AW883XX Android Driver(MTK)

版本： V1.7

时间： Sep 24, 2021

修订记录

日期	版本	描述	作者
2021-3-12	V1.0	First edition	Zhao Lei
2021-3-24	V1.1	1. Add i2c_log_en, phase_sync,spk_temp node description 2. Add re_range calibrate command description	Zhou Huidong
2021-6-1	V1.2	1. Modify MTK 5G description	Zhou Huidong
2021-7-2	V1.3	1. Modify Driver misc type set re command description	Zhou Huidong
2021-7-14	V1.4	1. Modify Hardware monitor func description	Zhou Huidong
2021-7-30	V1.5	1. Description of calibration node sample code 2. The default calibration re value range is modified to 1000-40000 mohm	Wangping
2021-8-3	V1.6	1. Modify MTK 5G description 2. Modify bin file loading description	Zhou Huidong
2021-8-24	V1.6.1	1. Modify cali file description. 2. Add spin function description.	Zhou Huidong
2021-9-24	V1.7	1. Add monitor switch control description. 2. Add dsp attr node	Zhou Huidong

目录

INFORMATION

HAL File	AudioParamOptions.xml SmartPa_ParamUnitDesc.xml
Driver File	aw883xx.c, aw883xx.h, aw_pid_2049_reg.h,aw_monitor.c, aw_monitor.h,aw_log.h,aw_init.c,aw_device.c,aw_device.h, aw_data_type.h,aw_calib.h,aw_calib.c,aw_bin_parse.c, aw_bin_parse.h,aw_spin.c,aw_spin.h
Smart PA	aw88363、aw88394、aw88395
I²C Address	0x34/0x35/0x36/0x37
ADB Debug	yes
Platform	MT6739, MT6853
Android version	android 9.0, android Q

PROJECT CONFIG

```
#add aw883xx smartpa in ProjectXXX.mk
MTK_AUDIO_SPEAKER_PATH = smartpa_awinic_aw883xx
```

AUIDO DEVICE

Add aw883xx option

Add aw883xx option in xxx/audio_param/AudioParamOptions.xml.
(Note: this file is automatically generated during overall compilation)

```
<Param name="MTK_AUDIO_SPEAKER_PATH" value="smartpa_awinic_aw883xx" />
```

Add aw883xx parameter configuration

Add aw883xx parameter configuration in device/mediatek/common/audio_param_smartpa/ SmartPa_AudioParam.xml

```
<Param path="smartpa_awinic_aw883xx" param_id="7"/>
#Add AW883XX parameter
<ParamUnit param_id="7">
    <Param name="have_dsp" value="1"/>
    <Param name="is_alsa_codec" value="1"/>
    <Param name="chip_delay_us" value="22000"/>
    <Param name="supported_rate_list"
value="8000,11025,12000,16000,22050,24000,32000,44100,48000"/>
    <Param name="spk_lib_path" value=""/>
    <Param name="spk_lib64_path" value=""/>
    <Param name="codec_ctl_name" value=""/>
    <Param name="is_apll_needed" value="1"/>
    <Param name="i2s_set_stage" value="5"/>
```



```
</ParamUnit>
```

Add the description of aw883xx

Add <Category name="smartpa_awinic_aw883xx"/> in device/mediatek/common/audio_param_smartpa/SmartPa_ParamUnitDesc.xml.

KERNEL DRIVER

AW883XX Smart PA Driver

CONFIGURATION DTS

The configuration information of aw883xx should be added in the kernel/arch/arm/boot/dts/mediatek/mt6853.dts

Sync-flag is the PA synchronization control mode. When it is configured to 1, the PA synchronization function is turned on, and when it is not configured, it is 0 by default, and the function is not turned on.

hw-monitor-delay is the interval time of the hardware monitor, the unit is ms. When not configured, the default time value is 1000ms. The DTS configuration can be modified according to actual usage requirements. According to the actual bin file parameters to decide whether to open this function.

