Check user space and kernel logs for sample rate settings. 2. Check for channel misconfiguration from user space and kernel logs, PCM data at AFE, and WCD register settings.
Track management	Audio repetition	Check PCM data at the user space and DSP output. Such issues are caused due to buffer management or seek operation errors.
	Seeking does not work as expected	1. Check user space implementation that handles DSP timestamp information. 2. Check user space logs and kernel logs for DSP timestamps to help isolate the issue.
	Progress bar does not move with playback	1. Check user space implementation that handles DSP timestamp information. 2. Check whether issue is seen with a specific type of playback (normal, LPA, or Tunnel).
	Issue when switching from one song to another	1. Check if the playback state is bad. 2. Check user space logs.
Codec	Media formats not supported	1. Check if codec chosen to decode the bit stream is correct. Sometimes the parser may not be able to detect the type of bit stream and chooses a wrong codec. 2. Check if switching between hardware and software codec helps resolve the issue.
Performance	Playback does not start immediately, resuming playback takes time, routing audio to a different device takes time, and so on.	1. Carefully examine log timestamps to determine which component or step is lagging. 2. Taking power-top logs can help further analyze the issue.
Power consumption	Power consumption is higher than expected	Check clock dumps for clocks enabled and the frequency of the clocks. Power rail breakdown can pinpoint exactly the hardware block consuming extra power.
Stability	Crash during stress testing	Check the stack trace captured in logs to isolate the issue.

7 Audio recording

QTI has enabled various modes of audio recording in Android:

- Normal recording – The default recording mode that is supported that can use software or hardware encoders based on hardware capabilities.
- Tunnel mode recording: Encoding is performed by a hardware encoder and the DSP sends large buffers to the user space after enough recording media is accumulated. Presently, on QTI chipsets, only AMR WB recording is supported in this mode.

7.1 Stereo recording

Figure 7-1 shows the paths taken by the control and data flows through the user space, kernel, and DSP components involved in media playback on Android.

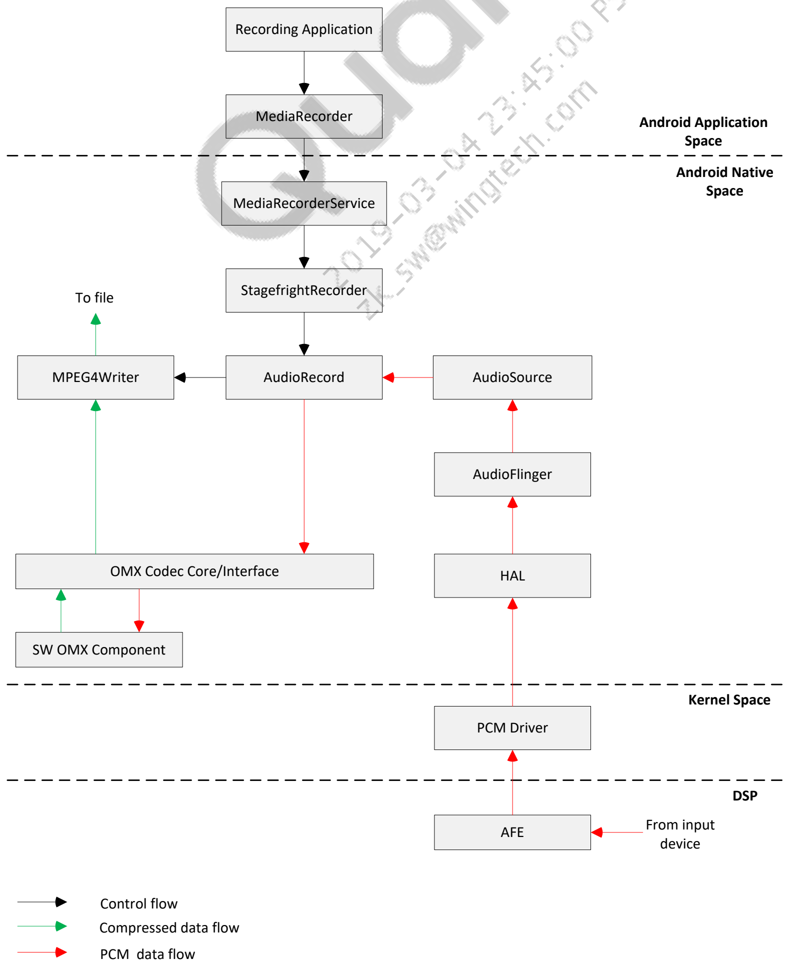


Figure 7-1 Android media recording components

The MediaPlayerService is a system service that links the Android user space to the media framework. Stagefright is the native Android media framework that facilitates media playback and capture through various physical and proxy devices. Other component functions are:

- MPEG4Writer is responsible for writing all of the encoded data to file when recording ends.
- The OMX layer is used for decoding/encoding data to and from raw PCM data into appropriate audio formats as supported by the system.
- AudioRecord manages the entire recording process by getting PCM data from the input device, encoding, and writing it to file.

7.1.1 Call flow

7.1.1.1 Starting record

Figure 7-2 shows the call and data flow during initialization and active recording in normal mode.

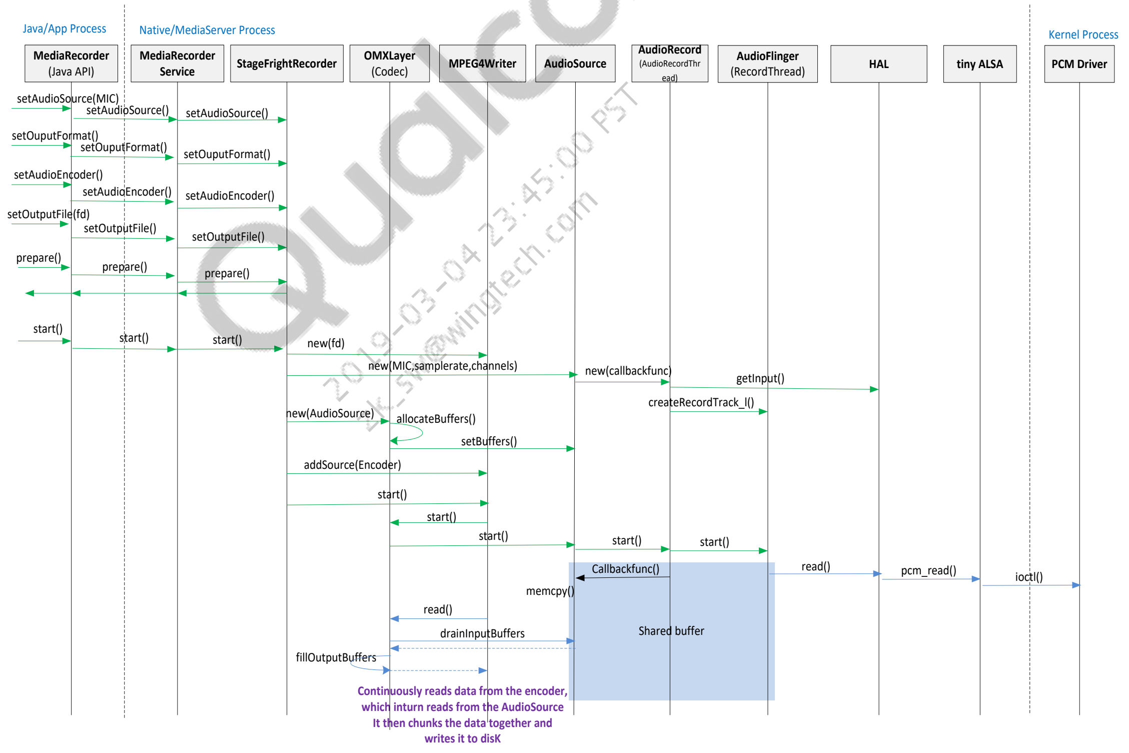


Figure 7-2 Normal recording call flow

7.1.2 Log collection

This section highlights the various files and/or components in which logging is enabled to troubleshoot issues related to audio recording on Android.

7.1.2.1 User space logs

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

7.1.2.2 Kernel logs

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file msm-pcm-q6-v2.c +p > /sys/kernel/debug/dynamic_debug/control
echo file msm-pcm-routing-v2.c +p > /sys/kernel/debug/dynamic_debug/control
```

The encoder file where logging is enabled changes with the type of encoder.

```
echo -n "file aac_in.c  +p" > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging in msm-pcm-q6-v2.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

7.1.2.3 OMX layer logs

Depending on the OMX component used, logging in the respective OMX component is enabled.

If there is an issue in the QTI AAC encoder OMX component, the following logs are enabled by changing DEBUG_PRINT to LOGE and DEBUG_DETAIL to LOGE in the hardware/qcom/audio/mm-audio/aenc-aac/qdsp6/inc/aenc_svr.h file.

```
#define DEBUG_PRINT_ERROR LOGE
#define DEBUG_PRINT       LOGE /*LOGI*/
#define DEBUG_DETAIL      LOGE /* LOGV */
```

7.1.2.4 Additional logging

In the user space, enable logs in files listed at the following locations, if the facility exists:

In the user space, enable logs on files listed at the following locations.

```
\frameworks\av\media\libmediaplayerservice
MediaPlayerService.cpp
StagefrightRecorder.cpp

\frameworks\av\media\libmedia
AudioSystem.cpp
AudioRecord.cpp
mediarecorder.cpp
IOMX.cpp
```

```
\frameworks\av\media\libstagefright
AudioSource.cpp
MPEG4Writer.cpp
OMXClient.cpp
OMXCodec.cpp

\frameworks\av\services\audioflinger
AudioFlinger.cpp
AudioMixer.cpp
AudioPolicyService.cpp

\harwdware\qcom\audio\hal
audio_hw.c

\harwdware\qcom\audio\hal\msm8974
platform.c
hw_info.c

\harwdware\qcom\audio\hal\policy_hal
AudioPolicyManager.cpp

\harwdware\libhardware_legacy\audio
AudioPolicyManagerBase.cpp
```

On the kernel side, logs on the following files are enabled to debug issues related to APSS and DSP communication.

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6afe.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6asm.c  +p > /sys/kernel/debug/dynamic_debug/control
```

For issues related to calibration, enable log messages in audio_acdb.c.

```
echo file audio_acdb.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging on q6asm.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

7.1.2.5 QXDM Professional logs

This section is not applicable to this release.

7.1.3 Log analysis

7.1.3.1 User space logs

In the Android Native layer, the recording flow is initiated via MediaPlayerService. The MediaRecorder component creates a StagefrightRecorder instance.

```
12-13 15:48:55.572 14853 14853 V MediaRecorder: constructor
12-13 15:48:55.582   215   215 V StagefrightRecorder: Constructor
```

The following MediaRecorder APIs are called in sequence to initialize the recording session. Audio source 1 is the mic, defined in AudioSystemLegacy.h.

```
12-13 15:48:55.642   215  1758 V MediaRecorderService: setAudioSource(1)
12-13 15:48:55.642   215  1758 V StagefrightRecorder: setAudioSource: 1

12-13 15:48:55.642 14853 14853 V MediaRecorder: setParameters(audio-param-
sampling-rate=48000)
12-13 15:48:55.642   215  1120 V StagefrightRecorder: setParameters: audio-
param-sampling-rate=48000
```

Output format 1 is 3GPP, defined in mediarecorder.h.

```
12-13 15:48:55.642   215  1099 V MediaRecorderService: setOutputFormat(1)
12-13 15:48:55.642   215  1099 V StagefrightRecorder: setOutputFormat: 1
```

Encoder 3 is AAC, defined in mediarecorder.h.

```
12-13 15:48:55.642 14853 14853 V MediaRecorder: setAudioEncoder(3)
12-13 15:48:55.642   215  1758 V StagefrightRecorder: setAudioEncoder: 3

12-13 15:48:55.652 14853 14853 V MediaRecorder: setOutputFile(54, 0, 0)
12-13 15:48:55.652   215  1120 V StagefrightRecorder: setOutputFile: 27, 0,
0
```

Recording is started after a prepare() command is issued from the MediaRecorder.

```
12-13 15:48:55.652 14853 14853 V MediaRecorder: prepare
12-13 15:48:55.652 14853 14853 V MediaRecorder: start
```

StagefrightRecorder now creates AudioSource, AudioRecord (which internally creates AudioTrack), and MPEG4Writer (does not print log markers) instances.

```
12-13 15:48:55.652   215  1758 V AudioSource: sampleRate: 48000,
channelCount: 2
```

```
12-13 15:48:55.652   215  1758 V AudioRecord: sampleRate 48000, channelMask
0xc, format 1
12-13 15:48:55.652   215  1758 V AudioRecord: set(): mSessionId 21
```

AudioRecord then opens an input corresponding to the input device.

```
12-13 15:48:55.652   215  1758 V AudioPolicyManager:
getDeviceForInputSource()input source 1, device 80000004
12-13 15:48:55.652   215  1758 V AudioPolicyManager: getInput() inputSource
1, samplingRate 48000, format 1, channelMask c, acoustics 0
12-13 15:48:55.652   215  1758 V AudioFlinger: openInput() openInputStream
returned input 0xb8547028, SamplingRate 48000, Format 1, Channels c, status
0
12-13 15:48:55.652   215  1758 V AudioFlinger: openInput() created record
thread: ID 22 thread 0xb8548a18
```

The AAC encoder that is used to encode the input PCM stream is instantiated next; in this case, it is a software encoder.

```
12-13 15:48:55.662   215  1758 V OMXCodec: matching
'OMX.google.aac.encoder' quirks 0x00000000
12-13 15:48:55.662   215  1758 V OMXCodec: Attempting to allocate OMX node
'OMX.google.aac.encoder'
12-13 15:48:55.662   215  1758 V SoftOMXPlugin: makeComponentInstance
'OMX.google.aac.encoder'
12-13 15:48:55.672   215  1758 V SoftAACEncoder2: setAudioParams: 44100 Hz,
1 channels, 0 bps
12-13 15:48:55.672   215  1758 V OMXCodec: Successfully allocated OMX node
'OMX.google.aac.encoder'
12-13 15:48:55.672   215  1758 V OMXCodec: configureCodec protected=0
12-13 15:48:55.672   215  1758 V SoftAACEncoder2: setAudioParams: 48000 Hz,
2 channels, 96000 bps
```

Input and output buffers are allocated for the encoder to use, after which it moves to an idle state.

```
12-13 15:48:55.682   215  1758 I MPEG4Writer: limits: 2147483647/0
bytes/us, bit rate: 96000 bps and the estimated moov size 3072 bytes
12-13 15:48:55.682   215  1758 V MPEG4Writer: startWriterThread
12-13 15:48:55.682   215  1758 V MPEG4Writer: initTrackingProgressStatus
12-13 15:48:55.682   215  1758 V OMXCodec: [OMX.google.aac.encoder]
allocating 4 buffers of size 4096 on input port
12-13 15:48:55.682   215  1758 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb85d68a8 on input port
12-13 15:48:55.682   215  1758 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb85d6a60 on input port
12-13 15:48:55.682   215  1758 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb85d6b30 on input port
12-13 15:48:55.682   215  1758 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb85d6be8 on input port
```

```
12-13 15:48:55.682   215  1758 V OMXCodec: [OMX.google.aac.encoder]
allocating 4 buffers of size 8192 on output port
12-13 15:48:55.682   215  1758 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb85d6d70 on output port
12-13 15:48:55.682   215  1758 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb85d6f80 on output port
12-13 15:48:55.682   215  1758 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb85d70c0 on output port
12-13 15:48:55.682   215  1758 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb85d71e8 on output port
12-13 15:48:55.682   215 14945 V SimpleSoftOMXComponent: msgType = 0
12-13 15:48:55.682   215 14945 V OMX     : OnEvent(0, 0, 2)
12-13 15:48:55.682   215 14946 V OMXCodec: [OMX.google.aac.encoder]
onStateChange 2
12-13 15:48:55.682   215 14946 V OMXCodec: [OMX.google.aac.encoder] Now
Idle.
```

The encoder moves into the running state after some media configuration is performed.

```
12-13 15:48:55.682   215 14945 V OMX     : OnEvent(0, 0, 3)
12-13 15:48:55.682   215 14946 V OMXCodec: [OMX.google.aac.encoder]
onStateChange 3
12-13 15:48:55.682   215 14946 V OMXCodec: [OMX.google.aac.encoder] Now
Executing.
```

The encoder informs the framework on completion and the recording process is kicked off. The AudioRecord and RecordTrack threads are started.

```
12-13 15:48:55.682   215  1758 V AudioRecord: start, sync event 0 trigger
session 0
12-13 15:48:55.682   215  1758 V AudioRecord: mAudioRecord->start()

12-13 15:48:55.682   215  1758 V AudioFlinger: RecordHandle::start()
12-13 15:48:55.682   215  1758 V AudioFlinger: RecordThread::start event 0,
triggerSession 0
```

The input opened previously is now started which causes the audio path to be set up and calibration for the input device to be pushed.

```
12-13 15:48:55.682   215  1758 V AudioPolicyManagerBase: startInput() input
22
12-13 15:48:55.682   215  1758 V AudioPolicyManager:
getDeviceForInputSource()input source 1, device 80000004
12-13 15:48:55.682   215  1758 V AudioPolicyManagerBase:
AudioPolicyManager::startInput() input source = 1
12-13 15:48:55.682   215 14937 V AudioFlinger: RecordThread: loop starting
12-13 15:48:55.682   215  1758 V AudioFlinger: Signal record thread
12-13 15:48:55.682   215 14947 V MPEG4Writer: findChunkToWrite
```

```
12-13 15:48:55.682   215 14947 V MPEG4Writer: Nothing to be written after
all
12-13 15:48:55.682   215 14937 V msm8974_platform:
platform_update_usecase_from_source: input source :1
```

Input device is handset-stereo-dmic-ef defined in /hardware/qcom/audio/hal/msm8974/platform.c. Use case (0) is USECASE_AUDIO_RECORD defined in hardware/qcom/audio/hal/audio_hw.h.

```
12-13 15:48:55.682   215 14937 V audio_hw_primary: start_input_stream:
enter: usecase(6)
12-13 15:48:55.682   215 14937 V audio_hw_primary: start_input_stream:
usecase(6)
12-13 15:48:55.682   215 14937 V msm8974_platform:
platform_get_input_snd_device: enter: out_device(0) in_device(0x4)
12-13 15:48:55.682   215 14937 V msm8974_platform:
platform_get_input_snd_device: exit: in_snd_device(handset-stereo-dmic-ef)
12-13 15:48:55.682   215 14937 D hardware_info: hw_info_append_hw_type :
device_name = handset-stereo-dmic-ef
12-13 15:48:55.682   215 14937 V audio_hw_primary: enable_snd_device:
snd_device(65: handset-stereo-dmic-ef)
```

The mic device has the ACDB ID 34 associated with it. Calibration data for the mic is sent to the DSP, as shown in the following log comment.

```
12-13 15:48:55.682   215 14937 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 34, path =  1
12-13 15:48:55.682   215 14937 D ACDB-LOADER: ACDB -> send_adm_topology
12-13 15:48:55.682   215 14937 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
12-13 15:48:55.682   215 14937 D ACDB-LOADER: ACDB -> send_audtable
12-13 15:48:55.682   215 14937 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
12-13 15:48:55.682   215 14937 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
12-13 15:48:55.682   215 14937 D ACDB-LOADER: ACDB -> send_audvoltable
12-13 15:48:55.682   215 14937 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
12-13 15:48:55.682   215 14937 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
12-13 15:48:55.682   215 14937 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
```

card_id (0) SOUND_CARD and device_id (0) is AUDIO_RECORD_PCM_DEVICE defined in hardware/qcom/audio/hal/msm8974/platform.h.

```
12-13 15:48:55.692   215 14937 V audio_hw_primary: start_input_stream:
Opening PCM device card_id(0) device_id(0), channels 2
```

Input PCM data is read from AudioStreamInALSA and is handed to AudioSource via AudioTrack.

```
12-13 15:48:55.792   215  1758 V AudioFlinger: Record started OK
12-13 15:48:55.792   215 14938 V AudioRecord: obtainBuffer(480) returned
960 = 480 + 480
12-13 15:48:55.792   215 14938 V AudioRecord: obtainBuffer(480) returned
480 = 480 + 0
12-13 15:48:55.792   215 14950 V AudioSource: signalBufferReturned:
0xb85d72b0
```

The OMX layer reads this data and sends it to the encoder instance by calling emptyBuffer(). The encoder uses EmptyBufferDone() CB to inform the OMX core that the buffer is used. This action is performed in a loop for all input buffers.

```
12-13 15:48:55.792   215 14950 V AudioSource: signalBufferReturned:
0xb85d72b0
12-13 15:48:55.792   215 14950 V OMXCodec: [OMX.google.aac.encoder] Calling
emptyBuffer on buffer 0xb85d68a8 (length 1920), timestamp 123213 us (0.12
secs)

12-13 15:48:55.792   215 14950 V AudioSource: signalBufferReturned:
0xb85d7a38
12-13 15:48:55.792   215 14950 V OMXCodec: [OMX.google.aac.encoder] Calling
emptyBuffer on buffer 0xb85d6a60 (length 1920), timestamp 133213 us (0.13
secs)
```

The OMX layer then reads encoded data from the encoder instance by calling fillBuffer(). The encoder uses FillBufferDone() CB to inform the OMX core that the buffer is filled. This action is performed in a loop for all input buffers.

```
12-13 15:48:55.812   215 14950 V OMXCodec: [OMX.google.aac.encoder] Calling
fillBuffer on buffer 0xb85d6d70
12-13 15:48:55.812   215 14945 V OMX     : OnFillBufferDone
buffer=0xb85d6d70

12-13 15:48:55.812   215 14950 V OMXCodec: [OMX.google.aac.encoder] Calling
fillBuffer on buffer 0xb85d6f80
12-13 15:48:55.812   215 14945 V OMX     : OnFillBufferDone
buffer=0xb85d6f80
```

The read() call from the encoder returns and MPEG4Writer pulls the encoded data. It packages the data into chunks that are written to file at the end. The read() call is done in a loop and MPEG4Writer is blocked until more data is made available by the encoder.

```
12-13 15:48:55.852   215 14938 V AudioSource: dataCallbackTimestamp:
5734988580 us
```

When the recording is stopped, StagefrightRecorder, MPEG4Writer, encoders, and other components stop too.

```
12-13 15:49:01.212 14853 14853 V MediaRecorder: stop
12-13 15:49:01.212   215   215 V StagefrightRecorder: stop

12-13 15:49:01.212   215   215 D MPEG4Writer: Stopping Audio track
12-13 15:49:01.212   215 14950 V MPEG4Writer: Audio media time stamp:
5375979 and previous paused duration 123213
12-13 15:49:01.212   215 14950 V MPEG4Writer: Audio
timestampUs/lastTimestampUs: 5375979/5354646

12-13 15:49:01.222   215 14950 I MPEG4Writer: Received total/0-length
(254/0) buffers and encoded 253 frames. - audio
12-13 15:49:01.222   215 14950 V MPEG4Writer: getDriftTimeUs: 0 us
12-13 15:49:01.222   215 14950 I MPEG4Writer: Audio track drift time: 0 us
12-13 15:49:01.222   215   215 D MPEG4Writer: Stopping Audio track source
```

The encoder power-down is initiated, and it flushes all the data in its buffers as part of the process.

```
12-13 15:49:01.222   215   215 V OMXCodec: [OMX.google.aac.encoder] stop
mState=4
12-13 15:49:01.232   215   215 V OMXCodec: [OMX.google.aac.encoder]
stopOmxComponent_l mState=4

12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder]
FILL_BUFFER_DONE(buffer: 0xb85d6d70, size: 0, flags: 0x00000000, timestamp:
5477859 us (5.48 secs))

12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder]
FILL_BUFFER_DONE(buffer: 0xb85d6f80, size: 0, flags: 0x00000000, timestamp:
5499192 us (5.50 secs))

12-13 15:49:01.242   215 14945 V OMX     : OnEvent(0, 0, 2)
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder]
onStateChange 2
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder] Now
Idle..
```

The encoder then releases all its input and output buffers and moves to a Loaded state.

```
09-02 03:29:01.569   283  3371 E QC_AACENC: Inside
omx_aac_aenc::release_done
09-02 03:29:01.569   283  3371 E QC_AACENC: SCP--> Idle to Loaded-Pending
09-02 03:29:01.569   283  3371 E QC_AACENC: posting sem_States
```

```
09-02 03:29:01.569   283  3373 E QC_AACENC: send_command : recieved state
after semwait
09-02 03:29:01.569   283  3373 V OMXCodec: [OMX.qcom.audio.encoder.aac]
freeing buffer 0xb7e7bc48 on port 0
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb85d6be8 on port 0
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb85d6b30 on port 0
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb85d6a60 on port 0
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb85d68a8 on port 0
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb85d71e8 on port 1
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb85d70c0 on port 1
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb85d6f80 on port 1
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb85d6d70 on port 1
12-13 15:49:01.242   215 14945 V SimpleSoftOMXComponent: msgType = 0

12-13 15:49:01.242   215 14945 V OMX     : OnEvent(0, 0, 1)
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder]
onStateChange 1
12-13 15:49:01.242   215 14946 V OMXCodec: [OMX.google.aac.encoder] Now
Loaded.
```

AudioRecord and RecordTrack threads are stopped next.

```
12-13 15:49:01.242   215   215 V AudioFlinger: RecordHandle::stop()
12-13 15:49:01.242   215   215 V AudioFlinger: RecordThread::stop
```

The device and modifier are disabled.

```
12-13 15:49:01.302   215 14937 V audio_hw_primary: stop_input_stream:
enter: usecase(6: audio-record)
12-13 15:49:01.302   215 14937 V audio_hw_primary: disable_audio_route:
enter: usecase(6)
12-13 15:49:01.302   215 14937 V audio_hw_primary: disable_audio_route:
reset mixer path: audio-record
12-13 15:49:01.302   215 14937 D hardware_info: hw_info_append_hw_type :
device_name = handset-stereo-dmic-ef
12-13 15:49:01.302   215 14937 V audio_hw_primary: disable_snd_device:
snd_device(65: handset-stereo-dmic-ef)
12-13 15:49:01.322   215 14937 V audio_hw_primary: stop_input_stream: exit:
status(0)
```

Device routing that was set up previously is reset.

```
12-13 15:49:01.322   215   215 V AudioFlinger: Record stopped OK
12-13 15:49:01.322   215   215 V AudioPolicyManagerBase: stopInput() input
22
```

MPEG4Writer is stopped, and it writes all of the media chunks it received previously.

```
12-13 15:49:01.332   215   215 V OMXCodec: [OMX.google.aac.encoder] stopped
in state 1
12-13 15:49:01.332   215   215 D MPEG4Writer: Audio track stopped
12-13 15:49:01.332   215   215 D MPEG4Writer: Stopping writer thread
12-13 15:49:01.332   215 14947 V MPEG4Writer: writeAllChunks
12-13 15:49:01.332   215 14947 V MPEG4Writer: findChunkToWrite
12-13 15:49:01.332   215 14947 V MPEG4Writer: Nothing to be written after
all
12-13 15:49:01.332   215 14947 D MPEG4Writer: 0 chunks are written in the
last batch
12-13 15:49:01.332   215   215 D MPEG4Writer: Writer thread stopped
12-13 15:49:01.332   215   215 V MPEG4Writer: writeCompositionMatrix
12-13 15:49:01.332   215   215 V MPEG4Writer: Audio track time scale: 48000
12-13 15:49:01.332   215   215 V MPEG4Writer: writeCompositionMatrix
12-13 15:49:01.332   215   215 D MPEG4Writer: Stopping Audio track
```

The encoder threads are stopped and its instance released.

```
12-13 15:49:01.332   215   215 V OMXNodeInstance: calling
destroyComponentInstance
12-13 15:49:01.332   215   215 V OMXNodeInstance: destroyComponentInstance
returned err 0
12-13 15:49:01.332   215   215 V OMXNodeInstance: OMXNodeInstance going
away.
```

The input stream and mic device are closed.

```
12-13 15:49:01.332   215   215 V AudioFlinger: RecordThread::stop
12-13 15:49:01.332   215   215 V AudioFlinger: closeInput() 22
12-13 15:49:01.342   215 14937 V AudioFlinger: RecordThread 0xb8548a18
exiting
12-13 15:49:01.342   215   215 V audio_hw_primary: adev_close_input_stream
 android::AudioFlinger::RecordThread::RecordTrack::~RecordTrack()
12-13 15:49:01.342   215   215 V AudioFlinger: removeClient_l() pid 14853,
calling pid 14853
12-13 15:49:01.342   215   215 V AudioFlinger: purging stale effects
```

Framework recorder instances are released.

```
12-13 15:49:01.352   215  1758 V StagefrightRecorder: Destructor
12-13 15:49:01.352   215  1758 V StagefrightRecorder: stop
12-13 15:49:01.352 14853 14853 V MediaRecorder: destructor
```

7.1.3.2 Kernel logs

PCM path from BACKEND DAI ID 3 (MSM_BACKEND_DAI_SLIMBUS_0_TX) to FRONTEND DAI ID 0 (MSM_FRONTEND_DAI_MULTIMEDIA1) is set up first. The BACKEND DAI IDs and FRONT END DAI IDs are defined in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[ 5734.833855] msm_pcm_routing_process_audio: reg 3 val 0 set 1
<7>[ 5734.834152] msm_pcm_hw_params: perf: 0
<7>[ 5734.834158] msm_pcm_hw_params Opening 2-ch PCM read stream
<7>[ 5734.834164] __q6asm_open_read:session[1]
```

Recording is started by setting the hardware parameters and calling msm_pcm_capture_prepare followed by the msm_pcm_trigger() function. PCM data is copied from DSP buffers. There are 16 buffers each of size 4096 bytes.

```
<7>[ 5734.837409] msm_pcm_hw_params: session ID 1
<7>[ 5734.837421] q6asm_audio_client_buf_alloc_contiguous:
session[1]bufsz[3840]bufcnt[2]
<7>[ 5734.837552] q6asm_memory_map_regions: Session[1]
<7>[ 5734.837805] q6asm_memory_map_regions: i=0, bufadd[i] = 0x7ee70000,
maphdl[i] = 0xf09ba3f8
<7>[ 5734.837811] q6asm_memory_map_regions: i=1, bufadd[i] = 0x7ee70f00,
maphdl[i] = 0xf09ba3f8

<7>[ 5734.906823] msm_pcm_capture_prepare
<7>[ 5734.906830] Samp_rate = 48000
<7>[ 5734.906834] Channel = 2
<7>[ 5734.910126] msm_pcm_trigger: Trigger start
<7>[ 5734.910134] session[1]
```

The pcm_irq_pos value increments by the size of a buffer each time a READ returns.

```
<7>[ 5734.936802] ASM_DATA_EVENT_READ_DONE_V2
<7>[ 5734.936818] pcm_irq_pos=3840
<7>[ 5734.937047] msm_pcm_capture_copy
<7>[ 5734.937053] appl_ptr 0
<7>[ 5734.937057] hw_ptr 960
<7>[ 5734.937089] idx = 0
<7>[ 5734.937124] msm_pcm_capture_copy:fbytes = 0: size=0: xfer=3840
<7>[ 5734.937129]  Sending next buffer to dsp
<7>[ 1199.097007] q6asm_read:session[3]dsp-
buf[0][f03e0000]cpu_buf[1][3f285000]
```

```
<7>[ 1199.097059] Returning from capture_copy... 0

<7>[ 5734.956772] ASM_DATA_EVENT_READ_DONE_V2
<7>[ 5734.956781] pcm_irq_pos=7680
<7>[ 5734.956818] msm_pcm_capture_copy
<7>[ 5734.956824] appl_ptr 960
<7>[ 5734.956828] hw_ptr 1920
<7>[ 5734.956860] idx = 1
<7>[ 5734.956895] msm_pcm_capture_copy:fbytes = 0: size=0: xfer=3840
<7>[ 5734.956900]  Sending next buffer to dsp
<7>[ 5734.956915] q6asm_read:session[1]dsp-
buf[1][c61f8f00]cpu_buf[0][7ee70000]
<7>[ 1199.117795] Returning from capture_copy... 0
```

Capture gets stopped by sending SNDRV_PCM_TRIGGER_STOP.

```
<7>[ 5740.397236] SNDRV_PCM_TRIGGER_STOP
<7>[ 5740.397262] msm_pcm_capture_close
```

The DSP stops filling the buffers and returns them to the driver (pcm_ireq_pos stays the same). The driver frees all the buffers.

```
<7>[ 5740.397286] __q6asm_cmd:CMD_CLOSE
<7>[ 5740.397697] ASM_DATA_EVENT_READ_DONE_V2
<7>[ 5740.397713] pcm_irq_pos=0
<7>[ 5740.397774] ASM_DATA_EVENT_READ_DONE_V2
<7>[ 5740.397790] pcm_irq_pos=0
```

DSP buffers are released.

```
<7>[ 5740.399604] q6asm_memory_unmap: Session[1]
<7>[ 5740.399616] q6asm_add_mmaphdr:pkt size=24 cmd_flg=1
<7>[ 5740.399627] q6asm_memory_unmap: Found the element
<7>[ 5740.399639] q6asm_memory_unmap: mem_unmap-mem_map_handle: 0xf09ba3f8
<7>[ 5740.399831] q6asm_srvc_callback:ptr0[0x10d94]ptr1[0x0]opcode[0x110e8]
token[0x101]payload_s[8] src[0] dest[0]sid[1]dir[0]
<7>[ 5740.399847] q6asm_srvc_callback:Payload = [0x10d94] status[0x0]
<7>[ 5740.399868] q6asm_srvc_callback:Payload = [0x10d94] status[0x0]
<7>[ 5740.399907]
q6asm_audio_client_buf_free_contiguous:data[c61f8000]phys[7ee70000][f1c39a8
0] , client[f2fe4b80] handle[f362bdc0]
<7>[ 5740.400009] q6asm_audio_client_free: Session id 1
<7>[ 5740.400027] q6asm_session_free: sessionid[1]
```

PCM routing from FRONTEND DAI to BACK END DAI is disabled.

```
<7>[ 5740.444761] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 5740.446562] msm_pcm_routing_process_audio: reg 3 val 0 set 0
```

7.1.4 Customization

This section is not applicable to this release.

7.1.5 Debugging

See Section 7.4.

7.2 Surround Sound recording

Figure 7-3 shows the paths taken by the control and data flows through the user space, kernel, and DSP components involved in multichannel media capture on Android.

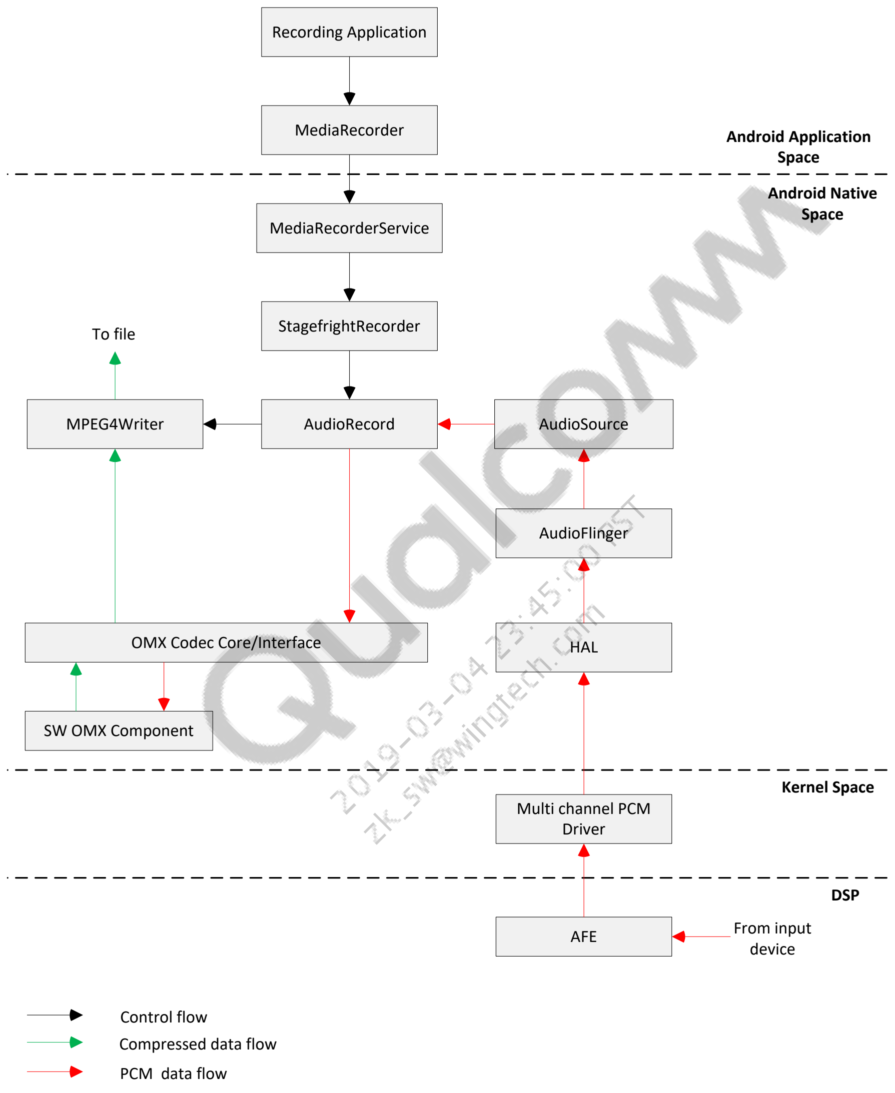


Figure 7-3 Android media recording components

Surround Sound recording involves most of the same components as the regular recording case except the PCM driver. In this instance, the multichannel PCM driver is used. When the data path is set up through the audio stack, configuration for multiple channels (as opposed to the default stereo) is used.

7.2.1 Starting record

Figure 7-4 shows the call and data flow during initialization and active recording in Normal mode.

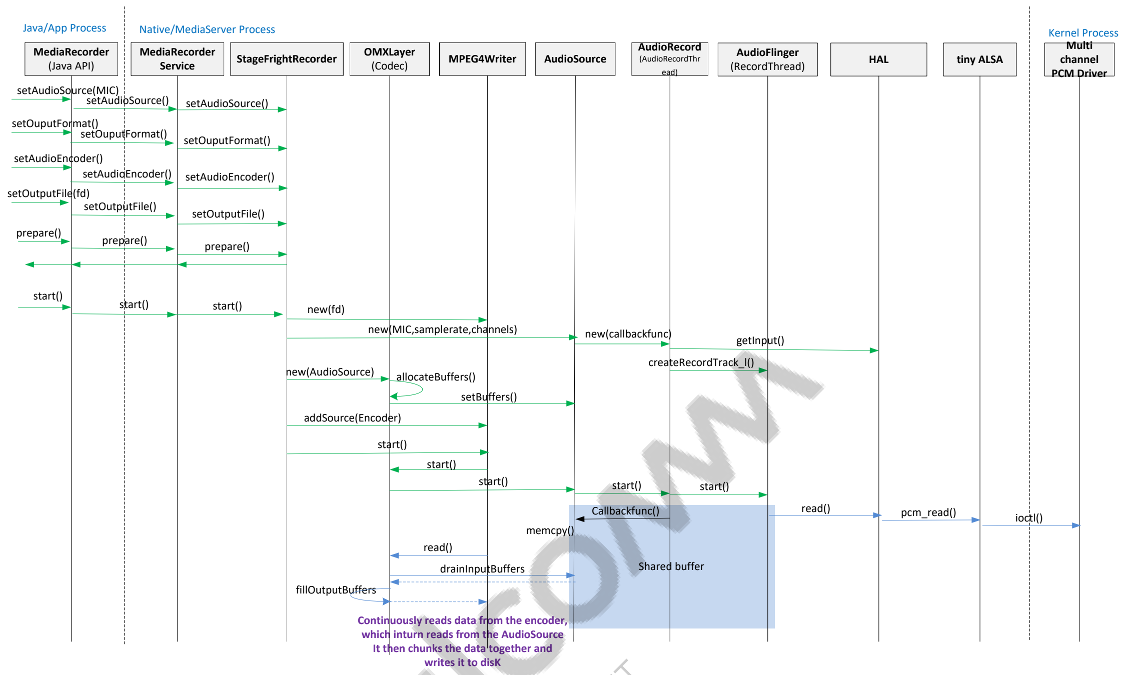


Figure 7-4 Normal recording call flow

7.2.2 Log collection

This section highlights the various files and/or components in which logging is enabled to troubleshoot issues related to audio recording on Android.

7.2.2.1 User space logs

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

7.2.2.2 Kernel logs

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file msm-multi-ch-pcm-q6-v2.c +p >
/sys/kernel/debug/dynamic_debug/control
echo file msm-pcm-routing-v2.c +p > /sys/kernel/debug/dynamic_debug/control
```

The encoder file where logging is enabled changes with the type of encoder.

```
echo -n "file aac_in.c  +p" > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging on msm-multi-ch-pcm-q6-v2.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

7.2.2.3 OMX layer logs

Depending on the OMX component used, logging in the respective OMX component is enabled.

To analyze the issue in the QTI AAC encoder OMX component, enable the following logs by changing DEBUG_PRINT to LOGE and DEBUG_DETAIL to LOGE in the hardware/qcom/audio/mm-audio/aenc-aac/qdsp6/inc/aenc_svr.h file.

```
#define DEBUG_PRINT_ERROR LOGE
#define DEBUG_PRINT       LOGE /*LOGI*/
#define DEBUG_DETAIL      LOGE /* LOGV */
```

7.2.2.4 In the user space, enable logs on files listed at the following locations.

In the user space, enable logs in files listed at the following locations, if the facility already exists:

```
\frameworks\av\media\libmediaplayerservice
MediaPlayerService.cpp
StagefrightRecorder.cpp

\frameworks\av\media\libmedia
AudioSystem.cpp
AudioRecord.cpp
mediarecorder.cpp
IOMX.cpp

\frameworks\av\media\libstagefright
AudioSource.cpp
MPEG4Writer.cpp
OMXClient.cpp
OMXCodec.cpp

\frameworks\av\services\audioflinger
AudioFlinger.cpp
AudioMixer.cpp
AudioPolicyService.cpp

\harwdware\qcom\audio\hal
audio_hw.c
\harwdware\qcom\audio\hal\msm8974
platform.c
hw_info.c

\harwdware\qcom\audio\hal\policy_hal
AudioPolicyManager.cpp

\harwdware\libhardware_legacy\audio
AudioPolicyManagerBase.cpp
```

On the kernel side, logs on the following files are enabled to debug issues related to APSS and DSP communication.

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6afe.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6asm.c  +p > /sys/kernel/debug/dynamic_debug/control
```

For issues related to calibration, enable log messages in audio_acdb.c.

```
echo file audio_acdb.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging in q6asm.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

7.2.2.5 QXDM Professional logs

This section is not applicable to this release.

7.2.3 Log analysis

7.2.3.1 User space logs

In the Android Native layer, the recording flow is initiated via MediaPlayerService. The MediaRecorder component creates a StagefrightRecorder instance.

```
01-02 01:21:41.349  2441  2441 V MediaRecorder: constructor
01-02 01:21:41.349   218  1111 V StagefrightRecorder: Constructor
```

The following MediaRecorder APIs are called in sequence to initialize the recording session. Audio source 1 is the mic, defined in AudioSystemLegacy.h.

```
01-02 01:21:41.349  2441  2441 V MediaRecorder: setAudioSource(1) 01-02
01:21:41.349   218  1111 V StagefrightRecorder: setAudioSource: 1
```

Next, the number of channels and sampling rate are set.

```
01-02 01:21:41.349  2441  2441 V MediaRecorder: setParameters(audio-param-
number-of-channels=6)
01-02 01:21:41.349   218   218 V StagefrightRecorder:
setParamAudioNumberOfChannels: 6
01-02 01:21:41.349  2441  2441 V MediaRecorder: setParameters(audio-param-
sampling-rate=48000)
01-02 01:21:41.349   218  1145 V StagefrightRecorder:
```

```
setParamAudioSamplingRate: 48000
```

Output format 1 is 3GPP, defined in mediarecorder.h.

```
01-02 01:21:41.349  2441  2441 V MediaRecorder: setOutputFormat(1)
01-02 01:21:41.349   218  1757 V StagefrightRecorder: setOutputFormat: 1
```

Encoder 3 is AAC, defined in mediarecorder.h.

```
01-02 01:21:41.349  2441  2441 V MediaRecorder: setAudioEncoder(3)
01-02 01:21:41.349   218  1111 V StagefrightRecorder: setAudioEncoder: 3

01-02 01:21:41.349  2441  2441 V MediaRecorder: setOutputFile(52, 0, 0)
01-02 01:21:41.349   218   218 V StagefrightRecorder: setOutputFile: 36, 0,
0
```

Recording is started after a prepare() command is issued from the MediaRecorder.

```
01-02 01:21:41.359  2441  2441 V MediaRecorder: prepare
01-02 01:21:41.359  2441  2441 V MediaRecorder: start
```

StagefrightRecorder now creates AudioSource, AudioRecord (which internally creates AudioTrack), and MPEG4Writer (does not print log markers) instances.

```
01-02 01:21:41.359   218  1757 V AudioSource: sampleRate: 48000,
channelCount: 6
01-02 01:21:41.359   218  1757 V AudioRecord: sampleRate 48000, channelMask
0x3f0000, format 1
01-02 01:21:41.359   218  1757 V AudioRecord: set(): mSessionId 21
```

AudioRecord then opens an input corresponding to the input device.

```
01-02 01:21:41.359   218  1757 V AudioPolicyManager:
getDeviceForInputSource()input source 1, device 80000004
01-02 01:21:41.359   218  1757 V AudioPolicyManager: getInput() inputSource
1, samplingRate 48000, format 1, channelMask 3f0000, acoustics 0
```

This action causes the HAL to load the Surround Sound library and allocate buffers.

```
01-02 01:21:41.359   218  1757 V audio_hw_primary: adev_open_input_stream:
enter
01-02 01:21:41.359   218  1757 D audio_hw_ssr: audio_extn_ssr_init: ssr
case
01-02 01:21:41.359   218  1757 D audio_hw_ssr: audio_extn_ssr_init:
buffer_size: 2048
```

```
01-02 01:21:41.359   218  1757 V AudioFlinger: openInput() openInputStream
returned input 0xb79607c8, SamplingRate 48000, Format 1, Channels 3f0000,
status 0
01-02 01:21:41.359   218  1757 V AudioFlinger: openInput() created record
thread: ID 22 thread 0xb797f578
```

The AAC encoder that is used to encode the input PCM stream is instantiated next; in this case, it is a software encoder.

```
01-02 01:21:41.369   218  1757 V OMXCodec: matching
'OMX.google.aac.encoder' quirks 0x00000000
01-02 01:21:41.369   218  1757 V OMXCodec: Attempting to allocate OMX node
'OMX.google.aac.encoder'
01-02 01:21:41.369   218  1757 V SoftOMXPlugin: makeComponentInstance
'OMX.google.aac.encoder'
12-13 15:48:55.672   215  1758 V SoftAACEncoder2: setAudioParams: 44100 Hz,
1 channels, 0 bps
01-02 01:21:41.369   218  1757 V OMXCodec: Successfully allocated OMX node
'OMX.google.aac.encoder'
01-02 01:21:41.369   218  1757 V OMXCodec: configureCodec protected=0
01-02 01:21:41.369   218  1757 V SoftAACEncoder2: setAudioParams: 48000 Hz,
6 channels, 96000 bps
```

Input and output buffers are allocated for the encoder to use, after which it moves to an idle state.

```
01-02 01:21:41.369   218  1757 I MPEG4Writer: limits: 2147483647/0
bytes/us, bit rate: 96000 bps and the estimated moov size 3072 bytes
01-02 01:21:41.369   218  1757 V MPEG4Writer: startWriterThread
01-02 01:21:41.369   218  1757 V MPEG4Writer: initTrackingProgressStatus
01-02 01:21:41.369   218  1757 V OMXCodec: [OMX.google.aac.encoder]
allocating 4 buffers of size 4096 on input port
01-02 01:21:41.369   218  1757 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb7a145e8 on input port
01-02 01:21:41.369   218  1757 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb7a147a0 on input port
01-02 01:21:41.369   218  1757 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb7a14870 on input port
01-02 01:21:41.369   218  1757 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb7a14928 on input port
01-02 01:21:41.369   218  1757 V OMXCodec: [OMX.google.aac.encoder]
allocating 4 buffers of size 8192 on output port
01-02 01:21:41.369   218  1757 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb7a14ab0 on output port
01-02 01:21:41.369   218  1757 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb7a14cc0 on output port
01-02 01:21:41.369   218  1757 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb7a14e00 on output port
01-02 01:21:41.369   218  1757 V OMXCodec: [OMX.google.aac.encoder]
allocated buffer 0xb7a14f28 on output port
```

```
01-02 01:21:41.369   218  3444 V SimpleSoftOMXComponent: msgType = 0
01-02 01:21:41.369   218  3444 V OMX     : OnEvent(0, 0, 2)
01-02 01:21:41.369   218  3445 V OMXCodec: [OMX.google.aac.encoder]
onStateChange 2
01-02 01:21:41.369   218  3445 V OMXCodec: [OMX.google.aac.encoder] Now
Idle.
```

The encoder moves into the running state after some media configuration is performed.

```
01-02 01:21:41.369   218  3444 V OMX     : OnEvent(0, 0, 3)
01-02 01:21:41.369   218  3445 V OMXCodec: [OMX.google.aac.encoder]
onStateChange 3
01-02 01:21:41.369   218  3445 V OMXCodec: [OMX.google.aac.encoder] Now
Executing.
```

The encoder informs the framework afer initializion and the recording process is kicked off. This action involves starting AudioRecord and RecordTrack threads.

```
01-02 01:21:41.369   218  1757 V AudioRecord: start, sync event 0 trigger
session 0
01-02 01:21:41.369   218  1757 V AudioRecord: mAudioRecord->start()

01-02 01:21:41.369   218  1757 V AudioFlinger: RecordHandle::start()
01-02 01:21:41.369   218  1757 V AudioFlinger: RecordThread::start event 0,
triggerSession 0
```

The input opened previously is started, which causes the audio path to be set up and calibration for the input device to be pushed.

```
01-02 01:21:41.369   218  1757 V AudioPolicyManagerBase: startInput() input
22
01-02 01:21:41.369   218  1757 V AudioPolicyManager:
getDeviceForInputSource()input source 1, device 80000004
01-02 01:21:41.369   218  1757 V AudioPolicyManagerBase:
AudioPolicyManager::startInput() input source = 1
01-02 01:21:41.369   218  3442 V AudioFlinger: RecordThread: loop starting
01-02 01:21:41.369   218  1757 V AudioFlinger: Signal record thread
01-02 01:21:41.379   218  3442 V msm8974_platform:
platform_update_usecase_from_source: input source :1
```

Input device is quad-mic defined in /hardware/qcom/audio/hal/msm8974/platform.c. Use case (6) is USECASE_AUDIO_RECORD, defined in hardware/qcom/audio/hal/audio_hw.h.

```
01-02 01:21:41.379   218  3442 V audio_hw_primary: start_input_stream:
enter: usecase(6)
01-02 01:21:41.379   218  3442 V audio_hw_primary: start_input_stream:
usecase(6)
01-02 01:21:41.379   218  3442 V msm8974_platform:
```

```
platform_get_input_snd_device: enter: out_device(0) in_device(0x4)
01-02 01:21:41.379   218  3442 V msm8974_platform:
platform_get_input_snd_device: exit: in_snd_device(quad-mic)
01-02 01:21:41.379   218  3442 D hardware_info: hw_info_append_hw_type :
device_name = quad-mic
01-02 01:21:41.379   218  3442 V audio_hw_primary: enable_snd_device:
snd_device(64: quad-mic)
```

The mic device has the ACDB ID 46 associated with it. Calibration data for the mic is sent to the DSP, as shown in the following log comment.

```
01-02 01:21:41.379   218  3442 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 46, path =  1
01-02 01:21:41.379   218  3442 D ACDB-LOADER: ACDB -> send_adm_topology
01-02 01:21:41.379   218  3442 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
01-02 01:21:41.379   218  3442 D ACDB-LOADER: ACDB -> send_audtable
01-02 01:21:41.379   218  3442 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
01-02 01:21:41.379   218  3442 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
01-02 01:21:41.379   218  3442 D ACDB-LOADER: ACDB -> send_audvoltable
01-02 01:21:41.379   218  3442 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
01-02 01:21:41.379   218  3442 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
01-02 01:21:41.379   218  3442 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
```

card_id (0) SOUND_CARD and device_id (0) is AUDIO_RECORD_PCM_DEVICE, defined in hardware/qcom/audio/hal/msm8974/platform.h.

```
01-02 01:21:41.389   218  3442 V audio_hw_primary: start_input_stream:
Opening PCM device card_id(0) device_id(0), channels 4
01-02 01:21:41.399   218  3442 D audio_hw_ssr: audio_extn_ssr_read:
period_size: 256
```

Input PCM data is read from AudioStreamInALSA and is handed to AudioSource via AudioTrack.

```
01-02 01:21:41.599   218  1757 V AudioFlinger: Record started OK
01-02 01:21:41.599   218  3442 D audio_hw_ssr: audio_extn_ssr_read:
period_size: 256
01-02 01:21:41.599   218  3443 V AudioRecord: obtainBuffer(320) returned
512 = 320 + 192
01-02 01:21:41.599   218  3443 V AudioRecord: obtainBuffer(320) returned
192 = 192 + 0
01-02 01:21:41.609   218  3443 V AudioRecord: obtainBuffer(320) returned
192 = 192 + 0
01-02 01:21:41.609   218  3443 V AudioRecord: obtainBuffer(128) returned 0
= 0 + 0
```

The OMX layer reads this data and sends it to the encoder instance by calling emptyBuffer(). The encoder uses EmptyBufferDone() CB to inform the OMX core that the buffer is used. This action is performed in a loop for all input buffers.

```
01-02 01:21:41.609   218  3447 V AudioSource: signalBufferReturned: 
0xb7a14ff0
01-02 01:21:41.609   218  3447 V OMXCodec: [OMX.google.aac.encoder] Calling 
emptyBuffer on buffer 0xb7a145e8 (length 3840), timestamp 232990 us (0.23 
secs)

01-02 01:21:41.609   218  3447 V AudioSource: signalBufferReturned: 
0xb7a15ef8
01-02 01:21:41.609   218  3447 V OMXCodec: [OMX.google.aac.encoder] Calling 
emptyBuffer on buffer 0xb7a147a0 (length 2304), timestamp 239657 us (0.24 
secs)

01-02 01:21:41.689   218  3447 V AudioSource: signalBufferReturned: 
0xb7a14ff0
01-02 01:21:41.689   218  3447 V OMXCodec: [OMX.google.aac.encoder] Calling 
emptyBuffer on buffer 0xb7a14870 (length 3840), timestamp 243657 us (0.24 
secs)

01-02 01:21:41.699   218  3447 V AudioSource: signalBufferReturned: 
0xb7a14ff0
01-02 01:21:41.699   218  3447 V OMXCodec: [OMX.google.aac.encoder] Calling 
emptyBuffer on buffer 0xb7a14928 (length 2304), timestamp 250324 us (0.25 
secs)
```

The OMX layer then reads encoded data from the encoder instance by calling fillBuffer(). The encoder uses FillBufferDone() CB to inform the OMX core that the buffer is filled. This action is performed in a loop for all input buffers.

```
01-02 01:21:41.699   218  3447 V OMXCodec: [OMX.google.aac.encoder] Calling 
fillBuffer on buffer 0xb7a14ab0
01-02 01:21:41.699   218  3444 V OMX     : OnFillBufferDone 
buffer=0xb7a14ab0

01-02 01:21:41.699   218  3447 V OMXCodec: [OMX.google.aac.encoder] Calling 
fillBuffer on buffer 0xb7a14cc0
01-02 01:21:41.699   218  3444 V OMX     : OnFillBufferDone 
buffer=0xb7a14cc0

01-02 01:21:41.699   218  3447 V OMXCodec: [OMX.google.aac.encoder] Calling 
fillBuffer on buffer 0xb7a14e00

01-02 01:21:41.699   218  3447 V OMXCodec: [OMX.google.aac.encoder] Calling 
fillBuffer on buffer 0xb7a14f28
```

The read() call from the encoder returns and MPEG4Writer pulls the encoded data. It packages the data into chunks that are written to file at the end. The read() call is done in a loop and MPEG4Writer is blocked until more data is made available by the encoder.

```
12-13 15:48:55.852   215 14938 V AudioSource: dataCallbackTimestamp:
5734988580 us
```

This action continues until the recording is stopped, which causes StagefrightRecorder, MPEG4Writer, and the encoder to stop followed by other components.

```
01-02 01:21:46.639  2441  2441 V MediaRecorder: stop
01-02 01:21:46.639   218   218 V StagefrightRecorder: stop

01-02 01:21:46.699   218  3447 V MPEG4Writer: Audio media time stamp:
597352 and previous paused duration 232990
01-02 01:21:46.699   218  3447 V MPEG4Writer: Audio
timestampUs/lastTimestampUs: 597352/576018
01-02 01:21:46.699   218  3447 I MPEG4Writer: Received total/0-length
(30/0) buffers and encoded 29 frames. - audio
01-02 01:21:46.699  2441  2452 V MediaRecorder: message received msg=101,
ext1=268436456, ext2=0
01-02 01:21:46.699   218  3447 V MPEG4Writer: getDriftTimeUs: 0 us
01-02 01:21:46.699   218  3447 I MPEG4Writer: Audio track drift time: 0 us
01-02 01:21:46.699   218   218 D MPEG4Writer: Stopping Audio track source
```

The encoder power-down is initiated, and it flushes all the data in its buffers as part of the process.

```
01-02 01:21:46.699   218   218 V OMXCodec: [OMX.google.aac.encoder] stop
mState=4

01-02 01:21:47.219   218  3445 V OMXCodec: [OMX.google.aac.encoder]
FILL_BUFFER_DONE(buffer: 0xb7a14ab0, size: 249, flags: 0x00000010,
timestamp: 894344 us (0.89 secs))

01-02 01:21:47.219   218  3445 V OMXCodec: [OMX.google.aac.encoder]
FILL_BUFFER_DONE(buffer: 0xb7a14cc0, size: 0, flags: 0x00000000, timestamp:
830342 us (0.83 secs))

01-02 01:21:47.219   218  3444 V OMX     : OnEvent(0, 0, 2)
01-02 01:21:47.219   218  3445 V OMXCodec: [OMX.google.aac.encoder]
onStateChange 2
01-02 01:21:47.219   218  3445 V OMXCodec: [OMX.google.aac.encoder] Now
Idle.
```

The encoder then releases all its input and output buffers and moves to a loaded state.

```
01-02 01:21:47.219   218  3445 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb7a14928 on port 0
```

```
01-02 01:21:47.219   218  3445 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb7a14870 on port 0
01-02 01:21:47.219   218  3445 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb7a147a0 on port 0
01-02 01:21:47.219   218  3445 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb7a145e8 on port 0
01-02 01:21:47.219   218  3445 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb7a14f28 on port 1
01-02 01:21:47.219   218  3445 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb7a14e00 on port 1
01-02 01:21:47.229   218  3445 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb7a14cc0 on port 1
01-02 01:21:47.229   218  3445 V OMXCodec: [OMX.google.aac.encoder] freeing
buffer 0xb7a14ab0 on port 1
01-02 01:21:47.229   218  3444 V SimpleSoftOMXComponent: msgType = 0

01-02 01:21:47.229   218  3444 V OMX     : OnEvent(0, 0, 1)
01-02 01:21:47.229   218  3445 V OMXCodec: [OMX.google.aac.encoder]
onStateChange 1
01-02 01:21:47.229   218  3445 V OMXCodec: [OMX.google.aac.encoder] Now
Loaded.
```

AudioRecord and RecordTrack threads are stopped next.

```
01-02 01:21:47.229   218   218 V AudioFlinger: RecordHandle::stop()
01-02 01:21:47.229   218   218 V AudioFlinger: RecordThread::stop
```

The device and modifier are disabled.

```
01-02 01:21:47.329   218  3442 V audio_hw_primary: stop_input_stream:
enter: usecase(6: audio-record)
01-02 01:21:47.329   218  3442 V audio_hw_primary: disable_audio_route:
enter: usecase(6)
01-02 01:21:47.329   218  3442 V audio_hw_primary: disable_audio_route:
reset mixer path: audio-record
01-02 01:21:47.329   218  3442 D hardware_info: hw_info_append_hw_type :
device_name = quad-mic
01-02 01:21:47.329   218  3442 V audio_hw_primary: disable_snd_device:
snd_device(64: quad-mic)
01-02 01:21:47.359   218  3442 V audio_hw_primary: stop_input_stream: exit:
status(0)
```

Device routing that was set up previously is reset.

```
01-02 01:21:47.359   218   218 V AudioFlinger: Record stopped OK
01-02 01:21:47.359   218   218 V AudioPolicyManagerBase: stopInput() input
22
```

MPEG4Writer is stopped, and it writes all of the media chunks it received previously.

```
01-02 01:21:47.359   218   218 V OMXCodec: [OMX.google.aac.encoder] stopped
in state 1
01-02 01:21:47.359   218   218 D MPEG4Writer: Audio track stopped
01-02 01:21:47.359   218   218 D MPEG4Writer: Stopping writer thread
01-02 01:21:47.359   218  3446 V MPEG4Writer: writeAllChunks
01-02 01:21:47.359   218  3446 V MPEG4Writer: findChunkToWrite
01-02 01:21:47.359   218  3446 V MPEG4Writer: Nothing to be written after
all
01-02 01:21:47.359   218  3446 D MPEG4Writer: 0 chunks are written in the
last batch
01-02 01:21:47.359   218   218 D MPEG4Writer: Writer thread stopped
01-02 01:21:47.359   218   218 V MPEG4Writer: writeCompositionMatrix
01-02 01:21:47.359   218   218 V MPEG4Writer: Audio track time scale: 48000
01-02 01:21:47.359   218   218 V MPEG4Writer: writeCompositionMatrix
01-02 01:21:47.359   218   218 D MPEG4Writer: Stopping Audio track
```

The encoder threads are stopped and its instance released.

```
01-02 01:21:47.359   218   218 V OMXNodeInstance: calling
destroyComponentInstance
01-02 01:21:47.359   218   218 V OMXNodeInstance: destroyComponentInstance
returned err 0
01-02 01:21:47.359   218   218 V OMXNodeInstance: OMXNodeInstance going
away.
```

The input stream and mic device are closed.

```
01-02 01:21:47.359   218   218 V AudioFlinger: RecordThread::stop
01-02 01:21:47.359   218   218 V AudioFlinger: closeInput() 22
01-02 01:21:47.359   218  3442 V AudioFlinger: RecordThread 0xb797f578
exiting
01-02 01:21:47.359   218   218 V audio_hw_primary: adev_close_input_stream
01-02 01:21:47.359   218   218 V AudioFlinger: virtual
android::AudioFlinger::RecordThread::RecordTrack::~RecordTrack()
01-02 01:21:47.359   218   218 V AudioFlinger: removeClient_l() pid 2441,
calling pid 01-02 01:21:47.359   218   218 V AudioFlinger: purging stale
effects
```

Framework recorder instances are released.

```
01-02 01:21:47.359   218  1145 V StagefrightRecorder: Destructor
01-02 01:21:47.359   218  1145 V StagefrightRecorder: stop
01-02 01:21:47.359  2441  2441 V MediaRecorder: destructor
```

7.2.3.2 Kernel logs

The PCM path from BACKEND DAI ID 3 (MSM_BACKEND_DAI_SLIMBUS_0_TX) to FRONTEND DAI ID 0 (MSM_FRONTEND_DAI_MULTIMEDIA1) is set up first. The BACKEND DAI IDs and FRONT END DAI IDs are defined in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[11070.358725] msm_pcm_routing_process_audio: reg 3 val 0 set 1
<7>[11070.359309] msm_pcm_hw_params: perf: 0
<7>[11070.359318] msm_pcm_hw_params Opening 4-ch PCM read stream
<7>[11070.359361] __q6asm_open_read:session[2]
```

Recording is started by setting the hardware parameters and calling msm_pcm_capture_prepare followed by the msm_pcm_trigger() function. PCM data is copied from DSP buffers. There are 16 buffers each of size 4096 bytes.

```
<7>[11070.366269] msm_pcm_hw_params: session ID 2
<7>[11070.366286] q6asm_audio_client_buf_alloc_contiguous:
session[2]bufsz[2048]bufcnt[8]
<7>[11070.366560] q6asm_audio_client_buf_alloc_contiguous
data[c7838800]phys[7ee71800][f603941c]
<7>[11070.366569] q6asm_audio_client_buf_alloc_contiguous
data[c7839000]phys[7ee72000][f6039438]
<7>[11070.366577] q6asm_audio_client_buf_alloc_contiguous
data[c7839800]phys[7ee72800][f6039454]
<7>[11070.366586] q6asm_audio_client_buf_alloc_contiguous
data[c783a000]phys[7ee73000][f6039470]
<7>[11070.366594] q6asm_audio_client_buf_alloc_contiguous
data[c783a800]phys[7ee73800][f603948c]
<7>[11070.366602] q6asm_audio_client_buf_alloc_contiguous
data[c783b000]phys[7ee74000][f60394a8]
<7>[11070.366610] q6asm_audio_client_buf_alloc_contiguous
data[c783b800]phys[7ee74800][f60394c4]

<7>[11070.366620] q6asm_memory_map_regions: Session[2]
<7>[11070.367323] q6asm_memory_map_regions: i=0, bufadd[i] = 0x7ee71000,
maphdl[i] = 0xf09517d8
<7>[11070.367332] q6asm_memory_map_regions: i=1, bufadd[i] = 0x7ee71800,
maphdl[i] = 0xf09517d8
<7>[11070.367340] q6asm_memory_map_regions: i=2, bufadd[i] = 0x7ee72000,
maphdl[i] = 0xf09517d8
<7>[11070.367349] q6asm_memory_map_regions: i=3, bufadd[i] = 0x7ee72800,
maphdl[i] = 0xf09517d8
<7>[11070.367357] q6asm_memory_map_regions: i=4, bufadd[i] = 0x7ee73000,
maphdl[i] = 0xf09517d8
<7>[11070.367365] q6asm_memory_map_regions: i=5, bufadd[i] = 0x7ee73800,
maphdl[i] = 0xf09517d8
<7>[11070.367373] q6asm_memory_map_regions: i=6, bufadd[i] = 0x7ee74000,
maphdl[i] = 0xf09517d8
<7>[11070.367381] q6asm_memory_map_regions: i=7, bufadd[i] = 0x7ee74800,
maphdl[i] = 0xf09517d8
```

```
<7>[11070.467756] msm_pcm_capture_prepare
<7>[11070.467774] Samp_rate = 48000
<7>[11070.467787] Channel = 4
<7>[11070.476293] msm_pcm_trigger: Trigger start
<7>[11070.476312] session[2]
```

The pcm_irq_pos value increments by the size of a buffer each time a READ returns.

```
<7>[11070.495467] ASM_DATA_EVENT_READ_DONE_V2
<7>[11070.495491] pcm_irq_pos=2048
<7>[11070.495582] msm_pcm_capture_copy
<7>[11070.495596] appl_ptr 0
<7>[11070.495608] hw_ptr 256
<7>[11070.495750] idx = 0
<7>[11070.495782] msm_pcm_capture_copy:fbytes = 0: size=1792: xfer=256
<7>[11070.495796]  Sending next buffer to dsp
<7>[11070.495835] q6asm_read:session[2]dsp-
buf[0][c7838000]cpu_buf[1][7ee71800]
<7>[11070.495917] Returning from capture_copy... 0

<7>[11070.505454] ASM_DATA_EVENT_READ_DONE_V2
<7>[11070.505479] pcm_irq_pos=4096
<7>[11070.505841] msm_pcm_capture_copy:fbytes = 0: size=1792: xfer=256
<7>[11070.506073] appl_ptr 64
<7>[11070.506085] hw_ptr 768
<7>[11070.506170] idx = 2
<7>[11070.506201] msm_pcm_capture_copy:fbytes = 0: size=1792: xfer=256
<7>[11070.506215]  Sending next buffer to dsp
<7>[11070.506254] q6asm_read:session[2]dsp-
buf[2][c7839000]cpu_buf[3][7ee72800] <7>[11070.506336] Returning from
capture_copy... 0
```

Capture gets stopped by sending SNDRV_PCM_TRIGGER_STOP.

```
<7>[11076.436425] SNDRV_PCM_TRIGGER_STOP
<7>[11076.436445] msm_pcm_capture_close
```

The DSP stops filling the buffers and returns them to the driver(pcm_ireq_pos stays the same). The driver frees all the buffers.

```
<7>[11076.436462] __q6asm_cmd:CMD_CLOSE
<7>[11076.437114] ASM_DATA_EVENT_READ_DONE_V2
<7>[11076.437125] pcm_irq_pos=11520
<7>[11076.437893] ASM_DATA_EVENT_READ_DONE_V2
<7>[11076.437905] pcm_irq_pos=11520
```

DSP buffers are released.

```
<7>[11076.440869] q6asm_memory_unmap: Session[2]
<7>[11076.440874] q6asm_add_mmaphdr:pkt size=24 cmd_flg=1
<7>[11076.440879] q6asm_memory_unmap: Found the element
<7>[11076.440885] q6asm_memory_unmap: mem_unmap-mem_map_handle: 0xf09517d8
<7>[11076.441201] q6asm_srvc_callback:ptr0[0x10d94]ptr1[0x0]opcode[0x110e8]
token[0x201]payload_s[8] src[0] dest[0]sid[2]dir[0]
<7>[11076.441208] q6asm_srvc_callback:Payload = [0x10d94] status[0x0]
<7>[11076.441219] q6asm_srvc_callback:Payload = [0x10d94] status[0x0]
<7>[11076.441241]
q6asm_audio_client_buf_free_contiguous:data[c7838000]phys[7ee71000][f603940
0] , client[c341ea00] handle[f48ae640]
<7>[11076.441307] q6asm_audio_client_free: Session id 2
<7>[11076.441316] q6asm_session_free: sessionid[2]
```

PCM routing from FRONTEND DAI to BACK END DAI is disabled.

```
<7>[11076.465918] msm-pcm-routing msm-pcm-routing: reg 0
<7>[11076.466838] msm_pcm_routing_process_audio: reg 3 val 0 set 0
```

7.3 Customization

This section is not applicable to this release.

7.4 Debugging

- Recording is not working
  - Check if there are any errors in the Kernel driver, user space.
  - Check the Sample rate and number of channels configured in the DSP.
  - Check the Sample rate and Number of channels configured for the codec.
- Recording volume is low
  - In such scenarios, check various gains in the Tx path.
  - Tune analog and digital gains in the WCD codec and the gains in DSP.
  - PCM logging in the DSP at various PCM logging points as mentioned in *Hexagon Access Audio/Voice PCM/Bit Stream Logging with QXDM Professional* (80-N3470-4) can help in finding the root cause for low volume.
- Recorded audio is silent
  - Check whether the mixer commands to configure the Tx path in the codec are correct.
  - Verify the recording path using ALSA area application to help in isolating the issue and finding whether it is in user space, DSP, or codec.
  - QXDM Professional logging as mentioned in *Hexagon Access Audio/Voice PCM/Bit Stream Logging with QXDM Professional* (80-N3470-4) helps if there is an issue in the DSP or the WCD codec configuration.

- Noise in the recorded audio
  - The issue is isolated by collecting PCM logs using QXDM Professional as mentioned in *Hexagon Access Audio/Voice PCM/Bit Stream Logging with QXDM Professional* (80-N3470-4). If the PCM captured at AFE is noisy, there could be an issue with the WCD codec configurations.
- Power consumption during recording is high
  - This issue is debugged by taking clock dumps, power top, and other power-related debugging techniques.
- More number of interrupts when the DSP-based encoder is used
  - If QTI OMX components using DSP-based encoder are enabled, the number of interrupts from DSP to APSS increases due to Nontunnel mode encoding. In Nontunnel mode encoding, PCM data is captured from the DSP and again sent back to the DSP for encoding. The encoded data is again read from the DSP. This action results in extra interrupts from the DSP to the APSS as compared to software-based encoding.

8 Compress offload playback

8.1 Compress offload playback

In Android 4.4 KitKat release, Google added the support for compress offload playback.

The main concept is offloading the CPU from doing audio decoding and instead decoding it in DSP. Following this design, the CPU sends large buffers (32 kB) of compressed data to DSP.

The supported formats are MP3 and AAC.

Figure 8-1 shows the main modules involved during audio playback through the Android audio framework, Audio HAL, kernel drivers, and aDSP firmware.

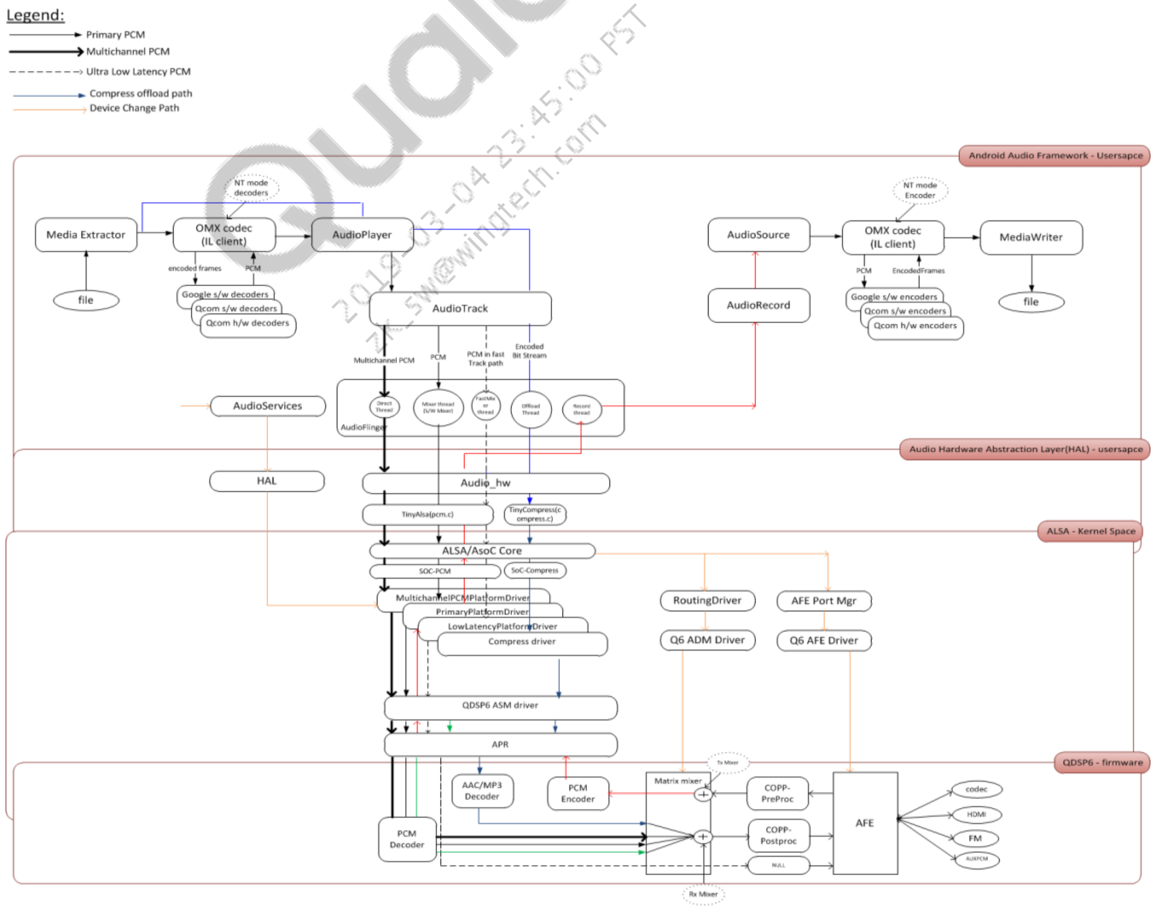


Figure 8-1 Compress offload architecture diagram

8.1.1 Android audio framework

This section contains the changes done to support compress offload.

8.1.1.1 AwesomePlayer

If a stream is offloaded, the AwesomePlayer queries the audio policy manager (APM) to decide.

The following conditions are met to offload a stream:

- audio.offload.disable property is not set
- Clip duration is more than 1 min
- Stream type is music
- Format is AAC, MP3, or AC3/EAC3 (Dolby)
- If it gets the profiles for direct output

The AwesomePlayer also creates an OMX decoder as a fallback and sets the AudioTrack as source. During audio tear-down events, the AwesomePlayer rechecks if compress offload is possible; it recreates the AudioPlayer, connects the OMX decoder, and seek to the last position.

8.1.1.2 AudioPlayer

AudioPlayer takes care of getting the current playback position.

In gapless playback, the AudioPlayer sets the metadata to the HAL via setParameters:

- Encoder delay: Initial silence
- Padding size: Trailing silence

These parameters are required for the decoder to compensate for the compression artifacts (encode delay and padding size) that insert silence at the beginning and end of a track. To have the gapless effect, the decoder takes care of dropping initial and trailing silence using the metadata parameters present in the file.

8.1.1.3 AudioFlinger

AudioFlinger is responsible for creating the offload thread derived from the direct output thread. The offload thread supports:

- Asynchronous writes and drains
- Pause, resume, and drain events

8.1.2 APM

A new function, *isOffloadSupported* is added to check if a stream is offloaded.

The new compress offload profile added in audio_policy.conf is:

```
compress_offload {
  sampling_rates 8000|11025|16000|22050|32000|44100|48000
  channel_masks
AUDIO_CHANNEL_OUT_MONO|AUDIO_CHANNEL_OUT_STEREO|AUDIO_CHANNEL_OUT_2POINT1|A
UDIO_CHANNEL_OUT_QUAD|AUDIO_CHANNEL_OUT_PENTA|AUDIO_CHANNEL_OUT_5POINT1|AUD
IO_CHANNEL_OUT_6POINT1|AUDIO_CHANNEL_OUT_7POINT1
  formats
AUDIO_FORMAT_MP3|AUDIO_FORMAT_AAC|AUDIO_FORMAT_AC3|AUDIO_FORMAT_EAC3
  devices
AUDIO_DEVICE_OUT_SPEAKER|AUDIO_DEVICE_OUT_EARPIECE|AUDIO_DEVICE_OUT_WIRED_H
EADSET|AUDIO_DEVICE_OUT_WIRED_HEADPHONE|AUDIO_DEVICE_OUT_ALL_SCO|AUDIO_DEVI
CE_OUT_AUX_DIGITAL|AUDIO_DEVICE_OUT_PROXY|AUDIO_DEVICE_OUT_ANLG_DOCK_HEADSE
T|AUDIO_DEVICE_OUT_FM_TX
  flags
AUDIO_OUTPUT_FLAG_DIRECT|AUDIO_OUTPUT_FLAG_COMPRESS_OFFLOAD|AUDIO_OUTPUT_FL
AG_NON_BLOCKING
}
```

This profile contains the supported parameters for a stream to be offloaded. Only those streams matching the sample rates, number of channels, and output devices are offloaded. In the case of flags, the following three flags are set so the stream is offloaded:

- AUDIO_OUTPUT_FLAG_DIRECT: Indicates cases bypassed to AudioFlinger
- AUDIO_OUTPUT_FLAG_COMPRESS_OFFLOAD: Indicates the stream is offloaded
- AUDIO_OUTPUT_FLAG_NON_BLOCKING: Indicates that the writes for this stream are nonblocking

8.1.3 HAL

A new C-based HAL was introduced in KitKat.

Similarly to the previous HAL in Jelly Bean, this new HAL opens and closes hardware drivers, talking to these drivers directly for audio rendering or capturing.

Figure 8-2 shows how the HAL interacts with higher and lower layers.

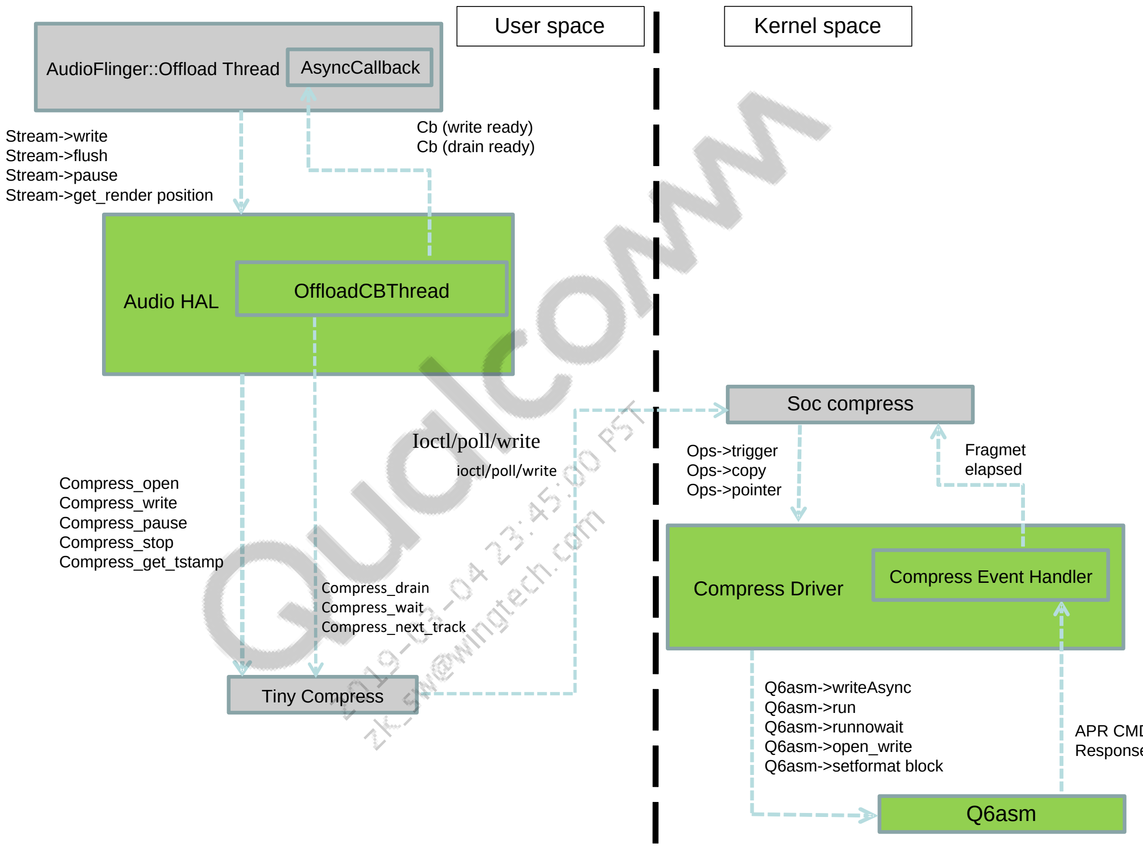


Figure 8-2 Frameworks, HAL, and kernel diagram

The main APIs are:

- get_render_position()
  - Gives the current playback position in frames; the timestamp is queried from aDSP through compress_get_tstamp()
- write()
  - Calls compress_write(), which is a nonblocking call; does not block if enough space is available in the driver
  - A new thread, OffloadCBThread, calls compress_wait() and notifies AudioFlinger on completion
  - Returns actual number of bytes written
  - OffloadCBThread notifies AudioFlinger with WRITE_READY events

- drain()
  - compress_drain() and compress_partial_drain() block calls that are called from OffloadCBThread
  - drain() returns when all data sent to the driver is played
  - compress_partial_drain() is used specifically for gapless use case
  - OffloadCBThread notifies AudioFlinger with DRAIN_READY events
- flush()
  - Calls compress_stop(), which stops a running stream; the main difference between flush and drain is that drain plays until end of buffers and then stops, while flush discards buffers
- pause()
  - Calls compress_pause(), which pauses the running stream
  - If there is no activity after 60 sec of playback, the tear-down function is called and any remaining data in the HAL is flushed

8.2 Call flow

8.2.1 Offload playback initialization

Figure 8-3 shows the initialization sequence for compress offload playback.

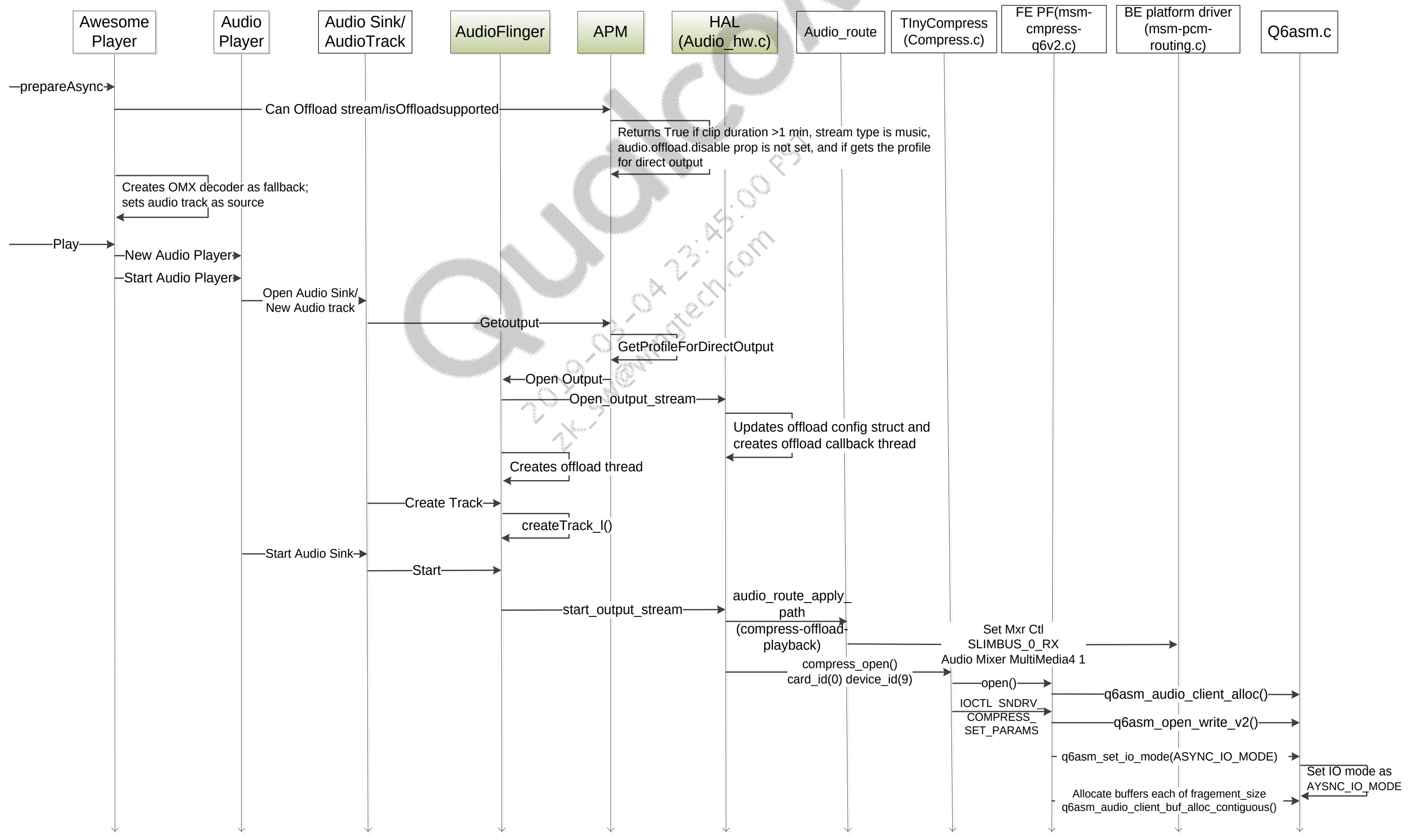

Awesome Player
Audio Player
Audio Sink/ AudioTrack
AudioFlinger
APM
HAL (Audio_hw.c)
Audio_route
TInyCompress (Compress.c)
FE PF(msm-cmpress-q6v2.c)
BE platform driver (msm-pcm-routing.c)
Q6asm.c
prepareAsync
Can Offload stream/isOffloadsupported
Returns True if clip duration >1 min, stream type is music, audio.offload.disable prop is not set, and if gets the profile for direct output
Creates OMX decoder as fallback; sets audio track as source
Play
New Audio Player
Start Audio Player
Open Audio Sink/ New Audio track
Getoutput
GetProfileForDirectOutput
Open Output
Open_output_stream
Updates offload config struct and creates offload callback thread
Creates offload thread
Create Track
createTrack_l()
Start Audio Sink
Start
start_output_stream
audio_route_apply_ path (compress-offload-playback)
Set Mxr Ctl SLIMBUS_0_RX Audio Mixer MultiMedia4 1
compress_open() card_id(0) device_id(9)
open()
IOCTL_SNDRV_ COMPRESS_ SET_PARAMS
q6asm_audio_client_alloc()
q6asm_open_write_v2()
q6asm_set_io_mode(ASYNC_IO_MODE)
Set IO mode as AYSNC_IO_MODE
Allocate buffers each of fragement_size q6asm_audio_client_buf_alloc_contiguous()

Figure 8-3 Offload playback initialization

The prepareAsync() call issued by the MediaPlayer, then AwesomePlayer queries the APM to decide if a stream is offloaded.

To offload a stream the following conditions are met:

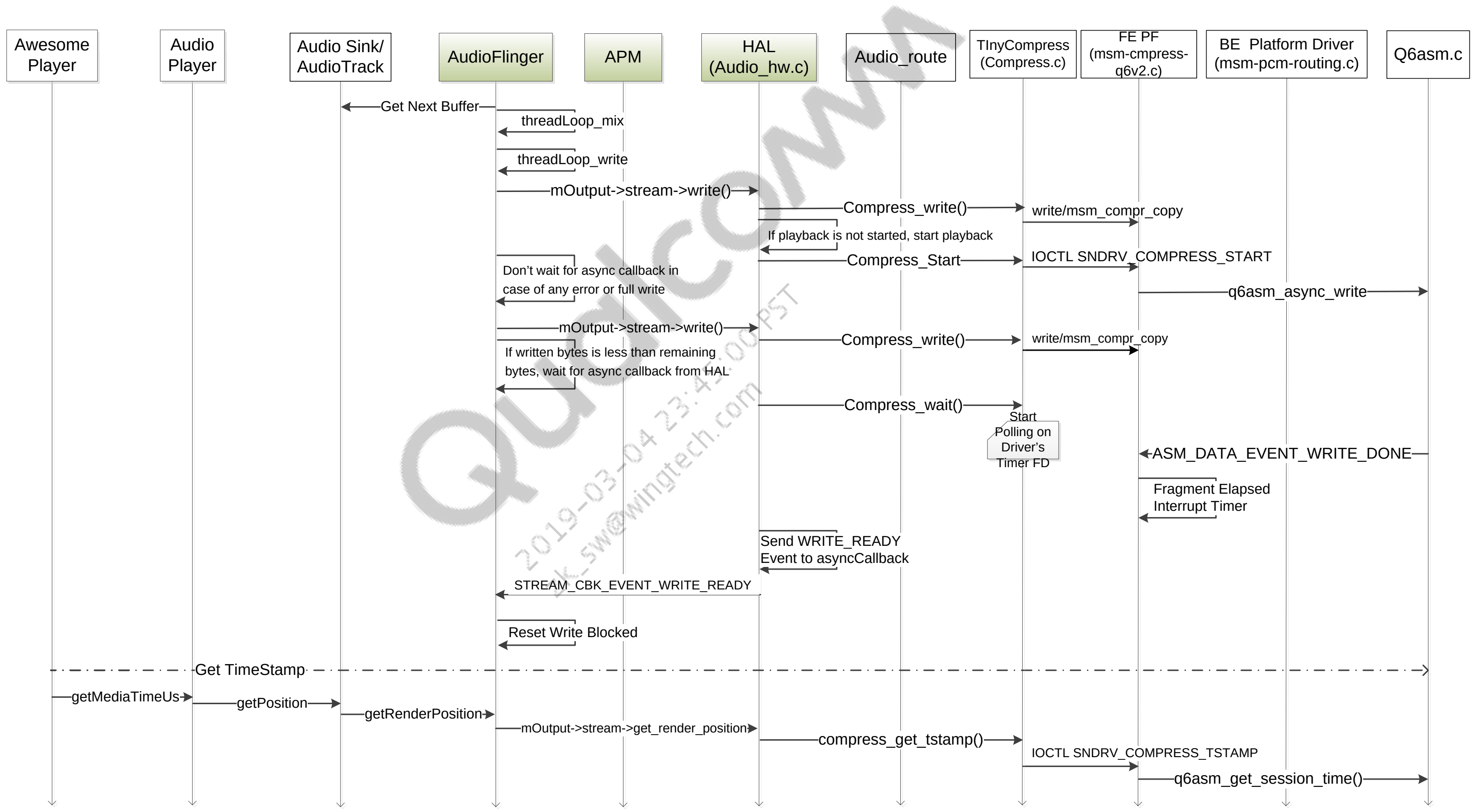
- audio.offload.disable prop is not set
- Clip duration is more than 1 min
- Stream type is music
- Format is AAC, MP3, or AC3/EAC3 (Dolby)
- If it gets the profiles for direct output

Once the play command is issued:

1. AwesomePlayer creates a AudioPlayer, which then creates a AudioTrack. Based on the information AwesomePlayer has received from the APM. It sets the flag USE_OFFLOAD, which passes to AudioPlayer.
2. AwesomePlayer also creates an OMX decoder as a fallback and sets the AudioTrack as its source.
3. AudioTrack then calls getOutput(), which gets the profile information from the APM. The APM then calls openOutput(), which goes to AudioFlinger. Based on the AUDIO_OUTPUT_FLAG_COMPRESS_OFFLOAD flag previously set by AudioPlayer, AudioFlinger creates the offload thread. AudioFlinger also calls open_output_stream() which goes to the HAL. The HAL then creates the offload callback thread used to notify the write and drain completion events.
4. AudioTrack then creates the track to be used for this stream.
5. AudioPlayer issues a start command to AudioSink, which then calls start_output_stream(). The HAL sends the appropriate mixer control to the BE platform driver to configure the output device.
6. The HAL opens the compress driver, and sends the codec and stream parameters to the compress driver though the IOCTL SNDRV_COMPRESS_SET_PARAMS. The compress driver opens an ASM session accordingly and requests the ASM driver to allocate the buffers to be used for this session.

8.2.2 Offload playback

Figure 8-4 shows the sequence during compress offload playback.



Awesome Player
Audio Player
Audio Sink/ AudioTrack
AudioFlinger
APM
HAL (Audio_hw.c)
Audio_route
TInyCompress (Compress.c)
FE PF (msm-cmpress-q6v2.c)
BE Platform Driver (msm-pcm-routing.c)
Q6asm.c
Get Next Buffer
threadLoop_mix
threadLoop_write
mOutput->stream->write()
Compress_write()
write/msm_compr_copy
If playback is not started, start playback
Compress_Start
IOCTL SNDRV_COMPRESS_START
Don't wait for async callback in case of any error or full write
q6asm_async_write
mOutput->stream->write()
Compress_write()
write/msm_compr_copy
If written bytes is less than remaining bytes, wait for async callback from HAL
Compress_wait()
Start Polling on Driver's Timer FD
ASM_DATA_EVENT_WRITE_DONE
Fragment Elapsed Interrupt Timer
Send WRITE_READY Event to asyncCallback
STREAM_CBK_EVENT_WRITE_READY
Reset Write Blocked
Get TimeStamp
getMediaTimeUs
getPosition
getRenderPosition
mOutput->stream->get_render_position
compress_get_tstamp()
IOCTL SNDRV_COMPRESS_TSTAMP
q6asm_get_session_time()

Figure 8-4 Offload playback

Compress offload playback is controlled by the offload thread created from AudioFlinger.

1. AudioFlinger within threadLoop_mix gets the next buffer available from AudioTrack.
2. AudioFlinger writes the buffer to the HAL.
3. The HAL sends the buffer to the compress driver through TinyAlsa calling compress_write().
4. The HAL checks if playback has already started and starts it calling compress_start() in case it was not started.
5. As mentioned in Section 8.1.3, this write is a nonblocking call. A callback mechanism is used to notify about error or write complete events; it returns the actual number of bytes written.
    - If written bytes are less than remaining bytes, it waits for async callback from HAL.
6. Figure 8-4 shows how MediaPlayer can query the timestamp of a stream. The call gets propagated through frameworks, HAL, and the compress driver to reach the ASM driver, which queries the timestamp from the ASM session in aDSP.

8.2.3 Compress offload pause and resume

Figure 8-5 illustrates compress offload pause and resume.

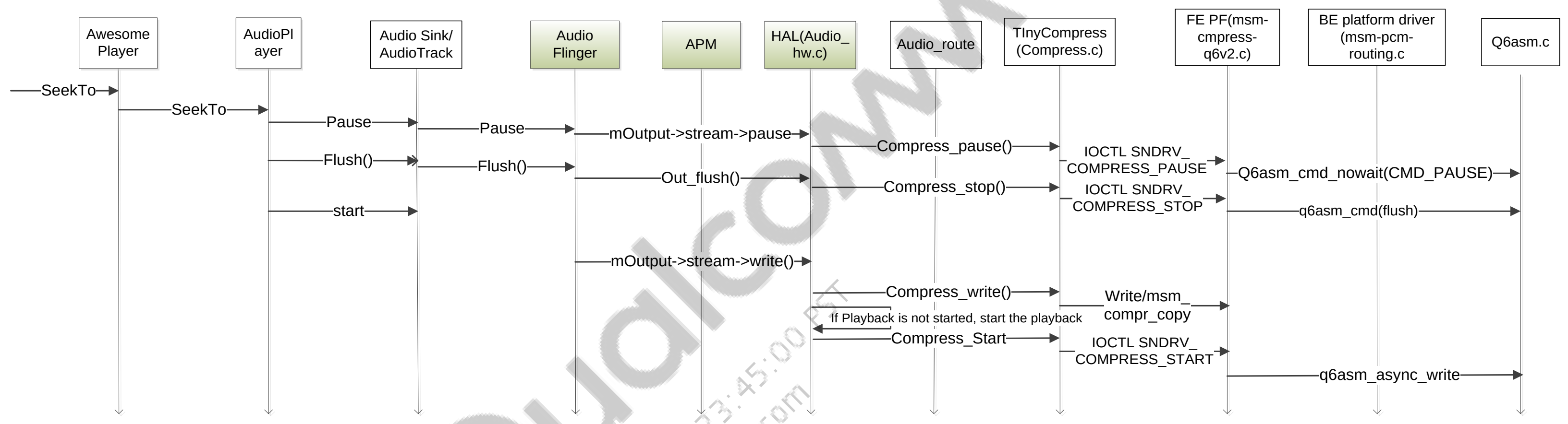


Figure 8-5 Compress offload seek call flow

1. In a pause event, AwesomePlayer calls seekTo in AudioPlayer. AudioPlayer then sends a Pause command to AudioTrack, which is propagated to AudioFlinger, HAL, compress, and the ASM driver.
2. AudioPlayer then starts a flush, because there could be a large amount of data queued in the hardware. If played, that unwanted buffered data could delay a track switch.
3. AudioPlayer continues issuing a start command as part of the seekTo() function. AudioPlayer writes the buffer to the HAL in the new position.
4. The HAL sends the buffer to the compress driver through TinyAlsa by calling compress_write().
5. The HAL checks if playback has already started and if not, starts it by calling compress_start().

8.2.4 Volume control

Figure 8-6 illustrates volume control in compress offload.

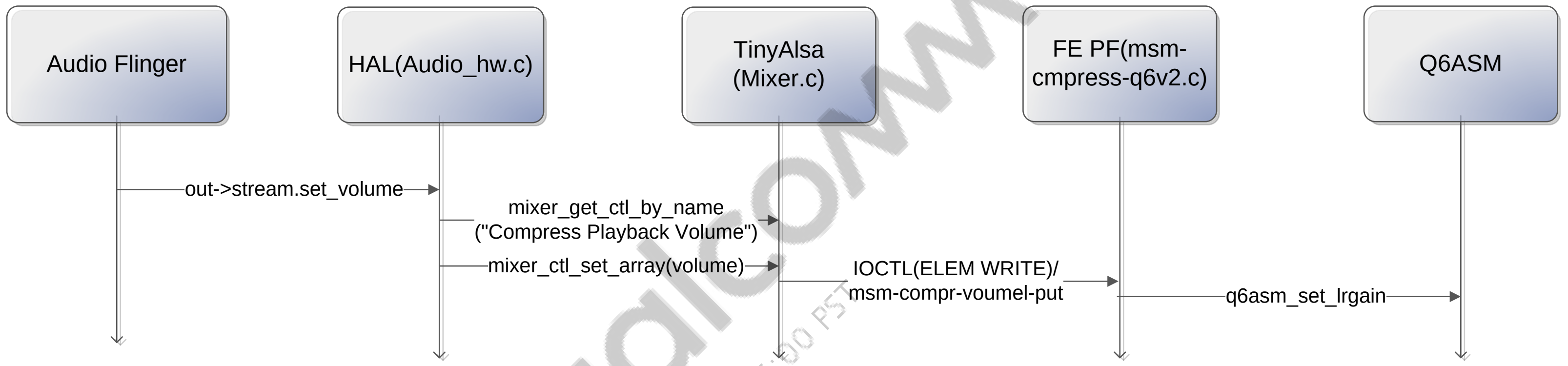

Audio Flinger
HAL(Audio_hw.c)
TinyAlsa (Mixer.c)
FE PF(msm-cmpress-q6v2.c)
Q6ASM
out->stream.set_volume
mixer_get_ctl_by_name ("Compress Playback Volume")
mixer_ctl_set_array(volume)
IOCTL(ELEM WRITE)/ msm-compr-voumel-put
q6asm_set_lrgain

Figure 8-6 Volume control in compress offload

The stream volume is controlled from AudioFlinger, which calls to out > stream.set_volume to set the volume in HAL. The HAL then gets the appropriate mixer controls and sets the volume control to the ASM session though the compress driver.

