目录

[Android Audio 1](#_Toc278891006)

[ALSA 适配 4](#_Toc278891007)

[Audio，MMC，TTY驱动 12](#_Toc278891008)

# Android Audio

音频系统java相关的代码在目录下frameworks/base/media/java/android/media/

主要有如下文件：

AudioManager.java

为上层应用提供了声音设置管理接口.

AudioService.java

音频设置服务, 它在SystemServer中启动，为所有的音频相关的设置提供服务。

在AudioService 中定义了了一个AudioSystemThread 的类，用来监控音频控制

相关的信号，当有请求时，它会通过调用AudioSystem 的接口实现音频的控制，

这里的消息处理是异步的。此外在AudioService还抽象出了一套发送音频控制信

号的接口为AudioManager提供支持。

AudioSystem.java:

提供了音频系统的基本类型定义，以及基本操作的接口。它对应于

frameworks/base/core/jni/android\_media\_AudioSystem.cpp

Ringtone.java

RingtoneManager.java

为铃声、闹钟等提醒提供了快速的播放以及管理接口。

AudioTrack.java

直接为PCM数据提供支持,对应于frameworks/base/core/jni/android\_media\_AudioTrack.cpp

SoundPool.java

提供了为引用播放声音的接口，在加载文件等方面做了优化。

ToneGenerator.java

提供了播放DTMF tones 的支持，

如电话的拨号音,对应于直接为PCM数据提供支持,对应于frameworks/base/core/jni/android\_media\_ToneGenerator.cpp

AudioRecord.java

这个是音频系统对外的录制接口，对应于：frameworks/base/core/jni/android\_media\_AudioRecord.cpp

Android音频简介

<http://my.unix-center.net/~Simon_fu/?p=582>

最近移植Android，当Android能够在设备上面运行之后，首先想到的是让音频设备跑起来。“没有声音，再好的戏也出不来”。本文简单介绍一下Android音频适配层。

这个世界音频设备千变万化，Android也不可能为每种设备都提供支持。Android定义了一个框架，这个框架来适配底层的音频设备。该适配层的定义位于：

hardware/libhardware\_legacy/include/hardware\_legacy/AudioHardwareInterface.h

要想视频底层的音频设备必须要继承该文件中定义的AudioStreamOut，AudioStreamIn，AudioHardwareInterface等类，并实现createAudioHardware函数。

下面我们看一下Android创建音频设备的代码，代码位于：

frameworks/base/libs/audioflinger/AudioHardwareInterface.cpp

该文件有如下代码：

AudioHardwareInterface\* AudioHardwareInterface::create()

{

/\*

\* FIXME: This code needs to instantiate the correct audio device

\* interface. For now - we use compile-time switches.

\*/

AudioHardwareInterface\* hw = 0;

char value[PROPERTY\_VALUE\_MAX];

#ifdef GENERIC\_AUDIO

hw = new AudioHardwareGeneric();

#else

// if running in emulation - use the emulator driver

if (property\_get("ro.kernel.qemu", value, 0)) {

LOGD("Running in emulation - using generic audio driver");

hw = new AudioHardwareGeneric();

}

else {

LOGV("Creating Vendor Specific AudioHardware");

hw = createAudioHardware();

}

#endif

if (hw->initCheck() != NO\_ERROR) {

LOGW("Using stubbed audio hardware. No sound will be produced.");

delete hw;

hw = new AudioHardwareStub();

}

#ifdef WITH\_A2DP

hw = new A2dpAudioInterface(hw);

#endif

#ifdef ENABLE\_AUDIO\_DUMP

// This code adds a record of buffers in a file to write calls made by AudioFlinger.

// It replaces the current AudioHardwareInterface object by an intermediate one which

// will record buffers in a file (after sending them to hardware) for testing purpose.

// This feature is enabled by defining symbol ENABLE\_AUDIO\_DUMP.

// The output file is set with setParameters("test\_cmd\_file\_name=<name>"). Pause are not recorded in the file.

LOGV("opening PCM dump interface");

hw = new AudioDumpInterface(hw); // replace interface

#endif

return hw;

}

从代码中我们可以看出如果定义了GENERIC\_AUDIO的宏，则会创建AudioHardwareGeneric，如果是模拟器的话，AudioHardwareGeneric会不能初始化，进而创建AudioHardwareStub。这两个类都是Audio设备的适配层，是Android默认提供的。模拟器都是用AudioHardwareStub，不会有声音输出。设备都是用AudioHardwareGeneric，因为默认GENERIC\_AUDIO是设置的。

一般我们只关心AudioHardwareGeneric实现，谁会去给模拟器去调试声音呢，反正我没这个闲心。首先说明一下这个音频适配层是Android自带的，可以保证你的音频设备正常运行，但是不能发挥设备的最佳性能。通过后面的描述你将会了解。AudioHardwareGeneric的定义位于：

frameworks/base/libs/audioflinger/AudioHardwareGeneric.cpp

查看源码你会发现这个适配层需要实现设备/dev/eac，并且该设备只输出44.1khz采样率的音频数据给/dev/eac设备，如果不是44.1khz的采样率的数据，AudioHardwareGeneric会经过Resample过程把它转换成44.1kHZ的音频数据，然后再输出给音频设备。44.1kHZ音频数据是最普遍的音频采样率，大部分Mp3都是以这个采样率压缩的，所以选择这个采样率做为默认采样率还是有一定的合理性的。AudioHardwareGeneric是软件实现Resample过程是，效率会比较低。很多音频设备支持不同采样率的数据，可以理解成硬件实现Resample过程。

通过上面的描述我们可以知道这个通用音频适配层只是让你的设备可以用而已，不能发挥设备的性能优势，如果你的设备对音频质量有更高的要求，必须要自己实现音频适配层。谷歌只能保证你的音频可以播放，但是不能保证效率（他也没有办法保证效率）。