re-min and re-max are the minimum and maximum values of the calibration Re value range respectively, and the default range is 1000-40000mohm when not configured.
mono configuration：

```
diff --git a/arch/arm/boot/dts/mediatek/mt6853.dtsi
b/arch/arm/boot/dts/mediatek/mt6853.dtsi
index f22db2e..a340a32 100644
--- a/arch/arm/boot/dts/mediatek/mt6853.dtsi
+++ b/arch/arm/boot/dts/mediatek/mt6853.dtsi
@@ -549,6 +549,8 @@
    i2c_x {      /*x means bus number*/
+          /* AWINIC AW883XX mono Smart PA */
+          aw883xx_smartpa_0: aw883xx_smartpa@34 {
+              compatible = "awinic,aw883xx_smartpa";
+              #sound-dai-cells = <0>;
+              reg = <0x34>;
+              reset-gpio = <&pio 89 0>;
+              irq-gpio = <&pio 37 0x0>;
+              sound-channel = <0>;
+              re-min = <1000>;
+              re-max= <40000>;
+              aw-cali-mode = "aw_attr";
+              status = "okay";
+      };
+      /* AWINIC AW883XX mono Smart PA End */
```

For multiple PA devices,different value of “sound-channel” associate to different devices. An example that stereo configuration as shown.
Notes：Different I2C nodes require different sound-channel attributes，set as/*0:pri_l 1:pri_r 2:sec_l 3:sec_r*/.


```
diff --git a/arch/arm/boot/dts/mediatek/mt6853.dtsi
b/arch/arm/boot/dts/mediatek/mt6853.dtsi
index f22db2e..a340a32 100644
--- a/arch/arm/boot/dts/mediatek/mt6853.dtsi
+++ b/arch/arm/boot/dts/mediatek/mt6853.dtsi
@@ -549,6 +549,8 @@
&i2c_x {        /*x means bus number*/
+       aw883xx_smartpa_0: aw883xx@34 {
+           compatible = "awinic,aw883xx";
+           #sound-dai-cells = <0>;
+           reg = <0x34>;
+           reset-gpio = <&pio 89 0x0>;
+           irq-gpio = <&pio 37 0x0>;
+           sound-channel = <0>;
+           re-min = <1000>;
+           re-max= <40000>;
+           aw-cali-mode = "aw_attr";
+           status = "okay";
+       };
+       aw883xx_smartpa_1: aw883xx@35 {
+           compatible = "awinic,aw883xx";
+           #sound-dai-cells = <0>;
+           reg = <0x35>;
+           reset-gpio = <&pio 17 0x0>;
+           irq-gpio = <&pio 19 0x0>;
+           sound-channel = <1>;
+           re-min = <1000>;
+           re-max= <40000>;
+           aw-cali-mode = "aw_attr";
+           status = "okay";
+      };
+};
```

ADD DRIVER FILES

The following driver files need to be copied to the directory of /kernel/sound/soc/codecs/aw883xx that created manually.

aw883xx.c,aw883xx.h,aw_pid_2049_reg.h,aw_monitor.c,aw_monitor.h,aw_log.h,aw_init.c,aw_device.c,aw_device.h,aw_data_type.h,aw_calib.h,aw_calib.c,aw_bin_parse.c,aw_bin_parse.h,aw_spin.c,aw_spin.h

Note: If the bin file cannot be loaded in the codec_probe function, please modify the following macros in aw883xx.h to increase the delay loading time

```
#define AW883XX_LOAD_FW_DELAY_TIME     (3000)
```

You can also modify the bin file loading retry times in aw883xx.c to meet the requirements, as follows:

```
#define AW_REQUEST_FW_RETRIES      5   /* 5 times */
```

ADD AW883XX Kconfig AND Makefile FILES

1) Add codes in kernel/sound/soc/codecs/Kconfig


```
config SND_SMARTPA_AW883XX
    tristate "SoC Audio for awinic aw883xxseries"
    depends on I2C
    help
       This option enables support for aw883xxseries Smart PA.
```

2) Add codes in kernel/sound/soc/codecs/Makefile

```
#for AWINIC AW883XXSmart PA
obj-$(CONFIG_SND_SMARTPA_AW883XX) += aw883xx/aw883xx.o
aw883xx/aw_monitor.o aw883xx/aw_bin_parse.o aw883xx/aw_device.o
aw883xx/aw_init.o aw883xx/aw_calib.o aw883xx/aw_spin.o
```

