

AW882XX Android Driver(Qcom AudioReach)

Version: V1.5

Date: June, 2024

Content

1. INFORMATION	3
2. RX DRIVER INTEGRATION.....	3
2.1 AW882XX DRIVER PORTING	3
2.2 PLATFORM DRIVER CONFIGURATION	8
2.3 PLATFORM ROUTES CONFIGURATION	9
2.4 DRIVER PORTING VERIFICATION	10
3. TX DRIVER INTEGRATION	11
3.1 HAL LAYER PORTING	11
3.2 XML CONFIGURATION	11
4. CALIBRATION AND LOW VOLTAGE PROTECTION.....	12
5. DEBUG INTERFACE.....	12
5.1 NODE.....	12
5.2 KCONTROL	14

1. INFORMATION

Smart PA(with IV)	aw88257、aw88258、aw88261、aw88261s、aw88262、aw88263、aw88263h、aw88263s、aw88264、aw88265、aw88266、aw88266s、aw88270、aw88274、aw88299、aw88461、aw88271
Smart PA(no IV)	aw88298、aw88298g、aw88266a、aw88230、aw88082、aw88252
I ² C Address	0x30/0x31/0x32/0x33/0x34/0x35/0x36/0x37

2. RX DRIVER INTEGRATION

2.1 AW882XX DRIVER PORTING

2.1.1 DTS Configuration

Open file `/vendor/qcom/propriety/audio-devicetree/xxxx.dtsi` and add DTS Configuration.

Single PA Configuration

```
i2c_x { /*x means the i2c bus number*/
+ /* AWINIC AW882XX mono Smart PA */
+ aw882xx_smartpa@34 {
+ compatible = "awinic,aw882xx_smartpa";
+ reg = <0x34>;
+ /* You cannot configure reset-gpio for aw88230, aw88261, aw88257,
aw88082 or aw88265*/
+ reset-gpio = <&tlmm 84 0>;
+ irq-gpio = <&tlmm 136 0x2008>;
+ sync-load = <1>; /*Firmware loading mode, qcom platform uses
synchronous loading*/
+ aw-re-min = <4000>; /*Minimum calibration value (mohms)*/
+ aw-re-max= <30000>; /*Maximum calibration value (mohms)*/
+ aw-cali-mode = "none";
+ status = "okay";
+ };
+ /* AWINIC AW882XX mono Smart PA End */
```

Multiple PA Configuration

```
i2c_x { /*x means the i2c bus number*/
+ /* AWINIC AW882XX Smart PA */
+ aw882xx_smartpa@34 {
+ compatible = "awinic,aw882xx_smartpa";
+ reg = <0x34>;
+ reset-gpio = <&tlmm 84 0>;
+ irq-gpio = <&tlmm 136 0x2008>;
+ sync-load = <1>;
+ sound-channel = <0>; /*0:pri_l 1:pri_r 2:sec_l 3:sec_r*/
+ aw-re-min = <4000>;
+ aw-re-max= <30000>;
+ aw-cali-mode = "none";
+ status = "okay";
+ };
```

```
+      aw882xx_smartpa@35 {
+      compatible = "awinic,aw882xx_smartpa";
+      reg = <0x35>;
+      reset-gpio = <&tlmm 82 0>;
+      irq-gpio = <&tlmm 143 0x2008>;
+      sync-load = <1>;
+      sound-channel = <1>;      /*0:pri_l 1:pri_r 2:sec_l 3:sec_r*/
+      aw-re-min = <4000>;
+      aw-re-max= <30000>;
+      aw-cali-mode = "none ";
+      status = "okay";
+  };
+
+      /* AWINIC AW882XX Smart PA End */
```

