AW87XXX Android Driver(MTK)

版本： V3.4

时间： 2021 年 1 月

修订记录

日期	版本	描述	作者
2020-03-05	V1.0	驱动兼容	张鹏彪
2020-07-31	V2.0	驱动兼容	姜拴雷
2020-09-30	V3.0	加入动态降噪相关节点	姜拴雷
2020-10-28	V3.1	1.修复驱动中描述性错误 2.将动态降噪功能单独列出	王萍
2020-11-5	V3.2	增加 no dsp 和 with dsp 的描述	赵磊
2020-12-17	V3.3	删除动态降噪功能描述	王萍
2021-1-27	V3.4	修改不同产品配置的 vmax 固件名描述	赵忠波

目录

AW87XXX Android Driver

1. Information

Driver File	aw87xxx.c, aw87xxx.h, aw87339.h, aw87359.h, aw87369.h,aw87519.h,aw87549.h,aw87559.h,aw87569.h, aw87579.h,aw87xxx monitor.c, aw87xxx monitor.h, aw_bin_parse.c, aw_bin_parse.h, awinic_dsp.c,awinic_dsp.hs
Support i2c	aw87329, aw87339, aw87349, aw87359, aw87389, aw87509, aw87519, aw87529, aw87539, aw87369, aw87549, aw87559,aw87569, aw87579
I²C Address	请参考对应的 datasheet
ADB Debug	Yes
Platform	mt6739

2. Project Config

```
#add aw87xxx smartpa
CONFIG_SND_SOC_AW87XXX=y
```

3. Kernel Driver

3.1 AW87XXX Smart K PA Driver

3.1.1 dts

打开 kernel-4.4/arch/arm/boot/dts/*.dtsi 文件，根据项目类型 添加 aw87xxx 的配置
注意：由于 aw87359 和 aw87389 是没有复位引脚的，故 aw87359 和 aw87389 不需要配置 reset-gpio；

单 PA 配置：

```
&i2c_x {     /*x 表示对应的总线号*/
      /* AWINIC AW87XXX Smart K PA */
        aw87xxx_pa_58@58 {
                  compatible = "awinic,aw87xxx_pa";
                  reg = <0x58>;
                  reset-gpio = <&pio 63 0>;
                  pa-channel = < 0 >;
                  status = "okay";
         };
       /* AWINIC AW87XXX Smart K PA End */
};
```

双 PA 配置：

Pa-channel = < 0 >为左声道，pa-channel = < 1 >为右声道。

```
&i2c_x {     /*x 表示对应的总线号*/
      /* AWINIC AW87XXX Smart K PA */
        aw87xxx_pa_58@58 {
                  compatible = "awinic,aw87xxx_pa";
```



```
                    reg = <0x58>;
                    reset-gpio = <&pio 63 0>;
                    pa-channel = < 0 >;
                    status = "okay";
          };

          aw87xxx_pa_59@59 {
                    compatible = "awinic,aw87xxx_pa";
                    reg = <0x59>;
                    reset-gpio = <&pio 1 0>;
                    pa-channel = < 1 >;
                    status = "okay";
        };
        /* AWINIC AW87XXX Smart K PA End */
};
```

3.1.2 Driver

在 kernel-4.4/sound/soc/awinic 目录下添加 aw87xxx.c, aw87xxx_monitor.c, aw_bin_parse.c, awinic_dsp.c, awinic_dsp.h, aw87xxx.h, aw87xxx_monitor.h, aw_bin_parse.h, aw87579.h, aw87569.h, aw87559.h, aw87549.h, aw87519.h, aw87359.h, aw87339.h , aw87369.h 文件。注意：若平台带有 open dsp，请在 aw87xxx_monitor.h 中的增加宏定义#define AW_MTK_OPEN_DSP_PLATFORM

3.1.3 Kconfig && Makefile

1) 在 kernel-4.4/sound/soc/mediatek/Kconfig 中添加

```
config SND_SOC_AW87XXX
    tristate "SoC Audio for awinic AW87XXX Smart K PA"
    depends on I2C
    help
      This option enables support for AW87XXX Smart K PA.
```

2) 在 kernel-4.4/sound/soc/mediatek/Makefile 中添加

```
#for AWINIC AW889X Smart K PA
obj-$(CONFIG_SND_SOC_AW87XXX) += awinic/aw87xxx.o
awinic/aw87xxx_monitor.o awinic/aw_bin_parse.o awinic/awinic_dsp.o
```

3.1.4 AW87XXX Config Bin File

1) 在 kernel-4.4/drivers/base/firmware_class.c 中添加 bin 文件目录，目录由系统决定，一般目录为 /system/vendor/firmware 或/system/etc/firmware