8.2.5 Device switch Bluetooth (offload to nonoffload)

Figure 8-7 shows the device switch Bluetooth (offload > nonoffload).

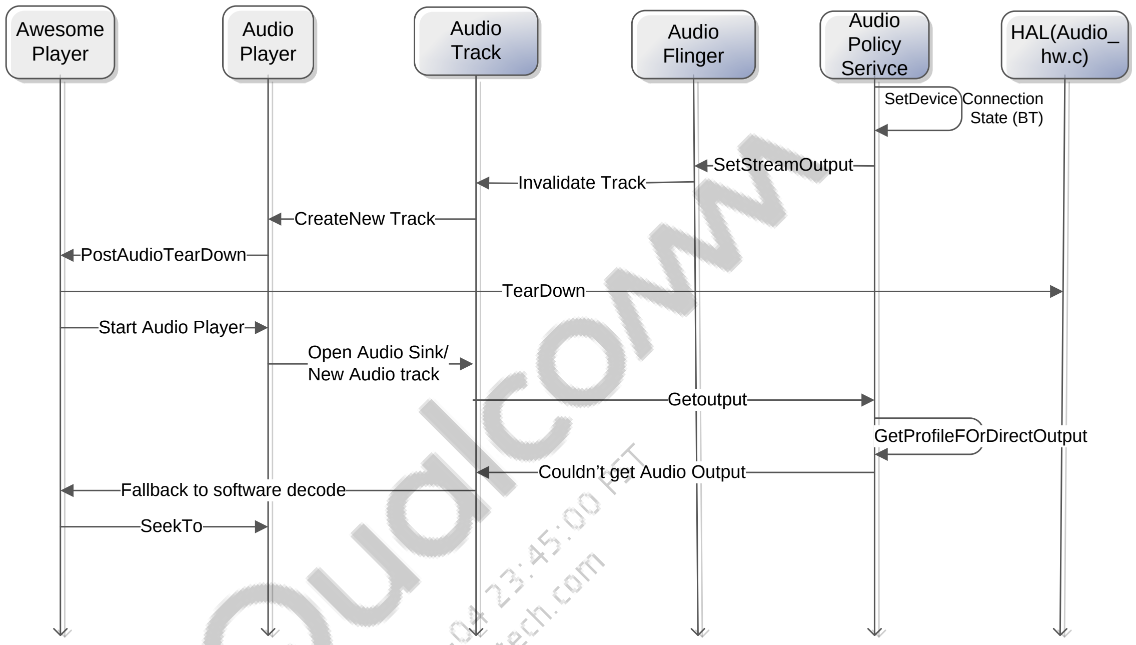


Figure 8-7 Device switch Bluetooth call flow

6. If a Bluetooth device is paired, the AudioPolicyService notifies AudioFlinger of this event by calling setStreamOutput(). AudioFlinger then invalidates the current track and a new track is created to resubmit the unwritten data.
7. AudioPlayer then requests AwesomePlayer to get into postAudioTearDownEvent. A pause and a flush are issued in the entire pipeline as shown in Figure 8-5. The flush is required to resend data already sent to DSP when changing audio session, as the source has changed.
8. AwesomePlayer then calls start() for AudioPlayer, which then creates a AudioTrack.
9. AudioTrack then calls getOutput(), which gets the profile information from the APM. The APM then calls openOutput(), which goes to AudioFlinger. However, since there is no valid profile for Bluetooth, it fails and propagates the error to AudioTrack. AudioTrack then returns the error to AwesomePlayer to indicate that starting the AudioPlayer was unsuccessful.
10. AwesomePlayer then decides to fallback to software decode and seeks to the last position before AudioTearDownEvent.

8.3 Log collection

This section contains the files and/or components in which logging is enabled to troubleshoot issues related to audio playback on Android.

8.3.1 User space logs

This command clears the current logcat logs and starts logging the user space logs to the file logcat.txt and the stdout simultaneously.

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

NOTE: For the tee command to work, Cygwin is installed or the command is executed in a Linux environment.

8.3.2 Kernel logs

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo -n "file msm-compress-q6-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file compress_offload.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file msm-pcm-routing-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file q6asm.c +p" > /sys/kernel/debug/dynamic_debug/control
```

8.3.3 Enable additional logging

Depending on the issue, additional logging is enabled in the user space and kernel space.

In the user space, enable logs on files listed at the following locations, if the facility exists:

```
\frameworks\av\media\libmediaplayerservice
MediaPlayerService.cpp
StagefrightPlayer.cpp

\frameworks\av\media\libmedia
AudioSystem.cpp
AudioTrack.cpp
mediaplayer.cpp
```

```
\frameworks\av\media\libstagefright
AwesomePlayer.cpp
AudioPlayer.cpp
MP3Extractor.cpp
OMXClient.cpp
OMXCodec.cpp

\frameworks\av\services\audioflinger
AudioFlinger.cpp
AudioMixer.cpp
AudioPolicyService.cpp

\harwdware\qcom\audio\hal
audio_hw.c

\harwdware\qcom\audio\hal\msm8974
platform.c
hw_info.c

\harwdware\qcom\audio\hal\policy_hal
AudioPolicyManager.cpp

\harwdware\libhardware_legacy\audio
AudioPolicyManagerBase.cpp
```

On the kernel side, logs on the following files are enabled to debug issues related to APSS and DSP communication.

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6afe.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6asm.c  +p > /sys/kernel/debug/dynamic_debug/control
```

For calibration issues, enable log messages in audio_acdb.c.

```
echo file audio_acdb.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging q6asm.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

8.4 Log analysis

8.4.1 User space logs

Audio playback is initiated from the application layer through the MediaPlayer service. MediaPlayer in turn uses the Stagefright framework to fulfill the actual playout. StagefrightPlayer instantiates AwesomePlayer in the playback scenario.

```
01-02 00:03:17.679  2602  3217 V MediaPlayer: constructor
01-02 00:03:17.679  2602  2617 V MediaPlayer:
MediaPlayer::setAudioSessionId(12)
01-02 00:03:17.679   215  1141 V AudioFlinger: acquiring 12 from 2602
01-02 00:03:17.699  2602  3217 V MediaPlayer: setDataSource(67, 0,
576460752303423487)
01-02 00:03:17.699   215  1141 V MediaPlayerService: Client(6) constructor
01-02 00:03:17.699   215  1141 V MediaPlayerService: Create new client(6)
from pid 2602, uid 10067,
01-02 00:03:17.709   215  1114 V MediaPlayerService: setDataSource fd=25,
offset=0, length=576460752303423487
01-02 00:03:17.709   215  1114 V MediaPlayerService: calculated length =
8158464
01-02 00:03:17.709   215  1114 V StagefrightPlayer: StagefrightPlayer
01-02 00:03:17.709   215  1114 V AwesomePlayer: AwesomePlayer running on
behalf of uid 10067
01-02 00:03:17.709   215  1114 V AudioSink: AudioOutput(12)
01-02 00:03:17.709   215  1114 V StagefrightPlayer: setDataSource(25, 0,
8158464)
```

Once MediaPlayer, StagefrightPlayer, and AwesomePlayer instances are created, the input file data is extracted and a MediaSource is created from it.

```
01-02 00:03:17.719   215  1114 V MP3Extractor: skipped ID3 tag, new
starting offset is 4096 (0x0000000000001000)
01-02 00:03:17.719   215  1114 V MP3Extractor: found possible 1st frame at
5056 (header = 0xfffbe400)
01-02 00:03:17.719   215  1114 V MP3Extractor: subsequent header is
fffbe400
01-02 00:03:17.719   215  1114 V MP3Extractor: found subsequent frame #2 at
6016
01-02 00:03:17.719   215  1114 V MP3Extractor: subsequent header is
fffbe400
01-02 00:03:17.719   215  1114 V MP3Extractor: found subsequent frame #3 at
6976
01-02 00:03:17.719   215  1114 V MP3Extractor: subsequent header is
fffbe400
01-02 00:03:17.719   215  1114 V MP3Extractor: found subsequent frame #4 at
7936
```

```
01-02 00:03:17.729   214  3185 D PRDrmPlugIn: PRDrmPlugin::onInitialize,
uniqueId = 5983
01-02 00:03:17.729   214  3185 D PRDrmPlugIn:
PRDrmPlugin::onSetOnInfoListener: uniqueId = 5983
01-02 00:03:17.729   214   214 D PRDrmPlugIn: PRDrmPlugin::onTerminate,
uniqueId = 5983
01-02 00:03:17.729   215  1114 D ExtendedUtils: extended extractor not
needed, return default
01-02 00:03:17.729   215  1114 V AwesomePlayer: mBitrate = 320000 bits/sec
01-02 00:03:17.729   215  1114 V MediaPlayerService:  setDataSource
01-02 00:03:17.729  2602  3217 V MediaPlayer:
MediaPlayer::setAudioStreamType
01-02 00:03:17.729  2602  3217 V MediaPlayer: setVideoSurfaceTexture
01-02 00:03:17.729   215   215 V MediaPlayerService: [6]
setVideoSurfaceTexture(0x0)
```

A prepare() API is then invoked, which instantiates the decoder that is used to decode the compressed media. As playback is in Compress Offload mode, data from the media file is directly piped to AwesomePlayer (internally AudioPlayer) in encoded form. The framework instantiates an instance of the hardware decoder to decode the encoded media input.

```
01-02 00:03:17.739  2602  3217 V MediaPlayer: prepare
01-02 00:03:17.739   215  1141 V MediaPlayerService: [6]
setAudioStreamType(3)
01-02 00:03:17.739   215  1114 V MediaPlayerService: [6] prepareAsync
```

This call goes to the APM and checks for these conditions:

- audio.offload.disable prop is not set
- Clip duration is more than 1 min
- Stream type is music

If it gets the profiles for direct output, as follows:

```
01-02 00:03:17.739   215  3227 V AudioSystem: isOffloadSupported()
01-02 00:03:17.739   215  3227 V OMXCodec: matching
'OMX.google.mp3.decoder' quirks 0x00000000
01-02 00:03:17.739   215  3227 V OMXCodec: matching 'MP3Decoder' quirks
0x00000000
01-02 00:03:17.739   215  3227 V OMXCodec: Attempting to allocate OMX node
'OMX.google.mp3.decoder'
```

OMX decoder is created as a fallback.

```
01-02 00:03:17.739   215  3227 V OMXCodec: Successfully allocated OMX node
'OMX.google.mp3.decoder'
01-02 00:03:17.739   215  3227 V OMXCodec: configureCodec protected=0
```

```
01-02 00:03:17.739   215  3227 V AwesomePlayer: createAudioPlayer: bypass
OMX (offload)
01-02 00:03:17.739  2602  2614 V MediaPlayer: prepared
01-02 00:03:17.739  2602  3217 V MediaPlayer: prepare complete - status=0
```

Once initialized, the start () API is invoked, which kicks off media playback. StagefrightPlayer creates an instance of AudioPlayer and starts it to perform the actual playout of media.

```
01-02 00:03:17.749  2602  3217 V MediaPlayer: start
01-02 00:03:17.759   215  1114 V MediaPlayerService: [6] start
01-02 00:03:17.759   215  1114 V StagefrightPlayer: start
```

AudioSink, backed by the appropriate output interface (offload thread in this instance) created previously, is opened, which instantiates an AudioTrack instance.

The sample rate of the clip is 44.1 kHz and the number of channels is 2. In this case, the flag is set to AUDIO_OUTPUT_FLAG_COMPRESS_OFFLOAD. Types of output flags are defined in audio_output_flags_t in the file system/core/include/system/audio.h.

```
01-02 00:03:17.759   215  1114 V AudioPlayer: Mime type "audio/mpeg" mapped
to audio_format 0x1000000
01-02 00:03:17.759   215  1114 V AudioSink: open(48000, 2, 0x0, 0x1000000,
4, 12 0x10)
01-02 00:03:17.759   215  1114 V AudioSink: no track available to recycle
```

AudioSink creates a AudioTrack with stream parameters.

```
01-02 00:03:17.759   215  1114 V AudioSink: creating new AudioTrack
01-02 00:03:17.759   215  1114 V AudioTrack: sampleRate 48000, channelMask
0x3, format 16777216
01-02 00:03:17.759   215  1114 V AudioTrack: streamType 3
```

AudioTrack in turn creates an instance of AudioOuput, which is the entity associated with the stream abstraction of the various types of audio in Android.

```
01-02 00:03:17.759   215  1114 V AudioTrack: set() streamType 3 frameCount
0 flags 0010
01-02 00:03:17.759   215  1114 V AudioTrack: Offload request, forcing to
Direct Output
```

AudioTrack then calls getOutput(), which gets the profile information from the APM. The APM then calls openOutput(), which goes to AudioFlinger. AudioFlinger creates the offload thread based on the AUDIO_OUTPUT_FLAG_COMPRESS_OFFLOAD flag previously set by AudioPlayer.

```
01-02 00:03:17.759   215  1114 V AudioPolicyService: getOutput()
01-02 00:03:17.759   215  1114 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 2
01-02 00:03:17.759   215  1114 V AudioFlinger: openOutput(), module 1
Device 2, SamplingRate 48000, Format 0x1000000, Channels 3, flags 31
01-02 00:03:17.759   215  1114 V AudioFlinger: openOutput(), offloadInfo
0xb57feb18 version 0x0001
```

AudioFlinger then calls open_output_stream(), which goes to the HAL. HAL then creates the offload callback thread used to notify the write and drain completion events.

```
01-02 00:03:17.759   215  1114 V audio_hw_primary: adev_open_output_stream:
enter: sample_rate(48000) channel_mask(0x3) devices(0x2) flags(0x31)
01-02 00:03:17.759   215  1114 V audio_hw_primary: adev_open_output_stream:
offloaded output offload_info version 0001 bit rate 320000
01-02 00:03:17.759   215  1114 V audio_hw_primary: adev_open_output_stream:
exit
01-02 00:03:17.759   215  1114 V AudioFlinger: openOutput()
openOutputStream returned output 0xb88ab418, SamplingRate 48000, Format
0x1000000, Channels 3, status 0
01-02 00:03:17.759   215  1114 V audio_hw_primary: out_set_callback
01-02 00:03:17.759   215  1114 I AudioFlinger: HAL output buffer size 32768
frames, normal mix buffer size 32768 frames
01-02 00:03:17.759   215  3234 V audio_hw_primary: offload_thread_loop
01-02 00:03:17.759   215  3234 V audio_hw_primary: offload_thread_loop
offload_cmd_list 1 out->offload_state 0
01-02 00:03:17.759   215  3234 V audio_hw_primary: offload_thread_loop
SLEEPING
```

Once created, effects associated with the original session are also moved to the offload thread.

```
01-02 00:03:17.759   215  1114 V AudioFlinger: openOutput() created offload
output: ID 14 thread 0xb88bdf40
01-02 00:03:17.759   215  1114 V AudioFlinger:
PlaybackThread::audioConfigChanged_l, thread 0xb88bdf40, event 0, param 0
01-02 00:03:17.759   215  3236 I AudioFlinger: AudioFlinger's thread
0xb88bdf40 ready to run
01-02 00:03:17.759   215  1114 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 2
01-02 00:03:17.759   215  1114 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 2
01-02 00:03:17.759   215  1114 V AudioFlinger: moveEffects() session 0,
srcOutput 2, dstOutput 14
01-02 00:03:17.759   215  3236 V AudioFlinger: acquireWakeLock_l()
AudioOut_E status 0
01-02 00:03:17.759   215  3236 V SRS_ProcWS: SRS_Processing - SourceOutAdd
- No Available Slot for 0xb88bdf40
```

```
01-02 00:03:17.759   215  1114 V AudioFlinger: moveEffectChain_l() session
0 from thread 0xb5b74008 to thread 0xb88bdf40
01-02 00:03:17.759   215  1114 W AudioFlinger: moveEffectChain_l() effect
chain for session 0 not on source thread 0xb5b74008
01-02 00:03:17.759   215  3236 V AudioFlinger: releaseWakeLock_l()
AudioOut_E
```

AudioTrack then creates the track that is used for this stream.

```
01-02 00:03:17.759   215  1114 V AudioSystem: getLatency() streamType 3,
output 14, latency 96
01-02 00:03:17.759   215  1114 V AudioSystem: getFrameCount() streamType 3,
output 14, frameCount 32768
01-02 00:03:17.759   215  1114 V AudioSystem: getOutputSamplingRate()
reading from output desc
01-02 00:03:17.759   215  1114 V AudioSystem: getSamplingRate() streamType
3, output 14, sampling rate 48000
01-02 00:03:17.759   215  1114 V AudioTrack: createTrack_l() output 14
afLatency 96
01-02 00:03:17.759   215  1114 V AudioTrack: Offload: new frameCount =
65536
01-02 00:03:17.759   215  1114 V AudioFlinger: createTrack() sessionId: 12
01-02 00:03:17.759   215  1114 V AudioFlinger: createTrack() lSessionId: 12
01-02 00:03:17.759   215  3236 V AudioFlinger: thread 0xb88bdf40 type 4 TID
3236 going to sleep
01-02 00:03:17.759   215  1114 V AudioFlinger: Track constructor name 0,
calling pid 2602
01-02 00:03:17.759   215  1114 V AudioTrack: AUDIO_OUTPUT_FLAG_OFFLOAD
successful
01-02 00:03:17.759   215  1114 V AudioFlinger: acquiring 12 from 2602
01-02 00:03:17.759   215  1114 V AudioFlinger:  incremented refcount to 2
01-02 00:03:17.759   215  1114 V AudioSink: deleteRecycledTrack
01-02 00:03:17.759   215  1114 V AudioSink: setVolume
01-02 00:03:17.759   215  3236 V AudioFlinger: thread 0xb88bdf40 type 4 TID
3236 waking up
01-02 00:03:17.759   215  1114 V AudioSink: open() DONE status 0
```

AudioPlayer issues a start command to AudioSink, which then calls start_output_stream().

```
01-02 00:03:17.759   215  1114 V AudioFlinger: ThreadBase::setParameters()
music_offload_avg_bit_rate=320000;music_offload_sample_rate=48000
01-02 00:03:17.769   215   215 V MediaPlayerService: setNextPlayer
01-02 00:03:17.799   964  1143 D dalvikvm: GC_EXPLICIT freed 886K, 28% free
8847K/12248K, paused 2ms+3ms, total 36ms
01-02 00:03:17.799   215  3236 V AudioFlinger: acquireWakeLock_l()
AudioOut_E status 0
```

```
01-02 00:03:17.799   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 0
01-02 00:03:17.799   215  3236 V AudioFlinger: acquireWakeLock_l()
AudioOut_E status 0
01-02 00:03:17.819   215  3236 D audio_hw_primary: out_set_parameters:
enter: usecase(3: compress-offload-playback) kvpairs:
music_offload_avg_bit_rate=320000;music_offload_sample_rate=48000
01-02 00:03:17.819   215  3236 V audio_hw_primary: out_set_parameters:
exit: code(-2)
01-02 00:03:17.819   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 0
01-02 00:03:17.819   215  1114 V AudioSink: start
01-02 00:03:17.819   215  1114 V AudioFlinger: start(0), calling pid 2602
session 12
01-02 00:03:17.819   215  1114 V AudioFlinger: ? => ACTIVE (0) on thread
0xb88a8928
01-02 00:03:17.819   215  1114 V AudioPolicyService: startOutput()
01-02 00:03:17.819   215  1114 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 2
```

Compress offload playback is controlled from the offload thread created from AudioFlinger. It starts by AudioFlinger within threadLoop_mix getting the next buffer available from AudioTrack. It then writes the buffer to the HAL.

```
01-02 00:03:17.819   215  1114 V AudioFlinger: signal playback thread
01-02 00:03:17.819   215  3237 V AudioTrack: obtainBuffer(65536) returned
65536 = 65536 + 0 err 0
01-02 00:03:17.819   215  3237 V AudioPlayer: AudioCallback
01-02 00:03:17.819   215  3227 V MediaPlayerService: [6] notify
(0xb88c32a8, 6, 0, 0)
01-02 00:03:17.819  2602  3190 V MediaPlayer: message received msg=6,
ext1=0, ext2=0
01-02 00:03:17.819  2602  3190 V MediaPlayer: unrecognized message: (6, 0,
0)
01-02 00:03:17.819  2602  3217 V MediaPlayer: getCurrentPosition
01-02 00:03:17.819  2602  3190 V MediaPlayer: callback application
01-02 00:03:17.819  2602  3190 V MediaPlayer: back from callback
01-02 00:03:17.819   215   215 V MediaPlayerService: getCurrentPosition
01-02 00:03:17.819   215   215 V StagefrightPlayer: getCurrentPosition
01-02 00:03:17.819   215   215 V AudioPlayer: getOutputPlayPositionUs_l
59276000
01-02 00:03:17.819   215   215 V AudioPlayer: getMediaTimeUs
getOutputPlayPositionUs_l() playPosition = 59276000,
mPositionTimeRealUs 59276000
01-02 00:03:17.819   215   215 V MediaPlayerService: [6] getCurrentPosition
= 59276
01-02 00:03:17.819  2602  3217 E MediaPlayer: Should have subtitle
controller already set
```

```
01-02 00:03:17.819  2602  3217 V MediaPlayer: getCurrentPosition
01-02 00:03:17.819   215  1141 V MediaPlayerService: getCurrentPosition
01-02 00:03:17.819   215  1141 V StagefrightPlayer: getCurrentPosition
01-02 00:03:17.819   215  1141 V AudioPlayer: getOutputPlayPositionUs_l
59276000
01-02 00:03:17.819   215  1141 V AudioPlayer: getMediaTimeUs
getOutputPlayPositionUs_l() playPosition = 59276000,
mPositionTimeRealUs 59276000
01-02 00:03:17.819   215  1141 V MediaPlayerService: [6] getCurrentPosition
= 59276
01-02 00:03:17.819  2602  3217 V MediaPlayer: getCurrentPosition
01-02 00:03:17.819   215  1114 V MediaPlayerService: getCurrentPosition
01-02 00:03:17.819   215  1114 V StagefrightPlayer: getCurrentPosition
01-02 00:03:17.819   215  1114 V AudioPlayer: getOutputPlayPositionUs_l
59276000
01-02 00:03:17.819   215  1114 V AudioPlayer: getMediaTimeUs
getOutputPlayPositionUs_l() playPosition = 59276000,
mPositionTimeRealUs 59276000
01-02 00:03:17.819   215  1114 V MediaPlayerService: [6] getCurrentPosition
= 59276
01-02 00:03:17.819   215  3237 V AudioPlayer: getOutputPlayPositionUs_l
59276000
01-02 00:03:17.819   215  3237 V AudioPlayer: mPositionTimeMediaUs=60.91
mPositionTimeRealUs=59.28
01-02 00:03:17.819   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:03:17.819   215  3236 V AudioFlinger: acquireWakeLock_l()
AudioOut_E status 0
01-02 00:03:17.819   215  3236 V AudioTrackShared: mAvailToClient=0
stepCount=32768 minimum=32768
```

When the first buffer is written to the HAL, the HAL sets up audio routing and pushes down the calibration for the correct output device.

The HAL sends the appropriate mixer control to the back-end platform driver to configure the output device.

The HAL then opens the compress driver and sends the codec and stream parameters to the compress driver though the IOCTL SNDRV_COMPRESS_SET_PARAMS. The compress driver opens an ASM session accordingly and requests the ASM driver to allocate the buffers to be used for this session.

The output path for playback is opened up through Audio HAL. The device is speaker defined in /hardware/qcom/audio/hal/msm8974/platform.c.

```
01-02 00:03:17.819   215  3236 V audio_hw_primary: start_output_stream:
enter: usecase(3: compress-offload-playback) devices(0x2)
01-02 00:03:17.819   215  3236 V msm8974_platform:
platform_get_output_snd_device: enter: output devices(0x2)
```

```
01-02 00:03:17.819   215  3236 V msm8974_platform:
platform_get_output_snd_device: exit: snd_device(speaker)
01-02 00:03:17.819   215  3236 D audio_hw_primary: select_devices:
out_snd_device(2: speaker) in_snd_device(0: )
01-02 00:03:17.819   215  3236 D hardware_info: hw_info_append_hw_type :
device_name = speaker
01-02 00:03:17.819   215  3236 V audio_hw_primary: enable_snd_device:
snd_device(2: speaker)
01-02 00:03:17.819   215  3236 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 14, path =  0
01-02 00:03:17.819   215  3236 D ACDB-LOADER: ACDB -> send_adm_topology
01-02 00:03:17.819   215  3236 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
01-02 00:03:17.819   215  3236 D ACDB-LOADER: ACDB -> send_audtable
01-02 00:03:17.819   215  3236 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
01-02 00:03:17.819   215  3236 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
01-02 00:03:17.819   215  3236 D ACDB-LOADER: ACDB -> send_audvoltable
01-02 00:03:17.819   215  3236 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
01-02 00:03:17.819   215  3236 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
01-02 00:03:17.819   215  3236 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
01-02 00:03:17.819   215  3236 V listen_hal_loader:
audio_extn_listen_update_status(): no need to notify listen. device =
speaker. Event = 1
01-02 00:03:17.819   215  3237 V AudioTrack: obtainBuffer(65536) returned
32768 = 32768 + 0 err 0
```

Use case (3) is USECASE_AUDIO_PLAYBACK_OFFLOAD as defined in the hardware/qcom/audio/hal/audio_hw.h. The acdb_id = 14 means DEVICE_SPEAKER_MONO_RX_ACDB_ID.

```
01-02 00:03:17.819   215  3236 V audio_hw_primary: enable_audio_route:
enter: usecase(3)
01-02 00:03:17.819   215  3236 V audio_hw_primary: enable_audio_route:
apply mixer path: compress-offload-playback
01-02 00:03:17.819   215  3236 V audio_hw_primary: enable_audio_route: exit
```

card_id (0) is SOUND_CARD and device_id (0) is PLAYBACK_OFFLOAD_DEVICE as defined in hardware/qcom/audio/hal/msm8974/platform.h.

The end-to-end path is set up and audio is played out of the device. The offload thread starts to write data to the decoder.

The HAL then checks if playback had already started and, if not, starts it by calling compress_start().

At first, eight buffers are available. All eight buffers are written to at the start of playback and then the offload thread goes to sleep.

```
01-02 00:03:17.819   215  3236 V audio_hw_primary: start_output_stream:
Opening PCM device card_id(0) device_id(9)
01-02 00:03:17.869   215  3236 V offload_visualizer:
visualizer_hal_start_output output 14 pcm_id 9
01-02 00:03:17.869   215  3236 V audio_hw_primary: start_output_stream:
exit
01-02 00:03:17.869   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device
01-02 00:03:17.869   215  3236 V audio_hw_primary: send new gapless
metadata
01-02 00:03:17.869   215  3240 D offload_visualizer: thread enter
01-02 00:03:17.869   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device returned 32768
01-02 00:03:17.879   215  3237 V audio_hw_primary: out_get_render_position
rendered frames 0 sample_rate 48000
01-02 00:03:17.879   215  3237 V AudioPlayer: getOutputPlayPositionUs_l
59276000
01-02 00:03:17.879   215  3237 V AudioPlayer: mPositionTimeMediaUs=61.73
mPositionTimeRealUs=59.28
01-02 00:03:17.879   215  3237 V AudioTrack: obtainBuffer(32768) returned 0
= 0 + 0 err -11
01-02 00:03:17.879   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:03:17.879   215  3236 V AudioTrackShared: mAvailToClient=0
stepCount=32768 minimum=32768
01-02 00:03:17.879   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device
01-02 00:03:17.879   215  3237 V AudioTrack: obtainBuffer(32768) returned
32768 = 32768 + 0 err 0
01-02 00:03:17.879   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device returned 32768
01-02 00:03:17.879   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:03:17.879   215  3236 V AudioTrackShared: mAvailToClient=32768
stepCount=32768 minimum=32768
01-02 00:03:17.879   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device
01-02 00:03:17.879   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device returned 32768
01-02 00:03:17.879   215  3237 V audio_hw_primary: out_get_render_position
rendered frames 0 sample_rate 48000
01-02 00:03:17.879   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:03:17.879   215  3237 V AudioPlayer: getOutputPlayPositionUs_l
59276000
01-02 00:03:17.879   215  3237 V AudioPlayer: mPositionTimeMediaUs=62.54
mPositionTimeRealUs=59.28
```

```
01-02 00:03:17.879   215  3237 V AudioTrack: obtainBuffer(65536) returned
32768 = 32768 + 0 err 0
01-02 00:03:17.889   215  3237 V audio_hw_primary: out_get_render_position
rendered frames 240 sample_rate 48000
01-02 00:03:17.889   215  3237 V AudioPlayer: getOutputPlayPositionUs_l
59281000
01-02 00:03:17.889   215  3237 V AudioPlayer: mPositionTimeMediaUs=63.36
mPositionTimeRealUs=59.28
01-02 00:03:17.889   215  3237 V AudioTrack: obtainBuffer(32768) returned 0
= 0 + 0 err -11
01-02 00:03:17.889   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:03:17.889   215  3236 V AudioTrackShared: mAvailToClient=0
stepCount=32768 minimum=32768
01-02 00:03:17.889   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device
01-02 00:03:17.889   215  3237 V AudioTrack: obtainBuffer(32768) returned
32768 = 32768 + 0 err 0
01-02 00:03:17.899   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device returned 32768
01-02 00:03:17.899   215  3237 V audio_hw_primary: out_get_render_position
rendered frames 720 sample_rate 48000
01-02 00:03:17.899   215  3237 V AudioPlayer: getOutputPlayPositionUs_l
59291000
01-02 00:03:17.899   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:03:17.899   215  3237 V AudioPlayer: mPositionTimeMediaUs=64.18
mPositionTimeRealUs=59.29
01-02 00:03:17.899   215  3236 V AudioTrackShared: mAvailToClient=32768
stepCount=32768 minimum=32768
01-02 00:03:17.899   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device
01-02 00:03:17.899   215  3237 V AudioTrack: obtainBuffer(65536) returned
32768 = 32768 + 0 err 0
01-02 00:03:17.899   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device returned 32768
01-02 00:03:17.899   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:03:17.899   215  3236 V AudioTrackShared: mAvailToClient=32768
stepCount=32768 minimum=32768
01-02 00:03:17.899   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device
```

This write is a nonblocking call. A callback mechanism is used to notify about error or write complete events. It returns the actual number of bytes written; if written bytes are less than remaining bytes, it waits for an async callback from the HAL.

The following is a call to send_offload_cmd_l with command = 3, which indicates to wait for the buffer released by DSP.

```
01-02 00:03:17.899   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device returned 0
01-02 00:03:17.899   215  3236 V audio_hw_primary: send_offload_cmd_l 3
01-02 00:03:17.899   215  3236 V AudioFlinger: releaseWakeLock_l()
AudioOut_E
01-02 00:03:17.899   215  3234 V audio_hw_primary: offload_thread_loop
RUNNING
01-02 00:03:17.899   215  3234 V audio_hw_primary: offload_thread_loop
offload_cmd_list 0 out->offload_state 1
01-02 00:03:17.899   215  3234 V audio_hw_primary: offload_thread_loop
STATE 1 CMD 3 out->compr 0xb88aa568
01-02 00:03:17.899   215  3236 V AudioFlinger: wait async completion
01-02 00:03:17.899   215  3237 V audio_hw_primary: out_get_render_position
rendered frames 960 sample_rate 48000
01-02 00:03:17.899   215  3237 V AudioPlayer: getOutputPlayPositionUs_l
59296000
01-02 00:03:17.899   215  3237 V AudioPlayer: mPositionTimeMediaUs=64.99
mPositionTimeRealUs=59.30
01-02 00:03:17.899   215  3237 V AudioTrack: obtainBuffer(32768) returned
32768 = 32768 + 0 err 0
01-02 00:03:17.909   215  3237 V audio_hw_primary: out_get_render_position
rendered frames 960 sample_rate 48000
01-02 00:03:17.909   215  3237 V AudioPlayer: getOutputPlayPositionUs_l
59296000
01-02 00:03:17.909   215  3237 V AudioPlayer: mPositionTimeMediaUs=65.83
mPositionTimeRealUs=59.30
01-02 00:03:18.609   215  3234 V AudioFlinger: asyncCallback() event 0
01-02 00:03:18.609   215  3234 V audio_hw_primary: offload_thread_loop
offload_cmd_list 1 out->offload_state 1
01-02 00:03:18.609   215  3234 V audio_hw_primary: offload_thread_loop
SLEEPING
01-02 00:03:18.609   215  3235 V AudioFlinger: AsyncCallbackThread
mWriteAckSequence 27 mDrainSequence 0
```

The offload thread is woken up when a buffer becomes available, and it writes data to that buffer. The write continues until playback is stopped.

```
01-02 00:03:18.609   215  3236 V AudioFlinger: async completion/wake
01-02 00:03:18.609   215  3236 V AudioFlinger: acquireWakeLock_l()
AudioOut_E status 0
01-02 00:03:18.609   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:03:18.609   215  3236 V AudioFlinger: acquireWakeLock_l()
AudioOut_E status 0
```

```
01-02 00:03:18.609   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device
01-02 00:03:18.609   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device returned 32768
01-02 00:03:18.609   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:03:18.609   215  3236 V AudioTrackShared: mAvailToClient=0
stepCount=32768 minimum=32768
01-02 00:03:18.609   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device
01-02 00:03:18.609   215  3237 V AudioTrack: obtainBuffer(65536) returned
32768 = 32768 + 0 err 0
01-02 00:03:18.619   215  3236 V audio_hw_primary: out_write: writing
buffer (32768 bytes) to compress device returned 0
01-02 00:03:18.619   215  3236 V audio_hw_primary: send_offload_cmd_l 3
01-02 00:03:18.619   215  3234 V audio_hw_primary: offload_thread_loop
RUNNING
01-02 00:03:18.619   215  3234 V audio_hw_primary: offload_thread_loop
offload_cmd_list 0 out->offload_state 1
01-02 00:03:18.619   215  3234 V audio_hw_primary: offload_thread_loop
STATE 1 CMD 3 out->compr 0xb88aa568
01-02 00:03:18.619   215  3237 V audio_hw_primary: out_get_render_position
rendered frames 35280 sample_rate 48000
01-02 00:03:18.619   215  3237 V AudioPlayer: getOutputPlayPositionUs_l
60011000
01-02 00:03:18.619   215  3236 V AudioFlinger: releaseWakeLock_l()
AudioOut_E
01-02 00:03:18.619   215  3237 V AudioPlayer: mPositionTimeMediaUs=66.65
mPositionTimeRealUs=60.01
01-02 00:03:18.619   215  3237 V AudioTrack: obtainBuffer(32768) returned 0
= 0 + 0 err -11
01-02 00:03:18.619   215  3236 V AudioFlinger: wait async completion
```

When the EOS is reached, a drain event is generated to stop the stream.

```
01-02 00:05:38.669   215  3234 V AudioFlinger: asyncCallback() event 0
01-02 00:05:38.669   215  3235 V AudioFlinger: AsyncCallbackThread
mWriteAckSequence 1395 mDrainSequence 0
01-02 00:05:38.669   215  3236 V AudioFlinger: async completion/wake
01-02 00:05:38.669   215  3234 V audio_hw_primary: offload_thread_loop
offload_cmd_list 1 out->offload_state 1
01-02 00:05:38.669   215  3234 V audio_hw_primary: offload_thread_loop
SLEEPING
01-02 00:05:38.679   215  3236 V AudioFlinger: acquireWakeLock_l()
AudioOut_E status 0
01-02 00:05:38.679   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
```

```
01-02 00:05:38.689   215  3236 V AudioFlinger: acquireWakeLock_l()
AudioOut_E status 0
01-02 00:05:38.689   215  3236 V audio_hw_primary: out_write: writing
buffer (14912 bytes) to compress device
01-02 00:05:38.689   215  3236 V audio_hw_primary: out_write: writing
buffer (14912 bytes) to compress device returned 14912
01-02 00:05:38.689   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:05:38.689   215  3236 V AudioFlinger: OffloadThread: underrun and
STOPPING_1 -> draining, STOPPING_2
01-02 00:05:38.689   215  3236 V AudioFlinger: draining early
01-02 00:05:38.689   215  3236 V audio_hw_primary: out_drain
01-02 00:05:38.689   215  3236 V audio_hw_primary: send_offload_cmd_l 2
01-02 00:05:38.689   215  3236 V AudioFlinger: releaseWakeLock_l()
AudioOut_E
01-02 00:05:38.689   215  3236 V AudioFlinger: wait async completion
01-02 00:05:38.689   215  3234 V audio_hw_primary: offload_thread_loop
RUNNING
01-02 00:05:38.689   215  3234 V audio_hw_primary: offload_thread_loop
offload_cmd_list 0 out->offload_state 1
01-02 00:05:38.689   215  3234 V audio_hw_primary: offload_thread_loop
STATE 1 CMD 2 out->compr 0xb88aa568
```

Once drain is completed, AwesomePlayer is notified of the end of the stream.

```
01-02 00:05:42.349   215  3234 V AudioFlinger: asyncCallback() event 1
01-02 00:05:42.349   215  3234 V audio_hw_primary: offload_thread_loop
offload_cmd_list 1 out->offload_state 1
01-02 00:05:42.349   215  3235 V AudioFlinger: AsyncCallbackThread
mWriteAckSequence 1396 mDrainSequence 7
01-02 00:05:42.349   215  3236 V AudioFlinger: async completion/wake
01-02 00:05:42.349   215  3234 V audio_hw_primary: offload_thread_loop
SLEEPING
01-02 00:05:42.349   215  3236 V AudioFlinger: acquireWakeLock_l()
AudioOut_E status 0
01-02 00:05:42.349   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
01-02 00:05:42.349   215  3236 V AudioFlinger: presentationComplete()
mPresentationCompleteFrames 0 framesWritten 5782080
01-02 00:05:42.349   215  3236 V AudioFlinger: presentationComplete()
reset: mPresentationCompleteFrames 5786688 audioHalFrames 4608
01-02 00:05:42.349   215  3236 V AudioFlinger: presentationComplete()
session 12 complete: framesWritten 5782080
01-02 00:05:42.349   215  3236 V AudioFlinger: removeTracks_l removing
track on session 12
01-02 00:05:42.349   215  3237 V AudioTrackShared: stream end received
01-02 00:05:42.349   215  3237 V AudioSink: callbackwrapper: deliver
EVENT_STREAM_END
```

```
01-02 00:05:42.349   215  3237 V AudioPlayer: AudioSinkCallback: stream end
01-02 00:05:42.349   215  3237 V AudioPlayer: AudioPlayer@0x0xb88be9b8
notifyAudioEOS
01-02 00:05:42.349   215  3236 V AudioFlinger: acquireWakeLock_l()
AudioOut_E status 0
01-02 00:05:42.349   215  3237 V AudioPlayer: Notified observer of EOS!
01-02 00:05:42.349   215  3227 V AwesomePlayer: MEDIA_PLAYBACK_COMPLETE
01-02 00:05:42.349   215  3227 V MediaPlayerService: [6] notify
(0xb88c32a8, 2, 0, 0)
01-02 00:05:42.359  2602  2843 V MediaPlayer: message received msg=2,
ext1=0, ext2=0
01-02 00:05:42.359   215  3227 V MediaPlayerService: [6] notify
(0xb88c32a8, 7, 0, 0)
01-02 00:05:42.359  2602  2843 V MediaPlayer: playback complete
01-02 00:05:42.359  2602  2843 V MediaPlayer: callback application
01-02 00:05:42.359  2602  2843 V MediaPlayer: back from callback
```

At this moment, the session is closed.

Output is closed, buffers are deallocated, and the offload thread is stopped and released. Finally, the routing for the use case is reset.

```
01-02 00:05:42.359  2602  2843 V MediaPlayer: back from callback
01-02 00:05:42.359   215  3227 V AudioSink: stop
01-02 00:05:42.359  2602  2843 V MediaPlayer: message received msg=7,
ext1=0, ext2=0
01-02 00:05:42.359  2602  2843 V MediaPlayer: unrecognized message: (7, 0,
0)
01-02 00:05:42.359  2602  2843 V MediaPlayer: callback application
01-02 00:05:42.359  2602  2843 V MediaPlayer: back from callback
01-02 00:05:42.359   215  3236 V AudioPolicyService: stopOutput()
01-02 00:05:42.359   215  3236 V AudioPolicyService: inserting command: 5
at index 0, num commands 0
01-02 00:05:42.359   215  3236 V AudioPolicyService: AudioCommandThread()
adding stop output 14
01-02 00:05:42.359   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 0
01-02 00:05:42.359   215   610 V AudioPolicyService: AudioCommandThread()
waking up
01-02 00:05:42.359   215   610 V AudioPolicyService: AudioCommandThread()
processing stop output 14
01-02 00:05:42.359   215   610 V AudioPolicyService: doStopOutput from tid
610
01-02 00:05:42.359   215   610 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 2
01-02 00:05:42.369   215   610 V AudioPolicyService: AudioCommandThread()
going to sleep
01-02 00:05:42.369   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 0
```

```
01-02 00:05:42.379   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 0
01-02 00:05:42.389   215  3236 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 0
01-02 00:05:43.359   215  3236 V AudioFlinger: Audio hardware entering
standby, mixer 0xb88bdf40, suspend count 0
01-02 00:05:43.359   215  3236 V audio_hw_primary: out_standby: enter:
usecase(3: compress-offload-playback)
01-02 00:05:43.419   215  3236 V audio_hw_primary: stop_output_stream:
enter: usecase(3: compress-offload-playback)
01-02 00:05:43.419   215  3236 V offload_visualizer:
visualizer_hal_stop_output output 14 pcm_id 9
01-02 00:05:43.419   215  3240 D offload_visualizer: thread exit
01-02 00:05:43.419   215  3236 V audio_hw_primary: disable_audio_route:
enter: usecase(3)
01-02 00:05:43.419   215  3236 V audio_hw_primary: disable_audio_route:
reset mixer path: compress-offload-playback
01-02 00:05:43.419   215  3236 V audio_hw_primary: disable_audio_route:
exit
01-02 00:05:43.419   215  3236 D hardware_info: hw_info_append_hw_type :
device_name = speaker
01-02 00:05:43.419   215  3236 V audio_hw_primary: disable_snd_device:
snd_device(2: speaker)
01-02 00:05:43.429   215  3236 V listen_hal_loader:
audio_extn_listen_update_status(): no need to notify listen. device =
speaker. Event = 0
01-02 00:05:43.429   215  3236 V audio_hw_primary: stop_output_stream:
exit: status(0)
01-02 00:05:43.429   215  3236 V audio_hw_primary: out_standby: exit
01-02 00:05:43.429   215  3236 V AudioFlinger: releaseWakeLock_l()
AudioOut_E
01-02 00:05:43.429   215  3236 V AudioFlinger: thread 0xb88bdf40 type 4 TID
3236 going to sleep
01-02 00:05:52.369   215  3227 W TimedEventQueue: delay_us exceeds max
timeout: 49997322 us
```

8.4.1.1 Music playback pause

Audio is paused from the media player, which causes playback to be paused. Output is stopped and the offload thread remains in the sleeping state.

```
12-13 14:47:45.840  4459  4459 V MediaPlayer: pause
12-13 14:47:45.840  4409  4426 V StagefrightPlayer: pause

12-13 14:47:45.840  4409  4426 V AudioSink: pause
12-13 14:47:45.840  4409  4554 V AudioTrackShared: obtainBuffer()
interrupted by client
12-13 14:47:45.840  4409  4554 V AudioTrack: obtainBuffer(32768) returned 0
= 0 + 0 err -4
```

```
12-13 14:47:45.840  4409  4426 V AudioFlinger: pause(0), calling pid 4459
12-13 14:47:45.840  4409  4426 V AudioFlinger: ACTIVE/RESUMING => PAUSING
(0) on thread 0xb7d6cac0
12-13 14:47:45.840  4409  4553 V AudioFlinger: async completion/wake
12-13 14:47:45.840  4409  4553 V AudioFlinger: acquireWakeLock_l()
AudioOut_9 status 0
12-13 14:47:45.840  4409  4553 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 1
12-13 14:47:45.840  4409  4553 V audio_hw_primary: out_pause
12-13 14:47:45.840  4409  4553 V AudioFlinger: removeTracks_l removing
track on session 8

12-13 14:47:45.850  4409  4553 V AudioPolicyService: stopOutput()
12-13 14:47:45.850  4409  4421 V AudioPolicyManagerBase: stopOutput()
output 9, stream 3, session 8
12-13 14:47:45.850  4409  4421 V AudioPolicyManagerBase: changeRefCount()
stream 3, count 0

12-13 14:47:45.850  4409  4421 V AudioPolicyManagerBase: getNewDevice()
selected device 0
12-13 14:47:45.850  4409  4421 V AudioPolicyManagerBase: setOutputDevice()
output 9 device 0000 delayMs 192
12-13 14:47:45.850  4409  4421 V AudioPolicyManagerBase: setOutputDevice()
prevDevice 0008
12-13 14:47:45.850  4409  4421 V AudioPolicyManagerBase: setOutputDevice()
setting same device 0000 or null device for output 9
```

If there is no activity after 60 sec of playback, the teardown function is called, which stops and releases all components involved. As part of this process, the decoder is released and any remaining data in the HAL is flushed.

```
12-13 14:48:45.860  4409  4527 V AwesomePlayer: onAudioTearDownEvent
12-13 14:48:45.860  4409  4527 V OMXNodeInstance: calling
destroyComponentInstance
12-13 14:48:45.870  4409  4527 V OMXNodeInstance: destroyComponentInstance
returned err 0
12-13 14:48:45.870  4409  4527 V OMXNodeInstance: OMXNodeInstance going
away.

12-13 14:48:45.870  4409  4527 V AudioSink: stop
12-13 14:48:45.870  4409  4527 V AudioSink: flush

12-13 14:48:45.880  4409  4527 V AudioFlinger: flush(0)
12-13 14:48:45.880  4409  4527 V AudioFlinger: flush: offload flush
12-13 14:48:45.880  4409  4553 V AudioFlinger: async completion/wake
12-13 14:48:45.890  4409  4553 V AudioFlinger: acquireWakeLock_l()
AudioOut_9 status 0
```

```
12-13 14:48:45.890  4409  4553 V AudioFlinger:
OffloadThread::prepareTracks_l active tracks 0
12-13 14:48:45.890  4409  4553 V audio_hw_primary: out_flush
```

Output is closed, buffers are deallocated, and the offload thread is stopped and released. Finally, the routing for the use case is reset.

```
12-13 14:48:45.890  4409  4527 V AudioSink: close
12-13 14:48:45.890  4409  4551 V audio_hw_primary: offload_thread_loop
SLEEPING
12-13 14:48:45.890  4409  4552 V AudioFlinger: AsyncCallbackThread
mWriteAckSequence 355 mDrainSequence 0

12-13 14:48:45.900  4409  4527 V AudioPolicyService: releaseOutput()
12-13 14:48:45.900  4409  4421 V AudioPolicyService: doReleaseOutput from
tid 4421
12-13 14:48:45.900  4409  4421 V AudioPolicyManagerBase: releaseOutput() 9
12-13 14:48:45.900  4409  4421 V AudioPolicyManagerBase: closeOutput(9)
12-13 14:48:45.900  4409  4420 V AudioFlinger: ThreadBase::setParameters()
closing=true
12-13 14:48:45.900  4409  4527 V AudioFlinger: PlaybackThread::Track
destructor

12-13 14:48:45.900  4409  4527 V AudioPlayer: AudioPlayer releasing input
buffer.
12-13 14:48:45.910  4409  4553 D audio_hw_primary: out_set_parameters:
enter: usecase(3: compress-offload-playback) kvpairs: closing=true
12-13 14:48:45.910  4409  4553 V audio_hw_primary: out_set_parameters:
exit: code(-2)
12-13 14:48:45.910  4409  4421 V AudioSystem: ioConfigChanged() output 9
closed
12-13 14:48:45.910  4409  4421 V AudioFlinger: ThreadBase::exit
12-13 14:48:45.910  4409  4421 V AudioFlinger:   preExit()
12-13 14:48:45.910  4409  4421 D audio_hw_primary: out_set_parameters:
enter: usecase(3: compress-offload-playback) kvpairs: exiting=1
12-13 14:48:45.910  4409  4421 V audio_hw_primary: out_set_parameters:
exit: code(-2)
12-13 14:48:45.910  4459  4492 V AudioSystem: ioConfigChanged() output 9
closed
```

8.4.2 Kernel logs

```
<7>[  227.582466] msm_pcm_routing_process_audio: reg 2 val 3 set 1
```

This command indicates the PCM routing to BACKEND DAI ID 2 (MSM_BACKEND_DAI_SLIMBUS_0_RX,) from FRONTEND DAI ID 3 (MSM_FRONTEND_DAI_MULTIMEDIA4). The BACKEND DAI IDs and FRONTEND DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[  227.583277] q6asm_audio_client_alloc: session[2]
<7>[  227.586618] msm_compr_open: req: 1
<7>[  227.583243] q6asm_audio_client_alloc Registering the common port with
APR
<7>[  227.583254] q6asm_audio_client_alloc: mem_map_handle list init'ed
<7>[  227.583267] send_asm_custom_topology: no cal to send addr= 0xe16ffcbc
<7>[  227.583277] q6asm_audio_client_alloc: session[2]
<6>[  227.583286] msm_compr_open: session ID 2
<7>[  227.586605] populate_codec_list
<7>[  227.586618] msm_compr_open: req: 1
<7>[  227.586877] q6asm_write: WRITE SUCCESS
<7>[  227.590213] SNDRV_COMPRESS_GET_CAPS
<7>[  227.590270] SNDRV_COMPRESS_SET_PARAMS:
<7>[  227.590280] SND_AUDIOCODEC_MP3

<7>[  227.590368] msm_compr_hw_params
<7>[  227.590379] __q6asm_open_write: session[2] wr_format[0x4]
<7>[  227.597972] adm_open: port 0x4000 path:1 rate:48000 mode:1
perf_mode:0
<7>[  227.597986] adm_open: Port ID 0x4000, index 15
<7>[  227.597998] send_adm_custom_topology: no cal to send addr= 0xe16ffc58
<7>[  227.598008] adm_open:opening ADM: perf_mode: 0
<7>[  227.598020] adm_open: port_id=0x4000 rate=48000 topology_id=0x10314
```

The following log indicates that four buffers are allocated, each of size 256 kB. This code snippet shows the msm_pcm_hardware structure that defines buffer size, period size, supported sample rate, supported number of channels, and supported ALSA PCM parameters.

```
<7>[  227.620268] q6asm_audio_client_buf_alloc_contiguous:
session[2]bufsz[245760]bufcnt[4]
<7>[  227.622498] q6asm_audio_client_buf_alloc_contiguous
data[f0e3c000]phys[3f2ac000][cdf5071c]
<7>[  227.622516] q6asm_audio_client_buf_alloc_contiguous
data[f0e78000]phys[3f2e8000][cdf50738]
<7>[  227.622530] q6asm_audio_client_buf_alloc_contiguous
data[f0eb4000]phys[3f324000][cdf50754]
<7>[  227.622542] q6asm_memory_map_regions: Session[2]
```

```
<7>[  227.622566] mmap_region=0xe3a36440 token=0x200
<7>[  227.622575] map_regions->nregions = 1
<7>[  227.623187]
q6asm_mmapcallback:ptr0[0xf0993f88]ptr1[0x0]opcode[0x10d93]
token[0x200]payload_s[4] src[0] dest[0]sid[2]dir[0]
<7>[  227.623207] q6asm_mmapcallback:Payload = [0xf0993f88] status[0x0]
<7>[  227.623220] q6asm_mmapcallback:PL#0[0xf0993f88]PL#1 [0x0] dir=0
s_id=2
<7>[  227.623248] compr_event_handler opcode =00010d93
```

Allocate memory for four buffers, each of size 256 kB.

```
<7>[  227.623318] q6asm_memory_map_regions: i=0, bufadd[i] = 0x3f270000,
maphdl[i] = 0xf0993f88
<7>[  227.623332] q6asm_memory_map_regions: i=1, bufadd[i] = 0x3f2ac000,
maphdl[i] = 0xf0993f88
<7>[  227.623345] q6asm_memory_map_regions: i=2, bufadd[i] = 0x3f2e8000,
maphdl[i] = 0xf0993f88
<7>[  227.623359] q6asm_memory_map_regions: i=3, bufadd[i] = 0x3f324000,
maphdl[i] = 0xf0993f88
<7>[  227.623374] q6asm_memory_map_regions: exit
<7>[  227.623393] msm_compr_hw_params: buf[cdf50700]dma_buf-
>area[f0e00000]dma_buf->addr[3f270000]
```

A prepare() is called to pass in the number of channels, bit rate, and so on.

```
<7>[  227.624512] compressed stream prepare
<7>[  203.791056] q6asm_set_io_mode ac->mode after anding with FF00:0x[0],
<7>[  203.791066] q6asm_set_io_mode:Set Mode to 0x[42]
<7>[  203.791077] __q6asm_media_format_block_pcm:session[2]rate[44100]ch[2]
<7>[  203.791090] q6asm_add_hdr:pkt_size=44 cmd_flg=1 session=2
<7>[  203.791101] q6asm_map_channels channels passed: 2
```

Playback is started by calling msm_pcm_trigger().

```
<7>[  227.658053] msm_compr_trigger
<7>[  227.658064] msm_compr_trigger: Trigger start
<7>[  227.658082] pkt_size = 32, cmd_flg = 1, session = 2
<7>[  227.658895] q6asm_callback: nowait_cmd_cnt 1
<7>[  227.658909] q6asm_callback: session[2]opcode[0x110e8]
token[0x2]payload_s[8] src[513] dest[513]
<7>[  227.658921] q6asm_callback:Payload = [0x10daa] status[0x0]
<7>[  227.658931] q6asm_callback:Payload = [0x10daa]
<7>[  227.658940] q6asm_callback:Payload = [0x10daa]stat[0x0]
<7>[  227.658950] compr_event_handler opcode =000110e8
<7>[  227.658960] compr_event_handler:writing 245760 bytes of buffer[0] to
dsp
```

```
<7>[  227.658972] compr_event_handler:writing buffer[0] from 0x3f270000
<7>[  227.658983] meta_data_length: 64, frame_length: 180558
<7>[  227.658993] timestamp_msw: 0, timestamp_lsw: 0
<7>[  227.659002] pkt_size = 52, cmd_flg = 0, session = 2
```

The DSP uses the buffer and triggers the ASM_DATA_EVENT_WRITE_DONE when the kernel wakes up the user space to send more data.

```
<7>[  239.629116] ASM_DATA_EVENT_WRITE_DONE
<7>[  239.629131] Buffer Consumed = 0xc0b05d08
<7>[  239.632824] q6asm_callback: session[1]opcode[0x10d99]
token[0x1]payload_s[16] src[257] dest[257]
<7>[  239.632848] q6asm_callback:Payload = [0x3f26e400] status[0x0]
<7>[  239.632868] q6asm_callback: Rxed opcode[0x3f26e400] status[0x0]
token[1]
<7>[  239.633029] q6asm_is_cpu_buf_avail:session[1]index[1]
data[f03d4400]size[1024]
<7>[  239.633054] q6asm_write: session[1] len=1024
<7>[  239.633073] q6asm_add_hdr:pkt_size=52 cmd_flg=0 session=1
<7>[  239.633153] q6asm_write:ab->phys[0x3f26e400]bufadd[0x3f26e400]
token[0x1]buf_id[0x1]buf_size[0x400]mmaphdl[0xf0988e58]
<7>[  239.633212] q6asm_write: WRITE SUCCESS
<7>[  239.633497] q6asm_callback: session[2]opcode[0x110e8]
token[0x2]payload_s[8] src[513] dest[513]
<7>[  239.633521] q6asm_callback:Payload = [0x10bce] status[0x0]
<7>[  239.633538] q6asm_callback:Payload = [0x10bce]
<7>[  239.633555] q6asm_callback:Payload = [0x10bce]stat[0x0]
```

Playback is paused, which causes the DSP to flush its buffers.

```
<7>[  236.629526] msm_compr_trigger
<7>[  236.629534] SNDRV_PCM_TRIGGER_PAUSE
<7>[  236.629543] pkt_size = 20, cmd_flg = 1, session = 2
<7>[  236.629552] q6asm_cmd_nowait:CMD_PAUSE
<7>[  236.629561] q6asm_cmd_nowait:session[2]opcode[0x10bd3]

<7>[  239.628296] q6asm_cmd:CMD_FLUSH
<7>[  239.628313] q6asm_cmd:session[2]opcode[0x10bce]
<7>[  239.629030] q6asm_callback: session[2]opcode[0x10d99]
token[0x3f270040]payload_s[16] src[513] dest[513]
<7>[  239.639347] ASM_STREAM_CMD_FLUSH
<7>[  239.640329] q6asm_cmd:session[2]opcode[0x10bcd]
<7>[  239.643869] q6asm_callback: session[1]opcode[0x10d99]
token[0x3]payload_s[16] src[257] dest[257]
```

Finally, the ASM session is closed, which releases the DSP buffers and completes playout.

```
<7>[  239.640255] msm_compr_playback_close
<7>[  239.640313] q6asm_cmd:CMD_CLOSE

<7>[  239.640329] q6asm_cmd:session[2]opcode[0x10bcd]
<7>[  239.645766] q6asm_callback: session[2]opcode[0x110e8]
token[0x2]payload_s[8] src[513] dest[513]
<7>[  239.646013] q6asm_memory_unmap: Session[2]
<7>[  239.646033] q6asm_add_mmaphdr:pkt size=24 cmd_flg=1
<7>[  239.646051] q6asm_memory_unmap: Found the element
<7>[  239.646068] q6asm_memory_unmap: mem_unmap-mem_map_handle: 0xf0993f88
<7>[  239.647780] q6asm_mmapcallback:ptr0[0x10d94]ptr1[0x0]opcode[0x110e8]
token[0x200]payload_s[8] src[0] dest[0]sid[2]dir[0]
<7>[  239.647806] q6asm_mmapcallback:Payload = [0x10d94] status[0x0]
<7>[  239.647840] q6asm_mmapcallback:Payload = [0x10d94] status[0x0]
<7>[  239.648024]
q6asm_audio_client_buf_free_contiguous:data[f0e00000]phys[3f270000][cdf5070
0] , client[cdf50380] handle[e3a36540]
<7>[  239.648063] adm_close port_id=0x4000 index 15 perf_mode: 0
<7>[  239.648082] adm_close:Closing ADM: perf_mode: 0
<7>[  239.648102] adm_close:coppid 1 portid=0x4000 index=15 coppcnt=0
<7>[  239.648828] q6asm_callback: session[1]opcode[0x10d99]
token[0x4]payload_s[16] src[257] dest[257]
<7>[  239.648853] q6asm_callback:Payload = [0x3f26f000] status[0x0]
<7>[  239.648875] q6asm_callback: Rxed opcode[0x3f26f000] status[0x0]
token[4]
<7>[  239.653176] q6asm_audio_client_free: Session id 2
<7>[  239.653196] q6asm_session_free: sessionid[2]
<7>[  239.653217] q6asm_audio_client_free: APR De-Register
<7>[  239.653607] msm-pcm-routing msm-pcm-routing: reg 0
<7>[  239.654224] msm_pcm_routing_process_audio: reg 2 val 3 set 0
```

8.5 Customization

8.5.1 Enable compress offload playback

Compress offload playback is enabled by default on the baseline software. To ensure that the settings are correct, use the following commands:

```
adb root
adb shell setprop audio.offload.disable 0
adb shell getprop | grep off
```

To enable Compress Offload mode playback statically in a build, set these values in the system.prop file, which is at \device\qcom\msm8874, and recompile.

8.5.2 Customize buffer size

Buffer size is customizable using the following property:

```
adb shell audio.offload.buffer.size.kbytes=32
```

To enable Compress Offload mode playback statically in a build, set these values in the system.prop file, which is at \device\qcom\msm8874, and recompile.

8.5.3 Enable compress offload for AV playback

Enabling compress offload for audio video playback is possible using below property:

```
adb shell av.offload.enable=true
```

To enable Compress Offload mode playback statically in a build, set these values in the system.prop file, which is at \device\qcom\msm8874, and recompile.

8.6 Debugging

Table 8-1 lists common playback issues that were encountered during previous integration activities and some troubleshooting steps to resolve them.

Table 8-1 Debugging and troubleshooting tips

Issue type	Symptoms	Troubleshooting steps
Audio mute	No audio is heard from the output device	▪ Check the software volume, DSP volume, WCD codec digital, and analog gain. ▪ Check that data logged at the AFE (DSP output) is proper using QXDM Professional logging. ▪ Check mixer controls to verify if the codec path and DSP routing are correct for the specific output device.
Noise	Noise at the start, end, or pause/resume of playback	Check if DSP buffers are not flushed, or if the flush command is failing. Logging PCM data at the AFE using QXDM Professional can help narrow down the issue.
	Pop noise when disabling/enabling device	Check that the bringup or power-down sequence for the QCD codec is correct.
	Noise when changing audio effects	Check PCM data before and after the effects module to determine if the noise is introduced in the effects module. Such issues are seen during LPA playback.
	Continuous background noise during playback	Check if PCM data at the AFE is correct using QXDM Professional. Capture codec-related logs to isolate codec issues. Hardware teams may need to be involved to root cause such issues.
Volume	Volume is not at the value expected or changes unexpectedly, for example, during concurrency scenarios	Check user space logs, calibration data for the output device used, and audio effects applied.

Issue type	Symptoms	Troubleshooting steps
Discontinuous audio	Audio playback contains gaps	Check PCM data at the user space and DSP output; if the gaps are isolated to the user space, buffer management logic, and CPU loading are analyzed.
Configuration	Audio is either slow or fast	▪ Check user space and kernel logs for sample rate settings. ▪ Check for channel misconfiguration from user space and kernel logs, PCM data at AFE, and WCD register settings.
Track management	Audio repetition	Check PCM data at the user space and DSP output. Such issues are caused due to buffer management or seek operation errors.
	Seeking does not work as expected	▪ Check user space implementation that handles DSP timestamp information. ▪ Check user space logs and kernel logs for DSP timestamps to help isolate the issue.
	Progress bar does not move with playback	▪ Check user space implementation that handles DSP timestamp information. ▪ Check whether issue is seen with a specific type of playback (normal, LPA, or Tunnel).
	Issue when switching from one song to another	▪ Check if the playback state is bad. ▪ Check user space logs.
Codec	Media formats not supported	▪ Check if the codec chosen to decode the bit stream is correct; sometimes the parser may not be able to detect the type of bit stream and choose a wrong codec. ▪ Check if switching between hardware and software codec helps resolve the issue.
Performance	Playback does not start immediately, resuming playback takes time, routing audio to a different device takes time, and so on.	Carefully examine log timestamps to determine which component or step is lagging. Power top logs can help further analyze the issue.
Power consumption	Power consumption is higher than expected	Check clock dumps for clocks enabled and the frequency of the clocks. Power rail breakdown can pinpoint exactly the hardware block consuming extra power.
Stability	Crash during stress testing	Check the stack trace captured in logs to isolate the issue.

9 USB audio

9.1 USB dock

QTI has not modified the standard Android behavior; only the standard Google features are supported.

In the case of USB dock, AudioPlayer uses a normal AudioTrack. AudioPlayer registers with AudioFlinger for this requirement to get the notification on when a USB device capable of audio playback is connected. AudioPlayer also handles different cases, for example, pause, resume, and suspend/resume. It also takes care of starting playback from the correct position when a device switch happens to USB.

Figure 9-1 shows the path taken by frameworks as it sends the decoded PCM data to the USB audio HAL.

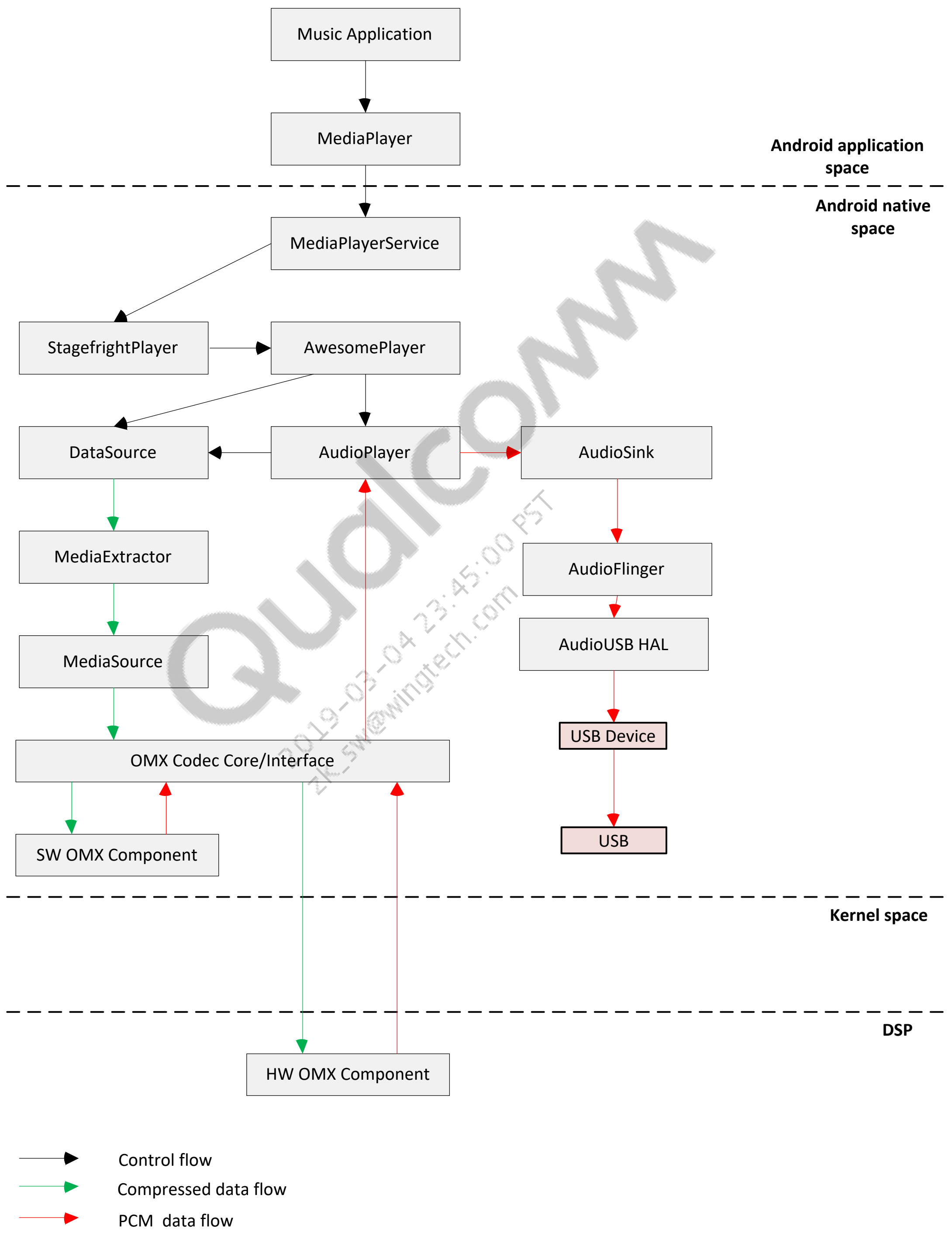


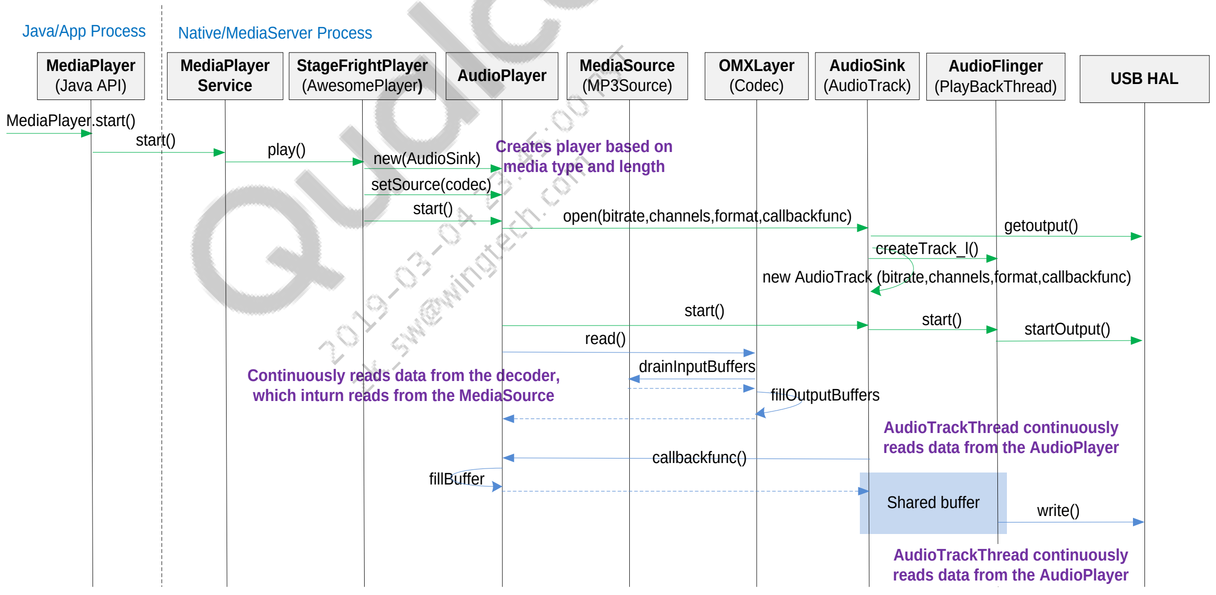
Figure 9-1 USB dock diagram

The MediaPlayerService is a system service that links the Android user space to the media framework. Stagefright is the native Android media framework that facilitates media playback and capture through various physical and proxy devices. Other component functions are:

- MediaExtractor is responsible for retrieving track data and the corresponding metadata from the underlying file system or http stream.
- OMX layer is used for decoding/encoding data to and from raw PCM data into appropriate audio formats as supported by the system. Software or hardware instances of codecs are used to service a request depending on the capabilities of a platform.
- AudioPlayer is responsible for rendering audio and providing timebase for AV synchronization in AwesomePlayer, if an audio track is present.
- AwesomePlayer works as the engine to coordinate the modules and is finally connected into the Android media framework through the adapter that is the StagefrightPlayer.
- AudioFlinger resamples and routes the media stream to the actual devices that are involved with the output or input operation.

9.1.1 Call flow

Figure 9-2 shows the call and data flows during active audio playback.



Java/App Process
Native/MediaServer Process
MediaPlayer (Java API)
MediaPlayer Service
StageFrightPlayer (AwesomePlayer)
AudioPlayer
MediaSource (MP3Source)
OMXLayer (Codec)
AudioSink (AudioTrack)
AudioFlinger (PlayBackThread)
USB HAL
MediaPlayer.start()
start()
play()
new(AudioSink)
Creates player based on media type and length
setSource(codec)
start()
open(bitrate,channels,format,callbackfunc)
getoutput()
createTrack_l()
new AudioTrack (bitrate,channels,format,callbackfunc)
start()
start()
startOutput()
read()
drainInputBuffers
fillOutputBuffers
Continuously reads data from the decoder, which inturn reads from the MediaSource
AudioTrackThread continuously reads data from the AudioPlayer
callbackfunc()
fillBuffer
Shared buffer
write()
AudioTrackThread continuously reads data from the AudioPlayer

Figure 9-2 USB dock call flow

9.1.2 Log collection

This section highlights the various files and/or components in which logging is enabled to troubleshoot issues related to audio playback on Android.

9.1.2.1 User space logs

This command clears the current logcat logs and starts logging the user space logs to the file logcat.txt and the stdout simultaneously.

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

NOTE: For the tee commands to work, Cygwin is installed or the command is executed in a Linux environment.

9.1.2.2 Kernel logs

Not required.

9.1.2.3 Additional logging

Depending on the issue, additional logging is enabled in the user space and kernel space.

In the user space, enable logs on files listed at the following locations:

```
\harwdware\libhardware\modules\usbaudio
audio_hw.c

\hardware\qcom\audio\hal\audio_extn
usb.c

\frameworks\av\media\libmediaplayerservice
MediaPlayerService.cpp
StagefrightPlayer.cpp

\frameworks\av\media\libmedia
AudioSystem.cpp
AudioTrack.cpp
mediaplayer.cpp

\frameworks\av\media\libstagefright
AwesomePlayer.cpp
AudioPlayer.cpp
MP3Extractor.cpp
OMXClient.cpp
OMXCodec.cpp
```

```
\frameworks\av\services\audioflinger
AudioFlinger.cpp
AudioMixer.cpp
AudioPolicyService.cpp

\harwdware\qcom\audio\hal
audio_hw.c

\harwdware\qcom\audio\hal\msm8974
platform.c
hw_info.c

\harwdware\qcom\audio\hal\policy_hal
AudioPolicyManager.cpp

\harwdware\libhardware_legacy\audio
AudioPolicyManagerBase.cpp
```

9.1.3 Log analysis

9.1.3.1 User space logs

The USB HAL is loaded by AudioFlinger during bootup.

```
01-02 00:00:01.140   194   194 I AudioFlinger: loadHwModule() Loaded usb
audio interface from USB audio HW HAL (audio) handle 5
```

When a USB device capable of USB audio playback is plugged in, it registers a sound card with the Audio Service.

```
01-02 00:01:45.109  1051  1051 V AudioService: Broadcast Receiver: Got
ACTION_USB_AUDIO_ACCESSORY_PLUG, state = 1, card: 1, device: 0
01-02 00:01:45.109   194   194 V AudioFlinger: openOutput(), module 1
Device 2000, SamplingRate 8000, Format 0x000001, Channels 1, flags 8001
01-02 00:01:45.109   194   194 V AudioFlinger: openOutput(), offloadInfo
0xbee45518 version 0x0001
01-02 00:01:45.109   194   194 V audio_hw_primary: adev_open_output_stream:
enter: sample_rate(8000) channel_mask(0x1) devices(0x2000) flags(0x8001)
01-02 00:01:45.109   194   194 E voice_extn:
voice_extn_check_and_set_incall_music_usecase: Invalid session id 0
01-02 00:01:45.109   194   194 E audio_hw_primary: adev_open_output_stream:
Incall music delivery usecase cannot be set error:-22
01-02 00:01:45.109   194   194 D audio_hw_primary: adev_open_output_stream:
exit: ret -22
01-02 00:01:45.109   194   194 V AudioFlinger: openOutput()
openOutputStream returned output 0x0, SamplingRate 8000, Format 0x000001,
Channels 1, status -22
```

```
01-02 00:01:45.109   194   194 W AudioPolicyManagerBase:
checkOutputsForDevice() could not open output for device 2000
01-02 00:01:45.109   194   194 V AudioFlinger: openOutput(), module 5
Device 2000, SamplingRate 44100, Format 0x000001, Channels 3, flags 0
01-02 00:01:45.109   194   194 V AudioFlinger: openOutput(), offloadInfo
0xbee45518 version 0x0001
01-02 00:01:45.109   194   194 V AudioFlinger: openOutput()
openOutputStream returned output 0xb8565b18, SamplingRate 44100, Format
0x000001, Channels 3, status 0
01-02 00:01:45.109   194   194 I AudioFlinger: HAL output buffer size 1024
frames, normal mix buffer size 1024 frames
01-02 00:01:45.109   194   194 I AudioMixer: found effect "Multichannel
Downmix To Stereo" from The Android Open Source Project
```

AudioFlinger creates an output for this session.

```
01-02 00:01:45.109   194   194 V AudioFlinger: openOutput() created mixer
output: ID 15 thread 0xb52ae008
01-02 00:01:45.109   194  3204 I AudioFlinger: AudioFlinger's thread
0xb52ae008 ready to run
```

AudioFlinger sets the parameters for this stream, which USB HAL also sets.

```
01-02 00:01:45.109   194  3204 D usb_audio_hw: out_standby
01-02 00:01:45.109   194   194 I AudioFlinger: HAL output buffer size 1024
frames, normal mix buffer size 1024 frames
01-02 00:01:45.109   194   194 I AudioMixer: found effect "Multichannel
Downmix To Stereo" from The Android Open Source Project
01-02 00:01:45.109   194  3205 I AudioFlinger: AudioFlinger's thread
0xb516d008 ready to run
01-02 00:01:45.119   194   541 V AudioFlinger: setParameters(): io 15,
keyvalue card=1;device=0, calling pid 194
01-02 00:01:45.119   433   433 D charger_monitor: init target 500000
01-02 00:01:45.129   194  3204 D usb_audio_hw: out_set_parameters card [1]
device[0]
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 7
to output 15
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 0
to output 15
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 6
to output 15
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 2
to output 16
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 4
to output 16
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 5
to output 16
```

```
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 1
to output 15
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 3
to output 15
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 9
to output 15
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 10
to output 15
01-02 00:01:45.129   194   194 V AudioFlinger: setStreamOutput() stream 8
to output 15
01-02 00:01:45.129   194   541 V AudioFlinger: setParameters(): io 2,
keyvalue routing=0, calling pid 194
01-02 00:01:45.129   194   823 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: routing=0
01-02 00:01:45.129   194   823 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-02 00:01:45.129   194   823 V audio_hw_primary: out_set_parameters:
exit: code(1)
01-02 00:01:45.129   194   541 V AudioFlinger: setParameters(): io 3,
keyvalue routing=0, calling pid 194
01-02 00:01:45.129   194  1098 D audio_hw_primary: out_set_parameters:
enter: usecase(1: low-latency-playback) kvpairs: routing=0
01-02 00:01:45.129   194  1098 V audio_hw_primary: out_set_parameters:
exit: code(1)
01-02 00:01:45.129   194   541 V AudioFlinger: setParameters(): io 15,
keyvalue routing=0, calling pid 194
01-02 00:01:45.139   194  3204 D usb_audio_hw: out_set_parameters card [1]
device[0]
```

A ringtone is normally played when connecting a USB device.

A new AudioTrack is created for the ringtone.

```
01-02 00:01:53.679   194   194 V AwesomePlayer: AwesomePlayer running on
behalf of uid 10012
01-02 00:01:53.689  1142  1142 D Ringtone: Successfully created local
player
01-02 00:01:53.699   194  1099 V AwesomePlayer: track of type
'audio/vorbis' does not publish bitrate
01-02 00:01:53.709   194  1137 V AudioTrack: set() streamType 1 frameCount
4096 flags 0000
01-02 00:01:53.709   194  1137 V AudioTrack: createTrack_l() output 15
afLatency 92
01-02 00:01:53.709   194  1137 V AudioTrack: afFrameCount=1024,
minBufCount=4, afSampleRate=44100, afLatency=92
01-02 00:01:53.709   194  1137 V AudioTrack: minFrameCount: 4096,
afFrameCount=1024, minBufCount=4, sampleRate=44100, afSampleRate=44100,
afLatency=92
01-02 00:01:53.709   194  1137 V AudioFlinger: createTrack() sessionId: 18
```

```
01-02 00:01:53.709   194  1137 V AudioFlinger: createTrack() lSessionId: 18
01-02 00:01:53.709   194  1137 V AudioFlinger: acquiring 18 from 1142
01-02 00:01:53.709   194  1137 V AudioFlinger:  incremented refcount to 2
01-02 00:01:53.709   194   541 V AudioFlinger: setParameters(): io 15,
keyvalue routing=8192, calling pid 194
01-02 00:01:53.709   194  3204 D usb_audio_hw: out_set_parameters card [1]
device[0]
01-02 00:01:53.709   194  3245 V AudioTrack: obtainBuffer(2048) returned
4096 = 2048 + 2048 err 0
01-02 00:01:53.709   194  3245 V AudioPlayer: AudioCallback
01-02 00:01:53.709   194  3245 V AudioPlayer: buffer->size() = 0,
mPositionTimeMediaUs=-0.00 mPositionTimeRealUs=0.00
01-02 00:01:53.709   194  3245 V AudioPlayer: buffer->size() = 1152,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.00
01-02 00:01:53.709   194  3245 V AudioPlayer: buffer->size() = 1152,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.01
01-02 00:01:53.709   194  3245 V AudioPlayer: buffer->size() = 256,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.03
```

AudioTrack starts writing buffers to the USB HAL.

```
01-02 00:01:53.709   194  3245 V AudioTrack: obtainBuffer(2048) returned
2048 = 2048 + 0 err 0
01-02 00:01:53.719   194  3204 D usb_audio_hw: out_write() USB HAL writing
4096
```

USB playback starts when start_output_stream is called.

```
01-02 00:01:53.719   194  3204 D usb_audio_hw: start_output_stream()
01-02 00:01:53.719   194  3204 D usb_audio_hw: out_write() USB HAL writing
4096
```

USB playback continues until it is stopped.

```
01-02 00:01:53.719   194  3245 V AudioTrack: obtainBuffer(2048) returned
2048 = 2048 + 0 err 0
01-02 00:01:53.719   194  3245 V AudioPlayer: buffer->size() = 1152,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.10
01-02 00:01:53.719   194  3245 V AudioPlayer: buffer->size() = 256,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.12
01-02 00:01:53.719   194  3245 V AudioPlayer: buffer->size() = 256,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.12
01-02 00:01:53.719   194  3245 V AudioPlayer: buffer->size() = 256,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.12
01-02 00:01:53.719   194  3245 V AudioPlayer: buffer->size() = 256,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.12
```

```
01-02 00:01:53.719   194  3245 V AudioPlayer: buffer->size() = 256,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.13
01-02 00:01:53.719   194  3245 V AudioPlayer: buffer->size() = 256,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.13
01-02 00:01:53.719   194  3245 V AudioPlayer: buffer->size() = 256,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.13
01-02 00:01:53.719   194  3245 V AudioPlayer: buffer->size() = 256,
mPositionTimeMediaUs=0.55 mPositionTimeRealUs=0.14
01-02 00:01:53.719   194  3204 D usb_audio_hw: out_write() USB HAL writing
4096
01-02 00:01:53.719   194  3204 D usb_audio_hw: out_write() USB HAL writing
4096
01-02 00:01:53.719   194  3204 D usb_audio_hw: out_write() USB HAL writing
4096
01-02 00:01:53.719   194  3245 V AudioTrack: obtainBuffer(2048) returned
3072 = 2048 + 1024 err 0
```

When the USB device is unplugged, the session is closed and audio is routed to local playback for speakers.

```
01-02 00:01:54.629  1051  1051 V AudioService: Broadcast Receiver: Got
ACTION_USB_AUDIO_ACCESSORY_PLUG, state = 0, card: -1, device: -1
01-02 00:01:55.629   194   194 V AudioFlinger: setStreamOutput() stream 7
to output 2
01-02 00:01:55.629   194   194 V AudioFlinger: setStreamOutput() stream 0
to output 2
01-02 00:01:55.629   194   194 V AudioFlinger: setStreamOutput() stream 6
to output 2
01-02 00:01:55.629   194   194 V AudioFlinger: setStreamOutput() stream 2
to output 2
01-02 00:01:55.639   194   194 V AudioFlinger: setStreamOutput() stream 4
to output 2
01-02 00:01:55.639   194   194 V AudioFlinger: setStreamOutput() stream 5
to output 2
01-02 00:01:55.639   194   194 V AudioFlinger: setStreamOutput() stream 1
to output 2
```

Output is closed.

```
01-02 00:01:55.639   194   194 V AudioFlinger: closeOutput() 16
01-02 00:01:55.639   194   194 W AudioSystem: ioConfigChanged() closing
unknown output! 16
01-02 00:02:03.720   194  3204 W AudioFlinger: write blocked for 9992
msecs, 1 delayed writes, thread 0xb52ae008
01-02 00:02:03.749   194   194 D usb_audio_hw: out_set_parameters card [1]
device[0]
01-02 00:02:03.759   194  3204 D usb_audio_hw: out_standby
01-02 00:02:03.759   194   194 D usb_audio_hw: out_standby
```

```
01-02 00:02:03.769   194   823 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: routing=0
01-02 00:02:03.769   194   823 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-02 00:02:03.769   194   823 V audio_hw_primary: out_set_parameters:
exit: code(1)
```

9.1.3.2 Kernel logs

Not required.

9.1.4 Customization

No customizations recommended.

9.1.5 Debugging

Table 9-1 Debugging and troubleshooting tips

Issue type	Symptoms	Troubleshooting steps
Audio mute	No audio is heard from the output device	▪ Ensure that the USB framework has propagated the USB Connect event. ▪ Ensure that the USB device is capable of audio playback. ▪ Confirm that the sound card is created when the USB device is plugged in.

9.2 USB headset

QTI has modified the standard Android behavior to support applying effects in aDSP; the proxy port is used in this case to send decoded PCM data to aDSP.

In the case of USB headset, the AudioPlayer uses normal AudioTrack. The AudioPlayer registers with AudioFlinger for this requirement to get the notification on when a USB device capable of audio playback is connected. AudioPlayer also handles different cases, for example, pause, resume, and suspend/resume. It also takes care of starting playback from the correct position when a device switch happens to USB.

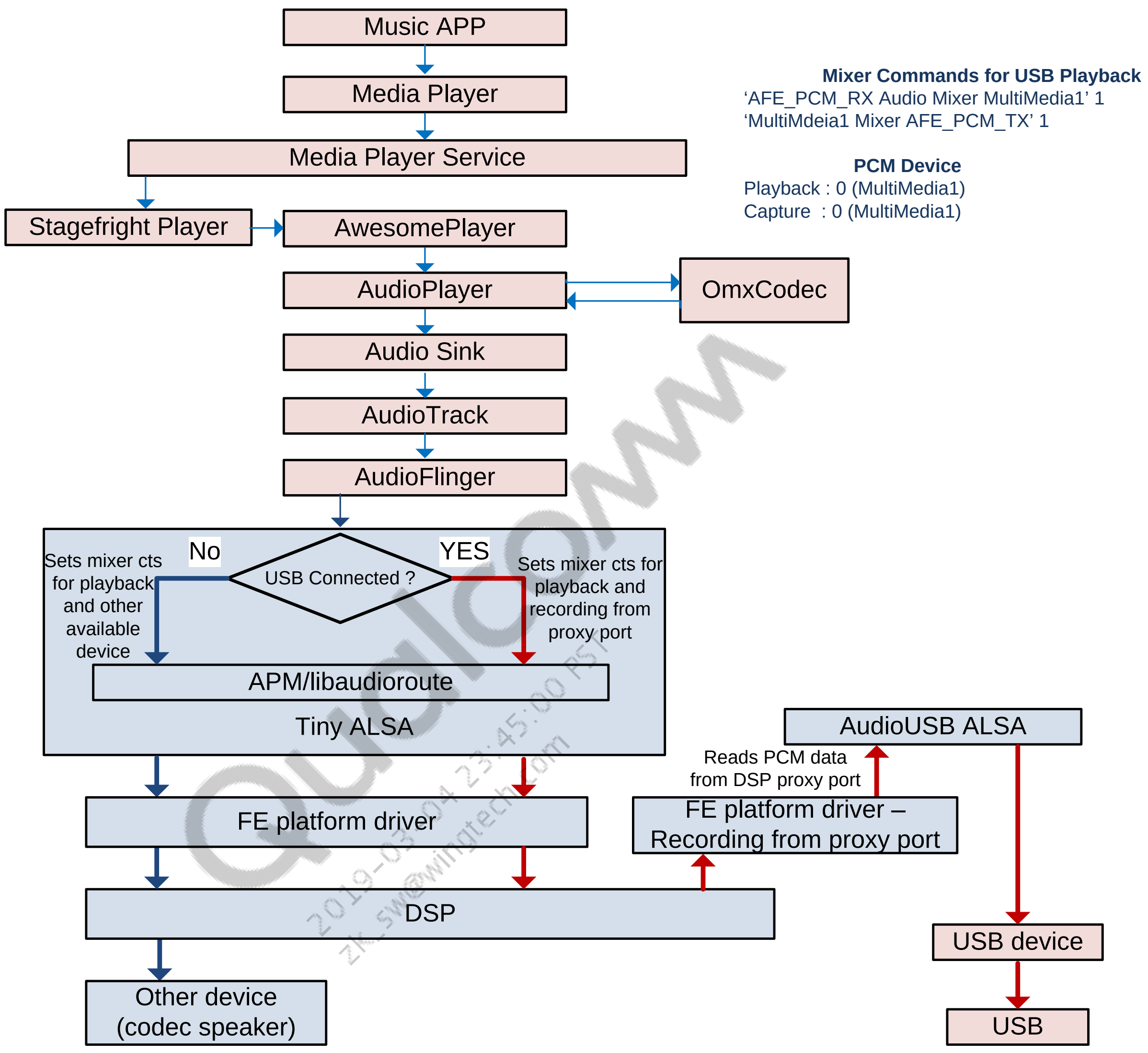

Music APP
Media Player
Media Player Service
Stagefright Player
AwesomePlayer
AudioPlayer
OmxCodec
Audio Sink
AudioTrack
AudioFlinger
Mixer Commands for USB Playback
'AFE_PCM_RX Audio Mixer MultiMedia1' 1
'MultiMdeia1 Mixer AFE_PCM_TX' 1
PCM Device
Playback : 0 (MultiMedia1)
Capture : 0 (MultiMedia1)
No
YES
USB Connected ?
Sets mixer cts for playback and other available device
Sets mixer cts for playback and recording from proxy port
APM/libaudioroute
Tiny ALSA
AudioUSB ALSA
Reads PCM data from DSP proxy port
FE platform driver
FE platform driver – Recording from proxy port
DSP
USB device
USB
Other device (codec speaker)

Figure 9-3 Audio playback over USB headset

Figure 9-3 shows the path taken by the framework as it sends the decoded PCM data to the USB Audio HAL. During playback, PCM data is read from the proxy port and written to the USB audio hardware interface, which in turn writes to the USB device.

The MediaPlayerService is a system service that links the Android user space to the media framework. Stagefright is the native Android media framework that facilitates media playback and capture through various physical and proxy devices. Other component functions are:

- MediaExtractor is responsible for retrieving track data and the corresponding metadata from the underlying file system or http stream.
- OMX layer is used for decoding/encoding data to and from raw PCM data into appropriate audio formats as supported by the system. Software or hardware instances of codecs are used to service a request depending on the capabilities of a platform.
- AudioPlayer is responsible for rendering audio and providing timebase for AV synchronization in AwesomePlayer, if an audio track is present.

- AwesomePlayer works as the engine to coordinate the modules and is finally connected into the Android media framework through the adapter that is the StagefrightPlayer.
- AudioFlinger resamples and routes the media stream to the actual devices that are involved with the output or input operation.

9.2.1 Call flow

This section is not applicable to this release.

9.2.2 Log collection

This section highlights the various files and/or components in which logging is enabled to troubleshoot issues related to audio playback on Android.

9.2.2.1 User space logs

This command clears the current logcat logs and starts logging the user space logs to the file logcat.txt and the stdout simultaneously.

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

NOTE: For the tee command to work, Cygwin is installed or the command is executed in a Linux environment.

9.2.2.2 Kernel logs

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo -n "file msm-pcm-afe-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file msm-pcm-routing-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file q6afe.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file q6adm.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file soc-pcm.c +p" > /sys/kernel/debug/dynamic_debug/control
```

NOTE: If the “mount: Device or resource busy” message printed when the mount -t debugfs debugfs /sys/kernel/debug command is executed, it is ignored.

9.2.2.3 Additional logging

Depending on the issue, additional logging is enabled in the user space and kernel space.

In the user space, enable logs on files listed at the following locations:

```
\harwdware\libhardware\modules\usbaudio
audio_hw.c

\hardware\qcom\audio\hal\audio_extn
usb.c

\frameworks\av\media\libmediaplayerservice
MediaPlayerService.cpp
StagefrightPlayer.cpp

\frameworks\av\media\libmedia
AudioSystem.cpp
AudioTrack.cpp
mediaplayer.cpp

\frameworks\av\media\libstagefright
AwesomePlayer.cpp
AudioPlayer.cpp
MP3Extractor.cpp
OMXClient.cpp
OMXCodec.cpp

\frameworks\av\services\audioflinger
AudioFlinger.cpp
AudioMixer.cpp
AudioPolicyService.cpp

\harwdware\qcom\audio\hal
audio_hw.c

\harwdware\qcom\audio\hal\msm8974
platform.c
hw_info.c

\harwdware\qcom\audio\hal\policy_hal
AudioPolicyManager.cpp

\harwdware\libhardware_legacy\audio
AudioPolicyManagerBase.cpp
```

9.2.3 Log analysis

9.2.3.1 User space logs

This section is not applicable to this release.

9.2.3.2 Kernel logs

This section is not applicable to this release.

9.2.4 Customization

No customizations recommended.

9.2.5 Debugging

Table 9-2 Debugging and troubleshooting tips

Issue type	Symptoms	Troubleshooting steps
Audio mute	No audio is heard from the output device	▪ Ensure that the USB framework has propagated the USB Connect event. ▪ Ensure that the USB device is capable of audio playback. ▪ Confirm that the sound card was created when the USB device was plugged in.

10 A2DP

10.1 A2DP playback

QTI has not modified the standard Android behavior; only the standard Google features are supported.

In the case of A2DP, AudioPlayer uses the normal AudioTrack. AudioPlayer registers with AudioFlinger for this requirement to get the notification on when A2DP is connected. AudioPlayer also handles different cases, for example, pause, resume, and suspend/resume. It also takes care of starting playback from the correct position when device switch happens to A2DP.

Figure 10-1 shows the path taken by the framework as it sends the decoded PCM data to the A2DP audio HAL.

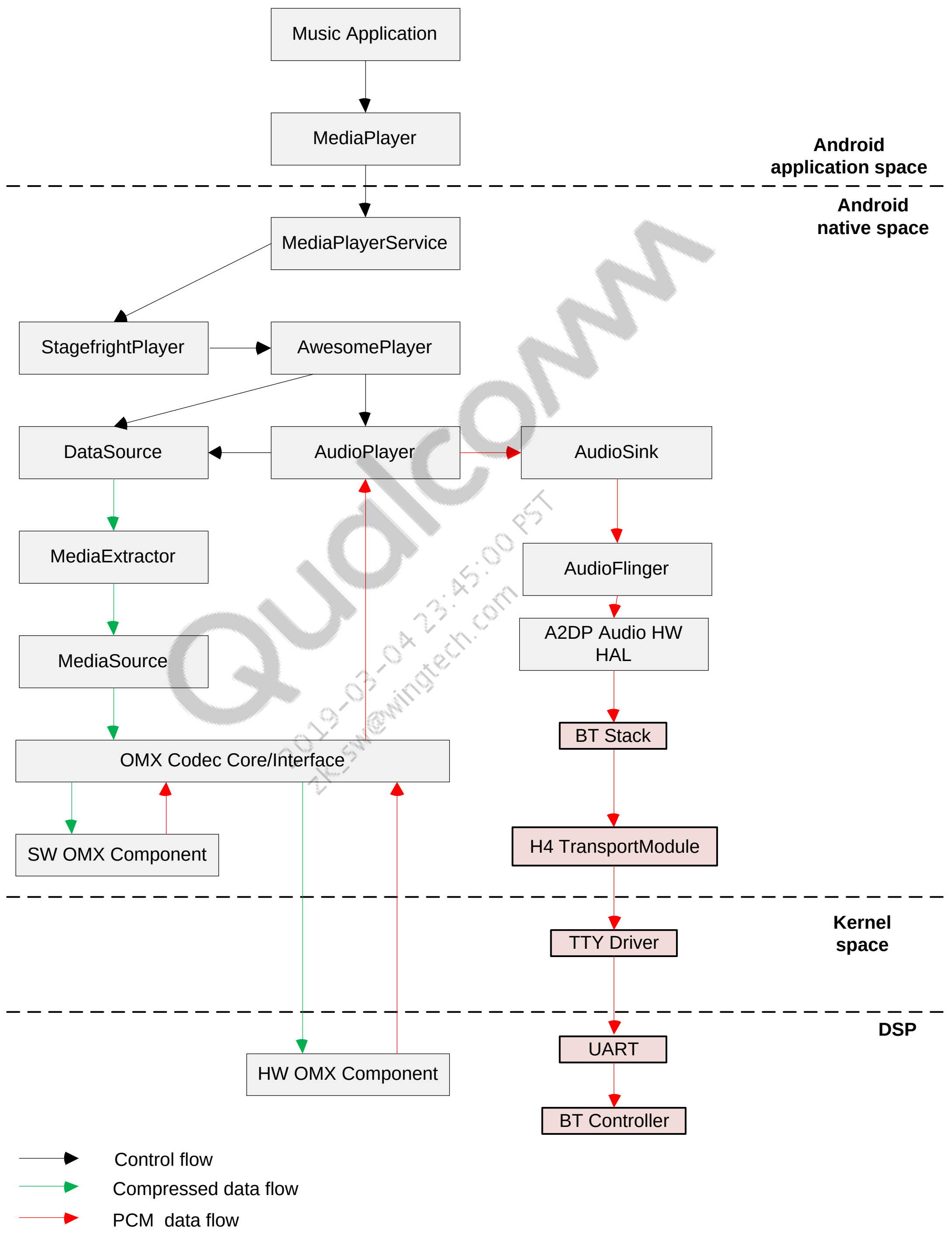


Figure 10-1 A2DP diagram

The MediaPlayerService is a system service that links the Android user space to the media framework. Stagefright is the native Android media framework that facilitates media playback and capture through various physical and proxy devices. Other component functions are:

- MediaExtractor is responsible for retrieving track data and the corresponding metadata from the underlying file system or http stream.
- OMX layer is used for decoding/encoding data to and from raw PCM data into appropriate audio formats as supported by the system. Software or hardware instances of codecs are used to service a request depending on a given capability of the platform.
- If an audio track is present, the AudioPlayer is responsible for rendering audio and providing timebase for AV synchronization in the AwesomePlayer.
- AwesomePlayer works as the engine to coordinate the modules and is finally connected into the Android media framework through the adapter that is the StagefrightPlayer.
- AudioFlinger resamples and routes the media stream to the actual devices that are involved with the output or input operation.

10.1.1 Call flow

10.1.1.1 A2DP playback

Figure 10-2 shows the call and data flows during active audio playback.

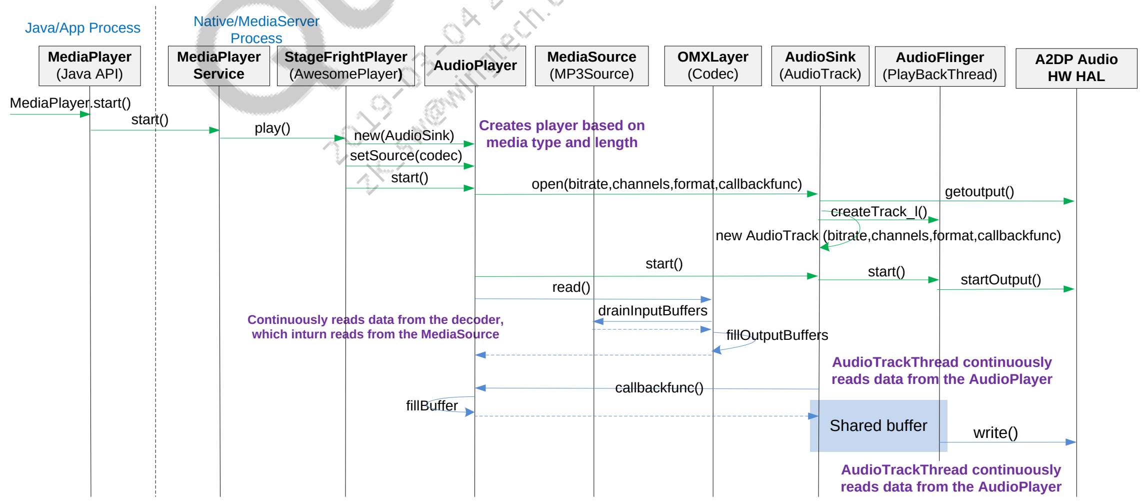

Java/App Process
Native/MediaServer Process
MediaPlayer (Java API)
MediaPlayer Service
StageFrightPlayer (AwesomePlayer)
AudioPlayer
MediaSource (MP3Source)
OMXLayer (Codec)
AudioSink (AudioTrack)
AudioFlinger (PlayBackThread)
A2DP Audio HW HAL
MediaPlayer.start()
start()
play()
new(AudioSink)
setSource(codec)
start()
Creates player based on media type and length
open(bitrate,channels,format,callbackfunc)
getoutput()
createTrack_l()
new AudioTrack (bitrate,channels,format,callbackfunc)
start()
start()
startOutput()
read()
drainInputBuffers
fillOutputBuffers
Continuously reads data from the decoder, which inturn reads from the MediaSource
AudioTrackThread continuously reads data from the AudioPlayer
callbackfunc()
fillBuffer
Shared buffer
write()
AudioTrackThread continuously reads data from the AudioPlayer

Figure 10-2 A2DP call flow

10.1.1.2 Device switch Bluetooth (offload to nonoffload)

Figure 10-3 illustrates device switch Bluetooth (offload→nonoffload).

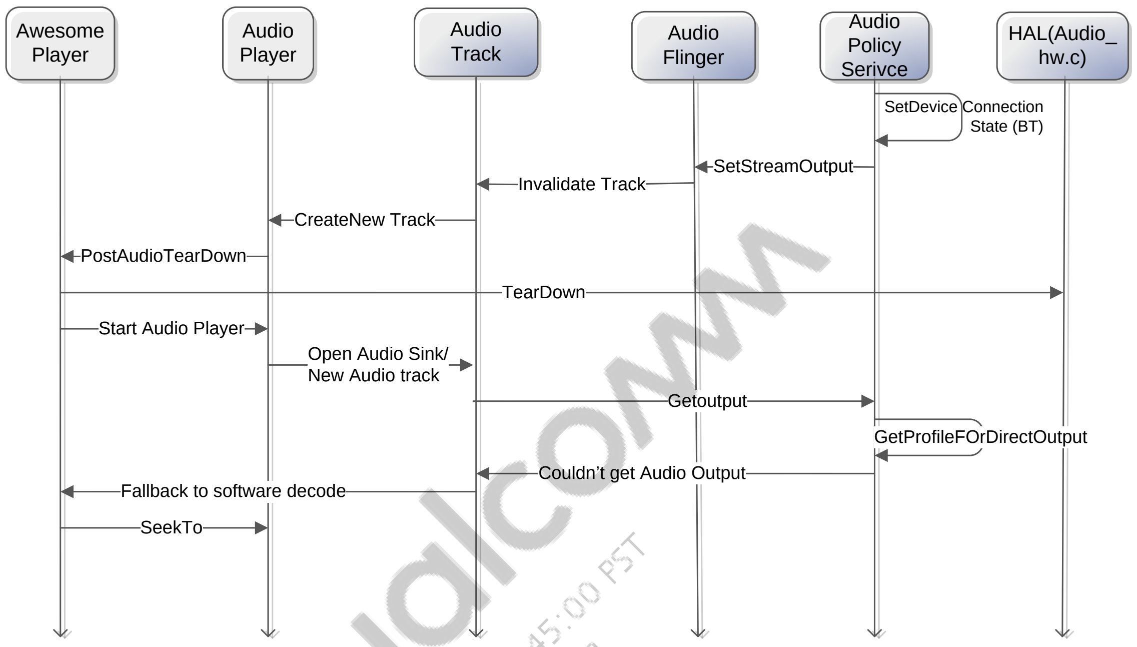


Figure 10-3 Device switch Bluetooth call flow

1. If a Bluetooth device is paired, the AudioPolicyService notifies AudioFlinger of this event by calling setStreamOutput(). The AudioFlinger then invalidates the current track and a new track is created to resubmit the unwritten data.
2. AudioPlayer then requests AwesomePlayer to get into postAudioTearDownEvent. A pause and a flush is issued into the entire pipeline as shown in Figure 10-3. Compress offload pause and resume. The flush is required to resend data already sent to DSP when changing audio session, as the source has changed.
3. AwesomePlayer then calls start() for the AudioPlayer, which then creates a new audio track.
4. AudioTrack then calls getOutput() which gets the profile information from the APM. The APM then calls openOutput(), which goes to AudioFlinger. However, since there is no valid profile for Bluetooth, it fails and propagates the error to AudioTrack. AudioTrack then returns the error to AwesomePlayer to indicate that starting the AudioPlayer was unsuccessful.
5. AwesomePlayer then decides to fallback to software decode and seeks to the last position before AudioTearDownEvent.

10.1.2 Log collection

This section highlights the various files and/or components in which logging is enabled to troubleshoot issues related to audio playback on Android.

10.1.2.1 User space logs

This command clears the current logcat logs and starts logging the user space logs to the file logcat.txt and the stdout simultaneously.

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

NOTE: For the “tee” command to work, Cygwin is installed or the command is executed in a Linux environment.

10.1.2.2 Kernel logs

Not required.

10.1.2.3 Additional logging

Depending on the issue, additional logging is enabled in the user space and kernel space. In the user space, enable logs in files listed at the following locations:

`external/bluetooth/bluedroid/audio_a2dp_hw/`
`audio_a2dp_hw.c`

`\frameworks\av\media\libmediaplayerservice`
`MediaPlayerService.cpp`
`StagefrightPlayer.cpp`

`\frameworks\av\media\libmedia`
`AudioSystem.cpp`
`AudioTrack.cpp`
`mediaplayer.cpp`

`\frameworks\av\media\libstagefright`
`AwesomePlayer.cpp`
`AudioPlayer.cpp`
`MP3Extractor.cpp`
`OMXClient.cpp`
`OMXCodec.cpp`

`\frameworks\av\services\audioflinger`
`AudioFlinger.cpp`
`AudioMixer.cpp`
`AudioPolicyService.cpp`

```
\harwdware\qcom\audio\hal
audio_hw.c

\harwdware\qcom\audio\hal\msm8974
platform.c
hw_info.c

\harwdware\qcom\audio\hal\policy_hal
AudioPolicyManager.cpp

\harwdware\libhardware_legacy\audio
AudioPolicyManagerBase.cpp
```

10.1.2.3.1 Enable Bluetooth logs

Enable external/bluetooth/bluedroid/audio_a2dp_hw/audio_a2dp_hw.c logs. Uncomment `//#define LOG_NDEBUG 0` in the external/bluetooth/bluedroid/audio_a2dp_hw/audio_a2dp_hw.c file.

10.1.2.3.2 Collect HCI dumps

Run the HCI dump command before pairing the device.