2.1.2 Driver Configuration

Add Source Code

Create the aw882xx folder under the [/vendor/qcom/opensource/audio-kernel/asoc/codecs](#) path and add the driver source code.

```
aw882xx_calib.c, aw882xx_calib.h, aw882xx_monitor.c, aw882xx_monitor.h,
aw882xx.c, aw882xx.h, aw882xx_device.c, aw882xx_device.h, aw882xx_dsp.c,
aw882xx_dsp.h, aw882xx_init.c, aw882xx_init.h, aw882xx_log.h,
aw882xx_spin.c, aw882xx_spin.h, aw882xx_data_type.h, aw882xx_bin_parse.c,
aw882xx_bin_parse.h
```

Open AudioReach Macro

Comment AW_QCOM_PLATFORM macro definition, and open AW_AUDIOREACH_PLATFORM macro definition.

```
#ifndef __AW882XX_DSP_H__
#define __AW882XX_DSP_H__

/*#define AW_QCOM_PLATFORM*/
#define AW_AUDIOREACH_PLATFORM
```

Add Makefile And Kbuild

Create Makefile and Kbuild files in aw882xx path. The following is an example of SM8450(sound card named **WAPIO**) configuration. For details, please refer to other modules of the platform Makefile and Kbuild.

1) Kbuild Configuration

```
# We can build either as part of a standalone Kernel build or as
# an external module. Determine which mechanism is being used
ifeq ($(MODNAME),)
    KERNEL_BUILD := 1
else
    KERNEL_BUILD := 0
endif
```

```

ifeq ($(KERNEL_BUILD), 1)
    # These are configurable via Kconfig for kernel-based builds
    # Need to explicitly configure for Android-based builds
    AUDIO_BLD_DIR := $(shell pwd)/kernel/msm-5.4
    AUDIO_ROOT := $(AUDIO_BLD_DIR)/techpack/audio
endif

ifeq ($(KERNEL_BUILD), 0)
    ifeq ($(CONFIG_ARCH_LAHAINA), y)
        include $(AUDIO_ROOT)/config/lahainaauto.conf
        INCS += -include $(AUDIO_ROOT)/config/lahainaautoconf.h
    endif
    ifeq ($(CONFIG_ARCH_WAIPIO), y)
        include $(AUDIO_ROOT)/config/waipioauto.conf
        INCS += -include $(AUDIO_ROOT)/config/waipioautoconf.h
    endif
    ifeq ($(CONFIG_ARCH_LITO), y)
        include $(AUDIO_ROOT)/config/litoauto.conf
        INCS += -include $(AUDIO_ROOT)/config/litoautoconf.h
    endif
endif

# As per target team, build is done as follows:
# Defconfig : build with default flags
# Slub      : defconfig + CONFIG_SLUB_DEBUG := y +
#            CONFIG_SLUB_DEBUG_ON := y + CONFIG_PAGE_POISONING := y
# Perf      : Using appropriate msmXXXX-perf_defconfig
#
# Shipment builds (user variants) should not have any debug feature
# enabled. This is identified using 'TARGET_BUILD_VARIANT'. Slub builds
# are identified using the CONFIG_SLUB_DEBUG_ON configuration. Since
# there is no other way to identify defconfig builds, QTI internal
# representation of perf builds (identified using the string 'perf'),
# is used to identify if the build is a slub or defconfig one. This
# way no critical debug feature will be enabled for perf and shipment
# builds. Other OEMs are also protected using the TARGET_BUILD_VARIANT
# config.

##### UAPI #####
UAPI_DIR := uapi/audio
UAPI_INC := -I$(AUDIO_ROOT)/include/$(UAPI_DIR)

##### COMMON #####
COMMON_DIR :=include
COMMON_INC :=-I$(AUDIO_ROOT)/$(COMMON_DIR)

##### AWINIC #####

# for AW882XX Codec
ifdef CONFIG_SND_SOC_AW882XX
    AW882XX_OBJS += aw882xx.o
    AW882XX_OBJS += aw882xx_device.o
    AW882XX_OBJS += aw882xx_calib.o
    AW882XX_OBJS += aw882xx_monitor.o
    AW882XX_OBJS += aw882xx_init.o
    AW882XX_OBJS += aw882xx_dsp.o