本文只是粗略的对Android音频系统进行了简单介绍。如果有错误之处请不吝指教。

# ALSA 适配

externel/libaudio

external/alsa-utils

external/alsa-lib

./external/alsa-lib/include/sound/asound\_fm.h

system/core/init/devices.c 涉及了 SLSI\_S3C6410，

external/libaudio/Android.mk: LOCAL\_CFLAGS += -DSLSI\_S3C6410

external/alsa-utils/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

external/alsa-utils/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

external/alsa-utils/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

external/alsa-lib/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

external/opencore/android/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

external/opencore/codecs\_v2/omx/omx\_h264enc/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

external/opencore/codecs\_v2/omx/omx\_m4venc/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

external/opencore/codecs\_v2/omx/omx\_h264/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

external/opencore/codecs\_v2/omx/omx\_m4v/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

system/core/vold/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

system/core/init/Android.mk:LOCAL\_CFLAGS += -DSLSI\_S3C6410

gettext.h \_() 宏

> > BOARD\_USES\_GENERIC\_AUDIO := false

> > BOARD\_USES\_ALSA\_AUDIO := true

> > BUILD\_WITH\_ALSA\_UTILS := true

build/target/board/generic/BoardConfig.mk:BOARD\_USES\_GENERIC\_AUDIO := true

build/target/board/sim/BoardConfig.mk:BOARD\_USES\_GENERIC\_AUDIO := true

vendor/sec/smdk6410/BoardConfig.mk:#BOARD\_USES\_GENERIC\_AUDIO := true

external/alsa-utils/Android.mk

make:进入目录'/home/leo/SMDK6410\_Eclair/smdk6410/android'

target Non-prelinked: alsa\_amixer (out/target/product/smdk6410/symbols/system/bin/alsa\_amixer)

Install: out/target/product/smdk6410/system/bin/alsa\_amixer

target Non-prelinked: alsa\_aplay (out/target/product/smdk6410/symbols/system/bin/alsa\_aplay)

Install: out/target/product/smdk6410/system/bin/alsa\_aplay

target Non-prelinked: alsa\_ctl (out/target/product/smdk6410/symbols/system/bin/alsa\_ctl)

Install: out/target/product/smdk6410/system/bin/alsa\_ctl

make:离开目录“/home/leo/SMDK6410\_Eclair/smdk6410/android”

android$ TARGET\_PRODUCT=sec\_smdk6410 mmm external/alsa-utils/

alsa\_aplay, alsa\_ctl, alsa\_amixer

# alsa\_aplay Noise.wav

aplay: main:594: audio open error: Device or resource busy

The error

"aplay: main:590: audio open error: Device or resource busy"

is because the alsa is being used by media server, hence you need to

stop it using

#stop media

Then run

#alsa\_aplay

#start media 恢复 media, AudioFlinger 需要他。

# alsa\_aplay -L 2>&1

default:CARD=smdk6410

smdk6410,

Default Audio Device

null

Discard all samples (playback) or generate zero samples (capture)

主机上的信息：

tftpboot$ aplay -L 2>&1

pulse

Playback/recording through the PulseAudio sound server

front:CARD=SB,DEV=0

HDA ATI SB, ALC889A Analog

Front speakers

surround40:CARD=SB,DEV=0

HDA ATI SB, ALC889A Analog

4.0 Surround output to Front and Rear speakers

surround41:CARD=SB,DEV=0

HDA ATI SB, ALC889A Analog

4.1 Surround output to Front, Rear and Subwoofer speakers

surround50:CARD=SB,DEV=0

HDA ATI SB, ALC889A Analog

5.0 Surround output to Front, Center and Rear speakers

surround51:CARD=SB,DEV=0

HDA ATI SB, ALC889A Analog

5.1 Surround output to Front, Center, Rear and Subwoofer speakers

surround71:CARD=SB,DEV=0

HDA ATI SB, ALC889A Analog

7.1 Surround output to Front, Center, Side, Rear and Woofer speakers

iec958:CARD=SB,DEV=0

HDA ATI SB, ALC889A Digital

IEC958 (S/PDIF) Digital Audio Output

hdmi:CARD=HDMI

HDA ATI HDMI, ATI HDMI

HDMI Audio Output

tftpboot$ aplay -L 2>&1 | egrep surround

surround40:CARD=SB,DEV=0

surround41:CARD=SB,DEV=0

surround50:CARD=SB,DEV=0

surround51:CARD=SB,DEV=0

surround71:CARD=SB,DEV=0

#alsa\_ctl store 0

#alsa\_ctl –help

生成的文件就是/etc/asound.state

# cat /etc/asound.state

android$ find . -name asound.conf

./vendor/sec/smdk6410/asound.conf

./out/target/product/smdk6410/system/etc/asound.conf

CyanogenMod 源码的 device 、 vendor 目录下都没有asound.conf。

ALSA的一些conf文件放在external/alsa-lib/src/conf目录下， 比如 external/alsa-lib/src/conf/cards/HDA-Intel.conf，我的开发主机的配置。

$ cat /proc/asound/cards

0 [SB ]: HDA-Intel - HDA ATI SB

HDA ATI SB at 0xfe024000 irq 16

1 [HDMI ]: HDA-Intel - HDA ATI HDMI

HDA ATI HDMI at 0xfdffc000 irq 19

$ adb shell

# cat /proc/asound/cards

0 [smdk6410 ]: WM8580 - smdk6410

smdk6410 (WM8580)

/etc/asound.conf

/usr/share/alsa/alsa.conf

~/.asoundrc

type route; # 虚拟设备，环绕立体声

type hw

pcm.card-f

{

@args [ PCM ]