ADD AW883XX FW AND CFG FILES

1) Add the bin file's path directory in kernel/drivers/base/firmware_class.c. The path directory is determined by the system. The general path directory is "/system/vendor/firmware/" or "/system/etc/firmware/"

```
static const char * const fw_path[] = {
     fw_path_para,
     "vendor/firmware/awinic",
     "/system/etc/firmware",
     "/lib/firmware/updates/" UTS_RELEASE,
     "/lib/firmware/updates",
     "/lib/firmware/" UTS_RELEASE,
     "/lib/firmware"
};
```

2) 使用 adb 将 config 文件 push 到手机中，根据需求 push ap/config/ 路径下的 bin 文件。

```
adb push aw883xx_acf.bin vendor/firmware/awinic/
```

ASoc Machine Driver

4G Platform Configuration

Add AW883XX Dai Link configuration to mt_soc_extspk_dai in the file kernel/sound/soc/mediatek/common_int/mtk-soc-machine.c

mono configuration notes: 6-0034 is the corresponding I2C bus and address

```
struct snd_soc_dai_link_component awinic_codecs[] = {
    {
        .of_node = NULL,
        .dai_name = "aw883xx-aif-6-34",
        .name = "aw883xx_smartpa.6-0034",
    },
};
```

For multiple PA devices need to continue to add dai link information. An example that stereo configuration as shown.

```
struct snd_soc_dai_link_component awinic_codecs[] = {
```



```
    {
        .of_node = NULL,
        .dai_name = "aw883xx-aif-6-34",
        .name = "aw883xx_smartpa.6-0034",
    },
    {
        .of_node = NULL,
        .dai_name = "aw883xx-aif-6-35",
        .name = "aw883xx_smartpa.6-0035",
    },
};
```

```
{
    .name = "Ext_Speaker_Multimedia",
    .stream_name = MT_SOC_SPEAKER_STREAM_NAME,
    .cpu_dai_name = "snd-soc-dummy-dai",
    .platform_name = "snd-soc-dummy",
#ifdef CONFIG_SND_SMARTPA_AW883XX
    .num_codecs = ARRAY_SIZE(awinic_codecs),
    .codecs = awinic_codecs,
#endif
    .ops = &mt_machine_audio_ops,
},
```

5G Platform Configuration

For single PA, add the corresponding sound-dai information according to the I2C node <aw883xx_smartpa_0> configured in the front dts.

```
diff --git a/arch/arm/boot/dts/mediatek/mt6853.dtsi
b/arch/arm/boot/dts/mediatek/mt6853.dtsi
index f22db2e..a340a32 100644
--- a/arch/arm64/boot/dts/mediatek/mt6853.dts
+++ b/arch/arm/boot/dts/mediatek/mt6853.dts
@@ -2824,7 +2824,7 @@
        mtk_spk_i2s_in = <0>;
        /* mtk_spk_i2s_mck = <3>; */
        mediatek,speaker-codec {
-           sound-dai = <&speaker_amp>;
+           sound-dai = <&aw883xx_smartpa_0>;
        };
    };
```

For multiple PAs, configure DAI_LINK according to the I2C node information added by DTS. Here, two PAs are used as an example to configure DAI_LINK.

```
diff --git a/arch/arm/boot/dts/mediatek/mt6853.dtsi
b/arch/arm/boot/dts/mediatek/mt6853.dtsi
index f22db2e..a340a32 100644
--- a/arch/arm64/boot/dts/mediatek/mt6853.dts
+++ b/arch/arm/boot/dts/mediatek/mt6853.dts
@@ -2824,7 +2824,7 @@
        mtk_spk_i2s_in = <0>;
        /* mtk_spk_i2s_mck = <3>; */
        mediatek,speaker-codec {
-           sound-dai = <&speaker_amp>;
+           sound-dai = <&aw883xx_smartpa_0 &aw883xx_smartpa_1>;
```



```
        };
    };
```

Make sure RX driver porting all right, general steps are:
1) The compiler can pass;
2) I2C communication is successful;
3) Sound card registered successfully;
4) PA can output sound normally.
If nothing failed or error, the RX driver porting is completed.

