AW8624 Android Driver

INFORMATION

Driver File	aw8624.c, aw8624.h, aw8624_reg.h
Haptic Driver	aw8624
I²C Address	0x5a/0x5b
ADB Debug	Yes
Platform	mtk6739

PROJECT CONFIG

```
#add aw8624 haptic
CONFIG_AW8624_HAPTIC =y
```

KERNEL DRIVER

AW8624 Haptic Driver

1. dts config

打开 kernel/arch/arm/boot/dts/ *.dts 文件，添加 aw8624 的配置

```
/* AWINIC AW8624 Haptic */
&i2c3 {
    aw8624_haptic@5A {
        compatible = "awinic,aw8624_haptic";
        reg = <0x5A>;
        reset-gpio = <&tlm 63 0>;
        irq-gpio = <&tlm 4 0>;
        vib_f0_pre = < 2350 >;    /*根据马达型号定义 1700 2350 2600*/
        vib_f0_cali_percen = < 7 >;
        vib_cont_drv_lev = < 125 >;
        vib_cont_drv_lvl_ov = < 155 >;
        vib_cont_td = < 0xF06C >;
        vib_cont_zc_thr = < 0x08F8 >;
        vib_cont_num_brk = < 3 >;
        vib_f0_coeff = < 260 >; /*Don't modify it*/
        vib_duration_time = < 15 60 0 0 0 >;
        vib_brake_ram_config = < 1 1 90 60 20 3 1 3    1 1 90 60 30 5 1 3    0 0 50
40 25 0 5 3 >;
        vib_brake_cont_config = < 1 1 90 42 20 5 2 2 0 0 0  0  0  0 0 0 0 0 0 0  0  0
0 0 0 >;
        vib_f0_trace_parameter = < 9 3 1 31 >;
        vib_bemf_config = < 0 8 3 0xf8 >;
        vib_sw_brake = < 0x2c >;
        vib_wavseq = < 0 1 1 2 0 0 0 0 0 0 0 0 0 0 0 0 >; /*wavseq1-seq8 0x07-
0x0E*/
```



```
        vib_wavloop = < 0 6 1 15 0 0 0 0 0 0>;/*loop1-mainloog 0x0f-0x13*/
        vib_td_brake = < 42720 42720 42720 >; /*level0  level1  level2*/
        vib_tset = < 0x11 >;
        vib_func_parameter1 = < 1000 >;  /*开机时马达震动时间，单位 ms*/
        status = "okay";
    };
};
/* AWINIC AW8624 Haptic End */
```

2. add driver file

在 kernel/driver/misc/目录下添加 aw8624_haptic 驱动文件夹(aw8624.c, aw8624.h, aw8624_reg.h)

3. update Kconfig and Makefile

1) 在 kernel/driver/misc/Kconfig 中添加

```
source "drivers/misc/aw8624_haptic/Kconfig"
```

2) 在 kernel/driver/misc/Makefile 中添加

```
obj-$(CONFIG_AW8624_HAPTIC) += aw8624_haptic/
```

4. add aw8624 haptic waveform file

1) 在 kernel/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 ram 文件 push 到手机中

```
adb push aw8624_haptic.bin /system/vendor/firmware/
adb push aw8624_rtp.bin /system/vendor/firmware/
```

DEBUG INTERFACE

adb

I2c device

AW8624 Driver 会创建 reg/ram 2 个设备节点文件，路径是 sys/bus/i2c/driver/aw8624_haptic/*-00xx，其中*为 i2c bus number，xx 为 i2c address。
可以使用 adb 配置 reg/ram 参数，调试 AW8624 震动效果。

reg

用于读写 AW8624 的所有寄存器。

节点使用：
读寄存器值：cat reg
写寄存器值：echo addr data > reg （16 进制操作）
参考例程：
cat reg
echo “0x04 0x04” > reg （向 0x04 寄存器写值 0x04）

ram

用于读写 ram 和 rtp 模式的 bin 文件。
节点使用：
读取 ram：cat ram
更新 ram：echo 1 > ram （更新 aw8624_haptic.bin 和 aw8624_rtp.bin）

vibrator device

AW8624 Driver 会创建设备节点文件，路径是 sys/class/timed_output/virbrator_aw8624 或者 sys/class/leds/virbrator_aw8624，virbrator_aw8624 是驱动中 virbrator 的设备名，android 默认的为 virbrator。可以使用 adb 配置参数，调试 AW8624 震动效果。