@args.PCM { type string; }

type route;

slave.pcm $PCM;

slave.channels 2;

ttable.0.0 1.0;

ttable.1.1 1.0;

}

# ls -l /dev/snd

crw-rw---- system audio 14, 0 2030-10-15 19:22 mixer

crw-rw-r-- root audio 116, 5 2030-10-15 19:22 controlC0

crw-rw-r-- root audio 14, 4 2030-10-15 19:22 audio

crw-rw---- system audio 14, 3 2030-10-15 19:22 dsp

crw-rw-r-- root audio 116, 4 2030-10-15 19:22 pcmC0D0p

crw-rw-r-- root audio 116, 3 2030-10-15 19:22 seq

crw-rw-r-- root audio 14, 8 2030-10-15 19:22 sequencer2

crw-rw-r-- root audio 14, 1 2030-10-15 19:22 sequencer

crw-rw-r-- root audio 116, 2 2030-10-15 19:22 timer

# ls -l /proc/asound

lrwxrwxrwx root root 2030-10-16 00:34 smdk6410 -> card0

dr-xr-xr-x root root 2030-10-16 00:34 card0

-r--r--r-- root root 0 2030-10-16 00:34 pcm

-r--r--r-- root root 0 2030-10-16 00:34 timers

-r--r--r-- root root 0 2030-10-16 00:34 cards

-r--r--r-- root root 0 2030-10-16 00:34 devices

-r--r--r-- root root 0 2030-10-16 00:34 version

dr-xr-xr-x root root 2030-10-16 00:34 seq

dr-xr-xr-x root root 2030-10-16 00:34 oss

tftpboot$ adb shell cat /proc/asound/devices

2: : timer

3: : sequencer

4: [ 0- 0]: digital audio playback

5: [ 0] : control

在AudioHardwareALSA.cpp中const char \*elementName = snd\_mixer\_selem\_id\_get\_name(sid);后加上输出LOG

LOGD ("Travis Mixer: route '%s'.", elementName);

android$ . build/envsetup.sh

android$ TARGET\_PRODUCT=sec\_smdk6410 mmm external/libaudio/

android$adb remount

android$adb push out/target/product/smdk6410/system/lib/libaudio.so /system/lib

android$reboot

D/AndroidRuntime( 1863): >>>>>>>>>>>>>> AndroidRuntime START <<<<<<<<<<<<<<

D/AndroidRuntime( 1863): CheckJNI is ON

I/ ( 1864): ServiceManager: 0xad08

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'Line'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'Mic'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'Playback Target'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'ADC High-Pass Filter'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'ADC Mute'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC ZC'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC1'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC1 Deemphasis'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC1 Invert'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC2'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC2 Deemphasis'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC2 Invert'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC3'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC3 Deemphasis'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC3 Invert'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'Off'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'Line'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'Mic'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'Playback Target'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'ADC High-Pass Filter'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'ADC Mute'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC ZC'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC1'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC1 Deemphasis'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC1 Invert'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC2'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC2 Deemphasis'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC2 Invert'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC3'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC3 Deemphasis'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'DAC3 Invert'.

D/AudioHardwareALSA( 1864): !!!!!Travis Mixer: route 'Off'.

D/AudioHardwareInterface( 1864): setMode(NORMAL)

下面是使用了WM9713驱动出现的问题，导致Android系统不能启动，而使用WM8580驱动是可以的。但是察看板子，发现使用的确实是WM9713。

E/AudioHardwareALSA( 1885): Unable to attach mixer to device AndroidPlayback: No such file or directory

E/AudioHardwareALSA( 1885): Unable to attach mixer to device default: No such file or directory

E/AudioHardwareALSA( 1885): Unable to attach mixer to device AndroidRecord: No such file or directory

E/AudioHardwareALSA( 1885): Unable to attach mixer to device default: No such file or directory

[ 2.640597] Failed to reset WM9713: AC97 link error

[ 2.645645] Entered s3c\_ac97\_remove

... ...

[ 2.654363] ALSA device list:

[ 2.656884] No soundcards found.

CC sound/soc/codecs/wm9713.o

CC sound/soc/s3c/s3c-ac97.o

LD sound/soc/codecs/snd-soc-wm9713.o

LD sound/soc/codecs/built-in.o

CC sound/soc/s3c/smdk6410\_wm9713.o

LD sound/soc/s3c/snd-soc-smdk6410-wm9713.o

CC sound/soc/s3c/s3c-i2s.o

CC sound/soc/s3c/smdk6410\_wm8580.o

LD sound/soc/s3c/snd-soc-smdk6410-wm8580.o

kernel$ find . | xargs grep --color -n -e "DAC1 Deemphasis"

./sound/soc/codecs/wm8580.c:336:SOC\_SINGLE("DAC1 Deemphasis Switch", WM8580\_DAC\_CONTROL3, 0, 1, 0),

android$ find external vendor/ system/ | xargs grep "alsa.mixer.playback.speaker"

external/libaudio/AudioHardwareALSA.cpp: {ROUTE\_SPEAKER, "alsa.mixer.playback.speaker", "Speaker", NULL},

Android 系统属性察看：

tftpboot$ adb shell getprop

[ro.secure]: [0]

[ro.allow.mock.location]: [1]

[ro.debuggable]: [1]

[persist.service.adb.enable]: [1]

[ro.factorytest]: [0]

[ro.serialno]: []

[ro.bootmode]: [unknown]

[ro.baseband]: [unknown]

[ro.carrier]: [unknown]

[ro.bootloader]: [unknown]

[ro.hardware]: [smdk6410]

[ro.revision]: [0]

[wifi.interface]: [eth0]

[wlan.driver.status]: [ok]

[ro.build.id]: [ERD79]

[ro.build.display.id]: [sec\_smdk6410-eng 2.1 ERD79 eng.leo.20101101.173258 test-keys]