SPEAKER CALI

METHODS OF CALIBRATION

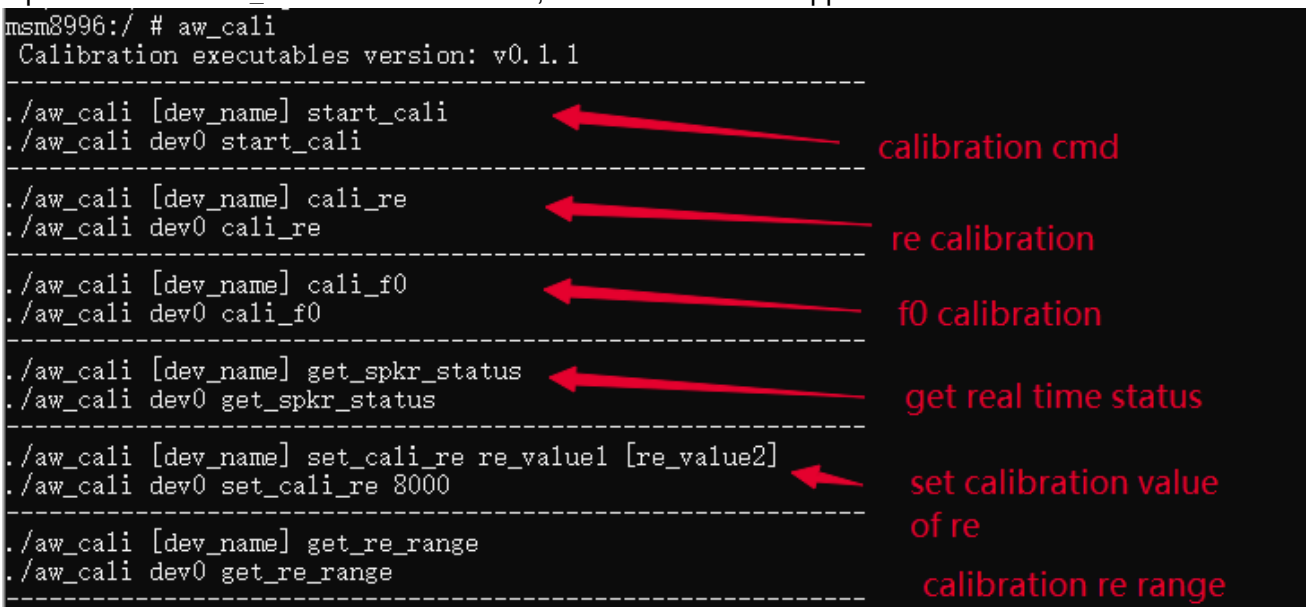
Awinic provides three methods of calibration: misc, class and attr. the method of calibration should be configured in the DTS.

1) Misc calibration

This method is always available. The calibration is directly controlled by the cali executable file.
Push aw_cali file in the system/bin directory and modify the permissions:

adb shell chmod 0777 system/bin/aw_cali

Input "adb shell aw_cali" in Shell terminal, the instructions will appear in window.



Parameter explanation (Note: [ ] means this option can be left blank)

dev_name	Calibrate single device "devx", x corresponds to the sound-channel configured in dts, When node is not configurated, the application can be automatically searched.

Calibration steps:
1) Play mute music;
2) Start the calibration, after the end, the calibration value will be output;

./system/bin/aw_cali start_cali

```
msm8996:/ # aw_cali start_cali
dev[0]cali_re = 6718
dev[1]cali_re = 6903
dev[0]cali_f0 = 946
dev[1]cali_f0 = 846
```

3）If the calibration value is within a reasonable range, the re value will be set to the cali bin file by default. When customers have customized requirements, they can set re through the following commands.

./system/bin/aw_cali set_cali_re 6718 6903

```
msm8996:/ # aw_cali set_cali_re 6718 6903
dev[0]:set cali re 6718
dev[1]:set cali re 6903
```

2) Class calibration

When set "aw-cali-mode = "aw_class"" in DTS, the class calibration method is selected. Nodes are created in the /sys/class/smartpa, calibration directories and nodes are created by the class file system. nodes as shown:

nodes	function
/sys/calss/smartpa/cali_time	1. Set calibration time 2.Show current calibration time
/sys/calss/smartpa/f0_calib	Calibrate f0
/sys/calss/smartpa/re25_calib	1. Calibrate re 2. Save calibration re value
/sys/calss/smartpa/f0_q_calib	Calibrate f0 and q
/sys/calss/smartpa/re_range	Show re_range

Customer can use fgets/fread ops to read the value of the node.