短震

1. seq
用于 ram 模式下的波形选择，seq 设置为 0 时，停止播放。
节点使用：
配置波形：
echo wavseq seq > seq （wavseq: wavform addr，波形地址，范围：0x00-0x07 ）
（seq: wavform sequency number，波形号，范围：0x00-0x0F）
读取波形：
cat seq

示例：
echo 0x00 0x01 > seq （选择 seq1 播放）
echo 0x01 0x00 > seq （防止下一个寄存器不为 0）

echo 0x00 0x02 > seq （选择 seq2 播放）
echo 0x01 0x00 > seq

echo 0x00 0x01 > seq （选择 seq1+seq2 播放）
echo 0x01 0x02 > seq
echo 0x02 0x00 > seq

echo 0x00 0x04 > seq （选择 seq4+seq3+seq2+seq1 播放）
echo 0x01 0x03 > seq
echo 0x02 0x02 > seq
echo 0x03 0x01 > seq
echo 0x04 0x00 > seq

cat seq （读取 seq1-8）

2. loop
用于 ram 模式下的波形次数选择，配置的的范围为 0x00-0x0E（次数为 1-15），0x0F（一直循环）
节点使用：
配置循环：
echo wavseq loop > loop （wavseq: wavform addr，波形地址，范围：0x00-0x07）
（loop: wavform loop number，波形循环次数，范围：0x00-0x0F）
读取循环：
cat loop

示例：
echo 0x00 0x00 > loop (wavseq1 循环 1 次)
echo 0x00 0x01 > loop (wavseq1 循环 2 次)
echo 0x01 0x01 > loop (wavseq2 循环 2 次)
echo 0x01 0x0f > loop (wavseq2 一直循环)

cat loop (读取 wavseq 的 loop)

3. enable/brightness
用于 ram 模式下的震动开始和停止
节点使用：
echo 1 > enable （震动使能）
echo 0 > enable （震动停止）
或
echo 1 > brightness （震动使能）
echo 0 > brightness （震动停止）

长震

长震可由 RAM 循环或者 CONT 模式实现，通过节点 activate_mode 选择。

activate_mode	模式	说明
0	RAM 循环模式	需要配置 index，用于选择长震的波形，一般为正弦波
1	CONT 模式	无需配置 index，波形有芯片内部产生 波形的参数有 cont_td/cont_drv/cont_num_brk/cont_zc_thr

1. duration
用于长震的时间选择，单位为 ms
节点使用：
echo 2000 > duration （长震 2s）
cat duration （显示剩余时间）

2. index
用于 RAM 长震模式的波形选择
节点使用：
echo 1 > index （选择波形 1）

3. activate

用于长震的启动和停止。（请配置 gain/drv_lvl，确保输出电压(gain/128)*(drv_lvl/128)小于马达标准电压）
节点使用：
echo 1 > activate （长震开始）
echo 0 > activate （长震停止）

RTP 模式

- rtp

用于 rtp 模式的震动启动和停止
节点使用：
echo 1 > rtp （选择 1 号 RTP 文件，震动使能）
echo 2 > rtp （选择 2 号 RTP 文件，震动使能）
echo 0 > rtp （震动停止）

Cont 模式

- cont

实时追踪模式，在长震时，可以实时追踪 F0
节点使用：
echo 1 > cont （开启 Cont 模式，实时追踪 F0）
echo 0 > cont （关闭 Cont 模式）