[ro.build.version.incremental]: [eng.leo.20101101.173258]

[ro.build.version.sdk]: [7]

[ro.build.version.codename]: [REL]

[ro.build.version.release]: [2.1]

[ro.build.date]: [2010年 11月 01日 星期一 17:33:52 CST]

[ro.build.date.utc]: [1288604032]

[ro.build.type]: [eng]

[ro.build.user]: [leo]

[ro.build.host]: [leo-desktop]

[ro.build.tags]: [test-keys]

[ro.product.model]: [smdk6410]

[ro.product.brand]: [sec]

[ro.product.name]: [sec\_smdk6410]

[ro.product.device]: [smdk6410]

[ro.product.board]: []

[ro.product.cpu.abi]: [armeabi]

[ro.product.manufacturer]: [Samsung]

[ro.product.locale.language]: [hdpi]

[ro.product.locale.region]: []

[ro.wifi.channels]: []

[ro.board.platform]: [s3c6410]

[ro.build.product]: [smdk6410]

[ro.build.description]: [sec\_smdk6410-eng 2.1 ERD79 eng.leo.20101101.173258 test-keys]

[ro.build.fingerprint]: [sec/sec\_smdk6410/smdk6410/:2.1/ERD79/eng.leo.20101101.173258:eng/test-keys]

[rild.libpath]: [/system/lib/libreference-ril.so]

[rild.libargs]: [-d /dev/ttyS0]

[keyguard.no\_require\_sim]: [true]

[ro.sf.lcd\_density]: [240]

[ro.config.notification\_sound]: [OnTheHunt.ogg]

[ro.config.alarm\_alert]: [Alarm\_Classic.ogg]

[ro.kernel.android.checkjni]: [1]

[ro.setupwizard.mode]: [OPTIONAL]

[net.bt.name]: [Android]

[net.change]: [net.tcp.buffersize.gprs]

[ro.config.sync]: [yes]

[dalvik.vm.stack-trace-file]: [/data/anr/traces.txt]

[persist.sys.language]: [zh]

[persist.sys.localevar]: []

[persist.sys.country]: [CN]

[ro.FOREGROUND\_APP\_ADJ]: [0]

[ro.VISIBLE\_APP\_ADJ]: [1]

[ro.SECONDARY\_SERVER\_ADJ]: [2]

[ro.BACKUP\_APP\_ADJ]: [2]

[ro.HOME\_APP\_ADJ]: [4]

[ro.HIDDEN\_APP\_MIN\_ADJ]: [7]

[ro.CONTENT\_PROVIDER\_ADJ]: [14]

[ro.EMPTY\_APP\_ADJ]: [15]

[ro.FOREGROUND\_APP\_MEM]: [1536]

[ro.VISIBLE\_APP\_MEM]: [2048]

[ro.SECONDARY\_SERVER\_MEM]: [4096]

[ro.BACKUP\_APP\_MEM]: [4096]

[ro.HOME\_APP\_MEM]: [4096]

[ro.HIDDEN\_APP\_MEM]: [5120]

[ro.CONTENT\_PROVIDER\_MEM]: [5632]

[ro.EMPTY\_APP\_MEM]: [6144]

[net.tcp.buffersize.default]: [4096,87380,110208,4096,16384,110208]

[net.tcp.buffersize.wifi]: [4095,87380,110208,4096,16384,110208]

[net.tcp.buffersize.umts]: [4094,87380,110208,4096,16384,110208]

[net.tcp.buffersize.edge]: [4093,26280,35040,4096,16384,35040]

[net.tcp.buffersize.gprs]: [4092,8760,11680,4096,8760,11680]

[init.svc.console]: [running]

[init.svc.servicemanager]: [running]

[init.svc.vold]: [running]

[init.svc.zygote]: [running]

[init.svc.media]: [running]

[init.svc.dbus]: [running]

[init.svc.installd]: [running]

[init.svc.keystore]: [running]

[init.svc.ril-daemon]: [running]

[ro.radio.noril]: [no]

[ro.radio.use-ppp]: [yes]

[ro.config.nocheckin]: [yes]

[init.svc.adbd]: [running]

[adb.connected]: [1]

[init.svc.bootanim]: [stopped]

[hw.keyboards.0.devname]: [s3c-keypad-rev0000]

[sys.settings\_secure\_version]: [3]

[dev.bootcomplete]: [1]

[sys.settings\_system\_version]: [5]

[gsm.version.ril-impl]: [android reference-ril 1.0]

[gsm.sim.operator.numeric]: []

[gsm.sim.operator.alpha]: []

[gsm.sim.operator.iso-country]: []

[gsm.sim.state]: [UNKNOWN]

[gsm.current.phone-type]: [1]

[gsm.operator.alpha]: []

[gsm.operator.numeric]: []

[gsm.operator.iso-country]: []

[gsm.operator.isroaming]: [false]

## Audio，MMC，TTY驱动

Compile the kernel with debug info ：CONFIG\_DEBUG\_INFO:

打开该选项时，编译出的内核将会包含全部的调试信息，使用gdb时需要这些调试信息。

Stack utilization instrumentation ：CONFIG\_DEBUG\_STACK\_USAGE:

该选项用于跟踪内核栈的溢出错误，一个内核栈溢出错误的明显的现象是产生oops错误却没有列出系统的调用栈信息。该选项将使内核进行栈溢出检查，并使内核进行栈使用的统计。

Driver Core verbose debug messages：CONFIG\_DEBUG\_DRIVER:

该选项位于"Device drivers-> Generic Driver Options"下，打开该选项使得内核驱动核心产生大量的调试信息，并将他们记录到系统日志中。

include/linux/device.h

dev\_dbg

CONFIG\_DEBUG\_DRIVER

./.config:# CONFIG\_DEBUG\_DRIVER is not set

./.config.old:# CONFIG\_DEBUG\_DRIVER is not set

./arch/\*/configs/xxx\_defconfig

drivers/base/Makefile

ifeq ($(CONFIG\_DEBUG\_DRIVER),y)

EXTRA\_CFLAGS += -DDEBUG

endif

config DEBUG\_DRIVER

bool "Driver Core verbose debug messages"

depends on DEBUG\_KERNEL

help

Say Y here if you want the Driver core to produce a bunch of

debug messages to the system log. Select this if you are having a

problem with the driver core and want to see more of what is

going on.

