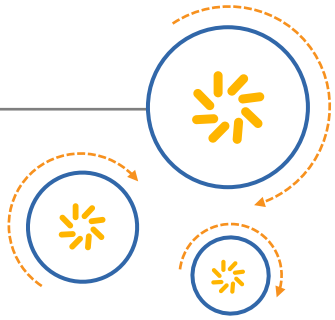
Qualcomm Technologies, Inc.



Linux Audio Device Management

80-NL239-28 E

March 14, 2017

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

Revision history

Revision	Date	Description
A	April 2014	Initial release
B	June 2014	▪ Updated the document title to include MSM8936/MSM8939 chipsets ▪ Updated Sections 1.1 and 1.2
C	December 2015	▪ Updated document title to include MSM8937chipset ▪ Updated Section 1.1
D	March 2016	▪ Updated document title to be generic ▪ Updated Section 1.1
E	March 2017	Numerous updates were made to this document to include SDM630 chipset information; to be read in its entirety

Contents

1 Introduction

1.1 Purpose

This document describes the process of adding new audio devices in the MSM8916/MSM8936/MSM8939/MSM8937/MSM8953/SDM660/SDM630 chipsets. It is intended for customers who need to know how to add and calibrate new audio devices in the MSM8916/MSM8936/MSM8939/MSM8937/MSM8953SDM660/SDM630 chipsets.

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.

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 Add and calibrate audio device

This chapter provides the procedure to add and calibrate a new device.

2.1 Add a new device with new calibration data

1. Add the device entry in the Audio Calibration Data Base (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, add an appropriate device pair 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 the device_table in hardware/qcom/audio/hal/msm8916/platform.h and platform.c.
5. Include the ACDB device ID for the device in the acdb_device_table in hardware/qcom/audio/hal/msm8916/platform.c.
6. Map the device name mentioned in the device_table to the Android device. This requires a change in the platform_get_output_snd_device () function in the hardware/qcom/audio/hal/msm8916/platform.c file.

2.2 Device table

In hardware/qcom/audio/hal/msm8916/platform.h, 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 hardware/qcom/audio/hal/msm8916/platform.c, the device_table defines the string name for each device.

```
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",
```

2.3 Mixer path XML

NOTE: This section describes the mixer paths, definitions, and specifications in the MSM8916 chipset XML files. The XML files for other chipsets follow similar naming conventions.

The mixer path XML contains the list of mixer control definition and the mixer control path. Based on the mixer path XML setting, the libaudioroute library module from Google gets the route for each stream mixer controls. The mixer path XML is located in device/qcom/msm8916_32/mixer_paths.xml.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<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="HPHL Volume" value="9" />
<ctl name="HPHR Volume" value="9" />
<ctl name="RX1 Digital Volume" value="84" />
<ctl name="RX2 Digital Volume" value="84" />
<ctl name="RX3 Digital Volume" value="84" />
```

2.3.1 Mixer path definition

2.3.1.1 Audio route (frontend (FE) to backend (BE))-specific mixer settings

This section describes the 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="PRI_MI2S_RX Audio Mixer MultiMedia1" value="1" />
  </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">
      <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="deep-buffer-playback usb-headphones">
      <path name="deep-buffer-playback afe-proxy" />
  </path>

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

  <path name="deep-buffer-playback transmission-fm">
      <ctl name="INTERNAL_FM_RX Audio Mixer MultiMedia1" value="1" />
```

```
    </path>

    <path name="low-latency-playback">
        <ctl name="PRI_MI2S_RX Audio Mixer MultiMedia5" value="1" />
    </path>
```

2.3.2 Sound device-specific mixer settings

This section describes sound device-specific mixer settings.

```
<!-- These are actual sound device specific mixer settings -->
    <path name="adc1">
        <ctl name="DEC1 MUX" value="ADC1" />
    </path>

    <path name="adc2">
        <ctl name="DEC1 MUX" value="ADC2" />
    </path>

    <path name="speaker">
        <ctl name="RX3 MIX1 INP1" value="RX1" />
        <ctl name="SPK DAC Switch" value="1" />
    </path>

    <path name="speaker-mic">
        <path name="adc1" />
        <ctl name="IIR1 INP1 MUX" value="DEC1" />
    </path>

    <path name="handset">
        <ctl name="RX1 MIX1 INP1" value="RX1" />
        <ctl name="RDAC2 MUX" value="RX1" />
        <ctl name="RX1 Digital Volume" value="84" />
        <ctl name="EAR_S Switch" value="1" />
    </path>

    <path name="handset-mic">
        <path name="adc1" />
    </path>

    <path name="headphones">
        <ctl name="MI2S_RX Channels" value="Two" />
        <ctl name="RX1 MIX1 INP1" value="RX1" />
        <ctl name="RX2 MIX1 INP1" value="RX2" />
        <ctl name="RDAC2 MUX" value="RX2" />
        <ctl name="HPHL Switch" value="1" />
        <ctl name="HPHR Switch" value="1" />
```

```
    </path>

    <path name="headset-mic">
        <path name="adc2" />
        <ctl name="ADC2 MUX" value="INP2" />
        <ctl name="IIR1 INP1 MUX" value="DEC1" />
    </path>
```