```

```

AW882XX_OBJS += aw882xx_bin_parse.o
AW882XX_OBJS += aw882xx_spin.o
endif

LINUX_INC += -Iinclude/linux

INCS +=      $(COMMON_INC) \
            $(UAPI_INC)

EXTRA_CFLAGS += $(INCS)

CDEFINES += -DANI_LITTLE_BYTE_ENDIAN \
            -DANI_LITTLE_BIT_ENDIAN \
            -DDOT11F_LITTLE_ENDIAN_HOST \
            -DANI_COMPILER_TYPE_GCC \
            -DANI_OS_TYPE_ANDROID=6 \
            -DPTT_SOCKET_SVC_ENABLE \
            -Wall\
            -Werror\
            -D__linux__

KBUILD_CPPFLAGS += $(CDEFINES)

# Currently, for versions of gcc which support it, the kernel Makefile
# is disabling the maybe-uninitialized warning. Re-enable it for the
# AUDIO driver. Note that we must use EXTRA_CFLAGS here so that it
# will override the kernel settings.
ifeq ($(call cc-option-yn, -Wmaybe-uninitialized),y)
EXTRA_CFLAGS += -Wmaybe-uninitialized
endif
#EXTRA_CFLAGS += -Wmissing-prototypes

ifeq ($(call cc-option-yn, -Wheader-guard),y)
EXTRA_CFLAGS += -Wheader-guard
endif

# Module information used by KBuild framework
obj-$(CONFIG_SND_SOC_AW882XX) += aw882xx_dlm.o
aw882xx_dlm-y := $(AW882XX_OBJS)

# inject some build related information
DEFINES += -DBUILD_TIMESTAMP=\"$(shell date -u +'%Y-%m-%dT%H:%M:%SZ')\"

```

2) Makefile Configuration

Makefile writing reference is as follows, please determine whether to add according to the platform.

```

modules:
    $(MAKE) -C $(KERNEL_SRC) M=$(M) modules $(KBUILD_OPTIONS) VERBOSE=1
modules_install:
    $(MAKE) M=$(M) -C $(KERNEL_SRC) modules_install
clean:
    $(MAKE) -C $(KERNEL_SRC) M=$(M) clean

```

3) The modification of Kbuild

Modify the **Kbuild** file while under the **/vendor/qcom/opensource/audio-kernel/asoc/codecs** path.

```
diff --git a/asoc/codecs/Kbuild b/asoc/codecs/Kbuild
index 545904f..f257823 100644
--- a/asoc/codecs/Kbuild
+++ b/asoc/codecs/Kbuild
@@ -260,6 +260,7 @@ ifeq ($(KERNEL_BUILD), 1)
    obj-y += lpass-cdc/
    obj-y += wsa883x/
    obj-y += rouleux/
+   obj-y += aw882xx/
endif
# Module information used by KBuild framework
obj-$(CONFIG_WCD9XXX_CODEC_CORE) += wcd_core_dlm.o
```

4) The modification of Android.mk

Modify the **Android.mk** file while under the **/vendor/qcom/opensource/audio-kernel** path.

```
diff --git a/Android.mk b/Android.mk
index a3bb903..021e820 100644
--- a/Android.mk
+++ b/Android.mk
@@ -374,6 +374,15 @@ include
$(DLKM_DIR)/Build_external_kernelmodule.mk
+#####
+include $(CLEAR_VARS)
+LOCAL_SRC_FILES := $(AUDIO_SRC_FILES)
+LOCAL_MODULE := aw882xx_dlm.ko
+LOCAL_MODULE_KBUILD_NAME := asoc/codecs/aw882xx/aw882xx_dlm.ko
+LOCAL_MODULE_TAGS := optional
+LOCAL_MODULE_DEBUG_ENABLE := true
+LOCAL_MODULE_PATH := $(KERNEL_MODULES_OUT)
+include $(DLKM_DIR)/Build_external_kernelmodule.mk
```