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

2) 使用 adb 将场景 bin 文件和 vmax bin 文件 push 到手机中的/system/vendor/firmware，

注意：

场景的 bin 文件可以通过各自产品的 UI 生成。

场景 bin 文件命名的格式为：aw87xxx_pid_num_scene_channel.bin，其中 num 为芯片的 id，scene 为该 bin 文件对应的场景，channel 根据 dts 中的 pa-channel 生成，表示声道编号，如下面 push 的 bin 文件其对应的芯片 id 号为 0x59，场景为 music，channel 为 0：

```
adb push aw87xxx_pid_59_music_0.bin /system/vendor/firmware
```

vmax bin 可以有移植包下的 vmax_bin 目录中的可执行文件生成，根据需求修改 aw87xxx_vmax.txt 文件。

vmax bin 文件的命名格式为：aw87xxx_pid_num_vmax_channel.bin，其中 num 为芯片的 id，其中 channel 根据 dts 中的 pa-channel 生成，表示声道编号，如下面 push 的 bin 文件其对应的芯片 id 号为 0x59，channel 为 0：

```
adb push aw87xxx_pid_59_vmax_0.bin  /system/vendor/firmware
```

chip id 与 Smart K PA 的对应关系如下图：

Chip id	Smart K PA
0x39	aw87329, aw87339, aw87349
0x59	aw87359, aw87509, aw87519, aw87529, aw87539, aw87389
0x5A	aw87549, aw87559, aw87569, aw87579
0x69	aw87369

3.1.5 AW87XXX Config update time

在驱动源文件 aw87xxx.h 中设置了可以修改的初始化固件延时加载时间控制宏，默认驱动注册 5000ms 之后加载固件，设置如下：

```
#define AWINIC_CFG_UPDATE_DELAY
#define AWINIC_CFG_UPDATE_DELAY_TIME     (5000)
```

其中宏 AWINIC_CFG_UPDATE_DELAY 定义时则按照 AWINIC_CFG_UPDATE_DELAY_TIME 时间延时之后再加载固件。AWINIC_CFG_UPDATE_DELAY 不定义时则不延时直接加载固件。

该设置可由用户根据客户端文件系统加载时间来修改，以保证能正常发加载固件参数。

3.2 MT6357 Codec Driver

在 kernel-4.4 /sound/soc/mediate/codec/mt6357/mtk-soc-codec-6357.c 中添加

```
enum {
    AW87XXX_OFF_MODE = 0,
    AW87XXX_MUSIC_MODE = 1,
    AW87XXX_VOICE_MODE = 2,
    AW87XXX_FM_MODE = 3,
    AW87XXX_RCV_MODE = 4,
    AW87XXX_MODE_MAX = 5,
};

enum {
    AW87XXX_LEFT_CHANNEL = 0,
    AW87XXX_RIRHT_CHANNEL = 1,
};

extern unsigned char aw87xxx_show_current_mode(int32_t channel);
```

```
extern int aw87xxx_audio_scene_load(uint8_t mode, int32_t channel);

if (enable) {
    AudDrv_GPIO_EXTAMP_Select(false, 3);
    usleep_range(1 * 1000, 2 * 1000);
    #if defined(CONFIG_MTK_LEGACY)
    #elif (defined CONFIG_SND_SOC_AW87XXX)
        aw87xxx_audio_scene_load(AW87XXX_MUSIC_MODE, AW87XXX_LEFT_CHANNEL);
    #else
        AudDrv_GPIO_EXTAMP_Select(true, 3);
    #endif
        usleep_range(5 * 1000, 10 * 1000)
} else {
    #if defined(CONFIG_MTK_LEGACY)
    #elif (defined CONFIG_SND_SOC_AW87XXX)
        aw87xxx_audio_scene_load(AW87XXX_OFF_MODE, AW87XXX_LEFT_CHANNEL);
    #else
        AudDrv_GPIO_EXTAMP_Select(false, 3);
    #endif
        udelay(500);
}
```

4. Debug Interface

AW87XXX Driver 会创建不同设备节点，路径是 sys/bus/i2c/driver/aw87xxx_pa/*-00xx，在每个设备文件下各创建 hwen/reg/update/mode 4 个设备节点,其中*为 i2c bus number,xx 为 i2c address。

4.1 hwen

节点名字	hwen
功能描述	用于控制 AW87xxx 的硬件关闭
使用方法	cat hwen （获取 AW87xxx 硬件硬件状态） echo 1 > hwen （AW87xxx 硬件使能） echo 0 > hwen （AW87xxx 硬件关闭）

4.2 reg

节点名字	reg
功能描述	用于读写 aw87xxx 的所有寄存器
使用方法	读寄存器值：cat reg 写寄存器值：echo reg_addr reg_data > reg （16 进制操作）
参考例程	cat reg （获取所有可读寄存器上的值） echo 0x01 0x07 > reg （向 0x01 寄存器写入 0x07 的值）