```
adb shell
hcidump -Rw /data/dump.log
```

10.1.3 Log analysis

10.1.3.1 User space logs

AudioFlinger loads the A2DP HAL during bootup.

```
01-02 00:00:26.079   211  1115 V AudioFlinger: thread 0xb59df008 type 0 TID
1115 waking up
01-02 00:00:26.079   211  1115 V AudioFlinger: acquireWakeLock_l()
AudioOut_3 status 0
01-02 00:00:26.089   211   211 I audio_a2dp_hw: adev_open:  adev_open in
A2dp_hw module
01-02 00:00:26.089   211   211 V audio_a2dp_hw: adev_open
01-02 00:00:26.089   211   211 V audio_a2dp_hw: adev_init_check
01-02 00:00:26.089   211   211 V audio_a2dp_hw: adev_set_master_volume
01-02 00:00:26.089   211   211 I AudioFlinger: loadHwModule() Loaded a2dp
audio interface from A2DP Audio HW HAL (audio) handle 4
01-02 00:00:26.089   211  1115 V AudioFlinger: acquireWakeLock_l()
AudioOut_3 status 0
```

A2DP profile is added when a Bluetooth device capable of audio playback is connected.

```
01-02 00:00:47.039  1488  1488 D A2dpProfile: Bluetooth service connected
01-02 00:00:47.039  1728  1745 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
01-02 00:00:47.039  1488  1488 D BluetoothHeadset: Proxy object connected
01-02 00:00:47.039  1488  1488 D HeadsetProfile: Bluetooth service
connected
01-02 00:00:47.039  1728  1808 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
01-02 00:00:47.399  1728  1728 D BluetoothAdapterService(1100012016):
Message: 50
01-02 00:00:47.399  1728  1728 D BluetoothAdapterService(1100012016):
MESSAGE_AUTO_CONNECT_PROFILES
01-02 00:00:47.399  1728  1728 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
01-02 00:00:47.399  1728  1728 D BluetoothAdapterService(1100012016): Quiet
mode Enabled = false
01-02 00:00:47.399  1728  1728 D BluetoothAdapterService(1100012016):
Initiate auto connection on BT on...
01-02 00:00:47.399  1728  1728 D BluetoothAdapterService(1100012016): Get
Bonded Devices being called
01-02 00:00:47.399  1728  1728 D BluetoothAdapterProperties:
getBondedDevices: length=1
```

This log message indicates the Bluetooth headset device name to which the device is paired.

```
01-02 00:00:47.409  1728  1728 D BluetoothAdapterService: Auto Connecting
Headset Profile with device 00:0D:FD:5C:9E:13
01-02 00:00:47.409  1728  1810 D BluetoothAdapterService:
getAdapterService(): returning
com.android.bluetooth.btservice.AdapterService@4190d9f0
01-02 00:00:47.409  1728  1728 D BluetoothAdapterService(1100012016): Get
Bonded Devices being called
01-02 00:00:47.409  1728  1728 D BluetoothAdapterProperties:
getBondedDevices: length=1
01-02 00:00:47.409  1728  1728 D BluetoothAdapterService: Auto Connecting
A2DP Profile with device 00:0D:FD:5C:9E:13
01-02 00:00:47.409  1728  1810 I BluetoothHeadsetServiceJni:
connectHfpNative: sBluetoothHfpInterface: 0x5f06e050
01-02 00:00:47.409  1309  1309 D BluetoothPhoneService: handleMessage: 4
01-02 00:00:47.409  1309  1309 V BluetoothPhoneService: Call state
Converted2: IDLE/IDLE -> 6
```

```
01-02 00:00:47.409  1728  1745 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
01-02 00:00:47.409  1728  1728 D BluetoothAdapterService(1100012016):
Message: 20
01-02 00:00:47.409  1728  1728 D BluetoothAdapterService(1100012016):
MESSAGE_PROFILE_CONNECTION_STATE_CHANGED
01-02 00:00:47.409  1728  1820 D A2dpStateMachine: Disconnected process
message: 1
01-02 00:00:47.409  1728  1820 E BluetoothA2dpServiceJni: Failed HF
disconnection, status: 1
01-02 00:00:47.409  1728  1728 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
01-02 00:00:47.409  1309  1309 D BluetoothManager: mReceiver:
HEADSET_STATE_CHANGED_ACTION
01-02 00:00:47.409  1309  1309 D BluetoothManager: ==> new state: 1
01-02 00:00:47.409  1309  1309 D AudioRouter: onBluetoothIndicationChange
false
01-02 00:00:47.409  1728  1820 D A2dpStateMachine: Peer Device is SNK
01-02 00:00:47.409  1728  1728 D BluetoothAdapterProperties:
CONNECTION_STATE_CHANGE: 00:0D:FD:5C:9E:13: 0 -> 1
01-02 00:00:47.409  1309  1309 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-02 00:00:47.409  1309  1309 D BluetoothManager:
isBluetoothAvailable()...
01-02 00:00:47.409  1728  1810 E HeadsetStateMachine:
terminateScoUsingVirtualVoiceCall:No present call to terminate
01-02 00:00:47.409  1728  1810 W bt-btif : BTHF: phone_state_change: SLC
connection not up. state=BTHF_CONNECTION_STATE_CONNECTING
01-02 00:00:47.409  1728  1810 E BluetoothHeadsetServiceJni: Failed report
phone state change, status: 2
01-02 00:00:47.409  1488  1488 D LocalBluetoothProfileManager: HEADSET
state change 0 -> 1
01-02 00:00:47.409  1728  1745 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
01-02 00:00:47.409  1488  1488 D CachedBluetoothDevice:
onProfileStateChanged: profile HEADSET newProfileState 1
01-02 00:00:47.409  1728  1744 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
01-02 00:00:47.419  1309  1309 D BluetoothManager:   ==> false
01-02 00:00:47.419  1309  1309 D AudioRouter: AudioMode: EARPIECE
01-02 00:00:47.419  1309  1309 D AudioRouter: Supported AudioMode:
EARPIECE, SPEAKER
01-02 00:00:47.419  1728  1820 I BluetoothA2dpServiceJni:
connectA2dpNative: sBluetoothA2dpInterface: 0x5f06eb98
01-02 00:00:47.419  1728  1820 D A2dpStateMachine: Exit Disconnected: 1
```

```
01-02 00:00:47.419  1728  1745 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
01-02 00:00:47.419  1728  1820 D A2dpStateMachine: Enter Pending: 1
01-02 00:00:47.419  1309  1309 D CallHandlerServiceProxy:
CallHandlerService not conneccted. Skipping onAudioModeChange().
01-02 00:00:47.419  1309  1309 D CallHandlerServiceProxy:
CallHandlerService not conneccted. SkippingonSupportedAudioModeChange().
01-02 00:00:47.419  1728  1808 D BluetoothAdapterService(1100012016): Get
Bonded Devices being called
01-02 00:00:47.419  1728  1808 D BluetoothAdapterProperties:
getBondedDevices: length=1
01-02 00:00:47.419  1728  1728 D A2dpStateMachine: Connection state
00:0D:FD:5C:9E:13: 0->1
01-02 00:00:47.419  1728  1728 D BluetoothAdapterService:
getAdapterService(): returning
com.android.bluetooth.btservice.AdapterService@4190d9f0
01-02 00:00:47.419  1728  1728 D BluetoothAdapterService(1100012016):
Message: 20
01-02 00:00:47.419  1728  1728 D BluetoothAdapterService(1100012016):
MESSAGE_PROFILE_CONNECTION_STATE_CHANGED
01-02 00:00:47.419  1728  1728 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
01-02 00:00:47.419  1728  1728 V HeadsetService: HeadsetService -  Received
BluetoothA2dp Conn State changed
01-02 00:00:47.419  1488  1488 D LocalBluetoothProfileManager: A2DP state
change 0 -> 1
01-02 00:00:47.419  1488  1488 D CachedBluetoothDevice:
onProfileStateChanged: profile A2DP newProfileState 1
01-02 00:00:47.419  1728  1745 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
```

When the device is paired, AudioPolicyManager registers the event and sets the state of the device as connected.

```
01-02 00:01:15.299  1728  1804 I BluetoothHeadsetServiceJni:
connection_state_callback
01-02 00:01:15.309  1728  1810 W HeadsetStateMachine: HFP Connected from
Disconnected state
01-02 00:01:15.309  1728  1810 D BluetoothAdapterService:
getAdapterService(): returning
com.android.bluetooth.btservice.AdapterService@4190d9f0
01-02 00:01:15.309  1728  1810 D BluetoothAdapterService(1100012016): Quiet
mode Enabled = false
01-02 00:01:15.309  1728  1810 I HeadsetStateMachine: Incoming Hf accepted
```

```
01-02 00:01:15.309  1728  1810 D BluetoothAdapterService:
getAdapterService(): returning
com.android.bluetooth.btservice.AdapterService@4190d9f0
01-02 00:01:15.309  1728  1728 D BluetoothAdapterService(1100012016):
Message: 20
01-02 00:01:15.309  1728  1728 D BluetoothAdapterService(1100012016):
MESSAGE_PROFILE_CONNECTION_STATE_CHANGED
01-02 00:01:15.309  1728  1728 D BluetoothAdapterService(1100012016):
Profile connected. Schedule missing profile connection if any
01-02 00:01:15.309  1728  1728 D BluetoothAdapterService(1100012016): Quiet
mode Enabled = false
01-02 00:01:15.319  1728  1728 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
01-02 00:01:15.329  1309  1309 D BluetoothManager: mReceiver:
HEADSET_STATE_CHANGED_ACTION
01-02 00:01:15.329  1309  1309 D BluetoothManager: ==> new state: 2
01-02 00:01:15.329  1309  1309 D AudioRouter: onBluetoothIndicationChange
false
01-02 00:01:15.329  1728  1728 D BluetoothAdapterProperties:
CONNECTION_STATE_CHANGE: 00:0D:FD:5C:9E:13: 0 -> 2
01-02 00:01:15.329   211  1116 V AudioPolicyService:
setDeviceConnectionState()
01-02 00:01:15.329   211  1116 V AudioPolicyManager:
setDeviceConnectionState() device: 20, state 1, address 00:0D:FD:5C:9E:13
01-02 00:01:15.329   211  1116 V AudioPolicyManager:
setDeviceConnectionState() connecting device 20
```

When the device is paired, AudioFlinger opens an output.

```
01-02 00:01:15.329   211  1116 V AudioFlinger: openOutput(), module 1
Device 20, SamplingRate 8000, Format 0x1000000, Channels 1, flags 31
01-02 00:01:15.329   211  1116 V AudioFlinger: openOutput(), offloadInfo
0xb57d2a58 version 0x0001
01-02 00:01:15.329   211  1116 V audio_hw_primary: adev_open_output_stream:
enter: sample_rate(8000) channel_mask(0x1) devices(0x20) flags(0x31)
01-02 00:01:15.329   211  1116 V audio_hw_primary: adev_open_output_stream:
offloaded output offload_info version 0001 bit rate 0
01-02 00:01:15.329   211  1116 V audio_hw_primary: adev_open_output_stream:
exit
01-02 00:01:15.329   211  1116 V AudioFlinger: openOutput()
openOutputStream returned output 0xb71a9690, SamplingRate 8000, Format
0x1000000, Channels 1, status 0
01-02 00:01:15.329   211  1116 V audio_hw_primary: out_set_callback
01-02 00:01:15.329   211  1116 I AudioFlinger: HAL output buffer size 32768
frames, normal mix buffer size 32768 frames
01-02 00:01:15.329   211  1116 V AudioFlinger: openOutput() created offload
output: ID 11 thread 0xb71a44f0
```

```
01-02 00:01:15.329   211  1116 V AudioFlinger: 
PlaybackThread::audioConfigChanged_l, thread 0xb71a44f0, event 0, param 0
01-02 00:01:15.329   211  1116 V AudioSystem: ioConfigChanged() event 0
01-02 00:01:15.329   211  1116 V AudioSystem: ioConfigChanged() new output 
samplingRate 8000, format 16777216 channel mask 0x1 frameCount 32768 
latency 96
01-02 00:01:15.329   211  1116 V AudioFlinger: openOutput(), module 1 
Device 20, SamplingRate 8000, Format 0x000001, Channels 1, flags 4001
01-02 00:01:15.329  2048  2061 V AudioSystem: ioConfigChanged() event 0
01-02 00:01:15.329  2048  2061 V AudioSystem: ioConfigChanged() new output 
samplingRate 8000, format 16777216 channel mask 0x1 frameCount 32768 
latency 96
01-02 00:01:15.329   211  1116 V AudioFlinger: openOutput(), offloadInfo 
0xb57d2a58 version 0x0001
01-02 00:01:15.329   211  1116 V audio_hw_primary: adev_open_output_stream: 
enter: sample_rate(8000) channel_mask(0x1) devices(0x20) flags(0x4001)
01-02 00:01:15.329   211  1116 V audio_hw_primary: adev_open_output_stream: 
exit
01-02 00:01:15.329   211  1116 V AudioFlinger: openOutput() 
openOutputStream returned output 0xb7198280, SamplingRate 48000, Format 
0x000001, Channels 3, status 0
01-02 00:01:15.329   211  1116 I AudioFlinger: HAL output buffer size 960 
frames, normal mix buffer size 960 frames
01-02 00:01:15.329   211  1116 V AudioFlinger: openOutput() created direct 
output: ID 12 thread 0xb71a5660
01-02 00:01:15.339   211  1116 V AudioFlinger: 
PlaybackThread::audioConfigChanged_l, thread 0xb71a5660, event 0, param 0
01-02 00:01:15.339   211  1116 V AudioSystem: ioConfigChanged() event 0
01-02 00:01:15.339   211  1116 V AudioSystem: ioConfigChanged() new output 
samplingRate 48000, format 1 channel mask 0x3 frameCount 960 latency 160
01-02 00:01:15.339   211  1116 V AudioPolicyManager: 
setDeviceConnectionState() checkOutputsForDevice() returned 4 outputs
```

APM tries to offload the stream and use speakers as output device, then goes into teardown and uses the A2DP HAL instead.

```
01-02 00:01:15.339  1309  1309 D AudioRouter: calculateModeFromCurrentState 
EARPIECE
01-02 00:01:15.339   211  1116 V AudioPolicyManager: getDeviceForStrategy() 
from cache strategy 5, device 2
01-02 00:01:15.339  1309  1309 D BluetoothManager: 
isBluetoothAvailable()...
01-02 00:01:15.339   211  1116 V AudioPolicyManager: getDeviceForStrategy() 
strategy 5, device 2
01-02 00:01:15.339   211  1116 V AudioFlinger: setStreamOutput() stream 7 
to output 2
01-02 00:01:15.339   211  1116 V AudioFlinger: 
MixerThread::invalidateTracks() mixer 0xb5b4a008, streamType 7, 
mTracks.size 1
```

```
01-02 00:01:15.339   211  1116 V AudioFlinger:
MixerThread::invalidateTracks() mixer 0xb59df008, streamType 7,
mTracks.size 0
…
01-02 00:01:15.349   211  2534 V audio_hw_primary: offload_thread_loop
01-02 00:01:15.349   211  2534 V audio_hw_primary: offload_thread_loop
offload_cmd_list 1 out->offload_state 0
01-02 00:01:15.349   211  2534 V audio_hw_primary: offload_thread_loop
SLEEPING
01-02 00:01:15.349   211   670 V AudioPolicyService: AudioCommandThread()
processing set parameters string closing=true, io 11
01-02 00:01:15.349   211   670 V AudioFlinger: setParameters(): io 11,
keyvalue closing=true, calling pid 211
01-02 00:01:15.349   211   670 V AudioFlinger: ThreadBase::setParameters()
closing=true
01-02 00:01:15.349   211  2537 V AudioFlinger: Audio hardware entering
standby, mixer 0xb71a5660, suspend count 0
01-02 00:01:15.349   211  2537 V audio_hw_primary: out_standby: enter:
usecase(0: deep-buffer-playback)
01-02 00:01:15.349   211  2537 V audio_hw_primary: out_standby: exit
01-02 00:01:15.349   211  1116 V AudioFlinger: closeOutput() 11
01-02 00:01:15.349   211  1116 V AudioFlinger: ThreadBase::exit
01-02 00:01:15.349   211  1116 V AudioFlinger:   preExit()
```

As part of the teardown sequence, APM drains buffers and closes the stream.

```
01-02 00:01:15.349   211  2536 V AudioFlinger: draining full
01-02 00:01:15.349   211  2536 V audio_hw_primary: out_drain
01-02 00:01:15.349   211  2536 V audio_hw_primary: send_offload_cmd_l 1
01-02 00:01:15.349   211  2536 V AudioFlinger: AsyncCallbackThread::exit
01-02 00:01:15.349   211  1116 V audio_hw_primary:
adev_close_output_stream: enter
01-02 00:01:15.349   211  1116 V audio_hw_primary: out_standby: enter:
usecase(3: compress-offload-playback)
01-02 00:01:15.349   211  2534 V audio_hw_primary: offload_thread_loop
RUNNING
01-02 00:01:15.349   211  2534 V audio_hw_primary: offload_thread_loop
offload_cmd_list 0 out->offload_state 0
01-02 00:01:15.349   211  2534 V audio_hw_primary: offload_thread_loop
STATE 0 CMD 1 out->compr 0x0
01-02 00:01:15.349   211  2534 E audio_hw_primary: offload_thread_loop:
Compress handle is NULL
01-02 00:01:15.349   211  2534 V audio_hw_primary: offload_thread_loop
offload_cmd_list 1 out->offload_state 0
01-02 00:01:15.359   211  2534 V audio_hw_primary: offload_thread_loop
SLEEPING
01-02 00:01:15.359   211  1116 V audio_hw_primary: out_standby: exit
01-02 00:01:15.359   211  1116 V audio_hw_primary: send_offload_cmd_l 0
```

```
01-02 00:01:15.359   211  2534 V audio_hw_primary: offload_thread_loop
RUNNING
01-02 00:01:15.359   211  2534 V audio_hw_primary: offload_thread_loop
offload_cmd_list 0 out->offload_state 0
01-02 00:01:15.359   211  2534 V audio_hw_primary: offload_thread_loop
STATE 0 CMD 0 out->compr 0x0
01-02 00:01:15.359   211  1116 V audio_hw_primary:
adev_close_output_stream: exit
…
```

New parameters for A2DP are set.

```
01-02 00:01:15.369  1728  1808 V AudioSystem: ioConfigChanged() new output
samplingRate 48000, format 1 channel mask 0x3 frameCount 960 latency 50
01-02 00:01:15.369   211   211 V AudioFlinger: setParameters(): io 0,
keyvalue bt_headset_name=Motorola S305;bt_headset_nrec=on, calling pid 1728
01-02 00:01:15.369   211   211 V SRS_Proc: ParamSet string:
bt_headset_name=Motorola S305;bt_headset_nrec=on
01-02 00:01:15.369   211   211 D audio_hw_primary: adev_set_parameters:
enter: bt_headset_name=Motorola S305;bt_headset_nrec=on
01-02 00:01:15.369   211   211 V voice   : voice_set_parameters: enter:
bt_headset_name=Motorola S305;bt_headset_nrec=on
01-02 00:01:15.369   211   211 V voice_extn: voice_extn_set_parameters:
enter: bt_headset_name=Motorola S305;bt_headset_nrec=on
01-02 00:01:15.369   211   211 D voice_extn: voice_extn_set_parameters: Not
handled here
01-02 00:01:15.369   211   211 V voice_extn: voice_extn_set_parameters:
exit with code(-2)
01-02 00:01:15.369   211   211 V compress_voip:
voice_extn_compress_voip_set_parameters: enter: bt_headset_name=Motorola
S305;bt_headset_nrec=on
01-02 00:01:15.369   211   211 V compress_voip:
voice_extn_compress_voip_set_parameters: exit
01-02 00:01:15.369   211   211 V voice   : voice_set_parameters: exit with
code(-2)
01-02 00:01:15.369   211   211 V msm8974_platform: platform_set_parameters:
enter: bt_headset_name=Motorola S305;bt_headset_nrec=on
01-02 00:01:15.369   211   211 V msm8974_platform: platform_set_parameters:
exit with code(-2)
01-02 00:01:15.369   211   211 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-02 00:01:15.369   211   211 V audio_hw_fm: audio_extn_fm_set_parameters:
enter
01-02 00:01:15.369   211   211 V audio_hw_fm: audio_extn_fm_set_parameters:
exit
01-02 00:01:15.369   211   211 V listen_hal_loader:
audio_extn_listen_set_parameters: enter: bt_headset_name=Motorola
S305;bt_headset_nrec=on
```

```
01-02 00:01:15.369   211   211 V listen_hw: listen_hw_set_parameters:
Enter, kvpairs = bt_headset_name=Motorola S305;bt_headset_nrec=on
01-02 00:01:15.369   211   211 V listen_hw: handle_set_parameters: Enter
kvpairs=bt_headset_name=Motorola S305;bt_headset_nrec=on.
01-02 00:01:15.369   211   211 V listen_hw: handle_set_parameters: Exit
01-02 00:01:15.369   211   211 V listen_hw: listen_hw_set_parameters: Exit,
ret=0
01-02 00:01:15.369   211   211 V audio_hw_primary: adev_set_parameters:
exit with code(-2)
01-02 00:01:16.319  1728  1804 I BluetoothHeadsetServiceJni:
connection_state_callback
01-02 00:01:16.329  1728  1728 D HeadsetPhoneState: sendDeviceStateChanged.
mService=0 mSignal=0 mRoam=0 mBatteryCharge=0
01-02 00:01:16.329  1309  1309 D BluetoothPhoneService: handleMessage: 4
01-02 00:01:16.329  1309  1309 V BluetoothPhoneService: Call state
Converted2: IDLE/IDLE -> 6
…
```

AudioFlinger opens an output for the device AUDIO_DEVICE_OUT_BLUETOOTH_A2DP (0x80) defined in system/core/include/system/audio.h.

```
01-02 00:01:21.879  1728  1728 D A2dpStateMachine: Connection state
00:0D:FD:5C:9E:13: 1->2
01-02 00:01:21.879   211  1117 V AudioPolicyService:
setDeviceConnectionState()
01-02 00:01:21.879   211  1117 V AudioPolicyManager:
setDeviceConnectionState() device: 80, state 1, address 00:0D:FD:5C:9E:13
01-02 00:01:21.879   211  1117 V AudioPolicyManager:
setDeviceConnectionState() connecting device 80
01-02 00:01:21.879   211  1117 V AudioFlinger: openOutput(), module 4
Device 80, SamplingRate 44100, Format 0x000001, Channels 3, flags 0
01-02 00:01:21.879   211  1117 V AudioFlinger: openOutput(), offloadInfo
0xb56d2a58 version 0x0001
01-02 00:01:21.879  1728  1820 D BluetoothAdapterService:
getAdapterService(): returning
com.android.bluetooth.btservice.AdapterService@4190d9f0
```

Open a stream in the A2DP audio HAL.

```
01-02 00:01:21.879   211  1117 I audio_a2dp_hw: adev_open_output_stream:
opening output
01-02 00:01:21.879   211  1117 V audio_a2dp_hw: a2dp_stream_out_init
01-02 00:01:21.879   211  1117 V audio_a2dp_hw: out_get_format: format 0x1
01-02 00:01:21.879   211  1117 V audio_a2dp_hw: out_get_sample_rate: rate
48000
01-02 00:01:21.879   211  1117 V audio_a2dp_hw: out_get_channels: channels
0x3
```

```
01-02 00:01:21.879   211  1117 I audio_a2dp_hw: skt_connect: connect to
/data/misc/bluedroid/.a2dp_ctrl (sz 10240)
01-02 00:01:21.879   211  1117 I audio_a2dp_hw: skt_connect: connected to
stack fd = 25
01-02 00:01:21.879   211  1117 I audio_a2dp_hw: check_a2dp_ready: state 3
01-02 00:01:21.879   211  1117 V audio_a2dp_hw: a2dp_command: A2DP COMMAND
A2DP_CTRL_CMD_CHECK_READY
01-02 00:01:21.879  1728  1728 D BluetoothAdapterService:
getAdapterService(): returning
com.android.bluetooth.btservice.AdapterService@4190d9f0
01-02 00:01:21.879  1728  1728 V HeadsetService: HeadsetService -  Received
BluetoothA2dp Conn State changed
01-02 00:01:21.879  1728  1728 D BluetoothAdapterService(1100012016):
Message: 20
01-02 00:01:21.879  1728  1728 D BluetoothAdapterService(1100012016):
MESSAGE_PROFILE_CONNECTION_STATE_CHANGED
01-02 00:01:21.879  1728  1728 D BluetoothAdapterService(1100012016):
Profile connected. Schedule missing profile connection if any
01-02 00:01:21.879  1728  1728 D BluetoothAdapterService(1100012016): Quiet
mode Enabled = false
01-02 00:01:21.879   211  1117 V audio_a2dp_hw: a2dp_command: A2DP COMMAND
A2DP_CTRL_CMD_CHECK_READY DONE STATUS 0
01-02 00:01:21.879   211  1117 V audio_a2dp_hw: adev_open_output_stream:
success
01-02 00:01:21.879   211  1117 V AudioFlinger: openOutput()
openOutputStream returned output 0xb719b150, SamplingRate 48000, Format
0x000001, Channels 3, status 0
01-02 00:01:21.889  1728  1728 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
```

The A2DP audio HAL gets the stream parameters.

```
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_sample_rate: rate
48000
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_channels: channels
0x3
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_format: format 0x1
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_format: format 0x1
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_channels: channels
0x3
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_buffer_size:
buffer_size : 10240
01-02 00:01:21.889   211  1117 I AudioFlinger: HAL output buffer size 2560
frames, normal mix buffer size 2560 frames
01-02 00:01:21.889   211  1117 V AudioFlinger: MixerThread() id=13
device=0x80 type=0
```

```
01-02 00:01:21.889   211  1117 V AudioFlinger: mSampleRate=48000,
mChannelMask=0x3, mChannelCount=2, mFormat=1, mFrameSize=4,
mFrameCount=2560, mNormalFrameCount=2560
01-02 00:01:21.889   211  1117 V AudioMixer: EffectQueryNumberEffects()
numEffects=12
01-02 00:01:21.889   211  1117 V AudioMixer: effect 0 is called Noise
Suppression
01-02 00:01:21.889   211  1117 V AudioMixer: effect 1 is called Acoustic
Echo Canceler
01-02 00:01:21.889   211  1117 V AudioMixer: effect 2 is called Visualizer
01-02 00:01:21.889   211  1117 V AudioMixer: effect 3 is called
Multichannel Downmix To Stereo
01-02 00:01:21.889   211  1117 I AudioMixer: found effect "Multichannel
Downmix To Stereo" from The Android Open Source Project
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_buffer_size:
buffer_size : 10240
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_format: format 0x1
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_sample_rate: rate
48000
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_channels: channels
0x3
01-02 00:01:21.889   211  1117 V AudioFlinger: openOutput() created mixer
output: ID 13 thread 0xb4cb6008
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_latency
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_format: format 0x1
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_channels: channels
0x3
01-02 00:01:21.889   211  1117 V AudioFlinger:
PlaybackThread::audioConfigChanged_l, thread 0xb4cb6008, event 0, param 0
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_latency
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_format: format 0x1
01-02 00:01:21.889   211  1117 V audio_a2dp_hw: out_get_channels: channels
0x3
01-02 00:01:21.889   211  1117 V AudioSystem: ioConfigChanged() event 0
01-02 00:01:21.889   211  1117 V AudioSystem: ioConfigChanged() new output
samplingRate 48000, format 1 channel mask 0x3 frameCount 2560 latency 253
01-02 00:01:21.889   964  1322 V AudioSystem: ioConfigChanged() event 0
01-02 00:01:21.889   964  1322 V AudioSystem: ioConfigChanged() new output
samplingRate 48000, format 1 channel mask 0x3 frameCount 2560 latency 253
01-02 00:01:21.889  1309  1323 V AudioSystem: ioConfigChanged() event 0
01-02 00:01:21.889  1309  1323 V AudioSystem: ioConfigChanged() new output
samplingRate 48000, format 1 channel mask 0x3 frameCount 2560 latency 253
01-02 00:01:21.889  1539  1553 V AudioSystem: ioConfigChanged() event 0
01-02 00:01:21.889  1539  1553 V AudioSystem: ioConfigChanged() new output
samplingRate 48000, format 1 channel mask 0x3 frameCount 2560 latency 253
01-02 00:01:21.889  1728  1744 V AudioSystem: ioConfigChanged() event 0
01-02 00:01:21.889  1728  1744 V AudioSystem: ioConfigChanged() new output
samplingRate 48000, format 1 channel mask 0x3 frameCount 2560 latency 253
01-02 00:01:21.889  1748  1762 V AudioSystem: ioConfigChanged() event 0
```

```
01-02 00:01:21.889  1748  1762 V AudioSystem: ioConfigChanged() new output
samplingRate 48000, format 1 channel mask 0x3 frameCount 2560 latency 253
01-02 00:01:21.889  2048  2061 V AudioSystem: ioConfigChanged() event 0
01-02 00:01:21.889  2048  2061 V AudioSystem: ioConfigChanged() new output
samplingRate 48000, format 1 channel mask 0x3 frameCount 2560 latency 253
01-02 00:01:21.889   211  1117 V AudioPolicyManager: checkAndSetVolume:
index 4 output 13 device 80
01-02 00:01:21.889   211  1117 V AudioPolicyManager: checkAndSetVolume()
for output 13 stream 0, volume 0.387468, delay 0
01-02 00:01:21.889   211  1117 V AudioPolicyService: inserting command: 2
at index 0, num commands 0
01-02 00:01:21.889   211  1117 V AudioPolicyService: AudioCommandThread()
adding set volume stream 0, volume 0.387468, output 13
01-02 00:01:21.889   211   670 V AudioPolicyService: AudioCommandThread()
waking up
01-02 00:01:21.889   211   670 V AudioPolicyService: AudioCommandThread()
processing set volume stream 0,                             volume
0.387468, output 13
01-02 00:01:21.889   211   670 V AudioPolicyService: AudioCommandThread()
going to sleep
01-02 00:01:21.889   211  1117 V AudioPolicyManager: checkAndSetVolume:
index 5 output 13 device 80
01-02 00:01:21.889   211  1117 V AudioPolicyManager: checkAndSetVolume()
for output 13 stream 1, volume 0.042500, delay 0
01-02 00:01:21.889   211  1117 V AudioPolicyService: inserting command: 2
at index 0, num commands 0
01-02 00:01:21.889   211  1117 V AudioPolicyService: AudioCommandThread()
adding set volume stream 1, volume 0.042500, output 13
01-02 00:01:21.889  1728  1820 D BluetoothAdapterService(1100012016): Quiet
mode Enabled = false
01-02 00:01:21.889   211   670 V AudioPolicyService: AudioCommandThread()
waking up
01-02 00:01:21.889   211   670 V AudioPolicyService: AudioCommandThread()
processing set volume stream 1,                             volume
0.042500, output 13
01-02 00:01:21.889   211   670 V AudioPolicyService: AudioCommandThread()
going to sleep
01-02 00:01:21.889  1728  1820 I A2dpStateMachine: Ready to connect
incoming Connection from pending state
01-02 00:01:21.889  1728  1820 D A2dpStateMachine: Enter Connected: 101
01-02 00:01:21.889   211  1117 V AudioPolicyManager: checkAndSetVolume:
index 5 output 13 device 80
01-02 00:01:21.889   211  1117 V AudioPolicyManager: checkAndSetVolume()
for output 13 stream 2, volume 0.094183, delay 0
01-02 00:01:21.889   211  1117 V AudioPolicyService: inserting command: 2
at index 0, num commands 0
01-02 00:01:21.899  1728  1728 V HeadsetService: HeadsetService -  Received
BluetoothA2dp Play State changed
```

```
01-02 00:01:21.899  1728  1820 D A2dpStateMachine: A2DP Playing state :
device: 00:0D:FD:5C:9E:13 State:10->11
01-02 00:01:21.899   211  2671 I AudioFlinger: AudioFlinger's thread
0xb4cb6008 ready to run
…
```

Music playback is ready to start; a normal audio session starts here.

```
01-02 00:01:26.949  2048  2086 V MediaPlayer: reset
01-02 00:01:26.949  2048  2068 I LocalDevicePlayback: open: id=[1,
DEFAULT], pos=0, playPos=0, fromUser=true, track=Himalaya
01-02 00:01:26.949  2048  2673 V MediaPlayer: constructor
01-02 00:01:26.969   211  1116 V MediaPlayerService: Client(4) constructor
01-02 00:01:26.969   211   211 V StagefrightPlayer: StagefrightPlayer
01-02 00:01:26.969   211   211 V StagefrightPlayer: initCheck
01-02 00:01:26.969   211   211 V AwesomePlayer: AwesomePlayer running on
behalf of uid 10067
01-02 00:01:26.969   211   211 V AudioSink: AudioOutput(10)
01-02 00:01:26.979   211   211 V MP3Extractor: skipped ID3 tag, new
starting offset is 4096 (0x0000000000001000)
01-02 00:01:26.979   211   211 V MP3Extractor: found possible 1st frame at
5056 (header = 0xfffbe400)
01-02 00:01:26.979   211   211 V MP3Extractor: subsequent header is
fffbe400
01-02 00:01:26.989   211   211 V AwesomePlayer: mBitrate = 320000 bits/sec
01-02 00:01:26.989  2048  2673 V MediaPlayer: prepare
01-02 00:01:26.989   211  1116 V MediaPlayerService: [4]
setAudioStreamType(3)
01-02 00:01:26.989   211  1117 V MediaPlayerService: [4] prepareAsync
```

Playback starts here. APM checks for stream offloading and falls back to software decode.

```
01-02 00:01:26.989   211  2674 V AudioSystem: isOffloadSupported()
01-02 00:01:26.989   211  2674 V OMXCodec: matching
'OMX.google.mp3.decoder' quirks 0x00000000
01-02 00:01:26.989   211  2674 V OMXCodec: matching 'MP3Decoder' quirks
0x00000000
01-02 00:01:26.989   211  2674 V OMXCodec: Attempting to allocate OMX node
'OMX.google.mp3.decoder'
01-02 00:01:26.989   211  2674 V OMXCodec: Successfully allocated OMX node
'OMX.google.mp3.decoder'
01-02 00:01:26.989   211  2674 V OMXCodec: configureCodec protected=0
01-02 00:01:26.989   211  2674 V AwesomePlayer: createAudioPlayer: bypass
OMX (offload)
01-02 00:01:27.019  2048  2673 V MediaPlayer: start
01-02 00:01:27.019   211  1117 V AudioSink: setAuxEffectSendLevel(0.000000)
01-02 00:01:27.019   211   211 V MediaPlayerService: [4] start
```

```
01-02 00:01:27.019   211   211 V StagefrightPlayer: setLooping
01-02 00:01:27.019   211   211 V StagefrightPlayer: start
01-02 00:01:27.019   211   211 V AudioSink: creating new AudioTrack
01-02 00:01:27.019   211   211 V AudioTrack: sampleRate 48000, channelMask
0x3, format 16777216
01-02 00:01:27.019   211   211 V AudioTrack: streamType 3
01-02 00:01:27.019   211   211 V AudioTrack: set() streamType 3 frameCount
0 flags 0010
01-02 00:01:27.019   211   211 V AudioTrack: Offload request, forcing to
Direct Output
01-02 00:01:27.019   211   211 V AudioPolicyService: getOutput()
01-02 00:01:27.019   211   211 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 80
01-02 00:01:27.019   211   211 W AudioPolicyManagerBase: getOutput() could
not find output for stream 3, samplingRate 48000,format 16777216, channels
3, flags 11
01-02 00:01:27.019   211   211 E AudioTrack: Could not get audio output for
stream type 3
01-02 00:01:27.019   211   211 E AudioSink: Unable to create audio track
01-02 00:01:27.019   211   211 I AwesomePlayer: play_l() cannot create
offload output, fallback to sw decode
```

Resources are allocated for the software decoder.

```
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder]
allocating 4 buffers of size 8192 on input port
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb719a078 on input port
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb7192aa8 on input port
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb71a4b40 on input port
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb71a4bf8 on input port
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder]
allocating 4 buffers of size 9216 on output port
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb71a4d80 on output port
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb71a4f90 on output port
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb71a4010 on output port
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder]
allocated buffer 0xb71a4138 on output port
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder]
onStateChange 2
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder] Now
Idle.
```

Software decoders transition to idle, indicating it is ready to decode, then it transitions to executing to begin the buffer exchange.

```
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder]
onStateChange 3
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder] Now
Executing.
01-02 00:01:27.029   211   211 V MediaPlayerService: [4] notify
(0xb7194c58, 7, 0, 0)
01-02 00:01:27.029  2048  2085 V MediaPlayer: message received msg=7,
ext1=0, ext2=0
01-02 00:01:27.029   211   211 V AudioPlayer: seekTo( 67513000 )
01-02 00:01:27.029   211   211 V AudioSink: flush
01-02 00:01:27.029   211   211 V MP3Extractor: lost sync! header =
0x5cc08c05, old header = 0xfffbe400
01-02 00:01:27.029   211   211 V MP3Extractor: found possible 1st frame at
2706496 (header = 0xfffbe400)
01-02 00:01:27.029   211   211 V MP3Extractor: subsequent header is
fffbe400
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb719a078 (length 960), timestamp 67536000 us (67.54
secs)
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb7192aa8 (length 960), timestamp 67560000 us (67.56
secs)
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb71a4b40 (length 960), timestamp 67584000 us (67.58
secs)
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb71a4bf8 (length 960), timestamp 67608000 us (67.61
secs)
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder] Calling
fillBuffer on buffer 0xb71a4d80
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder] Calling
fillBuffer on buffer 0xb71a4f90
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder] Calling
fillBuffer on buffer 0xb71a4010
01-02 00:01:27.029   211   211 V OMXCodec: [OMX.google.mp3.decoder] Calling
fillBuffer on buffer 0xb71a4138
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder]
EMPTY_BUFFER_DONE(buffer: 0xb719a078)
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb719a078 (length 960), timestamp 67632000 us (67.63
secs)
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder]
FILL_BUFFER_DONE(buffer: 0xb71a4d80, size: 2492, flags: 0x00000000,
timestamp: 67536000 us (67.54 secs))
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder]
EMPTY_BUFFER_DONE(buffer: 0xb7192aa8)
```

```
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb7192aa8 (length 960), timestamp 67656000 us (67.66
secs)
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder]
FILL_BUFFER_DONE(buffer: 0xb71a4f90, size: 4608, flags: 0x00000000,
timestamp: 67560000 us (67.56 secs))
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder]
EMPTY_BUFFER_DONE(buffer: 0xb71a4b40)
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb71a4b40 (length 960), timestamp 67680000 us (67.68
secs)
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder]
FILL_BUFFER_DONE(buffer: 0xb71a4010, size: 4608, flags: 0x00000000,
timestamp: 67584000 us (67.58 secs))
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder]
EMPTY_BUFFER_DONE(buffer: 0xb71a4bf8)
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder] Calling
emptyBuffer on buffer 0xb71a4bf8 (length 960), timestamp 67704000 us (67.70
secs)
01-02 00:01:27.029   211  2676 V OMXCodec: [OMX.google.mp3.decoder]
FILL_BUFFER_DONE(buffer: 0xb71a4138, size: 4608, flags: 0x00000000,
timestamp: 67608000 us (67.61 secs))
```

A new track for this stream is created.

```
01-02 00:01:27.029   211   211 V AudioSink: open(48000, 2, 0x0, 0x1, 4, 10
0x8)
01-02 00:01:27.029   211   211 V AudioPolicyService: getOutput()
01-02 00:01:27.029   211   211 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 80
01-02 00:01:27.029   211   211 V AudioSystem: getFrameCount() streamType 3,
output 13, frameCount 2560
01-02 00:01:27.029   211   211 V AudioPolicyService: getOutput()
01-02 00:01:27.029   211   211 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 80
01-02 00:01:27.029   211   211 V AudioSystem: getOutputSamplingRate()
reading from output desc
01-02 00:01:27.029   211   211 V AudioSystem: getSamplingRate() streamType
3, output 13, sampling rate 48000
01-02 00:01:27.029   211   211 V AudioSink: no track available to recycle
01-02 00:01:27.029   211   211 V AudioSink: creating new AudioTrack
01-02 00:01:27.029   211   211 V AudioTrack: sampleRate 48000, channelMask
0x3, format 1
01-02 00:01:27.029   211   211 V AudioTrack: streamType 3
01-02 00:01:27.029   211   211 V AudioTrack: set() streamType 3 frameCount
10240 flags 0008
01-02 00:01:27.029   211   211 V AudioPolicyService: getOutput()
01-02 00:01:27.029   211   211 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 80
```

```
01-02 00:01:27.029   211   211 V AudioSystem: getLatency() streamType 3,
output 13, latency 253
01-02 00:01:27.029   211   211 V AudioSystem: getFrameCount() streamType 3,
output 13, frameCount 2560
01-02 00:01:27.029   211   211 V AudioSystem: getOutputSamplingRate()
reading from output desc
01-02 00:01:27.029   211   211 V AudioSystem: getSamplingRate() streamType
3, output 13, sampling rate 48000
01-02 00:01:27.029   211   211 V AudioTrack: createTrack_l() output 13
afLatency 253
01-02 00:01:27.029   211   211 V AudioTrack: afFrameCount=2560,
minBufCount=4, afSampleRate=48000, afLatency=253
01-02 00:01:27.029   211   211 V AudioTrack: minFrameCount: 10240,
afFrameCount=2560, minBufCount=4, sampleRate=48000, afSampleRate=48000,
afLatency=253
01-02 00:01:27.029   211   211 V AudioFlinger: createTrack() sessionId: 10
01-02 00:01:27.029   211   211 V AudioFlinger: createTrack() lSessionId: 10
01-02 00:01:27.029   211   211 V AudioMixer: add track (1)
01-02 00:01:27.029   211   211 V AudioMixer:
AudioMixer::unprepareTrackForDownmix(1)
01-02 00:01:27.029   211   211 V AudioMixer:  nothing to do, no downmixer
to delete
01-02 00:01:27.029   211   211 V AudioFlinger: Track constructor name 4097,
calling pid 2048
01-02 00:01:27.029   211   211 V AudioFlinger: acquiring 10 from 2048
01-02 00:01:27.029   211   211 V AudioFlinger:  incremented refcount to 2
01-02 00:01:27.029   211   211 V AudioSink: deleteRecycledTrack
01-02 00:01:27.029   211   211 V AudioSink: setVolume
01-02 00:01:27.029   211   211 V AudioPolicyService: getOutput()
01-02 00:01:27.029   211   211 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 80
01-02 00:01:27.029   211   211 V AudioSystem: getOutputSamplingRate()
reading from output desc
01-02 00:01:27.029   211   211 V AudioSystem: getSamplingRate() streamType
3, output 13, sampling rate 48000
01-02 00:01:27.029   211   211 V AudioSink: open() DONE status 0
01-02 00:01:27.029   211   211 V AudioSink: start
01-02 00:01:27.029   211   211 V AudioFlinger: start(4097), calling pid
2048 session 10
01-02 00:01:27.029   211   211 V AudioFlinger: ? => ACTIVE (4097) on thread
0xb71a4298
01-02 00:01:27.029   211   211 V AudioPolicyService: startOutput()
01-02 00:01:27.029   211   211 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 80
01-02 00:01:27.029   211   211 V AudioPolicyService: inserting command: 3
at index 0, num commands 0
01-02 00:01:27.029   211   211 V AudioPolicyService: AudioCommandThread()
adding set parameter string routing=128, io 13 ,delay 0
```

```
01-02 00:01:27.029   211   670 V AudioPolicyService: AudioCommandThread()
waking up
01-02 00:01:27.029   211   670 V AudioPolicyService: AudioCommandThread()
processing set parameters string routing=128, io 13
01-02 00:01:27.029   211   670 V AudioFlinger: setParameters(): io 13,
keyvalue routing=128, calling pid 211
01-02 00:01:27.029   211   670 V AudioFlinger: ThreadBase::setParameters()
routing=128
01-02 00:01:27.029   211  2671 V AudioFlinger: thread 0xb4cb6008 type 0 TID
2671 waking up
01-02 00:01:27.029   211  2671 V AudioFlinger: acquireWakeLock_l()
AudioOut_D status 0
01-02 00:01:27.029   211  2671 I audio_a2dp_hw: out_set_parameters: state 5
01-02 00:01:27.029   211  2671 I str_params: key: 'routing' value: '128'
01-02 00:01:27.029   211  2671 V AudioFlinger: acquireWakeLock_l()
AudioOut_D status 0
01-02 00:01:27.029   211   670 V AudioPolicyService: AudioCommandThread()
going to sleep
01-02 00:01:27.039   211   211 V AudioPolicyManager: checkAndSetVolume:
index 4 output 13 device 80
01-02 00:01:27.039   211   211 V AudioFlinger: signal playback thread
01-02 00:01:27.039   211  2684 V AudioTrack: obtainBuffer(5120) returned
10240 = 5120 + 5120 err 0
01-02 00:01:27.039   211  2684 V AudioPlayer: AudioCallback
01-02 00:01:27.059   211  2671 V audio_a2dp_hw: out_write: write 10240
bytes (fd -1)
01-02 00:01:27.059   211  2671 I audio_a2dp_hw: start_audio_datapath: state
5
01-02 00:01:27.059   211  2671 V audio_a2dp_hw: a2dp_command: A2DP COMMAND
A2DP_CTRL_CMD_START
01-02 00:01:27.059   211  2671 V audio_a2dp_hw: a2dp_command: A2DP COMMAND
A2DP_CTRL_CMD_START DONE STATUS 0
01-02 00:01:27.059   211  2671 I audio_a2dp_hw: skt_connect: connect to
/data/misc/bluedroid/.a2dp_data (sz 10240)
01-02 00:01:27.059   211  2671 I audio_a2dp_hw: skt_connect: connected to
stack fd = 33
01-02 00:01:27.059   211  2671 V audio_a2dp_hw: skt_write
01-02 00:01:27.059   211  2671 V audio_a2dp_hw: ts_log: [skt_write] ts
102585398, diff 102585398, val 10240
01-02 00:01:27.059   211  2671 V audio_a2dp_hw: out_write: wrote 10240
bytes out of 10240 bytes
01-02 00:01:27.059   211  2671 V AudioTrackShared: mAvailToClient=2560
stepCount=2560 minimum=5120
01-02 00:01:27.059   211  2671 V audio_a2dp_hw: out_write: write 10240
bytes (fd 33)
01-02 00:01:27.059   211  2671 V audio_a2dp_hw: skt_write
```

Buffer exchange continues until stopped.

```
01-02 00:01:30.029  1539  1539 V MediaPlayer: getCurrentPosition
01-02 00:01:30.029   211   211 V MediaPlayerService: getCurrentPosition
01-02 00:01:30.029   211   211 V StagefrightPlayer: getCurrentPosition
01-02 00:01:30.029   211   211 V MediaPlayerService: [1] getCurrentPosition
= 21390
01-02 00:01:30.039  1539  1539 V MediaPlayer: reset
01-02 00:01:30.039   211  1116 V MediaPlayerService: [1] reset
01-02 00:01:30.039   211  1116 V StagefrightPlayer: reset
01-02 00:01:30.039   211  1116 V MediaPlayerService: [1] notify
(0xb7198b10, 8, 0, 0)
01-02 00:01:30.039  1539  1554 V MediaPlayer: message received msg=8,
ext1=0, ext2=0
01-02 00:01:30.039  1539  1554 V MediaPlayer: notify(8, 0, 0) callback on
disconnected mediaplayer
01-02 00:01:30.039  1539  1539 V MediaPlayer: setListener
01-02 00:01:30.039  1539  1539 V MediaPlayer: disconnect
01-02 00:01:30.049   211  1117 V MediaPlayerService: Client(1) destructor
pid = 1539
01-02 00:01:30.049   964  1331 I MediaFocusControl:  AudioFocus
abandonAudioFocus() from
android.media.AudioManager@418ff640com.android.music.MediaPlaybackService$3
@418fe4c0
01-02 00:01:30.049   211  1117 V MediaPlayerService: disconnect(1) from pid
1539
01-02 00:01:30.049   211  1117 V StagefrightPlayer: reset
01-02 00:01:30.049   211  1117 V StagefrightPlayer: ~StagefrightPlayer
01-02 00:01:30.049  1539  1553 V MediaPlayer: destructor
01-02 00:01:30.049   211  1117 V StagefrightPlayer: reset
01-02 00:01:39.059  1728  1954 W bt-avp  : Dropped media packet
01-02 00:01:39.069   964  1126 I EventHub: Removing device AVRCP due to
epoll hang-up event.
01-02 00:01:39.069   964  1126 I EventHub: Removed device:
path=/dev/input/event5 name=AVRCP id=6 fd=223 classes=0x80000001
01-02 00:01:39.079  1728  1804 I BluetoothHeadsetServiceJni:
connection_state_callback
01-02 00:01:39.079  1728  1810 E HeadsetStateMachine:
terminateScoUsingVirtualVoiceCall:No present call to terminate
01-02 00:01:39.079  1728  1810 D BluetoothAdapterService:
getAdapterService(): returning
com.android.bluetooth.btservice.AdapterService@4190d9f0
01-02 00:01:39.079  1728  1728 D BluetoothAdapterService(1100012016):
Message: 20
01-02 00:01:39.079  1728  1728 D BluetoothAdapterService(1100012016):
MESSAGE_PROFILE_CONNECTION_STATE_CHANGED
01-02 00:01:39.079  1728  1728 D BluetoothAdapterService(1100012016):
getState(): mAdapterProperties:
com.android.bluetooth.btservice.AdapterProperties@4190fba0
```

```
01-02 00:01:39.079  1309  1309 D BluetoothManager: mReceiver:
HEADSET_STATE_CHANGED_ACTION
01-02 00:01:39.079  1309  1309 D BluetoothManager: ==> new state: 0
01-02 00:01:39.079  1309  1309 D AudioRouter: onBluetoothIndicationChange
false
01-02 00:01:39.079  1309  1309 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-02 00:01:39.079  1309  1309 D BluetoothManager:
isBluetoothAvailable()...
01-02 00:01:39.079  2703  2703 D LocalBluetoothProfileManager: HEADSET
state change 2 -> 0
01-02 00:01:39.079  1728  1954 W bt-avp  : Dropped media packet
01-02 00:01:39.079  2703  2703 D CachedBluetoothDevice:
onProfileStateChanged: profile HEADSET newProfileState 0
01-02 00:01:39.079  1309  1309 D BluetoothManager:   ==> false
01-02 00:01:39.089   211  1117 V AudioPolicyService:
setDeviceConnectionState()
01-02 00:01:39.089   211  1117 V AudioPolicyManager:
setDeviceConnectionState() device: 20, state 0, address 00:0D:FD:5C:9E:13
```

APM sets state to disconnect for device AUDIO_DEVICE_OUT_BLUETOOTH_A2DP (0x80) as defined in system/core/include/system/audio.h.

```
01-02 00:01:39.089   211  1117 V AudioPolicyManager:
setDeviceConnectionState() disconnecting device 20
01-02 00:01:39.179  2048  2673 V MediaPlayer: pause
01-02 00:01:39.199   211  1116 V MediaPlayerService: [4] pause
01-02 00:01:39.199   211  1116 V StagefrightPlayer: pause
01-02 00:01:39.199   211  1116 V MediaPlayerService: [4] notify
(0xb7194c58, 7, 0, 0)
01-02 00:01:39.199   211  1116 V AudioSink: pause
01-02 00:01:39.199   211  1116 V AudioFlinger: pause(4097), calling pid
2048
01-02 00:01:40.259   211  1116 V AudioFlinger: remove track (4096) and
delete from mixer
01-02 00:01:40.259   211  1116 V AudioFlinger: PlaybackThread::Track
destructor
01-02 00:01:40.269   211  1116 V AudioFlinger: closeOutput() 13
01-02 00:01:40.299   211  2671 V audio_a2dp_hw: out_standby
01-02 00:01:40.299   211  2671 I audio_a2dp_hw: suspend_audio_datapath:
state 2
01-02 00:01:40.299   211  1116 I audio_a2dp_hw: adev_close_output_stream:
closing output (state 2)
01-02 00:01:40.299   211  1116 I audio_a2dp_hw: stop_audio_datapath: state
2
01-02 00:01:40.299   211  1116 V audio_a2dp_hw: a2dp_command: A2DP COMMAND
A2DP_CTRL_CMD_STOP
```

```
01-02 00:01:40.299   211  1116 V audio_a2dp_hw: a2dp_command: A2DP COMMAND
A2DP_CTRL_CMD_STOP DONE STATUS 0
01-02 00:01:40.299   211  1116 I audio_a2dp_hw: skt_disconnect: fd -1
01-02 00:01:40.299   211  1116 I audio_a2dp_hw: skt_disconnect: fd 25
01-02 00:01:40.299   211  1116 V audio_a2dp_hw: adev_close_output_stream:
done
01-02 00:01:40.369   211  1116 V AudioSink: stop
01-02 00:01:40.369   211  1116 V AudioSink: flush
01-02 00:01:40.369   211  1116 V AudioFlinger: flush(4097)
01-02 00:01:40.369   211  1116 V AudioSink: close
01-02 00:01:40.369   211  1116 V AudioFlinger: removeClient_l() pid 2048,
calling pid 2048
```

10.1.3.2 Kernel logs

Not required.

10.1.4 Customization

No customizations recommended.

10.1.5 Debugging

Table 10-1 Debugging and troubleshooting tips

Issue type	Symptoms	Troubleshooting steps
Audio mute	No audio is heard from the output device	▪ Ensure that Bluetooth framework has properly detected device as A2DP. ▪ Ensure that audio is routed to A2DP; check for the output device in APM.

11 Hands-free profile (HFP) client

HFP is a Bluetooth profile that facilitates making voice calls on a connected mobile phone without physically interacting with the device. The profile defines two roles:

- Audio Gateway (AG) – This device is the gateway of the audio, both for input and output. Typical devices acting as AGs are cellular phones.
- Hands-free unit (HF or HFP client) – This device acts as the remote audio input and output mechanism of AG. It also provides some remote control means.

The HFP client role is implemented on select QTI SoCs. Hence, an equipped device acts as Bluetooth voice input/output device for a Bluetooth connected mobile phone (also includes some remote-control functions)

For example, assume that a tablet with no cellular communication capability is connected to a mobile phone over Bluetooth. With HFP, this tablet can make/receive voice calls while using its microphone/speakers to capture/playback the audio for the voice call.

Figure 11-1 shows one possible use case enabled by HFP client implementation.

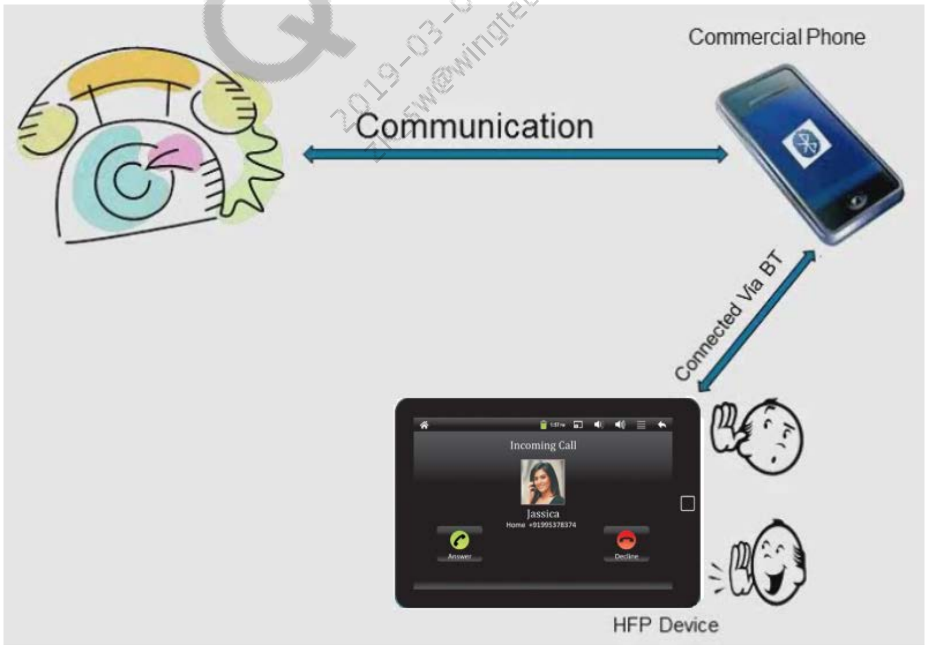


Figure 11-1 Data flow from and to the Bluetooth SCO in HFP client usage

The Bluetooth service in Android implements the HFP client role via code at packages/apps/Bluetooth/
src/com/android/bluetooth/hfpclient. A new parameter called hfp_enable is introduced and is set by the code here to enable or disable the audio path as necessary.

A new extension (hardware/qcom/audio/hal/audio_extn/hfp.c) is added to the Audio HAL and sets up or closes the audio path when it receives the indication to do so from the Bluetooth service. The extension is simple and performs a few steps to either enable or disable the audio path.

When the HFP session is started, the HFP extension first selects the physical input/output devices from which the audio is captured/played out. It then applies the following mixer controls to set up the audio paths involved.

```
<path name="hfp-sco">
     <ctl name="SLIMBUS_0_RX Port Mixer INTERNAL_BT_SCO_TX" value="1" />
     <ctl name="INTERNAL_BT_SCO_RX Audio Mixer MultiMedia6" value="1" />
     <ctl name="MultiMedia6 Mixer SLIM_0_TX" value="1" />
     <ctl name="SLIMBUS_DL_HL Switch" value="1" />
</path>
```

It opens and starts the PCM devices associated with HFP use case, at which point both the Tx and Rx paths are active.

When the HFP session is started, the PCM devices are closed and the use case is derouted.

Figure 11-2 shows the control sequence involved in enabling and disabling an HFP session.

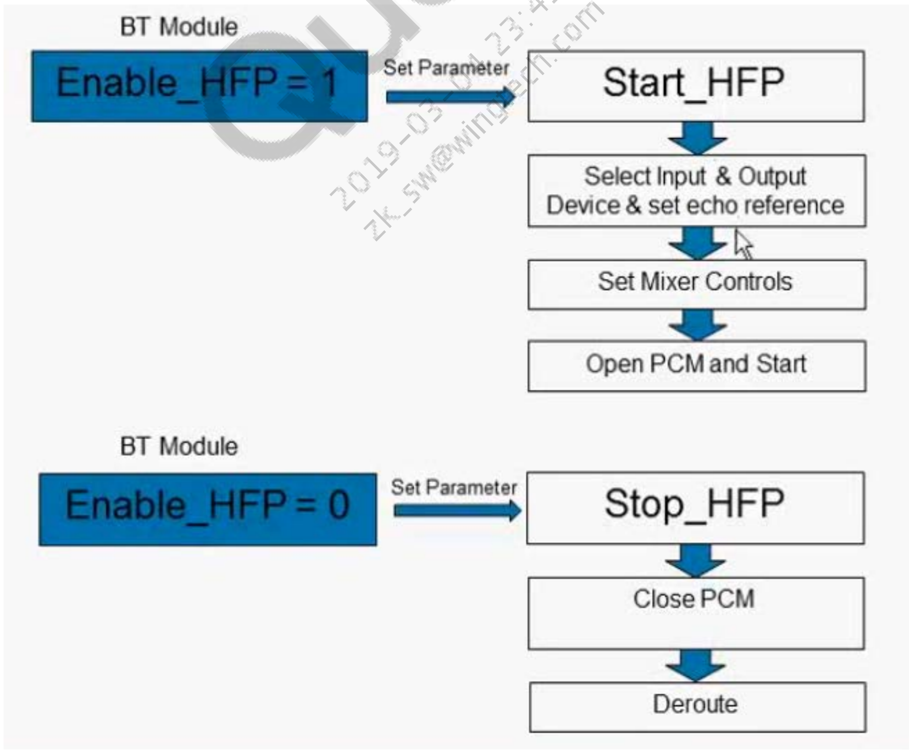


Figure 11-2 Control flow for enabling and disabling HFP session

The audio setup for an HFP client session involves setting up two loopback paths in the DSP:

- Downlink – An AFE loopback path is set up from the Bluetooth SCO Tx port (connected to the Bluetooth SoC downlink) to the SLIMbus Rx port (connected to a codec output device, such as a speaker) at the AFE. The platform driver used for this path is msm-pcm-hostless.c.
- Uplink – A path is set up so that the SLIMbus Tx port (connected to a codec input device, such as a microphone) data is fed to the Bluetooth SCO Tx port (connected to the Bluetooth SoC uplink) through an ASM session. This enables application of postprocessing effects such as ECNS on the uplink stream to improve its quality. This is the ASM loopback path.

A new driver, msm-pcm-loopback.c, is introduced to facilitate data movement in the ASM loopback scenario. It opens an ASM session in Loopback mode (Bidirectional mode). In Loopback mode, the ASM session creates two substreams. One captures the UL data from the input device and the other plays back the captured data (on which PP effects are applied) to the Bluetooth SoC Tx port.

Figure 11-3 shows the data flow paths for the uplink and downlink audio streams in HFP client operation.

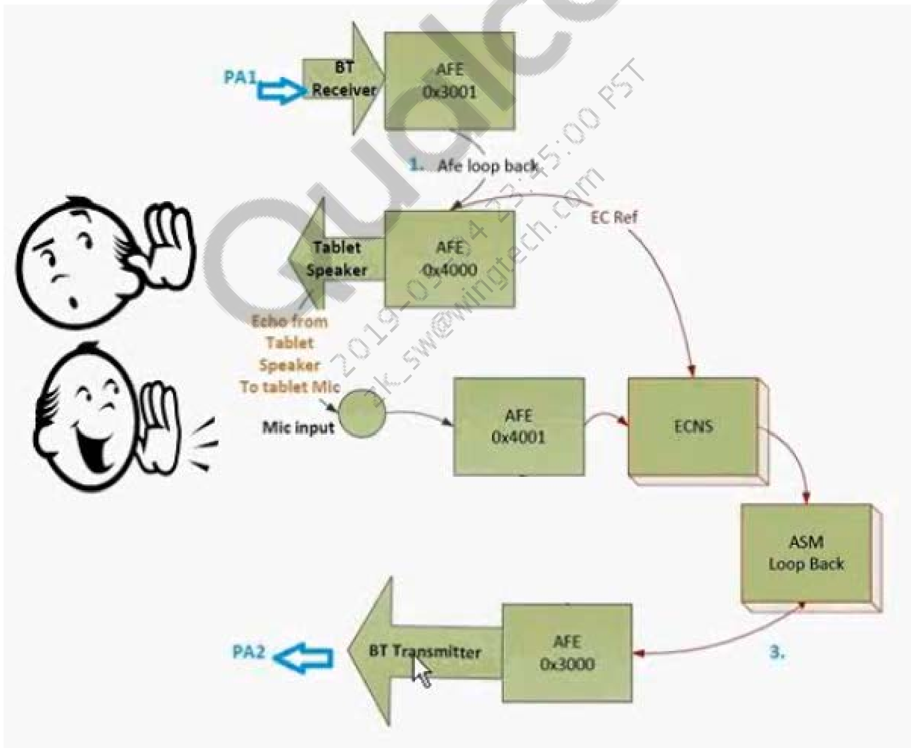


Figure 11-3 Sample use case enabled by HFP client implementation

11.1.1 Call flow

11.1.1.1 Starting HFP session

Figure 11-4 shows the call flow involved in the start of an HFP session

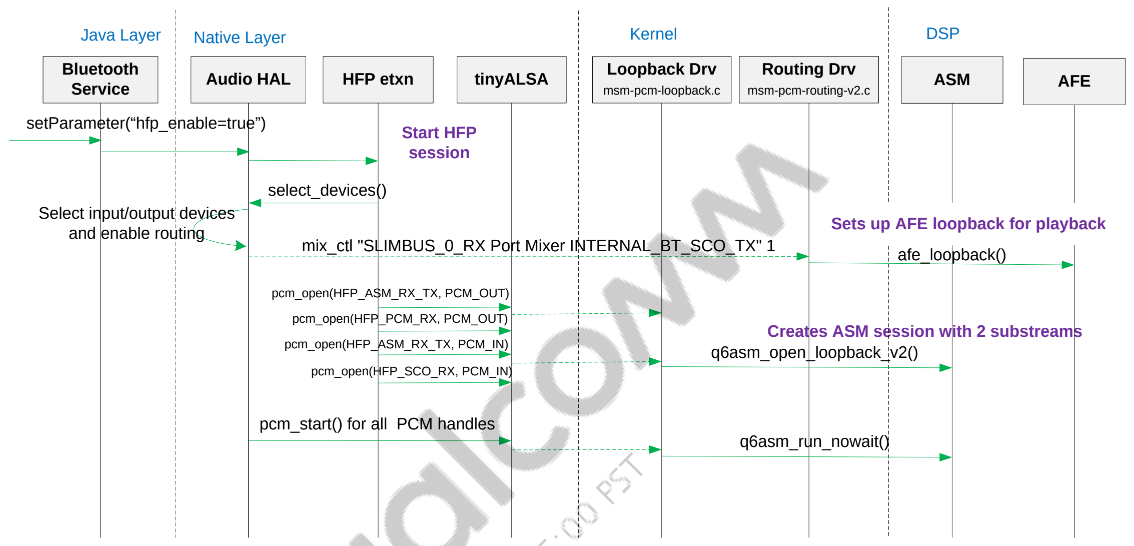

Java Layer
Native Layer
Kernel
DSP
Bluetooth Service
Audio HAL
HFP etxn
tinyALSA
Loopback Drv msm-pcm-loopback.c
Routing Drv msm-pcm-routing-v2.c
ASM
AFE
setParameter("hfp_enable=true")
Start HFP session
select_devices()
Select input/output devices and enable routing
Sets up AFE loopback for playback
mix_ctl "SLIMBUS_0_RX Port Mixer INTERNAL_BT_SCO_TX" 1
afe_loopback()
pcm_open(HFP_ASM_RX_TX, PCM_OUT)
pcm_open(HFP_PCM_RX, PCM_OUT)
pcm_open(HFP_ASM_RX_TX, PCM_IN)
pcm_open(HFP_SCO_RX, PCM_IN)
Creates ASM session with 2 substreams
q6asm_open_loopback_v2()
pcm_start() for all PCM handles
q6asm_run_nowait()

Figure 11-4 HFP session initialization call flow

11.1.1.2 Stopping HFP session

Figure 11-5 shows the call flow involved in the teardown of an HFP session

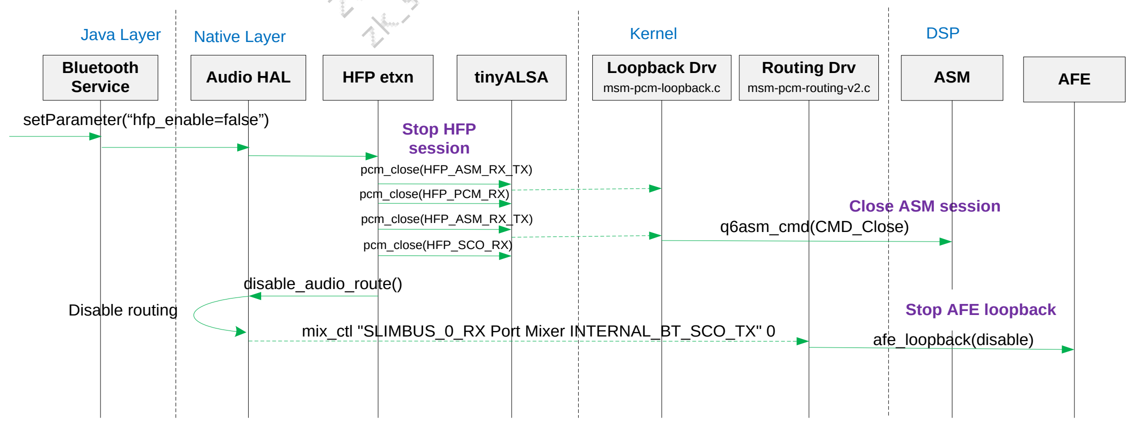


Figure 11-5 HFP session teardown call flow

11.1.2 Log collection

This section highlights the various files and/or components in which logging is enabled to troubleshoot issues related to audio playback on Android.

11.1.2.1 User space logs

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

11.1.2.2 Kernel logs

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file msm-pcm-routing-v2.c +p > /sys/kernel/debug/dynamic_debug/control
echo file msm-pcm-q6-v2.c +p > /sys/kernel/debug/dynamic_debug/control
echo file msm-pcm-loopback-v2.c +p > /sys/kernel/debug/dynamic_debug/control
echo file msm8974.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging on msm-pcm-q6-v2.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

11.1.2.3 Additional logging

Depending on the issue, additional logging is enabled in the user space and kernel space. In the user space, enable the logs on files listed at the following locations.

```
\platform\packages\apps\Bluetooth
\src\com\android\bluetooth\hfpclient\HandsfreeClientStateMachine.java

\frameworks\av\services\audioflinger
AudioFlinger.cpp

\harwdware\qcom\audio\hal
audio_hw.c
\audio_extn\hfp.c
\msm8974\platform.c

\harwdware\qcom\audio\hal\msm
platform.c
hw_info.c

\harwdware\qcom\audio\hal\policy_hal
AudioPolicyManager.cpp
```

In the kernel side, enable logs on the following files to debug issues related to APSS and DSP communication:

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6afe.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6asm.c  +p > /sys/kernel/debug/dynamic_debug/control
```

Enabling dynamic logging on q6asm.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

11.1.3 Log analysis

11.1.3.1 User space logs

The Bluetooth service starts the HFP with HF client role when connected to a device that advertises HFP with AG role.

```
01-02 00:43:41.759  3776  3776 D HandsfreeClientService: Received start
request. Starting profile...
01-02 00:43:41.759  3776  3776 I BluetoothHandsfreeClientServiceJni:
classInitNative succeeds
01-02 00:43:41.759  3776  3776 D HandsfreeClientStateMachine: make

01-02 00:43:41.929  3776  3776 D BluetoothAdapterService(1102872472):
MESSAGE_PROFILE_SERVICE_STATE_CHANGED
01-02 00:43:41.929  3776  3776 D BluetoothAdapterService:
onProfileServiceStateChange:
serviceName=com.android.bluetooth.hfpclient.HandsfreeClientService, state =
12, doUpdate = true
01-02 00:43:41.929  3776  3776 D BluetoothAdapterService: All profile
services started.

01-02 00:46:19.249  3776  3776 D BluetoothAdapterService(1102872472):
Message: 1
01-02 00:46:19.249  3776  3795 W BluetoothAdapterService: Stopping service
com.android.bluetooth.hfpclient.HandsfreeClientService
01-02 00:46:19.249  3776  3776 D BluetoothAdapterService(1102872472):
MESSAGE_PROFILE_SERVICE_STATE_CHANGED

01-02 00:46:19.269  3776  3776 D BluetoothMapService: Received stop
request...Stopping profile...
01-02 00:46:19.269  3776  3776 D BluetoothMapService: stop()
01-02 00:46:19.269  3776  3776 V BluetoothMapService: closeService
```

The HandsFreeClientStateMachine (a component of the HFP client profile) is initialized and moved to connected state. During this process, the device receives the information such as network status, battery level, operator name.

State definitions are defined in packages/apps/Bluetooth/src/com/android/bluetooth/hfp/HeadsetHalConstants.java

```
01-02 00:44:41.649  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_CONNECTION_STATE_CHANGED, value1:2,
value2:0, value3:0, value4:0, string: "null", device:BC:72:B1:0D:3B:E7}
01-02 00:44:41.649  3776  3810 D HandsfreeClientStateMachine: Disconnected:
Connection BC:72:B1:0D:3B:E7 state changed:2
01-02 00:44:41.649  3776  3810 W HandsfreeClientStateMachine: HFPClient
Connecting from Disconnected state
01-02 00:44:41.659  3776  3810 I HandsfreeClientStateMachine: Incoming AG
accepted

01-02 00:44:41.799  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_CALL, value1:0, value2:0, value3:0,
value4:0, string: "null", device:null}
01-02 00:44:41.799  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_CALLSETUP, value1:0, value2:0,
value3:0, value4:0, string: "null", device:null}
01-02 00:44:41.799  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_NETWORK_STATE, value1:1, value2:0,
value3:0, value4:0, string: "null", device:null}
01-02 00:44:41.799  3776  3810 D HandsfreeClientStateMachine: Connecting
process message: 100
01-02 00:44:41.799  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_NETWORK_SIGNAL, value1:4, value2:0,
value3:0, value4:0, string: "null", device:null}
01-02 00:44:41.799  3776  3810 D HandsfreeClientStateMachine: Connecting
process message: 100
01-02 00:44:41.799  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_ROAMING_STATE, value1:0, value2:0,
value3:0, value4:0, string: "null", device:null}
01-02 00:44:41.799  3776  3810 D HandsfreeClientStateMachine: Connecting
process message: 100
01-02 00:44:41.799  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_BATTERY_LEVEL, value1:4, value2:0,
value3:0, value4:0, string: "null", device:null}
01-02 00:44:41.799  3776  3810 D HandsfreeClientStateMachine: Connecting
process message: 100
01-02 00:44:41.799  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_CALLHELD, value1:0, value2:0, value3:0,
value4:0, string: "null", device:null}
01-02 00:44:41.919  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_OPERATOR_NAME, value1:0, value2:0,
value3:0, value4:0, string: "T-Mobile", device:null}
01-02 00:44:41.869  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_CONNECTION_STATE_CHANGED, value1:3,
value2:871, value3:43, value4:0, string: "null", device:BC:72:B1:0D:3B:E7}
01-02 00:44:41.869  3776  3810 D HandsfreeClientStateMachine: Connecting
process message: 100
01-02 00:44:41.869  3776  3810 D HandsfreeClientStateMachine: Connecting:
Connection BC:72:B1:0D:3B:E7 state changed:3
01-02 00:44:41.869  3776  3810 W HandsfreeClientStateMachine: HFPClient
Connected from Connecting state

01-02 00:44:41.909  3776  3810 D HandsfreeClientStateMachine: Connected:
Network state: 1
01-02 00:44:41.909  3776  3810 D HandsfreeClientStateMachine: Connected:
Signal level: 4
01-02 00:44:41.909  3776  3810 D HandsfreeClientStateMachine: Connected:
Roaming state: 0
01-02 00:44:41.909  3776  3810 D HandsfreeClientStateMachine: Connected:
Battery level: 4
01-02 00:44:41.929  3776  3810 D HandsfreeClientStateMachine: Connected:
Operator name: T-Mobile
```

When an incoming call is received, the HFP state machine controller sets the mode to AUDIO_MODE_RINGTONE and starts ringtone playback using the speaker device.

```
01-02 00:45:26.719  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_RING_INDICATION, value1:0, value2:0,
value3:0, value4:0, string: "null", device:null}
01-02 00:45:26.719  3776  3810 E HandsfreeClientStateMachine: start ringing
01-02 00:45:26.719   212  4193 V StagefrightPlayer: isPlaying
01-02 00:45:26.719  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_CLIP, value1:0, value2:0, value3:0,
value4:0, string: "18586518103", device:null}
01-02 00:45:26.719   212  4193 V MediaPlayerService: [3] isPlaying: 0
01-02 00:45:26.719  3776  3810 V MediaPlayer: isPlaying: 0
01-02 00:45:26.729   982  1446 I MediaFocusControl:  AudioFocus
requestAudioFocus() from AudioFocus_For_Phone_Ring_And_Calls
01-02 00:45:26.729  3776  3810 D HandsfreeClientStateMachine: setAudioMode
Setting audio mode from 0 to 1
01-02 00:45:26.729  3776  3806 V Avrcp   : New genId = 5, clearing = 1
01-02 00:45:26.729   212   212 V AudioPolicyService: setPhoneState()
01-02 00:45:26.729   212   212 D audio_hw_primary: adev_set_mode mode 1
01-02 00:45:26.739  3776  3810 V MediaPlayer: start

01-02 00:45:26.749   212   676 V audio_hw_primary: start_output_stream:
enter: usecase(0: deep-buffer-playback) devices(0x2)
01-02 00:45:26.749   212   676 V audio_hw_primary: enable_snd_device:
snd_device(2: speaker)
```

On accepting the call, the HFP state machine controller sets the mode to AUDIO_MODE_IN_CALL, which starts audio path setup.

```
01-02 00:45:33.819  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_AUDIO_STATE_CHANGED, value1:3,
value2:0, value3:0, value4:0, string: "null", device:BC:72:B1:0D:3B:E7}
01-02 00:45:33.819  3776  3810 D HandsfreeClientStateMachine: Connected
process message: 100
01-02 00:45:33.819  3776  3810 D HandsfreeClientStateMachine: Connected:
Audio state changed: BC:72:B1:0D:3B:E7: 3
01-02 00:45:33.819  3776  3844 E bt-rfcomm: PORT_DataInd,
p_port:0x5f179d34, p_data_co_callback is null
01-02 00:45:33.829   982  1398 I MediaFocusControl:  AudioFocus
requestAudioFocus() from AudioFocus_For_Phone_Ring_And_Calls
01-02 00:45:33.829  3776  3810 D HandsfreeClientStateMachine: setAudioMode
Setting audio mode from 1 to 2
01-02 00:45:33.829   212   212 V AudioPolicyService: setPhoneState()
01-02 00:45:33.829   212   212 D audio_hw_primary: adev_set_mode mode 2
```

HFP sample rate is set and then the hfp_enable flag is set, which get pushed down the stack to the Audio HAL.

```
01-02 00:45:33.999  3776  3810 D HandsfreeClientStateMachine:
hfp_enable=true
01-02 00:45:33.999  3776  3810 D HandsfreeClientStateMachine: mAudioWbs is
true
01-02 00:45:33.999  3776  3810 D HandsfreeClientStateMachine: Setting
sampling rate as 16000
01-02 00:45:33.999   212  1019 V AudioFlinger: setParameters(): io 0,
keyvalue hfp_set_sampling_rate=16000, calling pid 3776
01-02 00:45:33.999   212  1019 V SRS_Proc: ParamSet string:
hfp_set_sampling_rate=16000
01-02 00:45:33.999   212  1019 D audio_hw_primary: adev_set_parameters:
enter: hfp_set_sampling_rate=16000
01-02 00:45:33.999   212  1019 V voice   : voice_set_parameters: enter:
hfp_set_sampling_rate=16000
01-02 00:45:33.999   212  1019 V msm8974_platform: platform_set_parameters:
enter: hfp_set_sampling_rate=16000
01-02 00:45:33.999   212  1019 V msm8974_platform: platform_set_parameters:
exit with code(0)
01-02 00:45:33.999   212  4193 D audio_hw_primary: adev_set_parameters:
enter: hfp_enable=true
01-02 00:45:33.999   212  4193 V voice   : voice_set_parameters: enter:
hfp_enable=true
01-02 00:45:33.999   212  4193 V msm8974_platform: platform_set_parameters:
enter: hfp_enable=true
```

In the HAL, the HFP extension starts the HFP session. All use cases active on the device at the moment are disabled, and then use case hfp-sco-wb is enabled with output device voice-speaker and input device voice-speaker-mic.

```
01-02 00:45:33.999   212  4193 D audio_hw_hfp: start_hfp: enter
01-02 00:45:33.999   212  4193 D audio_hw_primary: select_devices:
out_snd_device(7: voice-speaker) in_snd_device(45: voice-speaker-mic)
01-02 00:45:33.999   212  4193 V audio_hw_primary: disable_audio_route:
reset mixer path: deep-buffer-playback
01-02 00:45:33.999   212  4193 V audio_hw_primary: disable_audio_route:
reset mixer path: low-latency-playback
01-02 00:45:34.039   212  4193 V audio_hw_primary: disable_snd_device:
snd_device(6: voice-handset)
01-02 00:45:34.039   212  4193 V audio_hw_primary: enable_snd_device:
snd_device(7: voice-speaker)
01-02 00:45:34.039   212  4193 V audio_hw_primary: enable_snd_device:
snd_device(45: voice-speaker-mic)

01-02 00:45:34.119   212  4193 V audio_hw_primary: enable_audio_route:
enter: usecase(6)
01-02 00:45:34.119   212  4193 V audio_hw_primary: enable_audio_route:
apply mixer path: hfp-sco-wb
01-02 00:45:34.119   212  4193 E audio_route: audio_route
audio_route_update_mixer SLIMBUS_0_RX Port Mixer INTERNAL_BT_SCO_TX
01-02 00:45:34.129   212  4193 E audio_route: audio_route
audio_route_update_mixer INTERNAL_BT_SCO_RX Audio Mixer MultiMedia6
01-02 00:45:34.129   212  4193 E audio_route: audio_route
audio_route_update_mixer MultiMedia6 Mixer SLIM_0_TX
01-02 00:45:34.129   212  4193 E audio_route: audio_route
audio_route_update_mixer SLIMBUS_DL_HL Switch
01-02 00:45:34.129   212  4193 E audio_route: audio_route
audio_route_update_mixer Internal BTSCO SampleRate
```

All PCM devices associated with HFP are then opened and started. Audio from the caller starts playing out of the speaker device and audio from the microphone device is sent to Bluetooth Tx for far end transmission.

```
01-02 00:45:34.129   212  4193 V audio_hw_hfp: start_hfp: HFP PCM devices
(hfp rx tx: 5 pcm rx tx: 23) for the usecase(6)
01-02 00:45:34.129   212  4193 V audio_hw_hfp: start_hfp: Opening PCM
playback device card_id(0) device_id(5)
01-02 00:45:34.129   212  4193 D audio_hw_hfp: start_hfp: Opening PCM
capture device card_id(0) device_id(23)
01-02 00:45:34.139   212  4193 V audio_hw_hfp: start_hfp: Opening PCM
capture device card_id(0) device_id(23)
.
.
.
```

The call is ended from the UI. The HandsFreeClientStateMachine controller receives the indication and terminates the call. When the call ended indication is received, it sets the mode to MODE_NORMAL.

```
01-02 00:45:46.009  3776  3810 D HandsfreeClientStateMachine:
terminateCall: 0
01-02 00:45:46.009  3776  3810 D HandsfreeClientStateMachine:
getFromCallsWithStates states:[2, 3]

01-02 00:45:47.499  3776  3798 D HandsfreeClientStateMachine:
incomingStackEvent {type:EVENT_TYPE_AUDIO_STATE_CHANGED, value1:0,
value2:0, value3:0, value4:0, string: "null", device:BC:72:B1:0D:3B:E7}
01-02 00:45:47.499  3776  3810 D HandsfreeClientStateMachine: AudioOn
process message: 100
01-02 00:45:47.499  3776  3810 D HandsfreeClientStateMachine: AudioOn audio
state changedBC:72:B1:0D:3B:E7: 0
01-02 00:45:47.509   212  1235 D audio_hw_primary: adev_set_mode mode 0
```

hfp_enable flag is then reset, which causes the HFP extension to stop the HFP session, deroute the HFP use case and the input/output devices, and also close the associated PCM devices.

```
01-02 00:45:47.759  3776  3810 D HandsfreeClientStateMachine:
abandonAudioFocus
01-02 00:45:47.769  3776  3810 D HandsfreeClientStateMachine:
hfp_enable=false
01-02 00:45:47.769   212  1235 D audio_hw_primary: adev_set_parameters:
enter: hfp_enable=false
01-02 00:45:47.769   212  1235 V msm8974_platform: platform_set_parameters:
enter: hfp_enable=false

01-02 00:45:47.849   212  1235 V audio_hw_primary: disable_audio_route:
enter: usecase(6)
01-02 00:45:47.849   212  1235 V audio_hw_primary: disable_audio_route:
reset mixer path: hfp-sco-wb
01-02 00:45:47.849   212  1235 E audio_route: audio_route
audio_route_update_mixer SLIMBUS_0_RX Port Mixer INTERNAL_BT_SCO_TX
01-02 00:45:47.849   212  1235 E audio_route: audio_route
audio_route_update_mixer INTERNAL_BT_SCO_RX Audio Mixer MultiMedia6
01-02 00:45:47.849   212  1235 E audio_route: audio_route
audio_route_update_mixer MultiMedia6 Mixer SLIM_0_TX
01-02 00:45:47.849   212  1235 E audio_route: audio_route
audio_route_update_mixer SLIMBUS_DL_HL Switch
01-02 00:45:47.859   212  1235 E audio_route: audio_route
audio_route_update_mixer Internal BTSCO SampleRate
```

11.1.3.2 Kernel logs

```
<7>[42094.735066] msm_pcm_routing_process_audio: reg 5 val 5 set 1
```

This indicates the PCM routing to BACKEND DAI ID 5 (MSM_BACKEND_DAI_INT_BT_SCO_RX) from FRONTEND DAI ID 5 (MSM_FRONTEND_DAI_MULTIMEDIA6). The BACKEND DAI IDs and FRONT END DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[ 2748.929642] msm_pcm_routing_process_audio: reg 3 val 5 set 1
```

This indicates the PCM routing to BACKEND DAI ID 3 (MSM_BACKEND_DAI_SLIMBUS_0_TX) from FRONTEND DAI ID 5 (MSM_FRONTEND_DAI_MULTIMEDIA6). The BACKEND DAI IDs and FRONT END DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[42094.738849] msm-pcm-routing msm-pcm-routing: dapm: power up widget
SLIMBUS_0_TX
<7>[42094.739737] msm-pcm-routing msm-pcm-routing: dapm: power up widget
MultiMedia6 Mixer
<7>[42094.739827] msm-pcm-routing msm-pcm-routing: dapm: power up widget
BE_IN
<7>[42094.740440] msm-pcm-routing msm-pcm-routing: dapm: power up widget
MM_UL6
<7>[42094.742318] msm-pcm-routing msm-pcm-routing: dapm: power up widget
INTERNAL_BT_SCO_RX Audio Mixer
<7>[42094.742455] msm-pcm-routing msm-pcm-routing: dapm: power up widget
INT_BT_SCO_RX
```

The pcm-loopback driver receives pcm_open requests for the capture and playback substreams. When the second substream request is received, the pcm-loopback driver opens a loopback session in ASM. An ASM session contains two substreams in it (one for capture and one for playback). Data from the capture stream is put into the playback stream, thus accomplishing loopback functionality.

```
<7>[42094.746995] msm-pcm-loopback msm-pcm-loopback: msm_pcm_open: pcm out
open: 1,0
<6>[42094.747007] msm-pcm-loopback msm-pcm-loopback: msm_pcm_open: Instance
= 1, Stream ID = MultiMedia6 (*)
<7>[42094.747092] msm-dai-q6-dev msm-dai-q6-dev.12288: channels 1 sample
rate 8000 entered
<7>[42094.747104] msm_dai_q6_bt_fm_hw_params: setting bt_fm parameters
<7>[42094.747124] msm-pcm-loopback msm-pcm-loopback: msm_pcm_hw_params: ASM
loopback
```

```
<7>[42094.751016] msm-pcm-loopback msm-pcm-loopback: msm_pcm_open: pcm out
open: 2,1
<6>[42094.755111] msm-pcm-loopback msm-pcm-loopback: msm_pcm_open: Instance
= 2, Stream ID = MultiMedia6 (*)
<7>[42094.755497] msm_slim_0_tx_be_hw_params_fixup()
<7>[42094.755512] msm_snd_hw_params: taiko_tx1_tx_dai_id_1_ch=0
<7>[42094.755531] msm_snd_hw_params:
msm_slim_0_tx_ch(1)user_set_tx_ch(1)tx_ch_cnt(1)
<7>[42094.755549] msm-dai-q6-dev msm-dai-q6-dev.16385: enter
msm_dai_q6_set_channel_map, id = 16385
<7>[42094.755567] msm_dai_q6_set_channel_map: find number of channels[0]
ch[134]
<7>[42094.755585] msm_dai_q6_set_channel_map:SLIMBUS_0_TX cnt[1] ch[134 0]
<7>[42094.755614] msm-dai-q6-dev msm-dai-q6-dev.16385:
msm_dai_q6_slim_bus_hw_params:slimbus_dev_id[0] bit_wd[16] format[0]
<7>[42094.755619] num_channel 1  shared_ch_mapping[0]  134
<7>[42094.755622] slave_port_mapping[1]  0 slave_port_mapping[2]  0
<7>[42094.755625] sample_rate 48000
<7>[42094.755672] msm-pcm-loopback msm-pcm-loopback: msm_pcm_hw_params: ASM
loopback
<7>[42094.756981] q6asm_open_loopback_v2: session[3]
```

The AFE port, 0x3000 (AFE_PORT_ID_INTERNAL_BT_SCO_RX) defined in /kernel/include/sound/apr_audio-v2.h is opened.

```
<7>[42094.756332] msm8974-asoc-taiko fe02c000.sound: Capture: checking
paths from INTHFP_UL_HL
<7>[42094.756347] msm-pcm-routing msm-pcm-routing: added INTHFP_UL_HL in
widget list pos 0
<7>[42094.756389] msm-pcm-routing msm-pcm-routing: added INT_BT_SCO_TX in
widget list pos 1
<7>[42094.756669] msm8974-asoc-taiko fe02c000.sound: Capture: found 1 paths
from INTHFP_UL_HL
<7>[42094.756808] msm-dai-q6-dev msm-dai-q6-dev.12289: channels 1 sample
rate 8000 entered
<7>[42094.756818] msm_dai_q6_bt_fm_hw_params: setting bt_fm parameters
<7>[42094.757639] adm_open: port 0x3000 path:1 rate:8000 mode:1 perf_mode:0
<7>[42094.757700] adm_open: Port ID 0x3000, index 27
<7>[42094.757711] send_adm_custom_topology: no cal to send addr= 0x3e373c
<7>[42094.757719] adm_open:opening ADM: perf_mode: 0
<7>[42094.757801] adm_open: port_id=0x3000 rate=8000 topology_id=0x10313
<7>[42094.765200] afe_port_start: port id: 0x3000
<7>[42094.765207] afe_q6_interface_prepare:
<<7>[42094.765279] afe_port_start: port_id 0x3000, mad_type 0
```

The pcm-loopback driver then performs pcm_prepare for the capture and playback substreams in the ASM. AFE port, 0x4001 (AFE_PORT_ID_SLIMBUS_MULTI_CHAN_0_TX) defined in /kernel/include/sound/apr_audio-v2.h is opened.

```
<7>[42094.769171] msm-pcm-loopback msm-pcm-loopback: msm_pcm_prepare: ASM
loopback stream:0
<7>[42094.769189]  MSM8974 HFP TX: rtd stream 0 event 1
<7>[42094.770098]  INT_BT_SCO_RX: rtd stream 0 event 1
<7>[42094.770127] msm-pcm-loopback msm-pcm-loopback: msm_pcm_trigger:
playback_start:1,capture_start:0
<7>[42094.770159] adm_open: port 0x4001 path:2 rate:48000 mode:1
perf_mode:0
<7>[42094.770168] adm_open: Port ID 0x4001, index 16
<7>[42094.770181] send_adm_custom_topology: no cal to send addr= 0x3e373c
<7>[42094.770192] adm_open:opening ADM: perf_mode: 0
<7>[42094.770202] adm_open: port_id=0x4001 rate=48000 topology_id=0x10315
<7>[42094.773335] adm_callback_debug_print: code = 0x10329 PL#0[0],
PL#1[f0e40009], size = 8
<7>[42094.773345] adm_callback: coppid rxed=9
<7>[42094.773384] adm_open: index: 16 coppid: 9
<7>[42094.776385] afe_port_start: port id: 0x4001
<7>[42094.776391] afe_q6_interface_prepare:
```

The starting of the capture stream causes widgets in the capture path (microphone, and so on) to be enabled.

```
<7>[42094.781679] taiko_codec taiko_codec: dapm: power up widget AIF1 CAP
<7>[42094.781734] taiko_codec taiko_codec: dapm: power up widget AIF1_CAP
Mixer
<7>[42094.781789] taiko_codec taiko_codec: dapm: power up widget SLIM TX7
MUX
<7>[42094.781847] taiko_codec taiko_codec: dapm: power up widget DEC7 MUX
<7>[42094.782006] taiko_codec taiko_codec: dapm: power up widget CDC_CONN
<7>[42094.782060] taiko_codec taiko_codec: dapm: power up widget DMIC1
<7>[42094.782160] taiko_codec taiko_codec: dapm: power up widget MIC BIAS1
External
<7>[42094.782214] taiko_codec taiko_codec: dapm: power up widget Digital
Mic2
<7>[42094.782269] taiko_codec taiko_codec: dapm: power up widget Digital
Mic1
<7>[42094.782740] taiko_codec taiko_codec: dapm: power up widget LDO_H

<7>[42094.811930] msm-pcm-loopback msm-pcm-loopback: msm_pcm_prepare: ASM
loopback stream:1
<7>[42094.811951]  MSM8974 HFP TX: rtd stream 1 event 1
<7>[42094.812679]  SLIMBUS_0_TX: rtd stream 1 event 1
<7>[42094.812716] msm-pcm-loopback msm-pcm-loopback: msm_pcm_trigger:
playback_start:1,capture_start:1
<7>[42094.812804]  SLIMBUS_0 Hostless: rtd stream 0 event 1
```

```
<7>[42094.812881] msm-pcm-routing msm-pcm-routing: dapm: power up widget
SLIM0_DL_HL
<7>[42094.813022] msm-pcm-routing msm-pcm-routing: dapm: power up widget
SLIMBUS_DL_HL
<7>[42094.813107] msm_pcm_loopback_event_handler
```

Next, the AFE ports associated with the AFE loopback are opened. Port 0x4000 (AFE_PORT_ID_SLIMBUS_MULTI_CHAN_0_RX) and port 0x3001(AFE_PORT_ID_INTERNAL_BT_SCO_TX) defined in /kernel/include/sound/apr_audio-v2.h are opened.

```
<7>[  224.143310] afe_port_start: port id: 0x3001
<7>[  224.143315] afe_q6_interface_prepare:
<7>[  224.143344] afe_send_hw_delay port_id 12289 rate 8000 delay_usec
572662306 status -22
<7>[  224.143350] afe_port_get_mad_type: Non Slimbus port_id 0x3001
<7>[  224.143355] afe_port_start: port_id 0x3001, mad_type 0

<7>[  224.362811] afe_port_start: port id: 0x4001
<7>[  224.362816] afe_q6_interface_prepare:
<7>[  224.362850] afe_send_hw_delay port_id 16385 rate 48000 delay_usec
572662306 status -22
<7>[  224.362857] afe_port_start: port_id 0x4001, mad_type 0
```

Audio playback and capture for the HFP session continues until it is stopped. When the HFP extension receives setParameters indication with hfp_enable=false, it closes all the PCM streams and rests the routing.

```
<7>[42101.951559] msm-pcm-loopback msm-pcm-loopback:
msm_pcm_trigger:Pause/Stop - playback_start:1,capture_start:1
<7>[42101.979904] msm-pcm-loopback msm-pcm-loopback: msm_pcm_close: end pcm
call:0
<7>[42101.995843] msm-pcm-loopback msm-pcm-loopback:
msm_pcm_trigger:Pause/Stop - playback_start:0,capture_start:1

<7>[42101.989528] afe_close: port_id 0x3000, mad_type 0

<7>[42102.023939] msm-pcm-loopback msm-pcm-loopback: msm_pcm_close: end pcm
call:1

<7>[42102.023974] afe_close: port_id 0x4001, mad_type 0

<7>[42102.031269]  MSM8974 HFP TX: rtd stream 1 event 2
<7>[42102.032372]  SLIMBUS_0_TX: rtd stream 1 event 2
<7>[42102.032457]  SLIMBUS_0 Hostless: rtd stream 0 event 2
<7>[42102.032561] msm-pcm-routing msm-pcm-routing: dapm: power down widget
SLIM0_DL_HL
<7>[42102.032615] msm-pcm-routing msm-pcm-routing: dapm: power down widget
SLIMBUS_DL_HL
```

```
<7>[42102.034245] afe_close: port_id 0x3001, mad_type 0

<7>[42102.040704]  INT_HFP_BT Hostless: rtd stream 1 event 2
<7>[42102.040794] msm-pcm-routing msm-pcm-routing: dapm: power down widget
INTHFP_UL_HL
<7>[42102.040848] msm-pcm-routing msm-pcm-routing: dapm: power down widget
INT_BT_SCO_TX
<7>[42102.041285] msm-pcm-routing msm-pcm-routing: reg 0
<7>[42102.042043]  INT_BT_SCO_TX: rtd stream 1 event 2

<7>[42103.437314] afe_close: port_id 0x4000, mad_type 0
```

Finally, the routing is disabled.

```
<7>[42102.044033] msm_pcm_routing_process_audio: reg 5 val 5 set 0
<7>[42102.044148] msm-pcm-routing msm-pcm-routing: dapm: power down widget
MM_UL6
<7>[42102.044285] msm-pcm-routing msm-pcm-routing: dapm: power down widget
INTERNAL_BT_SCO_RX Audio Mixer
<7>[42102.044340] msm-pcm-routing msm-pcm-routing: dapm: power down widget
MultiMedia6 Mixer
<7>[42102.044395] msm-pcm-routing msm-pcm-routing: dapm: power down widget
INT_BT_SCO_RX
<7>[42102.044449] msm-pcm-routing msm-pcm-routing: dapm: power down widget
SLIMBUS_0_TX
<7>[42102.044572] msm-pcm-routing msm-pcm-routing: dapm: power down widget
BE_IN

<7>[42102.046383] msm_pcm_routing_process_audio: reg 3 val 5 set 0
<7>[42103.445478] msm8974_snd_shudown(): substream = subdevice #0 stream
= 0
<7>[42103.446249] taiko_codec taiko_codec: dapm: power down widget AIF1 PB
<7>[42103.446376] taiko_codec taiko_codec: dapm: power down widget SLIM RX1
MUX
<7>[42103.446500] taiko_codec taiko_codec: dapm: power down widget SLIM RX1
<7>[42103.446624] taiko_codec taiko_codec: dapm: power down widget RX1 MIX1
INP1
<7>[42103.446750] taiko_codec taiko_codec: dapm: power down widget RX1 MIX1
<7>[42103.446932] taiko_codec taiko_codec: dapm: power down widget
COMP1_CLK
<7>[42103.447057] taiko_codec taiko_codec: dapm: power down widget RX1 MIX2
<7>[42103.447180] taiko_codec taiko_codec: dapm: power down widget RX1
CHAIN
<7>[42103.447303] taiko_codec taiko_codec: dapm: power down widget
CLASS_H_DSM MUX
<7>[42103.447426] taiko_codec taiko_codec: dapm: power down widget DAC1
<7>[42103.449482] taiko_codec taiko_codec: dapm: power down widget RX_BIAS
```

```
<7>[42103.449611] taiko_codec taiko_codec: dapm: power down widget
EAR_PA_MIXER
<7>[42103.451216] taiko_codec taiko_codec: dapm: power down widget MCLK
<7>[42103.451343] taiko_codec taiko_codec: dapm: power down widget EAR PA
<7>[42103.451466] taiko_codec taiko_codec: dapm: power down widget EAR
<7>[42103.468085] msm8974_mclk_event: event = 8
<7>[42103.468107] msm_snd_enable_codec_ext_clk: enable = 0 clk_users = 1
<7>[42103.470133]  MSM8974 LowLatency: rtd stream 0 event 2
<7>[42103.470322] msm-pcm-routing msm-pcm-routing: dapm: power down widget
MM_DL5
<7>[42103.470465] msm-pcm-routing msm-pcm-routing: dapm: power down widget
SLIMBUS_0_RX Audio Mixer
<7>[42103.470609] msm-pcm-routing msm-pcm-routing: dapm: power down widget
SLIMBUS_0_RX
<7>[42103.470790] msm-pcm-routing msm-pcm-routing: dapm: power down widget
BE_OUT
```

11.1.4 Customization

There are no customizations for this feature.