Define Compilation Options

1) Add config in **/vendor/qcom/opensource/audio-kernel/config/waipioauto.conf**

```
export CONFIG_SND_SOC_AW882XX=m
```

2) Add config in **/vendor/qcom/opensource/audio-kernel/config/waipioautoconf.h**

```
#define CONFIG_SND_SOC_AW882XX 1
```

2.1.3 Bin Configuration

PA needs to configure register parameters to work normally. The configuration steps of PA bin file are as follows:

Path Configuration

Add the directory of bin file in the firmware_class.c, the general directory is **vendor/firmware**

```
static const char * const fw_path[] = {
    fw_path_para,
    "/vendor/firmware",           /*Add a path*/
    "/lib/firmware/updates/" UTS_RELEASE,
    "/lib/firmware/updates",
    "/lib/firmware/" UTS_RELEASE,
    "/lib/firmware"
};
```

(PS: In the debugging stage, you can directly push the bin file to /vendor/ firmware/)

Bin Selection

Select the bin according to the platform I2S output format and PA quantity, taking aw88230 as an example:



2.2 PLATFORM DRIVER CONFIGURATION

2.2.1 DAI_LINK Configuration

1) Add DAI_LINK configuration in msm_dailink.h, take quin_mi2s for example.

Single PA Configuration

```
#ifdef CONFIG_SND_SOC_AW882XX
SND_SOC_DAILINK_DEFS(quin_mi2s_rx_smartpa,
    DAILINK_COMP_ARRAY(COMP_CPU("snd-soc-dummy-dai")),
    /*I2C BUS:0, I2C Address:34*/
    DAILINK_COMP_ARRAY(COMP_CODEC("aw882xx_smartpa.0-0034", "aw882xx-aif-0-34")),
    DAILINK_COMP_ARRAY(COMP_PLATFORM("snd-soc-dummy")));
/*If you need IV feedback, Add the following configuration*/
SND_SOC_DAILINK_DEFS(quin_mi2s_tx_smartpa,
    DAILINK_COMP_ARRAY(COMP_CPU("snd-soc-dummy-dai")),
```

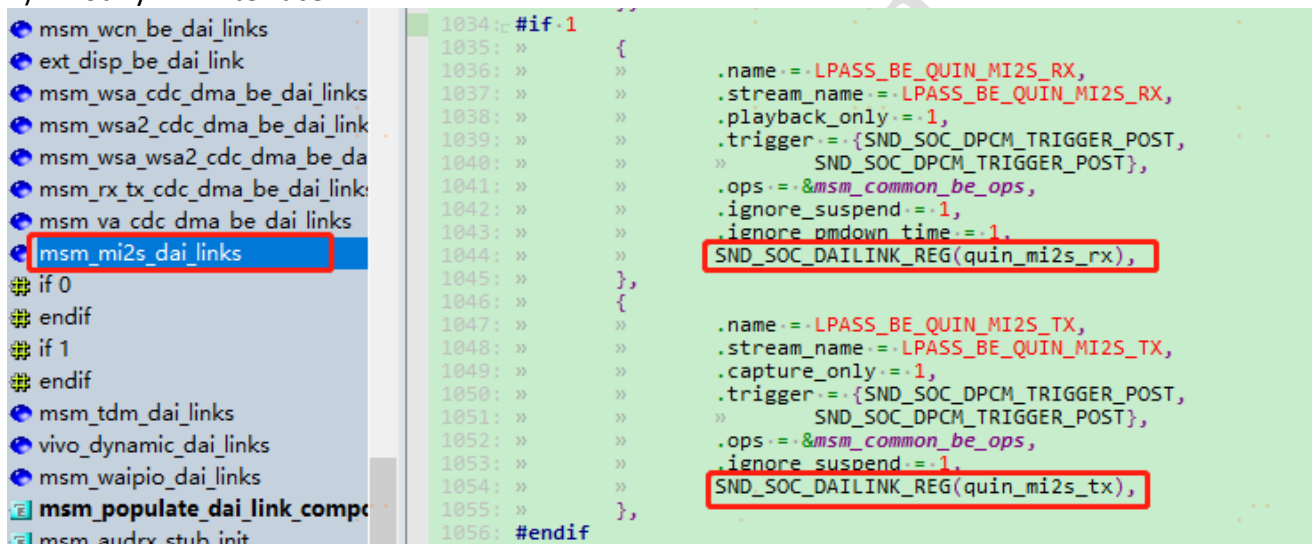


```
DAILINK_COMP_ARRAY (COMP_CODEC ("aw882xx_smartpa.0-0034", "aw882xx-aif-0-34")),
DAILINK_COMP_ARRAY (COMP_PLATFORM ("snd-soc-dummy")));
```