- cont_td

cont 模式脉冲间隔配置，间隔时间为（2*cont_td）
节点使用：
echo 0x005d > cont_td

- cont_drv

cont 模式驱动波形电压配置，包含正常驱动电压 drv_lvl 和启动、刹车的 drv_lvl_ov。
节点使用：
echo drv_lvl drv_lvl_ov > cont_drv
echo 80 127 > cont_drv

- cont_num_brk

cont 模式刹车脉冲周期数，范围：0~7
节点使用：
echo 3 > cont_num_brk

- cont_zc_thr

cont 模式过零阈值
节点使用：
echo 0x009a > cont_zc_thr

F0 校准

- cali

用于 F0 的检测和校准
节点使用：

echo 1 > cali （检测马达 F0，然后配置 LRA_TRIM 寄存器，与马达 F0 匹配，实现 F0 校准）
cat cali （读取校准的 F0*10）

➢ f0
用于获取马达的 f0
节点使用：
cat f0

驱动配置

➢ vmax
设置 boost voltage，范围：0x00-0x1F
节点使用：
echo 0x11 > vmax （设置 boost voltage 为 8.5V）
cat vmax （读取 boost voltage）

➢ gain
设置 gain，范围：0x00-0xFF
节点使用：
echo 0x80 > gain （设置 gain 为 0x80）
cat gain （读取 gain）

➢ auto_boost
设置 trig 触发时，是否需要自动启动 boost
节点使用：
echo 1 > auto_boost （启动自动 boost）
echo 0 > auto_boost （关闭自动 boost，保持之前的配置）

➢ trig
设置 trig 触发配置，配置参数:

参数编号	参数名	参数意义	参数范围
1	trig_num	Trig 引脚	1/2/3
2	enable	使能	1/0
3	default_levle	默认电平	1/0
4	dual_edge	支持双边沿	1/0
5	frist_seq	第一边沿脉冲波形	1~n
6	second_seq	第二边沿脉冲波形	1~n

节点使用：
echo 1 1 1 1 1 2 > trig （trig1 使能，默认高电平，支持双边沿，下降沿波形为 1 号，上升沿波形为 2 号）
echo 2 1 0 1 1 1 > trig （trig2 使能，默认低电平，支持双边沿，上降沿波形为 1 号，下降沿波形为 1 号）
echo 3 0 0 0 0 0 > trig （trig3 关掉）

➢ ram_vbat_comp
设置 ram 长震模式时，是否添加软件补偿功能
节点使用：
echo 1 > ram_vbat_comp （启动软件补偿）
echo 0 > ram_vbat_comp （关闭软件补偿）

➢ register
用于读写 AW8624 的所有寄存器。
节点使用：
读寄存器值：cat register
写寄存器值：echo addr data > register （16 进制操作）

马达保护

➢ vbat_monitor
电源检测功能，可以获取当前的电源电压，单位：mV
节点使用：
cat vbat_monitor （读取电源电压）

➢ lra_resistance
马达阻抗检测，可以获取马达的阻抗，用于短路开路检测，单位：mΩ
节点使用：
cat lra_resistance （读取马达阻抗）

➢ prctmode
马达保护功能，在输入信号为长时间直流时，可以关闭芯片输出，保护马达，同时马达保护的相关寄存器参数可以修改,寄存器为 0x2D,0x3E,0x3F。
节点使用：
echo 1 1 > prctmode （打开马达保护）
echo 0 0 > prctmode （关闭马达保护）
echo 0x2D val > prctmode
echo 0x3E val > prctmode
echo 0x3F val > prctmode

随音震动

➢ haptic_audio
用于 hal 层控制马达驱动芯片的接口

➢ haptic_audio_time
用于设置定时器的启动延时和循环时间，单位：us
节点使用：
echo delay timer > haptic_audio_time
echo 12000 20833 > haptic_audio_time （设置 timer 启动延时为 12ms，timer 的循环时间为 20.833ms）

RAM 更新

➢ ram_update
更新 ram，读取 ram 模式和 rtp 模式的 bin 文件配置
节点使用：
echo 1 > ram_update （更新 aw8624_haptic.bin 和 aw8624_rtp.bin）