11.1.5 Debugging

This section is not applicable to this release.

12 FM radio

The FM radio application is responsible for calling the APIs of the FM driver, which talks with the WCNSS and WCN for FM-related functionality. For example, turning on and off the FM chip scanning through channels. The WCN is the analog RF circuit, and the WCNSS is the digital core for wireless connectivity supporting the FM receiver. The WCNSS interfaces with the LPASS and WCN. It is responsible for converting the analog FM to digital FM and sending the PCM data to the LPASS. The digital PCM data then is directly routed to the output device, for example, headset or speaker. The FM application sends an Android intent to the audio service whenever the FM application is launched. The audio service sends the information to the audio policy manager and Audio HAL, which sets the routing in the LPASS so that the PCM data received from the WCNSS is sent to the output device.

Figure 12-1 describes the FM software components and dataflow.

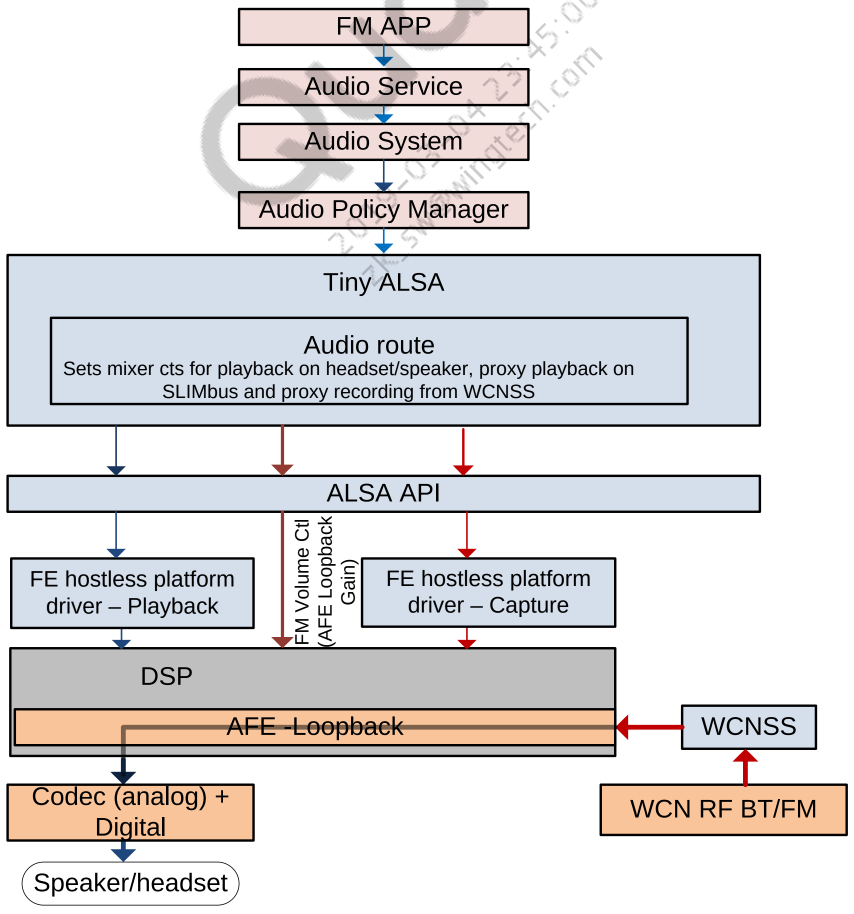

FM APP
Audio Service
Audio System
Audio Policy Manager
Tiny ALSA
Audio route
Sets mixer cts for playback on headset/speaker, proxy playback on SLIMbus and proxy recording from WCNSS
ALSA API
FE hostless platform driver – Playback
FM Volume Ctl (AFE Loopback Gain)
FE hostless platform driver – Capture
DSP
AFE -Loopback
WCNSS
WCN RF BT/FM
Codec (analog) + Digital
Speaker/headset

Figure 12-1 FM audio data flow

12.1.1 Call flow

This section is not applicable to this release.

12.1.2 Log collection

This section highlights the various files and/or components in which logging is enabled to troubleshoot issues related to audio playback on Android.

12.1.2.1 User space logs

This command clears the current logcat logs and starts logging the user space logs to the file logcat.txt and the stdout simultaneously.

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

NOTE: For the "tee" command to work, Cygwin must be installed or the command must be executed in the Linux environment.

12.1.2.2 Kernel logs

If kernel logs are required, enable them as follows.

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo -n "file q6afe.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm-pcm-routing-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
```

12.1.2.3 Depending on the issue, additional logging must be enabled in the user space and kernel space. In the user space, enable logs on files listed at the following locations, if the facility exists:

Depending on the issue, additional logging must be enabled in the user space and kernel space. In the user space, enable logs in files listed at the following locations, if the facility exists:

```
\hardware\qcom\audio\hal\audio_extn
fm.c

\frameworks\av\media\libmediaplayerservice
MediaPlayerService.cpp
StagefrightPlayer.cpp

\frameworks\av\media\libmedia
AudioSystem.cpp
AudioTrack.cpp
mediaplayer.cpp
```

```
\frameworks\av\media\libstagefright
AwesomePlayer.cpp
AudioPlayer.cpp
MP3Extractor.cpp
OMXClient.cpp
OMXCodec.cpp

\frameworks\av\services\audioflinger
AudioFlinger.cpp
AudioMixer.cpp
AudioPolicyService.cpp

\harwdware\qcom\audio\hal
audio_hw.c

\harwdware\qcom\audio\hal\msm8974
platform.c
hw_info.c

\harwdware\qcom\audio\hal\policy_hal
AudioPolicyManager.cpp

\harwdware\libhardware_legacy\audio
AudioPolicyManagerBase.cpp
```

In the kernel side, logs on the following files are enabled to debug issues related to APSS and DSP communication.

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file soc-pcm.c  +p > /sys/kernel/debug/dynamic_debug/control
echo file msm-dai-q6-v2.c  +p > /sys/kernel/debug/dynamic_debug/control
```

12.1.3 Log analysis

12.1.3.1 User space logs

FM application is started.

```
01-02 00:01:00.699  1330  1330 I ActivityManager: Timeline:
Activity_launch_request id:com.caf.fmradio time:76060
01-02 00:01:00.699   963  1323 I ActivityManager: START u0
{act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER]
flg=0x10200000 cmp=com.caf.fmradio/.FMRadio} from pid 1330
```

```
01-02 00:01:00.849  2189  2189 D FMRadio : onCreate - Height : 1280 -
Width  : 720
01-02 00:01:00.959  2189  2189 D FMRadio : FMRadio: onStart
01-02 00:01:00.959  2189  2189 E FMRadio : bindToService: Context with
serviceconnection callback
01-02 00:01:00.959  2189  2189 W ContextImpl: Calling a method in the
system process without a qualified user:
android.app.ContextImpl.startService:1487
android.content.ContextWrapper.startService:494
com.caf.fmradio.FMRadio.bindToService:2898
com.caf.fmradio.FMRadio.onStart:465
android.app.Instrumentation.callActivityOnStart:1171
01-02 00:01:00.969  2189  2189 D FMRadio : onStart: Start Service completed
successfully
01-02 00:01:00.969  2189  2189 W ContextImpl: Calling a method in the
system process without a qualified user:
android.app.ContextImpl.bindService:1551
android.content.ContextWrapper.bindService:517
com.caf.fmradio.FMRadio.bindToService:2901
com.caf.fmradio.FMRadio.onStart:465
android.app.Instrumentation.callActivityOnStart:1171
01-02 00:01:00.979  2189  2189 D FMRadio : FMRadio: onResume
01-02 00:01:00.989  2189  2189 D FmSharedPreferences: Load preferences
01-02 00:01:00.989  2189  2189 D FmSharedPreferences:
=====================================================
01-02 00:01:00.989  2189  2189 D FmSharedPreferences: Country     :0
01-02 00:01:00.989  2189  2189 D FmSharedPreferences: RadioBand   :0
01-02 00:01:00.989  2189  2189 D FmSharedPreferences: Emphasis    :0
01-02 00:01:00.989  2189  2189 D FmSharedPreferences: ChSpacing   :0
01-02 00:01:00.989  2189  2189 D FmSharedPreferences: RdsStd      :0
01-02 00:01:00.989  2189  2189 D FmSharedPreferences: LowerLimit  :87500
01-02 00:01:00.989  2189  2189 D FmSharedPreferences: UpperLimit  :107900
01-02 00:01:00.989  2189  2189 D FmSharedPreferences:
=====================================================
…
01-02 00:01:00.999   217   217 V audio_a2dp_hw: adev_get_parameters
01-02 00:01:00.999  2189  2189 D FMService:  is A2DP device Supported In
HALfalse
01-02 00:01:01.009  2189  2189 D FMService: onStart
01-02 00:01:01.009  2189  2189 D FMService: onBind
01-02 00:01:01.069  2189  2189 D FMService: onCallStateChanged: State - 0
01-02 00:01:01.069  2189  2189 D FMService: onCallStateChanged:
incomingNumber -
01-02 00:01:01.069  2189  2189 D FMService: onDataActivity – 0
```

A headset detect event is received.

```
01-02 00:01:01.069  2189  2189 D FMService: ACTION_HEADSET_PLUG Intent
received
01-02 00:01:01.069  2189  2189 D FMService: mReceiver: ACTION_HEADSET_PLUG
01-02 00:01:01.069  2189  2189 D FMService: ==> intent: Intent
{ act=android.intent.action.HEADSET_PLUG flg=0x40000010 (has extras) }
01-02 00:01:01.069  2189  2189 D FMService:     state: 1
01-02 00:01:01.069  2189  2189 D FMService:     name: h2w
01-02 00:01:01.069  2189  2189 E FMRadio : onServiceConnected: mCallback
01-02 00:01:01.069  2189  2189 E FMRadio : ServiceConnection:
onServiceConnected:
01-02 00:01:01.069  2189  2189 D FMService: fmOn: RadioBand   :0
01-02 00:01:01.069  2189  2189 D FMService: fmOn: Emphasis    :0
01-02 00:01:01.069  2189  2189 D FMService: fmOn: ChSpacing   :0
01-02 00:01:01.069  2189  2189 D FMService: fmOn: RdsStd      :0
01-02 00:01:01.069  2189  2189 D FMService: fmOn: LowerLimit  :87500
01-02 00:01:01.069  2189  2189 D FMService: fmOn: UpperLimit  :107900
01-02 00:01:01.069  2189  2189 V FMRadio : enable: CURRENT-STATE : FMOff --
-> NEW-STATE : FMRxStarting
01-02 00:01:01.069  2189  2189 D fmradio : VIDIOC_QUERYCAP returns :0:
version: 197632
01-02 00:01:01.069  2189  2189 D fmradio : Driver Version(Same as ChipId):
30400
01-02 00:01:01.079   353   353 E QCALOG  : [MessageQ] ProcessNewMessage:
[XTWiFi-PE] unknown deliver target [OS-Agent]
01-02 00:01:01.079   353   353 E QCALOG  : [MessageQ] ProcessNewMessage:
[XT-CS] unknown deliver target [OS-Agent]
01-02 00:01:01.109  2398  2398 I qcom-fm : /system/etc/init.qcom.fm.sh: In
FM shell Script
01-02 00:01:01.109  2399  2399 I qcom-fm : /system/etc/init.qcom.fm.sh:
mode: normal
01-02 00:01:01.109  2400  2400 I qcom-fm : /system/etc/init.qcom.fm.sh:
isAnalog:
01-02 00:01:01.119  2401  2401 I qcom-fm : /system/etc/init.qcom.fm.sh:
Transport : smd
01-02 00:01:01.119  2402  2402 I qcom-fm : /system/etc/init.qcom.fm.sh:
Version : 197632
01-02 00:01:01.129  2403  2403 I qcom-fm : /system/etc/init.qcom.fm.sh:
inserting the radio transport module
01-02 00:01:01.139  2406  2406 I qcom-fm : /system/etc/init.qcom.fm.sh: FM
QSoC calibration and firmware download succeeded
01-02 00:01:01.279  2189  2189 E fmradio : init_success:1 after 0.200000
seconds
01-02 00:01:01.279  2189  2189 D FmTransceiver: Opened 54
01-02 00:01:01.279  2189  2189 D FmTransceiver: turning on 1
01-02 00:01:01.279  2189  2189 E fmradio : id(8000004) value: 1
01-02 00:01:01.319  2189  2189 E fmradio : id(8000029) value: 0
```

```
01-02 00:01:01.319  2189  2189 D FmTransceiver: Calling fmConfigure
01-02 00:01:01.319  2189  2189 V FmConfig: In fmConfigure
01-02 00:01:01.319  2189  2189 V FmConfig: fmConfigure() : FM Srch Alg :
OLD
01-02 00:01:01.319  2189  2408 D FMRadio : Starting listener 54
01-02 00:01:01.319  2189  2189 D FMService: Analog Path is not supported
01-02 00:01:01.319  2189  2189 D FMService: mReceiver.enable done,
Status :true
01-02 00:01:01.319  2189  2189 D FMService: setLowPowerMode: false
01-02 00:01:01.319  2189  2189 E fmradio : id(8000011) value: 0
01-02 00:01:01.319  2189  2189 D FMService: setLowPowerMode done,
Status :true
01-02 00:01:01.319  2189  2189 D FMService: mAudioManager.setFmRadioOn =
true
01-02 00:01:01.319  2189  2189 D FMService: In startFM
01-02 00:01:01.319  2189  2408 D FMRadio : Received event. Count: 1
01-02 00:01:01.329  2189  2408 D FMRadio : Received <0>
01-02 00:01:01.329  2189  2408 D FMRadio : Got READY_EVENT
```

The FM state machine moved to an FMState_Rx_Turned_On state. The FM primary and intermediate states are listed in the vendor/qcom/opensource/fm/qcom/fmradio/FmTransceiver.java file.

```
01-02 00:01:01.329  2189  2408 V FMRadio : RxEvtList: CURRENT-STATE :
FMRxStarting ---> NEW-STATE : FMRxOn
01-02 00:01:01.329  2189  2408 D FMService: FmRxEvEnableReceiver
01-02 00:01:01.329  2189  2408 E fmradio : id(8000006) value: 40
01-02 00:01:01.329   963  1318 I MediaFocusControl:  AudioFocus
requestAudioFocus() from
android.media.AudioManager@420cdc28com.caf.fmradio.FMRadioService$16@420c33
80
01-02 00:01:01.329  2189  2189 D FMService: FM registering for
registerMediaButtonEventReceiver
01-02 00:01:01.329   963  1321 I MediaFocusControl:   Remote Control
registerMediaButtonIntent() for PendingIntent{41ea3230:
PendingIntentRecord{41ca67f0 com.caf.fmradio broadcastIntent}}
```

Set the audio path for FM.

```
01-02 00:01:01.329  2189  2189 D FMService: FMRadio: Requesting to start FM
01-02 00:01:01.329   217  1145 V AudioPolicyService:
setDeviceConnectionState()
```

AUDIO_DEVICE_OUT_FM device is 80000, as defined in system/core/include/system/audio.h

```
01-02 00:01:01.329   217  1145 V AudioPolicyManager:
setDeviceConnectionState() device: 80000, state 1, address
01-02 00:01:01.329   217  1145 V AudioPolicyManager:
setDeviceConnectionState() connecting device 80000
01-02 00:01:01.329   217  1145 V AudioPolicyManager:
setDeviceConnectionState() checkOutputsForDevice() returned 2 outputs
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 5, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 5, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 1, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 1, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 2, device 6
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 2, device 6
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 3, device 6
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 2, device 6
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 3, device 6
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 0, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 4, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 4, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 1, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 2, device 6
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 2, device 6
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 3, device 6
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
```

```
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 4, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 5, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:01.329   217  1145 V AudioPolicyManager:
setDeviceConnectionState() setParameters handle_fm
01-02 00:01:01.329   217  1145 V AudioPolicyService: inserting command: 3
at index 0, num commands 0
01-02 00:01:01.329   217  1145 V AudioPolicyService: AudioCommandThread()
adding set parameter string handle_fm=524292, io 2 ,delay 0
01-02 00:01:01.329   217   634 V AudioPolicyService: AudioCommandThread()
waking up
01-02 00:01:01.329   217   634 V AudioPolicyService: AudioCommandThread()
processing set parameters string handle_fm=524292, io 2
01-02 00:01:01.329   217   634 V AudioFlinger: setParameters(): io 2,
keyvalue handle_fm=524292, calling pid 217
01-02 00:01:01.329   217   634 V AudioFlinger: ThreadBase::setParameters()
handle_fm=524292
```

The output device selected is AUDIO_DEVICE_OUT_WIRED_HEADSET (0x4). The devices are listed in the system/core/include/system/audio.h file.

```
01-02 00:01:01.339   217   639 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: handle_fm=524292
01-02 00:01:01.339   217   639 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-02 00:01:01.339   217   639 V audio_hw_fm: audio_extn_fm_set_parameters:
enter
01-02 00:01:01.339   217   639 D audio_hw_fm: audio_extn_fm_set_parameters:
FM usecase
01-02 00:01:01.339   217   639 D audio_hw_fm: fm_start: enter
01-02 00:01:01.339   217   639 V msm8974_platform:
platform_get_output_snd_device: enter: output devices(0x2)
01-02 00:01:01.339   217   639 V msm8974_platform:
platform_get_output_snd_device: exit: snd_device(speaker)
01-02 00:01:01.339   217   639 D audio_hw_primary: select_devices:
out_snd_device(2: speaker) in_snd_device(0: )
01-02 00:01:01.339   217   639 D hardware_info: hw_info_append_hw_type :
device_name = speaker
01-02 00:01:01.339   217   639 V audio_hw_primary: enable_snd_device:
snd_device(2: speaker)
01-02 00:01:01.339   217   639 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 14, path =  0
01-02 00:01:01.339   217   639 D ACDB-LOADER: ACDB -> send_adm_topology
01-02 00:01:01.339   217   639 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
```

```
01-02 00:01:01.339   217   639 D ACDB-LOADER: ACDB -> send_audtable
01-02 00:01:01.339   217   639 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
01-02 00:01:01.339   217   639 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
01-02 00:01:01.339   217   639 D ACDB-LOADER: ACDB -> send_audvoltable
01-02 00:01:01.339   217   639 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
01-02 00:01:01.339   217   639 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
01-02 00:01:01.339   217   639 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
01-02 00:01:01.339   217   639 V listen_hal_loader:
audio_extn_listen_update_status(): no need to notify listen. device =
speaker. Event = 1
01-02 00:01:01.339  2361  2361 I Diag_Lib: Diag_LSM_Init call succeeded
01-02 00:01:01.349  2361  2361 D DiagService: onStart
01-02 00:01:01.349   217   639 V audio_hw_primary: enable_audio_route:
enter: usecase(4)
```

The FM digital radio use case is chosen, and mixer controls for play-fm are applied.

```
01-02 00:01:01.349   217   639 V audio_hw_primary: enable_audio_route:
apply mixer path: play-fm
01-02 00:01:01.349   217   639 V audio_hw_primary: enable_audio_route: exit
```

Corresponding PCM devices for FM are defined in hardware/qcom/audio/hal/msm8974/platform.h as FM_PLAYBACK_PCM_DEVICE (5) and FM_CAPTURE_PCM_DEVICE (6).

```
01-02 00:01:01.349   217   639 V audio_hw_fm: fm_start: FM PCM devices (rx:
5 tx: 6) for the usecase(4)
01-02 00:01:01.349   217   639 V audio_hw_fm: fm_start: Opening PCM
playback device card_id(0) device_id(5)
01-02 00:01:01.349   217   639 V audio_hw_fm: fm_start: Opening PCM capture
device card_id(0) device_id(6)
01-02 00:01:01.399   217   639 V audio_hw_fm: fm_set_volume: entry
01-02 00:01:01.399   217   639 D audio_hw_fm: fm_set_volume: (1.000000)
01-02 00:01:01.399   217   639 D audio_hw_fm: fm_set_volume: Setting FM
volume to 8192
01-02 00:01:01.399   217   639 V audio_hw_fm: fm_set_volume: exit
01-02 00:01:01.399   217   639 D audio_hw_fm: fm_start: exit: status(0)
01-02 00:01:01.399   217   639 V audio_hw_fm: audio_extn_fm_set_parameters:
exit
01-02 00:01:01.399   217   639 V listen_hal_loader:
audio_extn_listen_set_parameters: enter: handle_fm=524292
01-02 00:01:01.399   217   639 V listen_hw: listen_hw_set_parameters:
Enter, kvpairs = handle_fm=524292
01-02 00:01:01.399   217   639 V listen_hw: handle_set_parameters: Enter
kvpairs=handle_fm=524292.
```

```
01-02 00:01:01.399   217   639 V listen_hw: handle_set_parameters: Exit
01-02 00:01:01.399   217   639 V listen_hw: listen_hw_set_parameters: Exit,
ret=0
01-02 00:01:01.399   217   639 V audio_hw_primary: out_set_parameters:
exit: code(-2)
01-02 00:01:01.399   217  1145 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 0, device 4
01-02 00:01:01.409   217   639 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: fm_volume=0.0000000000
01-02 00:01:01.409   217   639 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-02 00:01:01.409   217   639 V audio_hw_fm: audio_extn_fm_set_parameters:
enter
01-02 00:01:01.409   217   639 D audio_hw_fm: audio_extn_fm_set_parameters:
set_fm_volume usecase
01-02 00:01:01.409   217   639 V audio_hw_fm: fm_set_volume: entry
01-02 00:01:01.409   217   639 D audio_hw_fm: fm_set_volume: (0.000000)
01-02 00:01:01.409   217   639 D audio_hw_fm: fm_set_volume: Setting FM
volume to 0
01-02 00:01:01.419   217   639 V audio_hw_fm: fm_set_volume: exit
01-02 00:01:01.419   217   639 V audio_hw_fm: audio_extn_fm_set_parameters:
exit
01-02 00:01:01.419   217   639 V listen_hal_loader:
audio_extn_listen_set_parameters: enter: fm_volume=0.0000000000
01-02 00:01:01.419   217   639 V listen_hw: listen_hw_set_parameters:
Enter, kvpairs = fm_volume=0.0000000000
01-02 00:01:01.419   217   639 V listen_hw: handle_set_parameters: Enter
kvpairs=fm_volume=0.0000000000.
01-02 00:01:01.419   217   639 V listen_hw: handle_set_parameters: Exit
01-02 00:01:01.419   217   639 V listen_hw: listen_hw_set_parameters: Exit,
ret=0
01-02 00:01:01.419   217   639 V audio_hw_primary: out_set_parameters:
exit: code(-2)
```

A headphone is chosen as the Rx device.

```
01-02 00:01:01.749   217   639 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: routing=4
01-02 00:01:01.749   217   639 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-02 00:01:01.749   217   639 V audio_hw_fm: audio_extn_fm_set_parameters:
enter
01-02 00:01:01.749   217   639 V msm8974_platform:
platform_get_output_snd_device: enter: output devices(0x4)
01-02 00:01:01.749   217   639 D audio_hw_extn: audio_extn_get_anc_enabled:
anc_enabled:0
01-02 00:01:01.749   217   639 V msm8974_platform:
platform_get_output_snd_device: exit: snd_device(headphones)
```

```
01-02 00:01:01.749   217   639 D audio_hw_primary: select_devices:
out_snd_device(4: headphones) in_snd_device(0: )
01-02 00:01:01.749   217   639 V audio_hw_primary: disable_audio_route:
enter: usecase(4)
01-02 00:01:01.749   217   639 V audio_hw_primary: disable_audio_route:
reset mixer path: play-fm
01-02 00:01:01.819   217   639 V audio_hw_primary: disable_audio_route:
exit
01-02 00:01:01.819   217   639 D hardware_info: hw_info_append_hw_type :
device_name = speaker
01-02 00:01:01.819   217   639 V audio_hw_primary: disable_snd_device:
snd_device(2: speaker)
01-02 00:01:01.819   217   639 V listen_hal_loader:
audio_extn_listen_update_status(): no need to notify listen. device =
speaker. Event = 0
```

Send the calibration data for ACDBI ID 10 for headphones.

```
01-02 00:01:01.819   217   639 D hardware_info: hw_info_append_hw_type :
device_name = headphones
01-02 00:01:01.819   217   639 V audio_hw_primary: enable_snd_device:
snd_device(4: headphones)
01-02 00:01:01.819   217   639 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 10, path =  0
01-02 00:01:01.819   217   639 D ACDB-LOADER: ACDB -> send_adm_topology
01-02 00:01:01.819   217   639 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
01-02 00:01:01.819   217   639 D ACDB-LOADER: ACDB -> send_audtable
01-02 00:01:01.819   217   639 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
01-02 00:01:01.819   217   639 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
01-02 00:01:01.819   217   639 D ACDB-LOADER: ACDB -> send_audvoltable
01-02 00:01:01.819   217   639 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
01-02 00:01:01.819   217   639 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
01-02 00:01:01.819   217   639 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
01-02 00:01:01.819   217   639 V listen_hal_loader:
audio_extn_listen_update_status(): no need to notify listen. device =
headphones. Event = 1
01-02 00:01:01.829   217   639 V audio_hw_primary: enable_audio_route:
enter: usecase(4)
01-02 00:01:01.829   217   639 V audio_hw_primary: enable_audio_route:
apply mixer path: play-fm
01-02 00:01:01.909   217   639 V audio_hw_primary: enable_audio_route: exit
01-02 00:01:01.909   217   639 V audio_hw_fm: audio_extn_fm_set_parameters:
exit
```

```
01-02 00:01:01.909   217   639 V listen_hal_loader:
audio_extn_listen_set_parameters: enter: routing=4
01-02 00:01:01.909   217   639 V listen_hw: listen_hw_set_parameters:
Enter, kvpairs = routing=4
01-02 00:01:01.909   217   639 V listen_hw: handle_set_parameters: Enter
kvpairs=routing=4.
01-02 00:01:01.909   217   639 V listen_hw: handle_set_parameters: Exit
01-02 00:01:01.909   217   639 V listen_hw: listen_hw_set_parameters: Exit,
ret=0
01-02 00:01:01.909   217   639 V audio_hw_primary: out_set_parameters:
exit: code(1)
01-02 00:01:01.909   217  1145 V AudioPolicyManager: checkAndSetVolume:
index 4 output 2 device 4
01-02 00:01:01.909   217  1145 V AudioPolicyManager: checkAndSetVolume()
for output 2 stream 0, volume 0.387468, delay 0
```

Turn on the FM module.

```
01-02 00:01:01.929  2189  2189 D FMService: Sending Recording intent for =
1
01-02 00:01:01.929  2189  2189 D FMService: mAudioManager.setFmRadioOn done
01-02 00:01:01.929  2189  2189 D FmRxRdsData: In rdsOn: RDS is true
01-02 00:01:01.929  2189  2189 E fmradio : id(800000f) value: 1
01-02 00:01:01.929  2189  2189 E fmradio : id(8000010)
01-02 00:01:01.929  2189  2189 D FmRxRdsData: In rdsOptions: rdsMask: 23
01-02 00:01:01.929  2189  2189 E fmradio : id(8000010) value: 38
01-02 00:01:01.929  2189  2189 E fmradio : id(8000014) value: 1
01-02 00:01:01.929  2189  2189 D FMService: registerRdsGroupProcessing
done, Status :true
01-02 00:01:01.929  2189  2189 D FMService: enableAutoAF: true
01-02 00:01:01.929  2189  2189 D FmRxRdsData: In rdsOn: RDS is true
01-02 00:01:01.929  2189  2189 E fmradio : id(800000f) value: 1
01-02 00:01:01.929  2189  2189 D FmRxRdsData: In enableAFjump: AFenable :
true
01-02 00:01:01.929  2189  2189 E fmradio : id(8000010)
01-02 00:01:01.929  2189  2189 D FmRxRdsData: Currently set
rds_group_mask : 120
01-02 00:01:01.929  2189  2189 E fmradio : id(800001b) value: 1
01-02 00:01:01.929  2189  2189 E fmradio : id(8000010)
01-02 00:01:01.929  2189  2189 D FmRxRdsData: After enabling the
rds_group_mask is : 248
01-02 00:01:01.929  2189  2189 D FMService: enableAutoAF done, Status :true
01-02 00:01:01.929  2189  2189 E fmradio : id(8000012) value: 0
01-02 00:01:01.929  2189  2189 D FMService: setInternalAntenna done,
Status :true
```

Set the FM volume.

```
01-02 00:01:01.929  2189  2189 E fmradio : id(8000012)
01-02 00:01:01.929  2189  2189 D FMService: getInternalAntenna: false
01-02 00:01:01.929  2189  2189 W ContextImpl: Calling a method in the
system process without a qualified user:
android.app.ContextImpl.sendBroadcast:1131
android.content.ContextWrapper.sendBroadcast:365
com.caf.fmradio.FMRadioService.sendRecordServiceIntent:731
com.caf.fmradio.FMRadioService.startFM:779
com.caf.fmradio.FMRadioService.fmOn:1606
01-02 00:01:01.939  2189  2189 D FMService: tuneRadio:  98.1
01-02 00:01:01.939  2189  2189 D FmRxControls: ** Tune Using: 54
01-02 00:01:01.989  2189  2189 D FmRxControls: ** Returned: 0
01-02 00:01:02.059   217   634 V AudioPolicyService: AudioCommandThread()
waking up
01-02 00:01:02.059   217   634 V AudioPolicyService: AudioCommandThread()
processing set parameters string fm_volume=0.1894530505, io 2
01-02 00:01:02.059   217   634 V AudioFlinger: setParameters(): io 2,
keyvalue fm_volume=0.1894530505, calling pid 217
01-02 00:01:02.059   217   634 V AudioFlinger: ThreadBase::setParameters()
fm_volume=0.1894530505
01-02 00:01:02.059   217   639 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: fm_volume=0.1894530505
01-02 00:01:02.059   217   639 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-02 00:01:02.059   217   639 V audio_hw_fm: audio_extn_fm_set_parameters:
enter
01-02 00:01:02.059   217   639 D audio_hw_fm: audio_extn_fm_set_parameters:
set_fm_volume usecase
01-02 00:01:02.059   217   639 V audio_hw_fm: fm_set_volume: entry
01-02 00:01:02.059   217   639 D audio_hw_fm: fm_set_volume: (0.189453)
01-02 00:01:02.059   217   639 D audio_hw_fm: fm_set_volume: Setting FM
volume to 1552
01-02 00:01:02.059   217   639 V audio_hw_fm: fm_set_volume: exit
01-02 00:01:02.059   217   639 V audio_hw_fm: audio_extn_fm_set_parameters:
exit
01-02 00:01:02.059   217   639 V listen_hal_loader:
audio_extn_listen_set_parameters: enter: fm_volume=0.1894530505
01-02 00:01:02.059   217   639 V listen_hw: listen_hw_set_parameters:
Enter, kvpairs = fm_volume=0.1894530505
01-02 00:01:02.059   217   639 V listen_hw: handle_set_parameters: Enter
kvpairs=fm_volume=0.1894530505.
01-02 00:01:02.059   217   639 V listen_hw: handle_set_parameters: Exit
01-02 00:01:02.059   217   639 V listen_hw: listen_hw_set_parameters: Exit,
ret=0
01-02 00:01:02.059   217   639 V audio_hw_primary: out_set_parameters:
exit: code(-2)
```

```
01-02 00:01:02.059   217   634 V AudioPolicyService: AudioCommandThread()
going to sleep
```

FM playback is activated.

```
01-02 00:01:02.069  2189  2408 D FMRadio : Received event. Count: 1
01-02 00:01:02.069  2189  2408 D FMRadio : Received <9>
01-02 00:01:02.069  2189  2408 D FMRadio : Got ABOVE_TH_EVENT
01-02 00:01:02.069  2189  2408 D FMService: FmRxEvServiceAvailable
01-02 00:01:02.069  2189  2408 D FMService: FmRxEvServiceAvailable: Tuned
frequency is above signal threshold level
01-02 00:01:02.069  2189  2408 D FMRadio : Received event. Count: 4
01-02 00:01:02.069  2189  2408 D FMRadio : Received <1>
01-02 00:01:02.069  2189  2408 D FMRadio : Got TUNE_EVENT
01-02 00:01:02.069  2189  2408 D FMService: FmRxEvRadioTuneStatus: Tuned
Frequency: 98100
01-02 00:01:02.069  2189  2408 D FmSharedPreferences: Save preferences
01-02 00:01:02.069  2189  2408 D FMRadio :
mServiceCallbacks.onTuneStatusChanged:
01-02 00:01:02.069  2189  2408 D FMService: enableStereo: true
```

The FM service is stopped.

```
01-02 00:01:07.629  2189  2189 D FMService: audioManager.setFmRadioOn =
false
01-02 00:01:07.629  2189  2189 D FMService: In stopFM
01-02 00:01:07.629  2189  2189 D FMService: FMRadio: Requesting to stop FM
01-02 00:01:07.629   217  1116 V AudioPolicyService:
setDeviceConnectionState()
```

A request to stop the device is sent.

```
01-02 00:01:07.629   217  1116 V AudioPolicyManager:
setDeviceConnectionState() device: 80000, state 0, address
01-02 00:01:07.629   217  1116 V AudioPolicyManager:
setDeviceConnectionState() disconnecting device 80000
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 5, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 5, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 1, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 1, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 2, device 6
```

```
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 2, device 6
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 3, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 3, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 0, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
from cache strategy 4, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 4, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 1, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 2, device 6
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 3, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 4, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 5, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager: getDeviceForStrategy()
strategy 0, device 4
01-02 00:01:07.629   217  1116 V AudioPolicyManager:
setDeviceConnectionState() setParameters handle_fm
01-02 00:01:07.629   217  1116 V AudioPolicyService: inserting command: 3
at index 0, num commands 0
01-02 00:01:07.629   217  1116 V AudioPolicyService: AudioCommandThread()
adding set parameter string handle_fm=4, io 2 ,delay 0
01-02 00:01:07.629   217   634 V AudioPolicyService: AudioCommandThread()
waking up
01-02 00:01:07.629   217   634 V AudioPolicyService: AudioCommandThread()
processing set parameters string handle_fm=4, io 2
01-02 00:01:07.629   217   634 V AudioFlinger: setParameters(): io 2,
keyvalue handle_fm=4, calling pid 217
```

```
01-02 00:01:07.629   217   634 V AudioFlinger: ThreadBase::setParameters()
handle_fm=4
01-02 00:01:07.629   217   639 V AudioFlinger: thread 0xb5bb2008 type 0 TID
639 waking up
01-02 00:01:07.629   217   639 V AudioFlinger: acquireWakeLock_l()
AudioOut_2 status 0
01-02 00:01:07.629   217   639 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: handle_fm=4
01-02 00:01:07.629   217   639 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-02 00:01:07.629   217   639 V audio_hw_fm: audio_extn_fm_set_parameters:
enter
```

Stop FM and reset mixer controls.

```
01-02 00:01:07.629   217   639 D audio_hw_fm: audio_extn_fm_set_parameters:
FM usecase
01-02 00:01:07.629   217   639 D audio_hw_fm: fm_stop: enter
01-02 00:01:07.699   217   639 V audio_hw_primary: disable_audio_route:
enter: usecase(4)
01-02 00:01:07.699   217   639 V audio_hw_primary: disable_audio_route:
reset mixer path: play-fm
01-02 00:01:07.709   217   639 V audio_hw_primary: disable_audio_route:
exit
01-02 00:01:07.709   217   639 D hardware_info: hw_info_append_hw_type :
device_name = headphones
```

Disable the headphone device.

```
01-02 00:01:07.709   217   639 V audio_hw_primary: disable_snd_device:
snd_device(4: headphones)
01-02 00:01:07.709   217   639 V listen_hal_loader:
audio_extn_listen_update_status(): no need to notify listen. device =
headphones. Event = 0
01-02 00:01:07.709   217   639 E audio_hw_primary: disable_snd_device:
Invalid sound device 0
01-02 00:01:07.709   217   639 D audio_hw_fm: fm_stop: exit: status(0)
01-02 00:01:07.709   217   639 V audio_hw_fm: audio_extn_fm_set_parameters:
exit
01-02 00:01:07.709   217   639 V listen_hal_loader:
audio_extn_listen_set_parameters: enter: handle_fm=4
01-02 00:01:07.709   217   639 V listen_hw: listen_hw_set_parameters:
Enter, kvpairs = handle_fm=4
01-02 00:01:07.709   217   639 V listen_hw: handle_set_parameters: Enter
kvpairs=handle_fm=4.
01-02 00:01:07.709   217   639 V listen_hw: handle_set_parameters: Exit
01-02 00:01:07.709   217   639 V listen_hw: listen_hw_set_parameters: Exit,
ret=0
```

```
01-02 00:01:07.709   217   639 V audio_hw_primary: out_set_parameters:
exit: code(-2)
01-02 00:01:07.709   217   639 V AudioFlinger: acquireWakeLock_l()
AudioOut_2 status 0
01-02 00:01:07.709   217   634 V AudioPolicyService: AudioCommandThread()
going to sleep
```

The state of the FM application is changed to FMTurningOff.

```
01-02 00:01:07.759  2189  2189 D FMService: audioManager.setFmRadioOn false
done
01-02 00:01:07.759  2189  2189 D FMService: Sending Recording intent for =
0
01-02 00:01:07.759  2189  2189 V FMRadio : disable: CURRENT-STATE : FMRxOn
---> NEW-STATE : FMTurningOff
01-02 00:01:07.759  2189  2189 E fmradio : id(8000004) value: 0
01-02 00:01:07.759  2189  2189 W ContextImpl: Calling a method in the
system process without a qualified user:
android.app.ContextImpl.sendBroadcast:1131
android.content.ContextWrapper.sendBroadcast:365
com.caf.fmradio.FMRadioService.sendRecordServiceIntent:731
com.caf.fmradio.FMRadioService.fmOperationsOff:1665
com.caf.fmradio.FMRadioService.fmOff:1711
```

The FM service is stopped.

```
01-02 00:01:07.769  2189  2189 D FMService: in stop
01-02 00:01:07.769  2189  2408 D FMRadio : Received event. Count: 1
01-02 00:01:07.769  2189  2408 D FMRadio : Received <18>
01-02 00:01:07.769  2189  2408 D FMRadio : Got RADIO_DISABLED
01-02 00:01:07.769  2189  2408 V FMRadio : RxEvtList: CURRENT-STATE :
FMTurningOff ---> NEW-STATE : FMOff
01-02 00:01:07.769  2189  2408 D FmTransceiver: Turned off: 0
01-02 00:01:07.769  2189  2408 D FMService: FmRxEvDisableReceiver
```

12.1.3.2 Kernel logs

Not required.

12.1.4 Customization

No customizations recommended.

12.1.5 Debugging

Table 12-1 Debugging and troubleshooting tips

Issue type	Symptoms	Troubleshooting steps
FM playback is not heard	No audio is routed to FM	▪ Check whether the headset is connected. Verify that the audio service receives the intent from the FM application to start the audio path by the audio service. The routing command should be correct and the codec setting should be proper. Sometimes, the FM volume is not set, and it may result in zero volume. ▪ Check whether the DSP is receiving data from the FM module in WCNSS by collecting the AFE PCM logs using QXDM Professional. See *Hexagon Access Audio/Voice PCM/Bit Stream Logging with QXDM Professional* (80-N3470-4) for QXDM Professional logging information. ▪ Ensure that kernel side AFE is configured for the loopback between WCNSS and codec.
FM playback is noisy	Playback is choppy	PCM logging using QXDM Professional logs as mentioned in *Hexagon Access Audio/Voice PCM/Bit Stream Logging with QXDM Professional* (80-N3470-4) helps in isolating the issue in the WCNSS or DSP. The kernel logs are analyzed in some cases to ensure that the sample rate setting is correct. Create a case in https://support.cdmatech.com to contact the Audio CE team for further analysis.

13 HDMI audio

13.1 Enable logs

13.1.1 Logcat logs

Clear the logs before reproducing any issue.

```
adb logcat –c
adb logcat –v threadtime | tee logcat.txt
```

The following command is also used before reproducing the issue.

```
adb logcat  -c  && adb logcat –v threadtime | tee logcat.txt
```

This command clears the current logcat logs and starts logging the user space logs to the logcat.txt file and the stdout simultaneously.

NOTE: For the tee command to work, Cygwin must be installed, or the command must be executed in a Linux environment.

13.1.2 Kernel logs

```
echo -n "file msm-dai-q6-hdmi-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file q6afe.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm-pcm-routing-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
```

13.1.3 QXDM Professional logging

See *Hexagon Access Audio/Voice PCM/Bit Stream Logging with QXDM Professional* (80-N3470-4) for QXDM Professional logging.

13.1.4 Additional logging

Depending on the issue, additional logging is enabled in the user space and kernel space.

In the user space, logs on the files AudioFlinger.cpp, AudioTrack.cpp, AudioRecord.cpp, AudioSource.cpp, audio_hw.c, platform.c, and so on, are enabled to receive additional logs.

On the kernel side, log messages in the q6afe.c and q6adm.c files are enabled to debug issues related to APSS and DSP communication.

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6afe.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file msm-pcm-routing-v2.c +p > /sys/kernel/debug/dynamic_debug/control
```

For calibration-related issues, log messages in audio_acdb.c are enabled.

```
echo file audio_acdb.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging on q6asm.c results in many messages that can affect system performance. Only the necessary logs are enabled depending on the issue.

13.2 Log analysis

13.2.1 Logcat logs – Plug in HDMI when phone is in idle state

This section is not applicable to this release.

13.2.2 Kernel logs – Plug in HDMI when phone is in idle state

```
<3>[ 1218.487558] hdmi_cec_isr: cec is not enabled. Just clear int and
return.
<3>[ 1218.508315] hdmi_cec_isr: cec is not enabled. Just clear int and
return.
<3>[ 1218.529388] hdmi_cec_isr: cec is not enabled. Just clear int and
return.
<6>[ 1218.535495] hdmi_edid_extract_3d_present: EDID: 3D present, 3D-len=8
<4>[ 1218.541992] hdmi_edid_find_block: EDID: type=7 block not found in
EDID block
<6>[ 1218.550330] hdmi_edid_read: V=1.3 #CEABlks=1[V3] ID=SEK IEEE=0c03
Ext=0x02
<6>[ 1218.560700] hdmi_tx_hpd_int_work: sense cable CONNECTED: state switch
to 1
<6>[ 1218.588448] power: ON (3840x2160 p30 16/9)
<6>[ 1218.590587] HDMI Audio: Enabled
```

```
<6>[ 1218.594201] hdmi_tx_set_audio_switch_node: hdmi_audio state switched
to 1
<6>[ 1218.594320] hdmi_tx_start: HDMI Core: Initialized
<6>[ 1218.594345] hdmi_tx_power_on: HDMI=ON DVI= OFF
```

The logs indicate that the HDMI core is initialized.

13.2.3 Logcat logs – Plug out HDMI when phone is in idle state

This section is not applicable to this release.

13.2.4 Kernel logs – Plug out HDMI when phone is in idle state

```
<3>[ 1090.606383] hdmi_cec_isr: cec is not enabled. Just clear int and
return.
<6>[ 1090.618124] hdmi_tx_set_audio_switch_node: hdmi_audio state switched
to 0
<6>[ 1090.624909] hdmi_tx_hpd_int_work: sense cable DISCONNECTED: state
switch to 0
<6>[ 1090.721243] HDMI Audio: Disabled
<6>[ 1090.723512] hdmi_tx_power_off_work: HDMI Core: OFF
```

These logs indicate that the HDMI core is turned off.

13.2.5 Logcat logs – Plug in HDMI when music is playing on speaker

This section is not applicable to this release.

13.2.6 Kernel logs – Plug in HDMI when music is playing on speaker

```
<7>[  965.004530] msm_pcm_routing_process_audio: reg 2 val 3 set 1
```

The PCM routing to BACKEND DAI ID 2 (MSM_BACKEND_DAI_SLIMBUS_0_RX,) from FRONTEND DAI ID 3 (MSM_FRONTEND_DAI_MULTIMEDIA4). The BACKEND DAI IDs and FRONTEND DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[  965.005482]  MSM8974 Compr: found 1 audio playback paths
<7>[  965.005506]  MSM8974 Compr:   connected new DSP playback path MSM8974
Compr -> SLIMBUS_0_RX
<7>[  965.005541]  MSM8974 Compr: found 1 new BEs
<7>[  965.005554]  SLIMBUS_0_RX: dpcm: open BE SLIMBUS_0_RX
<7>[  965.005575]  MSM8974 Compr: dpcm: open FE MSM8974 Compr
<6>[  965.005612] msm_compr_open: session ID 1
<7>[  965.012952]  SLIMBUS_0_RX: dpcm: hw_params BE MSM8974 Compr
```

```
<7>[  965.013040]  MSM8974 Compr: dpcm: hw_params FE MSM8974 Compr rate 
48000 chan 2 fmt 2
<7>[  965.020664]  MSM8974 Compr: dpcm: prepare FE MSM8974 Compr
<7>[  965.020677]  SLIMBUS_0_RX: dpcm: prepare BE MSM8974 Compr

<7>[  965.031117] afe_port_start: port id: 0x4000
```

Start the AFE port 0x4000 (SLIMBUS_0_RX). The port IDs are defined in kernel/include/sound/apr_audio-v2.h.

```
<7>[  965.031130] afe_q6_interface_prepare:
<7>[  965.031142] afe_send_cal
<7>[  965.031157] afe_send_cal_block: path 0
<7>[  965.031173] afe_send_cal_block: AFE cal sent for device port = 16384, 
path = 0, cal size = 1076, cal addr = 0x7df68000
<7>[  965.031612] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0 
size = 8
```

Opcode 0x110e8 (APR_BASIC_RSP_RESULT) defined in kernel/arch/arm/mach-msm/include/mach/qdsp6v2/apr.h for the command 0x100ef (AFE_PORT_CMD_SET_PARAM_V2) defined in kernel/include/sound/apr_audio-v2.h.

```
<7>[  965.031628] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0 
token=15
<7>[  965.031652] afe_callback:port_id = 0
<7>[  965.031693] afe_apr_send_pkt: leave 0
<7>[  965.031705] afe_send_cal_block: AFE cal sent for path 0 device!
<7>[  965.031718] afe_port_start: port_id 0x4000, mad_type 0
<7>[  965.032037] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0 
size = 8
<7>[  965.032052] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0 
token=15
<7>[  965.032075] afe_callback:port_id = 0
<7>[  965.032114] afe_apr_send_pkt: leave 0
<7>[  965.032126] afe_send_cmd_port_start: enter
<7>[  965.032139] afe_send_cmd_port_start: cmd device start opcode[0x100e5] 
port id[0x4000]
<7>[  965.038547] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0 
size = 8
<7>[  965.038562] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0 
token=15
<7>[  965.038585] afe_callback:port_id = 0
<7>[  965.038627] afe_apr_send_pkt: leave 0
<7>[  965.038639] task_name = mediaserver pid = 359
<7>[  965.038651] afe_send_cmd_port_start: leave 0
<7>[  965.159101] msm-pcm-routing msm-pcm-routing: reg 0
<7>[  965.159144] msm-pcm-routing msm-pcm-routing: reg 0 val 1
```

```
<7>[  965.160343]  SLIMBUS_0_RX: pm: BE SLIMBUS_0_RX stream MultiMedia4
Playback event 1 dir 0
<7>[  965.168455] msm_pcm_routing_process_audio: reg 2 val 4 set 1
<7>[  965.168644]  MSM8974 Compr: DPCM runtime update for FE MSM8974 Compr
<7>[  965.168849]  MSM8974 Compr: found 1 audio playback paths
<7>[  965.168908]  MSM8974 Compr: found 0 new BEs
<7>[  965.168966]  MSM8974 Compr: found 0 old BEs
<7>[  965.169153]  MSM8974 Compr: found 0 audio capture paths
<7>[  965.169188]  MSM8974 Compr: found 0 new BEs
<7>[  965.169218]  MSM8974 Compr: found 0 old BEs
<7>[  965.169993]  MSM8974 LowLatency: found 1 audio playback paths
<7>[  965.170055]  MSM8974 LowLatency:   connected new DSP playback path
MSM8974 LowLatency -> SLIMBUS_0_RX
<7>[  965.170151]  MSM8974 LowLatency: found 1 new BEs
<7>[  965.170191]  MSM8974 LowLatency: dpcm: open FE MSM8974 LowLatency
<7>[  965.170597]  MSM8974 LowLatency: dpcm: hw_params FE MSM8974
LowLatency rate 48000 chan 2 fmt 2
<7>[  965.185930]  MSM8974 LowLatency: dpcm: prepare FE MSM8974 LowLatency
<7>[  965.188476] msm-pcm-routing msm-pcm-routing: reg 0
<7>[  965.188518] msm-pcm-routing msm-pcm-routing: reg 0 val 1
<7>[  965.189912]  SLIMBUS_0_RX: pm: BE SLIMBUS_0_RX stream MultiMedia5
Playback event 1 dir 0
<7>[  965.190009]  SLIMBUS_0_RX: dpcm: trigger BE MSM8974 LowLatency cmd 1
<7>[  965.190049]  MSM8974 LowLatency: dpcm: post trigger FE MSM8974
LowLatency cmd 1
<7>[  965.221207]  MSM8974 Compr: dpcm: post trigger FE MSM8974 Compr cmd 1
<7>[  968.334036]  MSM8974 LowLatency: dpcm: post trigger FE MSM8974
LowLatency cmd 0
<7>[  968.334061]  MSM8974 LowLatency: dpcm: hw_free FE MSM8974 LowLatency
<7>[  968.334070]  MSM8974 LowLatency: dpcm: close FE MSM8974 LowLatency
<7>[  968.402064] msm-pcm-routing msm-pcm-routing: reg 0
<7>[  968.402228]  SLIMBUS_0_RX: pm: BE SLIMBUS_0_RX stream MultiMedia5
Playback event 2 dir 0
<7>[  968.402244]  MSM8974 LowLatency: BE playback disconnect check for
SLIMBUS_0_RX
<7>[  968.402258]  MSM8974 LowLatency:   freed DSP playback path MSM8974
LowLatency -> SLIMBUS_0_RX
<7>[  968.402273]  MSM8974 LowLatency:   reparent playback path MSM8974
Compr -> SLIMBUS_0_RX
<7>[  968.402694] msm_pcm_routing_process_audio: reg 2 val 4 set 0
<7>[  968.402754]  MSM8974 Compr: DPCM runtime update for FE MSM8974 Compr
<7>[  968.402820]  MSM8974 Compr: found 1 audio playback paths
<7>[  968.402843]  MSM8974 Compr: found 0 new BEs
<7>[  968.402863]  MSM8974 Compr: found 0 old BEs
<7>[  968.402922]  MSM8974 Compr: found 0 audio capture paths
<7>[  968.402934]  MSM8974 Compr: found 0 new BEs
<7>[  968.402944]  MSM8974 Compr: found 0 old BEs
```

```
<3>[  969.402097] hdmi_cec_isr: cec is not enabled. Just clear int and
return.
<3>[  969.423379] hdmi_cec_isr: cec is not enabled. Just clear int and
return.
<3>[  969.444376] hdmi_cec_isr: cec is not enabled. Just clear int and
return.
<6>[  969.450548] hdmi_edid_extract_3d_present: EDID: 3D present, 3D-len=8
<4>[  969.459884] hdmi_edid_find_block: EDID: type=7 block not found in
EDID block
<6>[  969.466851] hdmi_edid_read: V=1.3 #CEABlks=1[V3] ID=SEK IEEE=0c03
Ext=0x02
<6>[  969.477014] hdmi_tx_hpd_int_work: sense cable CONNECTED: state switch
to 1

<6>[  969.492029] power: ON (3840x2160 p30 16/9)
<6>[  969.494884] HDMI Audio: Enabled
<6>[  969.495084] hdmi_tx_set_audio_switch_node: hdmi_audio state switched
to 1
<6>[  969.495145] hdmi_tx_start: HDMI Core: Initialized
<6>[  969.495150] hdmi_tx_power_on: HDMI=ON DVI= OFF
```

The logs indicate that the HDMI cable plug-in was detected and the HDMI core was turned on.

```
<7>[  969.779595] msm_pcm_routing_process_audio: reg 2 val 3 set 0
```

Disable the routing to BACKEND DAI ID 2 (MSM_BACKEND_DAI_SLIMBUS_0_RX,) from FRONTEND DAI ID 3 (MSM_FRONTEND_DAI_MULTIMEDIA4).

```
<7>[  969.784299]  MSM8974 Compr: DPCM runtime update for FE MSM8974 Compr
<7>[  969.784362]  MSM8974 Compr: found 0 audio playback paths
<7>[  969.784369]  MSM8974 Compr: found 0 new BEs
<7>[  969.784382]  MSM8974 Compr: pruning playback BE SLIMBUS_0_RX for
MSM8974 Compr
<7>[  969.784388]  MSM8974 Compr: found 1 old BEs
<7>[  969.784396]  MSM8974 Compr: runtime playback close on FE MSM8974
Compr
<7>[  969.784403]  MSM8974 Compr: dpcm: trigger FE MSM8974 Compr cmd stop
<7>[  969.784422]  SLIMBUS_0_RX: dpcm: trigger BE MSM8974 Compr cmd 0
<7>[  969.784429]  SLIMBUS_0_RX: dpcm: hw_free BE MSM8974 Compr
<7>[  969.784438]  SLIMBUS_0_RX: dpcm: close BE MSM8974 Compr

<7>[  969.784448] afe_close: port_id=16384

<7>[  969.784456] afe_close: port_id 0x4000, mad_type 0
```

Close the AFE port SLIMBUS_0_RX.

```
<7>[  969.784462] afe_close: Not a MAD port
<7>[  969.791668] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
size = 8
<7>[  969.791677] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
token=15
<7>[  969.791689] afe_callback:port_id = 0
<7>[  969.791715] afe_apr_send_pkt: leave 0
<7>[  969.797121] msm-pcm-routing msm-pcm-routing: reg 0
<7>[  969.803560]  SLIMBUS_0_RX: pm: BE SLIMBUS_0_RX stream MultiMedia4
Playback event 0 dir 0
<7>[  969.803570]  MSM8974 Compr: BE playback disconnect check for
SLIMBUS_0_RX
<7>[  969.803577]  MSM8974 Compr:   freed DSP playback path MSM8974 Compr -
> SLIMBUS_0_RX
<7>[  969.803632]  MSM8974 Compr: found 0 audio capture paths
<7>[  969.803639]  MSM8974 Compr: found 0 new BEs
<7>[  969.803646]  MSM8974 Compr: found 0 old BEs
<7>[  969.809409] msm_pcm_routing_process_audio: reg 4 val 3 set 1
```

The data routing is to BACKEND DAI ID 4 (MSM_BACKEND_DAI_HDMI_RX) from the FRONTEND DAI ID 3 (MSM_FRONTEND_DAI_MULTIMEDIA4). The BACKEND DAI IDs and FRONTEND DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[  969.809483]  MSM8974 Compr: DPCM runtime update for FE MSM8974 Compr
<7>[  969.809549]  MSM8974 Compr: found 1 audio playback paths
<7>[  969.809588]  MSM8974 Compr:   connected new DSP playback path MSM8974
Compr -> HDMI
<7>[  969.809623]  MSM8974 Compr: found 1 new BEs
<7>[  969.809634]  MSM8974 Compr: runtime playback open on FE MSM8974 Compr
<7>[  969.809645]  HDMI: dpcm: open BE HDMI
<7>[  969.809663]  HDMI: dpcm: hw_params BE MSM8974 Compr
<7>[  969.809699] msm-dai-q6-hdmi msm-dai-q6-hdmi.8:
msm_dai_q6_hdmi_hw_params() minor version: 1 samplerate: 48000 bitwidth: 16
<7>[  969.809705] num_ch = 2 channel_allocation = 0 datatype = 0
<7>[  969.809720]  HDMI: dpcm: prepare BE MSM8974 Compr

<7>[  969.816723] afe_port_start: port id: 0x8
```

Start the AFE port 0x8 (HDMI_RX). The port IDs are defined in kernel/include/sound/apr_audio-v2.h.

```
<7>[  969.816745] afe_q6_interface_prepare:
<7>[  969.816760] afe_send_cal
<7>[  969.816781] afe_send_cal_block: path 0
<7>[  969.816794] afe_send_cal_block: No AFE cal to send!
```

```
<7>[  969.816807] afe_port_get_mad_type: Non Slimbus port_id 0x8
<7>[  969.816820] afe_port_start: port_id 0x8, mad_type 0
<7>[  969.817262] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
size = 8
<7>[  969.817284] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
token=8
<7>[  969.817340] afe_callback:port_id = 0
<7>[  969.817420] afe_apr_send_pkt: leave 0
<7>[  969.817434] afe_send_cmd_port_start: enter
<7>[  969.817452] afe_send_cmd_port_start: cmd device start opcode[0x100e5]
port id[0x100e]
<7>[  969.824579] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0
size = 8
<7>[  969.824600] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0
token=8
<7>[  969.824631] afe_callback:port_id = 0
<7>[  969.824679] afe_apr_send_pkt: leave 0
<7>[  969.824692] task_name = ApmCommand pid = 671
<7>[  969.824708] afe_send_cmd_port_start: leave 0
<7>[  969.825605] msm-pcm-routing msm-pcm-routing: reg 0
<7>[  969.825624] msm-pcm-routing msm-pcm-routing: reg 0 val 1
<7>[  969.826242]  HDMI: pm: BE HDMI stream MultiMedia4 Playback event 0
dir 0
<7>[  969.826263]  MSM8974 Compr: dpcm: trigger FE MSM8974 Compr cmd start
<7>[  969.826280]  HDMI: dpcm: trigger BE MSM8974 Compr cmd 1
<7>[  969.826299]  MSM8974 Compr: BE playback disconnect check for HDMI
<7>[  969.826324]  MSM8974 Compr: found 0 old BEs
<7>[  969.826408]  MSM8974 Compr: found 0 audio capture paths
<7>[  969.826424]  MSM8974 Compr: found 0 new BEs
<7>[  969.826438]  MSM8974 Compr: found 0 old BEs
<7>[  977.395637]  HDMI: dpcm: trigger BE MSM8974 Compr cmd 3
<7>[  977.395646]  MSM8974 Compr: dpcm: post trigger FE MSM8974 Compr cmd 3
<7>[  977.569009] msm_pcm_routing_process_audio: reg 4 val 4 set 1
```

Enable the data routing to BACKEND DAI ID 4 (MSM_BACKEND_DAI_HDMI_RX) from FRONTEND DAI ID 4 (MSM_FRONTEND_DAI_MULTIMEDIA5). The BACKEND DAI IDs and FRONTEND DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[  977.569258]  MSM8974 Compr: DPCM runtime update for FE MSM8974 Compr
<7>[  977.569524]  MSM8974 Compr: found 1 audio playback paths
<7>[  977.569590]  MSM8974 Compr: found 0 new BEs
<7>[  977.569643]  MSM8974 Compr: found 0 old BEs
<7>[  977.569828]  MSM8974 Compr: found 0 audio capture paths
<7>[  977.569863]  MSM8974 Compr: found 0 new BEs
<7>[  977.569894]  MSM8974 Compr: found 0 old BEs
<7>[  977.572061]  MSM8974 LowLatency: found 1 audio playback paths
```

```
<7>[  977.572139]  MSM8974 LowLatency:   connected new DSP playback path
MSM8974 LowLatency -> HDMI
<7>[  977.572238]  MSM8974 LowLatency: found 1 new BEs
<7>[  977.572279]  MSM8974 LowLatency: dpcm: open FE MSM8974 LowLatency
<7>[  977.572673]  MSM8974 LowLatency: dpcm: hw_params FE MSM8974
LowLatency rate 48000 chan 2 fmt 2
<7>[  977.587179]  MSM8974 LowLatency: dpcm: prepare FE MSM8974 LowLatency
<7>[  977.589721] msm-pcm-routing msm-pcm-routing: reg 0
<7>[  977.589764] msm-pcm-routing msm-pcm-routing: reg 0 val 1
<7>[  977.591962]  HDMI: pm: BE HDMI stream MultiMedia5 Playback event 1
dir 0
<7>[  977.592061]  MSM8974 LowLatency: dpcm: post trigger FE MSM8974
LowLatency cmd 1
<7>[  980.399675]  MSM8974 Compr: dpcm: prepare FE MSM8974 Compr
<7>[  980.403016]  HDMI: pm: BE HDMI stream MultiMedia4 Playback event 1
dir 0
<7>[  980.413711]  MSM8974 Compr: dpcm: prepare FE MSM8974 Compr
<7>[  980.419684]  HDMI: pm: BE HDMI stream MultiMedia4 Playback event 1
dir 0
<7>[  980.421865]  MSM8974 Compr: dpcm: hw_free FE MSM8974 Compr
<7>[  980.421938]  MSM8974 Compr: dpcm: hw_free FE MSM8974 Compr
<7>[  980.421984]  MSM8974 Compr: dpcm: close FE MSM8974 Compr
<7>[  980.469575] msm-pcm-routing msm-pcm-routing: reg 0
<7>[  980.469670]  HDMI: pm: BE HDMI stream MultiMedia4 Playback event 2
dir 0
<7>[  980.469678]  MSM8974 Compr: BE playback disconnect check for HDMI
<7>[  980.469684]  MSM8974 Compr:   freed DSP playback path MSM8974 Compr -
> HDMI
<7>[  980.469690]  MSM8974 Compr:   reparent playback path MSM8974
LowLatency -> HDMI

<7>[  980.469850] msm_pcm_routing_process_audio: reg 4 val 3 set 0
```

Disable the data routing to BACKEND DAI ID 4 (MSM_BACKEND_DAI_HDMI_RX) from FRONTEND DAI ID 3 (MSM_FRONTEND_DAI_MULTIMEDIA4). The BACKEND DAI IDs and FRONTEND DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[  980.469883]  MSM8974 LowLatency: DPCM runtime update for FE MSM8974
LowLatency
<7>[  980.469917]  MSM8974 LowLatency: found 1 audio playback paths
<7>[  980.469932]  MSM8974 LowLatency: found 0 new BEs
<7>[  980.469941]  MSM8974 LowLatency: found 0 old BEs
<7>[  980.469971]  MSM8974 LowLatency: found 0 audio capture paths
<7>[  980.469976]  MSM8974 LowLatency: found 0 new BEs
<7>[  980.469981]  MSM8974 LowLatency: found 0 old BEs
<7>[  980.741304]  MSM8974 LowLatency: dpcm: post trigger FE MSM8974
LowLatency cmd 0
```

```
<7>[  980.741330]  MSM8974 LowLatency: dpcm: hw_free FE MSM8974 LowLatency
<7>[  980.741338]  HDMI: dpcm: hw_free BE MSM8974 LowLatency
<7>[  980.741347]  MSM8974 LowLatency: dpcm: close FE MSM8974 LowLatency
<7>[  980.809008]  HDMI: dpcm: close BE MSM8974 LowLatency
<7>[  980.809017] afe_close: port_id=8
```

Close the AFE port 0x8 (HDMI_RX). The port IDs are defined in kernel/include/sound/apr_audio-v2.h.

```
<7>[  980.809023] afe_port_get_mad_type: Non Slimbus port_id 0x8
<7>[  980.809029] afe_close: port_id 0x8, mad_type 0
<7>[  980.809034] afe_close: Not a MAD port
<7>[  980.844201] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
size = 8
<7>[  980.844208] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
token=8
<7>[  980.844219] afe_callback:port_id = 0
<7>[  980.844238] afe_apr_send_pkt: leave 0
<7>[  980.844244] msm_dai_q6_hdmi_shutdown: dai_data->status_mask = 1
<7>[  980.845102] msm-pcm-routing msm-pcm-routing: reg 0
<7>[  980.845295]  HDMI: pm: BE HDMI stream MultiMedia5 Playback event 2
dir 0
<7>[  980.845303]  MSM8974 LowLatency: BE playback disconnect check for
HDMI
<7>[  980.845309]  MSM8974 LowLatency:   freed DSP playback path MSM8974
LowLatency -> HDMI

<7>[  980.845526] msm_pcm_routing_process_audio: reg 4 val 4 set 0
```

Disable the data routing to BACKEND DAI ID 4 (MSM_BACKEND_DAI_HDMI_RX) from FRONTEND DAI ID 4 (MSM_FRONTEND_DAI_MULTIMEDIA5). The BACKEND DAI IDs and FRONTEND DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

13.2.7 Logcat logs – Plug out HDMI when music is playing

This section is not applicable to this release.

13.2.8 Kernel logs – Plug out HDMI when music is playing

```
<7>[ 1331.303823] msm_pcm_routing_process_audio: reg 4 val 3 set 1
<7>[ 1331.304249]  MSM8974 Compr: found 1 audio playback paths
<7>[ 1331.304262]  MSM8974 Compr:   connected new DSP playback path MSM8974
Compr -> HDMI
<7>[ 1331.304279]  MSM8974 Compr: found 1 new BEs
<7>[ 1331.304286]  HDMI: dpcm: open BE HDMI
<7>[ 1331.304296]  MSM8974 Compr: dpcm: open FE MSM8974 Compr
```

```
<6>[ 1331.304315] msm_compr_open: session ID 1
<7>[ 1331.308418]  HDMI: dpcm: hw_params BE MSM8974 Compr
<7>[ 1331.308446] msm-dai-q6-hdmi msm-dai-q6-hdmi.8:
msm_dai_q6_hdmi_hw_params() minor version: 1 samplerate: 48000 bitwidth: 16
<7>[ 1331.308449] num_ch = 2 channel_allocation = 0 datatype = 0
<7>[ 1331.308458]  MSM8974 Compr: dpcm: hw_params FE MSM8974 Compr rate
48000 chan 2 fmt 2
<7>[ 1331.315225]  MSM8974 Compr: dpcm: prepare FE MSM8974 Compr
<7>[ 1331.315231]  HDMI: dpcm: prepare BE MSM8974 Compr

<7>[ 1331.321776] afe_port_start: port id: 0x8
```

Start the AFE port 0x8 (HDMI_RX). The port IDs are defined in kernel/include/sound/apr_audio-v2.h.

```
<7>[ 1331.321781] afe_q6_interface_prepare:
<7>[ 1331.321786] afe_send_cal
<7>[ 1331.321793] afe_send_cal_block: path 0
<7>[ 1331.321798] afe_send_cal_block: No AFE cal to send!
<7>[ 1331.321804] afe_port_get_mad_type: Non Slimbus port_id 0x8
<7>[ 1331.321809] afe_port_start: port_id 0x8, mad_type 0
<7>[ 1331.322159] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
size = 8
<7>[ 1331.322166] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
token=8
<7>[ 1331.322176] afe_callback:port_id = 0
<7>[ 1331.322197] afe_apr_send_pkt: leave 0
<7>[ 1331.322202] afe_send_cmd_port_start: enter
<7>[ 1331.322207] afe_send_cmd_port_start: cmd device start opcode[0x100e5]
port id[0x100e]
<7>[ 1331.329409] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0
size = 8
<7>[ 1331.329416] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0
token=8
<7>[ 1331.329425] afe_callback:port_id = 0
<7>[ 1331.329444] afe_apr_send_pkt: leave 0
<7>[ 1331.329449] task_name = mediaserver pid = 359
<7>[ 1331.329454] afe_send_cmd_port_start: leave 0
<7>[ 1331.331043] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1331.331050] msm-pcm-routing msm-pcm-routing: reg 0 val 1
<7>[ 1331.331241]  HDMI: pm: BE HDMI stream MultiMedia4 Playback event 1
dir 0
<7>[ 1331.333453] msm_pcm_routing_process_audio: reg 4 val 4 set 1
```

Enable the data routing to BACKEND DAI ID 4 (MSM_BACKEND_DAI_HDMI_RX) from FRONTEND DAI ID 4 (MSM_FRONTEND_DAI_MULTIMEDIA5). The BACKEND DAI IDs and FRONTEND DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[ 1331.333484]  MSM8974 Compr: DPCM runtime update for FE MSM8974 Compr
<7>[ 1331.333516]  MSM8974 Compr: found 1 audio playback paths
 . . .
<7>[ 1331.333961]  MSM8974 LowLatency: found 1 audio playback paths
<7>[ 1331.333971]  MSM8974 LowLatency:   connected new DSP playback path
MSM8974 LowLatency -> HDMI
<7>[ 1331.333986]  MSM8974 LowLatency: found 1 new BEs
<7>[ 1331.333992]  MSM8974 LowLatency: dpcm: open FE MSM8974 LowLatency
<7>[ 1331.334054]  MSM8974 LowLatency: dpcm: hw_params FE MSM8974
LowLatency rate 48000 chan 2 fmt 2
<7>[ 1331.348054]  MSM8974 LowLatency: dpcm: prepare FE MSM8974 LowLatency
<7>[ 1331.351530] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1331.351551] msm-pcm-routing msm-pcm-routing: reg 0 val 1
<7>[ 1331.354706]  HDMI: pm: BE HDMI stream MultiMedia5 Playback event 1
dir 0
<7>[ 1331.354773]  HDMI: dpcm: trigger BE MSM8974 LowLatency cmd 1
<7>[ 1331.354794]  MSM8974 LowLatency: dpcm: post trigger FE MSM8974
LowLatency cmd 1
<7>[ 1331.382100]  MSM8974 Compr: dpcm: post trigger FE MSM8974 Compr cmd 1
<7>[ 1334.502646]  MSM8974 LowLatency: dpcm: post trigger FE MSM8974
LowLatency cmd 0
<7>[ 1334.502720]  MSM8974 LowLatency: dpcm: hw_free FE MSM8974 LowLatency
<7>[ 1334.502744]  MSM8974 LowLatency: dpcm: close FE MSM8974 LowLatency
<7>[ 1334.571496] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1334.572026]  HDMI: pm: BE HDMI stream MultiMedia5 Playback event 2
dir 0
<7>[ 1334.572049]  MSM8974 LowLatency: BE playback disconnect check for
HDMI
<7>[ 1334.572068]  MSM8974 LowLatency:   freed DSP playback path MSM8974
LowLatency -> HDMI
<7>[ 1334.572088]  MSM8974 LowLatency:   reparent playback path MSM8974
Compr -> HDMI
<7>[ 1334.572791] msm_pcm_routing_process_audio: reg 4 val 4 set 0
 . . .
<3>[ 1349.815099] hdmi_cec_isr: cec is not enabled. Just clear int and
return.

<6>[ 1349.821000] hdmi_tx_set_audio_switch_node: hdmi_audio state switched
to 0
```

The HDMI audio state is switched to 0.

```
<7>[ 1349.896102]  HDMI: dpcm: trigger BE MSM8974 Compr cmd 3
<7>[ 1349.896112]  MSM8974 Compr: dpcm: post trigger FE MSM8974 Compr cmd 3
<7>[ 1352.898486]  MSM8974 Compr: dpcm: prepare FE MSM8974 Compr
<7>[ 1352.901681]  HDMI: pm: BE HDMI stream MultiMedia4 Playback event 1
dir 0
<7>[ 1352.919196]  MSM8974 Compr: dpcm: prepare FE MSM8974 Compr
<7>[ 1352.927396]  HDMI: pm: BE HDMI stream MultiMedia4 Playback event 1
dir 0
<7>[ 1352.932096]  MSM8974 Compr: dpcm: hw_free FE MSM8974 Compr
<7>[ 1352.932147]  HDMI: dpcm: hw_free BE MSM8974 Compr
<7>[ 1352.932215]  MSM8974 Compr: dpcm: hw_free FE MSM8974 Compr
<7>[ 1352.932257]  HDMI: dpcm: hw_free BE MSM8974 Compr
<7>[ 1352.932300]  MSM8974 Compr: dpcm: close FE MSM8974 Compr
<7>[ 1352.981053]  HDMI: dpcm: close BE MSM8974 Compr

<7>[ 1352.981101] afe_close: port_id=8
```

Close the AFE port 0x8 (HDMI_RX). The port IDs are defined in kernel/include/sound/apr_audio-v2.h.

```
<7>[ 1352.981135] afe_port_get_mad_type: Non Slimbus port_id 0x8
<7>[ 1352.981169] afe_close: port_id 0x8, mad_type 0
<7>[ 1352.981198] afe_close: Not a MAD port
<7>[ 1353.013780] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
size = 8
<7>[ 1353.013822] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
token=8
<7>[ 1353.013885] afe_callback:port_id = 0
<7>[ 1353.013981] afe_apr_send_pkt: leave 0
<7>[ 1353.014015] msm_dai_q6_hdmi_shutdown: dai_data->status_mask = 1
<6>[ 1353.018069] hdmi_tx_hpd_int_work: sense cable DISCONNECTED: state
switch to 0
<7>[ 1353.092514] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1353.092911]  HDMI: pm: BE HDMI stream MultiMedia4 Playback event 2
dir 0
<7>[ 1353.092927]  MSM8974 Compr: BE playback disconnect check for HDMI
<7>[ 1353.092941]  MSM8974 Compr:   freed DSP playback path MSM8974 Compr -
> HDMI
<7>[ 1353.096787] msm_pcm_routing_process_audio: reg 4 val 3 set 0
```

Disable the data routing to BACKEND DAI ID 4 (MSM_BACKEND_DAI_HDMI_RX) from FRONTEND DAI ID 3 (MSM_FRONTEND_DAI_MULTIMEDIA4). The BACKEND DAI IDs and FRONTEND DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<6>[ 1353.152767] HDMI Audio: Disabled
<6>[ 1353.155030] hdmi_tx_power_off_work: HDMI Core: OFF
```

The HDMI core is turned off.

13.3 Customization

This section is not applicable to this release.

13.4 Debugging

This section is not applicable to this release.

14 WFD

Figure 14-1 shows the paths taken by the control and data flows through the user space, kernel, and DSP components involved in a WFD use case on Android.

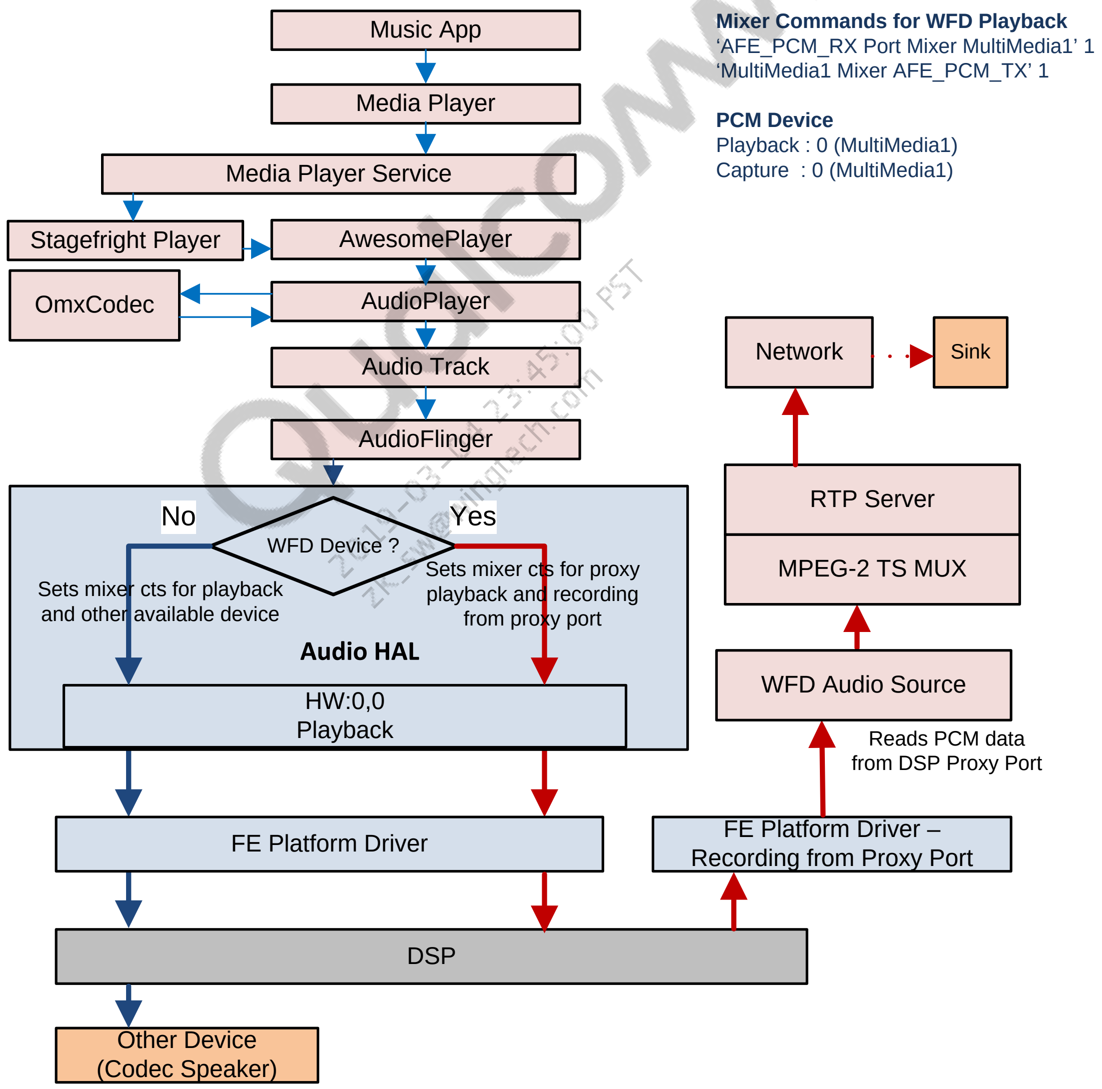

Music App
Media Player
Media Player Service
Stagefright Player
AwesomePlayer
OmxCodec
AudioPlayer
Audio Track
AudioFlinger
No
WFD Device ?
Yes
Sets mixer cts for playback and other available device
Sets mixer cts for proxy playback and recording from proxy port
Audio HAL
HW:0,0 Playback
FE Platform Driver
DSP
Other Device (Codec Speaker)
Mixer Commands for WFD Playback
'AFE_PCM_RX Port Mixer MultiMedia1' 1
'MultiMedia1 Mixer AFE_PCM_TX' 1
PCM Device
Playback : 0 (MultiMedia1)
Capture : 0 (MultiMedia1)
Network
Sink
RTP Server
MPEG-2 TS MUX
WFD Audio Source
Reads PCM data from DSP Proxy Port
FE Platform Driver – Recording from Proxy Port

Figure 14-1 Components involved in media WFD playback on Android

WFD is an interoperable mechanism to pair, connect, and render multimedia content sourced from a WFD source (source) at a WFD sink (sink). Devices with QTI SoCs act as source and are paired and connected to HDMI TVs using commercial wireless display adapters or other devices that can act as sinks.

When in WFD mode, audio is decoded in software only in the user space. The decoded PCM audio samples are then routed from HAL to the proxy port in the DSP, from which the WFD stack reads them. If no audio is played at the source, silence is read from the proxy port. Along with the raw LPCM audio data read from the proxy port (also encoded to AAC/AC3 based on the configuration), video content from the primary display panel (source) is duplicated and encoded as H.264. Audio and video are then multiplexed and sent to the sink over Wi-Fi (a P2P/RTSP connection is already established between the source and sink).

Supported audio type

- LPCM – Mandatory configuration (LPCM 48 kHz, 16-bit, 2 channels), all others optional
- AAC (48 kHz, 16-bit, 2 channels, 5.1 playback)
- AC3

14.1.1 Log collection

This section highlights the various files and/or components in which logging is enabled to troubleshoot issues related to audio playback on Android.

14.1.1.1 User space logs

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

14.1.1.2 Kernel logs

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file msm-pcm-q6-v2.c +p > /sys/kernel/debug/dynamic_debug/control
echo file msm-pcm-routing-v2.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging on msm-pcm-q6-v2.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

14.1.1.3 Additional logging

Depending on the issue, additional logging is enabled in the user space and kernel space. In the user space, enable logs on files listed at the following locations:

```
\frameworks\av\media\libmediaplayerservice
MediaPlayerService.cpp
StagefrightPlayer.cpp

\frameworks\av\media\libmedia
AudioSystem.cpp
AudioTrack.cpp
mediaplayer.cpp
```

```
IOMX.cpp
```

\frameworks\av\media\libstagefright
```
AwesomePlayer.cpp
AudioPlayer.cpp
FileSource.cpp
MPEG4Extractor.cpp
OMXClient.cpp
OMXCodec.cpp
```

\frameworks\av\services\audioflinger
```
AudioFlinger.cpp
AudioMixer.cpp
AudioPolicyService.cpp
```

\harwdware\qcom\audio\hal
```
audio_hw.c
```

\harwdware\qcom\audio\hal\msm
```
platform.c
hw_info.c
```

\harwdware\qcom\audio\hal\policy_hal
```
AudioPolicyManager.cpp
```

\harwdware\libhardware_legacy\audio
```
AudioPolicyManagerBase.cpp
```

On the kernel side, enable logs on the following files to debug issues related to APSS and DSP communication:

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6afe.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6asm.c  +p > /sys/kernel/debug/dynamic_debug/control
```

For calibration-related issues, enable log messages in audio_acdb.c.

```
echo file audio_acdb.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging on q6asm.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

14.1.2 Log analysis

14.1.2.1 User space logs

Audio playback is initiated from the application layer through the MediaPlayer service. MediaPlayer in turn uses the Stagefright framework to fulfill the actual playout. StagefrightPlayer instantiates AwesomePlayer in the playback scenario.

```
01-02 00:01:47.989   192  1072 V MediaPlayerService: Client(5) constructor
01-02 00:01:47.989  3456  3456 V MediaPlayer: setDataSource(64, 0,
576460752303423487)
01-02 00:01:47.989   192  1104 V StagefrightPlayer: StagefrightPlayer
01-02 00:01:47.989   192  1104 V AudioSink: AudioOutput(20)
01-02 00:01:47.989   192  1104 V StagefrightPlayer: setDataSource(35, 0,
4067983500)
```

Once MediaPlayer, StagefrightPlayer, and AwesomePlayer instances are created, the input file data is extracted and a MediaSource is created from it.

```
01-02 00:01:47.989   192  1104 V MPEG4Extractor: found metadata size:
1619507
01-02 00:01:48.009   192  1104 V MediaExtractor: Autodetected media content
as 'video/mp4' with confidence 0.40
01-02 00:01:48.009   192  1104 V MPEG4Extractor: entering parseChunk 0/0
01-02 00:01:48.009   192  1104 V MPEG4Extractor: chunk: ftyp @ 0, 0
01-02 00:01:48.009   192  1104 V MPEG4Extractor: entering parseChunk 24/0
01-02 00:01:48.009   192  1104 V MPEG4Extractor: chunk: moov @ 24, 0
01-02 00:01:48.009   192  1104 V MPEG4Extractor: entering parseChunk 32/1
01-02 00:01:48.009   192  1104 V MPEG4Extractor: chunk: mvhd @ 32, 1
.
.
.
1-02 00:01:48.919   192  1104 V MPEG4Extractor: entering parseChunk
1555147/5
01-02 00:01:48.919   192  1104 V MPEG4Extractor: chunk: stco @ 1555147, 5

01-02 00:01:48.919   192  1104 V AwesomePlayer: mBitrate = -1 bits/sec

01-02 00:01:48.919   192  1104 V MediaPlayerService:  setDataSource
01-02 00:01:48.919  3456  3456 V MediaPlayer: setVideoSurfaceTexture
01-02 00:01:48.919   192  1103 V MediaPlayerService: [5]
setVideoSurfaceTexture(0xb8acea48)
01-02 00:01:48.919   192  1103 V StagefrightPlayer: setVideoSurfaceTexture
01-02 00:01:48.919  3456  3456 V MediaPlayer:
MediaPlayer::setAudioStreamType
01-02 00:01:48.919  3456  3456 V MediaPlayer: setVideoSurfaceTexture
01-02 00:01:48.919   192   192 V MediaPlayerService: [5]
setVideoSurfaceTexture(0xb8aa26d0)
```

```
01-02 00:01:48.919  3456  3456 V MediaPlayer: prepareAsync
```

The prepare() API is then invoked, which instantiates the video decoder used to decode the compressed video. In this case, an MPEG-4 file is to be played out, so the software MPEG-3 decoder provided by Google is instantiated. Input and output buffers for the decoder are also allocated as part of its creation, and the decoder is started.

```
01-02 00:01:48.919   192  3563 V OMXCodec: matching
'OMX.qcom.video.decoder.mpeg4' quirks 0x000000a8
01-02 00:01:48.919   192  3563 V OMXCodec: Attempting to allocate OMX node
'OMX.qcom.video.decoder.mpeg4'
01-02 00:01:48.919   192  3563 V OMXCodec: Successfully allocated OMX node
'OMX.qcom.video.decoder.mpeg4'

01-02 00:01:48.929   192  3563 I OMXCodec: [OMX.qcom.video.decoder.mpeg4]
video dimensions are 1920 x 1088
01-02 00:01:48.929   192  3563 I OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Crop rect is 1920 x 1088 @ (0, 0)
01-02 00:01:48.929   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
allocating 4 buffers of size 1384448 on input port
01-02 00:01:48.929   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
allocated buffer 0xb8acb1e0 on input port
01-02 00:01:48.929   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
allocated buffer 0xb8acb230 on input port
01-02 00:01:48.929   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
allocated buffer 0xb8acb280 on input port
01-02 00:01:48.929   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
allocated buffer 0xb8acb2d0 on input port
01-02 00:01:48.939   192  3563 I OMXCodec: NOTE: Allocating 8 buffers on
output port
01-02 00:01:48.939   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
allocating 8 buffers from a native window of size 3137536 on output port
01-02 00:01:48.949   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
registered graphic buffer with ID 0xb8acf470 (pointer = 0xb8acf260)
01-02 00:01:48.949   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
registered graphic buffer with ID 0xb8acf4c0 (pointer = 0xb8acfce8)
01-02 00:01:48.959   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
registered graphic buffer with ID 0xb8acf510 (pointer = 0xb8acfe48)
01-02 00:01:48.959   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
registered graphic buffer with ID 0xb8acf560 (pointer = 0xb8ad0008)
01-02 00:01:48.979   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
registered graphic buffer with ID 0xb8acf5b0 (pointer = 0xb8ac96e8)

01-02 00:01:49.049   192  3565 V OMX     : OnEvent(0, 0, 2)
01-02 00:01:49.049   192  3566 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
onStateChange 2
01-02 00:01:49.049   192  3566 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Now Idle.
```

The decoder moves into the Executing state and waits for its buffers to be filled by the source.

```
01-02 00:01:49.049   192  3565 V OMX     : OnEvent(0, 0, 3)
01-02 00:01:49.049   192  3566 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
onStateChange 3
01-02 00:01:49.049   192  3566 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Now Executing.
```

The prepare() API is then invoked, which instantiates the decoder used to decode the compressed media. In this case, an AAC file is to be played out, so it follows the default compressed offload path.

```
01-02 00:01:49.049   192  3563 V AudioPolicyManagerBase:
isOffloadSupported: has_video == true, property                set to
enable offload
01-02 00:01:49.049   192  3563 V AudioPolicyManagerBase:
isOffloadSupported() profile found
01-02 00:01:49.049   192  3563 V OMXCodec: matching
'OMX.google.aac.decoder' quirks 0x00000000
01-02 00:01:49.049   192  3563 V OMXCodec: Attempting to allocate OMX node
'OMX.google.aac.decoder'
01-02 00:01:49.049   192  3563 V OMXCodec: Successfully allocated OMX node
'OMX.google.aac.decoder'
```

AwesomePlayer then creates an instance of AudioPlayer. The prepare sequence is complete.

```
01-02 00:01:49.049   192  3563 V AwesomePlayer: createAudioPlayer: bypass
OMX (offload)
01-02 00:01:49.049  3456  3472 V MediaPlayer: prepared
```

Once initialized, the start () API is invoked, which kicks off media playback. StagefrightPlayer creates an instance of AudioPlayer and starts it to perform the actual playout of media.

```
01-02 00:01:49.149  3456  3456 V MediaPlayer: start
01-02 00:01:49.149   192  1104 V StagefrightPlayer: start
```

StagefrightPlayer through AwesomePlayer provides data to and reads data from the video decoder buffers.

```
01-02 00:01:49.159   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Calling emptyBuffer on buffer 0xb8acb230 (length 22600), timestamp 0 us
(0.00 secs)
 01-02 00:01:49.159   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Calling emptyBuffer on buffer 0xb8acb280 (length 1028), timestamp 33333 us
(0.03 secs)
```

```
01-02 00:01:49.179   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Calling emptyBuffer on buffer 0xb8acb2d0 (length 1028), timestamp 66666 us
(0.07 secs)
01-02 00:01:49.179   192  3566 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Calling emptyBuffer on buffer 0xb8acb1e0 (length 1028), timestamp 100000 us
(0.10 secs)

01-02 00:01:49.179   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Calling fillBuffer on buffer 0xb8acf470
01-02 00:01:49.179   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Calling fillBuffer on buffer 0xb8acf4c0
01-02 00:01:49.179   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Calling fillBuffer on buffer 0xb8acf510
01-02 00:01:49.179   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Calling fillBuffer on buffer 0xb8acf560
01-02 00:01:49.179   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Calling fillBuffer on buffer 0xb8acf5b0
01-02 00:01:49.179   192  3563 V OMXCodec: [OMX.qcom.video.decoder.mpeg4]
Calling fillBuffer on buffer 0xb8acf600
```

The AudioSink created previously is opened, which instantiates an AudioTrack instance and adds it to the mixer.

```
01-02 00:01:49.259   192  3563 V AudioSink: open(48000, 2, 0x0, 0x4000000,
4, 20 0x10)
01-02 00:01:49.259   192  3563 V AudioSink: creating new AudioTrack
01-02 00:01:49.259   192  3563 V AudioPolicyService: getOutput()
01-02 00:01:49.259   192  3563 V AudioPolicyManagerBase: getOutput() device
262144, stream 3, samplingRate 48000, format 4000000, channelMask 3, flags
11
01-02 00:01:49.259   192  3563 V AudioFlinger: openOutput(), module 1
Device 40000, SamplingRate 48000, Format 0x4000000, Channels 3, flags 31

01-02 00:01:49.259   192  3563 V AudioFlinger: openOutput()
openOutputStream returned output 0xb8accef8, SamplingRate 48000, Format
0x4000000, Channels 3, status 0
01-02 00:01:49.259   192  3563 I AudioFlinger: HAL output buffer size 32768
frames, normal mix buffer size 32768 frames
01-02 00:01:49.259   192  3563 V AudioFlinger: openOutput() created offload
output: ID 22 thread 0xb8b1a150
```

Once AudioSink is opened, it starts and initiates audio path setup to the actual device where playback is needed.

```
01-02 00:01:49.269   192  3563 V AudioSink: start
01-02 00:01:49.269   192  3563 V AudioFlinger: start(0), calling pid 192
session 20
```

```
01-02 00:01:49.269   192  3563 V AudioPolicyService: startOutput()
01-02 00:01:49.269   192  3563 V AudioPolicyManagerBase: startOutput()
output 22, stream 3, session 20
01-02 00:01:49.269   192  3563 V AudioPolicyManagerBase: getNewDevice()
selected device 40000
01-02 00:01:49.269   192  3563 V AudioPolicyManagerBase: setOutputDevice()
output 22 device 40000 delayMs 0
```

The output path for playback is opened through Audio HAL. The device is AFE proxy-defined in /hardware/qcom/audio/hal/msm8974/platform.c. The use case (3) is USECASE_AUDIO_ PLAYBACK_COMPRESS_OFFLOAD defined in hardware/qcom/audio/hal/audio_hw.h.

```
01-02 00:01:49.289   192  3584 V audio_hw_primary: start_output_stream:
enter: usecase(3: compress-offload-playback) devices(0x40000)
01-02 00:01:49.289   192  3584 V msm8974_platform:
platform_get_output_snd_device: enter: output devices(0x40000)
01-02 00:01:49.289   192  3584 D msm8974_platform:
platform_get_output_snd_device: setting sink capability for Proxy
01-02 00:01:49.289   192  3584 D audio_hw_extn:
audio_extn_set_afe_proxy_channel_mixer: channels = 2
01-02 00:01:49.289   192  3584 V msm8974_platform:
platform_get_output_snd_device: exit: snd_device(afe-proxy)
01-02 00:01:49.289   192  3584 D audio_hw_primary: select_devices:
out_snd_device(16: afe-proxy) in_snd_device(0: )
01-02 00:01:49.289   192  3584 D hardware_info: hw_info_append_hw_type :
device_name = afe-proxy
01-02 00:01:49.289   192  3584 V audio_hw_primary: enable_snd_device:
snd_device(16: afe-proxy) is already active
01-02 00:01:49.289   192  3584 V audio_hw_primary: enable_audio_route:
enter: usecase(3)
01-02 00:01:49.289   192  3584 V audio_hw_primary: enable_audio_route:
apply mixer path: compress-offload-playback afe-proxy
01-02 00:01:49.289   192  3584 V audio_hw_primary: enable_audio_route: exit
```

card_id (0) is SOUND_CARD and device_id (0) is PLAYBACK_OFFLOAD_DEVICE defined in hardware/qcom/audio/hal/msm8974/platform.h.

```
01-02 00:01:49.289   192  3584 V audio_hw_primary: start_output_stream:
Opening PCM device card_id(0) device_id(9)
```

The end-to-end path is set up for both video and audio. Video is decoded in software and audio data is pumped to the DSP. Decoded PCM data from the proxy port is sent to the WFD stack.

14.1.2.2 Kernel logs

```
<7>[  109.044404] afe_register_get_events: port_id: 240
<7>[  109.044446] afe_apr_send_pkt: leave 0
<7>[  109.044458] afe_port_start: before incrementing proxy_afe_instance 0
port_id 240
<7>[  109.044469] afe_port_start: port id: 0x2000
<7>[  109.044720] afe_callback:opcode = 0x110e8 cmd = 0x100e0 status = 0x0
size = 8
<7>[  109.044733] afe_callback:opcode = 0x110e8 cmd = 0x100e0 status = 0x0
token=572662306
<7>[  109.044744] afe_callback:port_id = 0
<7>[  109.046189] afe_q6_interface_prepare:
<7>[  109.046197] afe_send_cal
<7>[  109.046207] afe_send_cal_block: path 0
<7>[  109.046214] afe_send_cal_block: No AFE cal to send!
<7>[  109.046220] afe_send_hw_delay
<7>[  109.046228] afe_send_hw_delay: Failed to get hw delay info
<7>[  109.046237] afe_send_hw_delay port_id 8192 rate 48000 delay_usec
572662306 status -22
<7>[  109.046246] afe_port_get_mad_type: Non Slimbus port_id 0x2000
<7>[  109.046254] afe_port_start: port_id 0x2000, mad_type 0
```

The open AFE port 0x2000 (AFE_PORT_ID_RT_PROXY_PORT_001_RX) for the AFE proxy port opens the WFD sink device to read. 0x2000 (AFE_PORT_ID_RT_PROXY_PORT_001_RX) is defined in /kernel/include/sound/apr_audio-v2.h to be opened.

```
<7>[  124.971767] msm_pcm_routing_process_audio: reg 9 val 3 set 1
```

The PCM routing is to BE DAI ID 9 (MSM_BACKEND_DAI_AFE_PCM_RX) from FE DAI ID 3 (MSM_FRONTEND_DAI_MULTIMEDIA4). The BE DAI and FE DAI IDs are mentioned in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[  124.973016] msm_compr_open
<7>[  124.973064] msm_compr_open: session ID 2
<7>[  124.973328] msm_compr_get_caps
<7>[  124.973367] msm_compr_set_params
<7>[  124.973380] msm_compr_set_params: sample_rate 48000
<7>[  124.973392] SND_AUDIOCODEC_AAC
<7>[  124.973405] msm_compr_configure_dsp
```

```
<7>[  124.982522] compr_event_handler opcode =000110e8
<7>[  124.982559] msm_compr_configure_dsp be_id 3
<7>[  124.982581] adm_open: port 0x2000 path:1 rate:48000 mode:2
perf_mode:0
<7>[  124.982593] adm_open: Port ID 0x2000, index 32
<7>[  124.982607] send_adm_custom_topology: no cal to send addr= 0x3e373c
<7>[  124.982616] adm_open:opening ADM: perf_mode: 0
<7>[  124.982627] adm_open: port_id=0x2000 rate=48000 topology_id=0x10313
```

The following log indicates that four buffers are allocated, each of size 256 kB. The code snippet shows the msm_pcm_hardware structure that defines buffer size, period size, supported sample rate, supported number of channels, and supported ALSA PCM parameters.