Multiple PA Configuration

```
#ifdef CONFIG_SND_SOC_AW882XX
SND_SOC_DAILINK_DEFS (quin_mi2s_rx_smartpa,
    DAILINK_COMP_ARRAY (COMP_CPU ("snd-soc-dummy-dai")),
    DAILINK_COMP_ARRAY (COMP_CODEC ("aw882xx_smartpa.0-0034", "aw882xx-aif-0-34"),
        COMP_CODEC ("aw882xx_smartpa.0-0035", "aw882xx-aif-0-35")),
    DAILINK_COMP_ARRAY (COMP_PLATFORM ("snd-soc-dummy")));
/*If you need IV feedback, Add the following configuration*/
SND_SOC_DAILINK_DEFS (quin_mi2s_tx_smartpa,
    DAILINK_COMP_ARRAY (COMP_CPU ("snd-soc-dummy-dai")),
    DAILINK_COMP_ARRAY (COMP_CODEC ("aw882xx_smartpa.0-0034", "aw882xx-aif-0-34"),
        COMP_CODEC ("aw882xx_smartpa.0-0035", "aw882xx-aif-0-35")),
    DAILINK_COMP_ARRAY (COMP_PLATFORM ("snd-soc-dummy")));
```

2) Modify DAI interface



2.3 PLATFORM ROUTES CONFIGURATION

2.3.1 XML Configuration

Resourcemanager.xml Configuration

1) Modify the PAL_DEVICE_OUT_SPEAKER DEVICE configuration.

```
<out-device>
  <id>PAL_DEVICE_OUT_SPEAKER</id>
  <!--back_end_name is consistent with the stream_name rx of quin_mi2s in dai_link-->
  <back_end_name>MI2S-LPAIF_VA-RX-PRIMARY</back_end_name>
  <max_channels>2</max_channels>
  <channels>2</channels>
  <samplerate>48000</samplerate>
  <snd_device_name>speaker</snd_device_name>
  <!--For products without IV, set speaker_protection_enabled to 0-->
  <speaker_protection_enabled>1</speaker_protection_enabled>
```

```

<!-- To enable external EC reference the flag extec_ref_enabled must
be added as 1 -->
<ext_ec_ref_enabled>0</ext_ec_ref_enabled>
<cps_enabled>0</cps_enabled>
<!--specify the supported bit format on speaker end point -->
<!--this should be in the form of pal_audio_fmt_id-->
<!--e.g
<supported_bit_format>PAL_AUDIO_FMT_PCM_S24_LE</supported_bit_format>-->
<supported_bit_format>PAL_AUDIO_FMT_PCM_S32_LE</supported_bit_format>
<!-- RAS should be enabled with speaker protection always -->
<ras_enabled>0</ras_enabled>
<!-- Flag to specify speaker in case Voice call is switched from
handset to speaker with only 1 speaker.
Values: Both speaker : 0
Right speaker: 1
Left speaker : 2