If you are unsure about this, say N here.

#从可用的SD 卡拷贝出来

~$ sudo dd if=/dev/sdb of=~/sdb bs=512 skip=3853774记录了8754+0 的读入记录了8754+0 的写出4482048字节(4.5 MB)已复制，0.462473 秒，9.7 MB/秒~$ sudo sync~$ sudo umount /dev/sdb\*umount: /dev/sdb 未挂载

#复制到目标SD 卡

~$ sudo dd of=/dev/sdb if=~/sdb bs=512 seek=3853774记录了8754+0 的读入记录了8754+0 的写出4482048字节(4.5 MB)已复制，1.45462 秒，3.1 MB/秒~$ sudo sync~$ sudo umount /dev/sdb\*umount: /dev/sdb 未挂载

注意一定要sync，我第一次使用的时候，忘了sync，结果是不可用。

kernel $ps axu

kernel$ cat /proc/1/cmdline

/sbin/init

查看启动 1号进程的命令行命令

Initramfs应用问题

CONFIG\_BLK\_DEV\_INITRD=y

CONFIG\_INITRAMFS\_SOURCE="root"

CONFIG\_INITRAMFS\_ROOT\_UID=0

CONFIG\_INITRAMFS\_ROOT\_GID=0

General setup ->Initial RAM filesystem and RAM disk(initramfs/initrd)support

Boot options->Default kernel command string (CMDLINE)

ubrifs

root=/dev/mtdblock2 rootfstype=cramfs init=/linuxrc console=ttySAC0,115200

initramfs

console=ttySAC0,115200

通过kernel menuconfig打开CONFIG\_BLK\_DEV\_INITRD，

并且设置CONFIG\_INITRAMFS\_SOURCE="/home/qianjiang/tmp/initramfs/"，

通过调试理解如下：

内核编译时，通过CONFIG\_INITRAMFS\_SOURCE指定的目录生成cpio文件（所以也可以直接指定cpio文件而不是指定目录）。

内核在初始化启动的时候会先注册一个rootfs的文件系统，然后通过rootfs\_initcall来生成其中的内容。

A. 当CONFIG\_BLK\_DEV\_INITRD未选中时，rootfs\_initcall调用noinitramfs.c中的default\_rootfs()来生成。

B. 如果选中时：

rootfs\_initcall调用initramfs.c中的populate\_rootfs()函数来填充。

注意：系统在启动的时候，完成了cpio到rootfs的填充后，会检查是否存在/init，如果该文件不存在，则调用prepare\_namespace()来挂载文件系统。

上述内容参考了：

http://liaowb1234.blog.163.com/blog/static/771555472010025114231594/

另外一个好的链接参考http://en.gentoo-wiki.com/wiki/Initramfs

确实写得很详细。

内核源码与initramfs 相关的目录有 init及usr。

kernel/arch/arm/mm/ioremap.c

kernel/arch/arm/mm/iomap.c

./arch/arm/include/asm/io.h:223:#define ioremap(cookie,size) \_\_arm\_ioremap(cookie, size, MT\_DEVICE)

**real6410** 2.6.28.6 版本mmc 驱动不支持从SD卡挂载文件系统。放弃了。

driver/mmc

include/linux/mmc

./arch/arm/plat-s3c/include/plat/partition.h

./drivers/mtd/mtdpart.c add\_mtd\_partitions

<5>Creating 2 MTD partitions on "NAND 1GiB 3,3V 8-bit":

<5>0x00400000-0x00800000 : "cramfs"

kernel$ find . -name "\*.[ch]" | xargs grep -n --color -e "\<ubifs\>"

./arch/arm/plat-s3c/include/plat/partition.h:35: .name = "ubifs",

**smdk6410**

[ 1.776692] S3C NAND Driver, (c) 2008 Samsung Electronics

[ 1.781242] S3C NAND Driver is using hardware ECC.

[ 1.784697] NAND device: Manufacturer ID: 0xec, Chip ID: 0xd3 (Samsung NAND 1GiB 3,3V 8-bit)

[ 1.793204] Creating 7 MTD partitions on "NAND 1GiB 3,3V 8-bit":

[ 1.799129] 0x0000000c0000-0x000000100000 : "misc"

[ 1.807419] 0x000000100000-0x000000600000 : "recovery"

[ 1.812367] 0x000000600000-0x000000900000 : "kernel"

[ 1.817193] 0x000000900000-0x000000a00000 : "ramdisk"

[ 1.821900] 0x000000a00000-0x000004d00000 : "system"

[ 1.833242] 0x000004d00000-0x000009000000 : "cache"

[ 1.845754] 0x000009000000-0x000040000000 : "userdata"

kernel$ find . -name "\*.[ch]" | xargs grep -n --color -e "\"userdata\""

./arch/powerpc/platforms/pseries/eeh\_driver.c:88: \* passed back in "userdata".

./arch/powerpc/platforms/pseries/eeh\_driver.c:122: \* Cumulative response passed back in "userdata".