```
<7>[  124.982630] q6asm_audio_client_buf_alloc_contiguous:
session[2]bufsz[245760]bufcnt[4]
<7>[  124.982630] q6asm_audio_client_buf_alloc_contiguous
data[f0e3c000]phys[3f2ac000][cdf5071c]
<7>[  124.982630] q6asm_audio_client_buf_alloc_contiguous
data[f0e78000]phys[3f2e8000][cdf50738]
<7>[  124.982630] q6asm_audio_client_buf_alloc_contiguous
data[f0eb4000]phys[3f324000][cdf50754]
<7>[  124.982630] q6asm_memory_map_regions: Session[2]

<7>[  124.982630] mmap_region=0xe3a36440 token=0x200
<7>[  124.982630] map_regions->nregions = 1
<7>[  124.982630]
q6asm_mmapcallback:ptr0[0xf0993f88]ptr1[0x0]opcode[0x10d93]
token[0x200]payload_s[4] src[0] dest[0]sid[2]dir[0]
<7>[  124.982630] q6asm_mmapcallback:Payload = [0xf0993f88] status[0x0]
<7>[  124.982630] q6asm_mmapcallback:PL#0[0xf0993f88]PL#1 [0x0] dir=0
s_id=2
<7>[  124.982630] compr_event_handler opcode =00010d93
```

Allocate memory for four buffers, each of size 256 kB.

```
<7>[  124.982630] q6asm_memory_map_regions: i=0, bufadd[i] = 0x3f270000,
maphdl[i] = 0xf0993f88
<7>[  124.982630] q6asm_memory_map_regions: i=1, bufadd[i] = 0x3f2ac000,
maphdl[i] = 0xf0993f88
<7>[  124.982630] q6asm_memory_map_regions: i=2, bufadd[i] = 0x3f2e8000,
maphdl[i] = 0xf0993f88
<7>[  124.982630] q6asm_memory_map_regions: i=3, bufadd[i] = 0x3f324000,
maphdl[i] = 0xf0993f88
<7>[  124.982630] q6asm_memory_map_regions: exit
<7>[  124.982630] msm_compr_hw_params: buf[cdf50700]dma_buf-
>area[f0e00000]dma_buf->addr[3f270000]
```

A prepare() is called to pass in the number of channels, bit rate, and so on.

```
<7>[  124.982630] compressed stream prepare
<7>[  124.982630] q6asm_set_io_mode ac->mode after anding with FF00:0x[0],
<7>[  124.982630] q6asm_set_io_mode:Set Mode to 0x[42]
<7>[  124.982630] __q6asm_media_format_block_pcm:session[2]rate[44100]ch[2]
<7>[  124.982630] q6asm_add_hdr:pkt_size=44 cmd_flg=1 session=2
<7>[  124.982630] q6asm_map_channels channels passed: 2
```

Playback is started by calling msm_pcm_trigger().

```
<7>[  125.024981] msm_compr_trigger: SNDRV_PCM_TRIGGER_START
<7>[  125.025000] pkt_size = 32, cmd_flg = 1, session = 2
<7>[  125.025000] q6asm_callback: nowait_cmd_cnt 1
<7>[  125.025000] q6asm_callback: session[2]opcode[0x110e8]
token[0x2]payload_s[8] src[513] dest[513]
<7>[  125.025000] q6asm_callback:Payload = [0x10daa] status[0x0]
<7>[  125.025000] q6asm_callback:Payload = [0x10daa]
<7>[  125.025000] q6asm_callback:Payload = [0x10daa]stat[0x0]
<7>[  125.025000] compr_event_handler opcode =000110e8
<7>[  125.025000] compr_event_handler:writing 245760 bytes of buffer[0] to
dsp
<7>[  125.025000] compr_event_handler:writing buffer[0] from 0x3f270000
<7>[  125.025000] meta_data_length: 64, frame_length: 180558
<7>[  125.025000] timestamp_msw: 0, timestamp_lsw: 0
<7>[  125.025000] pkt_size = 52, cmd_flg = 0, session = 2
```

ASM_DATA_EVENT_WRITE_DONE is triggered when the DSP consumes the buffer and the kernel wakes up the user space to send more data.

```
<7>[  125.025000] ASM_DATA_EVENT_WRITE_DONE
<7>[  125.025000] Buffer Consumed = 0xc0b05d08
<7>[  125.025000] q6asm_callback: session[1]opcode[0x10d99]
token[0x1]payload_s[16] src[257] dest[257]
<7>[  125.025000] q6asm_callback:Payload = [0x3f26e400] status[0x0]
<7>[  125.025000] q6asm_callback: Rxed opcode[0x3f26e400] status[0x0]
token[1]
<7>[  125.025000] q6asm_is_cpu_buf_avail:session[1]index[1]
data[f03d4400]size[1024]
<7>[  125.025000] q6asm_write: session[1] len=1024
<7>[  125.025000] q6asm_add_hdr:pkt_size=52 cmd_flg=0 session=1
<7>[  125.025000] q6asm_write:ab->phys[0x3f26e400]bufadd[0x3f26e400]
token[0x1]buf_id[0x1]buf_size[0x400]mmaphdl[0xf0988e58]
<7>[  125.025000] q6asm_write: WRITE SUCCESS
<7>[  125.025000] q6asm_callback: session[2]opcode[0x110e8]
token[0x2]payload_s[8] src[513] dest[513]
<7>[  125.025000] q6asm_callback:Payload = [0x10bce] status[0x0]
```

```
<7>[  125.025000] q6asm_callback:Payload = [0x10bce]
<7>[  125.025000] q6asm_callback:Payload = [0x10bce]stat[0x0]
```

Playback is paused, which causes the DSP to flush its buffers.

```
<7>[  262.563937] msm_compr_trigger: SNDRV_PCM_TRIGGER_STOP

<7>[  239.628296] q6asm_cmd:CMD_FLUSH
<7>[  239.628313] q6asm_cmd:session[2]opcode[0x10bce]
<7>[  239.629030] q6asm_callback: session[2]opcode[0x10d99]
token[0x3f270040]payload_s[16] src[513] dest[513]
<7>[  262.624105] ASM_STREAM_CMD_FLUSH
<7>[  239.640329] q6asm_cmd:session[2]opcode[0x10bcd]
<7>[  239.643869] q6asm_callback: session[1]opcode[0x10d99]
token[0x3]payload_s[16] src[257] dest[257]
```

Finally, the ASM session is closed, which releases the DSP buffers and completes playout.

```
<7>[  239.640255] msm_compr_playback_close
<7>[  239.640313] q6asm_cmd:CMD_CLOSE

<7>[  239.640329] q6asm_cmd:session[2]opcode[0x10bcd]
<7>[  239.645766] q6asm_callback: session[2]opcode[0x110e8]
token[0x2]payload_s[8] src[513] dest[513]
<7>[  239.646013] q6asm_memory_unmap: Session[2]
<7>[  239.646033] q6asm_add_mmaphdr:pkt size=24 cmd_flg=1
<7>[  239.646051] q6asm_memory_unmap: Found the element
<7>[  239.646068] q6asm_memory_unmap: mem_unmap-mem_map_handle: 0xf0993f88
<7>[  239.647780] q6asm_mmapcallback:ptr0[0x10d94]ptr1[0x0]opcode[0x110e8]
token[0x200]payload_s[8] src[0] dest[0]sid[2]dir[0]
<7>[  239.647806] q6asm_mmapcallback:Payload = [0x10d94] status[0x0]
<7>[  239.647840] q6asm_mmapcallback:Payload = [0x10d94] status[0x0]
<7>[  239.648024]
q6asm_audio_client_buf_free_contiguous:data[f0e00000]phys[3f270000][cdf5070
0] , client[cdf50380] handle[e3a36540]
<7>[  262.661340] afe_close: port_id=224
<7>[  262.661364] afe_close: before decrementing pcm_afe_instance 1
<7>[  262.661407] adm_close port_id=0x2000 index 32 perf_mode: 0
<7>[  262.661420] adm_close:Closing ADM: perf_mode: 0
<7>[  262.661437] adm_close:coppid 1 portid=0x2000 index=32 coppcnt=0
<7>[  239.648828] q6asm_callback: session[1]opcode[0x10d99]
token[0x4]payload_s[16] src[257] dest[257]
<7>[  239.648853] q6asm_callback:Payload = [0x3f26f000] status[0x0]
<7>[  239.648875] q6asm_callback: Rxed opcode[0x3f26f000] status[0x0]
token[4]
<7>[  239.653176] q6asm_audio_client_free: Session id 2
<7>[  239.653196] q6asm_session_free: sessionid[2]
```

```
<7>[  239.653217] q6asm_audio_client_free: APR De-Register
<7>[  239.653607] msm-pcm-routing msm-pcm-routing: reg 0
<7>[  262.683501] msm_pcm_routing_process_audio: reg 9 val 3 set 0
```

14.1.3 Customization

There is no customization.

14.1.4 Debugging

Table 14-1 shows common WFD issues that were encountered during previous integration activities and some troubleshooting steps to resolve them.

Table 14-1 Debugging and troubleshooting tips

Issue type	Symptoms	Troubleshooting steps
AV sync issues	Video and audio playback are not in sync	Handled by the WFD team, check logs on WFD framework to determine whether audio is leading/lagging.
Device switching issues	Audio is not coming out from WFD sink device as expected	Check logs on AudioPolicyManager and Audio HAL
Audio mute issues	There is no audio heard in the playback	Proxy dump; performed in the WFD stack
Volume issues	Volume up and volume down do not work as expected	Check logs on AudioPolicyManager and Audio HAL; also need proxy dump in WFD stack
Noise issues	Noise heard in the WFD sink device	Proxy dump; performed in the WFD stack
Error in reading from proxy	Choppy audio sound in the WFD sink device	Check Kernel logs, user space AudioPolicyManager, and Audio HAL logs to determine routing

15 Voice and VoLTE call

15.1 Architecture overview

In voice and VoLTE use cases, the APSS is involved only in the device control path. All data is sent directly from the DSP to the Modem Processor Subsystem (MPSS) without APSS intervention.

On the voice call side, Multimode Vocoder Services (MVS) APIs are used for voice call use cases and the IP Multimedia System (IMS) stack is used for VoLTE use cases.

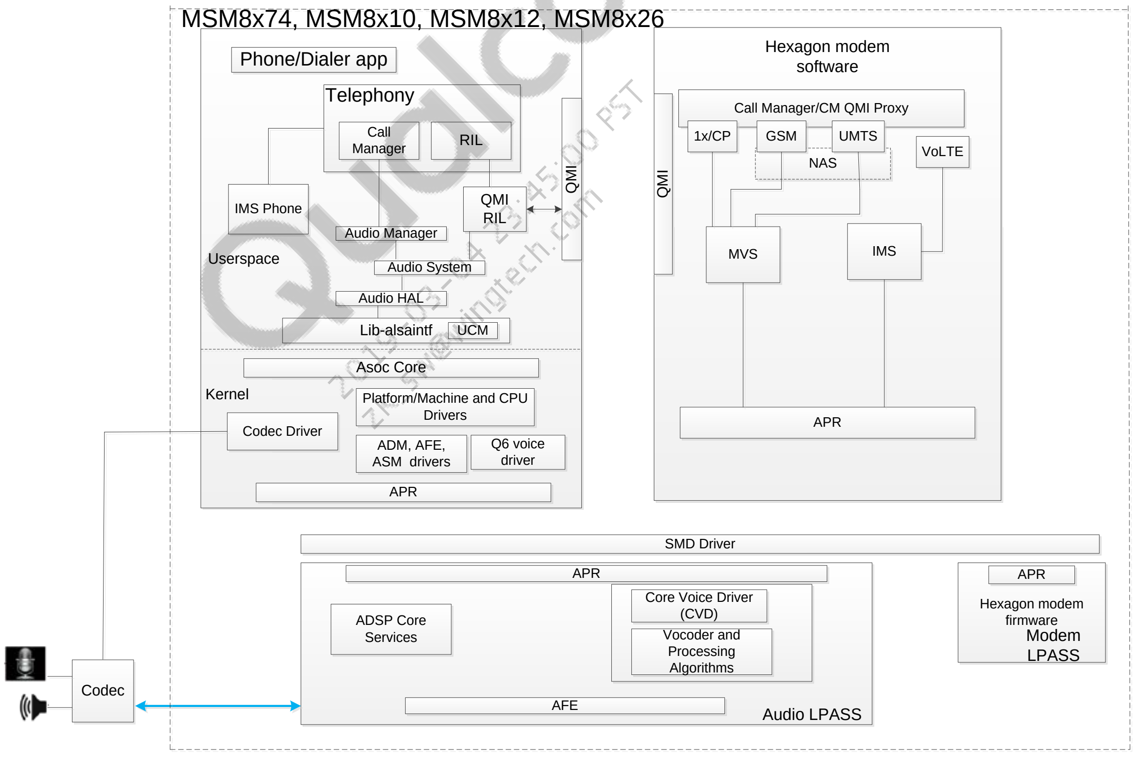


Figure 15-1 Voice and VoLTE architecture

The QCRIL module interfaces with the MPSS and exchanges Qualcomm Modem Interface (QMI) messages during an MO or MT voice call.

For a call to be established, the MPSS is responsible for connection and packet exchange over the network. It also informs the QCRIL layer on the APSS about the different states during the call establishment or teardown.

On the APSS side, the Call Manager module calls the setMode(MODE_INCALL) API in the AudioManager to indicate to the audio layer about setup of a voice/VoLTE call. Similarly, it calls setMode(MODE_NORMAL) in the AudioManager to indicate to the audio layer about teardown of voice/VoLTE layer.

The QCRIL has the information from the Radio Interface Layer (RIL) in the Android telephony about whether the call is voice or VoLTE. The QCRIL calls the AudioSytem API AudioSystem::setParameters() and passes the following information:

- Call state information indicating the state to which the voice/VoLTE session should transition
- Session ID of the voice/VoLTE call

The function qcril_am_call_audio_api(uint32 audio_vsid, uint32 audio_call_state) in the file vendor/qcom/proprietary/qcril/qcril_qmi/qcril_am.cc calls AudioSystem::setParameters(0, keyValPairs) keyValPairs has the format “vsid=voice_vsid;call_state=state,” where:

- vsid is the key and voice_vsid is a unique session number indicating a VoLTE, Voice, or QChat session
- call_state is the key and state is a number indicating CALL_ACTIVE, CALL_INACTIVE, CALL HOLD, and CALL_LOCAL_HOLD
- In the audio layer, the session IDs for VoLTE, voice, and Qchat are defined in the /hardware/qcom/audio/hal/voice_extn/voice_extn.c and /hardware/qcom/audio/hal/voice_extn.h files

```
#define VOICE_VSID 0x10C01000
#define VOICE2_VSID 0x10DC1000
#define VOLTE_VSID 0x10C02000
#define QCHAT_VSID 0x10803000
#define ALL_VSID 0xFFFFFFFF
```

Call states are defined as:

```
#define BASE_CALL_STATE 1
#define CALL_INACTIVE (BASE_CALL_STATE)
#define CALL_ACTIVE (BASE_CALL_STATE + 1)
#define CALL_HOLD (BASE_CALL_STATE + 2)
#define CALL_LOCAL_HOLD (BASE_CALL_STATE + 3)
```

- The unique session ID for a VoLTE/voice call is passed from QCRIL to the audio layer by calling AudioSystem::setParameters(). Eventually, the control flows through the APIs.

```
 static int adev_set_parameters(struct audio_hw_device *dev, const char
*kvpairs)
 |
 int voice_set_parameters(struct audio_device *adev, struct str_parms
*parms)
 |
 int voice_extn_set_parameters(struct audio_device *adev,struct str_parms
*parms)
```

```
 |
 static int update_call_states(struct audio_device *adev,const uint32_t
vsid, const int call_state)
 |
 static int update_calls(struct audio_device *adev)
```

- update_calls() has the logic to start_call(), stop_call(), put the call on hold, and update call states for the particular session whose session ID is passed to it.
- start_call() is used to set up the hostless PCM playback and capture sessions for the use case passed to it.
- stop_call() is used to tear down the hostless PCM playback and capture sessions for the use case passed to it.
- The use case mentioned contains the following values:

  [USECASE_VOICE_CALL] = "voice-call",

  [USECASE_VOICE2_CALL] = "voice2-call",

  [USECASE_VOLTE_CALL] = "volte-call",

  [USECASE_QCHAT_CALL] = "qchat-call“
- The mixer controls for setting up the data path in DSP for a voice/VoLTE call are listed in the file device/qcom/<chipset>/mixer_paths.xml.
- The function enable_audio_route() sets the mixer controls for voice/VoLTE use cases. The mixer controls for VoLTE invoke functions in the kernel, which in turn sends commands to DSP to set up the data path.
- The voice driver, implemented in the file msm-pcm-voice-v2.c in the kernel, is responsible for sending the necessary commands to the DSP for managing the voice/VoLTE session.
- The string “default volte voice”/“default modem voice” is passed as part of the payload of the command sent from APSS to the DSP voice module to indicate whether the commands are specific for a voice/VoLTE session.

15.2 Call flow

This section is not applicable to this release.

15.3 Collecting logs

15.3.1 Kernel logs

```
adb root
adb shell
echo -n "file msm-pcm-voice-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file msm-pcm-voip-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file q6voice.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm-pcm-routing-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
```

15.3.2 Logcat logs

The following command clears the current logcat logs and starts logging the user space logs to the file logcat.txt and the stdout simultaneously.

```
adb logcat –c  && adb logcat –v threadtime  | tee logcat.txt
```

NOTE: For the tee command to work, Cygwin must be installed or the command must be executed in a Linux environment.

15.3.3 Radio logs

This command starts logging the radio logs to the file radio and to the stdout simultaneously.

```
adb logcat -b radio -v threadtime | tee  radio.txt
```

NOTE: For the tee command to work, Cygwin must be installed or the command must be executed in a Linux environment.

15.3.4 QXDM Professional logs

See *Hexagon Access Audio/Voice PCM/Bit Stream Logging with QXDM Professional* (80-N3470-4) for QXDM Professional logging.

15.3.5 Additional logging

Depending on the issue, additional logging is enabled in the kernel space.

```
adb root
adb shell
echo -n "file q6adm.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file q6afe.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file audio_acdb.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm-dai-q6-v2.c +p" > /sys/kernel/debug/dynamic_debug/control
```

15.3.6 Compile-time enable of kernel messages

Sometimes it is easier to enable the messages at compile time rather than dynamically enabling messages in audio files in the kernel.

To enable logs at compile time, add the following line before including the header files.

```
#define DEBUG
#include <…>
```

15.4 Log analysis

The log analysis for voice and VoLTE call is similar except for the voice/VoLTE Session ID (VSID) information. The VSID for a voice call is VOICE_VSID (0x10C01000), whereas for a VoLTE call it is VOLTE_VSID (0x10C02000).

15.4.1 Logcat logs

15.4.1.1 Voice MO call

```
01-13 14:17:40.298   212  1150 D audio_hw_primary: out_set_parameters:
enter: usecase(1: low-latency-playback) kvpairs: routing=2
01-13 14:17:40.308   212  1149 D audio_hw_primary: select_devices:
out_snd_device(2: speaker) in_snd_device(0: )
01-13 14:17:40.308   212  1149 D hardware_info: hw_info_append_hw_type :
device_name = speaker
01-13 14:17:40.308   212  1149 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 14, path =  0
 . . .
01-13 14:17:40.378  1322  1322 V OutgoingCallBroadcaster: onCreate: this =
com.android.phone.OutgoingCallBroadcaster@419a18b8, icicle = null
01-13 14:17:40.378  1322  1322 V OutgoingCallBroadcaster:  - getIntent() =
Intent { act=android.intent.action.CALL_PRIVILEGED dat=tel:xxxxxxxxx-xxxx
flg=0x10800000 cmp=com.android.phone/.PrivilegedOutgoingCallBroadcaster
(has extras) }
01-13 14:17:40.378  1322  1322 V OutgoingCallBroadcaster:  - configuration
= {1.0 310mcc410mnc en_US ldltr sw360dp w360dp h615dp 320dpi nrml long port
finger -keyb/v/h -nav/h s.6}
01-13 14:17:40.378  1322  1322 V OutgoingCallBroadcaster: processIntent() =
Intent { act=android.intent.action.CALL_PRIVILEGED dat=tel:xxxxxxxxx-xxxx
flg=0x10800000 cmp=com.android.phone/.PrivilegedOutgoingCallBroadcaster
(has extras) }, thread: Thread[main,5,main]
01-13 14:17:40.378  1322  1322 D OutgoingCallBroadcaster: subscription when
there is (from Extra):0
01-13 14:17:40.378  1322  1322 D OutgoingCallBroadcaster:
outGoingcallBroadCaster action is android.intent.action.CALL_PRIVILEGED
number = (858) 845-5679
01-13 14:17:40.388  1322  1322 V OutgoingCallBroadcaster:  - updating
action from CALL_PRIVILEGED to android.intent.action.CALL
01-13 14:17:40.388  1322  1322 D OutgoingCallBroadcaster: for non emergency
call,sub is  :0
01-13 14:17:40.388  1322  1322 D PhoneApp: pulse screen lock
01-13 14:17:40.388  1322  1322 D OutgoingCallBroadcaster:
processAddParticipant return = false
01-13 14:17:40.388  1322  1322 D CallGatewayManager:
checkAndCopyPhoneProviderExtras: some or all extras are missing.
01-13 14:17:40.388  1322  1322 V OutgoingCallBroadcaster:  - Broadcasting
intent: Intent { act=android.intent.action.NEW_OUTGOING_CALL flg=0x10000000
(has extras) }.
```

```
01-13 14:17:40.398  1322  1322 V OutgoingCallBroadcaster: At the end of
onCreate(). isFinishing(): false
01-13 14:17:40.418  3696  3696 W ResourceType: Failure getting entry for
0x7f0b03b6 (t=10 e=950) in package 0 (error -75)
01-13 14:17:40.418  3696  3696 W Resources: Converting to string:
TypedValue{t=0x1/d=0x7f0b03b6 a=-1 r=0x7f0b03b6}
01-13 14:17:40.458   222   285 D ThermalEngine: CPU[2] online
 . ..
01-13 14:17:40.488  1322  1322 D CallController: placeCallInternal()...
intent = Intent { act=android.intent.action.CALL dat=tel:xxxxxxxxx-xxxx
(has extras) }
01-13 14:17:40.488  1322  1322 D PhoneUtils: getInitialNumber(): Intent
{ act=android.intent.action.CALL dat=tel:xxxxxxxxx-xxxx (has extras) }
01-13 14:17:40.488  1322  1322 D PhoneUtils: ==> got
EXTRA_ACTUAL_NUMBER_TO_DIAL; returning 'xxxxxxxxxx'
01-13 14:17:40.488  1322  1322 D PhoneUtils: pickPhoneBasedOnNumber: scheme
tel, number xxxxxxxxxx, sipUri null, subscription0
01-13 14:17:40.488  1322  1322 D PhoneUtils: placeCall '8588455679'
GW:'null' CallType:0
01-13 14:17:40.498  1322  1322 D PhoneUtils: PhoneUtils.startGetCallerInfo:
new query for phone number...
01-13 14:17:40.498  1322  1322 D PhoneUtils: - number (address): xxxxxxxxxx
01-13 14:17:40.498  1322  1322 D PhoneUtils: - c:  incoming: false state:
DIALING post dial state: NOT_STARTED
01-13 14:17:40.498  1322  1322 D PhoneUtils: - phone: Handler
(com.android.internal.telephony.gsm.GSMPhone) {418d2ab0}
01-13 14:17:40.498  1322  1322 D PhoneUtils: - phoneType: 1
01-13 14:17:40.498  1322  1322 D PhoneUtils:   ==> PHONE_TYPE_GSM
// Network type is GSM
01-13 14:17:40.498  1322  1322 D PhoneUtils: modifyForSpecialCnapCases:
initially, number=xxxxxxxxxx, presentation=1 ci
com.android.internal.telephony.CallerInfo@41a0e228 { name null, phoneNumber
null }
01-13 14:17:40.498  1322  1322 D PhoneUtils: checkCnapSpecialCases, normal
str. number: 8588455679
01-13 14:17:40.498  1322  1322 D PhoneUtils: modifyForSpecialCnapCases:
returning number string=xxxxxxxxxx
01-13 14:17:40.498  1322  1322 D PhoneUtils: startGetCallerInfo: query
based on number: xxxxxxxxxx
01-13 14:17:40.498  1322  1322 D PhoneUtils: setAudioMode()...OFFHOOK
01-13 14:17:40.498   212  1153 D audio_hw_primary: adev_set_parameters:
enter: vsid=281026560;call_state=1
01-13 14:17:40.498   212  1153 D voice_extn: update_call_states
is_in_call:0 mode:0
01-13 14:17:40.498   212  1153 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:17:40.498   212   212 D audio_hw_primary: adev_set_parameters:
enter: vsid=281022464;call_state=1
```

```
01-13 14:17:40.498   212   212 D voice_extn: update_call_states
is_in_call:0 mode:0
01-13 14:17:40.498   212   212 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:17:40.498   959  1369 I MediaFocusControl:  AudioFocus
requestAudioFocus() from AudioFocus_For_Phone_Ring_And_Calls
01-13 14:17:40.508   212  1152 D audio_hw_primary: adev_set_parameters:
enter: vsid=281026560;call_state=1
01-13 14:17:40.508   212  1152 D voice_extn: update_call_states
is_in_call:0 mode:0
01-13 14:17:40.508   212  1152 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
// ANC ( Active Noise Cancellation is not enabled
01-13 14:17:40.508   212  1151 D audio_hw_primary: adev_set_parameters:
enter: vsid=281022464;call_state=2
01-13 14:17:40.508   212  1151 D voice_extn: update_call_states
is_in_call:0 mode:0
01-13 14:17:40.508   212  1151 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:17:40.508   212  1153 D audio_hw_primary: adev_set_mode mode 2
01-13 14:17:40.528   212  1148 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: fm_volume=0.0000000000
01-13 14:17:40.528   212  1148 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:17:40.528  1322  1322 D PhoneUtils: about to activate speaker
01-13 14:17:40.528  1322  1322 D PhoneUtils: activateSpeakerIfDocked()...
01-13 14:17:40.528  1322  1322 D CallController: ==> placeCall(): success
from placeCallInternal(): SUCCESS
01-13 14:17:40.538  1322  1322 D PhoneUtils: query complete, updating
connection.userdata
01-13 14:17:40.538  1322  1322 D PhoneUtils: - onQueryComplete:
CallerInfo:com.android.internal.telephony.CallerInfo@41aaea18 { name null,
phoneNumber non-null }
01-13 14:17:40.538  1322  1322 D PhoneUtils: ==> Stashing CallerInfo
com.android.internal.telephony.CallerInfo@41a0e228 { name null, phoneNumber
non-null } into the connection...
01-13 14:17:40.558   222   285 D ThermalEngine: CPU[1] online
01-13 14:17:40.558  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:17:40.558  1322  1322 D PhoneUtils: Call is not active
01-13 14:17:40.558  1322  1322 D PhoneApp: requestWakeState(FULL)...
01-13 14:17:40.558  3818  3818 I InCall  : CallHandlerService - onCreate
01-13 14:17:40.558  3818  3818 I InCall  : CallHandlerService - onBind
01-13 14:17:40.558  1322  1322 D CallNotifier: onPhoneStateChanged: state =
OFFHOOK
01-13 14:17:40.558  1322  1322 D NotificationMgr: updateStatusBar: state =
0x40000
01-13 14:17:40.568  1157  1157 D PhoneStatusBar: disable: < expand icons
ALERTS* ticker system_info back home recent clock search >
```

```
01-13 14:17:40.568  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:17:40.568  1322  1322 D CallNotifier: onPhoneStateChanged: OFF
HOOK
01-13 14:17:40.568  1322  1322 D PhoneUtils: setAudioMode()...OFFHOOK
01-13 14:17:40.568  1322  1322 D PhoneUtils: setAudioMode() no change:
MODE_IN_CALL
01-13 14:17:40.568  1322  1322 D CallNotifier: stopRing()... (OFFHOOK
state)
01-13 14:17:40.568  1322  1322 D Ringer  : stopRing()...
01-13 14:17:40.568  1322  1322 D Ringer  : - stopRing: null mRingHandler!
01-13 14:17:40.568  1322  1322 D PhoneUtils: ConnectionHandler: updating
mute state for each connection
01-13 14:17:40.568  1322  1322 D PhoneUtils: setMuteInternal: using
setMicrophoneMute(false)...
01-13 14:17:40.568   212  1152 D audio_hw_primary: adev_set_mic_mute state
0
01-13 14:17:40.568  1322  1322 D NotificationMgr: updateMuteNotification:
not muted (or not offhook)
01-13 14:17:40.568  1322  1322 D AudioRouter: onMuteChange: false
01-13 14:17:40.568  1322  1322 D AudioRouter: AudioMode: EARPIECE
01-13 14:17:40.568  1322  1322 D AudioRouter: Supported AudioMode:
EARPIECE, SPEAKER
01-13 14:17:40.578  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:17:40.578  1322  1322 D PhoneUtils: Call is not active
01-13 14:17:40.578  1322  1322 D PhoneApp: requestWakeState(FULL)...
01-13 14:17:40.578  1322  1322 D CallNotifier: onPhoneStateChanged: state =
OFFHOOK
01-13 14:17:40.578  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:17:40.578  1322  1322 D CallNotifier: onPhoneStateChanged: OFF
HOOK
01-13 14:17:40.578  1322  1322 D PhoneUtils: setAudioMode()...OFFHOOK
01-13 14:17:40.578  1322  1322 D PhoneUtils: setAudioMode() no change:
MODE_IN_CALL
01-13 14:17:40.578  1322  1322 D CallNotifier: stopRing()... (OFFHOOK
state)
 . . .
01-13 14:17:40.588  3818  3830 D InCall  : CallHandlerService -
startCallService:
com.android.services.telephony.common.ICallCommandService$Stub$Proxy@418da0
70
01-13 14:17:40.588  3818  3830 I InCall  : CallHandlerService -
onSupportedAudioModeChange : EARPIECE, SPEAKER
01-13 14:17:40.588  3818  3818 D InCall  : CallHandlerService -
executeMessage 9
01-13 14:17:40.588  3818  3818 I InCall  : CallHandlerService - doStart
```

```
01-13 14:17:40.588  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
01-13 14:17:40.588  3818  3818 D InCall  : CallHandlerService -
executeMessage 5
01-13 14:17:40.588  3818  3818 I InCall  : CallHandlerService -
ON_SUPPORTED_AUDIO_MODE: EARPIECE, SPEAKER
01-13 14:17:40.588  3818  3829 I InCall  : CallHandlerService -
onAudioModeChange : EARPIECE
01-13 14:17:40.588  3818  3818 D InCall  : CallHandlerService -
executeMessage 4
01-13 14:17:40.588  3818  3818 I InCall  : CallHandlerService -
ON_AUDIO_MODE: EARPIECE, muted (false)
01-13 14:17:40.588  1322  1322 D CallNotifier: PHONE_ENHANCED_VP_OFF...
01-13 14:17:40.588  1322  1322 D CallNotifier: PHONE_ENHANCED_VP_OFF...
01-13 14:17:40.588  3818  3818 D InCall  : CallHandlerService -
executeMessage 2
01-13 14:17:40.588  3818  3818 D InCall  : CallList - onUpdate(...)
01-13 14:17:40.628  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:17:40.628  1322  1322 D PhoneUtils: Call is not active
01-13 14:17:40.628  1322  1322 D PhoneApp: requestWakeState(FULL)...
01-13 14:17:40.628  1322  1322 D CallNotifier: onPhoneStateChanged: state =
OFFHOOK
01-13 14:17:40.628  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:17:40.628  1322  1322 D CallNotifier: onPhoneStateChanged: OFF
HOOK
01-13 14:17:40.628  1322  1322 D PhoneUtils: setAudioMode()...OFFHOOK
01-13 14:17:40.628  1322  1322 D PhoneUtils: setAudioMode() no change:
MODE_IN_CALL
// Mode is changed to MODE_IN_CALL for a Voice/VoLTE call use case
01-13 14:17:40.628  1322  1322 D CallNotifier: stopRing()... (OFFHOOK
state)
01-13 14:17:40.628  1322  1322 D Ringer  : stopRing()...
01-13 14:17:40.628  1322  1322 D Ringer  : - stopRing: null mRingHandler!
01-13 14:17:40.628  3818  3818 D InCall  :  - updateInCallNotification:
timer scheduled
01-13 14:17:40.628  1322  1322 D PhoneUtils: ConnectionHandler: updating
mute state for each connection
01-13 14:17:40.638  1322  1322 D PhoneUtils: setMuteInternal: using
setMicrophoneMute(false)...
01-13 14:17:40.638  3818  3818 D InCall  : CallHandlerService -
executeMessage 2
01-13 14:17:40.638   212   212 D audio_hw_primary: adev_set_mic_mute state
0
01-13 14:17:40.638   212  1148 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: routing=1
01-13 14:17:40.638   212  1148 D voice_extn: update_calls: enter:
```

```
01-13 14:17:40.638   212  1148 D voice_extn: update_calls: cur_state=1
new_state=2 vsid=10c01000
01-13 14:17:40.638   212  1148 D voice_extn: update_calls: INACTIVE ->
ACTIVE vsid:10c01000
```

Use VSID 0x10c01000 (VOICE_VSID) to change the state of the voice session from CALL_INACTIVE(1) to CALL_ACTIVE (2). The VSID and the CALL states are defined in the file /hardware/qcom/audio/hal/voice_extn.h and /hardware/qcom/audio/hal/voice_extn/voice_extn.c.

```
01-13 14:17:40.638   212  1148 D voice   : start_call: enter usecase:voice-
call
```

Start call. The use case is voice-call. The use case is defined in the array use_case_table[AUDIO_USECASE_MAX] defined in /hardware/qcom/audio/hal/audio_hw.c.

```
01-13 14:17:40.638   212  1148 D audio_hw_primary: select_devices:
out_snd_device(6: voice-handset) in_snd_device(27: handset-mic)
```

The devices chosen are voice-handset as the Rx device and handset-mic as the Tx device. The device IDs are defined in the file /hardware/qcom/audio/hal/msm8974/platform.h.

```
01-13 14:17:40.638  1322  1322 D NotificationMgr: updateMuteNotification:
not muted (or not offhook)
01-13 14:17:40.638  1322  1322 D AudioRouter: onMuteChange: false
01-13 14:17:40.638  1322  1322 D AudioRouter: AudioMode: EARPIECE
01-13 14:17:40.638  1322  1322 D AudioRouter: Supported AudioMode:
EARPIECE, SPEAKER
01-13 14:17:40.638  3818  3873 I InCall  : CallHandlerService -
onAudioModeChange : EARPIECE
01-13 14:17:40.638  3818  3818 D InCall  : CallList - onUpdate(...)
01-13 14:17:40.638  3818  3830 I InCall  : CallHandlerService -
onSupportedAudioModeChange : EARPIECE, SPEAKER
01-13 14:17:40.638  3818  3818 D InCall  : CallHandlerService -
executeMessage 2
01-13 14:17:40.638  3818  3818 D InCall  : CallList - onUpdate(...)
01-13 14:17:40.638  3818  3818 D InCall  : CallHandlerService -
executeMessage 4
01-13 14:17:40.638  3818  3818 I InCall  : CallHandlerService -
ON_AUDIO_MODE: EARPIECE, muted (false)
01-13 14:17:40.638  3818  3818 D InCall  : CallHandlerService -
executeMessage 5
01-13 14:17:40.638  3818  3818 I InCall  : CallHandlerService -
ON_SUPPORTED_AUDIO_MODE: EARPIECE, SPEAKER
01-13 14:17:40.658  3818  3818 D InCall  : InCallActivity - onCreate()...
this = com.android.incallui.InCallActivity@41f5acc0
```

```
01-13 14:17:40.658   212  1148 D hardware_info: hw_info_append_hw_type :
device_name = speaker
01-13 14:17:40.658   212  1148 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:17:40.658   212  1148 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 7, path =  0
01-13 14:17:40.658   212  1148 D ACDB-LOADER: ACDB -> send_adm_topology
01-13 14:17:40.658   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
01-13 14:17:40.658   212  1148 D ACDB-LOADER: ACDB -> send_audtable
01-13 14:17:40.658   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
01-13 14:17:40.658   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
01-13 14:17:40.658   212  1148 D ACDB-LOADER: ACDB -> send_audvoltable
01-13 14:17:40.658   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
01-13 14:17:40.658   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
01-13 14:17:40.658   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
01-13 14:17:40.688   212  1148 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:17:40.688   212  1148 D hardware_info: hw_info_append_hw_type :
device_name = handset-mic
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 4, path =  1
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> send_adm_topology
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> send_audtable
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> send_audvoltable
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
01-13 14:17:40.688   212  1148 D         : Failed to fetch the lookup
information of the device 00000004
01-13 14:17:40.688   212  1148 E ACDB-LOADER: Error: ACDB AudProc vol
returned = -19
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
```

Send audio calibration for ACDB ID 7 for the Rx device and ACDB ID 4 for the Tx device.

```
01-13 14:17:40.688   212  1148 I listen_hal_loader:
audio_extn_listen_update_status(): stop listen. current active device =
handset-mic. Event = 1
```

```
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> send_voice_cal,
acdb_rx = 7, acdb_tx = 4, feature_set = 0
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
send_voice_rx_topology
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOCPROC_COMMON_TOPOLOGY_ID
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
send_voice_tx_topology
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOCPROC_COMMON_TOPOLOGY_ID
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> ACDB_CMD_GET_AFE_DATA
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_SIDETONE_CAL
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> send_voice_columns,
table 1
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_COLUMNS_INFO
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_PROC_COMMON_TABLE
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_VOCPROC_CAL
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_PROC_DEVICE_CFG
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_VOCPROC_DEV_CFG_CAL
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> send_voice_columns,
table 2
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_COLUMNS_INFO
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_PROC_GAIN_DEP_VOLTBL
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_VOCPROC_VOL_CAL
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> send_voice_columns,
table 3
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_COLUMNS_INFO
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_STREAM_COMMON_TABLE
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_VOCPROC_STREAM_CAL
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
01-13 14:17:40.688   212  1148 D ACDB-LOADER: Valid AANC Device for device
4 is 0
01-13 14:17:40.688   212  1148 D ACDB-LOADER: ACDB -> Sent VocProc Cal!
```

Send voice calibration for ACDB ID 7 for the Rx device and ACDB ID 4 for the Tx device.

```
01-13 14:17:40.728  3818  3818 D InCall  : InCallActivity - onCreate():
exit
01-13 14:17:40.728  3818  3818 D InCall  : CallerInfoUtils -
modifyForSpecialCnapCases: initially, number=xxxxxxxxxx, presentation=1 ci
com.android.incallui.CallerInfo@418f5760 { name null, phoneNumber null }
01-13 14:17:40.728  3818  3818 D InCall  : CallerInfoUtils -
checkCnapSpecialCases, normal str. number: 8588455679
01-13 14:17:40.728  3818  3818 D InCall  : CallerInfoUtils -
modifyForSpecialCnapCases: returning number string=xxxxxxxxxx
01-13 14:17:40.728  3818  3836 D InCall  : CallerInfoWorkerHandler -
Processing event: 1 token (arg1): 1 command: -1 query URI:
content://com.android.contacts/phone_lookup/xxxxxxx
01-13 14:17:40.728  3818  3818 D InCall  : InCallActivity - onStart()...
01-13 14:17:40.738  3818  3818 I InCall  : InCallActivity - onResume()...
01-13 14:17:40.738  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
01-13 14:17:40.748  3818  3818 V InCall  : String - getCallerInfo() based
on cursor...
01-13 14:17:40.748  3818  3818 V InCall  : String -
getGeoDescription('8588455679')...
01-13 14:17:40.748  3818  3818 V InCall  : String - parsing '8588455679'
for countryIso 'US'...
01-13 14:17:40.758  3818  3818 V InCall  : String - - parsed number:
Country Code: 1 National Number: 8588455679
01-13 14:17:40.758  3818  3818 V InCall  : String - - got description:
'California'
01-13 14:17:40.758  3818  3836 D InCall  : CallerInfoWorkerHandler -
Processing event: 3 token (arg1): 1 command: -1 query URI:
01-13 14:17:40.768   212  1148 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10dc1000
01-13 14:17:40.768   212  1148 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10c02000
01-13 14:17:40.768   212  1148 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10803000
```

Update the session states for voice/VoLTE/QChat sessions VOICE2_VSID (0x10DC1000), VOLTE_VSID (0x10C02000), and QCHAT_VSID (0x10803000). The state does not change and remains at CALL_INACTIVE.

```
01-13 14:17:40.768   212  1148 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:17:40.808   222   285 D ThermalEngine: CPU[1] offline
01-13 14:17:40.918   959   986 I ProcessStatsService: Prepared write state
in 7ms
01-13 14:17:41.258   222   285 D ThermalEngine: CPU[2] offline
01-13 14:17:41.538   212  1148 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: fm_volume=0.2668401897
01-13 14:17:41.538   212  1148 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
```

```
01-13 14:17:43.508   212  1150 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:17:44.148   212   212 D audio_hw_primary: adev_set_parameters:
enter: vsid=281026560;call_state=1

01-13 14:17:44.148   212   212 D voice_extn: update_call_states
is_in_call:1 mode:2
```

The mode is AUDIO_MODE_IN_CALL as defined in system/core/include/system/audio.h.

```
01-13 14:17:44.148   212   212 D voice_extn: update_calls: enter:
01-13 14:17:44.148   212   212 D voice_extn: update_calls: cur_state=2
new_state=2 vsid=10c01000
01-13 14:17:44.148   212   212 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10dc1000
01-13 14:17:44.148   212   212 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10c02000
01-13 14:17:44.148   212   212 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10803000
```

Update the session states for voice/VoLTE/QChat sessions VOICE_VSID (0x10c01000), VOICE2_VSID (0x10DC1000), VOLTE_VSID (0x10C02000), and QCHAT_VSID (0x10803000). The voice session with VSID VOICE_VSID is in the active state.

```
01-13 14:17:44.148   212   212 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:17:44.148   212  1152 D audio_hw_primary: adev_set_parameters:
enter: vsid=281022464;call_state=2
01-13 14:17:44.148   212  1152 D voice_extn: update_call_states
is_in_call:1 mode:2
01-13 14:17:44.148   212  1152 D voice_extn: update_calls: enter:
01-13 14:17:44.148   212  1152 D voice_extn: update_calls: cur_state=2
new_state=2 vsid=10c01000
01-13 14:17:44.148   212  1152 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10dc1000
01-13 14:17:44.148   212  1152 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10c02000
01-13 14:17:44.148   212  1152 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10803000
01-13 14:17:44.158   212  1152 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:17:44.158  1322  1322 D CallNotifier: PHONE_ENHANCED_VP_OFF...
01-13 14:17:44.168  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:17:44.168  3818  3818 D InCall  : CallHandlerService -
executeMessage 2
01-13 14:17:44.168  1322  1322 D PhoneUtils: Call is not active
```

```
01-13 14:17:44.168  3818  3818 D InCall  : CallList - onUpdate(...)
01-13 14:17:44.168  1322  1322 D PhoneApp: requestWakeState(SLEEP)...
01-13 14:17:44.168  1322  1322 D CallNotifier: onPhoneStateChanged: state =
OFFHOOK
01-13 14:17:44.168  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:17:44.168  1322  1322 D CallNotifier: onPhoneStateChanged: OFF
HOOK
01-13 14:17:44.168  1322  1322 D PhoneUtils: setAudioMode()...OFFHOOK
01-13 14:17:44.178  1322  1322 D PhoneUtils: setAudioMode() no change:
MODE_IN_CALL
```

The audio mode is AUDIO_MODE_IN_CALL.

```
01-13 14:17:44.178  1322  1322 D CallNotifier: stopRing()... (OFFHOOK
state)
01-13 14:17:44.178  1322  1322 D Ringer  : stopRing()...
01-13 14:17:44.178  1322  1322 D Ringer  : - stopRing: null mRingHandler!
01-13 14:17:44.178  1322  1322 D PhoneUtils: ConnectionHandler: updating
mute state for each connection
01-13 14:17:44.178  1322  1322 D PhoneUtils: setMuteInternal: using
setMicrophoneMute(false)...
01-13 14:17:44.178   212  1151 D audio_hw_primary: adev_set_mic_mute state
0
01-13 14:17:44.178  1322  1322 D NotificationMgr: updateMuteNotification:
not muted (or not offhook)
01-13 14:17:44.178  1322  1322 D AudioRouter: onMuteChange: false
01-13 14:17:44.178  1322  1322 D AudioRouter: AudioMode: EARPIECE
01-13 14:17:44.178  1322  1322 D AudioRouter: Supported AudioMode:
EARPIECE, SPEAKER
. . .
```

15.4.1.2 Voice MT call

```
01-13 14:19:10.648   959   986 I ActivityManager: Killing
2866:com.google.android.music:main/u0a67 (adj 15): empty for 1808s
01-13 14:19:16.308   212  1153 D audio_hw_primary: adev_set_parameters:
enter: vsid=281026560;call_state=1
01-13 14:19:16.308   212  1153 D voice_extn: update_call_states
is_in_call:0 mode:0
01-13 14:19:16.308   212  1153 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:16.318   212   212 D audio_hw_primary: adev_set_parameters:
enter: vsid=281022464;call_state=1
01-13 14:19:16.328   212   212 D voice_extn: update_call_states
is_in_call:0 mode:0
01-13 14:19:16.328   212   212 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
```

```
01-13 14:19:16.338  1322  1322 D CallNotifier: RING before NEW_RING,
skipping
01-13 14:19:17.268   212  1152 D audio_hw_primary: adev_set_parameters:
enter: vsid=281026560;call_state=1
01-13 14:19:17.268   212  1152 D voice_extn: update_call_states
is_in_call:0 mode:0
01-13 14:19:17.268   212  1152 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:17.278   212  1151 D audio_hw_primary: adev_set_parameters:
enter: vsid=281022464;call_state=1
01-13 14:19:17.278   212  1151 D voice_extn: update_call_states
is_in_call:0 mode:0
01-13 14:19:17.278   212  1151 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:17.308  1322  1322 D CallNotifier: PHONE_ENHANCED_VP_OFF...
01-13 14:19:17.368  1322  1322 D CallNotifier: RINGING... (new)
```

The log indicates an incoming call.

```
01-13 14:19:17.378  1322  1322 D CallNotifier: onNewRingingConnection():
state = RINGING, conn = {  incoming: true state: INCOMING post dial state:
NOT_STARTED }
01-13 14:19:17.378  1322  1322 D CallNotifier: stopSignalInfoTone: Stopping
SignalInfo tone player
01-13 14:19:17.398  3793  3793 D RCS_UI_LOG: CallBroadcastReceiver :
onReceive : Phone Number Incoming = 8588455679
01-13 14:19:17.408  1322  1322 D CallNotifier: - connection is ringing!
state = INCOMING
01-13 14:19:17.408  1322  1322 D CallNotifier: Holding wake lock on new
incoming connection.
01-13 14:19:17.408  1322  1322 D PhoneApp: requestWakeState(PARTIAL)...
01-13 14:19:17.418  1322  1322 D PhoneUtils: PhoneUtils.startGetCallerInfo:
new query for phone number...
01-13 14:19:17.418  1322  1322 D PhoneUtils: - number (address): xxxxxxxxxx
01-13 14:19:17.418  1322  1322 D PhoneUtils: - c:  incoming: true state:
INCOMING post dial state: NOT_STARTED
01-13 14:19:17.428  1322  1322 D PhoneUtils: - phone: Handler
(com.android.internal.telephony.gsm.GSMPhone) {418d2ab0}
01-13 14:19:17.428  1322  1322 D PhoneUtils: - phoneType: 1
01-13 14:19:17.428  1322  1322 D PhoneUtils:   ==> PHONE_TYPE_GSM
```

The log indicates an incoming call type as GSM call.

```
01-13 14:19:17.428  1322  1322 D PhoneUtils: modifyForSpecialCnapCases:
initially, number=xxxxxxxxxx, presentation=1 ci
com.android.internal.telephony.CallerInfo@41b0fe20 { name null, phoneNumber
null }
```

```
01-13 14:19:17.438  1322  1322 D PhoneUtils: checkCnapSpecialCases, normal
str. number: 8588455679
01-13 14:19:17.438  1322  4523 D CallNotifier:
SignalInfoTonePlayer.run(toneId = 98)...
01-13 14:19:17.438  1322  1322 D PhoneUtils: modifyForSpecialCnapCases:
returning number string=xxxxxxxxxx
01-13 14:19:17.458  1322  1322 D PhoneUtils: startGetCallerInfo: query
based on number: xxxxxxxxxx
01-13 14:19:17.458  1322  1322 D CallNotifier: - Starting query, posting
timeout message.
01-13 14:19:17.478  1322  1322 D CallNotifier: - onNewRingingConnection()
done.
01-13 14:19:17.508  1322  1322 D PhoneUtils: Call is not active
01-13 14:19:17.518  1322  1322 D PhoneApp: requestWakeState(FULL)...
01-13 14:19:17.528  1322  1322 D CallNotifier: onPhoneStateChanged: state =
RINGING
01-13 14:19:17.528  1322  1322 D NotificationMgr: updateStatusBar: state =
0x40000
01-13 14:19:17.538  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:19:17.558  1322  1322 D PhoneUtils: ConnectionHandler: updating
mute state for each connection
01-13 14:19:17.648  1322  1322 D PhoneUtils: query complete, updating
connection.userdata
01-13 14:19:17.648  1322  1322 D PhoneUtils: - onQueryComplete:
CallerInfo:com.android.internal.telephony.CallerInfo@41aff4a8 { name null,
phoneNumber non-null }
01-13 14:19:17.648  1322  1322 D PhoneUtils: ==> Stashing CallerInfo
com.android.internal.telephony.CallerInfo@41b0fe20 { name null, phoneNumber
non-null } into the connection...
01-13 14:19:17.658  1322  1322 D CallNotifier: CallerInfo query complete
(for CallNotifier), updating state for incoming call..
01-13 14:19:17.658  1322  1322 D Ringer  : ring()...
01-13 14:19:17.698  1322  4525 D Ringer  : mRingHandler: PLAY_RING_ONCE...
01-13 14:19:17.698  1322  4525 D Ringer  : creating ringtone:
content://settings/system/ringtone
01-13 14:19:17.738  1322  4525 E MediaPlayer-JNI: QCMediaPlayer mediaplayer
NOT present
01-13 14:19:18.088   212  1153 D ExtendedUtils: extended extractor not
needed, return default
01-13 14:19:18.088  1322  4525 D Ringtone: Successfully created local
player
01-13 14:19:18.088  1322  4525 D PhoneUtils: setAudioMode()...RINGING
01-13 14:19:18.108   959   970 I MediaFocusControl:  AudioFocus
requestAudioFocus() from AudioFocus_For_Phone_Ring_And_Calls
01-13 14:19:18.108   212   212 D audio_hw_primary: adev_set_mode mode 1
01-13 14:19:18.138  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:19:18.148  1322  1322 D PhoneUtils: Call is not active
```

```
01-13 14:19:18.148  1322  1322 D PhoneApp: requestWakeState(FULL)...
01-13 14:19:18.178  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
01-13 14:19:18.188  3818  3836 D InCall  : CallerInfoWorkerHandler -
Processing event: 1 token (arg1): 1 command: -1 query URI:
content://com.android.contacts/phone_lookup/xxxxxxx
01-13 14:19:18.208  1322  1338 D NotificationMgr: updateStatusBar: state =
0x3640000
01-13 14:19:18.208  1322  1338 D NotificationMgr: updateStatusBar: state =
0x3650000
01-13 14:19:18.208  3818  3818 V InCall  : String - getCallerInfo() based
on cursor...
01-13 14:19:18.218  3818  3818 V InCall  : String -
getGeoDescription('8588455679')...
01-13 14:19:18.218  3818  3818 V InCall  : String - parsing '8588455679'
for countryIso 'US'...
01-13 14:19:18.228  3818  3818 V InCall  : String - - parsed number:
Country Code: 1 National Number: 8588455679
01-13 14:19:18.228  3818  3818 V InCall  : String - - got description:
'California'
01-13 14:19:18.238  3818  3836 D InCall  : CallerInfoWorkerHandler -
Processing event: 3 token (arg1): 1 command: -1 query URI:
01-13 14:19:18.258   959  1369 I ActivityManager: START u0
{act=android.intent.action.MAIN flg=0x10840000
```

The log provides information related to the phone number of the incoming call.

```
cmp=com.android.dialer/com.android.incallui.InCallActivity} from pid -1
01-13 14:19:18.328  3818  3818 D InCall  : InCallActivity - onCreate()...
this = com.android.incallui.InCallActivity@41a49d98
01-13 14:19:18.338  3818  3818 D VideoCallPanel: onFinishInflate(this =
com.android.incallui.VideoCallPanel{41bac830 G.E..... ......I. 0,0-0,0
#7f0700c9 app:id/videoCallPanel})...
01-13 14:19:18.338  3818  3818 D VideoCallPanel: Is Media running in
loopback mode: false
01-13 14:19:18.338  3818  3818 D VideoCallManager: setCvoEventListener
01-13 14:19:18.398  3818  3818 D InCall  : InCallActivity - onCreate():
exit
01-13 14:19:18.408  3818  3818 D InCall  : AnswerPresenter - Showing
incoming for call id: 4 com.android.incallui.AnswerPresenter@41fb5888
01-13 14:19:18.408  3818  3818 D InCall  : InCallActivity - onStart()...
01-13 14:19:18.458   212  1148 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: routing=2
```

The use case selected is a deep-buffer playback for playback of the ringtone.

```
01-13 14:19:18.458   212  1148 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:18.468   212  1148 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: fm_volume=1.0000000000
01-13 14:19:18.468   212  1148 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:18.478  1322  4525 E MediaPlayer: Should have subtitle
controller already set
01-13 14:19:18.478  3818  3818 I InCall  : InCallActivity - onResume()...
01-13 14:19:18.478   212  1148 D audio_hw_primary: select_devices:
out_snd_device(2: speaker) in_snd_device(0: )
01-13 14:19:18.478   212  1148 D hardware_info: hw_info_append_hw_type :
device_name = speaker
01-13 14:19:18.478   212  1148 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 14, path =  0
01-13 14:19:18.478   212  1148 D ACDB-LOADER: ACDB -> send_adm_topology
01-13 14:19:18.488   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
01-13 14:19:18.488   212  1148 D ACDB-LOADER: ACDB -> send_audtable
01-13 14:19:18.488   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
01-13 14:19:18.488   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
01-13 14:19:18.488   212  1148 D ACDB-LOADER: ACDB -> send_audvoltable
01-13 14:19:18.488   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
01-13 14:19:18.488   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
01-13 14:19:18.488   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
```

Send the calibration data for the audio path for device speaker having ACDB ID 14.

```
01-13 14:19:18.538   212  4535 E AudioSink: received unknown event type: 1
inside CallbackWrapper !
01-13 14:19:18.578   212  4535 E AudioSink: received unknown event type: 1
inside CallbackWrapper !
01-13 14:19:18.588  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
01-13 14:19:18.598  3818  3818 D InCall  : InCallActivity - onPause()...
01-13 14:19:18.598  3818  3818 D InCall  : DialpadFragment - Notifying dtmf
key up.
01-13 14:19:18.768   959   986 W ActivityManager: Activity pause timeout
for ActivityRecord{41a834a0 u0
com.android.dialer/com.android.incallui.InCallActivity t9}
01-13 14:19:18.968  3818  3818 D InCall  : InCallActivity - onStop()...
```

```
01-13 14:19:19.088   959   985 I ActivityManager: Displayed
com.android.dialer/com.android.incallui.InCallActivity: +816ms (total
+819ms)
01-13 14:19:19.088   959   985 I ActivityManager: Timeline:
Activity_windows_visible id: ActivityRecord{41a834a0 u0
com.android.dialer/com.android.incallui.InCallActivity t9} time:1914376
01-13 14:19:19.088   959  2245 I QCOM PowerHAL: Got set_interactive hint
01-13 14:19:19.088   959   959 V KeyguardServiceDelegate:
onScreenTurnedOn(showListener =
com.android.internal.policy.impl.PhoneWindowManager$18@41d0fe60)
01-13 14:19:19.088   959   985 I PowerManagerService: Waking up from
sleep...
01-13 14:19:19.088   209   209 D SurfaceFlinger: Screen acquired, type=0
flinger=0xb73c8450
01-13 14:19:19.088   209   209 D qdhwcomposer: hwc_blank: Unblanking
display: 0
01-13 14:19:19.138   959  1373 V KeyguardServiceDelegate: **** SHOWN CALLED
****
01-13 14:19:19.138   959  1373 I WindowManager: No lock screen!
windowToken=android.os.BinderProxy@4203eaa0
01-13 14:19:19.348  1322  1322 D CallNotifier: RINGING...
(PHONE_INCOMING_RING event)
01-13 14:19:19.348  1322  1322 D Ringer  : ring()...
01-13 14:19:19.348  1322  1322 D Ringer  : delaying ring by 797
01-13 14:19:19.708   209   209 D qdhwcomposer: hwc_blank: Done unblanking
display: 0
01-13 14:19:19.708   959  2245 D SurfaceControl: Excessive delay in
unblankDisplay() while turning screen on: 619ms
01-13 14:19:19.718  3818  3818 W IInputConnectionWrapper: showStatusIcon on
inactive InputConnection
01-13 14:19:19.738  3818  3818 D InCall  : InCallActivity - onStart()...
01-13 14:19:19.748   212  1152 D audio_hw_primary: adev_set_parameters:
enter: screen_state=on
01-13 14:19:19.748   212  1152 D voice_extn: voice_extn_set_parameters: Not
handled here
01-13 14:19:19.748   212  1152 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:19.778  3818  3818 I InCall  : InCallActivity - onResume()...
01-13 14:19:19.838  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
01-13 14:19:19.878   959   959 D Ulp_jni : JNI:system_update. Event-0
01-13 14:19:19.878   959  1415 E LocSvc_libulp: I/===> int
ulp_msg_process_system_update(UlpSystemEvent) line 1386
01-13 14:19:19.918  3818  3818 I ActivityManager: Timeline: Activity_idle
id: android.os.BinderProxy@419a07b0 time:1915211
01-13 14:19:19.918  3818  3818 I ActivityManager: Timeline: Activity_idle
id: android.os.BinderProxy@419a07b0 time:1915212
01-13 14:19:20.148  1322  4525 D Ringer  : mRingHandler: PLAY_RING_ONCE...
```

```
01-13 14:19:21.898   212  4535 E AudioSink: received unknown event type: 1
inside CallbackWrapper !
01-13 14:19:22.348  1322  1322 D CallNotifier: RINGING...
(PHONE_INCOMING_RING event)
01-13 14:19:22.348  1322  1322 D Ringer   : ring()...
01-13 14:19:22.348  1322  1322 D Ringer   : delaying ring by 797
01-13 14:19:23.148  1322  4525 D Ringer   : mRingHandler: PLAY_RING_ONCE...
01-13 14:19:25.348  1322  1322 D CallNotifier: RINGING...
(PHONE_INCOMING_RING event)
01-13 14:19:25.348  1322  1322 D Ringer   : ring()...
01-13 14:19:25.348  1322  1322 D Ringer   : delaying ring by 797
01-13 14:19:25.498   212  4535 E AudioSink: received unknown event type: 1
inside CallbackWrapper !
01-13 14:19:26.148  1322  4525 D Ringer   : mRingHandler: PLAY_RING_ONCE...
01-13 14:19:27.588  3818  3818 D InCall   : AnswerPresenter - onAnswer:
callId=4callType=0
01-13 14:19:27.588  1322  1481 V CallCommandService:
answerCallWithCallType4 calltype0
01-13 14:19:27.588  1322  1481 D PhoneUtils:
answerCall(INCOMING)...calltype:0
01-13 14:19:27.588  1322  1481 D CallNotifier: stopRing()...
(silenceRinger)
01-13 14:19:27.588  1322  1481 D Ringer   : stopRing()...
```

stopRing() is called once the call is answered.

```
01-13 14:19:27.588  1322  1481 D PhoneUtils: setAudioMode()...RINGING
01-13 14:19:27.588  1322  4525 D Ringer   : mRingHandler: STOP_RING...
01-13 14:19:27.588  1322  1481 D PhoneUtils: setAudioMode() no change:
MODE_RINGTONE
01-13 14:19:27.588  1322  1481 D PhoneUtils: answerCall: call state =
INCOMING
01-13 14:19:27.588  1322  1481 D PhoneUtils: getOtherActiveSub: sub = 0
count = 1
01-13 14:19:27.588  1322  1481 D PhoneUtils: setMuteInternal: using
setMicrophoneMute(false)...
01-13 14:19:27.588   212  1153 D audio_hw_primary: adev_set_mic_mute state
0
01-13 14:19:27.588  1322  1481 D NotificationMgr: updateMuteNotification:
not muted (or not offhook)
01-13 14:19:27.588  1322  1481 D AudioRouter: onMuteChange: false
01-13 14:19:27.588  1322  1481 D AudioRouter: AudioMode: EARPIECE
01-13 14:19:27.588  1322  1481 D AudioRouter: Supported AudioMode:
EARPIECE, SPEAKER
01-13 14:19:27.598   212  1151 D audio_hw_primary: adev_set_parameters:
enter: vsid=281022464;call_state=2
01-13 14:19:27.598   212  1151 D voice_extn: update_call_states
is_in_call:0 mode:1
```

```
01-13 14:19:27.598   212  1151 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:27.598  1322  1481 D PhoneUtils: setAudioMode()...RINGING
```

The phone state is defined in the file /frameworks/base/telephony/java/com/android/internal/telephony/PhoneConstants.java.

```
    public enum State {
        IDLE, RINGING, OFFHOOK;
   };
01-13 14:19:27.598  1322  1481 D PhoneUtils: setAudioMode() no change:
MODE_RINGTONE
```

setAudioMode is called with MODE_RINGTONE mode as defined in /frameworks/base/media/java/android/media/AudioManager.java, for example, MODE_NORMAL is mapped to AudioSystem. MODE_NORMAL is defined in frameworks/base/ media/java/android/media/AudioSystem.java. MODE_RINGTONE is mapped to AudioSystem.MODE_RINGTONE as defined in frameworks/base/ media/java/android/media/AudioSystem.java, and so on.

```
01-13 14:19:27.598  1322  1481 D PhoneUtils: activateSpeakerIfDocked()...
01-13 14:19:27.598  1322  1322 D CallNotifier: PHONE_ENHANCED_VP_OFF...
01-13 14:19:27.858   212  1153 D audio_hw_primary: adev_set_parameters:
enter: vsid=281026560;call_state=1
01-13 14:19:27.858   212  1153 D voice_extn: update_call_states
is_in_call:0 mode:1
01-13 14:19:27.858   212  1153 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:27.868   212  1152 D audio_hw_primary: adev_set_parameters:
enter: vsid=281022464;call_state=2
01-13 14:19:27.868   212  1152 D voice_extn: update_call_states
is_in_call:0 mode:1
01-13 14:19:27.868   212  1152 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:27.878  1322  1322 D CallNotifier: PHONE_ENHANCED_VP_OFF...
01-13 14:19:27.898  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:19:27.898  1322  1322 D PhoneUtils: In isImsVideoCall call=ACTIVE
01-13 14:19:27.898  1322  1322 D PhoneApp: requestWakeState(SLEEP)...
01-13 14:19:27.908  1322  1322 D CallNotifier: onPhoneStateChanged: state =
OFFHOOK
01-13 14:19:27.908  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:19:27.908  1322  1322 D CallNotifier: onPhoneStateChanged: OFF
HOOK
01-13 14:19:27.908  1322  1322 D PhoneUtils: setAudioMode()...OFFHOOK
// setAudioMode to OFFHook state
```

```
01-13 14:19:27.908   959  1217 I MediaFocusControl:  AudioFocus
requestAudioFocus() from AudioFocus_For_Phone_Ring_And_Calls
01-13 14:19:27.908   212  1151 D audio_hw_primary: adev_set_mode mode 2
01-13 14:19:27.908   212  1148 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: fm_volume=0.0000000000
01-13 14:19:27.908   212  1148 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:27.918  3818  3818 D InCall  : CallList - onUpdate(...)
01-13 14:19:27.918  1322  1339 D NotificationMgr: updateStatusBar: state =
0x50000
01-13 14:19:27.918  1322  1339 D NotificationMgr: updateStatusBar: state =
0x40000
01-13 14:19:27.928  1322  1322 D CallNotifier: stopRing()... (OFFHOOK
state)
01-13 14:19:27.928  1322  1322 D Ringer  : stopRing()...
```

When the call is accepted, stopRing() is called to stop the ringtone.

```
01-13 14:19:27.928  1322  1322 D Ringer  : - stopRing: null mRingHandler!
01-13 14:19:27.928  1322  1322 D PhoneUtils: ConnectionHandler: updating
mute state for each connection
01-13 14:19:27.928  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
01-13 14:19:27.928  1322  1322 D PhoneUtils: setMuteInternal: using
setMicrophoneMute(false)...
01-13 14:19:27.938   212  1151 D audio_hw_primary: adev_set_mic_mute state
0
01-13 14:19:27.938  1322  1322 D NotificationMgr: updateMuteNotification:
not muted (or not offhook)
01-13 14:19:27.938  1322  1322 D AudioRouter: onMuteChange: false
01-13 14:19:27.938  1322  1322 D AudioRouter: AudioMode: EARPIECE
01-13 14:19:27.938  1322  1322 D AudioRouter: Supported AudioMode:
EARPIECE, SPEAKER
01-13 14:19:27.978   212  1149 D audio_hw_primary: select_devices:
out_snd_device(6: voice-handset) in_snd_device(0: )
01-13 14:19:27.988  3352  3352 D YouTube :
youtube.app.prefetch.g.onReceive:367 Received:
android.intent.action.USER_PRESENT
01-13 14:19:28.008   212  1149 D hardware_info: hw_info_append_hw_type :
device_name = speaker
01-13 14:19:28.008   212  1149 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:19:28.008   212  1149 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 7, path =  0
01-13 14:19:28.008   212  1149 D ACDB-LOADER: ACDB -> send_adm_topology
01-13 14:19:28.008   212  1149 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
01-13 14:19:28.008   212  1149 D ACDB-LOADER: ACDB -> send_audtable
```

```
01-13 14:19:28.008   212  1149 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
01-13 14:19:28.008   212  1149 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
01-13 14:19:28.008   212  1149 D ACDB-LOADER: ACDB -> send_audvoltable
01-13 14:19:28.008   212  1149 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
01-13 14:19:28.008   212  1149 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
01-13 14:19:28.008   212  1149 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
```

Send the audio calibration data for device voice-handset with ACDB ID 7.

```
01-13 14:19:28.048   212  1149 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:19:28.238   212  1148 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: routing=1
01-13 14:19:28.238   212  1148 D voice_extn: update_calls: enter:
01-13 14:19:28.238   212  1148 D voice_extn: update_calls: cur_state=1
new_state=2 vsid=10c01000
01-13 14:19:28.238   212  1148 D voice_extn: update_calls: INACTIVE ->
ACTIVE vsid:10c01000
```

Change the state of the voice session with VSID VOICE_VSID (0x10c01000) from CALL_INACTIVE to CALL_ACTIVE.

```
01-13 14:19:28.238   212  1148 D voice   : start_call: enter usecase:voice-
call
```

Start a call with the use case as voice-call. The use case is defined in the array use_case_table[AUDIO_USECASE_MAX] as defined in /hardware/qcom/audio/hal/audio_hw.c.

```
01-13 14:19:28.248   212  1148 D audio_hw_primary: select_devices:
out_snd_device(6: voice-handset) in_snd_device(27: handset-mic)
```

The devices chosen are voice-handset as the Rx device and handset-mic as the Tx device. The device IDs are mentioned in the file /hardware/qcom/audio/hal/msm8974/platform.h.

```
01-13 14:19:28.248   212  1148 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:19:28.248   212  1148 D hardware_info: hw_info_append_hw_type :
device_name = handset-mic
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 4, path =  1
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> send_adm_topology
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
```

```
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> send_audtable
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> send_audvoltable
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
01-13 14:19:28.248   212  1148 D         : Failed to fetch the lookup
information of the device 00000004
01-13 14:19:28.248   212  1148 E ACDB-LOADER: Error: ACDB AudProc vol
returned = -19
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
01-13 14:19:28.248   212  1148 I listen_hal_loader:
audio_extn_listen_update_status(): stop listen. current active device =
handset-mic. Event = 1
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> send_voice_cal,
acdb_rx = 7, acdb_tx = 4, feature_set = 0
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
send_voice_rx_topology
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOCPROC_COMMON_TOPOLOGY_ID
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
send_voice_tx_topology
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOCPROC_COMMON_TOPOLOGY_ID
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> ACDB_CMD_GET_AFE_DATA
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_SIDETONE_CAL
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> send_voice_columns,
table 1
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_COLUMNS_INFO
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_PROC_COMMON_TABLE
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_VOCPROC_CAL
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_PROC_DEVICE_CFG
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_VOCPROC_DEV_CFG_CAL
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> send_voice_columns,
table 2
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_COLUMNS_INFO
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_PROC_GAIN_DEP_VOLTBL
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_VOCPROC_VOL_CAL
```

```
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> send_voice_columns,
table 3
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_COLUMNS_INFO
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_STREAM_COMMON_TABLE
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_VOCPROC_STREAM_CAL
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
01-13 14:19:28.248   212  1148 D ACDB-LOADER: Valid AANC Device for device
4 is 0
01-13 14:19:28.248   212  1148 D ACDB-LOADER: ACDB -> Sent VocProc Cal!
```

Send voice calibration for ACDB ID 7 for the Rx device and ACDB ID 4 for the Tx device.

```
01-13 14:19:28.378   212  1148 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10dc1000
01-13 14:19:28.378   212  1148 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10c02000
01-13 14:19:28.378   212  1148 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10803000
```

State of voice/VoLTE/QCHAT sessions with VSID VOICE2_VSID (0x10dc1000), VOLTE_VSID (0x10c02000), and QCHAT_VSID (0x10803000) remaining in CALL_INACTIVE.

```
01-13 14:19:28.378   212  1148 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:28.928   212  1148 D audio_hw_primary: out_set_parameters:
enter: usecase(0: deep-buffer-playback) kvpairs: fm_volume=0.2668401897
01-13 14:19:28.928   212  1148 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:19:30.828   212  1148 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:19:31.998   212  1150 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:19:33.958   209   209 D SurfaceFlinger: Screen released, type=0
flinger=0xb73c8450
01-13 14:19:33.958   209   209 D qdhwcomposer: hwc_blank: Blanking display:
0
01-13 14:19:34.248   209   209 D qdhwcomposer: hwc_blank: Done blanking
display: 0
01-13 14:19:34.248   959  2242 D SurfaceControl: Excessive delay in
blankDisplay() while turning screen off: 291ms
01-13 14:19:34.258   959  2242 I QCOM PowerHAL: Got set_interactive hint
01-13 14:19:34.428   959  2243 I QCOM PowerHAL: Got set_interactive hint
```

```
01-13 14:19:34.428   209   209 D SurfaceFlinger: Screen acquired, type=0
flinger=0xb73c8450
01-13 14:19:34.428   209   209 D qdhwcomposer: hwc_blank: Unblanking
display: 0
01-13 14:19:35.038   209   209 D qdhwcomposer: hwc_blank: Done unblanking
display: 0
01-13 14:19:35.038   959  2243 D SurfaceControl: Excessive delay in
unblankDisplay() while turning screen on: 609ms
```

15.4.1.3 End of voice call

```
01-13 14:16:03.948   212  1153 D audio_hw_primary: adev_set_parameters:
enter: vsid=281026560;call_state=1
01-13 14:16:03.948   212  1153 D voice_extn: update_call_states
is_in_call:1 mode:2
01-13 14:16:03.948   212  1153 D voice_extn: update_calls: enter:
01-13 14:16:03.948   212  1153 D voice_extn: update_calls: cur_state=2
new_state=2 vsid=10c01000
```

The current and new state is CALL_ACTIVE for a voice session with VSID VOICE_VSID (0x10c01000)

```
01-13 14:16:03.958   212  1153 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10dc1000
01-13 14:16:03.958   212  1153 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10c02000
01-13 14:16:03.958   212  1153 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10803000
01-13 14:16:03.958   212  1153 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:16:03.968   212  1151 D audio_hw_primary: adev_set_parameters:
enter: vsid=281022464;call_state=2
01-13 14:16:03.968   212  1151 D voice_extn: update_call_states
is_in_call:1 mode:2
01-13 14:16:03.968   212  1151 D voice_extn: update_calls: enter:
01-13 14:16:03.968   212  1151 D voice_extn: update_calls: cur_state=2
new_state=2 vsid=10c01000
01-13 14:16:03.968   212  1151 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10dc1000
01-13 14:16:03.968   212  1151 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10c02000
01-13 14:16:03.968   212  1151 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10803000
01-13 14:16:03.968   212  1151 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:16:04.228   212   212 D audio_hw_primary: adev_set_parameters:
enter: vsid=281026560;call_state=1
 . . .
```

```
01-13 14:16:04.248   212  1152 D voice_extn: update_call_states
is_in_call:1 mode:2
01-13 14:16:04.248   212  1152 D voice_extn: update_calls: enter:
01-13 14:16:04.258   212  1152 D voice_extn: update_calls: cur_state=2
new_state=1 vsid=10c01000
01-13 14:16:04.258   212  1152 D voice_extn: update_calls:
ACTIVE/HOLD/LOCAL_HOLD -> INACTIVE vsid:10c01000
// Update call is called  to change the state from CALL_ACTIVE to
CALL_INACTIVE .
01-13 14:16:04.258   212  1152 D voice   : stop_call: enter usecase:voice-
call
```

stop_call() is called to end the voice call.

```
01-13 14:16:04.348   212  1152 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:16:04.348   212  1152 D hardware_info: hw_info_append_hw_type :
device_name = handset-mic
01-13 14:16:04.428   212  1152 I listen_hal_loader:
audio_extn_listen_update_status(): start listen. current active device =
handset-mic. Event = 0
01-13 14:16:04.438   212  1152 D voice   : stop_call: exit: status(0)
```

Complete the stop_call() function.

```
01-13 14:16:04.438   212  1152 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10dc1000
01-13 14:16:04.448   212  1152 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10c02000
01-13 14:16:04.448   212  1152 D voice_extn: update_calls: cur_state=1
new_state=1 vsid=10803000
01-13 14:16:04.448   212  1152 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:16:04.498  1157  1157 D StatusBar.NetworkController:
onCallStateChanged state=0
01-13 14:16:04.728  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:16:04.738   959  1302 I PowerManagerService: Waking up from
sleep...
01-13 14:16:04.748   959  2245 I QCOM PowerHAL: Got set_interactive hint
 . .. .
01-13 14:16:04.758  3818  3818 D InCall  : InCallActivity - onStart()...
01-13 14:16:04.758  3818  3818 D InCall  : CallHandlerService -
executeMessage 6
01-13 14:16:04.768  3818  3818 D InCall  : CallList - disconnect cause:
NORMAL
```

```
01-13 14:16:04.768  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
 . . .
01-13 14:16:05.398  1322  1322 D PhoneUtils: Call is not active
01-13 14:16:05.398  1322  1322 D PhoneApp: requestWakeState(SLEEP)...
01-13 14:16:05.398  1322  1322 D Ringer  : stopRing()...
01-13 14:16:05.398  1322  1322 D Ringer  : - stopRing: null mRingHandler!
01-13 14:16:05.398  3818  3818 I InCall  : InCallActivity - onResume()...
01-13 14:16:05.398  1322  1322 D PhoneApp: - isOtaCallInActiveState false
01-13 14:16:05.408  1322  1322 D PhoneUtils: modifyForSpecialCnapCases:
initially, number=xxxxxxxxx-xxxx, presentation=1 ci
com.android.internal.telephony.CallerInfo@41b05b30 { name null, phoneNumber
non-null }
01-13 14:16:05.408  1322  1322 D PhoneUtils: checkCnapSpecialCases, normal
str. number: (858) 845-5679
01-13 14:16:05.408  1322  1322 D PhoneUtils: modifyForSpecialCnapCases:
returning number string=xxxxxxxxx-xxxx
01-13 14:16:05.408  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
01-13 14:16:05.418  3818  3818 I InCall  : CallHandlerService - onUnbind
01-13 14:16:05.418  3818  3818 I InCall  : CallHandlerService - onDestroy
01-13 14:16:05.428  3818  3818 D InCall  : CallHandlerService -
executeMessage 10
01-13 14:16:05.428  3818  3818 I InCall  : CallHandlerService - doStop
01-13 14:16:05.428   212  1152 D audio_hw_primary: adev_set_parameters:
enter: screen_state=on
01-13 14:16:05.428   212  1152 D voice_extn: voice_extn_set_parameters: Not
handled here
01-13 14:16:05.428   212  1152 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:0
01-13 14:16:05.428  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
01-13 14:16:05.438  1322  1322 D PhoneApp: mReceiver:
ACTION_ANY_DATA_CONNECTION_STATE_CHANGED
01-13 14:16:05.438  1322  1322 D PhoneApp: - state: DISCONNECTED
01-13 14:16:05.438  1322  1322 D PhoneApp: - reason: 2GVoiceCallEnded
```

The call is disconnected.

```
01-13 14:16:05.438  1322  1322 D PhoneApp: mReceiver:
ACTION_ANY_DATA_CONNECTION_STATE_CHANGED
01-13 14:16:05.438  1322  1322 D PhoneApp: - state: DISCONNECTED
01-13 14:16:05.438  1322  1322 D PhoneApp: - reason: 2GVoiceCallEnded
 . ..
01-13 14:16:05.448  1322  1322 D PhoneApp: mReceiver:
ACTION_ANY_DATA_CONNECTION_STATE_CHANGED
01-13 14:16:05.448   212  1148 D audio_hw_primary: select_devices:
out_snd_device(6: voice-handset) in_snd_device(0: )
```

```
01-13 14:16:05.448   212  1148 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:16:05.448   212  1148 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 7, path =  0
01-13 14:16:05.448   212  1148 D ACDB-LOADER: ACDB -> send_adm_topology
01-13 14:16:05.458   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
01-13 14:16:05.458   212  1148 D ACDB-LOADER: ACDB -> send_audtable
01-13 14:16:05.458   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
01-13 14:16:05.458   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
01-13 14:16:05.458   212  1148 D ACDB-LOADER: ACDB -> send_audvoltable
01-13 14:16:05.458   212  1148 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
01-13 14:16:05.458   212  1148 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
01-13 14:16:05.458   212  1148 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
. . .
01-13 14:16:05.468  1322  1322 D PhoneUtils: Call is not active
01-13 14:16:05.468  1322  1322 D PhoneApp: requestWakeState(SLEEP)...
01-13 14:16:05.478  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:16:05.478  1322  1322 D PhoneUtils: ConnectionHandler: updating
mute state for each connection
01-13 14:16:05.478  1322  1322 D PhoneUtils: connection ' incoming: false
state: DISCONNECTED post dial state: COMPLETE' not accounted for, removing.
01-13 14:16:05.478  1322  1322 D PhoneUtils: setMuteInternal: using
setMicrophoneMute(false)...
01-13 14:16:05.478   212  1153 D audio_hw_primary: adev_set_mic_mute state
0
01-13 14:16:05.508  1157  1157 I Choreographer: Skipped 44 frames!  The
application may be doing too much work on its main thread.
01-13 14:16:05.518   212  1149 D audio_hw_primary: select_devices:
out_snd_device(6: voice-handset) in_snd_device(0: )
01-13 14:16:05.518   212  1149 D hardware_info: hw_info_append_hw_type :
device_name = voice-handset
01-13 14:16:05.518  1322  1322 D AudioRouter: onMuteChange: false
01-13 14:16:05.518  1322  1322 D AudioRouter: AudioMode: EARPIECE
01-13 14:16:05.518  1322  1322 D AudioRouter: Supported AudioMode:
EARPIECE, SPEAKER
. . .
01-13 14:16:05.538  1322  1322 D PhoneUtils: Call is not active
01-13 14:16:05.538  1322  1322 D PhoneApp: requestWakeState(SLEEP)...
01-13 14:16:05.538  1322  1322 D AudioRouter: calculateModeFromCurrentState
EARPIECE
01-13 14:16:05.538  1322  1322 D PhoneUtils: ConnectionHandler: updating
mute state for each connection
```

```
01-13 14:16:05.538  1322  1322 D PhoneUtils: setMuteInternal: using
setMicrophoneMute(false)...
01-13 14:16:05.538   212   212 D audio_hw_primary: adev_set_mic_mute state
0
01-13 14:16:05.538  1322  1322 D AudioRouter: onMuteChange: false
01-13 14:16:05.538  1322  1322 D AudioRouter: AudioMode: EARPIECE
01-13 14:16:05.538  1322  1322 D AudioRouter: Supported AudioMode:
EARPIECE, SPEAKER
01-13 14:16:05.558  1157  1157 D PhoneStatusBar: disable: < expand icons
alerts* ticker system_info back home RECENT* clock search >
01-13 14:16:05.668  1322  4155 D PhoneUtils: turnOnSpeaker(flag=false,
store=true)...
01-13 14:16:05.668  1322  4155 D PhoneUtils: Call is not active
01-13 14:16:05.668  1322  4155 D PhoneApp: requestWakeState(SLEEP)...
01-13 14:16:05.668  1322  4155 D PhoneUtils: setAudioMode()...IDLE
```

setAudioMode is called to set the phone state to idle.

```
01-13 14:16:05.678   212  1153 D audio_hw_primary: adev_set_mode mode 0
01-13 14:16:05.678   959  1373 V KeyguardServiceDelegate: **** SHOWN CALLED
****
01-13 14:16:05.678   959  1373 I WindowManager: No lock screen!
windowToken=android.os.BinderProxy@4203eaa0
01-13 14:16:05.688   212  1150 D audio_hw_primary: out_set_parameters:
enter: usecase(1: low-latency-playback) kvpairs: routing=1
 . . .
01-13 14:16:06.118   212  1150 D audio_hw_primary: select_devices:
out_snd_device(2: speaker) in_snd_device(0: )
01-13 14:16:06.118   212  1150 D hardware_info: hw_info_append_hw_type :
device_name = handset
01-13 14:16:06.138   212  1150 D hardware_info: hw_info_append_hw_type :
device_name = handset
01-13 14:16:06.138   212  1150 D hardware_info: hw_info_append_hw_type :
device_name = speaker
01-13 14:16:06.138   212  1150 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 14, path =  0
01-13 14:16:06.138   212  1150 D ACDB-LOADER: ACDB -> send_adm_topology
 . ..
01-13 14:16:06.188   212  1150 D hardware_info: hw_info_append_hw_type :
device_name = speaker
01-13 14:16:06.768  3818  3818 I InCall  : InCallActivity - finish().
Dialog showing: false
01-13 14:16:06.858  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
01-13 14:16:06.868  3818  3818 D InCall  : InCallActivity - onPause()...
01-13 14:16:06.868  3818  3818 D InCall  : DialpadFragment - Notifying dtmf
key up.
```

```
01-13 14:16:06.868  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
01-13 14:16:06.888  3696  3696 D DialpadFragment: mAddParticipant = false
01-13 14:16:06.888  3696  3696 D DialpadFragment: showDialConference false
01-13 14:16:06.888  3696  3696 D DialpadFragment: showDialConference false
01-13 14:16:06.938  1157  1157 D PhoneStatusBar: disable: < expand icons
alerts ticker system_info BACK* HOME* RECENT CLOCK* search >
01-13 14:16:07.418  3818  3818 D InCall  : InCallActivity - onStop()...
01-13 14:16:07.418  3818  3818 D InCall  : InCallActivity - onDestroy()...
this = com.android.incallui.InCallActivity@41f561c8
01-13 14:16:07.428  3818  3818 D InCall  :  - updateInCallNotification:
timer cleared
 . . .
```

15.4.2 Kernel logs

15.4.2.1 Voice MO/MT call

```
. ..
<7>[ 1815.940353] adm_ec_ref_rx_id ec_ref_rx:16384
<7>[ 1815.940361] msm_routing_ec_ref_rx_put: msm_route_ec_ref_rx = 0
```

The EC reference is set to SLIMBUS_0_RX.

```
<7>[ 1815.940479] msm_pcm_routing_process_audio: reg 2 val 4 set 0
<7>[ 1815.940488] adm_close port_id=0x4000 index 15 perf_mode: 1
<7>[ 1815.940493] adm_close:Closing ADM: perf_mode: 1
<7>[ 1815.940498] adm_close:coppid 0 portid=0x4000 index=15 coppcnt=0
<7>[ 1815.941373] adm_callback_debug_print: code = 0x110e8 PL#0[10327],
PL#1[0], size = 8
<7>[ 1815.941380] APR_BASIC_RSP_RESULT id 10327
<7>[ 1815.941385] adm_callback: Basic callback received, wake up.
<7>[ 1815.941537] afe_close: port_id=16384
<7>[ 1815.941544] afe_close: port_id 0x4000, mad_type 0
<7>[ 1815.941549] afe_close: Not a MAD port
<7>[ 1815.951899] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
size = 8
<7>[ 1815.951907] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
token=15
<7>[ 1815.951924] afe_callback:port_id = 0
<7>[ 1815.951969] afe_apr_send_pkt: leave 0
<7>[ 1815.956466] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1815.963493] acdb_ioctl
<7>[ 1815.963528] acdb_ioctl
<7>[ 1815.963535] store_audproc_cal, path = 0
<7>[ 1815.963560] acdb_ioctl
<7>[ 1815.963568] store_audvol_cal, path = 0
```

```
<7>[ 1815.963587] acdb_ioctl
<7>[ 1815.963593] store_afe_cal, path = 0
<7>[ 1815.964493] msm_pcm_routing_process_audio: reg 2 val 4 set 1
<7>[ 1815.964616] adm_open: port 0x4000 path:1 rate:48000 mode:1
perf_mode:1
<7>[ 1815.964622] adm_open: Port ID 0x4000, index 15
<7>[ 1815.964628] adm_open:opening ADM: perf_mode: 1
<7>[ 1815.964633] adm_open: port_id=0x4000 rate=48000 topology_id=0x10312
<7>[ 1815.965295] adm_callback_debug_print: code = 0x10329 PL#0[0],
PL#1[f0e40000], size = 8
<7>[ 1815.965302] adm_callback: coppid rxed=0
<7>[ 1815.965325] adm_open: index: 15 coppid: 0
<7>[ 1815.965336] adm_matrix_map: session 0x1 path:1 num_copps:1
port_id[0]:0x4000 coppid[0]
<7>[ 1815.965343] adm_matrix_map: port_id[0x0]: 16384, index: 15 act
coppid[0x63]
<7>[ 1815.966260] adm_callback_debug_print: code = 0x110e8 PL#0[10325],
PL#1[0], size = 8
<7>[ 1815.966266] APR_BASIC_RSP_RESULT id 10325
<7>[ 1815.966271] adm_callback: Basic callback received, wake up.
<7>[ 1815.966297] afe_port_start: port id: 0x4000
```

Start the aDSP AFE port 0x4000, which is the SLIMBUS_0_RX port.

```
<7>[ 1815.966302] afe_q6_interface_prepare:
<7>[ 1815.966306] afe_send_cal
<7>[ 1815.966311] get_spk_protection_cfg,
<7>[ 1815.966317] afe_send_cal_block: path 0
<7>[ 1815.966322] get_afe_cal, path = 0
<7>[ 1815.966326] afe_send_cal_block: No AFE cal to send!
<7>[ 1815.966331] afe_send_hw_delay
<7>[ 1815.966335] get_hw_delay,
<7>[ 1815.966340] ACDB=> get_hw_delay Invalid delay/ delay entries
<7>[ 1815.966346] ACDB=> get_hw_delay: Path = 0 samplerate = 48000 usec =
572662306 status -22
<7>[ 1815.966351] afe_send_hw_delay: Failed to get hw delay info
<7>[ 1815.966357] afe_send_hw_delay port_id 16384 rate 48000 delay_usec
572662306 status -22
<7>[ 1815.966363] afe_port_start: port_id 0x4000, mad_type 0
<7>[ 1815.966699] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
size = 8
<7>[ 1815.966705] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
token=15
<7>[ 1815.966716] afe_callback:port_id = 0
<7>[ 1815.966734] afe_apr_send_pkt: leave 0
<7>[ 1815.966740] afe_send_cmd_port_start: enter
<7>[ 1815.966745] afe_send_cmd_port_start: cmd device start opcode[0x100e5]
port id[0x4000]
```

```
<7>[ 1815.973053] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0
size = 8
<7>[ 1815.973063] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0
token=15
<7>[ 1815.973086] afe_callback:port_id = 0
<7>[ 1815.973123] afe_apr_send_pkt: leave 0
<7>[ 1815.973131] task_name = AudioOut_2 pid = 1148
<7>[ 1815.973138] afe_send_cmd_port_start: leave 0
<7>[ 1815.988962] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1815.988969] msm-pcm-routing msm-pcm-routing: reg 0 val 1
<7>[ 1815.995848] acdb_ioctl
<7>[ 1815.995880] acdb_ioctl
<7>[ 1815.995887] store_audproc_cal, path = 1
<7>[ 1815.995972] acdb_ioctl
<7>[ 1815.995978] store_audvol_cal, path = 1
<7>[ 1815.995990] acdb_ioctl
<7>[ 1815.995996] store_afe_cal, path = 1
<7>[ 1815.996060] acdb_ioctl
<7>[ 1815.996078] acdb_ioctl
<7>[ 1815.996100] acdb_ioctl
<7>[ 1815.996106] store_sidetone_cal,
<7>[ 1815.996122] acdb_ioctl
<7>[ 1815.996128] store_voice_col_data,
<7>[ 1815.996200] acdb_ioctl
<7>[ 1815.996206] store_vocproc_cal,
<7>[ 1815.996321] acdb_ioctl
<7>[ 1815.996327] store_vocproc_dev_cfg_cal,
<7>[ 1815.996344] acdb_ioctl
<7>[ 1815.996349] store_voice_col_data,
<7>[ 1815.996591] acdb_ioctl
<7>[ 1815.996598] store_vocvol_cal,
<7>[ 1815.996619] acdb_ioctl
<7>[ 1815.996625] store_voice_col_data,
<7>[ 1815.996653] acdb_ioctl
<7>[ 1815.996659] store_vocstrm_cal,
<7>[ 1815.996680] acdb_ioctl
<7>[ 1815.996686] store_afe_cal, path = 1
<7>[ 1815.996697] acdb_ioctl
<7>[ 1815.996702] store_afe_cal, path = 0
```

Store the voice calibration data in the ACDB driver.

```
<7>[ 1816.024987] msm_pcm_routing_process_voice: reg 3 val 9 set 1
```

Enable routing from BE DAI 3(MSM_BACKEND_DAI_SLIMBUS_0_TX) to 9 (MSM_FRONTEND_DAI_CS_VOICE). The BE DAI and the FE DAI enums are defined in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[ 1816.024995] voc_get_session_id: Voice session has session id
0x10c01000
// vsid of the VOice session is  0x10c01000 ( VOICE_VSID)
<7>[ 1816.025001] msm_pcm_routing_process_voice: FE DAI 0x9 session_id
0x10c01000
<7>[ 1816.025009] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1816.025016] voc_set_route_flag: path_dir=1, set=1
```

Path_dir =1 is the Tx path.

```
<7>[ 1816.025022] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1816.025042] voc_set_rxtx_port: port_id=16385, type=1
<7>[ 1816.025058] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1816.025242] msm_pcm_routing_process_voice: reg 2 val 9 set 1
```

Enable routing from BE DAI 2(MSM_BACKEND_DAI_SLIMBUS_0_RX) to 9 (MSM_FRONTEND_DAI_CS_VOICE). The BE DAI and the FE DAI enums are defined in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[ 1816.025248] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.025254] msm_pcm_routing_process_voice: FE DAI 0x9 session_id
0x10c01000
<7>[ 1816.025261] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1816.025266] voc_set_route_flag: path_dir=0, set=1
<7>[ 1816.025272] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1816.025278] voc_set_rxtx_port: port_id=16384, type=0
<7>[ 1816.025284] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1816.025290] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1816.025297] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1816.025515] msm_pcm_open: Open VOICE Substream Id=CS-Voice (*)
<7>[ 1816.025521] msm_pcm_open: Instance = 1, Stream ID = CS-Voice (*)
<7>[ 1816.025540] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.025546] msm_pcm_ioctl: Falling into default snd_lib_ioctl cmd 0x1
<7>[ 1816.025552] msm_pcm_ioctl: ret 0
<7>[ 1816.025571] voc_get_session_id: Voice session has session id
0x10c01000
```

```
<7>[ 1816.025576] msm_pcm_ioctl: Falling into default snd_lib_ioctl cmd 0x4
<7>[ 1816.025582] msm_pcm_ioctl: ret 0
<7>[ 1816.025626] msm_pcm_hw_params: Voice
<7>[ 1816.026098] msm_pcm_open: Open VOICE Substream Id=CS-Voice (*)
<7>[ 1816.026105] msm_pcm_open: Instance = 2, Stream ID = CS-Voice (*)
<7>[ 1816.026121] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.026126] msm_pcm_ioctl: Falling into default snd_lib_ioctl cmd 0x1
<7>[ 1816.026131] msm_pcm_ioctl: ret 0
<7>[ 1816.026148] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.026154] msm_pcm_ioctl: Falling into default snd_lib_ioctl cmd 0x4
<7>[ 1816.026159] msm_pcm_ioctl: ret 0
<7>[ 1816.026168] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.026173] msm_pcm_ioctl: Falling into default snd_lib_ioctl cmd 0x4
<7>[ 1816.026178] msm_pcm_ioctl: ret 0
<7>[ 1816.026184] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.026189] msm_pcm_ioctl: Falling into default snd_lib_ioctl cmd 0x4
<7>[ 1816.026194] msm_pcm_ioctl: ret 0
<7>[ 1816.026211] msm_pcm_hw_params: Voice
<7>[ 1816.026599] msm_pcm_playback_prepare
<7>[ 1816.026606] msm_pcm_playback_prepare
<7>[ 1816.026973] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1816.026979] msm-pcm-routing msm-pcm-routing: reg 0 val 1
<7>[ 1816.027193] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.027199] msm_pcm_ioctl: Falling into default snd_lib_ioctl cmd 0x0
<7>[ 1816.027204] msm_pcm_ioctl: ret 0
<7>[ 1816.027215] msm_pcm_trigger: cmd = 1
<7>[ 1816.027220] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.027225] Start & Stop Voice call not handled in Trigger.
<7>[ 1816.027241] afe_port_start: port id: 0x4001
<7>[ 1816.027247] afe_q6_interface_prepare:
<7>[ 1816.027252] afe_send_cal
<7>[ 1816.027256] get_spk_protection_cfg,
<7>[ 1816.027263] afe_send_cal_block: path 1
<7>[ 1816.027267] get_afe_cal, path = 1
<7>[ 1816.027272] afe_send_cal_block: No AFE cal to send!
<7>[ 1816.027277] afe_send_hw_delay
<7>[ 1816.027282] get_hw_delay,
<7>[ 1816.027287] ACDB=> get_hw_delay Invalid delay/ delay entries
<7>[ 1816.027294] ACDB=> get_hw_delay: Path = 1 samplerate = 48000 usec =
572662306 status -22
<7>[ 1816.027299] afe_send_hw_delay: Failed to get hw delay info
```

```
<7>[ 1816.027305] afe_send_hw_delay port_id 16385 rate 48000 delay_usec
572662306 status -22
<7>[ 1816.027311] afe_port_start: port_id 0x4001, mad_type 0
<7>[ 1816.027640] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
size = 8
<7>[ 1816.027646] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
token=16
<7>[ 1816.027658] afe_callback:port_id = 0
<7>[ 1816.027676] afe_apr_send_pkt: leave 0
<7>[ 1816.027681] afe_send_cmd_port_start: enter
<7>[ 1816.027687] afe_send_cmd_port_start: cmd device start opcode[0x100e5]
port id[0x4001]
<7>[ 1816.033023] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0
size = 8
<7>[ 1816.033030] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0
token=16
<7>[ 1816.033043] afe_callback:port_id = 0
<7>[ 1816.033146] afe_apr_send_pkt: leave 0
<7>[ 1816.033152] afe_send_cmd_port_start: leave 0
<7>[ 1816.037244] msm_pcm_capture_prepare
<7>[ 1816.037253] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.037260] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1816.037266] voice_apr_register
```

Register with the APR driver.

```
<7>[ 1816.037272] voice_get_mvm_handle: mvm_handle 32
<7>[ 1816.037277] voice_get_cvs_handle: cvs_handle 70
<7>[ 1816.037282] voice_get_cvp_handle: cvp_handle 0
<7>[ 1816.037288] voice_create_mvm_cvs_session: mvm_hdl=32, cvs_hdl=70
<7>[ 1816.037294] voc_alloc_cal_shared_memory: Calibration shared buffer
already allocated
<7>[ 1816.037301] voice_mem_map_cal_block: Cal block already mem mapped
<7>[ 1816.037307] voice_send_dual_control_cmd: Send Dual Control command to
MVM
<7>[ 1816.037312] voice_get_mvm_handle: mvm_handle 32
<7>[ 1816.037317] voice_send_dual_control_cmd: send mvm Voice Ctl pkt size
= 21
<7>[ 1816.037579] qdsp_mvm_callback: Payload Length = 4, opcode=100be
<7>[ 1816.037585] qdsp_mvm_callback: session_id 0x0
<7>[ 1816.037803] qdsp_mvm_callback: Payload Length = 8, opcode=110e8
<7>[ 1816.037808] qdsp_mvm_callback: session_id 0x0
<7>[ 1816.037813] 11327 0
//11327 0 indicates the VSS_IMVM_CMD_SET_POLICY_DUAL_CONTROL command to the
// DSP and its callback status. The commands are listed in the
kernel/sound/soc/msm/qdsp6v2/
```

```
// q6voice.h file.

<7>[ 1816.037817] qdsp_mvm_callback: cmd = 0x11327
<7>[ 1816.037842]  send create cvp session, pkt size = 64
<7>[ 1816.037857] tx_topology: 69489 tx_port_id=16385, rx_port_id=16384,
mode: 0x10f7c
<7>[ 1816.037863] rx_topology: 69495, profile_id: 0x1135e, pkt_size: 64
<7>[ 1816.044956] 112bf 0
```

112bf 0 indicates a VSS_IVOCPROC_CMD_CREATE_FULL_CONTROL_SESSION_V2 command to the DSP and its callback status. The commands are listed in kernel/sound/soc/msm/qdsp6v2/q6voice.h.

```
<7>[ 1816.044961] qdsp_cvp_callback: cmd = 0x112bf
<7>[ 1816.044966] voice_set_cvp_handle: cvp_handle 205
```

cvphdl is set to a nonzero value.

```
<7>[ 1816.044971] status: 0, cvphdl=205
<7>[ 1816.044999] get_vocstrm_cal,
<7>[ 1816.045005] voice_get_cvs_handle: cvs_handle 70
<7>[ 1816.045009] get_voice_col_data,
<7>[ 1816.045247] qdsp_cvs_callback: session_id 0x0
<7>[ 1816.045252] qdsp_cvs_callback: Payload Length = 4, opcode=100be
<7>[ 1816.045257] qdsp_cvs_callback: APR_RSP_ACCEPTED for 0x11369:
<7>[ 1816.045565] qdsp_cvs_callback: session_id 0x0
<7>[ 1816.045570] qdsp_cvs_callback: Payload Length = 8, opcode=110e8
<7>[ 1816.045575] 11369 0
```

11369 0 indicates a VSS_ISTREAM_CMD_REGISTER_CALIBRATION_DATA_V2 voice command to the DSP and its callback status. The commands are listed in kernel/sound/soc/msm/qdsp6v2/q6voice.h.