```

(PS: As shown below, the string corresponding to **stream_name** is **MI2S-LPAIF_VA-RX-PRIMARY**)

Mixer_paths_xxx.xml Configuration

Invalidate path name="speaker" (where "speaker" corresponds to snd_device_name in resourcemanager.xml), and codec output is not required for I2S output.

```

<path name="speaker">
    <!-- name="SmartPA Switch" value="1" /-->
</path>

```

2.4 DRIVER PORTING VERIFICATION

The above operations have completed the driver integration. Confirm that is effective through the following driver log:

I2C communication succeeded:

```

[Awinic] [6-0034] aw882xx_dai_drv_append_suffix: dai name [aw882xx-aif-6-34]
[Awinic] [6-0034] aw882xx_dai_drv_append_suffix: pstream_name name [Speaker_Playback-6-34]
[Awinic] [6-0034] aw882xx_dai_drv_append_suffix: cstream_name name [Speaker_Capture-6-34]
[Awinic] [6-0034] aw882xx_i2c_probe: dev_cnt 1

```

Sound card registration succeeded:

```
[Awinic] [6-0034] aw882xx_codec_probe: enter
[Awinic] [6-0034] aw882xx_add_codec_controls: enter
[Awinic] [6-0034] aw882xx_request_firmware: load [aw882xx_acf.bin] , file size: [2016]
[Awinic] aw_dev_parse_check_acf_by_hdr: project name [A2113]
[Awinic] aw_dev_parse_check_acf_by_hdr: custom name [Awinic]
```

PA Bin loaded successfully:

```
[Awinic] [6-0034] aw_monitor_parse_vol_data_v_0_1_1: ==parse vol end ==
[Awinic] [6-0034] aw_dev_parse_skt_type: enter
[Awinic] [6-0034] aw_dev_parse_skt_type: get dsp data prof cnt is 0
[Awinic] [6-0034] aw_dev_parse_get_vaild_prof: get vaild profile:2
[Awinic] [6-0034] aw_dev_parse_acf: parse cfg success
[Awinic] [6-0034] aw_dev_soft_reset: soft reset done
[Awinic] [6-0034] aw_dev_reg_fw_update: amppd_st=0x0000
```

Play music and PA makes sound:

```
[Awinic] [6-0034] aw_dev_set_intmask: done
[Awinic] [6-0034] aw_monitor_start: enter
[Awinic] [6-0034] aw_check_bop_status: enter
[Awinic] [6-0034] aw_check_bop_status: check done! bop status is 0
[Awinic] [6-0034] aw_device_start: done
[Awinic] [6-0034] aw882xx_start_pa: start success
```

3. TX DRIVER INTEGRATION

3.1 HAL LAYER PORTING

Modify the TX route according to the patch file in the following path;

```
AW882XX_Driver_QCOM_v1.11.0
├── ap
│   ├── Readme
│   ├── AudioReach
│   │   ├── hardware
│   │   │   └── 0001-Project-AW882XX_AudioReach_TX.patch /*Tx porting patch*/
│   │   └── porting_guide
│   │       ├── AW882XX_Android_Driver_Qcom_AudioReach_V1.0.docx
│   │       ├── AW882XX_Android_Driver_Qcom_AudioReach_V1.0.pdf
│   │       ├── AW882XX_Android_Driver_Qcom_AudioReach_V1.0_EN.docx
│   │       ├── AW882XX_Android_Driver_Qcom_AudioReach_V1.0_EN.pdf
│   │       └── Revision.docx
```

3.2 XML Configuration

Resourcemanager.xml Configuration

1) Modify the configuration of the VI device (Products without IV do not need to be configured).

```
<in-device>
  <id>PAL_DEVICE_IN_VI_FEEDBACK</id>
  <!--back_end_name is consistent with the stream_name tx of quin_mi2s in dai_link-->
  <back_end_name>MI2S-LPAIF_VA-TX-PRIMARY</back_end_name>
  <max_channels>2</max_channels>
  <channels>2</channels>
```

```
<snd_device_name>vi-feedback</snd_device_name>
<supported_bit_format>PAL_AUDIO_FMT_PCM_S32_LE</supported_bit_format>
</in-device>
```