./arch/arm/plat-s3c/include/plat/partition.h:52: .name = "userdata",

改了之后，发现没有效果。

real6410

<6>s3c-sdhci s3c-sdhci.0: clock source 0: hsmmc (133333333 Hz)

s3c-sdhci s3c-sdhci.0: clock source 0: hsmmc (133333333 Hz)

<6>s3c-sdhci s3c-sdhci.0: clock source 1: hsmmc (133333333 Hz)

s3c-sdhci s3c-sdhci.0: clock source 1: hsmmc (133333333 Hz)

<6>s3c-sdhci s3c-sdhci.0: clock source 2: mmc\_bus (44333333 Hz)

s3c-sdhci s3c-sdhci.0: clock source 2: mmc\_bus (44333333 Hz)

smdk6410

[ 3.277675] s3c-sdhci s3c-sdhci.0: clock source 0: hsmmc (133333333 Hz)

[ 3.284569] s3c-sdhci s3c-sdhci.0: clock source 1: hsmmc (133333333 Hz)

[ 3.290735] s3c-sdhci s3c-sdhci.0: clock source 2: hsmmc (133333333 Hz)

./drivers/mmc/host/sdhci-s3c.c:380: dev\_info(dev, "clock source %d: %s (%ld Hz)\n",

arch/arm/mach-s3c6410/setup-sdhci.c s3c6410\_hsmmc\_clksrcs

arch/arm/mach-s3c6410/cpu.c 调用函数 s3c6410\_default\_sdhci2， 此函数在文件

arch/arm/plat-s3c/include/plat/sdhci.h 中定义。又涉及到宏定义

CONFIG\_S3C\_DEV\_HSMMC2 . include/linux/autoconf.h

./arch/arm/plat-s3c/Makefile:41:obj-$(CONFIG\_S3C\_DEV\_HSMMC2) += dev-hsmmc2.o

./arch/arm/plat-s3c/Kconfig:175:config S3C\_DEV\_HSMMC2

./arch/arm/mach-s3c6410/Kconfig:27: select S3C\_DEV\_HSMMC2

所以，在修改setup-sdhci.c，把s3c6410\_setup\_sdhci1\_cfg\_card， s3c6410\_setup\_sdhci2\_cfg\_card 两个函数干掉后，

arch/arm/plat-s3c/include/plat/sdhci.h

arch/arm/plat-s3c/include/plat/sdhci.h 内联函数 s3c6410\_default\_sdhci1 要修改。

修改./arch/arm/mach-s3c6410/Kconfig ，去掉select S3C\_DEV\_HSMMC2，这个不是必需的。

日志函数printk 在文件kernel/printk.c中实现。 make xconfig 中， kernel hacking项下 就有日志的相关的设置， 如，Show timing information on printks 选项会在log前打印 时间戳“[ 3.277675]”。

./lib/Kconfig.debug:3: bool "Show timing information on printks"

drivers/base/platform.c

drivers/base/Makefile $(info $$(CONFIG\_DEBUG\_DRIVER) is $(CONFIG\_DEBUG\_DRIVER))

drivers/base/dd.c

drivers/base/core.c

drivers/base/bus.c

sound/soc/soc-core.c

sound/soc/s3c/s3c-ac97.c

打开调试信息

SD/MMC : Device Drivers->MMC/SD/SDIO card support -> MMC debugging

Sound : Device Drivers-> Sound card support -> Advanced Linux Sound Architecture -> Debug

或者 Verbose printk

driver : Device Drivers-> Generic Driver Options -> Driver Core verbose debug message

U-Boot

U-Boot 1.1.6 (May 20 2010 - 15:15:46) for SMDK6410

CPU: S3C6410@800MHz

Fclk = 800MHz, Hclk = 133MHz, Pclk = 66MHz, Serial = CLKUART (SYNC Mode)

Board: SMDK6410

DRAM: 256 MB

Flash: 0 kB

NAND: Maf. ID is d3

1024 MB

MMC: 1887 MB

In: serial

Out: serial

Err: serial

Hit any key to stop autoboot: 0

这个提示时就可以按键进入u-boot shell了。

使用串口发送文件：

#loadb

Ctrl+/ C 切回kermit

kermit>send /tftpboot/zImage

# android 使用ALSA驱动不出声

标题:android 使用ALSA驱动不出声

作者:xiashangbaby

日期:2009-07-18 11:47

内容:

这几天一直在android上搞alsa，结果用android自带的Music播放wav文件不能出声，直接用命令行方式

# cat 11.wav > /dev/snd/audio

# cat 11.wav > /dev/snd/dsp

# cat 11.wav > /dev/snd/mixer

等都是一个结果，最后都报：invalid length。也不能出声，郁闷请大虾给看看是什么原因，不胜感激！

下面是/etc/asound.conf

# Android ALSA configuration file for the AK4671 audio.

########################################################################

#{name "Playback Path"value 2}# 0:OFF 1:RCV 2:SPK

## 3:HP 4:BT

#{name "Output Volume"value 30}# min:0 max:48

#{name "Output Volume - RCV"value 2}# min:0 max:7

#{name "Output Volume - SPK/EAR"value 8} # min:0 max:15

#{name "Voice Call Path"value 1}# 0:OFF 1:RCV 2:SPK

## 3:HP 4:BT

#{name "Voice Memo Path"value 1}# 0:OFF 1:MAIN 2:SUB

## 3:EAR 4:BT

#{name "MIC Path"value 1} # 0:Main Mic 1:Sub MIC

#{name "MIC Gain"value 5,5}# min:0 max:15

#{name "FM Radio Path"value 1}# 0:off 1:RCV 2:SPK

## 3:HP 4:BT

#{name "Idle Mode"value 1}# 0:off 1:on

#########################################################################

##

## Mixer Devices

##

ctl.AndroidPlayback {

type hw

card 0

}

ctl.AndroidRecord {

type hw

card 0

}

##

## Playback Devices

##

pcm.AndroidPlayback {

type hooks

slave.pcm {

type hw

ca ..