```
<7>[ 1816.045579] qdsp_cvs_callback: cmd = 0x11369
<7>[ 1816.045602] get_vocproc_dev_cfg_cal,
<7>[ 1816.045608] voice_get_cvp_handle: cvp_handle 205
<7>[ 1816.046033] 11371 0
```

11371 0 indicates a VSS_IVOCPROC_CMD_REGISTER_DEVICE_CONFIG command to the DSP and its callback status. The commands are listed in kernel/sound/soc/msm/qdsp6v2/q6voice.h.

```
<7>[ 1816.046056] get_vocproc_cal,
<7>[ 1816.046061] voice_get_cvp_handle: cvp_handle 205
<7>[ 1816.046066] get_voice_col_data,
<7>[ 1816.046605] 11373 0
```

11373 0 indicates a VSS_IVOCPROC_CMD_REGISTER_CALIBRATION_DATA_V2 command to the DSP and its callback status. The commands are listed in kernel/sound/soc/msm/qdsp6v2/q6voice.h.

```
<7>[ 1816.046627] get_vocvol_cal,
<7>[ 1816.046632] voice_get_cvp_handle: cvp_handle 205
<7>[ 1816.046637] get_voice_col_data,
<7>[ 1816.047913] 11374 0
```

11374 0 indicates a VSS_IVOCPROC_CMD_REGISTER_VOL_CALIBRATION_DATA command to the DSP and its callback status. The commands are listed in kernel/sound/soc/msm/qdsp6v2/q6voice.h.

```
<7>[ 1816.047935] voice_get_cvp_handle: cvp_handle 205
<7>[ 1816.047940] cvp_enable_cmd pkt size = 20, cvp_handle=205
<7>[ 1816.056157] 100c6 0
```

100c6 0 indicates a VSS_IVOCPROC_CMD_ENABLE voice command to the DSP and its callback status. The commands are listed in kernel/sound/soc/msm/qdsp6v2/q6voice.h.

```
<7>[ 1816.056187] voice_get_mvm_handle: mvm_handle 32
<7>[ 1816.056193] voice_get_cvp_handle: cvp_handle 205
<7>[ 1816.056197] send mvm_a_vocproc_cmd pkt size = 22
<7>[ 1816.056448] qdsp_mvm_callback: Payload Length = 4, opcode=100be
<7>[ 1816.056454] qdsp_mvm_callback: session_id 0x0
<7>[ 1816.063799] qdsp_mvm_callback: Payload Length = 8, opcode=110e8
<7>[ 1816.063806] qdsp_mvm_callback: session_id 0x0
<7>[ 1816.063811] 1123e 0
```

1123e 0 indicates a VSS_IMVM_CMD_ATTACH_VOCPROC voice command to the DSP and its callback status. The commands are listed in kernel/sound/soc/msm/qdsp6v2/q6voice.h.

```
<7>[ 1816.063815] qdsp_mvm_callback: cmd = 0x1123e
<7>[ 1816.063843] voice_get_mvm_handle: mvm_handle 32
<7>[ 1816.063849] voice_send_tty_mode_cmd: pkt size = 24
<7>[ 1816.063854] tty mode =4
<7>[ 1816.064070] qdsp_mvm_callback: Payload Length = 4, opcode=100be
<7>[ 1816.064076] qdsp_mvm_callback: session_id 0x0
<7>[ 1816.064249] qdsp_mvm_callback: Payload Length = 8, opcode=110e8
<7>[ 1816.064254] qdsp_mvm_callback: session_id 0x0
<7>[ 1816.064258] 11196 0
```

11196 indicates that DSP processed the VSS_ISTREAM_CMD_SET_TTY_MODE voice command. The commands are listed in kernel/sound/soc/msm/qdsp6v2/q6voice.h.

```
<7>[ 1816.064262] qdsp_mvm_callback: cmd = 0x11196
<7>[ 1816.064287] voice_get_cvs_handle: cvs_handle 70
<7>[ 1816.064292] voice_get_cvp_handle: cvp_handle 205
<7>[ 1816.064299] voice_get_cvp_handle: cvp_handle 205
<7>[ 1816.064305] voice_send_vol_step_cmd step_value:1, ramp_duration_ms:20
<7>[ 1816.064775] 112c2 0
```

112c2 0 indicates a VSS_IVOLUME_CMD_SET_STEP voice command and its callback response. For more information on commands, see kernel/sound/soc/msm/qdsp6v2/q6voice.h. For more information on response error codes, see kernel/include/sound/apr_audio-v2.h.

```
<7>[ 1816.064994] voice_get_cvs_handle: cvs_handle 70
<7>[ 1816.065217] qdsp_cvs_callback: session_id 0x0
<7>[ 1816.065222] qdsp_cvs_callback: Payload Length = 4, opcode=100be
<7>[ 1816.065230] qdsp_cvs_callback: APR_RSP_ACCEPTED for 0x1138b:
<7>[ 1816.065978] qdsp_cvs_callback: session_id 0x0
<7>[ 1816.065984] qdsp_cvs_callback: Payload Length = 8, opcode=110e8
<7>[ 1816.065989] 1138b 0
```

1138b 0 indicates a VSS_IVOLUME_CMD_MUTE_V2 voice command and its callback response. For more information on commands, see kernel/sound/soc/msm/qdsp6v2/q6voice.h. For more information on response error codes, see kernel/include/sound/apr_audio-v2.h.

```
<7>[ 1816.065993] qdsp_cvs_callback: cmd = 0x1138b
<7>[ 1816.066026] voice_get_mvm_handle: mvm_handle 32
<7>[ 1816.066038] send mvm_start_voice_cmd pkt size = 20
<7>[ 1816.066275] qdsp_mvm_callback: Payload Length = 4, opcode=100be
<7>[ 1816.066280] qdsp_mvm_callback: session_id 0x0
<7>[ 1816.067585] qdsp_mvm_callback: Payload Length = 8, opcode=110e8
<7>[ 1816.067591] qdsp_mvm_callback: session_id 0x0
<7>[ 1816.067597] 11190 0
```

11190 0 indicates a VSS_IMVM_CMD_START_VOICE voice command callback and its response. For more information on commands, see kernel/sound/soc/msm/qdsp6v2/q6voice. A voice call starts once the DSP returns no error when processing the command.

```
<7>[ 1816.067602] qdsp_mvm_callback: cmd = 0x11190
<7>[ 1816.067688] msm_pcm_capture_prepare
<7>[ 1816.067694] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.067700] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1816.075583] msm-pcm-routing msm-pcm-routing: reg 0
```

```
<7>[ 1816.075590] msm-pcm-routing msm-pcm-routing: reg 0 val 1
<7>[ 1816.075670] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.075676] msm_pcm_ioctl: Falling into default snd_lib_ioctl cmd 0x0
<7>[ 1816.075683] msm_pcm_ioctl: ret 0
<7>[ 1816.075696] msm_pcm_trigger: cmd = 1
<7>[ 1816.075701] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1816.075706] Start & Stop Voice call not handled in Trigger.
<7>[ 1816.075726] msm_voice_gain_put: volume: 1 session_id: 0xffffffff
ramp_duration: 20
<7>[ 1816.075732] voc_set_rx_vol_step session id = 0xffffffff vol = 1
<7>[ 1816.075738] voice_itr_get_next_session : cur idx = 0 session idx = 4
<7>[ 1816.075745] voice_get_cvp_handle: cvp_handle 205
<7>[ 1816.075750] voice_send_vol_step_cmd step_value:1, ramp_duration_ms:20
<7>[ 1816.076160] 112c2 0
```

112c2 0 indicates a VSS_IVOLUME_CMD_SET_STEP command callback and its response. For more information on commands, see kernel/sound/soc/msm/qdsp6v2/q6voice.

```
<7>[ 1816.076186] voice_itr_get_next_session : cur idx = 1 session idx = 4
<7>[ 1816.076194] voice_itr_get_next_session : cur idx = 2 session idx = 4
<7>[ 1816.076200] voice_itr_get_next_session : cur idx = 3 session idx = 4
<7>[ 1816.076207] voice_itr_get_next_session : cur idx = 4 session idx = 4
<7>[ 1816.076213] voice_itr_get_next_session : cur idx = 5 session idx = 4
<7>[ 1818.802778] adm_close port_id=0x4000 index 15 perf_mode: 1
<7>[ 1818.802821] adm_close:Closing ADM: perf_mode: 1
<7>[ 1818.802863] adm_close:coppid 0 portid=0x4000 index=15 coppcnt=0
<7>[ 1818.803616] adm_callback_debug_print: code = 0x110e8 PL#0[10327],
PL#1[0], size = 8
<7>[ 1818.803657] APR_BASIC_RSP_RESULT id 10327
<7>[ 1818.803687] adm_callback: Basic callback received, wake up.
<7>[ 1818.806242] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1818.808038] msm_pcm_routing_process_audio: reg 2 val 4 set 0
<7>[ 1819.484562] msm_voice_mute_put: mute=0 session_id=0xffffffff
ramp_duration=20
<7>[ 1819.484576] voice_itr_get_next_session : cur idx = 0 session idx = 4
<7>[ 1819.484588] voice_get_cvs_handle: cvs_handle 70
<7>[ 1819.484771] qdsp_cvs_callback: session_id 0x0
<7>[ 1819.484779] qdsp_cvs_callback: Payload Length = 4, opcode=100be
<7>[ 1819.484788] qdsp_cvs_callback: APR_RSP_ACCEPTED for 0x1138b:
<7>[ 1819.485032] qdsp_cvs_callback: session_id 0x0
<7>[ 1819.485039] qdsp_cvs_callback: Payload Length = 8, opcode=110e8
<7>[ 1819.485047] 1138b 0
```

1138b 0 indicates a VSS_IVOLUME_CMD_MUTE_V2 command callback and its response. For more information on commands, see kernel/sound/soc/msm/qdsp6v2/q6voice.

15.4.2.2 Voice end call

```
<7>[ 1667.521786] msm_pcm_routing_process_audio: reg 2 val 4 set 1
```

Enable routing of PCM data to BE DAI 2 (MSM_BACKEND_DAI_SLIMBUS_0_RX) from FE DAI MSM_FRONTEND_DAI_MULTIMEDIA4.The BE DAI and FE DAI enums are defined in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[ 1667.522332] get_asm_custom_topology,
<7>[ 1667.524566] adm_open: port 0x4000 path:1 rate:48000 mode:1
perf_mode:1
 .. .
<7>[ 1671.609977] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1671.611737] msm_pcm_routing_process_audio: reg 2 val 4 set 0
```

Disable routing of PCM data to BE DAI 2 (MSM_BACKEND_DAI_SLIMBUS_0_RX) from FE DAI MSM_FRONTEND_DAI_MULTIMEDIA4. The BE DAI and the FE DAI enums are defined in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[ 1719.560469] msm_pcm_trigger: cmd = 0
<7>[ 1719.560516] voc_get_session_id: Voice session has session id
0x10c01000
```

voc_get_session_id() returns the VSID 0x10c01000 (VOICE_VSID), which is defined in /hardware/qcom/audio/hal/voice_extn/voice_extn.c.

```
<7>[ 1719.560551] Start & Stop Voice call not handled in Trigger.
<7>[ 1719.560685] msm_pcm_playback_close
<7>[ 1719.560739] afe_close: port_id=16384
<7>[ 1719.560779] afe_close: port_id 0x4000, mad_type 0
<7>[ 1719.560809] afe_close: Not a MAD port
<7>[ 1719.565814] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
size = 8
<7>[ 1719.565858] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
token=15
<7>[ 1719.565926] afe_callback:port_id = 0
<7>[ 1719.566131] afe_apr_send_pkt: leave 0
<7>[ 1719.585547] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1719.586961] msm_pcm_trigger: cmd = 0
<7>[ 1719.586998] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1719.587030] Start & Stop Voice call not handled in Trigger.
<7>[ 1719.587146] msm_pcm_capture_close
<7>[ 1719.587177] end voice call
<7>[ 1719.587208] voc_get_session_id: Voice session has session id
0x10c01000
```

```
<7>[ 1719.587247] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1719.587288] voc_end_voice_call: VOC_STATE: 1
```

End the voice call. The current VOC_STATE (vocoder state) is 1 (VOC_RUN). VOC_STATE is defined in /kernel/sound/soc/msm/qdsp6v2/q6voice.h.

```
enum {
   VOC_INIT = 0,
   VOC_RUN,
   VOC_CHANGE,
   VOC_RELEASE,
   VOC_ERROR,
   VOC_STANDBY,
};
```

```
<7>[ 1719.587320] voice_get_mvm_handle: mvm_handle 32
<7>[ 1719.587352] voice_get_cvp_handle: cvp_handle 107
<7>[ 1719.587384] voice_get_cvs_handle: cvs_handle 70
<7>[ 1719.587415] voice_cvs_stop_playback: Stop playback already sent
<7>[ 1719.587449] voice_get_cvs_handle: cvs_handle 70
<7>[ 1719.587478] voice_cvs_stop_record: Stop record already sent
<7>[ 1719.587511] voice_get_mvm_handle: mvm_handle 32
<7>[ 1719.587541] send mvm_stop_voice_cmd pkt size = 20
<7>[ 1719.587806] qdsp_mvm_callback: Payload Length = 4, opcode=100be
<7>[ 1719.587841] qdsp_mvm_callback: session_id 0x0
<7>[ 1719.590922] qdsp_mvm_callback: Payload Length = 8, opcode=110e8
<7>[ 1719.590956] qdsp_mvm_callback: session_id 0x0
<7>[ 1719.590984] 11192 0
```

11192 0 indicates the VSS_IMVM_CMD_STOP_VOICE voice command to the DSP and its callback status.

```
<7>[ 1719.591010] qdsp_mvm_callback: cmd = 0x11192
<7>[ 1719.599204] mvm_d_vocproc_cmd  pkt size = 22
<7>[ 1719.600208] qdsp_mvm_callback: Payload Length = 4, opcode=100be
<7>[ 1719.600243] qdsp_mvm_callback: session_id 0x0
<7>[ 1719.605245] qdsp_mvm_callback: Payload Length = 8, opcode=110e8
<7>[ 1719.605281] qdsp_mvm_callback: session_id 0x0
<7>[ 1719.605309] 1123f 0
```

1123f 0 indicates the VSS_IMVM_CMD_DETACH_VOCPROC command to the DSP and its callback status.

```
<7>[ 1719.605335] qdsp_mvm_callback: cmd = 0x1123f
<7>[ 1719.605558] get_vocvol_cal,
<7>[ 1719.605591] voice_get_cvp_handle: cvp_handle 107
<7>[ 1719.605870] 11375 0
```

11375 0 indicates the VSS_IVOCPROC_CMD_DEREGISTER_1
VOL_CALIBRATION_DATA command to the DSP and its callback status.

```
<7>[ 1719.606042] get_vocproc_cal,
<7>[ 1719.606075] voice_get_cvp_handle: cvp_handle 107
<7>[ 1719.606314] 11276 0
```

11276 0 indicates the VSS_IVOCPROC_CMD_DEREGISTER_CALIBRATION_DATA command to the DSP and its callback status.

```
<7>[ 1719.606446] get_vocproc_dev_cfg_cal,
<7>[ 1719.606478] voice_get_cvp_handle: cvp_handle 107
<7>[ 1719.606711] 11372 0
```

11372 0 indicates the VSS_IVOCPROC_CMD_DEREGISTER_DEVICE_CONFIG command to the DSP and its callback status.

```
<7>[ 1719.606841] get_vocstrm_cal,
<7>[ 1719.606871] voice_get_cvs_handle: cvs_handle 70
<7>[ 1719.607096] qdsp_cvs_callback: session_id 0x0
<7>[ 1719.607129] qdsp_cvs_callback: Payload Length = 4, opcode=100be
<7>[ 1719.607164] qdsp_cvs_callback: APR_RSP_ACCEPTED for 0x1127a:
<7>[ 1719.607278] qdsp_cvs_callback: session_id 0x0
<7>[ 1719.607309] qdsp_cvs_callback: Payload Length = 8, opcode=110e8
<7>[ 1719.607340] 1127a 0
```

1127a 0 indicates the VSS_ISTREAM_CMD_DEREGISTER_CALIBRATION_DATA command to the DSP and its callback status.

```
<7>[ 1719.607365] qdsp_cvs_callback: cmd = 0x1127a
<7>[ 1719.607426] cvp_destroy_session_cmd pkt size = 20
<7>[ 1719.610131] 1003c 0
```

1003c 0 indicates the APRV2_IBASIC_CMD_DESTROY_SESSION command to the DSP and its callback status.

```
<7>[ 1719.613816] voice_get_cvs_handle: cvs_handle 70
<7>[ 1719.613862] voice_set_cvp_handle: cvp_handle 0
<7>[ 1719.613947] voice_get_mvm_handle: mvm_handle 32
<7>[ 1719.614749] voice_get_cvs_handle: cvs_handle 70
<7>[ 1719.614810] afe_close: port_id=16385
<7>[ 1719.614843] afe_close: port_id 0x4001, mad_type 0
```

```
<7>[ 1719.614871] afe_close: Not a MAD port
<7>[ 1719.620027] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
size = 8
<7>[ 1719.620070] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
token=16
<7>[ 1719.620134] afe_callback:port_id = 0
<7>[ 1719.620257] afe_apr_send_pkt: leave 0
<7>[ 1719.644859] msm-pcm-routing msm-pcm-routing: reg 0
<7>[ 1719.648180] msm_pcm_routing_process_voice: reg 3 val 9 set 0
```

Disable routing from BE DAI 3(MSM_BACKEND_DAI_SLIMBUS_0_TX) to 9 (MSM_FRONTEND_DAI_CS_VOICE). The BE DAI and FE DAI enums are defined in kernel/sound/soc/msm/qdsp6v2/msm-pcm-routing-v2.h.

```
<7>[ 1719.648221] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1719.650712] msm_pcm_routing_process_voice: FE DAI 0x9 session_id
0x10c01000
<7>[ 1719.650765] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1719.650804] voc_set_route_flag: path_dir=1, set=0
<7>[ 1719.650844] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1719.651193] msm_pcm_routing_process_voice: reg 2 val 9 set 0
```

Disable routing to BE DAI 2(MSM_BACKEND_DAI_SLIMBUS_0_RX) from 9 (MSM_FRONTEND_DAI_CS_VOICE).

```
<7>[ 1719.651230] voc_get_session_id: Voice session has session id
0x10c01000
<7>[ 1719.651266] msm_pcm_routing_process_voice: FE DAI 0x9 session_id
0x10c01000
<7>[ 1719.651310] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1719.651346] voc_set_route_flag: path_dir=0, set=0
<7>[ 1719.651385] voice_get_session:session_id 0x10c01000 session handle
0xc1287594
<7>[ 1720.759643] acdb_ioctl
<7>[ 1720.760947] acdb_ioctl
<7>[ 1720.760976] store_audproc_cal, path = 0
<7>[ 1720.761043] acdb_ioctl
<7>[ 1720.761050] store_audvol_cal, path = 0
<7>[ 1720.761079] acdb_ioctl
<7>[ 1720.761445] store_afe_cal, path = 0
 .. .
```

15.4.3 QXDM Professional logs

See *Hexagon Access Audio/Voice PCM/Bit Stream Logging with QXDM Professional* (80-N3470-4) for QXDM Professional logging.

15.5 Customization

15.5.1 Telephony latency optimization

To enable telephony latency optimizations, the config_speed_up_audio_on_mt_calls config item in the frameworks/base/core/res/res/values/config.xml file is set to true and the code, recompiled.

15.6 Debugging

- Voice mute – Check the logcat logs for device mute or zero volume setting.
- Noise during voice call – Collect the QXDM Professional logs and check the PCM logs at various points in DSP to identify the issue.
- The voice CE and protocol teams analyzes the QXDM Professional logs.

16 VoIP call

This section is not applicable to this release.

17 Audio hardware interfaces

Figure 17-1 illustrates the audio hardware interfaces in the MSM8974.

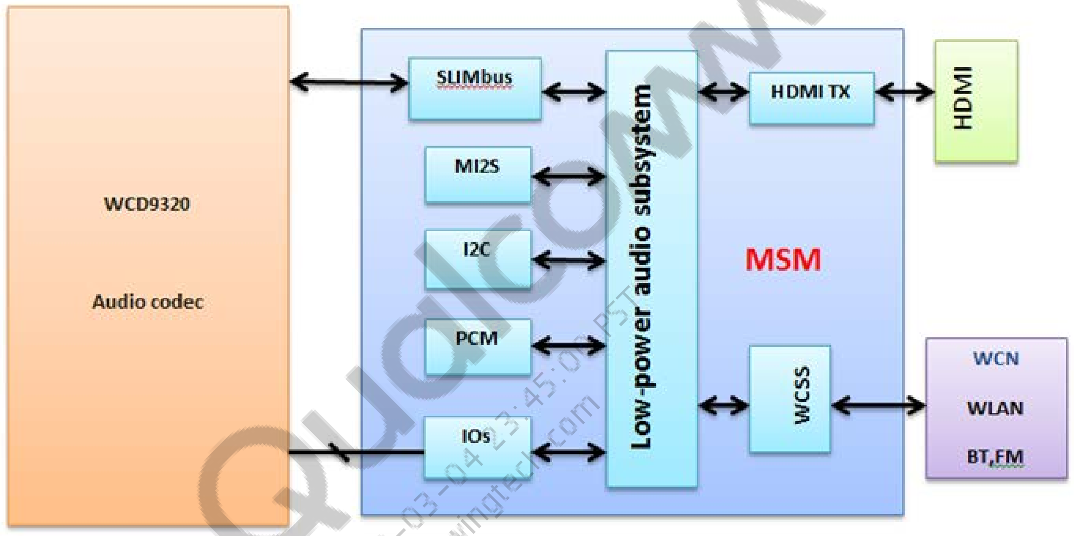


Figure 17-1 Audio hardware interfaces in MSM8974

Table 17-1 lists the GPIO pin assignments for the audio hardware interface in MSM8974.

Table 17-1 GPIO pin assignments for audio hardware interface in MSM8974

Other potential audio assignments	Legacy I2S assignments	PCM assignments	WCD SLIMbus reference designs	MSM8974 GPIO
—	MI2S_4_MCLK	—	General Purpose	57
—	MI2S_4_SCLK	SEC_PCM_CLK	General Purpose	58
—	MI2S_4_WS	SEC_PCM_SYNC	General Purpose	59
—	MI2S_4_SD0	SEC_PCM_DIN	General Purpose	60
—	MI2S_4_SD1	SEC_PCM_DOUT	General Purpose	61
—	MI2S_4_SD2	—	General Purpose	62
—	MI2S_4_SD3	—	General Purpose	63
—	MI2S_1_MCLK	—	General Purpose	64
—	MI2S_1_SCLK	PRI_PCM_CLK	General Purpose	65
—	MI2S_1_WS	PRI_PCM_SYNC	General Purpose	66
—	MI2S_1_SD0	PRI_PCM_DIN	General Purpose	67
—	MI2S_1_SD1	PRI_PCM_DOUT	SLIMBUS_MCLK	68
SPKR_I2S_MCLK	SLIMBUS_MCLK	—	SLIMBUS_CLK	69

Other potential audio assignments	Legacy I2S assignments	PCM assignments	WCD SLIMbus reference designs	MSM8974 GPIO
SPKR_I2S_SCK	SLIMBUS_CLK	—	SLIMBUS_DATA	70
SPKR_I2S_DOUT	SLIMBUS_DATA	—	CODEC_MAD_INT	71
SPKR_I2S_WS	CODEC_MAD_INT	—	General Purpose	72
—	MI2S_3_MCLK	—	General Purpose	73
—	MI2S_3_SCLK	PRI_PCM_CLK	General Purpose	74
—	MI2S_3_WS	PRI_PCM_SYNC	General Purpose	75
—	MI2S_3_SD0	PRI_PCM_DIN	General Purpose	76
—	MI2S_3_SD1	PRI_PCM_DOUT	General Purpose	77
—	MI2S_2_MCLK		General Purpose	78
—	MI2S_2_SCLK	SEC_PCM_CLK	General Purpose	79
—	MI2S_2_WS	SEC_PCM_SYNC	General Purpose	80
—	MI2S_2_SD0	SEC_PCM_DIN	General Purpose	81
—	MI2S_2_SD1	SEC_PCM_DOUT	General Purpose	82

17.1 SLIMbus

A SLIMbus system consists of a set of SLIMbus devices inside the components that communicate with one another using a shared data (DATA) line and a common clock (CLK) signal. The SLIMbus master driver is implemented in the LPASS and is responsible for device enumeration and audio PCM data transfer between LPASS and the WCD codec. The NGD driver on apps controls the channel, port setup on WCD codec and is used for codec register read/write. Inside the SLIMbus component, there are five SLIMbus devices on the MSM side and two devices on the codec side, as shown in Figure 17-2.

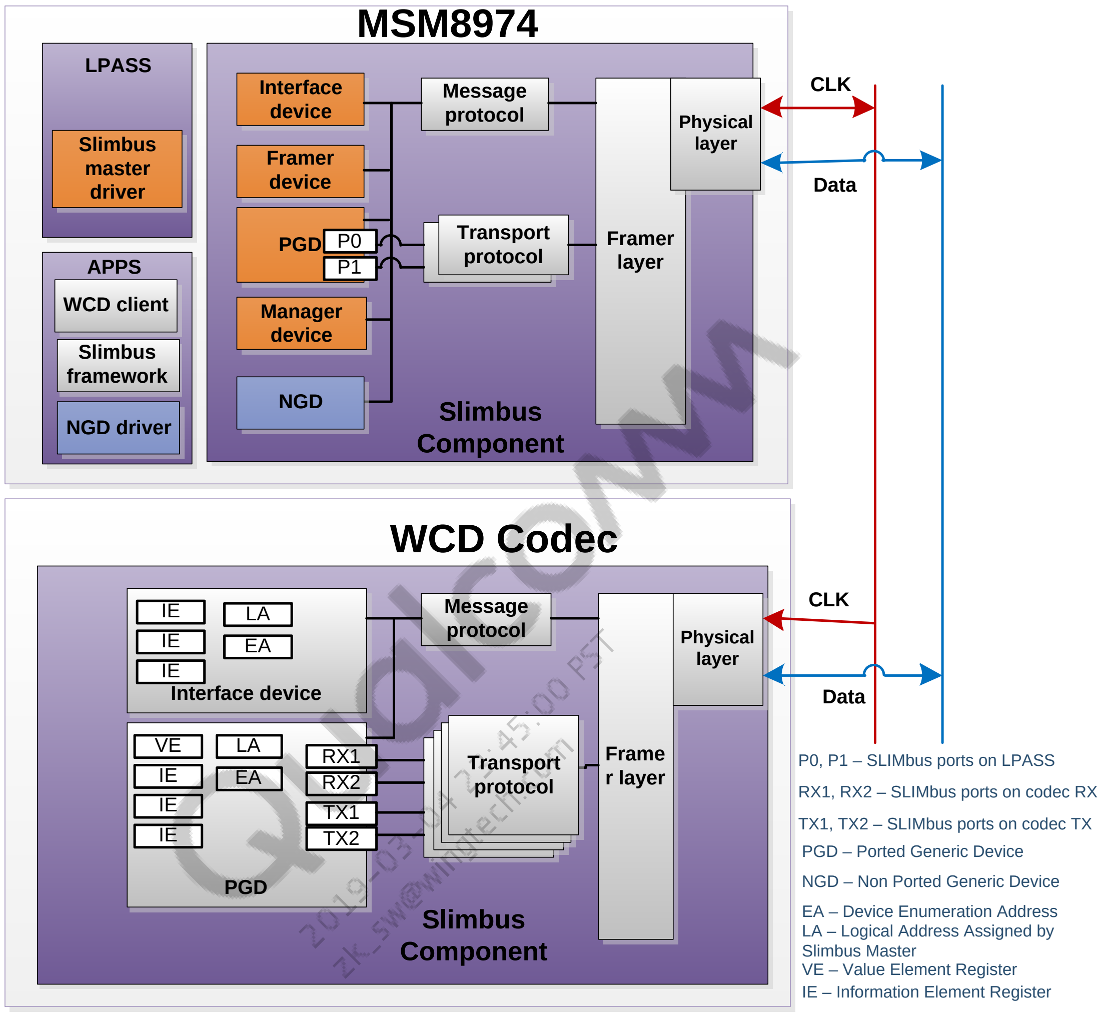


Figure 17-2 SLIMbus system on MSM8974 device

SLIMbus devices on MSM

The five SLIMbus devices on the MSM side are:

- Manager device – Responsible for booting up the SLIMbus; performs bus administration (enumeration of devices, bus configuration, channel allocation)
- Framer device – Provides clock signal on the bus
- Interface device – Provides bus management services; responsible for maintaining the sync between the different devices on SLIMbus; manages component reset, and reports information about status of the component
- Ported generic device (PGD) – Provides the basic SLIMbus functionality for a device. Responsible for configuring the ports and streams, and for the data transfer. For example, AFE connects with SLIMbus ports on LPASS and transfers data between codec and LPASS.
- Nonported generic device (NGD) – Does not have any ports and is controlled from the apps processor using a satellite SLIMbus driver

SLIMbus devices on WCD

The two SLIMbus devices on WCD are:

- PGD – Responsible for configuring ports and streams and for data transfer
- Interface device – Provides bus management services; responsible for maintaining the synchronization between the different devices on SLIMbus

17.1.1 Call flows

This section is not applicable to this release.

17.1.2 Log collection

17.1.2.1 User space logs

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

17.1.2.2 Kernel logs

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file slimbus.c +p > /sys/kernel/debug/dynamic_debug/control
echo file slimbus-msm-ngd.c +p > /sys/kernel/debug/dynamic_debug/control
echo file msm-pcm-routing-v2.c +p > /sys/kernel/debug/dynamic_debug/control
echo file msm8974.c +p > sys/kernel/debug/dynamic_debug/control
echo file wcd9320.c +p > /sys/kernel/debug/dynamic_debug/control
echo file wcd9xxx-slimslave.c +p > /sys/kernel/debug/dynamic_debug/control
echo file wcd9xxx-core.c +p > /sys/kernel/debug/dynamic_debug/control
```

17.1.2.3 Additional logging

In the user space, enable logs in files listed at the following locations, if the facility already exists:

```
\harwdware\qcom\audio\hal
audio_hw.c

\harwdware\qcom\audio\hal\msm8974
platform.c
hw_info.c

\harwdware\qcom\audio\hal\policy_hal
AudioPolicyManager.cpp

\harwdware\libhardware_legacy\audio
AudioPolicyManagerBase.cpp
```

On the kernel side, logs on the following files are enabled to debug issues related to APSS and DSP communication:

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6afe.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6asm.c  +p > /sys/kernel/debug/dynamic_debug/control
```

For issues related to calibration, enable log messages in audio_acdb.c.

```
echo file audio_acdb.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging on q6asm.c results in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

17.1.2.4 QXDM Professional logs

This section is not applicable to this release.

17.1.3 Log analysis

17.1.4 APPS – SLIMbus boot sequence

The SLIMbus probe is called during the bootup.

```
<6>[    5.701466] Slimbus NGD ngd_slim_probe() called
```

The probe is deferred and the SLIMbus driver waits for DSP firmware to load.

```
<6>[    5.705272] platform fe12f000.slim: Driver ngd_msm_ctrl requests
probe deferral
```

DSP firmware is loaded.

```
<6>[    5.959486] adsp_boot_store:going to call adsp_loader_do
<6>[    6.259889] pil-q6v5-lpass fe200000.qcom,lpass: adsp: Brought out of
reset
<6>[    6.265947] adsp_loader_do: Q6/ADSP image is loaded
```

The codec probe retrieves the platform data.

```
<6>[    6.312877] Slimbus NGD ngd_slim_probe() called
<6>[    6.317189] wcd9xxx_slim_probe
<6>[    6.319503] taiko-slim taiko-slim-pgd: Platform data from device tree
<6>[    6.326150] taiko-slim taiko-slim-pgd: cdc-vdd-buck: vol=[2150000
2150000]uV, curr=[650000]uA, ond 0
```

The QMI interface comes up.

```
<6>[    6.335214] Slimbus QMI NGD CB received event:2
```

DSP is initialized.

```
<6>[    6.335422] Slimbus NGD ngd_adsp_up() called
<6>[    6.335427] Slimbus NGD ngd_slim_enable() called 1
<6>[    6.335430] Slimbus.c msm_slim_qmi_init() called
<6>[    6.335693] Slimbus NGD ngd_slim_rx_msgq_thread() called
<6>[    6.347817] apr_tal:Q6 Is Up
```

WCD codec power parameters are set up and the codec is reset. Codec present report is checked.

```
<6>[    6.385073] taiko-slim taiko-slim-pgd: cdc-vdd-tx-h: vol=[1800000
1800000]uV, curr=[25000]uA, ond 0
<6>[    6.385097] taiko-slim taiko-slim-pgd: cdc-vdd-rx-h: vol=[1800000
1800000]uV, curr=[25000]uA, ond 0
<6>[    6.385121] taiko-slim taiko-slim-pgd: cdc-vddpx-1: vol=[1800000
1800000]uV, curr=[10000]uA, ond 0
<6>[    6.385144] taiko-slim taiko-slim-pgd: cdc-vdd-a-1p2v: vol=[1225000
1225000]uV, curr=[10000]uA, ond 0
<6>[    6.385168] taiko-slim taiko-slim-pgd: cdc-vddcx-1: vol=[1225000
1225000]uV, curr=[10000]uA, ond 0
<6>[    6.385192] taiko-slim taiko-slim-pgd: cdc-vddcx-2: vol=[1225000
1225000]uV, curr=[10000]uA, ond 0
<6>[    6.385235] wcd9xxx_init_supplies
<6>[    6.449121] Slimbus.c msm_slim_qmi_send_select_inst_req() called
<6>[    6.450115] wcd9xxx_enable_static_supplies: Enabled regulator cdc-
vdd-buck
<6>[    6.450124] wcd9xxx_enable_static_supplies: Enabled regulator cdc-
vdd-tx-h
<6>[    6.450132] wcd9xxx_enable_static_supplies: Enabled regulator cdc-
vdd-rx-h
<6>[    6.450138] wcd9xxx_enable_static_supplies: Enabled regulator cdc-
vddpx-1
<6>[    6.450868] wcd9xxx_enable_static_supplies: Enabled regulator cdc-
vdd-a-1p2v
```

```
<6>[    6.450899] wcd9xxx_enable_static_supplies: Enabled regulator cdc-
vddcx-1
<6>[    6.450904] wcd9xxx_enable_static_supplies: Enabled regulator cdc-
vddcx-2
<6>[    6.450944] wcd9xxx_reset

<6>[    6.507131] Slimbus.c msm_slim_qmi_notify() called
<6>[    6.507189] wcd9xxx_slim_get_laddr
```

aDSP SLIMbus is not yet up.

```
<6>[    6.507196] Slimbus NGD ngd_get_tid() called
<6>[    6.507202] Slimbus NGD ngd_xfer_msg() called
<3>[    6.507207] ngd_msm_ctrl fe12f000.slim: ADSP slimbus not up yet
```

SLIMbus clock is on.

```
<6>[    6.528695] Slimbus.c msm_slim_qmi_recv_msg() called
<6>[    6.533673] slim_clk_pause: txn-rsp for 0 pending
<6>[    6.538329] Slimbus NGD ngd_slim_power_up() called
<6>[    6.543080] Slimbus.c msm_slim_qmi_power_request() called
<6>[    6.548497] Slimbus.c msm_slim_qmi_send_power_request() called
<6>[    6.556237] Slimbus.c msm_slim_qmi_notify() called
<6>[    6.560003] Slimbus.c msm_slim_qmi_recv_msg() called
```

Master capability is received from aDSP.

```
<6>[    6.565370] Slimbus.c msm_slim_rx_enqueue() called
<6>[    6.569727] Slimbus.c msm_slim_rx_dequeue() called
<6>[    6.574480] Slimbus NGD ngd_slim_rx() called
<6>[    6.578755] SLIM SAT: Received master capability
```

The WCD codec comes up and registers itself. As part of that process, the codec type is also detected.

```
<6>[    6.583334] Slimbus NGD ngd_slim_setup_msg_path() called
<6>[    6.588648] Slimbus.c msm_slim_sps_init() called
<6>[    6.593330] sps:BAM 0xfe104000 (va:0xf2b00000) enabled: ver:0x13,
number of pipes:31
<6>[    6.601162] sps:BAM 0xfe104000 is registered.
<6>[    6.605296] Slimbus.c msm_slim_init_rx_msgq() called
<6>[    6.610269] Slimbus.c msm_slim_init_endpoint() called
<6>[    6.615285] Slimbus.c msm_slim_sps_mem_alloc() called
<6>[    6.620894] Slimbus.c msm_slim_sps_mem_alloc() called
<6>[    6.625520] Slimbus.c msm_slim_connect_endp() called
```

```
<6>[    6.792726] Slimbus.c msm_slim_init_tx_msgq() called
<6>[    6.797602] Slimbus.c msm_slim_init_endpoint() called
<6>[    6.802572] Slimbus.c msm_slim_sps_mem_alloc() called
<6>[    6.807898] Slimbus.c msm_slim_sps_mem_alloc() called
<6>[    6.812738] Slimbus.c msm_slim_connect_endp() called
<6>[    6.817855] Slimbus NGD ngd_xfer_msg() called
<6>[    6.822024] Slimbus NGD ngd_laddr_lookup() called
<6>[    6.822039] Slimbus NGD ngd_slim_runtime_resume() called
<6>[    6.822160] Slimbus NGD ngd_xferandwait_ack() called
<6>[    6.836868] index:0, phys:0x36cc7000, virt:0xee8c7000
<6>[    6.842064] index:1, phys:0x36cc7004, virt:0xee8c7004
<6>[    6.847068] index:2, phys:0x36cc7008, virt:0xee8c7008
<6>[    6.852125] Slimbus NGD ngd_get_tid() called
<6>[    6.856174] Slimbus NGD ngd_xfer_msg() called
<6>[    6.860564] Slimbus NGD ngd_slim_rx() called
<6>[    6.864808] slimbus:1 laddr:0xcb, EAPC:0x1:0xa0
<6>[    6.869414] Slimbus NGD ngd_xferandwait_ack() called
<6>[    6.869480] wcd9xxx_slim_device_up
<6>[    6.869514] index:3, phys:0x36cc700c, virt:0xee8c700c
<6>[    6.869534] index:4, phys:0x36cc7010, virt:0xee8c7010
<6>[    6.869551] index:5, phys:0x36cc7014, virt:0xee8c7014
<6>[    6.869566] Slimbus NGD ngd_slim_rx() called
<6>[    6.869805] wcd9xxx_slim_get_laddr
<6>[    6.869811] Slimbus NGD ngd_get_tid() called
<6>[    6.869815] Slimbus NGD ngd_xfer_msg() called
<6>[    6.869932] Slimbus NGD ngd_xferandwait_ack() called
<6>[    6.870284] index:6, phys:0x36cc7018, virt:0xee8c7018
<6>[    6.870294] index:7, phys:0x36cc701c, virt:0xee8c701c
<6>[    6.870302] index:8, phys:0x36cc7020, virt:0xee8c7020
<6>[    6.870309] Slimbus NGD ngd_slim_rx() called
<6>[    6.870338] slimbus:1 laddr:0xca, EAPC:0x0:0xa0
<6>[    6.870345] wcd9xxx_device_init
<6>[    6.870354] wcd9xxx_bring_up
<6>[    6.870370] wcd9xxx_slim_write_device
<6>[    6.870383] Slimbus NGD ngd_xfer_msg() called
<6>[    6.870433] wcd9xxx_slim_write_device
<6>[    6.870438] Slimbus NGD ngd_xfer_msg() called
<6>[    6.876900] wcd9xxx_slim_write_device
<6>[    6.876919] Slimbus NGD ngd_xfer_msg() called
<6>[    6.877045] wcd9xxx_slim_write_device
<6>[    6.877051] Slimbus NGD ngd_xfer_msg() called
<6>[    6.877219] wcd9xxx_check_codec_type
<6>[    6.877222] wcd9xxx_bulk_read
<6>[    6.877226] wcd9xxx_slim_read_device
<6>[    6.877231] Slimbus NGD ngd_xfer_msg() called
<6>[    6.877389] index:9, phys:0x36cc7024, virt:0xee8c7024
<6>[    6.877409] index:10, phys:0x36cc7028, virt:0xee8c7028
<6>[    6.877424] Slimbus NGD ngd_slim_rx() called
```

```
<6>[    6.877440] wcd9xxx_bulk_read
<6>[    6.877445] wcd9xxx_slim_read_device
<6>[    6.877453] Slimbus NGD ngd_xfer_msg() called
<6>[    6.878237] index:11, phys:0x36cc702c, virt:0xee8c702c
<6>[    6.878259] index:12, phys:0x36cc7030, virt:0xee8c7030
<6>[    6.878275] Slimbus NGD ngd_slim_rx() called
<6>[    6.878317] taiko-slim taiko-slim-pgd: wcd9xxx_check_codec_type:
detected taiko_codec, major 0x102, minor 0x1, ver 0x2
```

Configure the codec Tx and Rx ports.

```
<6>[    6.883999] taiko_codec taiko_codec: taiko_codec_probe()
<6>[    6.987842] wcd9xxx-slimslave.c wcd9xxx_init_slimslave() called
<6>[    6.987846] wcd9xxx-slimslave.c wcd9xxx_configure_ports() called
```

Rx and Tx channels are allocated.

```
<6>[    6.987850] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
<6>[    6.987855] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[0].port[16]
<6>[    6.987858] channels[0].sph[-889126896] path[1]
<6>[    6.987867] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[1].port[17]
<6>[    6.987870] channels[1].sph[-889126895] path[1]
<6>[    6.987875] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[2].port[18]
<6>[    6.987878] channels[2].sph[-889126894] path[1]
<6>[    6.987883] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[3].port[19]
<6>[    6.987885] channels[3].sph[-889126893] path[1]
<6>[    6.987891] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[4].port[20]
<6>[    6.987893] channels[4].sph[-889126892] path[1]
<6>[    6.987899] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[5].port[21]
<6>[    6.987901] channels[5].sph[-889126891] path[1]
<6>[    6.987907] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[6].port[22]
<6>[    6.987909] channels[6].sph[-889126890] path[1]
<6>[    6.987914] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[7].port[23]
<6>[    6.987917] channels[7].sph[-889126889] path[1]
<6>[    6.987922] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[8].port[24]
<6>[    6.987925] channels[8].sph[-889126888] path[1]
<6>[    6.987930] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[9].port[25]
```

```
<6>[    6.987932] channels[9].sph[-889126887] path[1]
<6>[    6.987938] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[10].port[26]
<6>[    6.987940] channels[10].sph[-889126886] path[1]
<6>[    6.987945] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[11].port[27]
<6>[    6.987948] channels[11].sph[-889126885] path[1]
<6>[    6.987953] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[12].port[28]
<6>[    6.987955] channels[12].sph[-889126884] path[1]
<6>[    6.987960] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
<6>[    6.987964] wcd9xxx_alloc_slim_sh_ch: pgd_la[203] channels[0].port[0]
<6>[    6.987965] channels[0].sph[-889192448] path[0]
<6>[    6.987971] wcd9xxx_alloc_slim_sh_ch: pgd_la[203] channels[1].port[1]
<6>[    6.987973] channels[1].sph[-889192447] path[0]
<6>[    6.987979] wcd9xxx_alloc_slim_sh_ch: pgd_la[203] channels[2].port[2]
<6>[    6.987981] channels[2].sph[-889192446] path[0]
<6>[    6.987986] wcd9xxx_alloc_slim_sh_ch: pgd_la[203] channels[3].port[3]
<6>[    6.987988] channels[3].sph[-889192445] path[0]
<6>[    6.987994] wcd9xxx_alloc_slim_sh_ch: pgd_la[203] channels[4].port[4]
<6>[    6.987996] channels[4].sph[-889192444] path[0]
<6>[    6.988002] wcd9xxx_alloc_slim_sh_ch: pgd_la[203] channels[5].port[5]
<6>[    6.988004] channels[5].sph[-889192443] path[0]
<6>[    6.988010] wcd9xxx_alloc_slim_sh_ch: pgd_la[203] channels[6].port[6]
<6>[    6.988012] channels[6].sph[-889192442] path[0]
<6>[    6.988017] wcd9xxx_alloc_slim_sh_ch: pgd_la[203] channels[7].port[7]
<6>[    6.988019] channels[7].sph[-889192441] path[0]
<6>[    6.988025] wcd9xxx_alloc_slim_sh_ch: pgd_la[203] channels[8].port[8]
<6>[    6.988027] channels[8].sph[-889192440] path[0]
<6>[    6.988032] wcd9xxx_alloc_slim_sh_ch: pgd_la[203] channels[9].port[9]
<6>[    6.988034] channels[9].sph[-889192439] path[0]
<6>[    6.988040] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[10].port[10]
<6>[    6.988042] channels[10].sph[-889192438] path[0]
<6>[    6.988048] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[11].port[11]
<6>[    6.988050] channels[11].sph[-889192437] path[0]
<6>[    6.988056] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[12].port[12]
<6>[    6.988058] channels[12].sph[-889192436] path[0]
<6>[    6.988064] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[13].port[13]
<6>[    6.988066] channels[13].sph[-889192435] path[0]
<6>[    6.988071] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[14].port[14]
<6>[    6.988073] channels[14].sph[-889192434] path[0]
<6>[    6.988079] wcd9xxx_alloc_slim_sh_ch: pgd_la[203]
channels[15].port[15]
```

SLIMbus then moves into the idle state.

```
<6>[    7.235193] Slimbus NGD ngd_slim_runtime_idle() called
<6>[    8.217202] Slimbus NGD ngd_slim_runtime_suspend() called
<6>[    8.217210] Slimbus NGD ngd_xfer_msg() called
<6>[    8.217215] Slimbus NGD ngd_xfer_msg() called
<6>[    8.217218] Slimbus NGD ngd_xfer_msg() called
<6>[    8.217221] Slimbus.c msm_slim_disconnect_endp() called
<6>[    8.217275] Slimbus.c msm_slim_disconnect_endp() called
<6>[    8.217320] Slimbus.c msm_slim_qmi_power_request() called
<6>[    8.217323] Slimbus.c msm_slim_qmi_send_power_request() called
<6>[    8.217841] Slimbus.c msm_slim_qmi_notify() called
<6>[    8.217862] Slimbus.c msm_slim_qmi_recv_msg() called
```

17.1.5 LPASS – SLIMbus boot sequence

The LPASS SLIMbus driver gets into Master mode per the QMI command from the HLOS.

```
0xC650191: [INFO] Driver Parameters (txq: 1) (rxq: 0x1) (lports: 21)
(pbase: 0) (cbase: 1)
0xC652D1A: [INFO] *** Driver Initialization (master: 0) (protocols: 0x1)
(workarounds: 0x100E)
0xC7EF976: [INFO] Switching driver mode (master: 1)
```

The SLIMbus device is opened and the SLIMbus clock is enabled.

```
0xC8220F5: [INFO] Slimbus device opened (client: 0x) (#open 1)
0xC822380: [INFO] Turning on slimbus ref clock
0xC826570: [INFO] Hardware initialization completed
```

The SLIMbus devices report present to the master with different Enumeration Addresses (EAs) and the master driver assigns different corresponding Logical Addresses (LAs).

```
0xC832384: [INFO] Assigning device logical address (EA: 0x021700A00100)
(DC: 0x00) (LA: 0xCB)
0xC832568: [INFO] Assigning device logical address (EA: 0x021700300300)
(DC: 0x00) (LA: 0xC2)
0xC8327A6: [INFO] Assigning device logical address (EA: 0x021700A00000)
(DC: 0xFD) (LA: 0xCA)
0xC8328E0: [INFO] Assigning device logical address (EA: 0x021700300500)
(DC: 0x00) (LA: 0xC4)
0xC832A00: [INFO] Assigning device logical address (EA: 0x021700300000)
(DC: 0xFD) (LA: 0xC0)
0xC832B24: [INFO] Assigning device logical address (EA: 0x021700300100)
(DC: 0xFE) (LA: 0xC1)
```

```
0xC887C21: [INFO] Received report satellite message (LA: 0xC2) (uSatProto:
0x1)
```

The HLOS SLIMbus NGD driver queries the codec LA through the QMI interface. The SLIMbus PGD present inside the WCD codec reports the following:

```
0xC887E30: [INFO] Got satellite LA query (satLA: 0xC2) (EA: 0x021700A00100)
(LA: 0xCB)
0xC888184: [INFO] Got satellite LA query (satLA: 0xC2) (EA: 0x021700A00100)
(LA: 0xCB)
```

The SLIMbus interface device inside the WCD codec reports the following:

```
0xC89C0D8: [INFO] Got satellite LA query (satLA: 0xC2) (EA: 0x021700A00000)
(LA: 0xCA)
0xC89C2D1: [INFO] Got satellite LA query (satLA: 0xC2) (EA: 0x021700A00000)
(LA: 0xCA)
```

Another SLIMbus device is opened and transactions are performed.

```
0xD3211B9: [INFO] Slimbus device opened (client: 0x) (#open 2)
0xD32142E: [INFO] Got LA query (client: 0x) (EA: 0x021700A00100)
0xD32162B: [INFO] Slimbus device closed (client: 0x) (#open 1)

0xD3217C6: [INFO] Slimbus device opened (client: 0x) (#open 2)
0xD3218F1: [INFO] Got IE/VE transaction (LA: 0xCB) (addr: 0x8AF) (#write:
0) (#read: 1)
0xD32234A: [INFO] Got IE/VE transaction (LA: 0xCB) (addr: 0x8AF) (#write:
1) (#read: 0)
0xD322CAD: [INFO] Slimbus device closed (client: 0x) (#open 1)

0xD322DAB: [INFO] Slimbus device opened (client: 0x) (#open 2)
0xD322EA3: [INFO] Got IE/VE transaction (LA: 0xCB) (addr: 0x8AF) (#write:
0) (#read: 1)
0xD3238F4: [INFO] Got IE/VE transaction (LA: 0xCB) (addr: 0x8AF) (#write:
1) (#read: 0)
```

When all open SLIMbus devices are closed, the SLIMbus clock is disabled.

```
0xD324231: [INFO] Slimbus device closed (client: 0x) (#open 1)
0xDDBC41C: [INFO] Slimbus device closed (client: 0x) (#open 0)
0xDDBC79D: [INFO] Processing reconfiguration sequence
0xDDBC868: [INFO] Initiating transition to idle pause clock at next
reconfig boundary
0xDDBF2F1: [INFO] Framer entered idle pause clock
```

```
0xDDBF3E1: [INFO] Turning off slimbus ref clock
0xDDC0714: [INFO] Bandwidth utilization stats (used data+msg slots: 0+30 /
1536) (CG: 9) (SM: 0)
```

17.1.6 APPS – SLIMbus playback sequence

The SLIMbus clock is enabled.

```
<7>[ 1090.804990] slim_rx_mux_put: wname SLIM RX1 MUX cname SLIM RX1 MUX
value 0 shift 0 item 1
<6>[ 1090.805009] wcd9xxx_rx_vport_validation: port_id 16
<6>[ 1090.809050] wcd9xxx_slim_write_device
<7>[ 1090.809071] slimbus sb-1: SB xfer msg:os:b80, len:1, MC:68, sl:0
<6>[ 1090.809087] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.809102] Slimbus NGD ngd_slim_runtime_resume() called
<6>[ 1090.809115] Slimbus NGD ngd_clk_pause_wakeup() called
<6>[ 1090.809125] Slimbus NGD ngd_slim_power_up() called
<6>[ 1090.809134] Slimbus.c msm_slim_qmi_power_request() called
<6>[ 1090.809144] Slimbus.c msm_slim_qmi_send_power_request() called
<6>[ 1090.810622] Slimbus.c msm_slim_qmi_notify() called
<6>[ 1090.810671] Slimbus.c msm_slim_qmi_recv_msg() called
<6>[ 1090.810766] Slimbus NGD ngd_slim_setup_msg_path() called
<6>[ 1090.810778] Slimbus.c msm_slim_connect_endp() called
<6>[ 1090.811807] Slimbus.c msm_slim_connect_endp() called
<6>[ 1090.812329] Slimbus NGD ngd_slim_runtime_idle() called
<6>[ 1090.814740] wcd9xxx_slim_write_device
```

The AFE port enable command is sent to the DSP with channel 144.

```
<7>[ 1090.821471] taiko_startup(): substream = subdevice #0  stream = 0
<7>[ 1090.821651] taiko_get_channel_map: slot_num 0 ch->ch_num 144
<7>[ 1090.821664] taiko_get_channel_map: rx_num 1
<7>[ 1090.821680] taiko_hw_params: dai_name = taiko_rx1 DAI-ID 0 rate 48000
num_ch 1
<7>[ 1090.821702] taiko_set_interpolator_rate: AIF_PB DAI(0) connected to
RX1
<7>[ 1090.821715] taiko_set_interpolator_rate: set RX1 sample rate to 48000
<6>[ 1090.821756] wcd9xxx_get_slave_port: ch_num[144] slave port[16]
<7>[ 1090.821771] taiko_codec taiko_codec: taiko_set_rxsb_port_format:
sb_ctl_reg 3ae field_shift 0
<7>[ 1090.832697] afe_port_start: port id: 0x4000

<7>[ 1090.845853] taiko_mclk_enable: mclk_enable = 1, dapm = 1
<7>[ 1090.845866] taiko_mclk_enable: Acquiring BG_CLK
<7>[ 1090.845879] taiko_mclk_enable: Acquiring BG_CLK done
<6>[ 1090.845906] wcd9xxx_slim_write_device
<7>[ 1090.845925] slimbus sb-1: SB xfer msg:os:901, len:1, MC:68, sl:0
```

```
<7>[ 1090.855710] taiko_codec_enable_slimrx: event called! codec name
taiko_codec num_dai 8
<7>[ 1090.855712] stream name AIF1 Playback event 2
<7>[ 1090.855718] taiko_codec_enable_slimrx: w->name AIF1 PB w->shift 0
event 2
<6>[ 1090.855724] wcd9xxx_get_slave_port: ch_num[144] slave port[16]
<6>[ 1090.855729] wcd9xxx-slimslave.c wcd9xxx_cfg_slim_sch_rx() called
<6>[ 1090.855733] list ch->ch_h 16 ch->sph -889126896
<6>[ 1090.855738] wcd9xxx_cfg_slim_sch_rx: ch_cnt[1] rate=48000
WATER_MARK_VAL 5
```

Channel definition is performed.

```
<6>[ 1090.855743] Before slim_define_ch:
<6>[ 1090.855744] ch_cnt 1,ch_h[0] 16 ch_h[1] 0, grph 272
<7>[ 1090.855751] slimbus sb-1: define_ch: ch:16, state:1
<7>[ 1090.855756] slimbus sb-1: ch:16, chan PR rate:83
<7>[ 1090.855762] slimbus sb-1: define_ch: ch:16, ret:0
<6>[ 1090.855768] wcd9xxx_cfg_slim_sch_rx: codec_port 16 rx 0xed3bc800,
payload 1
<6>[ 1090.855770] sh_ch.rx_port_ch_reg_base0 0x140
<6>[ 1090.855772] sh_ch.port_rx_cfg_reg_base 0x30
<6>[ 1090.855776] wcd9xxx_interface_reg_write
<6>[ 1090.855781] wcd9xxx_slim_write_device
<7>[ 1090.855787] slimbus sb-1: SB xfer msg:os:980, len:1, MC:68, sl:0
<6>[ 1090.855792] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.855940] wcd9xxx_interface_reg_write
<6>[ 1090.855945] wcd9xxx_slim_write_device
<7>[ 1090.855951] slimbus sb-1: SB xfer msg:os:840, len:1, MC:68, sl:0
<6>[ 1090.855956] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.856101] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.856107] Slimbus NGD ngd_get_tid() called
<6>[ 1090.856678] index:4, phys:0x36cc7010, virt:0xee8c7010
<6>[ 1090.856688] index:5, phys:0x36cc7014, virt:0xee8c7014
<6>[ 1090.856695] Slimbus NGD ngd_slim_rx() called
<7>[ 1090.856712] slimbus sb-1: chan:16,ctrl:0,def:0
<6>[ 1090.856719] Slimbus NGD ngd_allocbw() called
<6>[ 1090.856724] Slimbus NGD ngd_get_tid() called
<6>[ 1090.856728] slim define chan:144, tid:0x27

<6>[ 1090.856760] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.856940] Slimbus NGD ngd_xferandwait_ack() called
<6>[ 1090.857283] index:6, phys:0x36cc7018, virt:0xee8c7018
<6>[ 1090.857293] index:7, phys:0x36cc701c, virt:0xee8c701c
<6>[ 1090.857300] Slimbus NGD ngd_slim_rx() called
<6>[ 1090.857314] Slimbus NGD ngd_get_tid() called
<6>[ 1090.857319] Slimbus NGD ngd_xfer_msg() called
```

```
<6>[ 1090.857459] Slimbus NGD ngd_xferandwait_ack() called
<6>[ 1090.859711] index:8, phys:0x36cc7020, virt:0xee8c7020
<6>[ 1090.859723] index:9, phys:0x36cc7024, virt:0xee8c7024
<6>[ 1090.859730] Slimbus NGD ngd_slim_rx() called
<6>[ 1090.859748] Slimbus NGD ngd_xfer_msg() called
```

Port reconfiguration and playback are performed.

```
<7>[ 1090.859756] slimbus sb-1: sending begin_reconfig:ret:0
<7>[ 1090.859763] slimbus sb-1: define content, activate:90, 83, 0, 4
<6>[ 1090.859767] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.859771] Slimbus NGD ngd_xfer_msg() called
<7>[ 1090.859778] slimbus sb-1: new-intr:0, old-intr:0, dist:0
<7>[ 1090.859783] slimbus sb-1: new-off:0, old-off:0
<6>[ 1090.859787] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.859791] Slimbus NGD ngd_xfer_msg() called
<7>[ 1090.859796] slimbus sb-1: reconfig now:ret:0
<6>[ 1090.859809] wcd9xxx_slim_write_device
<7>[ 1090.859815] slimbus sb-1: SB xfer msg:os:9bc, len:1, MC:68, sl:0
<6>[ 1090.859820] Slimbus NGD ngd_xfer_msg() called
<7>[ 1090.859980] taiko_codec_enable_ear_pa EAR PA 2
<6>[ 1090.859987] wcd9xxx_slim_write_device
<7>[ 1090.859994] slimbus sb-1: SB xfer msg:os:985, len:1, MC:68, sl:0
<6>[ 1090.859999] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.860146] wcd9xxx_slim_write_device
<7>[ 1090.860153] slimbus sb-1: SB xfer msg:os:994, len:1, MC:68, sl:0
<6>[ 1090.860158] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.860301] wcd9xxx_slim_write_device
<7>[ 1090.860309] slimbus sb-1: SB xfer msg:os:983, len:1, MC:68, sl:0
<6>[ 1090.860313] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.860464] wcd9xxx_slim_write_device
<7>[ 1090.860471] slimbus sb-1: SB xfer msg:os:983, len:1, MC:68, sl:0
<6>[ 1090.860476] Slimbus NGD ngd_xfer_msg() called
<6>[ 1090.885870] qtaguid: ctrl_counterset(s 1 10050): insufficient priv
from pid=1015 tgid=929 uid=1000
<6>[ 1091.335441] qtaguid: ctrl_counterset(s 0 10040): insufficient priv
from pid=1015 tgid=929 uid=1000
<6>[ 1092.948130] msm_compr_open: session ID 2
```

When playback stops, the AFE port is closed.

```
<7>[ 1107.308416] afe_close: port_id=16384
<7>[ 1107.308430] afe_close: port_id 0x4000, mad_type 0
<7>[ 1107.308441] afe_close: Not a MAD port
<7>[ 1107.315393] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
size = 8
```

```
<7>[ 1107.315407] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
token=15
<7>[ 1107.315425] afe_callback:port_id = 0
<7>[ 1107.315459] afe_apr_send_pkt: leave 0
<7>[ 1107.315471] taiko_shutdown(): substream = subdevice #0  stream = 0
<6>[ 1107.317946] Slimbus NGD ngd_slim_runtime_idle() called
<7>[ 1107.318871] taiko_codec_enable_slimrx: event called! codec name
taiko_codec num_dai 8
<7>[ 1107.318877] stream name AIF1 Playback event 8
<7>[ 1107.318889] taiko_codec_enable_slimrx: w->name AIF1 PB w->shift 0
event 8
```

The channel is closed.

```
<6>[ 1107.318901] wcd9xxx_close_slim_sch_rx ch_cht 1, sph[0] -889126896
sph[1] 0
<6>[ 1107.318911] wcd9xxx_close_slim_sch_rx before slim_control_ch grph 272
```

The SLIMbus Rx port is closed.

```
<6>[ 1107.325520] Slimbus NGD ngd_slim_rx() called
<7>[ 1107.325551] taiko_slimbus_irq: RX port 0 closed value 4, bit 16
<7>[ 1107.325562] taiko_slimbus_irq: priv->dai[0].ch_mask = 0x10000
```

The SLIMbus clock is disabled.

```
<6>[ 1107.343516] Slimbus NGD ngd_xfer_msg() called
<6>[ 1107.343749] Slimbus NGD ngd_slim_runtime_idle() called
<6>[ 1109.007822] Slimbus NGD ngd_slim_runtime_suspend() called
<6>[ 1109.007881] Slimbus NGD ngd_xfer_msg() called
<6>[ 1109.007915] Slimbus NGD ngd_xfer_msg() called
<6>[ 1109.007943] Slimbus NGD ngd_xfer_msg() called
<6>[ 1109.007967] Slimbus.c msm_slim_disconnect_endp() called
<6>[ 1109.008095] Slimbus.c msm_slim_disconnect_endp() called
<6>[ 1109.008202] Slimbus.c msm_slim_qmi_power_request() called
<6>[ 1109.008230] Slimbus.c msm_slim_qmi_send_power_request() called
<6>[ 1109.009218] Slimbus.c msm_slim_qmi_notify() called
<6>[ 1109.009336] Slimbus.c msm_slim_qmi_recv_msg() called
```

17.1.7 LPASS – SLIMbus playback sequence

The following logs indicate the HLOS satellite SLIMbus driver request SLIMBUS_PM_ACTIVE_V01 for SLIMbus clock enable through the SLIMbus DAL driver in the ADSP side client. 0xF096FDB0 is the DAL client handle.

```
MSG        [08500/01] QDSP6/Medium                      00:01:11.645
SlimBus.c  03135  [INFO] Slimbus device opened (client: 0xF096FDB0) (#open
1)
MSG        [08500/01] QDSP6/Medium                      00:01:11.645
SlimBusMaster.c  02569  [INFO] Turning on slimbus ref clock
MSG        [08500/01] QDSP6/Medium                      00:01:11.645
SlimBusMaster.c  02578  [INFO] Forcing wakeup from idle pause clock
```

This GPIO is used to hook up the external interrupt to request the SLIMbus clock, for example, APQ8084+MDM9x25. Deregister here, as the clock is on.

```
MSG        [08500/00] QDSP6/Low                         00:01:11.645
SlimBusDal.c  01236  [INFO] Deregistering for GPIO interrupt (input: 1)
MSG        [08500/01] QDSP6/Medium                      00:01:11.645
SlimBusMaster.c  02708  [INFO] Framer restarted following idle pause clock
```

aDSP AFE driver opens SLIMbus here, since the pipe configuration is done locally in the aDSP side. Channel 144 is SLIM_RX_7.

```
MSG        [08500/01] QDSP6/Medium                      00:01:11.658
SlimBus.c  03135  [INFO] Slimbus device opened (client: 0xF0940980) (#open
2)
MSG        [08500/02] QDSP6/High                        00:01:11.658
AFESlimbusDriverUtils.cpp  00126  num_descptrs: 0x3, watermark: 0
MSG        [08500/00] QDSP6/Low                         00:01:11.658
SlimBusBamLib.c  00954  [INFO] Configure master pipe (client: 0xF0940980)
(hport: 0x15000) (flow: 0)
MSG        [08500/01] QDSP6/Medium                      00:01:11.658
SlimBus.c  03337  [INFO] detected SB core (version: 0x0) (workarounds:
0x100e)
MSG        [08500/00] QDSP6/Low                         00:01:11.658
SlimBus.c  05946  [INFO] Got connect master port request (client:
0xF0940980) (chanres: 0x65310) (portres: 0x15000) (channel: 144) (port: 0)
(flow: 0)
```

Channel group definition happens next. Multiple ports could be in the same group to avoid drifting in underflow in one of the channels, for example, two ports for stereo audio playback. There is only one channel for SLIMbus port 0 on MSM.

```
MSG        [08500/00] QDSP6/Low                         00:01:11.660
SlimBus.c  05518  [INFO] Got define channel request (client: 0xF0940980)
```

```
(resource: 0x55200) (channel: 144) (#channels: 1) (protocol: 1) (baserate:
2) (ratemult: 48000) (bitwidth: 16)
MSG        [08500/00] QDSP6/Low                         00:01:11.660
SlimBus.c  06666  [INFO] Got reconfigure now request (client: 0xF0940980)
MSG        [08500/00] QDSP6/Low                         00:01:11.660
SlimBusMaster.c  01291  [INFO] Processing reconfiguration sequence
MSG        [08500/00] QDSP6/Low                         00:01:11.660
SlimBusMaster.c  01983  [INFO] Active channel parameters (channel#: 144)
(interval: 8) (offset: 4) (refcnt: 1)
```

The SLIMbus clock gear is 6.

```
MSG        [08500/00] QDSP6/Low                         00:01:11.660
SlimBusMaster.c  01996  [INFO] Bandwidth utilization stats (used data+msg
slots: 48+30 / 1536) (CG: 6) (SM: 17)
```

The satellite/HLOS channel requests to connect such SLIM_RX_7 to SLIMbus port 16 on the codec is received.

```
MSG        [08500/00] QDSP6/Low                         00:01:11.671
SlimBusMaster.c  00662  [INFO] Got satellite connect port request (satLA:
0xc2) (channel: 144) (destLA: 0xcb) (port: 16) (MC: 0x11)
MSG        [08500/00] QDSP6/Low                         00:01:11.672
SlimBusMaster.c  00446  [INFO] Got satellite define channel request
(client: 0xc20b) (channel: 144) (#channels: 1) (protocol: 0) (eCoeff: 1)
(exp: 2) (bitwidth: 16)
MSG        [08500/00] QDSP6/Low                         00:01:11.673
SlimBusMaster.c  00549  [INFO] Got satellite reconfigure now request
(client: 0xc20b)
MSG        [08500/00] QDSP6/Low                         00:01:11.673
SlimBusMaster.c  01291  [INFO] Processing reconfiguration sequence
MSG        [08500/00] QDSP6/Low                         00:01:11.676
SlimBusMaster.c  01983  [INFO] Active channel parameters (channel#: 144)
(interval: 8) (offset: 4) (refcnt: 2)
MSG        [08500/00] QDSP6/Low                         00:01:11.676
SlimBusMaster.c  01996  [INFO] Bandwidth utilization stats (used data+msg
slots: 48+30 / 1536) (CG: 6) (SM: 17)
```

The aDSP AFE driver closes the SLIMbus port.

```
MSG        [08500/00] QDSP6/Low                         00:01:24.477
SlimBus.c  06236  [INFO] Got data channel control request (client:
0xF0940980) (resource: 0x55200) (channel: 144) (#channels: 1) (op: 2)
(#descs: 12819) (#bytes: 1230624)
MSG        [08500/00] QDSP6/Low                         00:01:24.477
SlimBus.c  06666  [INFO] Got reconfigure now request (client: 0xF0940980)
```

Overflow/underflow during channel removal is normal. If seen in HLOS logs, overflow/underflow markers indicate that there may be an issue on the WCD codec side. If seen in aDSP logs, the markers indicate that there may be an issue on the MSM SLIMbus side, which could be recovery.

```
MSG        [08500/01] QDSP6/Medium                      00:01:24.480
SlimBus.c  02155  [INFO] disabled port interrupt due to overflow/underflow
during removal (client: 0xF0940980) (resource: 0x15000) (port: 0)
MSG        [08500/00] QDSP6/Low                         00:01:24.480
SlimBusMaster.c  01291  [INFO] Processing reconfiguration sequence
MSG        [08500/00] QDSP6/Low                         00:01:24.480
SlimBusMaster.c  01983  [INFO] Active channel parameters (channel#: 144)
(interval: 8) (offset: 4) (refcnt: 1)
MSG        [08500/00] QDSP6/Low                         00:01:24.480
SlimBusMaster.c  01996  [INFO] Bandwidth utilization stats (used data+msg
slots: 48+30 / 1536) (CG: 6) (SM: 17)
MSG        [08500/01] QDSP6/Medium                      00:01:24.480
SlimBus.c  03222  [INFO] Slimbus device closed (client: 0xF0940980) (#open
1)
```

The HLOS removes the channel, which also closes the port.

```
MSG        [08500/00] QDSP6/Low                         00:01:24.488
SlimBusMaster.c  00456  [INFO] Got satellite data channel control request
(client: 0xc20b) (channel: 144) (#channels: 1) (op: 2)
MSG        [08500/00] QDSP6/Low                         00:01:24.490
SlimBusMaster.c  00549  [INFO] Got satellite reconfigure now request
(client: 0xc20b)
MSG        [08500/00] QDSP6/Low                         00:01:24.490
SlimBusMaster.c  01291  [INFO] Processing reconfiguration sequence
MSG        [08500/00] QDSP6/Low                         00:01:24.492
SlimBus.c  01727  [INFO] Got port disconnection (client: 0xF0940980)
(resource: 0xffffffff) (port: 0)
MSG        [08500/00] QDSP6/Low                         00:01:24.492
SlimBusMaster.c  01996  [INFO] Bandwidth utilization stats (used data+msg
slots: 0+30 / 1536) (CG: 9) (SM: 0)
MSG        [08500/00] QDSP6/Low                         00:01:24.493
SlimBus.c  02468  [INFO] disconnecting BAM pipe (client: 0xF0940980)
(resource: 0xffffffff) (port: 0)
```

The SLIMbus device is closed and the SLIMbus clock is disabled.

```
MSG        [08500/01] QDSP6/Medium                      00:01:26.060
SlimBus.c  03222  [INFO] Slimbus device closed (client: 0xF096FDB0) (#open
0)
MSG        [08500/00] QDSP6/Low                         00:01:26.060
SlimBusMaster.c  01291  [INFO] Processing reconfiguration sequence
```

```
MSG         [08500/01] QDSP6/Medium                      00:01:26.060
SlimBusMaster.c  01742  [INFO] Initiating transition to idle pause clock at
next reconfig boundary
MSG         [08500/00] QDSP6/Low                         00:01:26.060
SlimBusDal.c  01205  [INFO] Registering for GPIO interrupt (input: 1)
MSG         [08500/01] QDSP6/Medium                      00:01:26.060
SlimBusMaster.c  02684  [INFO] Framer entered idle pause clock
MSG         [08500/01] QDSP6/Medium                      00:01:26.060
SlimBusMaster.c  02635  [INFO] Turning off slimbus ref clock
```

17.1.8 Customization

This section is not applicable to this release.

17.1.9 Debugging

Table 17-2 Debugging and troubleshooting tips

Issue type	Symptoms	Troubleshooting steps
Codec not detected	Codec report present is not seen in the logs	1. Check whether the LPASS image is loaded and if the LPASS is up. 2. If the LPASS is up, and codec is not detected, there could be a problem with the codec power supply or with the SLIMbus GPIO connection. 3. Check clock and data lines with an oscilloscope.
Failure in the call flow	Some steps in the call flow could throw errors	1. Look for channel connect, allocate master ports, and define channel requests. 2. Ensure that all those requests are successful. 3. Look for the channel numbers to ensure that the requests are operated only on the designated/expected channel. 4. Sometimes the channel numbers are used incorrectly due to channel mismatch or problems with the audio driver.
Port allocation failure	Failures during port allocation failure are seen in the logs	Caused by port exhaustion, either due to problems in channel removal or repeatedly using new ports and not removing them effectively after use.

17.2 AUX PCM

MSM8974 supports AUX PCM on the primary and secondary audio interfaces. The AUX PCM interface supports the following:

- Short and long sync timing
- Mono full duplex PCM data transfer
- 8-bit or 16-bit
- Bit clock of 64, 128, 256, 512, 1024, or 2048 kHz; 64 kHz rate is valid only in 8-bit mode

- Nominal sync rate of 8 kHz. It can support other samples as long as the clock is retained at one of the bit clocks for a maximum of 2048 kHz. The software selects different frame sizes as follows:

  Sync rate = Clock rate/frame size

  Assume clock rate = 2048 kHz, for 8 kHz sync rate, frame size = 2048/8 = 256 and for 16 kHz, the frame size is 128.

17.2.1 Enable logs

17.2.1.1 Kernel logs

```
adb root
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo -n "file soc-pcm.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm8974.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm-dai-q6-v2.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm-pcm-routing-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
```

17.2.1.2 User space logs

The following command clears the current logcat logs and starts logging the user space logs to the logcat.txt and to the stdout simultaneously.

```
adb logcat –c  && adb logcat –v threadtime  | tee logcat.txt
```

NOTE: For the tee command to work, Cygwin must be installed or the command must be executed in a Linux environment.

17.2.1.3 QXDM Professional logs

See *Hexagon Access Audio/Voice PCM/Bit Stream Logging with QXDM Professional* (80-N3470-4) for QXDM Professional logging.

17.2.2 Log analysis

This section is not applicable to this release.

17.2.3 Verifying primary and secondary interface

17.2.3.1 Play back on primary interface

```
amix 'PRI_AUX_PCM_RX Audio Mixer MultiMedia1' 1
aplay playback_test.wav
```

17.2.3.2 Recording using primary PCM interface

```
amix 'MultiMedia1 Mixer PRI_AUX_PCM_UL_TX' 1
arec /data/recdata.wav -T 10
```

17.2.3.3 Playback on secondary interface

```
amix 'SEC_AUX_PCM_RX Audio Mixer MultiMedia1' 1
aplay playback_test.wav
```

17.2.3.4 Recording using secondary PCM interface

```
amix 'MultiMedia1 Mixer SEC_AUX_PCM_UL_TX' 1
arec /data/recdata.wav -T 10
```

17.2.4 Customization

See solution #027768 – PCM interface customization on the MSM8974 Linux Android at https://support.cdmatech.com/.

17.3 MI2S

MI2S bus is a serial bus to interface to audio chips. It is a simple data interface, without any form of address or device selection. There is only one bus master, one transmitter, and one receiver. Either the transmitter or the receiver can be the bus master. MI2S bus can carry stereo or multichannel audio on its data lines. Each data line carries up to two-channel data left and right, and the data alternates between the left and right channels, as controlled by a Word Select (WS) signal driven by the bus master.

MI2S on MSM configured in different modes is shown in Table 17-3 .

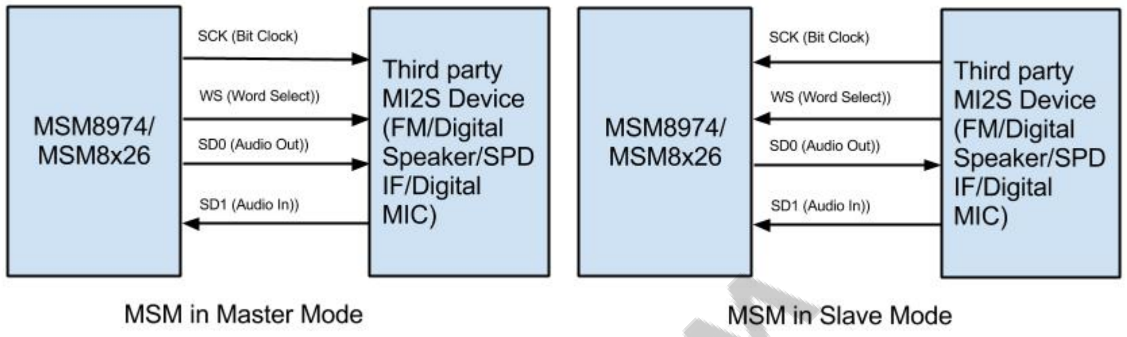


Figure 17-3 MI2S interface on MSM

MI2S on MSM

The following features of MI2S bus on MSM are supported:

- Supports both Master and Slave mode configuration
- Supported data formats
  - 16-bit I2S
  - 24-bit left justified (24-bit data in 32-bit frame left justified, LSBs are padded with 0s)
- Supported sample rates
  - 8, 16, 32, 48, 96, and 192 kHz in both Master and Slave mode
- Maximum bit clock supported is 12.288 MHz
- Serial data lines are configured for Rx/Tx and each data line can carry two channels of data (stereo data)
- Supports four MI2S hardware interfaces
  - MI2S #1, 2, 3 can support a maximum of four channels in Half Duplex mode (Rx or Tx) or support stereo in Full Duplex mode.
  - MI2S #4 can support up to eight channels in Half Duplex mode (Rx or Tx) or Full Duplex mode.

17.4 Call flows

17.4.1.1 Audio playback over MI2S

Figure 17-4 shows the call flow involved in audio playback.

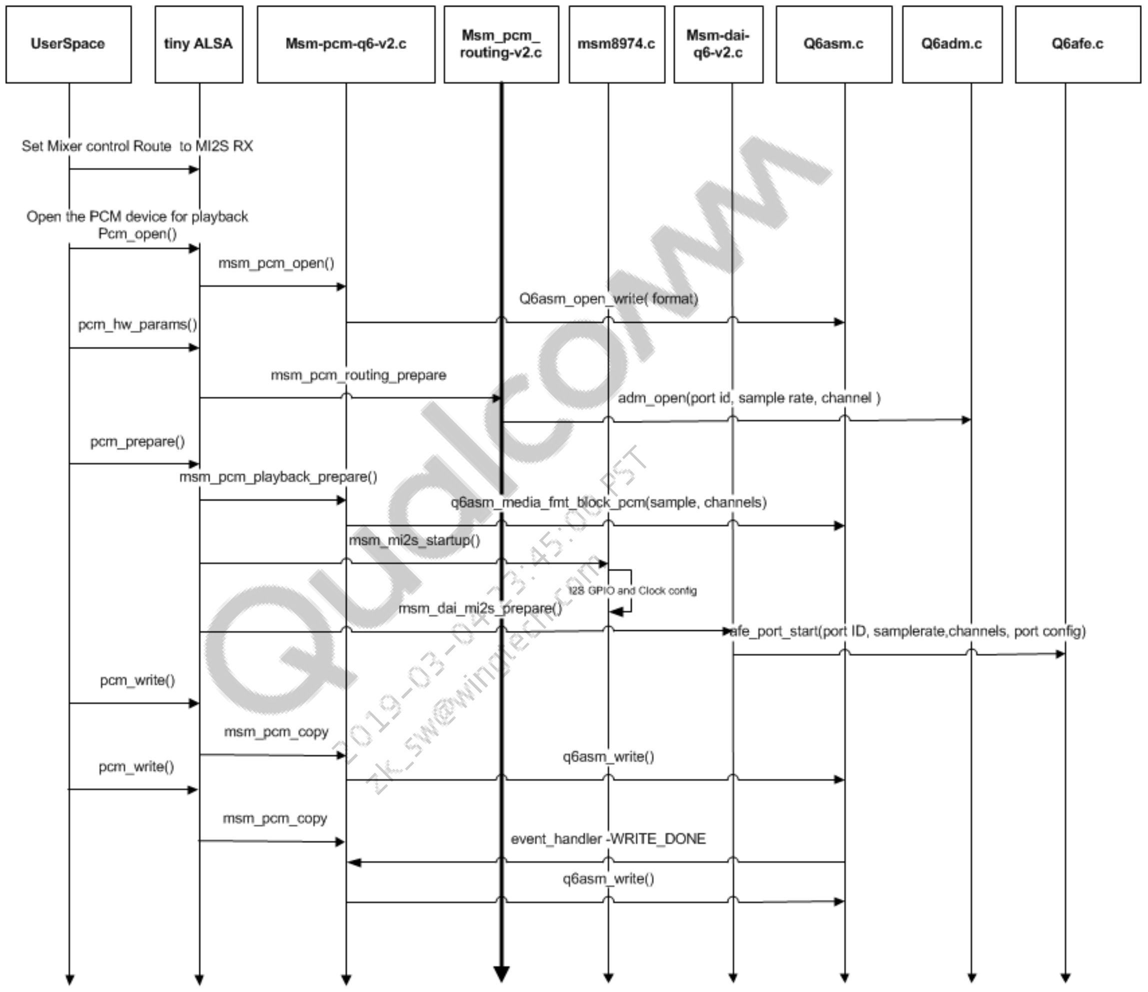

UserSpace
tiny ALSA
Msm-pcm-q6-v2.c
Msm_pcm_routing-v2.c
msm8974.c
Msm-dai-q6-v2.c
Q6asm.c
Q6adm.c
Q6afe.c
Set Mixer control Route to MI2S RX
Open the PCM device for playback Pcm_open()
msm_pcm_open()
Q6asm_open_write(format)
pcm_hw_params()
msm_pcm_routing_prepare
adm_open(port id, sample rate, channel)
pcm_prepare()
msm_pcm_playback_prepare()
q6asm_media_fmt_block_pcm(sample, channels)
msm_mi2s_startup()
128 GPIO and Clock config
msm_dai_mi2s_prepare()
afe_port_start(port ID, samplerate,channels, port config)
pcm_write()
msm_pcm_copy
q6asm_write()
pcm_write()
msm_pcm_copy
event_handler -WRITE_DONE
q6asm_write()

Figure 17-4 MI2S playback call flow

17.4.1.2 Audio capture over MI2S

Figure 17-5 shows the call flow involved in audio capture.

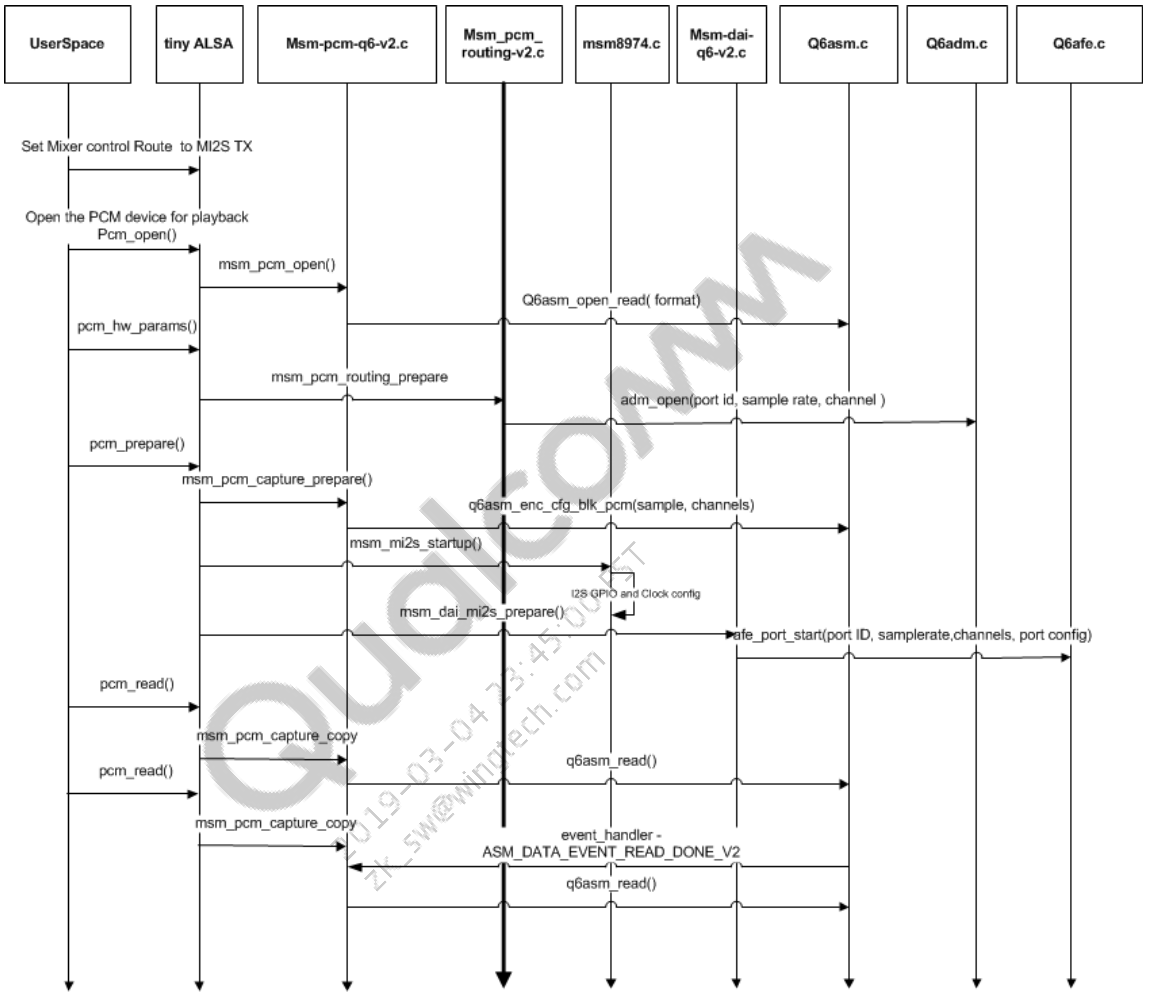


Figure 17-5 MI2S audio capture call flow

17.4.1.3 Voice call over MI2S Rx (external digital speaker) and Tx (external digital mic)

Figure 17-6 shows the call flow involved in voice calls over third-party MI2S devices.

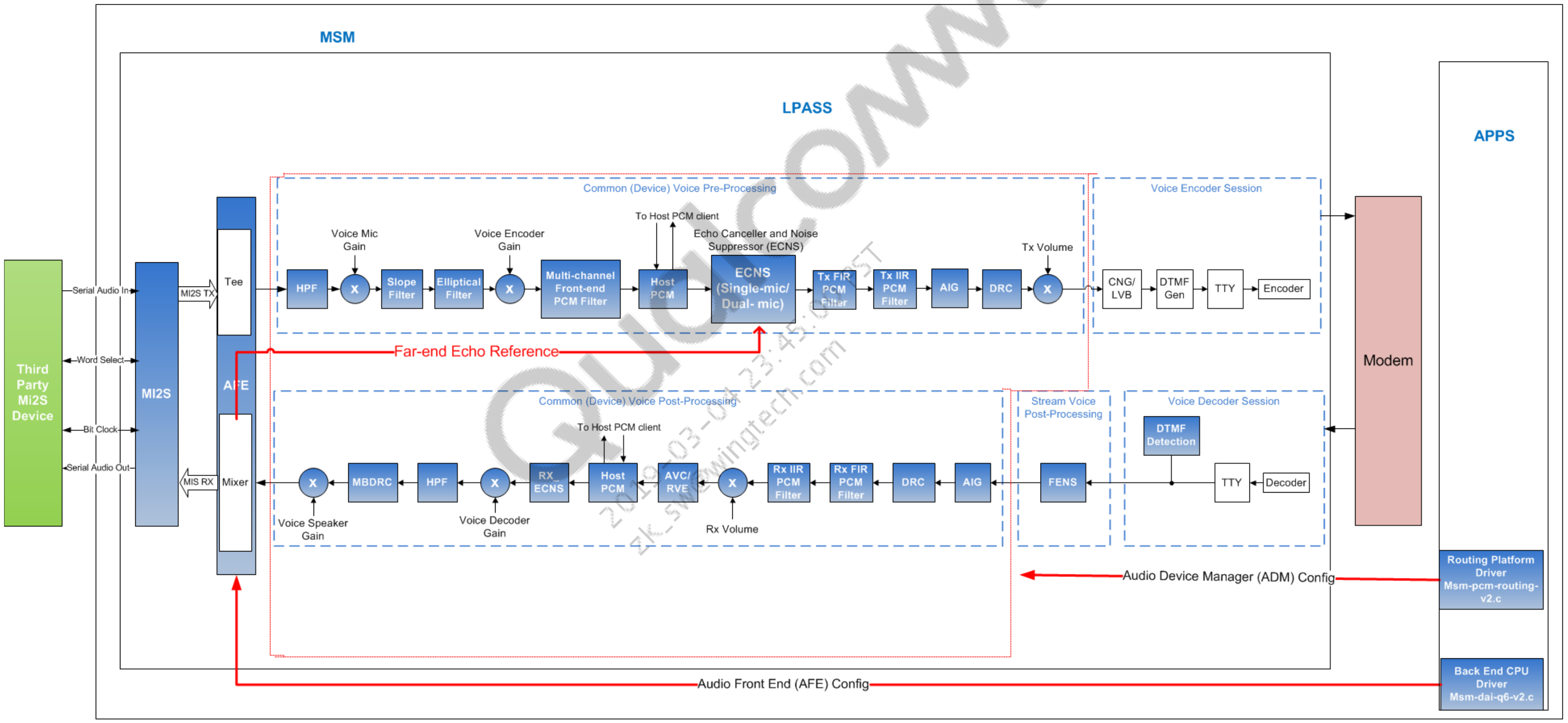


Figure 17-6 Call flow of voice call over external MI2S devices

17.4.1.4 Voice call over MI2S Rx (external digital speaker) and SLIMbus (WCD codec mic) using internal EC reference

Figure 17-7 shows the call flow involved in voice calls over a third-party MI2S device, an internal codec device, and an internal EC reference.

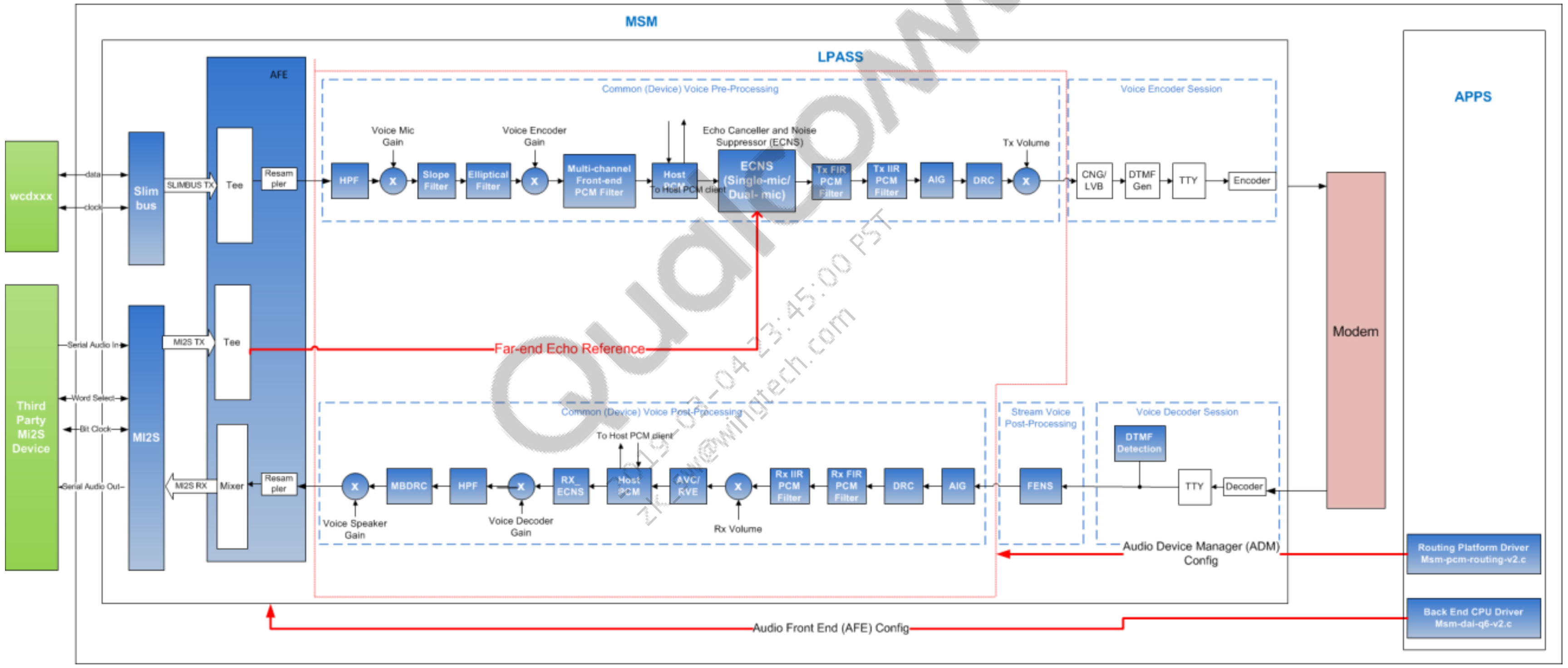


Figure 17-7 Call flow of voice call over external MI2S device, internal codec device, and internal EC reference

17.4.1.5 Voice call over MI2S Rx (external digital speaker) and SLIMbus (WCD codec mic) using external EC reference over MI2S Tx

Figure 17-8 shows the call flow involved in voice calls over a third-party MI2S device, an internal codec device, and an external EC reference over MI2S Tx.

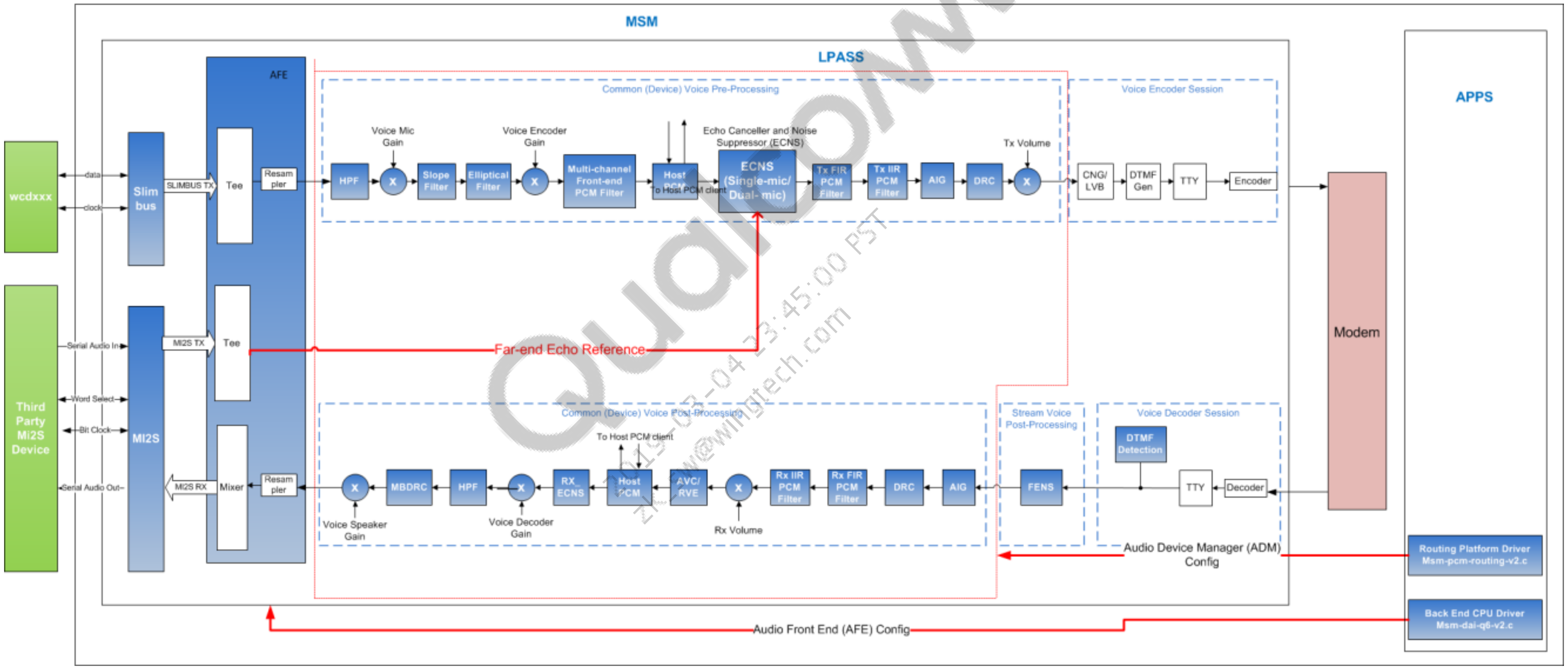


Figure 17-8 Call flow of voice call over external MI2S device, internal codec device, and external EC reference over MI2S Tx

17.5 Log collection

17.5.1.1 User space logs

```
adb logcat –c && adb logcat –v threadtime  | tee logcat.txt
```

17.5.1.2 Kernel logs

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo -n "file msm8974.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm-pcm-routing-v2.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file msm-pcm-q6-v2.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm-dai-q6-v2.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file soc-dapm.c +p" > /sys/kernel/debug/dynamic_debug/control
```

17.5.1.3 Additional logging

Based on MI2S code changes, logs from the files listed at the following locations may be helpful:

\kernel\sound\soc\msm\
msm8974.c

\kernel\arch\arm\mach-msm\
board-8974-gpiomux.c

\kernel\arch\arm\boot\dts\
msm8974.dtsi

On the kernel side, logs on the following files are enabled to debug issues related to APSS and DSP communication.

```
adb shell
mount -t debugfs debugfs /sys/kernel/debug
echo file q6afe.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6adm.c +p > /sys/kernel/debug/dynamic_debug/control
echo file q6asm.c  +p > /sys/kernel/debug/dynamic_debug/control
```

For issues related to calibration, enable log messages in audio_acdb.c.

```
echo file audio_acdb.c +p > /sys/kernel/debug/dynamic_debug/control
```

NOTE: Enabling dynamic logging on q6asm.c result in many messages that can affect system performance. Enable only the necessary logs depending on the issue.

17.5.1.4 QXDM Professional logs

This section is not applicable to this release.

17.6 Log analysis

This section is not applicable to this release.

17.7 Customization

While starting any playback or capture session, the machine driver configures MI2S GPIOs, clocks, and clock sources.

17.7.1 GPIO configuration

Table 17-3 shows the hardware GPIO pins that are used by the various MI2S interfaces.

Table 17-3 Hardware GPIO mappings for all the MI2S interfaces in MSM8974

MI2S interface	Hardware GPIO mapping
Primary	▪ MI2S_1_MCLK – GPIO 64 ▪ MI2S_1_SCK – GPIO 65 ▪ MI2S_1_WS – GPIO 66 ▪ MI2S_1_SD0 – GPIO 67 ▪ MI2S_1_SD1 – GPIO 68
Secondary	▪ MI2S_2_MCLK – GPIO 78 ▪ MI2S_2_SCK – GPIO 79 ▪ MI2S_2_WS – GPIO 80 ▪ MI2S_2_SD0 – GPIO 81 ▪ MI2S_2_SD1 – GPIO 82
Tertiary	▪ MI2S_3_MCLK – GPIO 73 ▪ MI2S_3_SCK – GPIO 74 ▪ MI2S_3_WS – GPIO 75 ▪ MI2S_3_SD0 – GPIO 76 ▪ MI2S_3_SD1 – GPIO 77
Quarternary	▪ MI2S_4_MCLK – GPIO 57 ▪ MI2S_4_SCK – GPIO 58 ▪ MI2S_4_WS – GPIO 59 ▪ MI2S_4_SD0 – GPIO 60 ▪ MI2S_4_SD1 – GPIO 61 ▪ MI2S_4_SD2 – GPIO 62 ▪ MI2S_4_SD3 – GPIO 63

As part of the playback or capture session start procedure, the MI2S back-end DAI is started and MI2S GPIOs are requested.

MI2S GPIOs use the gpiomux_setting defined in board-8974-gpiomux.c file as shown as follows.