(PS: As shown below, the string corresponding to **stream_name** is **MI2S-LPAIF_VA-TX-PRIMARY**)

```
#if 1
{
    .name = LPASS_BE_QUIN_MI2S_RX,
    .stream_name = LPASS_BE_QUIN_MI2S_RX,
    .playback_only = 1,
    .trigger = {SND_SOC_DPCM_TRIGGER_POST,
        SND_SOC_DPCM_TRIGGER_POST},
    .ops = &msm_common_be_ops,
    .ignore_suspend = 1,
    .ignore_pmdown_time = 1,
    SND_SOC_DAILINK_REG(quin_mi2s_rx),
},
{
    .name = LPASS_BE_QUIN_MI2S_TX,
    .stream_name = LPASS_BE_QUIN_MI2S_TX,
    .capture_only = 1,
    .trigger = {SND_SOC_DPCM_TRIGGER_POST,
        SND_SOC_DPCM_TRIGGER_POST},
    .ops = &msm_common_be_ops,
    .ignore_suspend = 1,
    SND_SOC_DAILINK_REG(quin_mi2s_tx),
},
#endif

#define LPASS_BE_PRI_MI2S_RX "MI2S-LPAIF-RX-PRIMARY"
#define LPASS_BE_PRI_MI2S_TX "MI2S-LPAIF-TX-PRIMARY"
#define LPASS_BE_SEC_MI2S_RX "MI2S-LPAIF_AUD-RX-PRIMARY"
#define LPASS_BE_SEC_MI2S_TX "MI2S-LPAIF_AUD-TX-PRIMARY"
#define LPASS_BE_TERT_MI2S_RX "MI2S-LPAIF-RX-TERTIARY"
#define LPASS_BE_TERT_MI2S_TX "MI2S-LPAIF-TX-TERTIARY"
#define LPASS_BE_QUAT_MI2S_RX "MI2S-LPAIF_RXTX-RX-PRIMARY"
#define LPASS_BE_QUAT_MI2S_TX "MI2S-LPAIF_RXTX-TX-PRIMARY"
#define LPASS_BE_QUIN_MI2S_RX "MI2S-LPAIF_VA-RX-PRIMARY"
#define LPASS_BE_QUIN_MI2S_TX "MI2S-LPAIF_VA-TX-PRIMARY"
#define LPASS_BE_SEN_MI2S_RX "MI2S-LPAIF_WSA-RX-PRIMARY"
#define LPASS_BE_SEN_MI2S_TX "MI2S-LPAIF_WSA-TX-PRIMARY"

#define LPASS_BE_SLIMBUS_0_RX "SLIM-DEV1-RX-0"
#define LPASS_BE_SLIMBUS_0_TX "SLIM-DEV1-TX-0"
#define LPASS_BE_SLIMBUS_1_RX "SLIM-DEV1-RX-1"
#define LPASS_BE_SLIMBUS_1_TX "SLIM-DEV1-TX-1"
#define LPASS_BE_SLIMBUS_2_RX "SLIM-DEV1-RX-2"
#define LPASS_BE_SLIMBUS_2_TX "SLIM-DEV1-TX-2"
#define LPASS_BE_SLIMBUS_3_RX "SLIM-DEV1-RX-3"
#define LPASS_BE_SLIMBUS_3_TX "SLIM-DEV1-TX-3"
```

4. CALIBRATION AND LOW VOLTAGE PROTECTION

For details, see <smartpa_cali_and_monitor> in the migration package

5. DEBUG INTERFACE

5.1 NODE

AW882XX driver will create multiple device node files with different functions. The path is sys/bus/i2c/drivers/aw882xx_smartpa/*-00xx, where * is the i2c bus number and xx is the i2c address. You can use adb to read and write nodes to debug AW882XX driver.