2.4 ACDB device ID table

Unlike Android Jelly Bean, in Android KitKat, the ACDB device ID information does not include the mixer path XML in UCM replacement. The tinyALSA audio HAL maintains the ACDB ID information. The acdb_device_table is located in hardware/qcom/audio/hal/msm8916/platform.c and defines the ACDB ID for each device.

The ACDB IDs for each sound device are listed below:

```
/* ACDB IDs (audio DSP path configuration IDs) for each sound device */
static 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] = 45,
    [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,
```

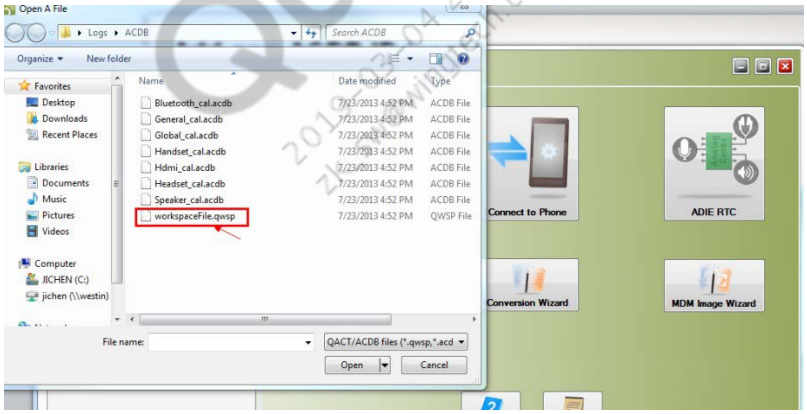
2.5 Add a device

2.5.1 Create an ACDB device ID and calibration data

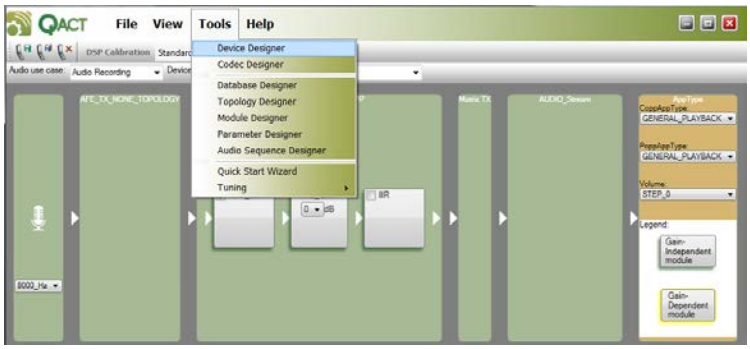
To add a new device file to ACDB using QACT™ platform in Offline mode:

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

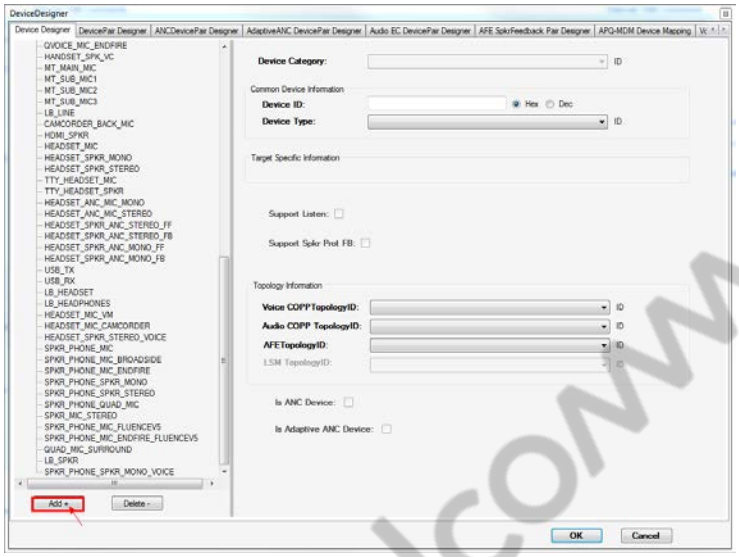
2. Select **workspaceFile.qwsp** located at vendor\qcom\proprietary\mm-audio\audcal\family-b\acdbdata\8916) and click **Open**.