4.3 update

节点名字	update
功能描述	用于更新 AW87XXX 参数配置文件

使用方法	echo 1 > update	（AW87XXX 参数更新）

4.4 mode

节点名字	mode	
功能描述	用于配置 AW87XXX 的工作模式	
使用方法	cat mode	（获取不同模式对应的数字）
	echo 0 > mode	（AW87XXX 关闭）
	echo 1 > mode	（AW87XXX 使能，工作在 music 模式）
	echo 2 > mode	（AW87XXX 使能，工作在 voice 模式）
	echo 3 > mode	（AW87XXX 使能，工作在 fm 模式）
	echo 4 > mode	（AW87XXX 使能，工作在 rcv 模式）

5. 低电量保护算法

低电量保护算法根据平台是否含有 open dsp，设计了两种方案: 1. No dsp 低电量保护算法；2.with dsp 低电量保护算法。

5.1 No dsp 低电量保护算法

若平台不含有 open dsp， 请参考算法移植文档《awinic_skt_mtk_porting.pdf》，进行移植。

注意：No dsp 低电量保护算法需要对 vmax 节点进行读操作，因此在保护算法加载之前应该 vmax 节点对用户有可读权限（可使用命令：chmod 777 /sys/bus/i2c/driver/aw87xxx_pa/*-00xx/vmax）。

5.2 With dsp 低电量保护算法

5.2.1 Dts 配置

在 aw87xxx.dtsi 文件中添加的配置。Dts 中可以配置是否在 pa 音频流启动时自动开启低电量保护。如果无需在 pa 音频流启动时自动开启低电量保护，则 dts 配置与 3.1.1 一致。如果需在 pa 音频流启动时自动开启低电量保护，dts 配置如下：

需添加标志位 monitor-flag，表示 pa 音频流启动时自动开启低电量保护；添加 monitor-timer-val 表示设置间隔多少毫秒读取一次电量；添加 monitor-timer-count-max 表示累加多少次电量值后求平均值向 dsp 设置 vmax。

单 PA 配置：

```
&i2c_x {     /*x 表示对应的总线号*/
      /* AWINIC AW87XXX Smart K PA */
        aw87xxx_pa_58@58 {
                    compatible = "awinic,aw87xxx_pa";
                    reg = <0x58>;
                    reset-gpio = <&pio 63 0>;
                    pa-channel = < 0 >;
        +           monitor-flag = <1>;
        +           monitor-timer-val = <3000>;
        +           monitor-timer-count-max = <5>;
                    status = "okay";
         };
```



```
        /* AWINIC AW87XXX Smart K PA End */
};
```

立体音配置：

```
&i2c_x {     /*x 表示对应的总线号*/
       /* AWINIC AW87XXX Smart K PA */
         aw87xxx_pa_58@58 {
                    compatible = "awinic,aw87xxx_pa";
                    reg = <0x58>;
                    reset-gpio = <&pio 63 0>;
                    pa-channel = < 0 >;
         +          monitor-flag = <1>;
         +          monitor-timer-val = <3000>;
         +          monitor-timer-count-max = <5>;
                    status = "okay";
          };

         aw87xxx_pa_59@59 {
                    compatible = "awinic,aw87xxx_pa";
                    reg = <0x59>;
                    reset-gpio = <&pio 1 0>;
                    pa-channel = < 1 >;
         +          monitor-flag = <1>;
         +          monitor-timer-val = <3000>;
         +          monitor-timer-count-max = <5>;
                    status = "okay";
         };
        /* AWINIC AW87XXX Smart K PA End */
};
```

5.2.2 With dsps 低电量保护算法 Debug Interface

AW87XXX Driver 会创建不同设备节点，路径是 sys/bus/i2c/driver/aw87xxx_pa/*-00xx，在每个设备文件下各创建 vmax/vbat/monitor/vmax_time 4 个设备节点，其中*为 i2c bus number，xx 为 i2c address。

5.2.2.1 vmax

节点名字	vmax
功能描述	用于设置和获取 vmax 值（获取的 vmax 为算法中计算出的 vmax 值，与设置的值不相同）
使用方法	cat vmax （获取当前 vmax 的值） echo N > vmax （发送计算后的 vmax 值）
参考例程	echo 0xfff95f7e > vmax （将 0xfff95f7e 值发送给 vmax）

5.2.2.2 vbat

节点名字	vbat
功能描述	用于设置和获取当前用户输入的电量值
使用方法	cat vbat （获取当前用户输入的 vbat 值） echo capacity > vbat （设置 vbat 电量值）
参考例程	echo 50 > vbat （设置当前电量为 50%）

5.2.2.3 monitor

节点名字	monitor
功能描述	用于设置 aw87xxx 的保护功能
使用方法	cat monitor （获取当前电量的保护状态） echo 0 > monitor （aw87xxx 关闭保护） echo 1 > monitor （aw87xxx 开启保护）

5.2.2.4 vmax_time

节点名字	vmax_time
功能描述	用于设置 vmax 的上报时间
使用方法	cat vmax_time （获取 vmax_time 的值） echo N > vmax _time （设置 vmax_time 上报时间）