reg

Node name	reg
Description	read and write all register value of aw882xx
Instructions	read register value: cat reg write register value: echo reg_addr reg_data > reg (Hexadecimal operation)
Example	cat reg (get all the values of the register with read permission) echo 0x04 0x0241 > reg (write the value of 0x0241 to the register of 0x04)

rw

Node name	rw
Description	read and write single register value of aw882xx
Instructions	read register value: echo reg_addr > rw (Hexadecimal operation) cat rw write register value: echo reg_addr reg_data > rw (Hexadecimal operation)
Example	echo 0x04 > rw (read the value of the 0x04 register) cat rw echo 0x04 0x0241 > rw (write the value of 0x0241 to the register of 0x04)

driver_ver

Node name	driver_ver
Description	get driver version number
Instructions	get version number: cat driver_ver

dsp_re

Node name	dsp_re
Description	set or get the re value set in the algorithm
Instructions	get re: cat dsp_re set re: echo 7000 > dsp_re

fade_step

Node name	fade_step
Description	set step of fade in and fade out
Instructions	set step: echo step > fade_step get step: cat fade_step
Example	<div> echo 6 > fade_step cat fade_step </div> <div> (set the step to 6) (get the current step of fade in and fade out) </div>

dbg_prof

Node name	dbg_prof
Description	switch scene function control node
Instructions	switch scene function enable: echo 1 > dbg_prof switch scene function disable: echo 0 > dbg_prof get dbg_prof status: cat dbg_prof

phase_sync

Node name	phase_sync
Description	Phase synchronization function control node
Instructions	Phase synchronization function enable: echo 1 > phase_sync Phase synchronization function disable: echo 0 > phase_sync get phase_sync status: cat phase_sync

print_dbg

Node name	print_dbg
Description	i2c data printing control node
Instructions	i2c data printing enable: echo 1 > print_dbg i2c data printing disable: echo 0 > print_dbg get print_dbg status: cat print_dbg

algo_ver

Node name	algo_ver
Description	get the version of algo
Instructions	get the version of algo: cat algo_ver

monitor

Node name	monitor
Description	switch monitor function
Instructions	turn on the monitor function: echo 1 > monitor turn off the monitor function: echo 0 > monitor get monitor status: cat monitor

monitor_update

Node name	monitor_update
Description	update monitor.bin file
Instructions	update monitor.bin file: echo 1 > monitor_update

5.2 KCONTROL

Kcontrol	function	example
aw_dev_0_prof	mode selection	tinymix aw_dev_0_prof Music select Music mode tinymix aw_dev_0_prof Receiver select Receiver mode
aw_dev_0_switch	switch chip	tinymix aw_dev_0_switch Enable turn on the chip tinymix aw_dev_0_switch Disable turn off the chip
aw_dev_0_monitor	switch monitor	tinymix aw_dev_0_monitor Enable Enable monitor tinymix aw_dev_0_monitor Disable Disable monitor
aw882xx_rx_switch	switch mec	tinymix aw882xx_rx_switch 0 turn off mec tinymix aw882xx_rx_switch 1 turn on mec
aw882xx_tx_switch	switch tx	tinymix aw882xx_tx_switch 0 turn off tx tinymix aw882xx_tx_switch 1 turn on tx
aw882xx_spin_switch	set spin angle	tinymix aw882xx_spin_switch spin_90 rotate 90 degrees tinymix aw882xx_spin_switch spin_180 rotate 180 degrees
aw882xx_fadein_us	set stepping time of fade in	tinymix aw882xx_fadein_us 500 set stepping time of fade in :500
aw882xx_fadeout_us	set stepping time of fade out	tinymix aw882xx_fadeout_us 500 set stepping time of fade out: 500