```
static struct gpiomux_setting sec_mi2s_act_cfg = {
   .func = GPIOMUX_FUNC_1,
    .drv = GPIOMUX_DRV_8M,
    .pull = GPIOMUX_PULL_NONE,
};
```

Driver strength controls GPIO pad strength; it is configurable from 2 mA to 16 mA, with a default of 8 mA.

GPIO pull is configured with internal pull-up, pull down, keeper function, or pull none (disable all pull). The required configuration for the MI2S is PULL_NONE.

17.7.2 MI2S clock configuration in Master mode

The bit clock frequency is configured based on the sample rate, number of channels, and bit width per the following formula:

```
bit clock frequency = sample rate * number of channels * bit width
```

In MI2S Master mode, both the bit clock and WS are provided by the MSM; the WS clock will be sourced from the bit clock and will be set equal to the sample rate. Clock configuration is not needed for WS.

In MI2S Master mode, both the bit clock and WS will be provided by the MSM; the WS clock will be sourced from the bit clock and will be set equal to the sample rate. Clock configuration is not needed for WS.

For MI2S Master mode configuration, the SoC DAI format is set with SND_SOC_DAIFMT_CBS_CFS in the machine driver while enabling the back-end MI2S DAI; the bit clock and WS are masters on the MSM side and slaves on a third-party MI2S device.

The back-end CPU driver (msm-dai-q6-v2.c) configures the WS source as internal for MI2S Master mode based on the SoC DAI format.

MI2S clock control is in LPASS, and APPS should configure the bit clock and OSR clock by calling the afe_set_lpass_clock (port ID, afe_clk_cfg) API with the following parameters from the machine driver MI2S DAI start-up function.

Port ID

- 0x1000 (AFE_PORT_ID_PRIMARY_MI2S_Rx) – Primary MI2S playback
- 0x1001 (AFE_PORT_ID_PRIMARY_MI2S_Tx) – Primary MI2S capture
- 0x1002 (AFE_PORT_ID_SECONDARY_MI2S_Rx) – Secondary MI2S playback
- 0x1003 (AFE_PORT_ID_SECONDARY_MI2S_Tx) – Secondary MI2S capture
- 0x1004 (AFE_PORT_ID_TERTIARY_MI2S_Rx) – Tertiary MI2S playback

- 0x1005 (AFE_PORT_ID_TERTIARY_MI2S_Tx) – Tertiary MI2S capture
- 0x1006 (AFE_PORT_ID_QUATERNARY_MI2S_Rx) – Quaternary MI2S playback
- 0x1007 (AFE_PORT_ID_QUATERNARY_MI2S_Tx) – Quaternary MI2S capture

afe_clk_cfg

Bit clock configuration in MSM MI2S Master mode for 16-bit width and 48 kHz sample rate:

```
static struct afe_clk_cfg lpass_mi2s_enable = {
        AFE_API_VERSION_I2S_CONFIG,
        Q6AFE_LPASS_IBIT_CLK_1_P536_MHZ,
        Q6AFE_LPASS_OSR_CLK_12_P288_MHZ,
        Q6AFE_LPASS_CLK_SRC_INTERNAL,
        Q6AFE_LPASS_CLK_ROOT_DEFAULT,
        Q6AFE_LPASS_MODE_BOTH_VALID,
        0,
};
```

Bit clock configuration in MSM MI2S Master mode for 24-bit width and 48 kHz sample rate:

```
static struct afe_clk_cfg lpass_mi2s_enable = {
        AFE_API_VERSION_I2S_CONFIG,
        Q6AFE_LPASS_IBIT_CLK_3_P072_MHZ,
        Q6AFE_LPASS_OSR_CLK_12_P288_MHZ,
        Q6AFE_LPASS_CLK_SRC_INTERNAL,
        Q6AFE_LPASS_CLK_ROOT_DEFAULT,
        Q6AFE_LPASS_MODE_BOTH_VALID,
        0,
};
```

17.7.3 MI2S clock configuration in Slave mode

In MI2S Slave mode, both the bit clock and WS should be provided by a third-party MI2S device.

For MI2S Slave mode configuration, SoC DAI format is set with SND_SOC_DAIFMT_CBM_CFM in the machine driver. The back-end MI2S DAI; the bit clock, and WS are slaves on the MSM side and masters on the third-party MI2S device.

The back-end CPU driver (msm-dai-q6-v2.c) configures the WS source as external for the MI2S Slave mode based on the SoC DAI format. The APPS processor configures the bit clock source as external for the Slave mode configuration by calling the afe_set_lpass_clock (port ID, afe_clk_cfg) API with the following parameters from the machine driver MI2S DAI start-up function:

Port ID

- 0x1000 (AFE_PORT_ID_PRIMARY_MI2S_Rx) – Primary MI2S playback
- 0x1001 (AFE_PORT_ID_PRIMARY_MI2S_Tx) – Primary MI2S capture
- 0x1002 (AFE_PORT_ID_SECONDARY_MI2S_Rx) – Secondary MI2S playback

- 0x1003 (AFE_PORT_ID_SECONDARY_MI2S_Tx) – Secondary MI2S capture
- 0x1004 (AFE_PORT_ID_TERTIARY_MI2S_Rx) – Tertiary MI2S playback
- 0x1005 (AFE_PORT_ID_TERTIARY_MI2S_Tx) – Tertiary MI2S capture
- 0x1006 (AFE_PORT_ID_QUATERNARY_MI2S_Rx) – Quaternary MI2S playback
- 0x1007 (AFE_PORT_ID_QUATERNARY_MI2S_Tx) – Quaternary MI2S capture

afe_clk_cfg

Bit clock configuration in MSM MI2S Slave mode for 16-bit width and 48 kHz sample rate:

```
static struct afe_clk_cfg lpass_mi2s_enable = {
        AFE_API_VERSION_I2S_CONFIG,
        Q6AFE_LPASS_IBIT_CLK_1_P536_MHZ,
        Q6AFE_LPASS_OSR_CLK_DISABLE,
        Q6AFE_LPASS_CLK_SRC_EXTERNAL,
        Q6AFE_LPASS_CLK_ROOT_DEFAULT,
        AFE_PORT_LPACLK_CLK_VALUE1_VALID_ONLY,
        0,
};
```

Bit clock configuration in MSM MI2S Slave mode for 24-bit width and 48 kHz sample rate:

```
static struct afe_clk_cfg lpass_mi2s_enable = {
        AFE_API_VERSION_I2S_CONFIG,
        Q6AFE_LPASS_IBIT_CLK_3_P072_MHZ,
        Q6AFE_LPASS_OSR_CLK_DISABLE,
        Q6AFE_LPASS_CLK_SRC_EXTERNAL,
        Q6AFE_LPASS_CLK_ROOT_DEFAULT,
       AFE_PORT_LPACLK_CLK_VALUE1_VALID_ONLY,
        0,
};
```

17.7.4 MI2S clock source selection

While starting up playback or capture sessions, the machine driver configures the MODE_MUXSEL register for the clock Source selection (SS) for MI2S Master/Slave mode.

By default, the MI2S hardware interface configures the clock SS as internal (LPASS), and this configuration is called as Master mode.

The clock SS is changed to Slave mode by changing the SEL bit to 1.

Table 17-4 MI2S clock SS

LPAIF_xxx_MODE_MUXSEL		
31:2	**RESERVED_BITS31_2**	**Troubleshooting steps**
1	I2S_PCM_SEL	Selects either I2S or PCM interface ▪ 0 – I2S ▪ 1 – PCM
0	SEL	Source select ▪ 0 – Clock is provided by the LPASS clock controller and is considered as Master mode ▪ 1 – Clock is provided by an external off-chip source; this is considered Slave mode

Note: xxx –PRI (MI2S Primary), SEC (MI2S Secondary), TER (MI2S Tertiary), QUAD (MI2S Quaternary)

The source file with these definitions is sound/soc/msm/msm8974.c.

17.7.5 Device pair and EC configuration for voice calls over MI2S

The following device pair combinations are used for voice call over MI2S usage scenarios:

- MI2S Rx (external digital speaker) and MI2S Tx (external digital mic)
- MI2S Rx (external digital speaker) and SLIMbus Tx (WCD codec mic) with internal EC reference (reference from AFE Rx)
- MI2S Rx (external digital speaker) and SLIMbus Tx (WCD codec mic) with external EC reference (reference from external digital speaker)

By default, the DSP configures internal EC reference for voice calls. Using mixer commands from APPS, the EC reference is changed to an external source.

The following solutions are used for enabling external CE over MI2S Tx and adding the MI2S Rx voice device into the UCM configuration file.

- 00027297 – Enables the external EC over MI2S on MSM8974/MSM8x26 targets
- 00027549 – Checks/debugs whether the external EC came over MI2S Tx properly on MSM8974/MSM8226 target
- 00027729 – Adds MI2S Rx device into the UCM configuration file on MSM8974

17.8 Debugging

Table 17-5 Mixer commands for testing MI2S playback/recording

MI2S Interface	Playback	Recording
Primary	tinymix "MI2S_Rx Audio Mixer MultiMedia1" 1 tinyplay /data/test.wav -D 0 -d 0	tinymix "MultiMedia1 Mixer MI2S_Tx" 1 tinycap /data/rec.wav -D 0 -d 0 -r 48000 -c 1
Secondary	tinymix "SEC_MI2S_Rx Audio Mixer MultiMedia1" 1 tinyplay /data/test.wav -D 0 -d 0	tinymix "MultiMedia1 Mixer SEC_MI2S_Tx" 1 tinycap /data/rec.wav -D 0 -d 0 -r 48000 -c 1
Tertiary	tinymix “TERT_MI2S_Rx Audio Mixer MultiMedia1" 1 tinyplay /data/test.wav -D 0 -d 0	tinymix "MultiMedia1 Mixer TERT_MI2S_Tx" 1 tinycap /data/rec.wav -D 0 -d 0 -r 48000 -c 1
Quarternary	tinymix "QUAT_MI2S_Rx Audio Mixer MultiMedia1" 1 tinyplay /data/test.wav -D 0 -d 0	tinymix "MultiMedia1 Mixer QUAT_MI2S_Tx" 1 tinycap /data/rec.wav -D 0 -d 0 -r 48000 -c 1

Table 17-6 Debugging and troubleshooting tips for audio over MI2S

Issue type	Symptoms	Troubleshooting steps
GPIO config	No activity on the MI2S lines	1. Check MI2S GPIO connections on the board. 2. Check the GPIO mux file (board-8974-gpiomux.c/board-8226-gpiomux.c) and the machine driver (msm8974.c/msm8226.c) to see if the device is configured with the right GPIOs. 3. Check in the device tree (msm8974.dtsi/msm8226.dtsi) for MI2S configuration if the right GPIOs are configured for MI2S serial data out (SDO – Playback) and serial data in (SDIN – Recording).
Clock config	MCLK and/or WS signal is not present	1. Using an oscilloscope, check whether required clocks inputs can be seen on the MI2S bit clock line and WS line during playback/recording. 2. If the clock signals are not present, check the logs to see if the MI2S clock enable API (afe_set_lpass_clock) is called from the MI2S DAI Startup function (mi2s_startup). 3. Check clock configurations based on MI2S Master/Slave mode setup.
AFE config	AFE port is not active	Check to see if afe_port_start() API is called with right MI2S parameters (Port ID, sample rate, bit width, Channel mode, WS source, and data format).

Table 17-7 Debugging and troubleshooting tips for voice over M12S

Issue type	Symptoms	Troubleshooting steps
Call failure	Voice call fails	1. Check if normal audio playback and recording are working over the MI2S if using MI2S Rx and MI2S Tx for the voice call. ▫ If MI2S Rx and SLIMbus Tx are used for the voice call, check if MI2S Rx audio playback and recording over SLIMbus Tx are working. 2. Check the logcat logs to see if the right device is selected for the voice call use case. 3. Check if the right MI2S voice device for both Rx and Tx in the UCM configuration file and the ALSADevice.cpp file are selecting MI2S voice devices for the voice call use case. 4. Check the kernel logs and confirm if MI2S DAI start-up and afe_port_startup() are called. 5. Check if the right MI2S voice device for both Rx and Tx in the UCM configuration file and the ALSADevice.cpp file are selecting MI2S voice devices for the voice call use case.

18 Audio effects/postprocessing

18.1 Dolby postprocessing

This section is not applicable to this release.

18.2 DTS postprocessing

This section is not applicable to this release.

18.3 HeadphoneX from DTS

This section is not applicable to this release.

18.4 Debugging

18.4.1 Log PCM data before and after Google effects

Define the LVM_PCM flag in ../frameworks/av/media/libeffects/lvm/wrapper/Bundle/Effectbundle.h as shown.

```
-//#define LVM_PCM
+#define LVM_PCM
```

Once the playback starts and effects are applied, the PCM data is logged in the /data/tmp/ folder on the target.

If some permission issues prevent creating this folder, the folder tmp can be manually created under /data/.

If postprocessing in the /frameworks/av/media/libeffects/ lvm/wrapper/Reverb/EffectReverb.cpp file is applied, the following files are created:

- /data/tmp/reverb_pcm_in.pcm
- /data/tmp/reverb_pcm_out.pcm

If postprocessing in the /frameworks/av/media/libeffects/ lvm/wrapper/Bundle/EffectBundle.cpp file is applied, the following files are created:

- /data/tmp/bundle_%p_pcm_in.pcm
- /data/tmp/bundle_%p_pcm_out.pcm

18.4.1.1 QXDM Professional logs

See *Hexagon Access Audio/Voice PCM/Bit Stream Logging with QXDM Professional* (80-N3470-4) for QXDM Professional logging.

QXDM Professional logs with PCM logging enabled help in isolating noise issues.

A noise issue can be isolated if the issue is due to DSP PCM processing blocks or related to the APSS or WCD codec.

19 Volume control for audio streams

19.1 Overview

Table 19-1 lists the volume control in various audio test cases.

Table 19-1 Volume control for different audio streams

Use case	Software-based volume control	DSP-based volume control	Comment
Deep buffer playback	√	–	Controlled via software in the user space
Compress offload	–	√	Controlled using Compress Rx Volume mixer control
Voice call	–	√	Controlled using Voice Rx Volume mixer control
VoIP call	–	√	Controlled using the VoIP Rx Volume mixer control
FM volume	–	√	Controlled using FM Rx Volume mixer control

Factors that also affect loudness or volume of audio played or recorded are:

- Calibration parameters and the fixed gain applied at different points in the audio path in the DSP
- Digital and analog gains in the WCD codec
- Type of stream – Volume depends on the Android stream type defined in audio_stream_type_t (system/core/include/system/audio.h)
- Logical device – Device chosen such as speaker, earpiece, HDMI.

Analysis of logs in the user and kernel space and PCM logging at various points in the DSP help in isolating volume-related issues.

19.2 Logs analysis

This section is not applicable to this release.

20 Encoders and decoders

The file /frameworks/av/media/libstagefright/MediaDefs.cpp has the list of MIME types supported by the Android device.

The list of supported decoders and encoders is maintained in /device/qcom/common/media/media_codecs.xml.

The media_codec.xml file resides in the /etc/ folder in the device file system.

The file frameworks/av/media/libstagefright/MediaCodecList.cpp has the code to parse the media_codecs.xml.

```
MediaCodecList::MediaCodecList()
    : mInitCheck(NO_INIT) {
    FILE *file = fopen("/etc/media_codecs.xml", "r");
```

The file /frameworks/av/media/libstagefright/OMXCodec.cpp contains the logic to choose the appropriate codec and instantiate the OXM component for encoding or decoding purposes.

Both simple and complex syntax exist to declare the availability of a MediaCodec.

A codec that correctly follows the OpenMAX specification [S1], and therefore does not have any quirks and supports only a single content type is declared:

```
    <MediaCodec name="OMX.foo.bar" type="something/interesting" />
```

If a codec has quirks or supports multiple content types, the following syntax is used:

```
    <MediaCodec name="OMX.foo.bar" >
        <Type name="something/interesting" />
        <Type name="something/else" />
        ...
        <Quirk name="requires-allocate-on-input-ports" />
        <Quirk name="requires-allocate-on-output-ports" />
        <Quirk name="output-buffers-are-unreadable" />
    </MediaCodec>
```

Currently, only the three quirks included in the code are recognized. The specification of input buffers using the OMX_UseBuffer(...) API is not supported, but OMX_AllocateBuffer is mandated.

- The requires-allocate-on-input-ports quirk must be advertised if the component does not support the specification of input buffers using the OMX_UseBuffer(...) API, but mandates the use of OMX_AllocateBuffer.
- The requires-allocate-on-output-ports quirk is advertised if the component does not support the specification of output buffers using the OMX_UseBuffer(...) API, but mandates the use of OMX_AllocateBuffer.
- The output-buffers-are-unreadable quirk is advertised if the emitted output buffers of a decoder component are not readable. Clients of such decoders are not able to access the decoded data, thus making the component less useful. The only use for a component with this quirk is to render the output to the screen. Audio decoders must not advertise this quirk. Video decoders that advertise this quirk are accompanied by a corresponding color space converter for thumbnail extraction and matching SurfaceFlinger support that can render the custom format to a texture and possibly other code. Thus, this quirk is not used.

Table 20-1 Decoders supported in MSM8974

Codec	Hardware/software decoding	Tunnel/nontunnel	Bits per sample	Sample rate
PCM playback	NA	NA	16/24	8 kHz to 92 kHz
MP3	Hardware/software	Tunnel/nontunnel	16	Per the spec
AAC/AAC+/eAAC+	Hardware/software	Tunnel/nontunnel	16	Per the spec
WMA (v 9 and v 10)	Hardware	Nontunnel	16	Per the spec
AMR-NB	Software	Nontunnel	16	8 kHz
AMR-WB	Software	Nontunnel	16	16 kHz
EVRC	Software (QTI)	Nontunnel	16	8 kHz
QCELP	Software (QTI)	Nontunnel	16	8 kHz
FLAC	Software	Nontunnel	16	8 kHz to 48 kHz

Table 20-2 Encoders supported in MSM8974

Codec	Hardware/software encoding	Tunnel/nontunnel	Bits per sample	Sample rate
PCM	NA	NA	16	8 kHz to 48 kHz
AAC	Hardware/software	Nontunnel	16	8 kHz to 48 kHz
AMR-NB	Software	Nontunnel	16	8 kHz
AMR-WB	Hardware	Tunnel/nontunnel	16	16 kHz
EVRC	Hardware	Nontunnel	16	8 kHz
QCELP	Hardware	Nontunnel	16	8 kHz

21 Multibutton headset control (MBHC)

21.1 MBHC overview

MBHC is a codec hardware block to detect headset insertion/removal into the 3.5 mm audio accessory jack, and headset button press/release. Primary functions of the MBHC are:

- Plug insertion/removal detection
- Plug type determination, for example, headset, headphone
- Button press/release detection (up to eight buttons)

Other functions are:

- U.S./Euro headset type detection
- Impedance detection

21.1.1 Plug types

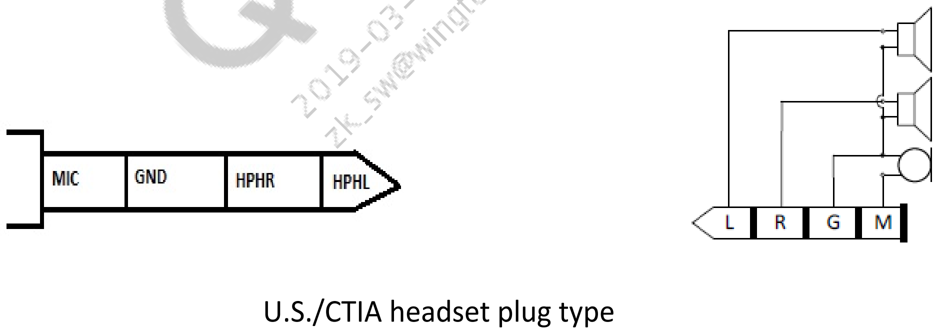


Figure 21-1 U.S./CTIA plug type

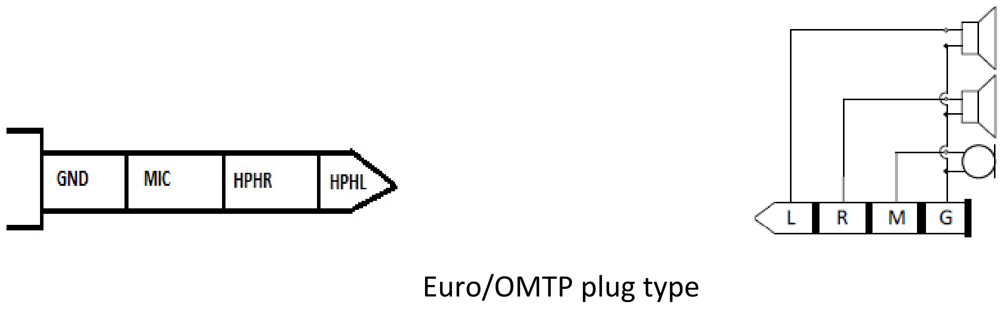


Figure 21-2 Euro/OMTP plug type

NOTE: GND and MIC are swapped in the U.S. and Euro headset

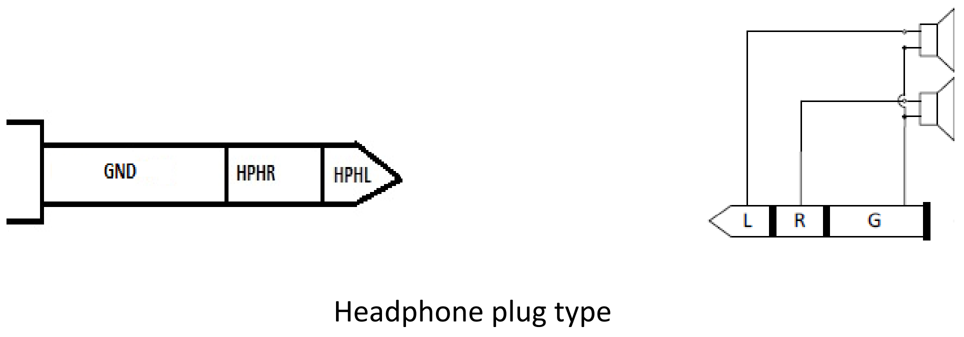


Figure 21-3 Headphone plug type

21.1.2 Normally open (NO) and normally closed (NC) jack type

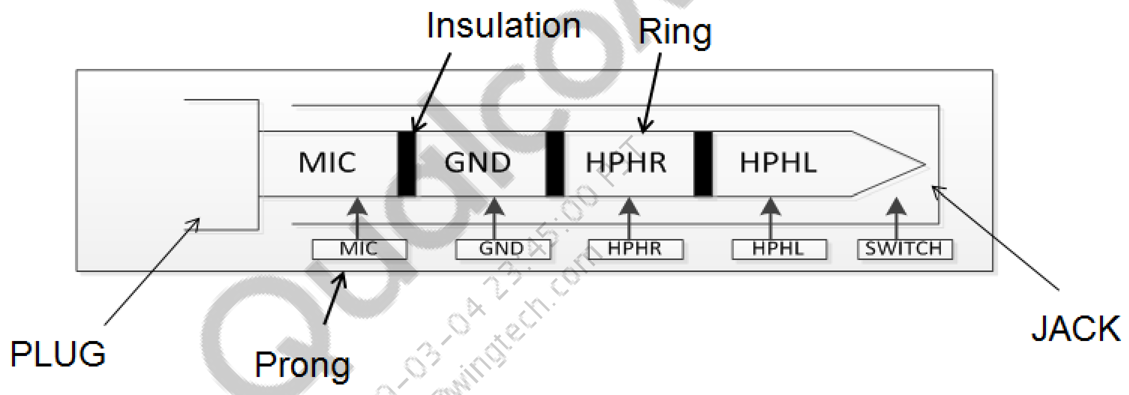


Figure 21-4 Different parts in jack and plug

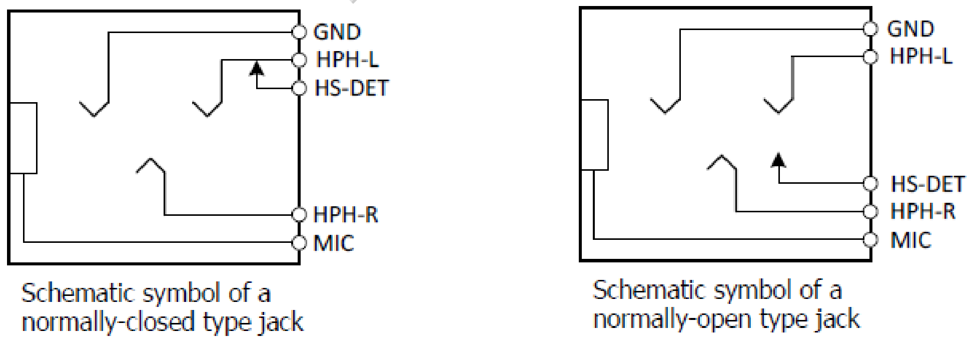

GND
HPH-L
HS-DET
HPH-R
MIC
Schematic symbol of a normally-closed type jack
GND
HPH-L
HS-DET
HPH-R
MIC
Schematic symbol of a normally-open type jack

Figure 21-5 Schematic symbol of NO and NC type jack

21.1.3 Different MBHC interrupts

The list of codec interrupts is present in the codec driver. For the WCD9306 codec, the list of interrupts is present in the */kernel/sound/soc/codecs/wcd9306.c* file. For the WCD9320 codec, the list of interrupts is present in the */kernel/sound/soc/codecs/wcd9320.c* file. For msm8x10-wcd codec, the list of interrupts is present in the *kernel/sound/soc/codecs/msm8x10-wcd.c* file.

```
static const struct wcd9xxx_mbhc_intr cdc_intr_ids = {
    .poll_plug_rem = WCD9XXX_IRQ_MBHC_REMOVAL,
    .shortavg_complete = WCD9XXX_IRQ_MBHC_SHORT_TERM,
    .potential_button_press = WCD9XXX_IRQ_MBHC_PRESS,
    .button_release = WCD9XXX_IRQ_MBHC_RELEASE,
    .dce_est_complete = WCD9XXX_IRQ_MBHC_POTENTIAL,
    .insertion = WCD9XXX_IRQ_MBHC_INSERTION,
    .hph_left_ocp = WCD9XXX_IRQ_HPH_PA_OCPL_FAULT,
    .hph_right_ocp = WCD9XXX_IRQ_HPH_PA_OCPR_FAULT,
    .hs_jack_switch = WCD9320_IRQ_MBHC_JACK_SWITCH,
};
```

Table 21-1 List of interrupts for WCD9320

Interrupt no.	Register name	Bit	Source	Interrupt pin	Destination
intr_0	INTR_REG0	0	SLIMbus	slimbus_core_irq	cdc_intr1
intr_1		1	MBHC digital	PollPlugRemoved	
intr_2		2	MBHC digital	ShortAvgComplete_int	
intr_3		3	MBHC digital	PotentialButtonPress_int	
intr_4		4	MBHC digital	ButtonRelease_int	
intr_5		5	MBHC digital	DCEstComplete_int	
intr_6		6	MBHC analog	d_cdc_mbhc_int (PlugInsRem)	
intr_7		7	Codec analog	d_cdc_bg_int	
intr_8	INTR_REG1	0	Codec analog	d_cdc_line_cnp_int_0	
intr_9		1	Codec analog	d_cdc_line_cnp_int_1	
intr_10		2	Codec analog	d_cdc_line_cnp_int_2	
intr_11		3	Codec analog	d_cdc_line_cnp_int_3	
intr_12		4	Codec analog	d_cdc_spkr_ocp_int	
intr_13		5	Codec analog	d_cdc_micb_int_0	
intr_14		6	Codec analog	d_cdc_micb_int_1	
intr_15		7	Codec analog	d_cdc_micb_int_2	
intr_16	INTR_REG2	0	Codec analog	d_cdc_hphl_ocp_int	
intr_17		1	Codec analog	d_cdc_hphr_ocp_int	
intr_18		2	Codec analog	d_cdc_ear_ocp_int	
intr_19		3	Codec analog	d_cdc_hphl_cnp_int	
intr_20		4	Codec analog	d_cdc_hphr_cnp_int	
intr_21		5	Codec analog	d_cdc_ear_cnp_int	

Interrupt no.	Register name	Bit	Source	Interrupt pin	Destination
intr_22	INTR_REG3	0	MAD digital	MAD – Audio interrupt	cdc_intr1 or cdc_intr2
intr_23		1	MAD digital	MAD – Beacon interrupt	
intr_24		2	MAD digital	MAD – Ultrasound	
intr_25		3	Codec analog	Speaker clipping interrupt	
intr_26		4	MBHC analog	HeadsetJackSwitch interrupt	cdc_intr1 or cdc_intr2
intr_27		5	Vbatt digital	Vbatt attack interrupt	
intr_28		6	VBAT digital	VBAT release interrupt	

The HeadsetJackSwitch interrupt (WCD9320_IRQ_MBHC_JACK_SWITCH) corresponds to mechanical insertion and removal detection. This interrupt is triggered whenever there is a change in the mechanical switch status. The NO/NC jack type communicates to the hardware whether the high signal on this line means insertion or removal. See the MBHC_INSERT_DETECT register definition in *WCD9320 Audio Codec Hardware Register Description for OEMs* (80-NA556-2).

Table 21-2 List of interrupts used in MBHC

Codec interrupt ID	Interrupt handler	Comment
WCD9XXX_IRQ_MBHC_REMOVAL cdc_intr_ids. poll_plug_rem	wcd9xxx_hs_remove_irq()	Used in electrical MBHC to detect plug removal during button polling
WCD9XXX_IRQ_MBHC_INSERTION cdc_intr_ids. Insertion	wcd9xxx_hs_insert_irq()	Used in electrical MBHC low-power removal of plug.
WCD9XXX_IRQ_MBHC_SHORT_TERM	No ISR registered	
WCD9XXX_IRQ_MBHC_PRESS	No ISR registered	
WCD9XXX_IRQ_MBHC_RELEASE cdc_intr_ids.button_release	wcd9xxx_release_handler()	Triggered when a button is released
WCD9XXX_IRQ_MBHC_POTENTIAL cdc_intr_ids. dce_est_complete	wcd9xxx_dce_handler()	Triggered when a button is pressed
WCD9320_IRQ_MBHC_JACK_SWITCH cdc_intr_ids .hs_jack_switch	wcd9xxx_mech_plug_detect_irq()	Corresponds to mechanical insertion and removal detection; this interrupt is triggered whenever there is a change in the mechanical switch status

21.1.4 Plug type detection

When a plug is inserted, wcd9xxx_mech_plug_detect_irq() is triggered. Once the insertion is detected, it is necessary to determine the type of plug that is inserted by calling wcd9xxx_mbhc_detect_plug_type(). If the current source method is used, wcd9xxx_codec_cs_get_plug_type() determines the plug type based on the DCE measurements. If the mic bias method is used, wcd9xxx_codec_get_plug_type() determines the plug type based on the DCE measurements. The plug types returned by wcd9xxx_codec_cs_get_plug_type()/ wcd9xxx_codec_get_plug_type() are based on the instantaneous DCE reading, and it is not known whether the plug type is completely or partially inserted. To determine the plug type correctly, wcd9xxx_codec_cs_get_plug_type()/ wcd9xxx_codec_get_plug_type() are called in the wcd9xxx_correct_swch_plug() scheduled work function every 100 ms (HS_DETECT_PLUG_INERVAL_MS) within a loop that runs for 5 sec (HS_DETECT_PLUG_TIME_MS). The plug type is reported to the user space using the wcd9xxx_report_plug() function.

21.1.4.1 Determining plug type based on DCE measurements

wcd9xxx_codec_cs_get_plug_type() is the core function that determines the plug type based on DCE measurement of the voltage at MIC2_INP with the current source method used. During a call to wcd9xxx_codec_cs_get_plug_type(), a minimum of four DCE measurements are taken to determine the plug type. wcd9xxx_codec_get_plug_type() is not discussed in this document.

- wcd9xxx_mbhc_detect is a structure that holds the DCE measurement value and information on whether mic bias was on, HPHR switch to detect GND_MIC swap, or VDDIO switch was enabled, and so on, while taking the DCE measurement. See Figure 21-7 where d and dt point to the array of structure of type wcd9xxx_mbhc_detect.
- dce_z, mb_mv, hs_max, and no_mic are used as thresholds to determine whether the vdces (voltage value corresponding to dce value) fall in the range of PLUG_TYPE_HEADSET, PLUG_TYPE_HEADPHONE, PLUG_TYPE_HIGH_HPH, or PLUG_TYPE_INVALID.

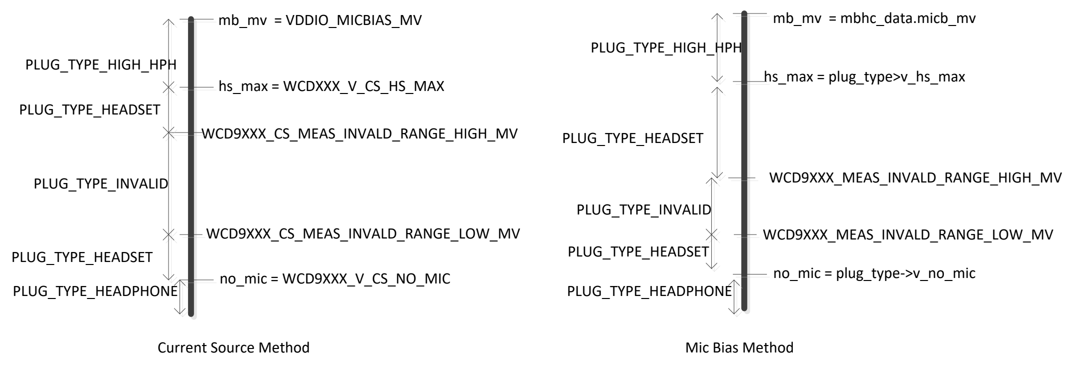

mb_mv = VDDIO_MICBIAS_MV
PLUG_TYPE_HIGH_HPH
hs_max = WCDXXX_V_CS_HS_MAX
PLUG_TYPE_HEADSET
WCD9XXX_CS_MEAS_INVALD_RANGE_HIGH_MV
PLUG_TYPE_INVALID
WCD9XXX_CS_MEAS_INVALD_RANGE_LOW_MV
PLUG_TYPE_HEADSET
no_mic = WCD9XXX_V_CS_NO_MIC
PLUG_TYPE_HEADPHONE
Current Source Method
mb_mv = mbhc_data.micb_mv
PLUG_TYPE_HIGH_HPH
hs_max = plug_type>v_hs_max
PLUG_TYPE_HEADSET
WCD9XXX_MEAS_INVALD_RANGE_HIGH_MV
PLUG_TYPE_INVALID
WCD9XXX_MEAS_INVALD_RANGE_LOW_MV
PLUG_TYPE_HEADSET
no_mic = plug_type->v_no_mic
PLUG_TYPE_HEADPHONE
Mic Bias Method

Figure 21-6 Plug type determined based on mic voltage comparison with thresholds

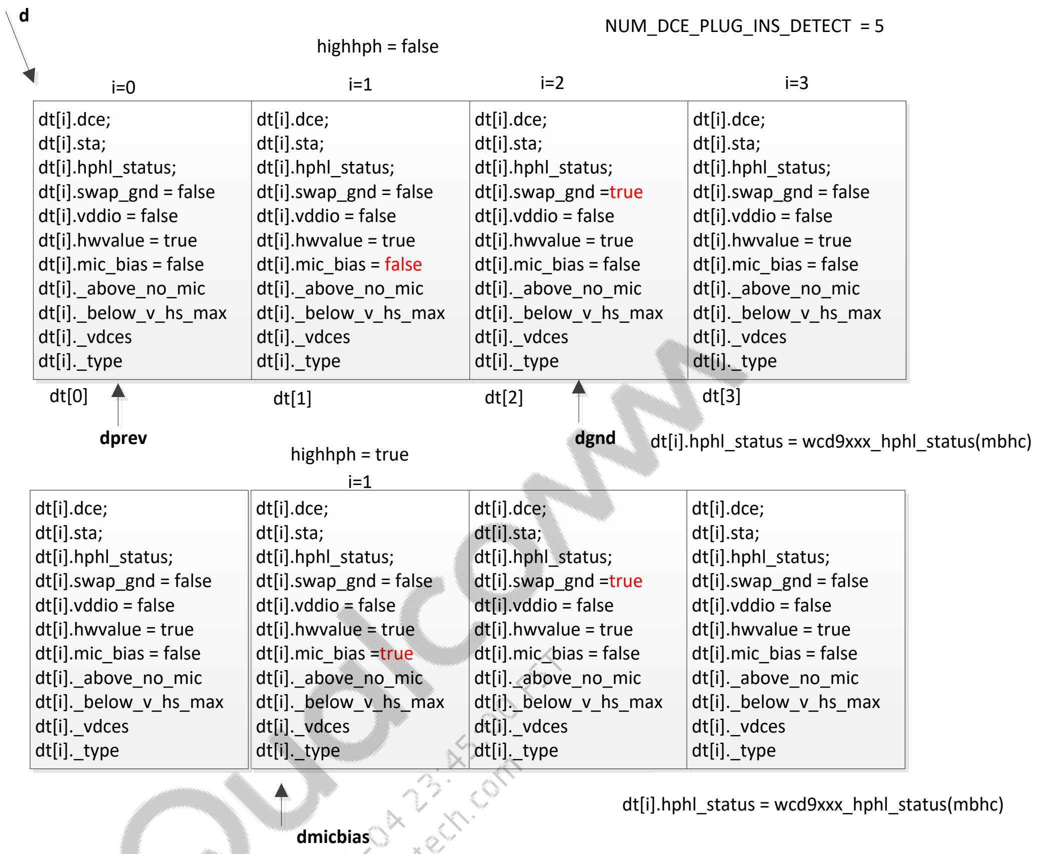


Figure 21-7 Data structure used for DCE measurement in wcd9xxx_codec_cs_get_plug_type()

- If all DCE measurements are consistent and the value lies in the range of headphone, then the headphone is reported. However, the MBHC driver still continues to make additional four set measurements, each having four DCE readings to ensure that a headphone was indeed inserted. This process continues for 5 sec if a headphone was inserted. Sometimes, a partially inserted headset is first reported as a headphone and later reported as a headset when fully inserted. If, during the 5 sec polling process, a headset is detected as a result of complete insertion of the plug, the polling stops, and the headset is reported.
- If all the DCE measurements are consistent and the voltage at the MIC2_INP lies in the range of high HPH, polling is continued for 5 sec to take few more sets of DCE measurements (each set comprising four DCE measurements). If high HPH is still detected, mic bias is enabled and a set of DCE measurements is taken to check whether the voltage measured at the MIC2_INP is in the range of the headset.

  Some headsets have mics that require mic bias of 2.7 V to enable them. As a result, such headsets are detected as high HPH with the current source enabled. When the mic bias is enabled and DCE measurement of voltage at MIC2_INP is done, the value could lie in the range of the headset and the headset is reported and further polling is stopped before 5 sec. If the voltage at MIC2_INP lies in the range of high HPH, irrespective of the current source or mic bias enabled method of detecting plug type, high HPH is reported.

- If all DCE measurements are consistent and the value lies in the range of an invalid headset, polling to measure the MIC2_INP voltage is continued for 5 sec. If within the 5 sec, a valid plug type is determined, the type of plug inserted is reported to the user space. If the plug type is still invalid at the end of 5 sec, an invalid plug type is reported.
- If any of the DCE measurements are not consistent, polling of the MIC2_INP continues for 5 sec. If within the 5 sec, a valid plug type is detected, it is reported to the user space.

21.1.4.2 Reporting the plug type

- Based on the plug type detected in wcd9xxx_codec_cs_get_plug_type()/ wcd9xxx_codec_cs_get_plug_type(), the plug type is reported to the user space.
- If the plug type is determined as PLUG_TYPE_HEADSET, additional plug type detection in wcd9xxx_correct_swch_plug() is stopped and SND_JACK_HEADSET is reported. If enable_anc_mic_detect is set to true in MBHC configuration, wcd9xxx_detect_anc_plug_type() is called to detect the five-pole ANC headphone having two mics before reporting SND_JACK_HEADSET, wcd9xxx_detect_anc_plug_type() is called. If two mics are detected, SND_JACK_ANC_HEADPHONE is reported.
- If the plug types PLUG_TYPE_HEADPHONE, SND_JACK_HEADPHONE are reported, and further detection continues in wcd9xxx_correct_swch_plug() as the plug may be partially inserted. After the timeout in wcd9xxx_correct_swch_plug(), if the plug type is PLUG_TYPE_HEADPHONE, further detection is stopped.
- If the plug type is PLUG_TYPE_HIGH_HPH, there is a possibility that the plug type has a mic that gets enabled above certain mic bias voltage or the plug type could be an extension cable/aux cable. In the case of a current source method, depending on the number of consecutive readings of a plug type as PLUG_TYPE_HIGH_HPH, the mic bias is enabled for one DCE reading out of the four DCE readings done in wcd9xxx_codec_cs_get_plug_type() to check whether PLUG_TYPE_HEADSET is detected. If PLUG_TYPE_HIGH_HPH is still detected and detect_extension_cable is true, SND_JACK_LINE_OUT is reported. Otherwise, it is reported as SND_JACK_HEADPHONE.
- If the plug type is PLUG_TYPE_INVALID and the number of iterations to detect the plug type is five (NUM_ATTEMPTS_TO_REPORT) and.detect_extn_cable is false in MBHC configuration, SND_JACK_HEADPHONE is reported.
- If the plug type PLUG_TYPE_GND_MIC_SWAP is detected, SND_JACK_UNSUPPORTED is reported and further detection is done. If PLUG_TYPE_GND_MIC_SWAP is detected twice (GND_MIC_SWAP_THRESHOLD), the function pointer mbhc→mbhc_cfg→swap_gnd_mic is called if not NULL. The device that supports both U.S. and Euro headsets implements the function registered with the function pointer swap_gnd_mic to configure a hardware switch that swaps the mic and GND so that the correct headset type (Euro/U.S.) is detected.
- When a plug is removed, wcd9xxx_mech_plug_detect_irq() is called and, depending on the MBHC register status, removal of the plug is reported.

21.1.4.3 adb shell getevent for plug type insertion and removal

Headset and button jacks are registered with the ALSA SoC by calling snd_soc_jack_new() in the wcd9xxx_mbhc_init()function. The event number is in the kernel logs during bootup.

```
<6>[   11.823145] input: msm8226-tapan-snd-card Button Jack as
/devices/sound.42/sound/card0/input3
<6>[   11.823701] input: msm8226-tapan-snd-card Headset Jack as
/devices/sound.42/sound/card0/input4

In the File system the inputs created for Button Jack and Headset Jack can
be found as  below  .
root@msm8226:/sys/devices/sound.42/sound/card0 # ls
 . ..
input3
input4

The event number can be found through the below adb shell command
adb shell getevent
add device 1: /dev/input/event4
  name:     "msm8974-taiko-snd-card Headset Jack"
add device 2: /dev/input/event3
  name:     "msm8974-taiko-snd-card Button Jack"
add device 3: /dev/input/event0
  name:     "synaptics_rmi4_i2c"
 . ..
```

Headset insertion event

```
/dev/input/event4: 0005 0002 00000001
/dev/input/event4: 0005 0004 00000001
/dev/input/event4: 0000 0000 00000000
```

Headset removal event

```
/dev/input/event4: 0005 0002 00000000
/dev/input/event4: 0005 0004 00000000
/dev/input/event4: 0000 0000 00000000
```

Headphone insertion event

```
/dev/input/event4: 0005 0002 00000001
/dev/input/event4: 0000 0000 00000000
```

Headphone insertion event

```
/dev/input/event4: 0005 0002 00000000
/dev/input/event4: 0000 0000 00000000
```

21.1.5 TTY headset detection

The TTY headset does not have a mic. Hence, polling is not required in such a scenario. mbhc_cfg.no_mic_headset_override is set to true for TTY headsets to ensure that polling is not required.

21.1.6 Button press detection

Button detection is done only if the plug type is a headset. After a valid headset is detected, the MBHC driver starts the MBHC digital state machine for button press detection. The MBHC digital state machine invokes the short-term averaging (STA) to measure Vmic and determine if the user pressed a button. wcd9xxx_dce_handler() is triggered when a button is pressed. Depending on the STA measurement result, the DC Estimation (DCE) block is called to perform a more precise measurement to determine the button press by the user. The MBHC resumes for button polling if no button was pressed. Otherwise, the button number that is pressed is reported to the user space and the DCE voltage measurement result is monitored to detect when the button is released. When the button is released, wcd9xxx_release_handler() is called.

21.1.7 Button tuning

See *Multibutton Headset Control (MBHC) Version 1 Application Not*e (80-NA556-11).

OEMs have the flexibility to tune the voltage level threshold for each button based on the headset button design requirements. A minimum voltage of 1 mV separation is allowed between the high thresholds voltage of a button and the low thresholds voltage of the subsequent button. Table 21-3 lists a reference to the button voltage tuning parameters that are used on the MTP8974 platform for an eight-button headset.

Table 21-3 QTI MBHC default button number threshold setting

Parameter	Description	Value (mV)
MBHC_VButLow[0]	Button 0 low	-50
MBHC_VButHigh[0]	Button 0 high	20
MBHC_VButLow[1]	Button 1 low	21
MBHC_VButHigh[1]	Button 1 high	61
MBHC_VButLow[2]	Button 2 low	62
MBHC_VButHigh[2]	Button 2 high	104
MBHC_VButLow[3]	Button 3 low	105
MBHC_VButHigh[3]	Button 3 high	148
MBHC_VButLow[4]	Button 4 low	149
MBHC_VButHigh[4]	Button 4 high	189
MBHC_VButLow[5]	Button 5 low	190
MBHC_VButHigh[5]	Button 5 high	228
MBHC_VButLow[6]	Button 6 low	229
MBHC_VButHigh[6]	Button 6 high	269
MBHC_VButLow[7]	Button 7 low	270
MBHC_VButHigh[7]	Button 7 high	500

21.1.8 Design example

See *Multibutton Headset Control (MBHC) Version 1 Application Not*e (80-NA556-11).

21.1.9 Impedance detection and usage

Impedance detection measures the impedance of the left and right speakers of headphone/headset. If the impedance is higher, the volume of the audio heard is lower. If the impedance is lower, the volume of the audio heard is higher.

Measuring the impedance of the left and right headphone speaker helps OEMs implement logic control to the gain in the Rx path to adjust the loudness of audio heard and improve the user experience, even if different headsets/headphones are used.

The wcd9xxx_detect_impedance() function is called to perform impedance detection.

The HPHL impedance and HPHR impedance mixer controls are available to the user space to query the headphone left and headphone right impedances.

```
SOC_SINGLE_EXT("HPHL Impedance", 0, 0, UINT_MAX, 0,
          tapan_hph_impedance_get, NULL),
SOC_SINGLE_EXT("HPHR Impedance", 0, 1, UINT_MAX, 0,
          tapan_hph_impedance_get, NULL),
```

To query the impedance, use the tinymix command:

```
tinymix "HPHR Impedance"
tinymix "HPHL Impedance"
```

21.1.10 MIC_BIAS2, VDDIO, and current source

The voltage at MIC_BIAS2 is maintained at 2.7 V and is provided by turning on the LDOH. MIC_BIAS2 is enabled in the following scenarios:

- When the current source method of plug type detection is disabled, MIC_BIAS2 is enabled for plug type detection.
- During button polling, MIC_BIAS2 is enabled and disabled periodically to check for button press events.
- If a headset is detected and the Tx path is enabled, for example, start recording, camcorder, voice call, VoIP call, or VoLTE call, MIC_BIAS2 is enabled.
- When the current source is enabled and the plug type is detected as high HPH for a few iterations of four DCE measurements, MIC_BIAS2 is enabled to determine the plugtype.
- To detect button press during playback over a headset, the voltage at MIC_BIAS2 output is switched to VDDIO at 1.8 V to prevent any audible polling.

The current source is enabled to detect the plug type using the current source method of detection. The current source is configurable and currently configured at 20 µA. It is recommended that the OEMs do not change this value.

VDDIO is the voltage source at 1.8 V. It is enabled during playback over a headset and remains constant. It is used to determine the voltage at MIC2_INP and detect button press and release.

21.1.11 MCLK and RCO clock

MCLK is turned on in the following use case:

- When there is audio playback or recording

The RCO clock is turned on in the following use cases.

- There is no audio activity to supply clocks to the digital blocks of the codec
- During button polling, when the headset is plugged in to detect button press and release

21.1.12 MBHC configuration

mbhc_cfg defines the MBHC configuration in the machine driver. For MSM8226, it is present in kernel/sound/soc/msm/msm8226.c.

```
static struct wcd9xxx_mbhc_config mbhc_cfg = {
   .read_fw_bin = false,
   .calibration = NULL,
   .micbias = MBHC_MICBIAS2,
   .anc_micbias = MBHC_MICBIAS2,
   .mclk_cb_fn = msm_snd_enable_codec_ext_clk,
   .mclk_rate = TAIKO_EXT_CLK_RATE,
   .gpio = 0,
   .gpio_irq = 0,
   .gpio_level_insert = 1,
   .detect_extn_cable = true,
   .micbias_enable_flags = 1 << MBHC_MICBIAS_ENABLE_THRESHOLD_HEADSET,
   .insert_detect = true,
   .swap_gnd_mic = NULL,
   .cs_enable_flags = (1 << MBHC_CS_ENABLE_POLLING |
                 1 << MBHC_CS_ENABLE_INSERTION |
                 1 << MBHC_CS_ENABLE_REMOVAL |
                 1 << MBHC_CS_ENABLE_DET_ANC),
   .do_recalibration = true,
   .use_vddio_meas = true,
   .enable_anc_mic_detect = false,
   .hw_jack_type = SIX_POLE_JACK,
};
```

Table 21-4 MBHC configuration

Member variable of wcd9xxx_mbhc_config	Comment
read_fw_bin	▪ If true, get the MBHC calibration from the ACDB. ▪ If false, get the MBHC calibration data from the machine driver defined in the def_tapan_mbhc_cal() function
Calibration	Holds the address of the MBHC calibration data

Member variable of wcd9xxx_mbhc_config	Comment
micbias	MIC_BIAS2 is used in the current implementation and it is recommended that OEMs follow QTI reference MBHC schematic and implementation
anc_micbias	Mic bias for the ANC mic
mclk_cb_fn	Function pointer to the function that configures the MCLK
mclk_rate	MCLK rate is set to 12.228 MHz or 9.6 MHz
gpio	GPIO number used for the external GPIO-based plug insertion and removal
gpio_irq	Function pointer to handle the ISR for external GPIO-based plug insertion and removal detection
gpio_level_insert	Logic level of the GPIO when plug is inserted; depends on the NO/NC type jack ▪ 0 – NO type jack ▪ 1 – NC type jack
detect_extn_cable	If true, enable electrical MBHC logic to detect extension cable (aux cable or lineout)
micbias_enable_flags	MBHC_MICBIAS_ENABLE_THRESHOLD_HEADSET – Enable the logic to detect headsets whose mic is enabled if MIC BIAS > threshold voltage, for example, 2.2 V MBHC_MICBIAS_ENABLE_REGULAR_HEADSET – Used for normal headset detection
insert_detect	If true, use the mechanical insertion and removal detection in MBHC hardware; else, do not use
swap_gnd_mic	Pointer to the function that is called when plug type is PLUG_TYPE_GND_MIC_SWAP to program the hardware switch
cs_enable_flags	▪ MBHC_CS_ENABLE_POLLING – Enable polling using the current source ▪ MBHC_CS_ENABLE_INSERTION – Enable the current source method of insertion detection ▪ MBHC_CS_ENABLE_REMOVAL – Enable the current source method for plug type removal ▪ MBHC_CS_ENABLE_DET_ANC – Enable the current source method to detect the second mic for ANC headphone
do_recalibration	Doing calibration of dce_z during insertion
use_vddio_meas	▪ If true, make one DCE measurement using VDDIO source in wcd9xxx_codec_get_plug_type() ▪ If false, do not use VDDIO source in wcd9xxx_codec_get_plug_type() to make the DCE measurement
enable_anc_mic_detect	If true, enable the logic to detect ANC headphone that has two mics
hw_jack_type	▪ FOUR_POLE_JACK – Defined for a jack that can detect up to 4-pole plug type ▪ FIVE_POLE_JACK – Defined for a jack that can detect up to 5-pole plug type, for example, ANC headphone ▪ SIX_POLE_JACK – Defined for a jack that can detect up to 5-pole plug type, for example, ANC headset

21.1.13 ACDB vs. machine driver MBHC calibration

When the ACDB workspaceFile.qwsp is opened, the MBHC calibration data is found in the TableGlobal→MBHC tab.

- During bootup, the acdb_loader_init_ACDB() is called. The acdb-loader parses the MBHC calibration data from the ACDB file xxx_Global_cal.acdb where xxx = Fluid, MTP, CDP, and converts the data to the /data/misc/audio/mbhc.bin binary file in the send_mbhc_data() function. This file path is symbolically linked to /etc/firmware/wcd9320/wcd9320_mbhc.bin. The symbolic link is defined in the vendor/qcom/proprietary/mm-audio/audio-acdb-util/acdb-loader/Android.mk file.
- If the mbhc_cfg.read_fw_bin flag is set to true, MBHC calibration data is read from the /etc/firmware/wcd9320/wcd9320_mbhc.bin file; otherwise, it is read from the machine driver. In the machine driver file, the MBHC calibration data is defined in the def_xxx_mbhc_cal() function where xxx = WCD9306 or WCD9320.

21.1.14 Important macros in wcd9xxx-mbhc.c

Table 21-5 lists the macros in wcd9xxx-mbhc.c.

Table 21-5 Macros in wcd9xxx-mbhc.c

Macro	Value	Comment
NUM_DCE_PLUG_DETECT	3	Number of times the DCE measurement is taken in the wcd9xxx_hs_remove_settle() function to ensure that the plug is removed
NUM_DCE_PLUG_INS_DETECT	5	Number of times the DCE measurement of MIC2_INP should be done; used in wcd9xxx_codec_cs_get_plug_type()/ wcd9xxx_codec_get_plug_type()
NUM_ATTEMPTS_INSERT_DETECT	25	Maximum iterations to detect the plug type during insertion
NUM_ATTEMPTS_TO_REPORT	5	Number of times INVALID plug is detected in wcd9xxx_correct_swch_plug() following which HEADPHONE is reported
FAKE_INS_LOW	10	—
FAKE_INS_HIGH	80	—
FAKE_INS_HIGH_NO_SWCH	150	—
FAKE_REMOVAL_MIN_PERIOD_MS	50	—
FAKE_INS_DELTA_SCALED_MV	300	—
HS_DETECT_PLUG_TIME_MS	(5 x 1000)	Timeout in ms within which the correct plug type is decided; it is used in the wcd9xxx_correct_swch_plug() function

Macro	Value	Comment
ANC_HPH_DETECT_PLUG_TIME_MS	(5 x 1000)	Timeout in ms within which the ANC mic is detected or not is decided; it is used in the wcd9xxx_detect_anc_plug_type() function
HS_DETECT_PLUG_INERVAL_MS	100	Time interval between successive calls to wcd9xxx_codec_cs_get_ plug_type() /wcd9xxx_codec_get_ plug_type() to determine the plugtype; it is used in the wcd9xxx_correct_swch_plug() function
SWCH_REL_DEBOUNCE_TIME_MS	50	Should not be changed
SWCH_IRQ_DEBOUNCE_TIME_US	5000	Should not be changed
BTN_RELEASE_DEBOUNCE_TIME_MS	25	Should not be changed
GND_MIC_SWAP_THRESHOLD	2	Number of times PLUG_TYPE_GND_MIC_SWAP is detected before calling the mbhc→mbhc_cfg→swap_gnd_mic function pointer
FW_READ_ATTEMPTS	15	Maximum number of times the kernel driver attempts to read the MBHC calibration data from the wcd9320/wcd9320_mbhc.bin
WCD9XXX_MEAS_DELTA_MAX_MV	120	Used to determine INVALID plug type in the wcd9xxx_find_plug_ type() function
WCD9XXX_MEAS_INVALD_RANGE_LOW_MV	20	Lower threshold level for INVALID plug type detection in the mic bias method of plug type detection
WCD9XXX_MEAS_INVALD_RANGE_HIGH_MV	80	Higher threshold level for INVALID plug type detection in the mic bias method of plug type detection
WCD9XXX_CS_MEAS_INVALD_RANGE_LOW_ MV	160	Lower threshold level for INVALID plug type detection in the current source method of plug type detection
WCD9XXX_CS_MEAS_INVALD_RANGE_ HIGH_MV	265	Higher threshold level for INVALID plug type detection in the current source method of plug type detection
WCD9XXX_GM_SWAP_THRES_MIN_MV	150	Used to determine the GND_MIC_SWAP plug type in wcd9xxx_find_plug_type()
WCD9XXX_GM_SWAP_THRES_MAX_MV	650	Used to determine the GND_MIC_SWAP plug type in wcd9xxx_find_plug_type()
WCD9XXX_THRESHOLD_MIC_THRESHOLD	200	Used to determine the headsets having threshold on mic in the wcd9xxx_find_plug_type() function
WCD9XXX_V_CS_HS_MAX	500	Voltage in mV above which HIGH_HPH is reported in the current source method of plug type detection for OEM customization.

Macro	Value	Comment
WCD9XXX_V_CS_NO_MIC	5	Voltage in mV below which HEADPHONE is reported in the current source method of plug type detection for OEM customization.

21.1.15 Reference schematic for MBHC and hardware requirement

See M*ultibutton Headset Control (MBHC) Version 1 Application Not*e (80-NA556-11) for the reference schematic and hardware requirement for MBHC. OEMs should ensure that their MBHC-related design is as per the QTI reference schematic.

21.1.16 Important MBHC registers

It is assumed that the OEM uses the MIC_BIAS2 for the MBHC solution as recommended by QTI.

Table 21-6 MBHC registers

Register	Address	Comment
0x12E	MICB_CFILT_2_CTL	
0x12F	MICB_CFILT_2_VAL	Set the CFILT voltage using this register. CFILT = (LDO_H_OUT-0.15 V) x K/44, where K is defined as follows: LDO_H_OUT can be any of the following Values: ▪ 1.95 V ▪ 2.35 V ▪ 2.75 V ▪ 2.85 V The LDO_H register controls LDO_H_OUT; default set to maximum K = 44
0x130	MICB_CFILT_2_PRECHRG	Internal precharge counter setting; default is set to 5 ms
0x131	MICB_2_CTL	Bit[4] – CAP_MODE ▪ 0 – EXT_BY_CAP ▪ 1 – NO_EXT_BY_CAP
0x132	MICB_2_INT_RBIAS	Has settings to provide internal mic bias
0x133	MICB_2_MBHC	Used to configure Schmitt trigger and current source for insertion detection
0x14A	MBHC_INSERT_DETECT	Bit[2] ▪ If 1, it is an NC type jack ▪ If 0, it is an NO type jack Bit[1] – If 0, MBHC is configured for removal detection, else it is configured for insertion detection
0x14B	MBHC_INSERT_DET_STATUS	Provides information if the plug is inserted or removed If bit[2] is 0, plug is inserted, else it is removed

Register	Address	Comment
0x14E	MBHC_SCALING_MUX_1	▪ 0x4 – To detect the first mic in regular headset or ANC headphone ▪ 0x8 – To detect the second mic in ANC headphone

21.1.17 User space code for MBHC events

In the user space, the frameworks/base/services/java/com/android/server/WiredAccessoryManager.java file has the implementation to handle events from the kernel about plug insertion and removal.

The notifyWiredAccessoryChanged() function is called that handles the event from the kernel.

```
  public void notifyWiredAccessoryChanged(long whenNanos, int switchValues,
int switchMask) {
        if (LOG) Slog.v(TAG, "notifyWiredAccessoryChanged: when=" +
whenNanos
                + " bits=" + switchCodeToString(switchValues, switchMask)
                + " mask=" + Integer.toHexString(switchMask));

        synchronized (mLock) {
            int headset;
            mSwitchValues = (mSwitchValues & ~switchMask) | switchValues;
/*
```

Depending on the BIT set in the event received, the value of the variable headset is set appropriately. The OEM must add the appropriate code to check which event is received from the kernel and set the value of the variable headset.

```
*/
            switch (mSwitchValues & (SW_HEADPHONE_INSERT_BIT |
SW_MICROPHONE_INSERT_BIT)) {
                case 0:
                    headset = 0;
                    break;
                case SW_HEADPHONE_INSERT_BIT:
                    headset = BIT_HEADSET_NO_MIC;
                    break;
                case SW_HEADPHONE_INSERT_BIT | SW_MICROPHONE_INSERT_BIT:
                    headset = BIT_HEADSET;
                    break;
                case SW_MICROPHONE_INSERT_BIT:
                    headset = BIT_HEADSET;
                    break;
                default:
                    headset = 0;
                    break;
            }
            updateLocked(NAME_H2W, (mHeadsetState & ~(BIT_HEADSET |
BIT_HEADSET_NO_MIC)) | headset);
        }
    }
/*
```

updateLocked() calls mHandler.sendMessage() that is handled in handleMessage(), which calls setDevicesState().

```
*/
      private void updateLocked(String newName, int newState) {
       . . .
       Message msg = mHandler.obtainMessage(MSG_NEW_DEVICE_STATE,
headsetState,
               mHeadsetState, newName);
       mHandler.sendMessage(msg);
      }

    private final Handler mHandler = new Handler(Looper.myLooper(), null,
true) {
        @Override
        public void handleMessage(Message msg) {
            switch (msg.what) {
                case MSG_NEW_DEVICE_STATE:
                    setDevicesState(msg.arg1, msg.arg2, (String)msg.obj);
                    mWakeLock.release();
            }
        }
    };
/*
```

setDevicesState() then calls the setDeviceStateLocked() where a call is made and the appropriate AudioManager.DEVICE_OUT_XXX depends on the value of the headset and AudioManager.setWiredDeviceConnectionState() is called. This is propagated to the audio policy manager and then to the Audio HAL where the device is set.

```
*/
    private void setDevicesState(
            int headsetState, int prevHeadsetState, String headsetName) {
        synchronized (mLock) {
            int allHeadsets = SUPPORTED_HEADSETS;
            for (int curHeadset = 1; allHeadsets != 0; curHeadset <<= 1) {
                if ((curHeadset & allHeadsets) != 0) {
                    setDeviceStateLocked(curHeadset, headsetState,
prevHeadsetState, headsetName);
                    allHeadsets &= ~curHeadset;
                }
            }
        }
    }


   private void setDeviceStateLocked(int headset,
           int headsetState, int prevHeadsetState, String headsetName) {
 . ..
           if (headset == BIT_HEADSET) {
               device = AudioManager.DEVICE_OUT_WIRED_HEADSET;
           } else if (headset == BIT_HEADSET_NO_MIC){
               device = AudioManager.DEVICE_OUT_WIRED_HEADPHONE;
           } else if (headset == BIT_USB_HEADSET_ANLG) {
               device = AudioManager.DEVICE_OUT_ANLG_DOCK_HEADSET;
           } else if (headset == BIT_USB_HEADSET_DGTL) {
               device = AudioManager.DEVICE_OUT_DGTL_DOCK_HEADSET;
           } else if (headset == BIT_HDMI_AUDIO) {
               device = AudioManager.DEVICE_OUT_AUX_DIGITAL;
           } else {
               Slog.e(TAG, "setDeviceState() invalid headset type:
"+headset);
               return;
           }

           if (LOG)
               Slog.v(TAG, "device "+headsetName+((state == 1) ? "
connected" : " disconnected"));

           mAudioManager.setWiredDeviceConnectionState(device, state,
headsetName);
       }

   }
```

21.1.18 MBHC limitations

21.1.18.1 Pop noise limitations

An unavoidable pop noise occurs in the following scenarios:

- When a four-pole jack is removed (1 out of 20 times): This action occurs because polling is not disabled immediately when the headset is removed.
- When an iPhone headset is inserted: The iPhone headset requires mic bias to be enabled for correct detection of a plug type.
- When the headset is inserted with impedance detection enabled: Mic bias is enabled for DCE and STA measurement to measure the impedance

- With a NO type jack compared to an NC type jack: A pop noise is louder with a NO type jack; however, the root cause of the issue is not known.
- When removing a headset during recording: With headset removal, the interrupt is not fast enough to disable the MIC_BIAS in time so that there is no pop noise.

NOTE: All pop noise limitations are resolved beginning with the WCD9330 codec. The pop level is minimized further in WCD9330 and future codecs.

21.1.18.2 Detection of lineout plug type

The lineout cable is correctly detected if the plug is inserted into the device having MBHC and the other end is then connected to:

- An external speaker in a car stereo in the case of an aux cable
- A headset/headphone in the case of a lineout cable that has a jack for the plug type to be inserted on one end and four or three pole plugs to be inserted into the device having MBHC

21.1.19 MBHC waveforms

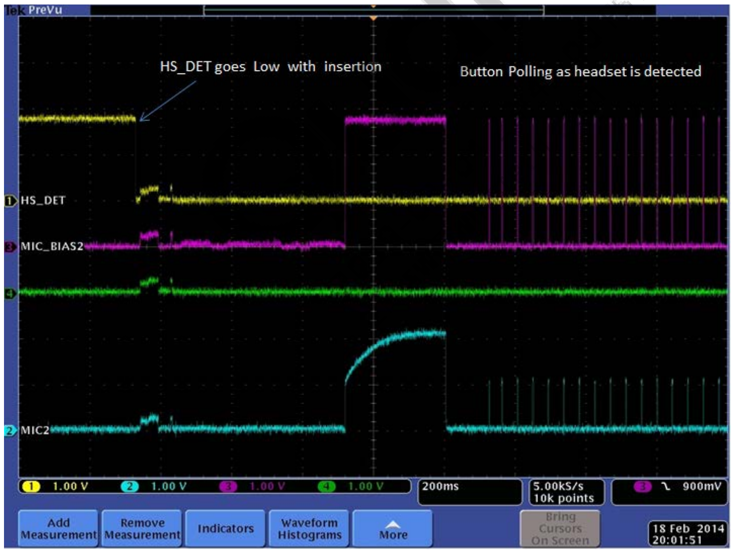


Figure 21-8 Waveform when headset is inserted

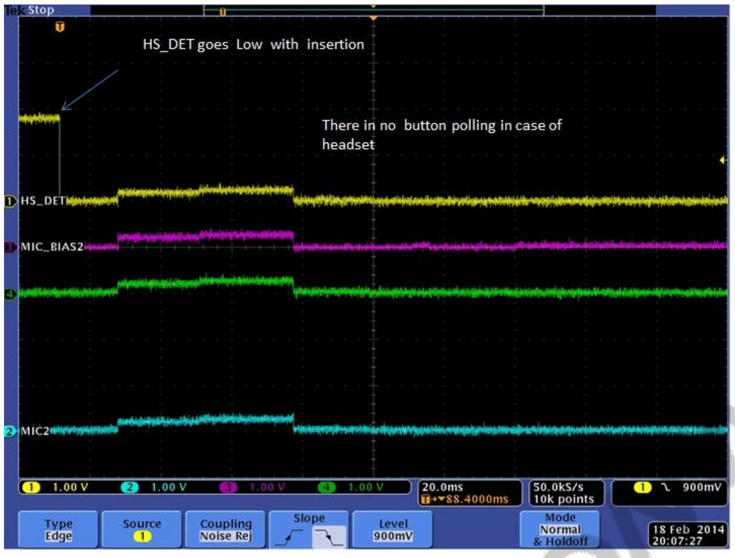


Figure 21-9 Waveform when headphone is inserted

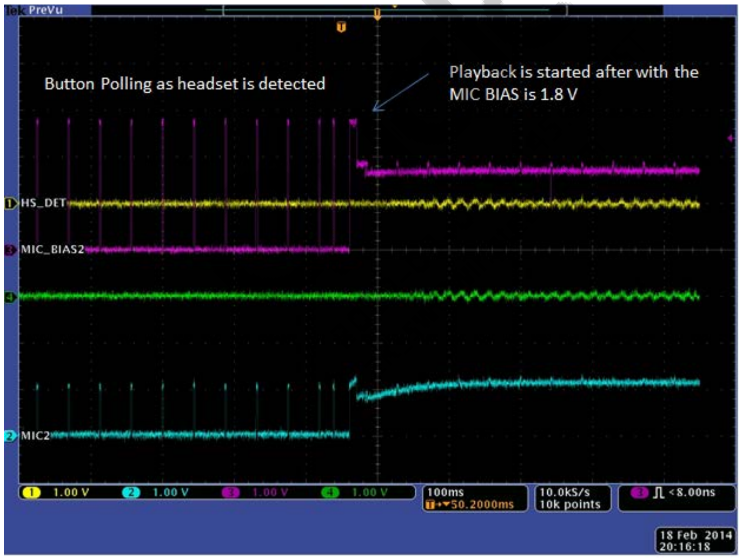


Figure 21-10 Waveform showing playback with headset inserted

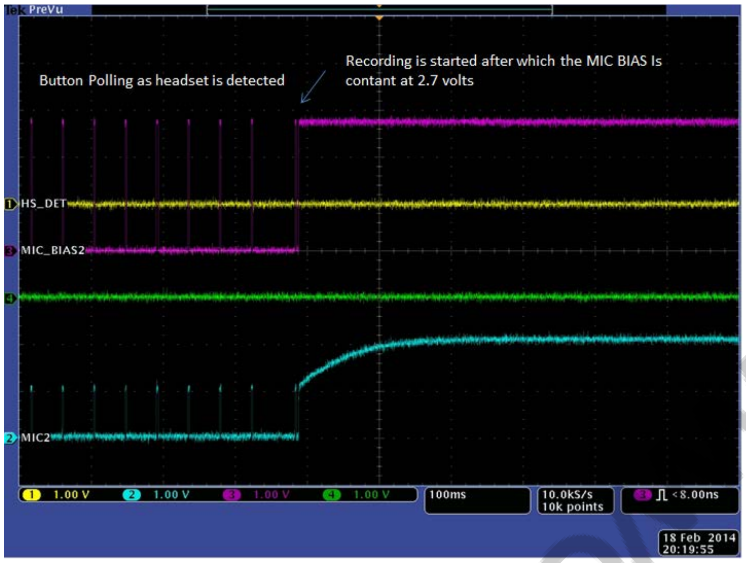


Figure 21-11 Waveform showing recording with headset inserted

21.2 Enabling logs

21.2.1 Kernel logs

21.2.1.1 Kernel logs for WCD9320 codec

Enabling logs dynamically

```
adb root
adb shell
echo -n "file wcd9320.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-core.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-irq.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-mbhc.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-resmgr.c +p" >
/sys/kernel/debug/dynamic_debug/control
```

Disabling logs dynamically

```
adb root
adb shell
echo -n "file wcd9320.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-core.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-irq.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-mbhc.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-resmgr.c -p" >
/sys/kernel/debug/dynamic_debug/control
```

21.2.1.2 Kernel logs for WCD9306/WCD9302 codec

Enabling logs dynamically

```
adb root
adb shell
echo -n "file wcd9306.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-core.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-irq.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-mbhc.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-resmgr.c +p" >
/sys/kernel/debug/dynamic_debug/control
```

Disabling logs dynamically

```
adb root
adb shell
echo -n "file wcd9306.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-core.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-irq.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-mbhc.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-resmgr.c -p" >
/sys/kernel/debug/dynamic_debug/control
```

21.2.1.3 Kernel logs for MSM8x10_WCD codec

Enabling logs dynamically

```
adb root
adb shell
echo -n "file msm8x10-wcd.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-irq.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-mbhc.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-resmgr.c +p" >
/sys/kernel/debug/dynamic_debug/control
```

Disabling logs dynamically

```
adb root
adb shell
echo -n "file msm8x10-wcd.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-irq.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-mbhc.c -p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-resmgr.c -p" >
/sys/kernel/debug/dynamic_debug/control
```

21.2.2 Dump the codec registers

```
adb root
adb shell cat /sys/kernel/debug/asoc/*snd-card/*_codec/codec_reg >
codecreg.txt
```

NOTE: Check the name of the *snd-card by reviewing the /sys/kernel/debug/asoc directory.

21.2.3 Dump MBHC calibration data

The MBHC calibration data is stored in sys/kernel/debug/wcd9xxx_mbhc.

```
adb root
adb shell cat /sys/kernel/debug/wcd9xxx_mbhc  > mbhc_data.txt
MBHC Calibration data is looks as below  :
root@msm8226:/sys/kernel/debug # cat wcd9xxx_mbhc
cat wcd9xxx_mbhc
dce_z = fbe7(0mv)
dce_mb = 932(2700mv)
dce_nsc_cs_z = f7d3(0mv)
sta_z = e5bd(0mv)
sta_mb = 3b1d(2700mv)
t_dce = 15040
t_sta = 1970
micb_mv = 2700mv
v_ins_hu = 3335(2449mv)
v_ins_h = 7f6(2449mv)
v_b1_hu = fffff8b5(599mv)
v_b1_h = fffffe9c(549mv)
v_brh = fffffe9c(549mv)
v_brl = 8000(-3217mv)
v_no_mic = e6af(29mv)
v_inval_ins_low = 15
v_inval_ins_high = 120
Insert detect insert = 0
```

- dce_z and sta_z – This indicates that register values are provided when the mic voltage is 0 V.
- dce_nsc_cs_z – This indicates that register values are provided when the current source is used.
- dce_mb and sta_mb – This indicates that register values are provided when the mic bias voltage is 2.7 V.
- micb_mv – This is the mic bias voltage in mV.
- t_dce – This is the time in µs for one DCE measurement.

- t_sta – This is the time in µs for one STA measurement.
- v_ins_hu – The maximum possible short-term averaging result that is encountered when the headset is inserted. When a value higher than the encountered value, it indicates that the headset is removed. The value depends on the headset used.
- v_ins_h – Also called V HS MAX, above which the plug type is reported as a high impedance headphone.
- v_b1_hu – The maximum possible short-term averaging result that is encountered when a button is pressed. If a value higher than this value is encountered, it indicates that no button is pressed. The value depends on the headset used.
- v_b1_h – The maximum button voltage.
- v_brh – Used during button release detection and represents the highest expected mic voltage after audio rejection is encountered when the button detected is pressed. If the voltage goes above this level, it indicates that the button is no longer pressed.
- v_brl – Used during the button release detection and represents the lowest expected mic voltage after audio rejection is encountered when the button detected is pressed. If the voltage goes below this level, it indicates that another button that results in a lower voltage is pressed before the original button was released.
- v_no_mic – The mic voltage below which there in no mic.
- v_inval_ins_low – Invalid plug low threshold in mV.
- v_inval_ins_high – Invalid plug high threshold in mV.

21.3 Log analysis

The plug type is defined in the kernel/sound/soc/codecs/wcd9xxx-mbhc.h file.

```
enum wcd9xxx_mbhc_plug_type {
    PLUG_TYPE_INVALID = -1,
    PLUG_TYPE_NONE,            /* 0 */
    PLUG_TYPE_HEADSET,         /* 1 */
    PLUG_TYPE_HEADPHONE,       /* 2 */
    PLUG_TYPE_HIGH_HPH,        /* 3 */
    PLUG_TYPE_GND_MIC_SWAP,    /* 4 */
    PLUG_TYPE_ANC_HEADPHONE,   /* 5 */

};
```

SoC jack type is defined in the kernel/include/sound/jack.h file.

```
enum snd_jack_types {
    SND_JACK_HEADPHONE = 0x0000001,
    SND_JACK_MICROPHONE= 0x0000002,
    SND_JACK_HEADSET   = SND_JACK_HEADPHONE | SND_JACK_MICROPHONE,
    SND_JACK_LINEOUT   = 0x0000004,
    SND_JACK_MECHANICAL= 0x0000008, /* If detected separately */
    SND_JACK_VIDEOOUT  = 0x0000010,
    SND_JACK_AVOUT     = SND_JACK_LINEOUT | SND_JACK_VIDEOOUT,
    /* */
    SND_JACK_LINEIN         = 0x0000020,
    SND_JACK_OC_HPHL   = 0x0000040,
    SND_JACK_OC_HPHR   = 0x0000080,
    SND_JACK_UNSUPPORTED    = 0x0000100,
    SND_JACK_MICROPHONE2    = 0x0000200,
    SND_JACK_ANC_HEADPHONE = SND_JACK_HEADPHONE | SND_JACK_MICROPHONE |
                   SND_JACK_MICROPHONE2,
    /* Kept separate from switches to facilitate implementation */
    SND_JACK_BTN_0     = 0x4000000,
    SND_JACK_BTN_1     = 0x2000000,
    SND_JACK_BTN_2     = 0x1000000,
    SND_JACK_BTN_3     = 0x0800000,
    SND_JACK_BTN_4     = 0x0400000,
    SND_JACK_BTN_5     = 0x0200000,
    SND_JACK_BTN_6     = 0x0100000,
    SND_JACK_BTN_7     = 0x0080000,
};
```

21.3.1 Headset insertion and detection of plug type

```
<7>[ 3108.899551] wcd9xxx_mech_plug_detect_irq: enter
```

Mechanical insertion/removal; IRQ is called.

```
<7>[ 3108.899607] wcd9xxx_swch_irq_handler: enter
<7>[ 3108.904735] wcd9xxx_swch_irq_handler: Acquiring BCL
<7>[ 3108.904778] wcd9xxx_swch_irq_handler: Acquiring BCL done
<7>[ 3108.905265] wcd9xxx_swch_irq_handler: Current plug type 0, insert 1
<7>[ 3108.905307] wcd9xxx_cancel_hs_detect_plug: Canceling
correct_plug_swch
<7>[ 3108.905344] wcd9xxx_cancel_hs_detect_plug: Release BCL
<7>[ 3108.905390] wcd9xxx_cancel_hs_detect_plug: Acquiring BCL
<7>[ 3108.905428] wcd9xxx_cancel_hs_detect_plug: Acquiring BCL done
<7>[ 3108.905863] wcd9xxx_mbhc_detect_plug_type: enter
<7>[ 3108.909036] wcd9xxx_mbhc_decide_swch_plug: enter
```

```
<7>[ 3108.909076] wcd9xxx_turn_onoff_current_source: enabling current
source
```

Turn on the current source to detect the plug type; this feature is introduced to reduce the pop noise observed during insertion and removal of the plug.

```
<7>[ 3108.911227] wcd9xxx_codec_cs_get_plug_type: enter
<7>[ 3108.913230] wcd9xxx_mbhc_setup_hs_polling: enter
<7>[ 3108.913269] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK
<7>[ 3108.913308] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK done
<7>[ 3108.913347] wcd9xxx_mbhc_setup_hs_polling: Releasing BG_CLK
<7>[ 3109.006892] wcd9xxx_cs_find_plug_type: enter
<7>[ 3109.006932] wcd9xxx_cs_find_plug_type: event_state 0x0
<7>[ 3109.006982] wcd9xxx_cs_find_plug_type: DCE #0, ffe8, V 0837(0837),
HPHL 1 TYPE 3
<7>[ 3109.007033] wcd9xxx_cs_find_plug_type: DCE #1, ffe8, V 0837(0837),
HPHL 1 TYPE 3
<7>[ 3109.007084] wcd9xxx_cs_find_plug_type: DCE #2, ffe8, V 0837(0837),
HPHL 1 TYPE 3
<7>[ 3109.007134] wcd9xxx_cs_find_plug_type: DCE #3, ffe8, V 0837(0837),
HPHL 1 TYPE 3
<7>[ 3109.007179] wcd9xxx_cs_find_plug_type: Plug type 3 detected
<7>[ 3109.007213] wcd9xxx_codec_cs_get_plug_type: plug_type:3
```

PLUG_TYPE_HIGH_HPH is detected – In such a scenario, MBHC continues for 5 sec to ensure that the headset or headphone is inserted. It may be possible that during the measurement, the plug was not inserted. The plug type enum is wcd9xxx_mbhc_plug_type and it is defined in kernel/sound/soc/codecs/wcd9xxx-mbhc.h.

- ffe8 – Register value of the DCE measurement
- V 0837 – Calculated voltage at the MIC2 input with the current source enabled
- (0837) – Scaled mic voltage ID; the configured mic bias is different from the current mic bias voltage
- HPHL 1 – Whether HPHL is fully connected
- TYPE 3 – Detected plug type; 1 – Headset, 2 – Headphone, 3 –HIGH_HPH, 4 – GND_MIC_SWAP

```
<7>[ 3109.007248] wcd9xxx_turn_onoff_current_source: disabling current
source
<7>[ 3109.008950] wcd9xxx_schedule_hs_detect_plug: scheduling
wcd9xxx_correct_swch_plug
<7>[ 3109.009023] wcd9xxx_mbhc_decide_swch_plug: leave
<7>[ 3109.009057] wcd9xxx_mbhc_detect_plug_type: leave
<7>[ 3109.009089] wcd9xxx_swch_irq_handler: Release BCL
<7>[ 3109.009122] wcd9xxx_swch_irq_handler: leave
<7>[ 3109.009162] wcd9xxx_mech_plug_detect_irq: leave 1
<7>[ 3109.009257] wcd9xxx_correct_swch_plug: enter
<7>[ 3109.009779] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_RCO_OFF(3)
<7>[ 3109.009819] wcd9xxx_event_notify: leave
```

```
<7>[ 3109.010673] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_RCO_OFF(4)
<7>[ 3109.010714] wcd9xxx_event_notify: leave
<7>[ 3109.010765] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_MCLK_ON(5)
<7>[ 3109.010801] wcd9xxx_event_notify: leave
<7>[ 3109.013663] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_MCLK_ON(6)
<7>[ 3109.013912] wcd9xxx_update_mbhc_clk_rate: Updating clock rate
dependents, rate = 9600000
```

The logs indicate the change in the clock source from RCO to MCLK. wcd9xxx_update_mbhc_clk_rate is called to update the MBHC calibration data required due to the change in the clock rate.

```
<7>[ 3109.014626] wcd9xxx_update_mbhc_clk_rate: leave
<7>[ 3109.014661] wcd9xxx_event_notify: leave
<7>[ 3109.014698] wcd9xxx_turn_onoff_current_source: enabling current
source
<7>[ 3109.126883] wcd9xxx_correct_swch_plug: Acquiring BCL
<7>[ 3109.126926] wcd9xxx_correct_swch_plug: Acquiring BCL done
<7>[ 3109.126961] wcd9xxx_codec_cs_get_plug_type: enter
<7>[ 3109.128847] wcd9xxx_mbhc_setup_hs_polling: enter
```

Start the polling to determine the correct plug type as the plug detected incorrect headphone insertion.

```
<7>[ 3109.128884] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK
<7>[ 3109.128923] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK done
<7>[ 3109.128961] wcd9xxx_mbhc_setup_hs_polling: Releasing BG_CLK
<7>[ 3109.218382] wcd9xxx_cs_find_plug_type: enter
<7>[ 3109.218422] wcd9xxx_cs_find_plug_type: event_state 0x0
<7>[ 3109.218470] wcd9xxx_cs_find_plug_type: DCE #0, f817, V 0034(0034),
HPHL 1 TYPE 1
<7>[ 3109.218521] wcd9xxx_cs_find_plug_type: DCE #1, f817, V 0034(0034),
HPHL 1 TYPE 1
<7>[ 3109.218571] wcd9xxx_cs_find_plug_type: DCE #2, f817, V 0034(0034),
HPHL 1 TYPE 1
<7>[ 3109.218621] wcd9xxx_cs_find_plug_type: DCE #3, f817, V 0034(0034),
HPHL 1 TYPE 1
<7>[ 3109.218664] wcd9xxx_cs_find_plug_type: Plug type 1 detected
```

All four DCE measurements are consistent,t, and plug type 1 (PLUG_TYPE_HEADSET) is detected.

The plug type is defined in the wcd9xxx_mbhc_plug_type enum inside the kernel/sound/soc/ codecs/wcd9xxx-mbhc.h file.

- f817 – Register value of the DCE measurement
- V 0034 – Calculated voltage at the MIC2 input with the current source enabled
- (0034) – Scaled mic voltage ID; the configured mic bias is different from the current mic bias voltage
- HPHL 1 – Whether HPHL is fully connected
- TYPE 1 – Detected plug type; 1 – Headset, 2 – Headphone, 3 –HIGH_HPH, 4 – GND_MIC_SWAP

```
<7>[ 3109.218699] wcd9xxx_codec_cs_get_plug_type: plug_type:1
```

PLUG_TYPE_HEADSET is detected. The plug type enum is wcd9xxx_mbhc_plug_type, and it is defined in kernel/sound/soc/codecs/wcd9xxx-mbhc.h.

```
<7>[ 3109.218732] wcd9xxx_correct_swch_plug: Release BCL
<7>[ 3109.218774] wcd9xxx_correct_swch_plug: attempt(1) current_plug(0)
new_plug(1)
<7>[ 3109.218813] wcd9xxx_correct_swch_plug: Acquiring BCL
<7>[ 3109.218850] wcd9xxx_correct_swch_plug: Acquiring BCL done
<7>[ 3109.218884] wcd9xxx_turn_onoff_current_source: disabling current
source
<7>[ 3109.220086] wcd9xxx_find_plug_and_report: enter current_plug(0)
new_plug(1)
<7>[ 3109.220134] wcd9xxx_report_plug: enter insertion 1 hph_status 0
<7>[ 3109.220172] wcd9xxx_detect_impedance: enter
<7>[ 3109.220208] wcd9xxx_detect_impedance: Acquiring BG_CLK
<7>[ 3109.220245] wcd9xxx_detect_impedance: Acquiring BG_CLK done
<7>[ 3109.220281] wcd9xxx_detect_impedance: Releasing BG_CLK
<7>[ 3109.220781] wcd9xxx_detect_impedance: Setting impedance detection
<7>[ 3109.252722] wcd9xxx_detect_impedance: Performing impedance detection
<7>[ 3109.419150] wcd9xxx_detect_impedance: Acquiring BG_CLK
<7>[ 3109.419194] wcd9xxx_detect_impedance: Acquiring BG_CLK done
<7>[ 3109.419232] wcd9xxx_detect_impedance: Releasing BG_CLK
<7>[ 3109.419750] wcd9xxx_detect_impedance: L0: 0x832(2098), L1:
0x7b5c(31580), L2: 0xe68(3688)
<7>[ 3109.419807] wcd9xxx_detect_impedance: R0: 0x7bc(1980), R1:
0x7ae2(31458), R2: 0xdae(3502)
<7>[ 3109.419852] wcd9xxx_detect_impedance: RL 34340 milliohm, RR 35869
milliohm
<7>[ 3109.419890] wcd9xxx_detect_impedance: Impedance detection completed
```

Impedance of HPHL and HPHR is calculated.

```
<7>[ 3109.419929] wcd9xxx_report_plug: Reporting insertion 3(3)
```

Reporting jack type 3 (SND_JACK_HEADSET = SND_JACK_MICROPHONE | SND_JACK_HEADPHONE).

The snd_jack_types enum is defined in the /kernel/include/sound/jack.h file.

```
<7>[ 3109.425114] wcd9xxx_insert_detect_setup: Setting up removal detection
```

Once the insertion is detected, MBHC is configured to detect removal.

```
<7>[ 3109.427066] wcd9xxx_report_plug: leave hph_status 3
<7>[ 3109.536250] wcd9xxx_start_hs_polling: enter
<7>[ 3109.537480] wcd9xxx_start_hs_polling: leave
<7>[ 3109.537519] wcd9xxx_find_plug_and_report: leave
<7>[ 3109.537553] wcd9xxx_correct_swch_plug: Release BCL
<7>[ 3109.537593] Attempt 1 found correct plug 1

<7>[ 3109.537651] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_MCLK_OFF(7)
<7>[ 3109.537690] wcd9xxx_pause_hs_polling: enter
<7>[ 3109.538310] wcd9xxx_pause_hs_polling: leave
<7>[ 3109.538346] wcd9xxx_event_notify: leave
<7>[ 3109.539212] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_MCLK_OFF(8)
<7>[ 3109.539252] wcd9xxx_event_notify: leave
<7>[ 3109.539303] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_RCO_ON(1)
<7>[ 3109.539340] wcd9xxx_event_notify: leave
<7>[ 3109.553848] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_RCO_ON(2)
<7>[ 3109.554100] wcd9xxx_update_mbhc_clk_rate: Updating clock rate
dependents, rate = 9600000
```

Change the clock source from MCLK to RCO and update the dependent MBHC calibration data.

```
<7>[ 3109.554814] wcd9xxx_update_mbhc_clk_rate: leave
<7>[ 3109.557686] wcd9xxx_start_hs_polling: enter
```

Start HS polling for button press detection.

```
<7>[ 3109.559350] wcd9xxx_start_hs_polling: leave
<7>[ 3109.559385] wcd9xxx_event_notify: leave
```

```
<7>[ 3109.560071] wcd9xxx_correct_swch_plug: Acquiring BCL
<7>[ 3109.560112] wcd9xxx_correct_swch_plug: Acquiring BCL done
<7>[ 3109.560146] wcd9xxx_correct_swch_plug: Release BCL
<7>[ 3109.560183] wcd9xxx_correct_swch_plug: leave current_plug(1)
```

21.3.2 Headset removal

```
<7>[ 4013.796447] wcd9xxx_mech_plug_detect_irq: enter
```

Mechanical insertion/removal; IRQ is called.

```
<7>[ 4013.796502] wcd9xxx_swch_irq_handler: enter
<7>[ 4013.801633] wcd9xxx_swch_irq_handler: Acquiring BCL
<7>[ 4013.801678] wcd9xxx_swch_irq_handler: Acquiring BCL done
<7>[ 4013.802131] wcd9xxx_swch_irq_handler: Current plug type 1, insert 0
```

insert 0 indicates removal. The log message indicates removal of the PLUG_TYPE_HEADSET plug type.

Plug types are defined in the wcd9xxx_mbhc_plug_type enum inside the kernel/sound/soc/codecs/wcd9xxx-mbhc.h file.

```
<7>[ 4013.802173] wcd9xxx_cancel_hs_detect_plug: Canceling
correct_plug_swch
<7>[ 4013.802211] wcd9xxx_cancel_hs_detect_plug: Release BCL
<7>[ 4013.802256] wcd9xxx_cancel_hs_detect_plug: Acquiring BCL
<7>[ 4013.802294] wcd9xxx_cancel_hs_detect_plug: Acquiring BCL done
<7>[ 4013.802329] wcd9xxx_pause_hs_polling: enter
<7>[ 4013.803022] wcd9xxx_pause_hs_polling: leave
<7>[ 4013.803060] wcd9xxx_shutdown_hs_removal_detect: Acquiring BG_CLK
<7>[ 4013.803100] wcd9xxx_shutdown_hs_removal_detect: Acquiring BG_CLK done
<7>[ 4013.803139] wcd9xxx_shutdown_hs_removal_detect: Releasing BG_CLK
<7>[ 4013.805072] wcd9xxx_shutdown_hs_removal_detect: Acquiring BG_CLK
<7>[ 4013.805117] wcd9xxx_shutdown_hs_removal_detect: Acquiring BG_CLK done
<7>[ 4013.805156] wcd9xxx_shutdown_hs_removal_detect: Releasing BG_CLK
<7>[ 4013.805386] wcd9xxx_cleanup_hs_polling: Acquiring BG_CLK
<7>[ 4013.805429] wcd9xxx_cleanup_hs_polling: Acquiring BG_CLK done
<7>[ 4013.805467] wcd9xxx_cleanup_hs_polling: Releasing BG_CLK
<7>[ 4013.805602] wcd9xxx_report_plug: enter insertion 0 hph_status 3
<7>[ 4013.805656] wcd9xxx_report_plug: Reporting removal 3(0)
```

Report the removal of the SND_JACK_HEADSET jack type; the snd_jack_types enum is defined in the /kernel/include/sound/jack.h file.

```
<7>[ 4013.805994] wcd9xxx_set_and_turnoff_hph_padac PA is off
```

```
<7>[ 4013.895202] __hphocp_off_report: clear ocp status 80
<7>[ 4013.895244] __hphocp_off_report: clear ocp status 40
<7>[ 4013.895281] wcd9xxx_insert_detect_setup: Setting up insert detection
<7>[ 4013.896029] wcd9xxx_report_plug: leave hph_status 0
<7>[ 4013.898357] wcd9xxx_swch_irq_handler: Release BCL
<7>[ 4013.898398] wcd9xxx_swch_irq_handler: leave
<7>[ 4013.898441] wcd9xxx_mech_plug_detect_irq: leave 1
```

21.3.3 Headphone insertion

```
<7>[ 4444.082541] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_HPHL_PA_OFF(44)
<7>[ 4444.082588] wcd9xxx_event_notify: leave
<7>[ 4444.096764] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_HPHR_PA_OFF(46)
<7>[ 4444.096809] wcd9xxx_event_notify: leave
<7>[ 4444.099574] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_MCLK_OFF(7)
<7>[ 4444.099616] wcd9xxx_event_notify: leave
<7>[ 4444.100740] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_MCLK_OFF(8)
<7>[ 4444.100782] wcd9xxx_event_notify: leave
<7>[ 4444.100834] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_RCO_ON(1)
<7>[ 4444.100870] wcd9xxx_event_notify: leave
<7>[ 4444.116060] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_RCO_ON(2)
<7>[ 4444.116370] wcd9xxx_update_mbhc_clk_rate: Updating clock rate
dependents, rate = 9600000
<7>[ 4444.117443] wcd9xxx_update_mbhc_clk_rate: leave
<7>[ 4444.117479] wcd9xxx_event_notify: leave
<6>[ 4444.118272] tapan_codec_chargepump_vdd_event: event = 8
<7>[ 4449.134848] wcd9xxx_mech_plug_detect_irq: enter
```

Mechanical insertion/removal; IRQ is called.

```
<7>[ 4449.134903] wcd9xxx_swch_irq_handler: enter
<7>[ 4449.140048] wcd9xxx_swch_irq_handler: Acquiring BCL
<7>[ 4449.140092] wcd9xxx_swch_irq_handler: Acquiring BCL done
<7>[ 4449.140555] wcd9xxx_swch_irq_handler: Current plug type 0, insert 1
<7>[ 4449.140596] wcd9xxx_cancel_hs_detect_plug: Canceling
correct_plug_swch
<7>[ 4449.140632] wcd9xxx_cancel_hs_detect_plug: Release BCL
<7>[ 4449.140677] wcd9xxx_cancel_hs_detect_plug: Acquiring BCL
<7>[ 4449.140715] wcd9xxx_cancel_hs_detect_plug: Acquiring BCL done
<7>[ 4449.141136] wcd9xxx_mbhc_detect_plug_type: enter
<7>[ 4449.141518] wcd9xxx_mbhc_decide_swch_plug: enter
```

```
<7>[ 4449.141557] wcd9xxx_turn_onoff_current_source: enabling current
source
<7>[ 4449.142826] wcd9xxx_codec_cs_get_plug_type: enter
<7>[ 4449.144774] wcd9xxx_mbhc_setup_hs_polling: enter
<7>[ 4449.144813] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK
<7>[ 4449.144851] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK done
<7>[ 4449.144890] wcd9xxx_mbhc_setup_hs_polling: Releasing BG_CLK
<7>[ 4449.238476] wcd9xxx_cs_find_plug_type: enter
<7>[ 4449.238516] wcd9xxx_cs_find_plug_type: event_state 0x0
<7>[ 4449.238566] wcd9xxx_cs_find_plug_type: DCE #0, ffe6, V 0836(0836),
HPHL 1 TYPE 3
<7>[ 4449.238618] wcd9xxx_cs_find_plug_type: DCE #1, ffe6, V 0836(0836),
HPHL 1 TYPE 3
<7>[ 4449.238669] wcd9xxx_cs_find_plug_type: DCE #2, ffe6, V 0836(0836),
HPHL 1 TYPE 3
<7>[ 4449.238719] wcd9xxx_cs_find_plug_type: DCE #3, ffe5, V 0836(0836),
HPHL 1 TYPE 3
<7>[ 4449.238763] wcd9xxx_cs_find_plug_type: Plug type 3 detected
<7>[ 4449.238798] wcd9xxx_codec_cs_get_plug_type: plug_type:3
```

PLUG_TYPE_HIGH_HPH is detected. In such a scenario, MBHC continues for 5 sec to ensure whether the headset or headphone is inserted. It may be possible that during the measurement, the plug was not inserted. The plug type enum is wcd9xxx_mbhc_
plug_type, and it is defined inside the kernel/sound/soc/codecs/wcd9xxx-mbhc.h file.

```
<7>[ 4449.238833] wcd9xxx_turn_onoff_current_source: disabling current
source
<7>[ 4449.240455] wcd9xxx_schedule_hs_detect_plug: scheduling
wcd9xxx_correct_swch_plug
<7>[ 4449.240516] wcd9xxx_mbhc_decide_swch_plug: leave
<7>[ 4449.240550] wcd9xxx_mbhc_detect_plug_type: leave
<7>[ 4449.240582] wcd9xxx_swch_irq_handler: Release BCL
<7>[ 4449.240614] wcd9xxx_swch_irq_handler: leave
<7>[ 4449.240654] wcd9xxx_mech_plug_detect_irq: leave 1
<7>[ 4449.240783] wcd9xxx_correct_swch_plug: enter
<7>[ 4449.241302] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_RCO_OFF(3)
<7>[ 4449.241343] wcd9xxx_event_notify: leave
<7>[ 4449.242192] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_RCO_OFF(4)
<7>[ 4449.242233] wcd9xxx_event_notify: leave
<7>[ 4449.242284] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_MCLK_ON(5)
<7>[ 4449.242320] wcd9xxx_event_notify: leave
<7>[ 4449.245171] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_MCLK_ON(6)
<7>[ 4449.245419] wcd9xxx_update_mbhc_clk_rate: Updating clock rate
dependents, rate = 9600000
```

```
<7>[ 4449.250223] wcd9xxx_update_mbhc_clk_rate: leave
```

The logs indicate the change in the clock source from RCO to MCLK. wcd9xxx_update_mbhc_clk_rate is called to update MBHC calibration data required due to the change in the clock rate.

```
<7>[ 4449.250361] wcd9xxx_event_notify: leave
<7>[ 4449.250402] wcd9xxx_turn_onoff_current_source: enabling current
source
<7>[ 4449.356821] wcd9xxx_correct_swch_plug: Acquiring BCL
<7>[ 4449.356865] wcd9xxx_correct_swch_plug: Acquiring BCL done
<7>[ 4449.356899] wcd9xxx_codec_cs_get_plug_type: enter
<7>[ 4449.358789] wcd9xxx_mbhc_setup_hs_polling: enter
```

Start the polling to determine the correct plug type as the plug detected incorrect headphone insertion.

```
<7>[ 4449.358827] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK
<7>[ 4449.358866] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK done
<7>[ 4449.358904] wcd9xxx_mbhc_setup_hs_polling: Releasing BG_CLK
<7>[ 4449.447708] wcd9xxx_cs_find_plug_type: enter
<7>[ 4449.447748] wcd9xxx_cs_find_plug_type: event_state 0x0
<7>[ 4449.447796] wcd9xxx_cs_find_plug_type: DCE #0, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4449.447846] wcd9xxx_cs_find_plug_type: DCE #1, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4449.447896] wcd9xxx_cs_find_plug_type: DCE #2, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4449.447946] wcd9xxx_cs_find_plug_type: DCE #3, f7c7, V 0002(0002),
HPHL 1 TYPE 2
```

All four DCE measurements are consistent, and the plug type 2 (PLUG_TYPE_HEADPHONE) is detected. The plug types defined are the wcd9xxx_mbhc_plug_type enum inside the kernel/sound/soc/codecs/wcd9xxx-mbhc.h file.

```
<7>[ 4449.447989] wcd9xxx_cs_find_plug_type: Plug type 2 detected
<7>[ 4449.448024] wcd9xxx_codec_cs_get_plug_type: plug_type:2
```

PLUG_TYPE_HEADPHONE is detected. In such a scenario, MBHC continues for 5 sec to ensure whether the headset or headphone is inserted. It may be possible that during the measurement, the plug was not inserted.

```
<7>[ 4449.448057] wcd9xxx_correct_swch_plug: Release BCL
<7>[ 4449.448098] wcd9xxx_correct_swch_plug: attempt(1) current_plug(0)
new_plug(2)
<7>[ 4449.448136] Good headphone detected, continue polling
```

PLUG_TYPE_HEADPHONE is detected. However, continue polling to ensure that the plug type is indeed headphone. Plug types are defined in the wcd9xxx_mbhc_plug_type enum inside the kernel/sound/soc/codecs/wcd9xxx-mbhc.h file.

```
<6>[ 4449.466618] wcd9xxx_report_plug: BCL should have acquired
 . . .
<7>[ 4449.494799] wcd9xxx_detect_impedance: Acquiring BG_CLK
<7>[ 4449.494837] wcd9xxx_detect_impedance: Acquiring BG_CLK done
<7>[ 4449.494873] wcd9xxx_detect_impedance: Releasing BG_CLK
<7>[ 4449.498811] wcd9xxx_detect_impedance: Setting impedance detection
<7>[ 4449.528825] wcd9xxx_detect_impedance: Performing impedance detection
<7>[ 4449.696523] wcd9xxx_detect_impedance: Acquiring BG_CLK
<7>[ 4449.696566] wcd9xxx_detect_impedance: Acquiring BG_CLK done
<7>[ 4449.696604] wcd9xxx_detect_impedance: Releasing BG_CLK
<7>[ 4449.697132] wcd9xxx_detect_impedance: L0: 0x832(2098), L1:
0x7b4e(31566), L2: 0xe68(3688)
<7>[ 4449.697190] wcd9xxx_detect_impedance: R0: 0x7be(1982), R1:
0x7ae6(31462), R2: 0xdac(3500)
<7>[ 4449.697234] wcd9xxx_detect_impedance: RL 34323 milliohm, RR 35966
milliohm
<7>[ 4449.697273] wcd9xxx_detect_impedance: Impedance detection completed
```

Impedance of HPHL and HPHR is calculated.

```
<7>[ 4449.697311] wcd9xxx_report_plug: Reporting insertion 1(1)
```

Report jack type 1 (SND_JACK_HEADPHONE); the snd_jack_types enum is defined in the /kernel/include/sound/jack.h file.

```
<7>[ 4449.702752] wcd9xxx_insert_detect_setup: Setting up removal detection
```

Once the insertion is detected, MBHC is configured to detect removal.