3) Attr calibration

When set "aw-cali-mode = "aw_attr"" in DTS, the attr calibration method is selected. Nodes are created in the /sys/bus/i2c/drivers/aw883xx_smartpa/*-00xx/directory ("*" mean i2c bus number; "xx" mean i2c address). nodes as shown:

nodes	function
cali_time	1. Set calibration time 2.Show current calibration time
cali_re	1. Calibrate re 2. Save calibration re value
cali_f0	Calibrate f0
cali_f0_q	Calibrate f0 and q
re_range	Show re_range

Customer calibration re steps:
1.play mute file
2.open cali_re node
3.read cali_re node

If the re value of calibration is within the re range configured by DTS, the re value will be directly stored in the bin files.

Set calibration value steps(according to the actual number of devices, here is cali-cmd of stereo

devices as an example):

1. write “dev[0]:xxxx dev[1]:xxxx” to cali_re node

Customer calibration f0 steps:
1. open cali_f0 node
2. read cali_f0 node

Calibrate f0_q steps also same as the above example. In addition, cali_time node can be used to set the Calibration time.

CALI_RE SETTING VALUE VERIFICATION

a. You can directly cat the file to check whether the re value is written into the file. The file path defaults to the definition in aw_calib.c, which can be modified according to the customer's situation:

```
#define AWINIC_CALI_FILE  "/mnt/vendor/persist/factory/audio/aw_cali.bin"
```

b. After replaying the music, cat the dsp_re node and confirm that the value in dsp is the same as the written value.

c. Restart the phone and play music, you can cat the dsp_re node to check the value in dsp and the value of re in the file.

SAVING METHOD OF CALIBRATION VALUE (EXAMPLE)

There is no memory for saving calibration values in PA, the calibration data should be saved to the platform memory after calibration；Driver read calibration re value when starting PA, and set it to MEC algorithm. The following is a reference example provided by awinic to write the calibration value into the persistent partition file (the code location is in awinic_cali.c)

```
/* customer need add function to set cali re to nv or get cali re from nv */
int aw_cali_write_re_to_nvram(int32_t cali_re, int32_t channel)
{
#ifdef AW_CALI_STORE_EXAMPLE
    return aw_cali_write_cali_re_to_file(cali_re, channel);
#else
    return -EBUSY;
#endif
}

int aw_cali_read_re_from_nvram(int32_t *cali_re, int32_t channel)
{
/*custom add, if success return value is 0 , else -1*/
#ifdef AW_CALI_STORE_EXAMPLE
    return aw_cali_get_cali_re_from_file(cali_re, channel);
#else
    return -EBUSY;
#endif
}
```


DEBUG INTERFACE

Node

AW883XX Driver will create multiple device node files with different functions. The path is sys/bus/i2c/drivers/aw883xx_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 AW883XX Driver.

reg

Node name	reg
Description	read and write all register value of aw883xx
Instruction	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 aw883xx
Instruction	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)

drv_ver

Node name	drv_ver
Description	get driver version
Instruction	get driver version：cat drv_ver

dsp_rw

Node name	dsp_rw
Description	read and write single dsp register value of aw883xx
Instruction	read dsp register： echo reg_addr > dsp_rw （Hexadecimal operation） cat dsp_rw

	write dsp register： echo reg_addr reg_data > dsp_rw （Hexadecimal operation）
Example	echo 0x8601 > dsp_rw （Read dsp register 0x8604） cat dsp_rw echo 0x8604 0x4011 > dsp_rw （Write 0x4011 to dsp register 0x8604）

dsp

Node name	dsp
Description	Get dsp firmware and dsp config info
Instruction	Get dsp firmware and dsp config： cat dsp
Example	cat dsp

fade_step

Node name	fade_step
Description	set step of fade in and fade out
Instruction	set step echo step > fade_step get step cat fade_step
Example	echo 6 > fade_step （set the step to 6） cat fade_step (get the current step of fade in and fade out)

dbg_prof

Node name	dbg_prof
Description	scene switching function control node
Instruction	switch scene function enable echo 1 > dbg_prof switch scene function disable echo 0 > dbg_prof

fade_en

Node name	fade_en
Description	fade in and fade out function control node
Instruction	fade in and fade out enable echo 1 > fade_en fade in and fade out disable echo 0 > fade_en

monitor

Node name	monitor
Description	Software Monitor function control node
Instruction	monitor enable echo 1 > monitor monitor disable echo 0 > monitor