#1 android 使用ALSA驱动不出声 [xiashangbaby 07-18 11:54]

需要声明的是在板子上没有alsaconf命令，也没有lspci命令，而且alaa\_amixer也不是界面方式的，执行

# alsa\_amixer输出如下：

Simple mixer control 'FM Radio Path',0

Capabilities: enum

Items: 'Off' 'RCV' 'SPK' 'HP' 'BT'

Item0: 'Off'

Simple mixer control 'Playback Path',0

Capabilities: enum

Items: 'Off' 'RCV' 'SPK' 'HP' 'BT' 'SPK\_HP'

Item0: 'SPK'

Simple mixer control 'Idle Mode',0

Capabilities: enum

Items: 'Off' 'ON'

Item0: 'ON'

Simple mixer control 'MIC Gain',0

Capabilities: volume

Playback channels: Front Left - Front Right

Capture channels: Front Left - Front Right

Limits: 0 - 15

Front Left: 5 [33%]

Front Right: 5 [33%]

Simple mixer control 'MIC Path',0

Capabilities: enum

Items: 'Main Mic' 'Sub Mic'

Item0: 'Main Mic'

Simple mixer control 'Output Volume - MUTE',0

Capabilities: pswitch pswitch-joined

Playback channels: Mono

Mono: Playback [off]

Simple mixer control 'Output Volume - RCV',0

Capabilities: volume volume-joined

Playback channels: Mono

Capture channels: Mono

Limits: 0 - 7

Mono: 5 [71%]

Simple mixer control 'Output Volume - SPK/EAR',0

Capabilities: volume volume-joined

Playback channels: Mono

Capture channels: Mono

Limits: 0 - 15

Mono: 12 [80%]

Simple mixer control 'Output Volume L',0

Capabilities: volume volume-joined

Playback channels: Mono

Capture channels: Mono

Limits: 0 - 48

Mono: 27 [56%]

Simple mixer control 'Output Volume R',0

Capabilities: volume volume-joined

Playback channels: Mono

Capture channels: Mono

Limits: 0 - 48

Mono: 27 [56%]

Simple mix ..

#2 [xiashangbaby 07-18 11:57]

下面分别是执行信息：

# alsa\_aplay -h

Usage: aplay [OPTION]... [FILE]...

-h, --help help

--version print current version

-l, --list-devices list all soundcards and digital audio devices

-L, --list-pcms list device names

-D, --device=NAME select PCM by name

-q, --quiet quiet mode

-t, --file-type TYPE file type (voc, wav, raw or au)

-c, --channels=# channels

-f, --format=FORMAT sample format (case insensitive)

-r, --rate=# sample rate

-d, --duration=# interrupt after # seconds

-M, --mmap mmap stream

-N, --nonblock nonblocking mode

-F, --period-time=# distance between interrupts is # microseconds

-B, --buffer-time=# buffer duration is # microseconds

--period-size=# distance between interrupts is # frames

--buffer-size=# buffer duration is # frames

-A, --avail-min=# min available space for wakeup is # microseconds

-R, --start-delay=# delay for automatic PCM start is # microseconds

(relative to buffer size if <= 0)

-T, --stop-delay=# delay for automatic PC ..

#3 [xiashangbaby 07-18 11:59]

执行

# alsa\_alsactl -h

Usage: alsactl <options> command

Available global options:

-h,--help this help

-d,--debug debug mode

-v,--version print version of this program

Available state options:

-f,--file # configuration file (default /etc/asound.state)

-F,--force try to restore the matching controls as much as possible

(default mode)

-g,--ignore ignore 'No soundcards found' error

-P,--pedantic do not restore mismatching controls (old default)

-I,--no-init-fallback

don't initialize even if restore fails

-r,--runstate # save restore and init state to this file (only errors)

default settings is 'no file set'

-R,--remove remove runstate file at first, otherwise append errors

Available init options:

-E,--env #=# set environment variable for init phase (NAME=VALUE)

-i,--initfile # main configuation file for init phase (default /system/usr/share/alsa/init/00main)

Available commands:

store <card #> save current driver setup for one or each soundcards

to configuration file

restore <card #> load current driver setup for one or each soundcards

..

#4 [xiashangbaby 07-18 12:05]

奇怪的是直接执行：

# alsa\_aplay ./11.wav报错：

aplay: main:590: audio open error: Device or resource busy

但用android自带的Music播放11.wav没看见报这样的错，但依旧没声音！

#5 [xiashangbaby 07-18 12:08]

下面是板子dmesg消息：

<5>Linux version 2.6.27 (yoohyuk@corona) (gcc version 4.3.1 (for S3C64XX Samsung Electronics AP Development Team) ) #2 Wed Jul 8 16:31:37 KST 2009

<4>CPU: ARMv6-compatible processor [410fb766] revision 6 (ARMv7), cr=00c5307f

<4>Machine: CYGNUS

<4>Memory policy: ECC disabled, Data cache writeback

<7>On node 0 totalpages: 43264

<7>free\_area\_init\_node: node 0, pgdat c06e4d0c, node\_mem\_map c077a000

<7> DMA zone: 42926 pages, LIFO batch:7

<4>CPU S3C6410 (id 0x36410101)