```
<7>[ 4449.703764] wcd9xxx_report_plug: leave hph_status 1
<7>[ 4449.806845] wcd9xxx_correct_swch_plug: Acquiring BCL
<7>[ 4449.806892] wcd9xxx_correct_swch_plug: Acquiring BCL done
<7>[ 4449.806928] wcd9xxx_codec_cs_get_plug_type: enter
<7>[ 4449.808850] wcd9xxx_mbhc_setup_hs_polling: enter
<7>[ 4449.808889] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK
<7>[ 4449.808929] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK done
<7>[ 4449.808969] wcd9xxx_mbhc_setup_hs_polling: Releasing BG_CLK
<7>[ 4449.897983] wcd9xxx_cs_find_plug_type: enter
<7>[ 4449.898026] wcd9xxx_cs_find_plug_type: event_state 0x0
```

```
<7>[ 4449.898075] wcd9xxx_cs_find_plug_type: DCE #0, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4449.898127] wcd9xxx_cs_find_plug_type: DCE #1, f7c6, V 0001(0001),
HPHL 1 TYPE 2
<7>[ 4449.898296] wcd9xxx_cs_find_plug_type: DCE #2, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4449.898349] wcd9xxx_cs_find_plug_type: DCE #3, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4449.898394] wcd9xxx_cs_find_plug_type: Plug type 2 detected
<7>[ 4449.898430] wcd9xxx_codec_cs_get_plug_type: plug_type:2
```

PLUG_TYPE_HEADPHONE is detected. The plug type enum is wcd9xxx_mbhc_plug_type and is defined inside the kernel/sound/soc/codecs/wcd9xxx-mbhc.h file.

```
<7>[ 4449.898463] wcd9xxx_correct_swch_plug: Release BCL
<7>[ 4449.898506] wcd9xxx_correct_swch_plug: attempt(2) current_plug(2)
new_plug(2)
<7>[ 4449.898544] Good headphone detected, continue polling
<7>[ 4450.006817] wcd9xxx_correct_swch_plug: Acquiring BCL
<7>[ 4450.006858] wcd9xxx_correct_swch_plug: Acquiring BCL done
<7>[ 4450.006893] wcd9xxx_codec_cs_get_plug_type: enter
<7>[ 4450.008779] wcd9xxx_mbhc_setup_hs_polling: enter
<7>[ 4450.008815] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK
<7>[ 4450.008854] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK done
<7>[ 4450.008892] wcd9xxx_mbhc_setup_hs_polling: Releasing BG_CLK
<7>[ 4450.097826] wcd9xxx_cs_find_plug_type: enter
<7>[ 4450.097865] wcd9xxx_cs_find_plug_type: event_state 0x0
<7>[ 4450.097912] wcd9xxx_cs_find_plug_type: DCE #0, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4450.097963] wcd9xxx_cs_find_plug_type: DCE #1, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4450.098013] wcd9xxx_cs_find_plug_type: DCE #2, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4450.098062] wcd9xxx_cs_find_plug_type: DCE #3, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4450.098104] wcd9xxx_cs_find_plug_type: Plug type 2 detected
<7>[ 4450.098139] wcd9xxx_codec_cs_get_plug_type: plug_type:2
<7>[ 4450.098171] wcd9xxx_correct_swch_plug: Release BCL
<7>[ 4450.098211] wcd9xxx_correct_swch_plug: attempt(3) current_plug(2)
new_plug(2)
 . ..
<7>[ 4454.098418] wcd9xxx_correct_swch_plug: attempt(23) current_plug(2)
new_plug(2)
<7>[ 4454.098455] Good headphone detected, continue polling
<7>[ 4454.206289] wcd9xxx_correct_swch_plug: Acquiring BCL
<7>[ 4454.206331] wcd9xxx_correct_swch_plug: Acquiring BCL done
<7>[ 4454.206366] wcd9xxx_codec_cs_get_plug_type: enter
<7>[ 4454.208253] wcd9xxx_mbhc_setup_hs_polling: enter
```

```
<7>[ 4454.208291] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK
<7>[ 4454.208330] wcd9xxx_mbhc_setup_hs_polling: Acquiring BG_CLK done
<7>[ 4454.208367] wcd9xxx_mbhc_setup_hs_polling: Releasing BG_CLK
<7>[ 4454.297547] wcd9xxx_cs_find_plug_type: enter
<7>[ 4454.297585] wcd9xxx_cs_find_plug_type: event_state 0x0
<7>[ 4454.297632] wcd9xxx_cs_find_plug_type: DCE #0, f7c5, V 0001(0001),
HPHL 1 TYPE 2
<7>[ 4454.297683] wcd9xxx_cs_find_plug_type: DCE #1, f7c6, V 0001(0001),
HPHL 1 TYPE 2
<7>[ 4454.297733] wcd9xxx_cs_find_plug_type: DCE #2, f7c5, V 0001(0001),
HPHL 1 TYPE 2
<7>[ 4454.297783] wcd9xxx_cs_find_plug_type: DCE #3, f7c7, V 0002(0002),
HPHL 1 TYPE 2
<7>[ 4454.297825] wcd9xxx_cs_find_plug_type: Plug type 2 detected
<7>[ 4454.297861] wcd9xxx_codec_cs_get_plug_type: plug_type:2
<7>[ 4454.297893] wcd9xxx_correct_swch_plug: Release BCL
<7>[ 4454.297933] wcd9xxx_correct_swch_plug: attempt(24) current_plug(2)
new_plug(2)
<7>[ 4454.297970] Good headphone detected, continue polling
<7>[ 4454.298006] wcd9xxx_turn_onoff_current_source: disabling current
source
<7>[ 4454.299226] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_MCLK_OFF(7)
<7>[ 4454.299268] wcd9xxx_event_notify: leave
<7>[ 4454.300091] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_MCLK_OFF(8)
<7>[ 4454.300131] wcd9xxx_event_notify: leave
<7>[ 4454.300182] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_RCO_ON(1)
<7>[ 4454.300219] wcd9xxx_event_notify: leave
<7>[ 4454.316342] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_RCO_ON(2)
<7>[ 4454.316593] wcd9xxx_update_mbhc_clk_rate: Updating clock rate
dependents, rate = 9600000
<7>[ 4454.317299] wcd9xxx_update_mbhc_clk_rate: leave
<7>[ 4454.317335] wcd9xxx_event_notify: leave
```

Change the clock source from MCLK to RCO and update the dependent MBHC calibration data.

```
<7>[ 4454.317823] wcd9xxx_correct_swch_plug: Acquiring BCL
<7>[ 4454.317863] wcd9xxx_correct_swch_plug: Acquiring BCL done
<7>[ 4454.317899] wcd9xxx_shutdown_hs_removal_detect: Acquiring BG_CLK
<7>[ 4454.317937] wcd9xxx_shutdown_hs_removal_detect: Acquiring BG_CLK done
<7>[ 4454.317973] wcd9xxx_shutdown_hs_removal_detect: Releasing BG_CLK
<7>[ 4454.319601] wcd9xxx_shutdown_hs_removal_detect: Acquiring BG_CLK
<7>[ 4454.319647] wcd9xxx_shutdown_hs_removal_detect: Acquiring BG_CLK done
<7>[ 4454.319685] wcd9xxx_shutdown_hs_removal_detect: Releasing BG_CLK
<7>[ 4454.319895] wcd9xxx_cleanup_hs_polling: Acquiring BG_CLK
```

```
<7>[ 4454.319936] wcd9xxx_cleanup_hs_polling: Acquiring BG_CLK done
<7>[ 4454.319973] wcd9xxx_cleanup_hs_polling: Releasing BG_CLK
```

After 5 sec, the polling is also stopped. [4449.448136] ~[4454.319973]

```
<7>[ 4454.320014] wcd9xxx_enable_hs_detect: enter insertion(0) trigger(0x0)
<7>[ 4454.320529] setup for removal detection
```

Once the insertion is detected, MBHC is configured to detect removal.

```
<7>[ 4454.323924] wcd9xxx_enable_hs_detect: leave
<7>[ 4454.323960] wcd9xxx_correct_swch_plug: Release BCL
<7>[ 4454.323997] wcd9xxx_correct_swch_plug: leave current_plug(2)
```

21.3.4 Headphone removal

```
<7>[ 4473.151034] wcd9xxx_hs_insert_irq: enter
```

Electrical MBHC IRQ is triggered, because electrical MBHC is also present to detect the aux cable insertion and removal.

```
<7>[ 4473.151076] wcd9xxx_hs_insert_irq: Acquiring BCL
<7>[ 4473.151113] wcd9xxx_hs_insert_irq: Acquiring BCL done
<7>[ 4473.152669] wcd9xxx_hs_insert_irq_swch: Removal
<7>[ 4473.153080] wcd9xxx_hs_insert_irq: Release BCL
<7>[ 4473.153353] wcd9xxx_mech_plug_detect_irq: enter
```

Mechanical insertion/removal; IRQ is called.

```
<7>[ 4473.153402] wcd9xxx_swch_irq_handler: enter
<7>[ 4473.158515] wcd9xxx_swch_irq_handler: Acquiring BCL
<7>[ 4473.158558] wcd9xxx_swch_irq_handler: Acquiring BCL done
<7>[ 4473.158975] wcd9xxx_swch_irq_handler: Current plug type 2, insert 0
```

Insert zero means removal. The log message indicates removal of the PLUG_TYPE_HEADPHONE plug type.

Plug types are defined in the wcd9xxx_mbhc_plug_type enum in the kernel/sound/soc/codecs/wcd9xxx-mbhc.h file.

```
<7>[ 4473.159017] wcd9xxx_cancel_hs_detect_plug: Canceling
correct_plug_swch
<7>[ 4473.159053] wcd9xxx_cancel_hs_detect_plug: Release BCL
<7>[ 4473.159097] wcd9xxx_cancel_hs_detect_plug: Acquiring BCL
```

```
<7>[ 4473.159135] wcd9xxx_cancel_hs_detect_plug: Acquiring BCL done
<7>[ 4473.159176] wcd9xxx_report_plug: enter insertion 0 hph_status 1
<7>[ 4473.159225] wcd9xxx_report_plug: Reporting removal 1(0)
```

Report the removal of SND_JACK_HEADPHONE jack type. The snd_jack_types enum is defined inside the /kernel/include/sound/jack.h file.

```
<7>[ 4473.159592] wcd9xxx_set_and_turnoff_hph_padac PA is off
<7>[ 4473.248803] __hphocp_off_report: clear ocp status 80
<7>[ 4473.248846] __hphocp_off_report: clear ocp status 40
<7>[ 4473.248884] wcd9xxx_insert_detect_setup: Setting up insert detection
<7>[ 4473.249547] wcd9xxx_report_plug: leave hph_status 0
<7>[ 4473.251191] wcd9xxx_swch_irq_handler: Release BCL
<7>[ 4473.251231] wcd9xxx_swch_irq_handler: leave
<7>[ 4473.251275] wcd9xxx_mech_plug_detect_irq: leave 1
```

21.3.5 Euro headset insertion

This section is not applicable to this release.

21.3.6 Button press logs

21.3.6.1 Single button press

```
<7>[ 4324.016670] wcd9xxx_dce_handler: enter
<7>[ 4324.016710] wcd9xxx_dce_handler: Acquiring BCL
<7>[ 4324.016746] wcd9xxx_dce_handler: Acquiring BCL done
<7>[ 4324.068580] wcd9xxx_dce_handler: Meas HW - STA 0xe5b5,1,1
<7>[ 4324.068629] wcd9xxx_dce_handler: Meas HW - DCE 0xfbe7,1,1 button 0
<7>[ 4324.068676] wcd9xxx_dce_handler: Meas 1 - DCE 0xfbe7,1,1 button 0
```

Mic voltage corresponding to button 0 is detected. The <7>[4324.068580] wcd9xxx_dce_handler: Meas HW - STA 0xe5b5,1,1 message is printed from pr_debug("%s: Meas HW - STA 0x%x,%d,%d\n", __func__, sta & 0xFFFF, stamv, stamv_s); where:

- sta & 0xFFFF – Value from the hardware register
- stamv – Value in mV converted using a formula in __wcd9xxx_codec_sta_dce_v()
- stamv_s – Scaled value in mV converted using the formula in scale_v_micb_vddio()

Similar logic applies to DCE measurement.

```
<7>[ 4324.068716] wcd9xxx_update_rel_threshold: reprogram vb1hu/vbrh to
60mv
<7>[ 4324.069486] wcd9xxx_dce_handler: leave
<7>[ 4324.069519] wcd9xxx_dce_handler: Release BCL
<7>[ 4324.168120] wcd9xxx_release_handler: enter
```

```
<7>[ 4324.168157] wcd9xxx_release_handler: Acquiring BCL
<7>[ 4324.168194] wcd9xxx_release_handler: Acquiring BCL done
<7>[ 4324.186633] wcd9xxx_is_fake_press: STA[0]: 6180,1587
<7>[ 4324.217768] wcd9xxx_is_fake_press: DCE[1]: 1150,1737
<7>[ 4324.248701] wcd9xxx_is_fake_press: DCE[2]: 1238,1806
<7>[ 4324.248740] wcd9xxx_release_handler: Reporting btn press
```

Report button press.

```
<7>[ 4324.248910] wcd9xxx_release_handler: Reporting btn release
```

Report button release.

```
<6>[ 4324.267399] msm_compr_open: session ID 1
```

Due to button press, a playback session is opened.

```
<7>[ 4324.297437] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_RCO_OFF(3)
<7>[ 4324.297450] wcd9xxx_pause_hs_polling: enter
<7>[ 4324.297915] wcd9xxx_pause_hs_polling: leave
<7>[ 4324.297925] wcd9xxx_event_notify: leave
<7>[ 4324.298569] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_RCO_OFF(4)
<7>[ 4324.298581] wcd9xxx_event_notify: leave
<7>[ 4324.298595] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_MCLK_ON(5)
<7>[ 4324.298605] wcd9xxx_event_notify: leave
<7>[ 4324.301239] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_MCLK_ON(6)
<7>[ 4324.301415] wcd9xxx_update_mbhc_clk_rate: Updating clock rate
dependents, rate = 9600000
```

The logs indicate the change in the clock source from RCO to MCLK. wcd9xxx_update_mbhc_clk_rate is called to update the MBHC calibration data required due to change in the clock rate.

```
<7>[ 4324.302076] wcd9xxx_update_mbhc_clk_rate: leave
<7>[ 4324.303738] wcd9xxx_start_hs_polling: enter
<7>[ 4324.304702] wcd9xxx_start_hs_polling: leave
<7>[ 4324.304712] wcd9xxx_event_notify: leave
<6>[ 4324.304872] tapan_codec_chargepump_vdd_event: event = 1
<7>[ 4324.305809] wcd9xxx_start_hs_polling: enter
<7>[ 4324.307337] wcd9xxx_start_hs_polling: leave
<7>[ 4324.307347] wcd9xxx_release_handler: leave
<7>[ 4324.307356] wcd9xxx_release_handler: Release BCL
```

```
<7>[ 4324.317543] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_HPHL_PA_ON(43)
<7>[ 4324.317558] wcd9xxx_pause_hs_polling: enter
<7>[ 4324.317997] wcd9xxx_pause_hs_polling: leave
```

Pause HS polling (polling of mic line voltage).

```
<7>[ 4324.340290] __wcd9xxx_switch_micbias: Programmed MBHC thresholds to
VDDIO
```

During playback, instead of using the micbias for polling, VDDIO source (1.8 V) is used for HS polling for button detection to avoid polling noise during playback.

```
<7>[ 4324.340912] wcd9xxx_start_hs_polling: enter
<7>[ 4324.341867] wcd9xxx_start_hs_polling: leave
<7>[ 4324.341877] __wcd9xxx_switch_micbias: VDDIO switch enabled
```

VDDIO switch is enabled.

```
<7>[ 4324.341886] wcd9xxx_event_notify: leave
<7>[ 4324.341899] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_HPHR_PA_ON(45)
<7>[ 4324.341910] wcd9xxx_event_notify: leave
```

21.3.6.2 Double button press

Double button press is like two single button presses. It is up to the user space to maintain the time interval in which it receives the two successive button presses to decide double button press.

```
<7>[ 4384.618769] wcd9xxx_dce_handler: enter
<7>[ 4384.618809] wcd9xxx_dce_handler: Acquiring BCL
<7>[ 4384.618846] wcd9xxx_dce_handler: Acquiring BCL done
<7>[ 4384.672883] wcd9xxx_dce_handler: Meas HW - STA 0xe5a0,1,1
<7>[ 4384.672949] wcd9xxx_dce_handler: Meas HW - DCE 0xfbe3,0,0 button 0
<7>[ 4384.673013] wcd9xxx_dce_handler: Meas 1 - DCE 0xfbe3,0,0 button 0
```

Mic voltage corresponding to button 0 is detected.

```
<7>[ 4384.673054] wcd9xxx_update_rel_threshold: reprogram vb1hu/vbrh to
60mv
<7>[ 4384.673985] wcd9xxx_dce_handler: leave
<7>[ 4384.674020] wcd9xxx_dce_handler: Release BCL
<7>[ 4384.687905] wcd9xxx_release_handler: enter
<7>[ 4384.687940] wcd9xxx_release_handler: Acquiring BCL
<7>[ 4384.687977] wcd9xxx_release_handler: Acquiring BCL done
```

```
<7>[ 4384.707529] wcd9xxx_is_fake_press: STA[0]: 1764,1046
<7>[ 4384.738902] wcd9xxx_is_fake_press: DCE[1]: 384,1133
<7>[ 4384.770572] wcd9xxx_is_fake_press: DCE[2]: 437,1175
<7>[ 4384.770610] wcd9xxx_release_handler: Reporting btn press
```

Report button press.

```
<7>[ 4384.770790] wcd9xxx_release_handler: Reporting btn release
```

Report button release.

```
<7>[ 4384.825791] wcd9xxx_start_hs_polling: enter
<7>[ 4384.826620] wcd9xxx_start_hs_polling: leave
```

Start HS polling (polling of mic line voltage).

```
<7>[ 4384.826630] wcd9xxx_release_handler: leave
<7>[ 4384.826639] wcd9xxx_release_handler: Release BCL
<7>[ 4384.851122] wcd9xxx_dce_handler: enter
<7>[ 4384.851133] wcd9xxx_dce_handler: Acquiring BCL
<7>[ 4384.851144] wcd9xxx_dce_handler: Acquiring BCL done
<7>[ 4384.898712] wcd9xxx_dce_handler: Meas HW - STA 0xe5a0,1,1
<7>[ 4384.898730] wcd9xxx_dce_handler: Meas HW - DCE 0xfbe3,0,0 button 0
<7>[ 4384.898747] wcd9xxx_dce_handler: Meas 1 - DCE 0xfbe3,0,0 button 0
<7>[ 4384.898758] wcd9xxx_update_rel_threshold: reprogram vb1hu/vbrh to
60mv
```

Mic voltage corresponding to button 0 is detected.

```
<7>[ 4384.899410] wcd9xxx_dce_handler: leave
<7>[ 4384.899419] wcd9xxx_dce_handler: Release BCL
<7>[ 4384.904464] wcd9xxx_release_handler: enter
<7>[ 4384.904473] wcd9xxx_release_handler: Acquiring BCL
<7>[ 4384.904484] wcd9xxx_release_handler: Acquiring BCL done
<7>[ 4384.921582] wcd9xxx_is_fake_press: STA[0]: 1700,1038
<7>[ 4384.953072] wcd9xxx_is_fake_press: DCE[1]: 375,1126
<7>[ 4384.984696] wcd9xxx_is_fake_press: DCE[2]: 434,1173
<7>[ 4384.984734] wcd9xxx_release_handler: Reporting btn press
```

Button press is reported.

```
<7>[ 4384.984912] wcd9xxx_release_handler: Reporting btn release
```

Button release is reported.

```
<7>[ 4385.045831] wcd9xxx_start_hs_polling: enter
<7>[ 4385.046336] wcd9xxx_start_hs_polling: leave
<7>[ 4385.046346] wcd9xxx_release_handler: leave
<7>[ 4385.046355] wcd9xxx_release_handler: Release BCL
<7>[ 4385.062478] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_HPHL_PA_OFF(44)
<7>[ 4385.062491] wcd9xxx_event_notify: leave
<7>[ 4385.076675] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_HPHR_PA_OFF(46)
<7>[ 4385.076688] wcd9xxx_pause_hs_polling: enter
<7>[ 4385.077010] wcd9xxx_pause_hs_polling: leave
<7>[ 4385.088142] __wcd9xxx_switch_micbias: Programmed MBHC thresholds to
MICBIAS
```

Program the threshold used in MBHC for button press and release detection based on the mic bias voltage.

```
<7>[ 4385.088278] wcd9xxx_start_hs_polling: enter
<7>[ 4385.088944] wcd9xxx_start_hs_polling: leave
<7>[ 4385.088954] __wcd9xxx_switch_micbias: VDDIO switch disabled
```

Disable the VDDIO switch.

```
<7>[ 4385.088964] wcd9xxx_event_notify: leave
<7>[ 4385.090690] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_MCLK_OFF(7)
<7>[ 4385.090702] wcd9xxx_pause_hs_polling: enter
<7>[ 4385.091012] wcd9xxx_pause_hs_polling: leave
<7>[ 4385.091022] wcd9xxx_event_notify: leave
<7>[ 4385.091543] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_MCLK_OFF(8)
<7>[ 4385.091554] wcd9xxx_event_notify: leave
<7>[ 4385.091569] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_RCO_ON(1)
<7>[ 4385.091578] wcd9xxx_event_notify: leave
<7>[ 4385.105118] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_RCO_ON(2)
<7>[ 4385.105258] wcd9xxx_update_mbhc_clk_rate: Updating clock rate
dependents, rate = 9600000
```

Change in the clock source from MCLK to RCO. wcd9xxx_update_mbhc_clk_rate is called to update the MBHC calibration data required due to a change in the clock rate.

```
<7>[ 4385.105750] wcd9xxx_update_mbhc_clk_rate: leave
<7>[ 4385.107369] wcd9xxx_start_hs_polling: enter
<7>[ 4385.108201] wcd9xxx_start_hs_polling: leave
<7>[ 4385.108211] wcd9xxx_event_notify: leave
<6>[ 4385.108584] tapan_codec_chargepump_vdd_event: event = 8
<6>[ 4385.222743] msm_compr_open: session ID 1
```

Due to button press, playback session is opened.

```
<7>[ 4385.253017] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_RCO_OFF(3)
<7>[ 4385.253031] wcd9xxx_pause_hs_polling: enter
<7>[ 4385.253536] wcd9xxx_pause_hs_polling: leave
<7>[ 4385.253546] wcd9xxx_event_notify: leave
<7>[ 4385.254190] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_RCO_OFF(4)
<7>[ 4385.254201] wcd9xxx_event_notify: leave
<7>[ 4385.254216] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_MCLK_ON(5)
<7>[ 4385.254226] wcd9xxx_event_notify: leave
<7>[ 4385.256869] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_POST_MCLK_ON(6)
<7>[ 4385.257037] wcd9xxx_update_mbhc_clk_rate: Updating clock rate
dependents, rate = 9600000
```

Change in the clock source from RCO to MCLK. wcd9xxx_update_mbhc_clk_rate is called to update the MBHC calibration data required due to a change in the clock rate.

```
<7>[ 4385.257772] wcd9xxx_update_mbhc_clk_rate: leave
<7>[ 4385.259413] wcd9xxx_start_hs_polling: enter
<7>[ 4385.260369] wcd9xxx_start_hs_polling: leave
<7>[ 4385.260379] wcd9xxx_event_notify: leave
<6>[ 4385.260536] tapan_codec_chargepump_vdd_event: event = 1
<7>[ 4385.272539] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_HPHL_PA_ON(43)
<7>[ 4385.272554] wcd9xxx_pause_hs_polling: enter
<7>[ 4385.272993] wcd9xxx_pause_hs_polling: leave
<7>[ 4385.295289] __wcd9xxx_switch_micbias: Programmed MBHC thresholds to
VDDIO
```

During playback, instead of using the micbias for polling, VDDIO source (1.8 V) is used for HS polling for button detection to avoid polling noise during playback.

```
<7>[ 4385.296374] wcd9xxx_start_hs_polling: enter
<7>[ 4385.297327] wcd9xxx_start_hs_polling: leave
<7>[ 4385.297337] __wcd9xxx_switch_micbias: VDDIO switch enabled
```

VDDIO switch is enabled.

```
<7>[ 4385.297347] wcd9xxx_event_notify: leave
<7>[ 4385.297359] wcd9xxx_event_notify: enter event
WCD9XXX_EVENT_PRE_HPHR_PA_ON(45)
<7>[ 4385.297370] wcd9xxx_event_notify: leave
```

21.3.6.3 Long button press

This section is not applicable to this release.

21.3.6.4 Invalid short button press

This section is not applicable to this release.

21.4 Customization

21.4.1 Micbias external capacitor configuration

The Micbias external capacitor mode is configured through the device tree. By default, all mic bias does not have an external cap. If the device needs an external capacitor mode, it is configured using qcom,cdc-micbiasX-ext-cap in the device tree file.

For MSM8226 MTP, the change is in arch/arm/boot/dts/msm8226-mtp.dtsi.

```
&slim_msm {
        tapan_codec {
                qcom,cdc-micbias2-ext-cap;
        };
     };
```

For MSM8974, the change is in the arch/arm/boot/dts/msm8974-mtp.dtsi file. The micbias 2 of MSM8974 MTP has external capacitor configuration.

```
&slim_msm {
        taiko_codec {
                qcom,cdc-micbias2-ext-cap;
        };
     };
```

21.4.2 MBHC calibration in ACDB

The current source method has been introduced to detect the plug type during the insertion. The current source is also enabled during removal. This helps reduce pop noise during insertion and removal. ACDB support has not been introduced to change the following thresholds through ACDB.

```
#define WCD9XXX_CS_MEAS_INVALD_RANGE_LOW_MV 110
#define WCD9XXX_CS_MEAS_INVALD_RANGE_HIGH_MV 265
#define WCD9XXX_V_CS_HS_MAX 500
#define WCD9XXX_V_CS_NO_MIC 5
```

Hence, currently the read_fw_bin variable in mbhc_cfg is set to false. After QACT support is provided, set the variable read_fw_bin to true in the /kernel/sound/soc/msm8226.c file or the OEM-specific machine driver so that calibration data from the ACDB is used.

```
static struct wcd9xxx_mbhc_config mbhc_cfg = {
   .read_fw_bin = false,   /* true means use ACDB MBHC calibraton data */
   .calibration = NULL,
   .micbias = MBHC_MICBIAS2,
   .mclk_cb_fn = msm_snd_enable_codec_ext_clk,
   .mclk_rate = TAPAN_EXT_CLK_RATE,
 . . .
```

21.4.3 MBHC calibration in the machine driver

There may be a difference in values or issues with the MBHC calibration data from the ACDB. To isolate the issue, the OEM can use the MBHC calibration data from the machine driver. The MBHC calibration data is defined in the msm8226.c machine driver in the def_tapan_mbhc_cal( ) function. The OEM can hardcode the MBHC calibration data and rebuild the APSS image for the updated calibration data to take effect.

If mbhc_cfg.read_fw_bin is true, MBHC calibration is picked up from the ACDB. Else, the MBHC calibration data is used from the machine driver. The flag is set to true or false in the msm_aud_rx_init() function in the kernel/sound/soc/msm/msm8226.c file.

21.4.4 MBHC tuning parameters

This section describes the MBHC tuning parameters. This section will be updated when QACT supports calibration data for the current source method.

21.5 Debugging

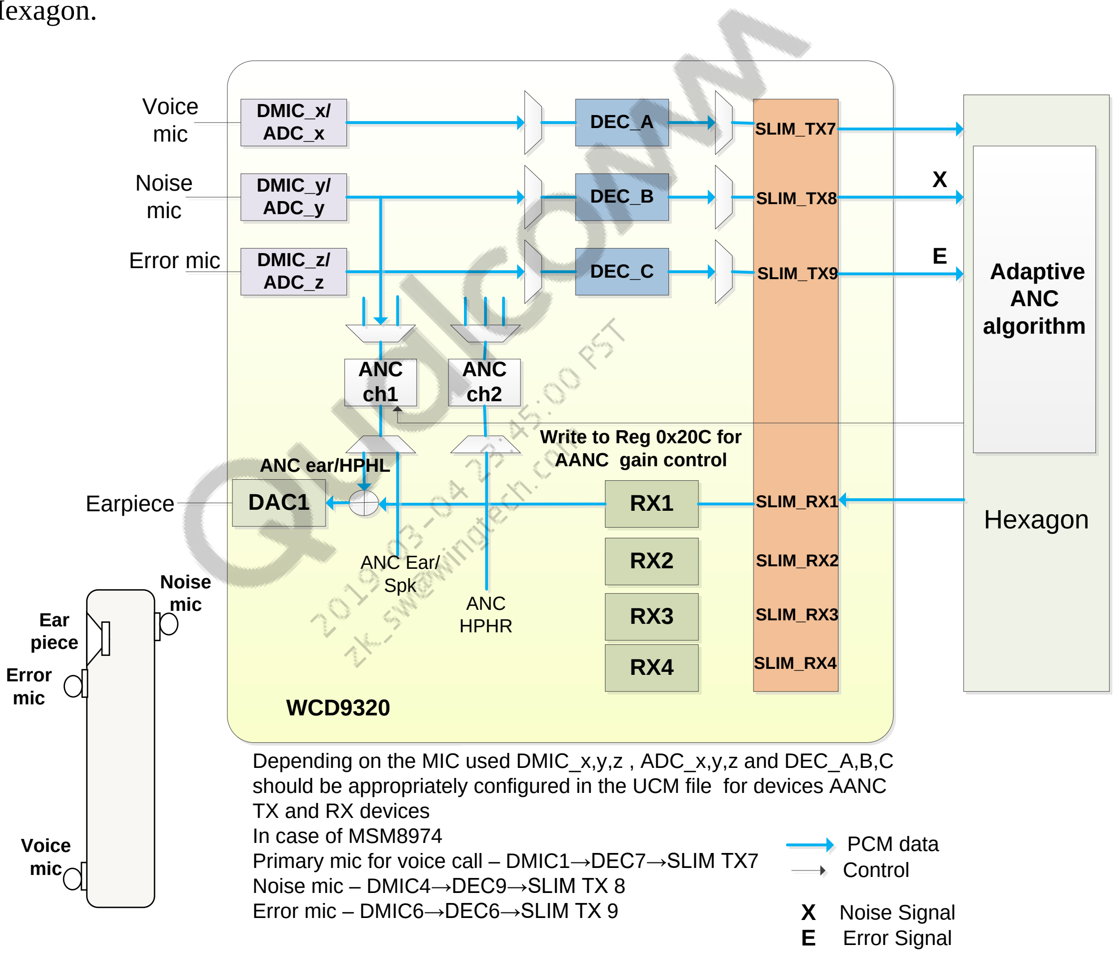
Table 21-7 shows common MBHC issues that were encountered during previous integration activities and some troubleshooting steps to resolve them.

Table 21-7 Debugging and troubleshooting tips

Issue type	Symptoms	Troubleshooting tips
Plug type detection	▪ Nothing is reported when the headset or headphone is inserted ▪ Headset is detected as headphone	▪ Ensure that the MBHC schematic is reviewed by the QTI hardware CE team by raising a case at https://support.cdmatech.com. ▪ Ensure that the software change is done depending on NO or NC configuration. ▪ Use MIC2 for the headset MIC_BIAS2 for mic bias. Ensure that the voltage is configured as 2.7 V in the kernel/arch/arm/boot/dts/ msm8226.dtsi device tree file. ▪ Collect MBHC-related logs. Check if there is any obvious issue by interpreting the logs; for example, is insert/remove IRQ getting triggered? ▪ Check if the HS_DET pin changes level with insertion/removal. ▪ If the issue is not resolved, collect the MBHC-related logs and raise a case at https://support.cdmatech.com.
Button detection	▪ No button press is detected ▪ Wrong button detected ▪ Double button press is not detected	▪ If a wrong button is detected, fix it by tuning the button low and high thresholds in the ACDB file. ▪ If calibration is done from MBCH calibration in the machine driver, appropriate changes are done in the machine driver file. ▪ If no button press is detected, the calibration data could be wrong. Collect the MBHC-related logs and raise a case in https://support.cdmatech.com. ▪ Double button press is like two single button presses. Collect the MBHC-related logs and check in the kernel if two button presses are reported to the user space. Check for messages such as fake button press or bogus button. In such a case, check if any latest fix is available or raise a case in https://support.cdmatech.com.
Noise	▪ Noise on the headset when playback is started ▪ Pop noise when a three-pole plug type is removed within 5 sec of insertion ▪ Pop noise when a four-pole plug type is removed during recording (100%) ▪ Pop noise when a four-pole plug type is removed during playback (rare, 1 in 20 times)	▪ Ensure that the polling with VDDIO source during playback when headset is connected. ▪ Pop noise during insertion and removal cannot be avoided. However, with the new way of detecting the plug type with a 20 µA current source method has reduced the level of POP significantly. ▪ The plots for HS_DET, mic voltage, and mic bias voltage is provided to check when the HS_DET pin level changes and when the mic bias is disabled.

22 Adaptive ANC (AANC)

Figure 22-1 shows the data path and AANC gain control between the WCD9320 codec and Hexagon.



Depending on the MIC used DMIC_x,y,z , ADC_x,y,z and DEC_A,B,C should be appropriately configured in the UCM file for devices AANC TX and RX devices
In case of MSM8974
Primary mic for voice call – DMIC1→DEC7→SLIM TX7
Noise mic – DMIC4→DEC9→SLIM TX 8
Error mic – DMIC6→DEC6→SLIM TX 9

Figure 22-1 Data path and AANC gain control between WCD9320 codec and Hexagon

In conventional Feed Forward (FF) ANC mode, ANC does a fixed filtering on an input signal to generate antinoise. This approach works well for headsets whose acoustic property remains constant over time. For a handset earpiece application, the acoustic property and a holding pattern of a user may change over time. Therefore, a fixed filtering approach does not produce consistent noise reduction.

In AANC, there is an adaptive algorithm in Hexagon that receives signals from two mics (noise signal X and error signal E), and calculates optimal filter depending on holding pattern and pressing pressure of the user. Conventional FF ANC requires only input signal X, but the adaptive algorithm needs a new error signal E. This signal is with an extra mic placed near the transducer. This extra mic monitors the error signal at the quiet zone between the output speaker and the ear canal. Hexagon updates every 160 samples (20 ms) gain of the CDC_ANCn_GAIN_CTL register in the WCD9xxx codec to minimize the error signal effect canceling the surrounding noise during a voice call in Handset mode.

22.1 Call flow

Figure 22-2 shows the call flow involved in initializing normal audio playback.

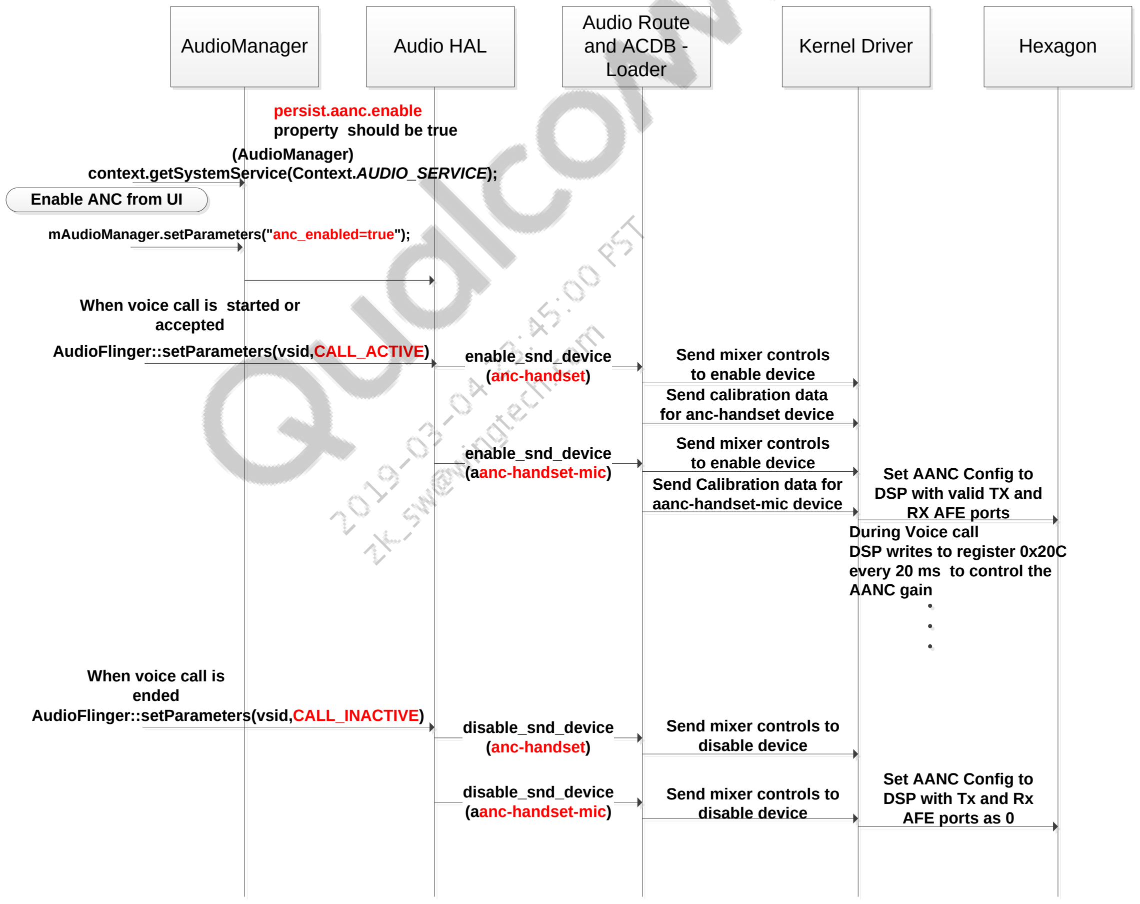

AudioManager
Audio HAL
Audio Route and ACDB - Loader
Kernel Driver
Hexagon
persist.aanc.enable property should be true
(AudioManager) context.getSystemService(Context.AUDIO_SERVICE);
Enable ANC from UI
mAudioManager.setParameters("anc_enabled=true");
When voice call is started or accepted
AudioFlinger::setParameters(vsid,CALL_ACTIVE)
enable_snd_device (anc-handset)
Send mixer controls to enable device
Send calibration data for anc-handset device
enable_snd_device (aanc-handset-mic)
Send mixer controls to enable device
Send Calibration data for aanc-handset-mic device
Set AANC Config to DSP with valid TX and RX AFE ports
During Voice call DSP writes to register 0x20C every 20 ms to control the AANC gain
When voice call is ended
AudioFlinger::setParameters(vsid,CALL_INACTIVE)
disable_snd_device (anc-handset)
Send mixer controls to disable device
disable_snd_device (aanc-handset-mic)
Send mixer controls to disable device
Set AANC Config to DSP with Tx and Rx AFE ports as 0

Figure 22-2 Call setup and teardown call flow

22.2 Log collection

22.2.1 User space logs

This command clears the current logcat logs and starts logging the user space logs to the logcat.txt file and the stdout simultaneously.

```
adb logcat –c  && adb logcat –v threadtime  | tee logcat.txt
```

NOTE: For the tee command to work, Cygwin must be installed or the command must be executed in a Linux environment.

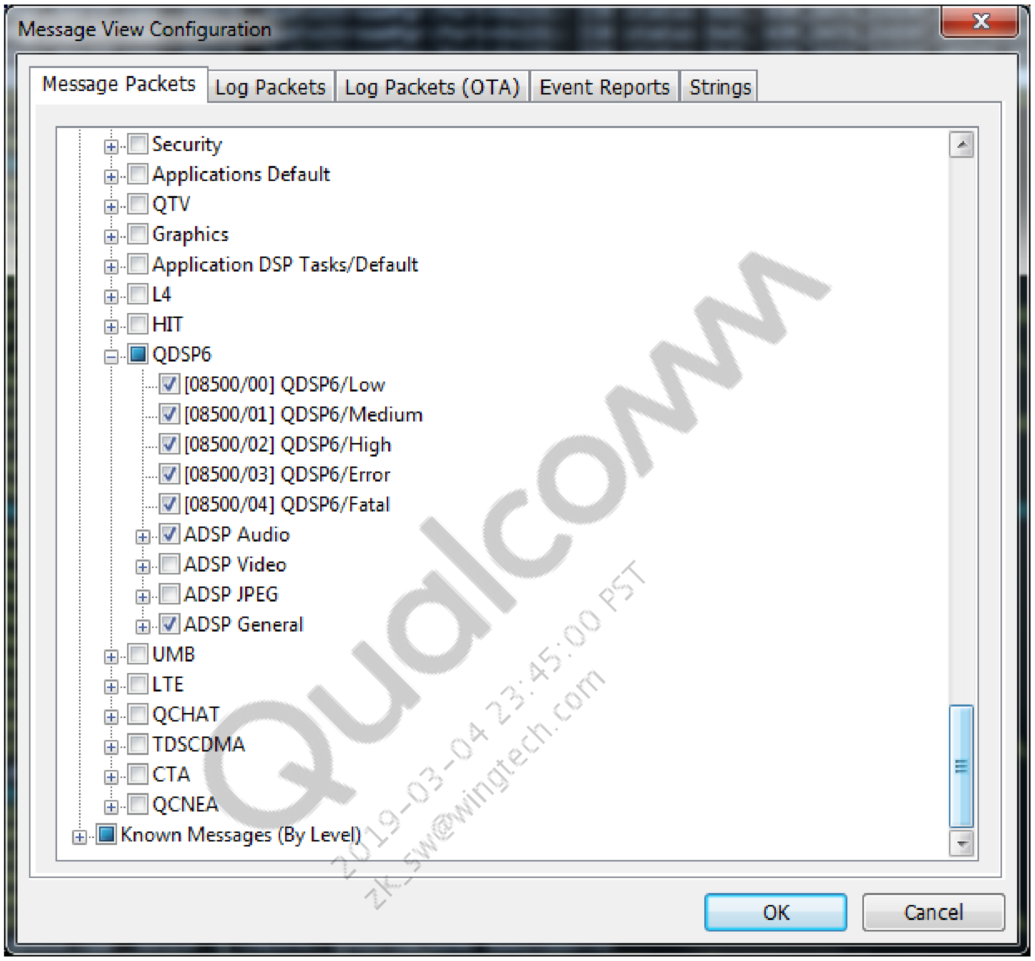
22.2.2 Kernel logs

```
adb shell mount -t debugfs debugfs /sys/kernel/debug
adb shell
echo -n "file q6afe.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm-dai-q6.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-core.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9xxx-slimslave.c +p" >
/sys/kernel/debug/dynamic_debug/control
echo -n "file soc-dapm.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file msm8974.c +p" > /sys/kernel/debug/dynamic_debug/control
echo -n "file wcd9320.c +p" > /sys/kernel/debug/dynamic_debug/control
```

22.2.3 QXDM Professional logs

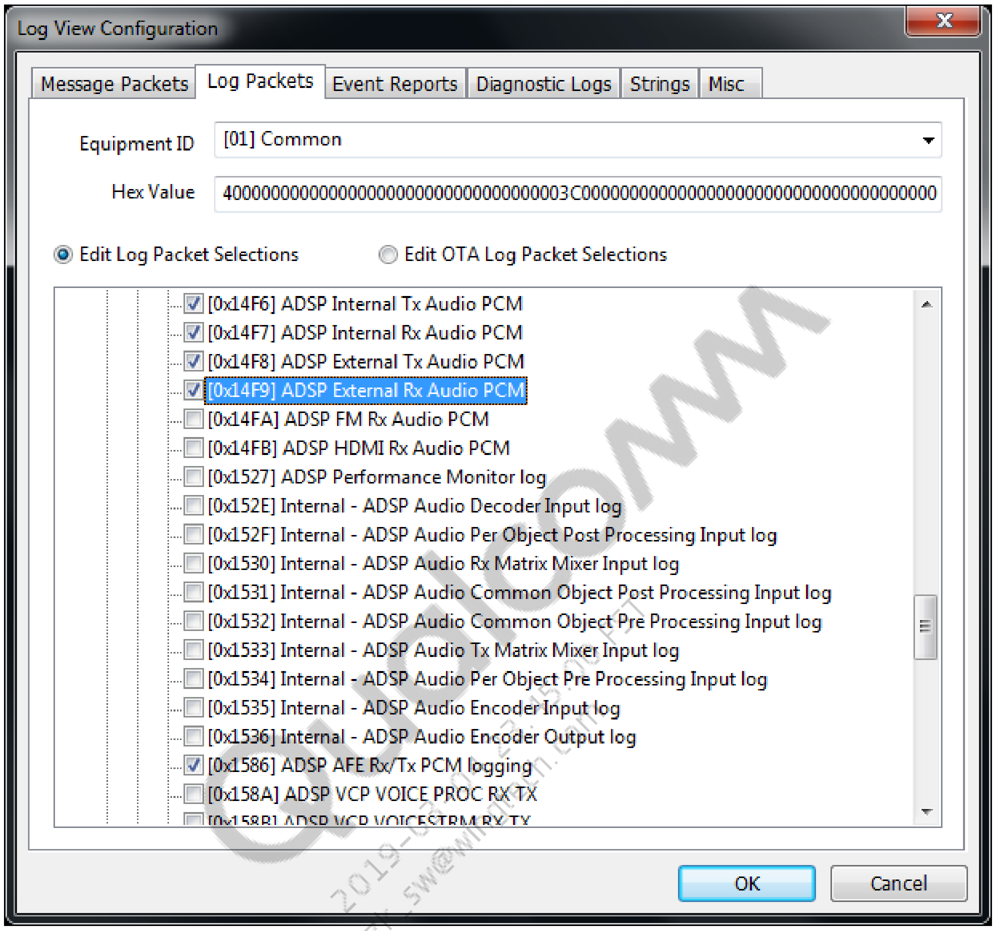
For QXDM Professional logs, PCM logging must be enabled and the Hexagon logs are required.

To enable Hexagon DSP logs on QXDM Pro, click **Options** > **Log View configuration**. In the Message Packets tab, select Hexagon DSP messages, as shown.



- [ ] Security
- [ ] Applications Default
- [ ] QTV
- [ ] Graphics
- [ ] Application DSP Tasks/Default
- [ ] L4
- [ ] HIT
- ■ QDSP6
  - [✓] [08500/00] QDSP6/Low
  - [✓] [08500/01] QDSP6/Medium
  - [✓] [08500/02] QDSP6/High
  - [✓] [08500/03] QDSP6/Error
  - [✓] [08500/04] QDSP6/Fatal
  - [✓] ADSP Audio
  - [ ] ADSP Video
  - [ ] ADSP JPEG
  - [✓] ADSP General
- [ ] UMB
- [ ] LTE
- [ ] QCHAT
- [ ] TDSCDMA
- [ ] CTA
- [ ] QCNEA
- ■ Known Messages (By Level)

Similarly, PCM logging is enabled in the Log Packets tab.



Equipment ID: [01] Common

Hex Value: 40000000000000000000000000000000003C0000000000000000000000000000000000

[● Edit Log Packet Selections] ○ Edit OTA Log Packet Selections

- ✓ [0x14F6] ADSP Internal Tx Audio PCM
- ✓ [0x14F7] ADSP Internal Rx Audio PCM
- ✓ [0x14F8] ADSP External Tx Audio PCM
- ✓ [0x14F9] ADSP External Rx Audio PCM
- ☐ [0x14FA] ADSP FM Rx Audio PCM
- ☐ [0x14FB] ADSP HDMI Rx Audio PCM
- ☐ [0x1527] ADSP Performance Monitor log
- ☐ [0x152E] Internal - ADSP Audio Decoder Input log
- ☐ [0x152F] Internal - ADSP Audio Per Object Post Processing Input log
- ☐ [0x1530] Internal - ADSP Audio Rx Matrix Mixer Input log
- ☐ [0x1531] Internal - ADSP Audio Common Object Post Processing Input log
- ☐ [0x1532] Internal - ADSP Audio Common Object Pre Processing Input log
- ☐ [0x1533] Internal - ADSP Audio Tx Matrix Mixer Input log
- ☐ [0x1534] Internal - ADSP Audio Per Object Pre Processing Input log
- ☐ [0x1535] Internal - ADSP Audio Encoder Input log
- ☐ [0x1536] Internal - ADSP Audio Encoder Output log
- ✓ [0x1586] ADSP AFE Rx/Tx PCM logging
- ☐ [0x158A] ADSP VCP VOICE PROC RX TX
- ☐ [0x158B] ADSP VCP VOICESTRM RX TX

OEMs must collect the logcat, kernel, and QXDM Professional logs and share them with QTI for initial analysis of the AANC issue.

NOTE: The Customer Engineering team is responsible for checking the control flow for AANC and UCM-related customizations. Hardware acoustic tuning and tuning of the AANC algorithm for optimal performance on the OEM device is done by the Tuning team in San Diego for AANC.

22.3 Log analysis

22.3.1 User space logs

To enable AANC, anc_enabled is set by the application and the system property persist.aanc.enable is set to true. The following logs are displayed when a voice call is established.

When the voice call is started, as part of the voice call state update, the value of AANC set previously is set again in HAL.

```
12-13 15:30:02.117   214  1122 V AudioFlinger: setParameters(): io 0,
keyvalue vsid=281026560;call_state=1, calling pid 210
12-13 15:30:02.117   214  1122 D audio_hw_primary: adev_set_parameters:
enter: vsid=281026560;call_state=1
12-13 15:30:02.117   214  1122 V voice   : voice_set_parameters: enter:
vsid=281026560;call_state=1
12-13 15:30:02.117   214  1122 D audio_hw_extn:
audio_extn_set_anc_parameters: anc_enabled:1
```

When the voice call moves to the Active state, as part of the voice call state update, the voice module in HAL enables the input/ouput devices, pushes calibration to them, and sets up routing.

```
12-13 15:30:02.117   214   214 V AudioFlinger: setParameters(): io 0,
keyvalue vsid=281022464;call_state=2, calling pid 210
12-13 15:30:02.117   214   214 V voice   : voice_set_parameters: enter:
vsid=281022464;call_state=2
12-13 15:30:02.117   214   214 D voice_extn: update_call_states
is_in_call:0 mode:2
12-13 15:30:02.117   214   214 D voice_extn: update_calls: cur_state=1
new_state=2 vsid=10c01000
12-13 15:30:02.117   214   214 D voice_extn: update_calls: INACTIVE ->
ACTIVE vsid:10c01000
```

When the voice call moves to the CALL_ACTIVE state (defined in hardware/qcom/audio/hal/voice.h), as part of the voice call state update, the voice module in HAL enables the input/ouput devices, pushes calibration to them, and sets up routing.

The input and output devices returned are ANC-enabled versions, anc-handset, and aanc-handset-mic, defined in /hardware/qcom/audio/hal/msm8974/platform.c. Use case (10) is USECASE_VOICE_CALL as defined in hardware/qcom/audio/hal/audio_hw.h.

```
12-13 15:30:02.117   214   214 D voice   : start_call: enter usecase:voice-
call
12-13 15:30:02.117   214   214 V msm8974_platform:
platform_get_output_snd_device: enter: output devices(0x1)
12-13 15:30:02.117   214   214 D audio_hw_extn:
audio_extn_should_use_handset_anc: AANC enabled in the property
12-13 15:30:02.117   214   214 V msm8974_platform:
platform_get_output_snd_device: exit: snd_device(anc-handset)
12-13 15:30:02.117   214   214 V msm8974_platform:
platform_get_input_snd_device: enter: out_device(0x1) in_device(0)
12-13 15:30:02.117   214   214 D audio_hw_extn:
audio_extn_should_use_handset_anc: AANC enabled in the property
12-13 15:30:02.117   214   214 V msm8974_platform:
platform_get_input_snd_device: exit: in_snd_device(aanc-handset-mic)
12-13 15:30:02.117   214   214 D audio_hw_primary: select_devices:
out_snd_device(25: anc-handset) in_snd_device(63: aanc-handset-mic)
```

The anc-handset device has the ACDB ID 103. It is enabled and its calibration is pushed.

```
12-13 15:30:02.117   214   214 D hardware_info: hw_info_append_hw_type :
device_name = anc-handset
12-13 15:30:02.117   214   214 V audio_hw_primary: enable_snd_device:
snd_device(25: anc-handset)
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 103, path =  0
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_adm_topology
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_audtable
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_audvoltable
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
```

The aanc-handset-mic device has the ACDB ID 104. It is enabled and its calibration is pushed.

```
12-13 15:30:02.117   214   214 D hardware_info: hw_info_append_hw_type :
device_name = aanc-handset-mic
12-13 15:30:02.117   214   214 V audio_hw_primary: enable_snd_device:
snd_device(63: aanc-handset-mic)
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_audio_cal,
acdb_id = 104, path =  1
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_adm_topology
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TOPOLOGY_ID
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_audtable
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_COMMON_TABLE
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> AUDIO_SET_AUDPROC_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_audvoltable
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_AUDPROC_GAIN_DEP_STEP_TABLE
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
AUDIO_SET_AUDPROC_VOL_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
```

Other calibration for the voice call, including AANC-related items, is then pushed.

```
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_voice_cal,
acdb_rx = 103, acdb_tx = 104, feature_set = 0
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
```

```
send_voice_rx_topology
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOCPROC_COMMON_TOPOLOGY_ID
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
send_voice_tx_topology
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOCPROC_COMMON_TOPOLOGY_ID
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> ACDB_CMD_GET_AFE_DATA
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
AUDIO_SET_SIDETONE_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_voice_columns,
table 1
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_COLUMNS_INFO
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_PROC_COMMON_TABLE
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> AUDIO_SET_VOCPROC_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_PROC_DEVICE_CFG
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
AUDIO_SET_VOCPROC_DEV_CFG_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_voice_columns,
table 2
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_COLUMNS_INFO
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_PROC_GAIN_DEP_VOLTBL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
AUDIO_SET_VOCPROC_VOL_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_voice_columns,
table 3
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_COLUMNS_INFO
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_VOC_STREAM_COMMON_TABLE
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
AUDIO_SET_VOCPROC_STREAM_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> AUDIO_SET_AFE_CAL
12-13 15:30:02.117   214   214 D ACDB-LOADER: Valid AANC Device for device
104 is 1
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> send_aanctable
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB ->
ACDB_CMD_GET_ADAPTIVE_ANC_CONFIG_TABLE
12-13 15:30:02.117   214   214 D ACDB-LOADER:
ACDB_CMD_GET_ADAPTIVE_ANC_CONFIG_TABLE result 0
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> AUDIO_SET_AANC_TABLE
12-13 15:30:02.117   214   214 D ACDB-LOADER: ACDB -> Sent VocProc Cal!

12-13 15:30:02.297   214   214 V audio_hw_primary:
enable_all_usecases_of_type: usecase id 10
```

```
12-13 15:30:02.297   214   214 V audio_hw_primary: enable_audio_route:
enter: usecase(10)
12-13 15:30:02.297   214   214 V audio_hw_primary: enable_audio_route:
apply mixer path: voice-call
12-13 15:30:02.297   214   214 V audio_hw_primary: enable_audio_route: exit
```

card_id (0) SOUND_CARD and device_id (0) is AUDIO_RECORD_PCM_DEVICE as defined in hardware/qcom/audio/hal/msm8974/platform.h.

```
12-13 15:30:02.297   214   214 V voice   : start_call: Opening PCM playback
device card_id(0) device_id(2)
.
.
.
```

The call continues until it is ended, at which point HAL is again informed of the voice call state transition. The device is then disabled and routing is reset.

```
12-13 15:30:14.367   214  1122 V AudioFlinger: setParameters(): io 0,
keyvalue vsid=281022464;call_state=1, calling pid 210
12-13 15:30:14.367   214  1122 V voice   : voice_set_parameters: enter:
vsid=281022464;call_state=1
12-13 15:30:14.367   214  1122 D voice_extn: update_call_states
is_in_call:1 mode:2
12-13 15:30:14.367   214  1122 D voice_extn: update_calls: enter:
12-13 15:30:14.367   214  1122 D voice_extn: update_calls: cur_state=2
new_state=1 vsid=10c01000
12-13 15:30:14.367   214  1122 D voice_extn: update_calls:
ACTIVE/HOLD/LOCAL_HOLD -> INACTIVE vsid:10c01000

12-13 15:30:14.367   214  1122 D voice   : stop_call: enter usecase:voice-
call
12-13 15:30:14.427   214  1122 V audio_hw_primary: disable_audio_route:
enter: usecase(10)
12-13 15:30:14.427   214  1122 V audio_hw_primary: disable_audio_route:
reset mixer path: voice-call
12-13 15:30:14.427   214  1122 V audio_hw_primary: disable_audio_route:
exit

12-13 15:30:14.427   214  1122 D hardware_info: hw_info_append_hw_type :
device_name = anc-handset
12-13 15:30:14.427   214  1122 V audio_hw_primary: disable_snd_device:
snd_device(25: anc-handset)

12-13 15:30:14.427   214  1122 D hardware_info: hw_info_append_hw_type :
device_name = aanc-handset-mic
12-13 15:30:14.427   214  1122 V audio_hw_primary: disable_snd_device:
snd_device(63: aanc-handset-mic)
```

22.3.2 Kernel logs

```
<7>[ 3705.643220] taiko_codec taiko_codec: taiko_put_anc_func: anc_func 1

<7>[ 3705.644702] aanc active is 1 rx port is 16384, tx port is 16385
<7>[ 3705.645109] taiko-slim taiko-slim-pgd: Write 07 to 0x397
<7>[ 3705.645997] taiko-slim taiko-slim-pgd: Write 4d to 0x3ab
<7>[ 3705.646716] taiko-slim taiko-slim-pgd: Write 50 to 0x3aa
<7>[ 3705.647433] taiko-slim taiko-slim-pgd: Write 4e to 0x3a9
```

SLIMbus mixer parameters are set up for voice, noise, and error mics on SLIM TX7, SLIM TX8, and SLIM TX9 as defined in the device definition of aanc-handset-mic in /device/qcom/msm8974/mixer_paths.xml.

```
<7>[ 3705.647789] slim_tx_mixer_put: wname AIF1_CAP Mixer cname AIF1_CAP
Mixer SLIM TX7 value 0 shift 1 item 1
<7>[ 3705.647797] wcd9xxx_tx_vport_validation: vtable 0x28 port_id 6 size
32
<7>[ 3705.647805] slim_tx_mixer_put: name AIF1_CAP Mixer sname (null)
updated value 64 shift 1
<7>[ 3705.648554] slim_tx_mixer_put: wname AIF1_CAP Mixer cname AIF1_CAP
Mixer SLIM TX8 value 64 shift 1 item 1
<7>[ 3705.648562] wcd9xxx_tx_vport_validation: vtable 0x28 port_id 7 size
32
<7>[ 3705.648568] slim_tx_mixer_put: name AIF1_CAP Mixer sname (null)
updated value 192 shift 1
<7>[ 3705.649459] slim_tx_mixer_put: wname AIF1_CAP Mixer cname AIF1_CAP
Mixer SLIM TX9 value 192 shift 1 item 1
<7>[ 3705.649467] wcd9xxx_tx_vport_validation: vtable 0x28 port_id 8 size
32
```

DAPM widgets are enabled when an AANC device is enabled.

```
<7>[ 3705.659025] taiko_codec taiko_codec: dapm: power up widget
CLASS_H_DSM MUX
<7>[ 3705.659081] taiko_codec taiko_codec: dapm: power up widget DAC1
<7>[ 3705.659136] taiko_codec taiko_codec: dapm: power up widget RX1 CHAIN
<7>[ 3705.659784] taiko_codec taiko_codec: dapm: power up widget RX_BIAS
<7>[ 3705.659859] taiko_codec taiko_codec: dapm: power up widget
EAR_PA_MIXER
<7>[ 3705.659914] taiko_codec taiko_codec: dapm: power up widget RX1 MIX2
<7>[ 3705.660016] taiko_codec taiko_codec: dapm: power up widget MCLK
<7>[ 3705.660070] taiko_codec taiko_codec: dapm: power up widget ANC EAR PA
<7>[ 3705.660312] taiko_codec taiko_codec: dapm: power up widget ANC1 MUX
<7>[ 3705.660418] taiko_codec taiko_codec: dapm: power up widget ANC EAR
<7>[ 3705.660746] taiko_codec taiko_codec: dapm: power up widget DMIC4
<7>[ 3705.660757] taiko_codec taiko_codec: dapm: power up widget MIC BIAS3
External
```

```
<7>[ 3705.660857] taiko_codec taiko_codec: dapm: power up widget Digital
Mic4
<7>[ 3705.660911] taiko_codec taiko_codec: dapm: power up widget Digital
Mic3
<7>[ 3705.660920] taiko_codec taiko_codec: dapm: power up widget LDO_H
```

Start the AFE port SLIMBUS_0_RX (0x4000). The AFE ports are defined in the file kernel/include/sound/apr_audio-v2.h.

```
<7>[ 3705.826856] taiko_startup(): substream = subdevice #0  stream = 1
<7>[ 3705.826862] msm8974_snd_startup(): substream = subdevice #0  stream = 1
<7>[ 3705.826924] msm_slim_0_tx_be_hw_params_fixup()
<7>[ 3705.826931] msm_snd_hw_params: taiko_tx1_tx_dai_id_1_ch=0
<7>[ 3705.826936] taiko_get_channel_map: slot_num 0 ch->ch_num 134
<7>[ 3705.826942] taiko_get_channel_map: slot_num 1 ch->ch_num 135
<7>[ 3705.826947] taiko_get_channel_map: slot_num 2 ch->ch_num 136
<7>[ 3705.826952] taiko_get_channel_map: tx_num 3
<7>[ 3705.826957] msm_snd_hw_params:
msm_slim_0_tx_ch(3)user_set_tx_ch(3)tx_ch_cnt(3)
<7>[ 3705.826964] taiko_hw_params: dai_name = taiko_tx1 DAI-ID 1 rate 48000
num_ch 3
<7>[ 3705.826971] taiko_set_decimator_rate: dai->id = 1, tx_port = 7
<7>[ 3705.826979] taiko_set_decimator_rate: set DEC7 (-> SLIM_TX7) rate to
48000
<7>[ 3705.826986] taiko_set_decimator_rate: dai->id = 1, tx_port = 8
<7>[ 3705.826993] taiko_set_decimator_rate: set DEC9 (-> SLIM_TX8) rate to
48000
<7>[ 3705.827000] taiko_set_decimator_rate: dai->id = 1, tx_port = 9
<7>[ 3705.827007] taiko_set_decimator_rate: set DEC6 (-> SLIM_TX9) rate to
48000
<7>[ 3705.827502] afe_port_start: port id: 0x4000
```

afe_q6_interface_prepare is called for port 0x4000. afe_callback is received with result opcode APR_BASIC_RSP_RESULT having status ADSP_EOK for command AFE_PORT_CMD_SET_PARAM_V2.

```
<7>[ 3705.827509] afe_q6_interface_prepare:
<7>[ 3705.827515] afe_send_cal
<7>[ 3705.827522] afe_send_cal_block: path 0
<7>[ 3705.827528] afe_send_cal_block: No AFE cal to send!
<7>[ 3705.827532] afe_send_hw_delay
<7>[ 3705.827539] afe_send_hw_delay: Failed to get hw delay info
<7>[ 3705.827546] afe_send_hw_delay port_id 16384 rate 48000 delay_usec
572662306 status -22
<7>[ 3705.827552] afe_port_start: port_id 0x4000, mad_type 0
<7>[ 3705.827950] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
size = 8
<7>[ 3705.827957] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
```

```
token=15
```

AFE_PORT_CMD_DEVICE_START (0x100e5) command is sent to AFE port SLIMBUS_0_ RX (0x4000). afe_callback is received with result opcode APR_BASIC_RSP_RESULT having status ADSP_EOK for command AFE_PORT_CMD_DEVICE_START. APR_BASIC_RSP_ RESULT is defined in kernel/arch/arm/mach-msm/include/mach/qdsp6v2/apr.h.

```
<7>[ 3705.827968] afe_callback:port_id = 0
<7>[ 3705.827993] afe_apr_send_pkt: leave 0
<7>[ 3705.827999] afe_send_cmd_port_start: enter
<7>[ 3705.828005] afe_send_cmd_port_start: cmd device start opcode[0x100e5]
port id[0x4000]
<7>[ 3705.836269] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0
size = 8
<7>[ 3705.836278] afe_callback:opcode = 0x110e8 cmd = 0x100e5 status = 0x0
token=15

<7>[ 3705.836290] afe_callback:port_id = 0
<7>[ 3705.836310] afe_apr_send_pkt: leave 0
<7>[ 3705.836316] task_name = mediaserver pid = 214
<7>[ 3705.836321] afe_send_cmd_port_start: leave 0
<7>[ 3705.836497] taiko_codec taiko_codec: dapm: power up widget AIF1 PB
<7>[ 3705.836541] taiko_codec taiko_codec: dapm: power up widget SLIM RX1
MUX
<7>[ 3705.836583] taiko_codec taiko_codec: dapm: power up widget SLIM RX1
<7>[ 3705.836626] taiko_codec taiko_codec: dapm: power up widget RX1 MIX1
INP1
<7>[ 3705.836669] taiko_codec taiko_codec: dapm: power up widget RX1 MIX1
<7>[ 3705.836785] taiko_codec taiko_codec: dapm: power up widget COMP1_CLK
<7>[ 3705.837261] taiko_config_compander: COMP1_CLK event 1 compander 1,
enabled 0
<7>[ 3705.837270] taiko_codec_enable_slimrx: event called! codec name
taiko_codec num_dai 8
<7>[ 3705.837272] stream name AIF1 Playback event 2
<7>[ 3705.837279] taiko_codec_enable_slimrx: w->name AIF1 PB w->shift 0
event 2
<7>[ 3705.837285] wcd9xxx_get_slave_port: ch_num[144] slave port[16]
<7>[ 3705.837291] list ch->ch_h 16 ch->sph -889126896
<7>[ 3705.837297] wcd9xxx_cfg_slim_sch_rx: ch_cnt[1] rate=48000
WATER_MARK_VAL 5
<7>[ 3705.837303] Before slim_define_ch:
<7>[ 3705.837304] ch_cnt 1,ch_h[0] 16 ch_h[1] 0, grph 272
<7>[ 3705.837314] wcd9xxx_cfg_slim_sch_rx: codec_port 16 rx 0xf3588800,
payload 1
<7>[ 3705.837317] sh_ch.rx_port_ch_reg_base0 0x140
<7>[ 3705.837318] sh_ch.port_rx_cfg_reg_base 0x30
<7>[ 3705.837326] taiko-slim taiko-slim-pgd: Write 01 to 0x180
<7>[ 3705.837500] taiko-slim taiko-slim-pgd: Write 05 to 0x40
```

Enable circuit-switch voice Rx path. Stream 0 is SNDRV_PCM_STREAM_PLAYBACK and event 1 is SND_SOC_DAPM_STREAM_START, which are defined in kernel/include/sound/asound.h.

```
<7>[ 3705.844427]  Circuit-Switch Voice: rtd stream 0 event 1
<7>[ 3705.844468] msm-pcm-routing msm-pcm-routing: dapm: power up widget
CS-VOICE_DL1
<7>[ 3705.844511] msm-pcm-routing msm-pcm-routing: dapm: power up widget
SLIM_0_RX_Voice Mixer
<7>[ 3705.844555] msm-pcm-routing msm-pcm-routing: dapm: power up widget
SLIMBUS_0_RX
<7>[ 3705.844752] msm-pcm-routing msm-pcm-routing: dapm: power up widget
BE_OUT
<7>[ 3705.845859]  SLIMBUS_0_RX: rtd stream 0 event 1
```

Start the AFE port SLIMBUS_0_TX(0x4001). The AFE ports are defined in the file kernel/include/sound/apr_audio-v2.h.

```
<7>[ 3705.845888] afe_port_start: port id: 0x4001

<7>[ 3705.845893] afe_q6_interface_prepare:
<7>[ 3705.845899] afe_send_cal
<7>[ 3705.845906] afe_send_cal_block: path 1
<7>[ 3705.845911] afe_send_cal_block: No AFE cal to send!
<7>[ 3705.845916] afe_send_hw_delay
<7>[ 3705.845923] afe_send_hw_delay: Failed to get hw delay info
<7>[ 3705.845929] afe_send_hw_delay port_id 16385 rate 48000 delay_usec
572662306 status -22
```

Send the AANC port config command for SLIMBUS_0_TX(16385) and SLIMBUS_0_RX(16384) ports.

```
<7>[ 3705.845936] afe_port_start: port_id 0x4001, mad_type 0
<7>[ 3705.845942] afe_aanc_start Tx port is 16385, Rx port is 16384
<7>[ 3705.845947] afe_aanc_port_cfg: tx_port 16385, rx_port 16384
<7>[ 3705.846436] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
size = 8
<7>[ 3705.846443] afe_callback:opcode = 0x110e8 cmd = 0x100ef status = 0x0
token=16
```

Enable DAPM widgets for the aanc-handset-mic device.

```
<7>[ 3705.860540] taiko_codec taiko_codec: dapm: power up widget AIF1 CAP
<7>[ 3705.860835] taiko_codec taiko_codec: dapm: power up widget AIF1_CAP
Mixer
<7>[ 3705.860891] taiko_codec taiko_codec: dapm: power up widget SLIM TX9
MUX
<7>[ 3705.860946] taiko_codec taiko_codec: dapm: power up widget SLIM TX8
MUX
<7>[ 3705.861002] taiko_codec taiko_codec: dapm: power up widget SLIM TX7
MUX
<7>[ 3705.861058] taiko_codec taiko_codec: dapm: power up widget DEC6 MUX
<7>[ 3705.861114] taiko_codec taiko_codec: dapm: power up widget DEC9 MUX
<7>[ 3705.861171] taiko_codec taiko_codec: dapm: power up widget DEC7 MUX
<7>[ 3705.861227] taiko_codec taiko_codec: dapm: power up widget CDC_CONN
<7>[ 3705.861283] taiko_codec taiko_codec: dapm: power up widget DMIC6
<7>[ 3705.861338] taiko_codec taiko_codec: dapm: power up widget DMIC1
<7>[ 3705.861440] taiko_codec taiko_codec: dapm: power up widget MIC BIAS4
External
<7>[ 3705.861496] taiko_codec taiko_codec: dapm: power up widget MIC BIAS1
External
<7>[ 3705.861551] taiko_codec taiko_codec: dapm: power up widget Digital
Mic6
<7>[ 3705.861606] taiko_codec taiko_codec: dapm: power up widget Digital
Mic5
<7>[ 3705.861706] taiko_codec taiko_codec: dapm: power up widget Digital
Mic2
<7>[ 3705.861761] taiko_codec taiko_codec: dapm: power up widget Digital
Mic1
```

Enable circuit-switch voice Tx path. Stream 0 is SNDRV_PCM_STREAM_CAPTURE and event 1 is SND_SOC_DAPM_STREAM_START, which are defined in kernel/include/sound/asound.h.

```
<7>[ 3705.944168]  Circuit-Switch Voice: rtd stream 1 event 1
<7>[ 3705.944263] msm-pcm-routing msm-pcm-routing: dapm: power up widget
CS-VOICE_UL1
<7>[ 3705.944306] msm-pcm-routing msm-pcm-routing: dapm: power up widget
Voice_Tx Mixer
<7>[ 3705.944350] msm-pcm-routing msm-pcm-routing: dapm: power up widget
SLIMBUS_0_TX
<7>[ 3705.944399] msm-pcm-routing msm-pcm-routing: dapm: power up widget
BE_IN
<7>[ 3705.945521]  SLIMBUS_0_TX: rtd stream 1 event 1
```

Enable DAPM widgets for the anc-handset device.

```
<7>[ 3717.903068] taiko_codec taiko_codec: dapm: power down widget AIF1 PB
<7>[ 3717.903111] taiko_codec taiko_codec: dapm: power down widget SLIM RX1
MUX
```

```
<7>[ 3717.903153] taiko_codec taiko_codec: dapm: power down widget SLIM RX1
<7>[ 3717.903195] taiko_codec taiko_codec: dapm: power down widget RX1 MIX1
INP1
<7>[ 3717.903238] taiko_codec taiko_codec: dapm: power down widget RX1 MIX1
<7>[ 3717.903299] taiko_codec taiko_codec: dapm: power down widget
COMP1_CLK
```

Disable circuit-switch voice Rx path. Stream 0 is SNDRV_PCM_STREAM_PLAYBACK and event 1 is SND_SOC_DAPM_STREAM_STOP, which are defined in kernel/include/sound/asound.h.

```
<7>[ 3717.906162]  Circuit-Switch Voice: rtd stream 0 event 2
<7>[ 3717.906214] msm-pcm-routing msm-pcm-routing: dapm: power down widget
CS-VOICE_DL1
<7>[ 3717.906256] msm-pcm-routing msm-pcm-routing: dapm: power down widget
SLIM_0_RX_Voice Mixer
<7>[ 3717.906300] msm-pcm-routing msm-pcm-routing: dapm: power down widget
SLIMBUS_0_RX
<7>[ 3717.906354] msm-pcm-routing msm-pcm-routing: dapm: power down widget
BE_OUT
<7>[ 3717.906798]  SLIMBUS_0_RX: rtd stream 0 event 2
```

Close AFE port SLIMBUS_0_TX (0x4001). afe_callback is received with result opcode APR_BASIC_RSP_RESULT having status ADSP_EOK for command AFE_PORT_CMD_DEVICE_STOP(0x100e6). APR_BASIC_RSP_RESULT is defined in kernel/arch/arm/mach-msm/include/mach/qdsp6v2/apr.h.

```
7>[ 3717.939946] afe_close: port_id=16385
<7>[ 3717.939952] afe_close: port_id 0x4001, mad_type 0
<7>[ 3717.939956] afe_close: Not a MAD port
<7>[ 3717.939962] afe_aanc_mod_enable: tx_port 16385
<7>[ 3717.939967] afe_q6_interface_prepare:
<7>[ 3717.945603] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
size = 8
<7>[ 3717.945609] afe_callback:opcode = 0x110e8 cmd = 0x100e6 status = 0x0
token=16
```

Disable the DAPM widgets for the aanc-handset-mic device.

```
<7>[ 3717.945734] taiko_codec taiko_codec: dapm: power down widget AIF1 CAP
<7>[ 3717.945776] taiko_codec taiko_codec: dapm: power down widget AIF1_CAP
Mixer
<7>[ 3717.945818] taiko_codec taiko_codec: dapm: power down widget SLIM TX9
MUX
<7>[ 3717.945861] taiko_codec taiko_codec: dapm: power down widget SLIM TX8
MUX
<7>[ 3717.945903] taiko_codec taiko_codec: dapm: power down widget SLIM TX7
MUX
```

```
<7>[ 3717.945945] taiko_codec taiko_codec: dapm: power down widget DEC6 MUX
<7>[ 3717.945987] taiko_codec taiko_codec: dapm: power down widget DEC9 MUX
<7>[ 3717.946030] taiko_codec taiko_codec: dapm: power down widget DEC7 MUX
<7>[ 3717.946376] taiko_codec taiko_codec: dapm: power down widget CDC_CONN
<7>[ 3717.946418] taiko_codec taiko_codec: dapm: power down widget DMIC6
<7>[ 3717.946496] taiko_codec taiko_codec: dapm: power down widget DMIC1
<7>[ 3717.946538] taiko_codec taiko_codec: dapm: power down widget MIC 
BIAS4 External
<7>[ 3717.946581] taiko_codec taiko_codec: dapm: power down widget MIC 
BIAS1 External
<7>[ 3717.946623] taiko_codec taiko_codec: dapm: power down widget Digital 
Mic6
<7>[ 3717.946664] taiko_codec taiko_codec: dapm: power down widget Digital 
Mic5
<7>[ 3717.946742] taiko_codec taiko_codec: dapm: power down widget Digital 
Mic2
<7>[ 3717.946783] taiko_codec taiko_codec: dapm: power down widget Digital 
Mic1
```

A message shows the powering down of DAPM widgets when AANC is disabled.

```
<7>[ 3717.955946] taiko_codec taiko_codec: taiko_put_anc_func: anc_func 0
```

Disable circuit-switch voice Tx path. Stream 0 is SNDRV_PCM_STREAM_CAPTURE and event 1 is SND_SOC_DAPM_STREAM_STOP, which are defined in kernel/include/sound/asound.h.

```
<7>[ 3717.953484]  Circuit-Switch Voice: rtd stream 1 event 2
<7>[ 3717.953538] msm-pcm-routing msm-pcm-routing: dapm: power down widget 
CS-VOICE_UL1
<7>[ 3717.953581] msm-pcm-routing msm-pcm-routing: dapm: power down widget 
Voice_Tx Mixer
<7>[ 3717.953776] msm-pcm-routing msm-pcm-routing: dapm: power down widget 
SLIMBUS_0_TX
<7>[ 3717.953824] msm-pcm-routing msm-pcm-routing: dapm: power down widget 
BE_IN
```

Disable the DAPM widgets for the anc-handset device.

```
<7>[ 3718.027131] taiko_codec taiko_codec: dapm: power down widget DMIC4
<7>[ 3718.027256] taiko_codec taiko_codec: dapm: power down widget ANC1 MUX
<7>[ 3718.027379] taiko_codec taiko_codec: dapm: power down widget MIC 
BIAS3 External
<7>[ 3718.027505] taiko_codec taiko_codec: dapm: power down widget RX1 MIX2
<7>[ 3718.027626] taiko_codec taiko_codec: dapm: power down widget Digital 
Mic4
<7>[ 3718.027746] taiko_codec taiko_codec: dapm: power down widget Digital 
Mic3
```

```
<7>[ 3718.030502] taiko_codec taiko_codec: dapm: power down widget LDO_H
<7>[ 3718.030657] taiko_codec taiko_codec: dapm: power down widget RX1
CHAIN
<7>[ 3718.032048] taiko_codec taiko_codec: dapm: power down widget MCLK
<7>[ 3718.032171] taiko_codec taiko_codec: dapm: power down widget
CLASS_H_DSM MUX
<7>[ 3718.032188] taiko_codec taiko_codec: dapm: power down widget DAC1
<7>[ 3718.032204] taiko_codec taiko_codec: dapm: power down widget RX_BIAS
<7>[ 3718.032324] taiko_codec taiko_codec: dapm: power down widget
EAR_PA_MIXER
<7>[ 3718.032444] taiko_codec taiko_codec: dapm: power down widget EAR PA
<7>[ 3718.032564] taiko_codec taiko_codec: dapm: power down widget EAR
```

22.3.3 QXDM Professional logs

```
MSG [08500/02] QDSP6/High   01:01:49.075 AFEPortAprHandler.cpp  00120
AFESvc: Executing    100ef

MSG [08500/01] QDSP6/Medium 01:01:49.075 AFEPortAprHandler.cpp  00793  AFE
Port Received in-band parameters of size 44 to port ::0x4000!!
MSG [08500/02] QDSP6/High   01:01:49.075 AFEPortAprHandler.cpp  00778
message SUCCESS        0
MSG [08500/02] QDSP6/High   01:01:49.075 AFEPortAprHandler.cpp  00120
AFESvc: Executing    100e5
MSG [08500/02] QDSP6/High   01:01:49.075 AFEPortAprHandler.cpp  00239
AFECmdDmaStart: port_id: 0x4000, sample_rate: 48000, nChannels: 1
MSG [08500/02] QDSP6/High   01:01:49.082 AFEPortAprHandler.cpp  00778
message SUCCESS        0
MSG [08500/02] QDSP6/High   01:01:49.092 AFEPortAprHandler.cpp  00120
AFESvc: Executing    100ef
MSG [08500/01] QDSP6/Medium 01:01:49.092 AFEPortAprHandler.cpp  00793  AFE
Port Received in-band parameters of size 36 to port ::0x4001!!
MSG [08500/02] QDSP6/High   01:01:49.092 AFEPortAprHandler.cpp  00728  AFE:
Received AANC Set Param on port 0x4001
MSG [08500/02] QDSP6/High   01:01:49.092 AFEPortAprHandler.cpp  00752
Passing APR packet to AANC Thread
MSG [08500/02] QDSP6/High   01:01:49.092 AFEPortAprHandler.cpp  00778
message SUCCESS        0
MSG [08500/02] QDSP6/High   01:01:49.092 AFEAancCmdHandler.cpp  00565
received message on AANC CmdQ with sec opcode 200012
MSG [08500/02] QDSP6/High   01:01:49.093 AFEAancCmdHandler.cpp  00050  AANC
Port Cfg command received
MSG [08500/02] QDSP6/High   01:01:49.093 AFEPortAprHandler.cpp  00778
message SUCCESS        0
MSG [08500/02] QDSP6/High   01:01:49.093 AFEPortAprHandler.cpp  00120
AFESvc: Executing    100ef
MSG [08500/02] QDSP6/High   01:01:49.093 AFEPortAprHandler.cpp  00806  AFE
Memory Map PhyLSW(7ee6f000) PhyMSW(0) virtual address(fb1a6000)
```

```
MSG [08500/01] QDSP6/Medium 01:01:49.093 AFEPortAprHandler.cpp  00819  AFE
Port Received out-of-band parameters of size 704 to port ::0x4001!!
MSG [08500/02] QDSP6/High   01:01:49.093 AFEPortAprHandler.cpp  00728  AFE:
Received AANC Set Param on port 0x4001
MSG [08500/02] QDSP6/High   01:01:49.093 AFEPortAprHandler.cpp  00752
Passing APR packet to AANC Thread
MSG [08500/02] QDSP6/High   01:01:49.095 AFEPortAprHandler.cpp  00778
message SUCCESS        0
MSG [08500/02] QDSP6/High   01:01:49.095 AFEAancCmdHandler.cpp  00565
received message on AANC CmdQ with sec opcode 200012
```

Receive the AANC configuration command.

```
MSG [08500/02] QDSP6/High   01:01:49.095 AFEAancCmdHandler.cpp  00383  AANC
enable command received
MSG [08500/02] QDSP6/High   01:01:49.095 AFEAancCmdHandler.cpp  00117  AANC
CFG1 command received
MSG [08500/02] QDSP6/High   01:01:49.095 AFEAancCmdHandler.cpp  00158  AANC
CFG2 command received
```

If AANC is functioning properly, the value of register 0xa0c changes and does not remain constant. DSP writes AANC adaptive gain values to the WCD codec.

```
MSG [08500/02] QDSP6/High   01:01:49.135 AFECodecHandler.cpp  00103
afe_cdc_update_register: reg_addr: 0xa0c, value: 128
MSG [08500/02] QDSP6/High   01:01:49.135 AFECodecHandler.cpp  00103
afe_cdc_update_register: reg_addr: 0xa02, value: 1
MSG [08500/02] QDSP6/High   01:01:49.135 AFECodecHandler.cpp  00103
afe_cdc_update_register: reg_addr: 0xa02, value: 0
MSG [08500/02] QDSP6/High   01:01:49.155 AFECodecHandler.cpp  00103
afe_cdc_update_register: reg_addr: 0xa0c, value: 128
MSG [08500/02] QDSP6/High   01:01:49.156 AFECodecHandler.cpp  00103
afe_cdc_update_register: reg_addr: 0xa02, value: 1
MSG [08500/02] QDSP6/High   01:01:49.156 AFECodecHandler.cpp  00103
afe_cdc_update_register: reg_addr: 0xa02, value: 0
MSG [08500/02] QDSP6/High   01:01:49.175 AFECodecHandler.cpp  00103
afe_cdc_update_register: reg_addr: 0xa0c, value: 128
MSG [08500/02] QDSP6/High   01:01:49.175 AFECodecHandler.cpp  00103
afe_cdc_update_register: reg_addr: 0xa02, value: 1
MSG [08500/02] QDSP6/High   01:01:49.175 AFECodecHandler.cpp  00103
afe_cdc_update_register: reg_addr: 0xa02, value: 0
MSG [08500/02] QDSP6/High   01:01:49.195 AFECodecHandler.cpp  00103
afe_cdc_update_register: reg_addr: 0xa0c, value: 128.
.
.
.
```

22.4 Customization

In /device/qcom/msm8974/system.prop, persist.aanc.enable may be set to false by default. Run the following command to enable AANC every time after the device boots up.

```
adb shell
# setprop persist.aanc.enable true
```

22.4.1 Enable/disable ANC from UI

To enable ANC and ANC during a voice call, the OEM software must enable ANC with the AudioManager in Java as follows.

```
AudioManager audioMgr = (AudioManager)
context.getSystemService(Context.AUDIO_SERVICE);
audioMgr.setParameters("anc_enabled=true"); // to enable
audioMgr.setParameters("anc_enabled=false"); // to disable
```

22.4.2 Verify DSP gain in the codec

```
C:\>adb shell
#mount -t debugfs debugfs /sys/kernel/debug
 exit
C:\> adb shell cat /sys/kernel/debug/asoc/msm8974-taiko-fluid-snd-
card/taiko_codec/codec_reg |grep 20c
```

Wait for a second.

```
C:\> adb shell cat /sys/kernel/debug/asoc/msm8974-taiko-fluid-snd-
card/taiko_codec/codec_reg |grep 20c
```

NOTE: The sound card name is checked on the device if any error is observed when executing this command..

If the value of 0x20c keeps changing, AANC is working. If the value is not updating, AANC may not be working.

In QXDM Professional logs messages that appear every 20 ms, such as the following example, the DSP updates the register 0xA0C every 20 ms. For adaptive ANC, the value changes.

```
MSG          [08500/02]   QDSP6/High          00:06:46.735
AFECodecHandler.cpp  00103  afe_cdc_update_register: reg_addr: 0xa0c,
value: 128
```

22.4.3 Device definition changes for AANC

The OEM may use different mics for AANC when compared to the QTI reference design, e.g., mic configurations are:

- Primary mic for voice call – DMIC1→DEC7→SLIM TX7
- Noise mic – DMIC4→DEC9→SLIM TX 8
- Error mic – DMIC6→DEC6→SLIM TX 9

Text in green typeface indicates the changes that are made in the mixer_paths.xml file for the mic configuration to work. The OEM must ensure that the recording via the mic is working and that the mic bias-related changes are done in the device tree file.

```
<path name="aanc-handset-mic">
     <ctl name="AIF1_CAP Mixer SLIM TX7" value="1" />
     <ctl name="AIF1_CAP Mixer SLIM TX8" value="1" />
     <ctl name="AIF1_CAP Mixer SLIM TX9" value="1" />
     <ctl name="SLIM_0_TX Channels" value="Three" />
     <ctl name="SLIM_0_RX AANC MUX" value="SLIMBUS_0_TX" />
     <ctl name="SLIM TX7 MUX" value="DEC7" />
     <ctl name="DEC7 MUX" value="DMIC1" />
     <ctl name="SLIM TX8 MUX" value="DEC9" />
     <ctl name="DEC9 MUX" value="DMIC4" />
     <ctl name="SLIM TX9 MUX" value="DEC6" />
     <ctl name="DEC6 MUX" value="DMIC6" />
     <ctl name="IIR1 INP1 MUX" value="DEC7" />
</path>

<path name="anc-handset">
     <ctl name="ANC Function" value="ON" />
     <ctl name="SLIM RX1 MUX" value="AIF1_PB" />
     <ctl name="SLIM_0_RX Channels" value="One" />
     <ctl name="RX1 MIX1 INP1" value="RX1" />
     <ctl name="CLASS_H_DSM MUX" value="DSM_HPHL_RX1" />
     <ctl name="DAC1 Switch" value="1" />
     <ctl name="RX1 Digital Volume" value="81" />
     <ctl name="ANC Slot" value="6" />
     <ctl name="ANC1 MUX" value="DMIC4" />
     <ctl
```

22.5 Debugging

Table 22-1 shows common AANC issues that were encountered during previous integration activities and some troubleshooting steps to resolve them.

Table 22-1 Common AANC issues

Issue type	Symptoms	Troubleshooting steps
AANC not working	There is no AANC effect felt	1. Check if the anc_enabled flag is set from UI. 2. Check if the persist.aanc.enable flag is enabled as true in device/qcom/msm8226/system.prop or the equivalent OEM file. 3. Check whether the mixer control settings in the UCM file are correct. 4. Check whether the Symbolic link from /data/misc/audio/wcd9320_anc.bin to /system/etc/firmware/wcd9306/wcd9306_anc.bin in the file qcom/common/rootdir/etc/init.qcom.audio.sh or equivalent OEM file.
Noise	There is noise in the Rx path with AANC enabled.	1. Check if the issue is seen with AANC disabled. If the issue disappears, check whether proper ACDB IDs are chosen for Tx and Rx AANC devices. 2. Collect QXDM Professional logs, kernel, and logcat logs, and contact the Customer Engineering Team at https://support.cdmatech.com.
Pop noise	There is pop noise at the start of voice call with AANC enabled	Ensure that the anc_gain is set to 0x0 in the calibration data for AANC using QACT.

23 DRE and codec sidetone

23.1 DRE

The compander (DRE) system is intended to increase the dynamic range of an input signal. This feature is used for improving SNR, THD, and noise without signal distortion. Compander (DRE) support is available on MSM8974 with WCD9320.

23.1.1 Compander mixer commands

The compander is enabled or disabled as follows:

- tinymix “COMP0 Switch” 1 – Enables the compander on speaker drive path
- tinymix “COMP0 Switch” 0 – Disables the compander on speaker drive path
- tinymix “COMP1 Switch” 1 – Enables the compander on headphone path
- tinymix “COMP1 Switch” 0 – Disables the compander on headphone path
- tinymix “COMP2 Switch” 1 – Enables the compander on lineout path
- tinymix “COMP2 Switch” 0 – Disables the compander on lineout path

Verify DRE performance using adb commands

To enable the DRE in the headset path and set up the headset for playback, run the following adb mixer commands:

```
adb shell "tinymix 'SLIMBUS_0_RX Audio Mixer MultiMedia1' 1"
adb shell "tinymix 'SLIM RX1 MUX' 'AIF1_PB'"
adb shell "tinymix 'SLIM RX2 MUX' 'AIF1_PB'"
adb shell "tinymix 'SLIM_0_RX Channels' 'Two'"
adb shell "tinymix 'RX1 MIX1 INP1' 'RX1'"
adb shell "tinymix 'RX2 MIX1 INP1' 'RX2'"
adb shell "tinymix 'HPHL DAC Switch' 1"
adb shell "tinymix 'HPHL Volume' '80'"
adb shell "tinymix 'HPHR Volume' '80'"
adb shell "tinymix 'CLASS_H_DSM MUX' 'DSM_HPHL_RX1'"
adb shell "tinymix 'RX1 Digital Volume' '67'"
adb shell "tinymix 'RX2 Digital Volume' '67'"
adb shell "tinymix 'COMP1 Switch' 1"

#Start the Playback using tiny play
adb shell "tinyplay /data/test.wav"
```

Check the audio on the headset device.

23.1.2 Compander limitations

- The codec driver does not handle switch-off compander while the playback path is still active.
- When the compander is on, analog gain is uncontrolled. Only codec digital gain is controlled.

23.2 Codec sidetone

The codec sidetone sends voice call codec Tx path mic recording data back to the codec Rx path as reference data to cancel noise. See Figure 23-1.

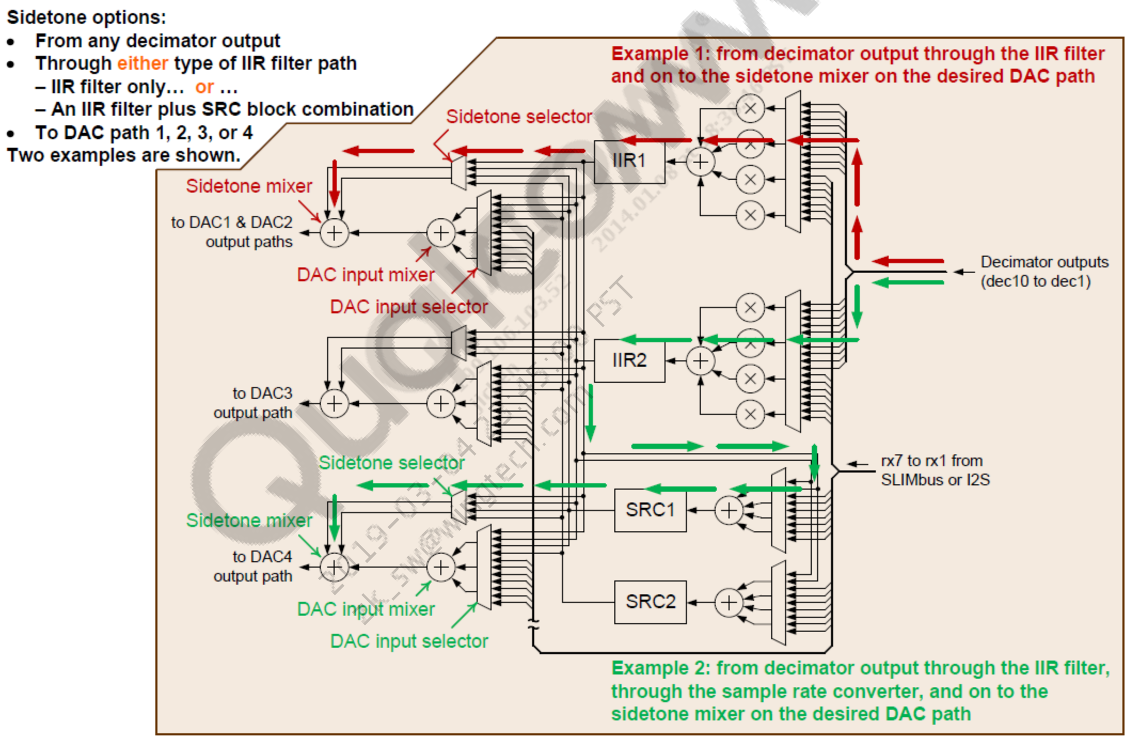

Sidetone options:
• From any decimator output
• Through either type of IIR filter path
– IIR filter only… or …
– An IIR filter plus SRC block combination
• To DAC path 1, 2, 3, or 4
Two examples are shown.
Example 1: from decimator output through the IIR filter and on to the sidetone mixer on the desired DAC path
Sidetone selector
IIR1
Sidetone mixer
to DAC1 & DAC2 output paths
DAC input mixer
DAC input selector
Decimator outputs (dec10 to dec1)
IIR2
to DAC3 output path
rx7 to rx1 from SLIMbus or I2S
Sidetone selector
SRC1
Sidetone mixer
to DAC4 output path
SRC2
DAC input mixer
DAC input selector
Example 2: from decimator output through the IIR filter, through the sample rate converter, and on to the sidetone mixer on the desired DAC path

Figure 23-1 Codec sidetone example

23.2.1 Configure sidetone

Codec IIR filters only accept 30-bit filter coefficients, so only 30-bit filter coefficients is generated.

The following example shows coefficients for the one-band HPF with an 800 Hz cutoff.

IIR1 Band1 <b0> < b1> <b2> <a1> <a2>

Band 1 – 0x0EDBA49B, 0x2248B6C9, 0x0EDBA49B, 0x225D9BA6, 0x0DCC2E13

While testing tinymixer commands from the command line, pass the only digital values as shown in the following example:

```
adb shell "tinymix 'IIR1 Band1' 249275547 575190729 249275547 576560038
231484947"
```

23.2.2 Test sidetone

The following commands are run to test sidetone.

```
adb shell "tinymix 'DEC5 MUX' ADC2"
adb shell "tinymix 'ADC2 Volume' '80'"
adb shell "tinymix 'DEC5 Volume' '67'"
adb shell "tinymix 'RX2 MIX1 INP1' 'IIR1'"
adb shell "tinymix 'RX1 MIX1 INP1' 'IIR1'"
adb shell "tinymix 'IIR1 INP1 MUX' 'DEC5'"
adb shell "tinymix 'IIR1 INP1 Volume' '80'"
adb shell "tinymix 'IIR1 Band1' 249275547 575190729 249275547 576560038
231484947"
adb shell "tinymix 'IIR1 Enable Band1' '1'"
```

In the Rx path, the sidetone has an HPF cutoff frequency of 800 Hz.

In the mixer_path.xml configuration, add the above IIR mixer controls into the corresponding path, as shown in the following example.

```
    <path name="sidetone-iir">
        <ctl name="IIR1 Enable Band1" value="1" />
        <ctl name="IIR1 Enable Band2" value="1" />
        <ctl name="IIR1 Enable Band3" value="1" />
        <ctl name="IIR1 Enable Band4" value="1" />
        <ctl name="IIR1 Enable Band5" value="1" />
    </path>

    <path name="sidetone-headphones">
        <path name="sidetone-iir" />
        <ctl name="IIR1 INP1 Volume" value="77" />
        <ctl name="RX1 MIX2 INP1" value="IIR1" />
        <ctl name="RX2 MIX2 INP1" value="IIR1" />
    </path>
```

24 24-bit playback

Tinyplay does not support zero padding for 24-bit PCM samples before enabling 24-bit playback on MSM8974 on KitKat. Hence, amix, aplay, and libalsa-intf.so are pushed to the device to replace tinyplay. The following steps set up amix and aplay. Amix, aplay, and libalsa-intf.so are attached.

```
adb push amix /system/bin
adb push aplay /system/bin/
adb push libalsa-intf.so /system/lib/
adb shell chmod 777 /system/bin/aplay
adb shell chmod 777 /system/bin/amix
```

24.1 Enable 24-bit playback

Start the 24-bit and 192 kHz sample rate audio playback.

```
adb shell "amix 'SLIMBUS_0_RX Audio Mixer MultiMedia1' 1"
adb shell "amix 'SLIM RX1 MUX' 'AIF1_PB'"
adb shell "amix 'SLIM RX2 MUX' 'AIF1_PB'"
adb shell "amix 'SLIM_0_RX Channels' 'Two'"
adb shell "amix 'RX1 MIX1 INP1' 'RX1'"
adb shell "amix 'RX2 MIX1 INP1' 'RX2'"
adb shell "amix 'HPHL DAC Switch' 1"
adb shell "amix 'HPHL Volume' '80%'"
adb shell "amix 'HPHR Volume' '80%'"
adb shell "amix 'CLASS_H_DSM MUX' 'DSM_HPHL_RX1'"
adb shell "amix 'RX1 Digital Volume' '67%'"
adb shell "amix 'RX2 Digital Volume' '67%'"
adb shell "amix 'SLIM_0_RX Format' 'S24_LE'"
adb shell "amix 'SLIM_0_RX SampleRate' 'KHZ_192'"

adb shell "aplay /sdcard/Bloom.wav"
```

24.2 Disable 24-bit playback

Stop audio playback by pressing Ctrl +C or waiting until playback is completed. Once playback is stopped, disable the device.

```
adb shell "amix 'SLIMBUS_0_RX Audio Mixer MultiMedia1' 0"
adb shell "amix 'SLIM RX1 MUX' 'ZERO'"
adb shell "amix 'SLIM RX2 MUX' 'ZERO'"
adb shell "amix 'RX1 MIX1 INP1' 'ZERO'"
adb shell "amix 'RX2 MIX1 INP1' 'ZERO'"
adb shell "amix 'HPHL DAC Switch' 0"
adb shell "amix 'SLIM_0_RX Format' 'S16_LE'"
adb shell "amix 'SLIM_0_RX SampleRate' 'KHZ_48'"
```

A References

A.1 Related documents

Title	Number
Qualcomm Technologies, Inc.	
QACT V2.X.X User Guide	80-VM407-2
Qualcomm Product Support Tool (QPST) 2.7 User Guide	80-V1400-3
Hexagon Access Audio/Voice PCM/BIT Stream Logging with QXDM Professional	80-N3470-4
WCD9320 Audio Codec IC Design Guidelines/Training Slides	80-NA556-5
Hexagon Multimedia: Core Service API Interface Specification	80-N3073-1
MSM8974 Android Audio Overview	80-NA157-28
WCD9320 Audio Codec Hardware Register Description for OEMs	80-NA556-2
Multibutton Headset Control (MBHC) Version 1 Application Note	80-NA556-11
Standards	
OpenMAX Integration Layer Application Programming Interface Specification	Version 1.1.2

A.2 Acronyms and terms

Acronym or term	Definition
AANC	Adaptive ANC
ACDB	Audio calibration database
APM	Audio policy manager
BE	Back-end
CE	Customer engineering
DCE	DC estimation
FE	Front end
HAL	Hardware abstraction layer
HFP	Hands-free profile
IMS	IP multimedia system
MBHC	Multibutton headset control
MPSS	Modem processor subsystem
MVS	Multimode vocoder services
NC	Normally closed
NGD	Nonported generic device
NO	Normally open

Acronym or term	Definition
PGD	Ported generic device
PIL	Peripheral image loader
QMI	Qualcomm modem interface
SS	Source selection
WFD	Wi-Fi display
WS	Word select