monitor_update

Node name	monitor_update
Description	monitor config update control node
Instruction	update monitor config echo 1 > monitor_update

dsp_re

Node name	dsp_re
Description	get re value of dsp
Instruction	cat dsp_re

i2c_log_en

Node name	I2c_log_en
Description	I2C data printing control node
Instruction	i2c data printing enable echo 1 > i2c_log_en i2c data printing disable echo 0 > i2c_log_en get node cat print_dbg

phase_sync

Node name	phase_sync
Description	phase synchronization function control node
Instruction	phase synchronization function enable echo 1 > phase_sync phase synchronization function disable echo 0 > phase_sync get phase_sync status cat phase_sync

spk_temp

Node name	spk_temp
Description	display the real-time status of the speaker
Instruction	cat spk_temp

Kcontrol

"x" represents the device number

aw_dev_x_switch

PA switch
tinymix aw_dev_ x _switch Enable the device x of PA allow to turn on
tinymix aw_dev_ x _switch Disable The device x of PA is not allowed to be turned on

aw_dev_x_prof

scene switch (assuming that Music and Receive scenes are configured in the bin file)
tinymix aw_dev_ x _prof Music the device x of PA switch scene to Music
tinymix aw_dev_ x _prof Receive the device x of PA switch scene to Receive

aw883xx_fadein_us

fade in step time interval setting
tinymix aw883xx_fadein_us 500 set 500us fade in step time interval

aw883xx_fadeout_us

fade out step time interval setting
tinymix aw883xx_fadeout_us 500 set 500us fade out step time interval

aw_dev_x_monitor_switch

PA monitor 功能开关
tinymix aw_dev_ x _switch Enable 第 x 个 PA 允许 monitor 开启
tinymix aw_dev_ x _switch Disable 第 x 个 PA 不允许 monitor 开启

MATTERS OF AW883XX NEEDING ATTENTION

The Calibration Node Sample Code

Awinic provides sample codes for calibrating attr attribute nodes and class attribute nodes in \cali\example_source_code; FAE and customers can refer to.

AW883XX SOUND CHANNEL ROTATION

Rotation Scheme

The AW883XX driver provides three schemes of sound channel rotation functions, which are ADSP rotation, HAL rotation, and REG rotation. Different schemes can be selected through the configuration of dts. The rotation function will not be turned on when dts is not configured.

ADSP Rotation

ADSP rotation is suitable for the situation that the algorithm runs on the platform adsp. The rotation function is realized by calling the algorithm interface. All PA dts need to add dsp_spin configuration and to integrate the adsp communication function.

```
&i2c_x {
+       aw883xx_smartpa_0: aw883xx@34 {
+           compatible = "awinic,aw883xx";
+           #sound-dai-cells = <0>;
+           reg = <0x34>;
+           reset-gpio = <&pio 89 0x0>;
+           irq-gpio = <&pio 37 0x0>;
+           sound-channel = <0>;
+           re-min = <1000>;
+           re-max= <40000>;
+           spin-mode = "dsp_spin";
+           aw-cali-mode = "aw_attr";
```



```
+           status = "okay";
+       };
+       aw883xx_smartpa_1: aw883xx@35 {
+           compatible = "awinic,aw883xx";
+           #sound-dai-cells = <0>;
+           reg = <0x35>;
+           reset-gpio = <&pio 17 0x0>;
+           irq-gpio = <&pio 19 0x0>;
+           sound-channel = <1>;
+           re-min = <1000>;
+           re-max= <40000>;
+           spin-mode = "dsp_spin";
+           aw-cali-mode = "aw_attr";
+           status = "okay";
+      };
+};
```

How to use: Control the rotation by calling the aw_spin_switch Kcontrol. The settable values are spin_0, spin_90, spin_180, spin_270, which in turn indicate rotation of 0 degrees, rotation of 90 degrees, rotation of 180 degrees, and rotation of 270 degrees. According to the configuration in the algorithm, different angles of rotation are realized.

```
msm8996:/ # tinymix aw_spin_switch
aw_spin_switch: >spin_0 spin_90 spin_180 spin_270
```

HAL Rotation

HAL rotation is suitable for the case where the algorithm is at the HAL layer. When rotating, the algorithm interface is called to mix left and right sound, then the PA processed sound channel is selected according to the configuration of spin-data in dts, and finally the mixing is turned off to complete the rotation. All PA dts need to add reg_spin configuration.