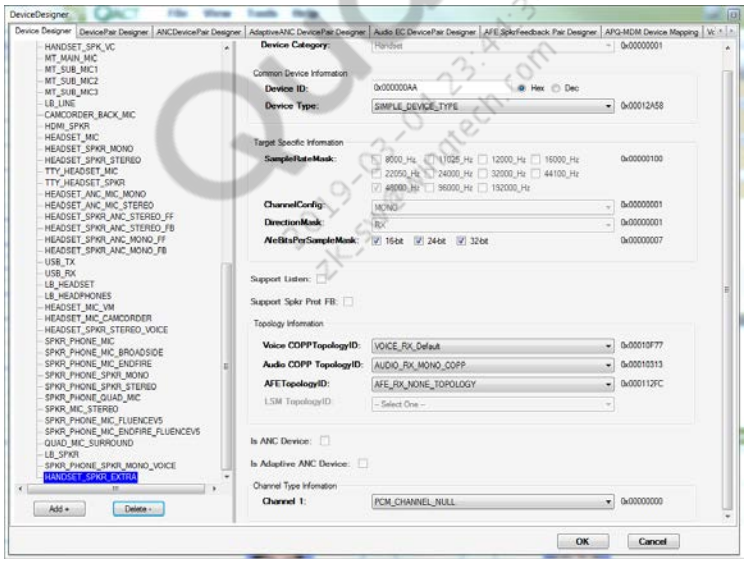
3. Select **Tools→Device Designer**.



4. In the DeviceDesigner window, under the Device Designer tab, click **Add+** to add a new ACDB device.



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



The following naming conventions apply to voice and audio devices:

- A microphone device is referred to as a Tx device or a mic device. This represents a signal path from a microphone to a codec to the mDSP. Such devices are clearly identifiable by the presence of the string MIC in the device name, that is, HANDSET_MIC.
- A speakerphone device is referred to as an Rx device or a SPKR device. This represents a signal path from the MSM™ mDSP to the codec to the output transducer (speaker, earpiece, headset, etc.). Such devices are clearly identifiable by the presence of the string SPKR in the device name, e.g., 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 part of a voice device pair must be 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 are individually defined. To create custom use cases for voice calls, a device pair is needed.

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

- Device Category – The category of the devices that are added, i.e., the physical representation of the device
- Device ID – Must be a unique value, i.e., a random number such as 0x000000AA
- Device Type – SIMPLE_DEVICE_TYPE can only be chosen
- SampleRateMask – Fill in the sampling rates that the device supports
- DirectionMask – Determine if the device will be 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

2.5.2 Create a mixer path XML element for the newly added device

NOTE: This section describes the mixer path for the MSM8916 chipset XML files. The XML files for other chipsets follow similar naming conventions.

Create an XML element for the new device in device/qcom/msm8916_32/mixer_paths.xml.

```
<path name="speaker-extra">
       <ctl name="RX3 MIX1 INP1" value="RX1" />
       <ctl name="SPK DAC Switch" value="1" />
 </path>
Note that by default, the given mixer control is set to value 0, which will
be called when the mixer path is torn down.
<?xml version="1.0" encoding="ISO-8859-1"?>
<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="RX3 MIX1 INP1" value="ZERO" />
        <ctl name="SPK DAC Switch" value="0" />
    …
    …
</mixer>
```

2.5.3 Audio HAL changes

2.5.3.1 Define the new device entry in device_table

Update the hardware/qcom/audio/hal/msm8916/platform.h and platform.c files to add the entry in the device_table for the new device name.

A meaningful string is used, i.e., Speaker Extra. This string must match the mixer path XML name in Section 2.5.2.

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

```
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 the 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",
```

2.5.3.2 Add ACDB ID mapping

Add the ACDB device ID to acdb_device_table in hardware/qcom/audio/hal/msm8916/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,
```

2.5.3.3 Add a new mapping from Android device to SND device

In the platform_get_input_snd_device ()/platform_get_output_snd_device() functions 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 the Android framework is DEVICE_OUT_SPEAKER (0x2). This ID mapping can be found in 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;
```

2.6 Log collection

This section describes the necessary logs to check whether the new device is added properly.

2.6.1 User space logs

To generate the user space logs:

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

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

2.6.2 Kernel logs

If kernel logs are required, enable them:

```
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
```

A References

A.1 Acronyms and terms

Acronym or term	Definition
ACDB	Audio calibration data base
BE	Backend
FE	Frontend