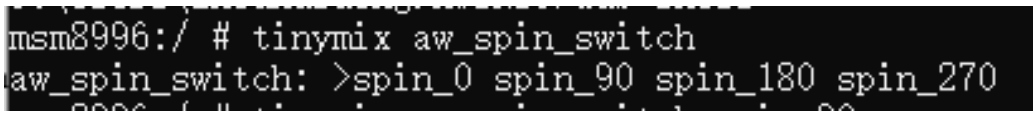
```
&i2c_x {
+       aw883xx_smartpa_0: aw883xx@34 {
+           compatible = "awinic,aw883xx";
+           #sound-dai-cells = <0>;
+           reg = <0x34>;
+           reset-gpio = <&pio 89 0x0>;
+           irq-gpio = <&pio 37 0x0>;
+           sound-channel = <0>;
+           re-min = <1000>;
+           re-max= <40000>;
+           spin-mode = "reg_spin";
+           spin-data = "l r l r";
+           aw-cali-mode = "aw_attr";
+           status = "okay";
+       };
+       aw883xx_smartpa_1: aw883xx@35 {
+           compatible = "awinic,aw883xx";
+           #sound-dai-cells = <0>;
+           reg = <0x35>;
+           reset-gpio = <&pio 17 0x0>;
+           irq-gpio = <&pio 19 0x0>;
+           sound-channel = <1>;
```



```
+           re-min = <1000>;
+           re-max= <40000>;
+           spin-mode = "reg_spin";
+           spin-data = "r l r l";
+           aw-cali-mode = "aw_attr";
+           status = "okay";
+       };
+};
```

How to use: Control the channel rotation by calling the aw_spin_switch control. The settable values are spin_0, spin_90, spin_180, spin_270, which in turn indicate rotation of 0 degrees, rotation of 90 degrees, rotation of 180 degrees, and rotation of 270 degrees.

According to the spin-data configuration in the dts, different angles of rotation are realized. "l r l r" represents the PA outputting the left, right, left, and right channels at 0 degrees, 90 degrees, 180 degrees, and 270 degrees, respectively. When using, modify the configuration according to actual needs.



REG Rotation

The REG channel rotation is suitable for algorithms running on the platform ADSP, and combined with the configuration of the PA channel register to jointly realize the rotation function. When rotating, first call the algorithm interface to mix left and right sound channel, then select the processed PA channel according to the configuration of spin-data in dts, and finally finish mixing to complete the channel rotation. All PA dts need to add the configuration of reg_mixer_spin, and need to integrate the adsp communication function.

```
&i2c_x {
+       aw883xx_smartpa_0: aw883xx@34 {
+           compatible = "awinic,aw883xx";
+           #sound-dai-cells = <0>;
+           reg = <0x34>;
+           reset-gpio = <&pio 89 0x0>;
+           irq-gpio = <&pio 37 0x0>;
+           sound-channel = <0>;
+           re-min = <1000>;
+           re-max= <40000>;
+           spin-mode = "reg_mixer_spin";
+           spin-data = "l r l r";
+           aw-cali-mode = "aw_attr";
+           status = "okay";
+       };
+       aw883xx_smartpa_1: aw883xx@35 {
+           compatible = "awinic,aw883xx";
+           #sound-dai-cells = <0>;
+           reg = <0x35>;
+           reset-gpio = <&pio 17 0x0>;
+           irq-gpio = <&pio 19 0x0>;
+           sound-channel = <1>;
+           re-min = <1000>;
+           re-max= <40000>;
+           spin-mode = "reg_mixer_spin";
+           spin-data = "r l r l";
+           aw-cali-mode = "aw_attr";
```



```
+           status = "okay";
+       };
+};
```

How to use: Control the channel rotation by calling the aw_spin_switch control. The settable values are spin_0, spin_90, spin_180, spin_270, which in turn indicate rotation of 0 degrees, rotation of 90 degrees, rotation of 180 degrees, and rotation of 270 degrees.

According to the spin-data configuration in the dts, different angles of rotation are realized. "l r l r" represents the PA outputting the left, right, left, and right channels at 0 degrees, 90 degrees, 180 degrees, and 270 degrees, respectively. When using, modify the configuration according to actual needs.