<7>s3c6410\_init\_clocks: initialising clocks

<6>S3C Clocks, (c) 2004 Simtec Electronics

<7>s3c6400\_setup\_clocks: registering clocks

<7>s3c6400\_setup\_clocks: clkdiv0 = 01043510

<7>s3c6400\_setup\_clocks: xtal is 12000000

<6>S3C64XX: PLL settings, A=800000000, M=266000000, E=24000000

<6>S3C64XX: HCLKx2=266666666, HCLK=133333333, PCLK=66666666

<6>mout\_apll: source is fout\_apll (1), rate is 800000000

<6>mout\_epll: source is fout\_epll (1), rate is 24000000

<6>mout\_mpll: source is mpll (1), rate is 266000000

<6>mmc\_bus: source is mout\_epll (0), rate is 24000000

<6>mmc\_bus: source is mout\_epll (0), rate is 24000000

<6>mmc\_bus: source is mout\_epll (0), rate is 24000000

<6>usb-host-bus: source is mout\_epll (0), rate is 24000000

<7>s3c64xx\_clk\_doutmpll\_get\_rate: parent is 266000000

<6>lcd: source is dout\_mpll (1), rate is 133000000

<7>s3c64xx\_clk\_doutmpll\_get\_rate: parent is 266000000

<6>uclk1: source is dout\_mpll (1), rate is 66500000

<6>spi-bus: source is mout\_epll (0), rate is 24000000

<6>spi-bus: source is mout\_epll (0), rate is 24000000

<6>audio-bus: source is mout\_epll (0), rate is 24000000

<6>audio-bus: source is mout\_epll (0), rate is 24000000

<6>irda-bus: source is mout\_epll (0), rate is 24000000

<4>CPU0: D VIPT write-back cache

<4>CPU0: I cache: 16384 b ..

<http://www.linuxforum.net/forum/showflat.php?Cat=&Board=embedded&Number=751020&page=0&view=collapsed&sb=5&o=0&fpart>=

问题搞定了，总结如下。

alsa 音频路径的问题：

在sound/soc/codecs目录中有很多音频codec的codec驱动，我使用的是wm9713，AP是s3c6410；这里个驱动文件中定义了很多widget和control，alsa在playback或record的时候，sound/soc/soc-dapm.c中的dapm\_power\_widgets函数会根据“配置情况”打开相应的widget，搭建一个完整的音频路径，只要该路径搭建成功，就可以正常工作；

sound/soc/codecs/wm9713.c中的audio\_map[]就是一个wm9713的路由表，根据wm9713手册中的Audio Paths Overview可以选择自己需要的音频路径，在audio\_map[]中测试一下，看audio\_map中是否支持这种路径。

alsa 的配置

alsa音频的调试最主要的是alsa的配置。alsa使用amixer命令打开audio\_map[]中的开关（control/switch）和其它一些controls，这些control设置后，使用aplay/arecord的时候即可搭建正确的路径，实现播放和录音。

比如我在调试的时候，在不用amixer控制时（默认状态），arecord可以正确录音，使用sound/soc/soc-dapm.c中的dump\_dapm函数dump出的路径是正确的；而aplay的时候，dump\_dapm出来的路径是错误的，原因是默认设置里没有打开playback的开关（switch），运行如下命令即可正确playback：

alsa\_amixer cset numid=4,iface=MIXER,name='Headphone Playback Switch' 1

alsa\_amixer cset numid=93,iface=MIXER,name='Left Headphone Out Mux' 2

alsa\_amixer cset numid=34,iface=MIXER,name='Out3 Playback Switch'1

alsa\_amixer cset numid=95,iface=MIXER,name='Left Speaker Out Mux' 4

alsa\_amixer cset numid=94,iface=MIXER,name='Right Speaker Out Mux' 2

alsa\_amixer cset numid=91,iface=MIXER,name='Out 3 Mux' 2

alsa\_amixer cset numid=81,iface=MIXER,name='Left HP Mixer PCM Playback Swit' 1

alsa\_amixer cset numid=75,iface=MIXER,name='Right HP Mixer PCM Playback Swi' 1

alsa\_amixer cset numid=3,iface=MIXER,name='Headphone Playback Volume' 26

alsa\_amixer cset numid=36,iface=MIXER,name='Out3 Playback Volume' 48

其实就是打开playback路径需要的开关，dapm\_power\_widgets会自动把这些开关连接的widget连接起来，构成一个播放路径。

Android中alsa的配置：

Andriod中使用alsa-lib，也需要对配置音频路径。配置方法有两个：

1、在AudioHardwareALSA.cpp中的doRouting中使用system函数调用amixer进行配置

system("alsa\_amixer cset numid=4,iface=MIXER,name='Headphone Playback Switch' 1");

2、编写asound.conf文件，AudioHardwareALSA.cpp中的ALSAMixer::ALSAMixer对象初始化的时候会通过alsa-lib的conf.c文件中的函数读取/etc/asound.conf文件，获取配置信息，对codec进行配置。

alsa的一些命令

alsa\_amixer该命令配置主要配置音频codec的mixer开关、mux对路选择、volume值等；

alsa\_amixer --help

alsa\_amixer contents

alsa\_amixer contents

numid=30,iface=MIXER,name='Headphone Playback ZC Switch'

; type=BOOLEAN,access=rw------,values=2

: values=off,off

numid=4,iface=MIXER,name='Headphone Playback Switch'

; type=BOOLEAN,access=rw------,values=2

: values=off,off

numid=3,iface=MIXER,name='Headphone Playback Volume'

; type=INTEGER,access=rw------,values=2,min=0,max=31,step=0

: values=31,31

numid=6,iface=MIXER,name='PCM Playback Volume'

; type=INTEGER,access=rw------,values=2,min=0,max=31,step=0

: values=23,23

numid=5,iface=MIXER,name='Line In Volume'

; type=INTEGER,access=rw------,values=2,min=0,max=31,step=0

: values=23,23

numid=7,iface=MIXER,name='Mic 1 Volume'

; type=INTEGER,access=rw------,values=1,min=0,max=31,step=0

: values=23

numid=8,iface=MIXER,name='Mic 2 Volume'

; type=INTEGER,access=rw------,values=1,min=0,max=31,step=0

: values=23

numid=85,iface=MIXER,name='Mic A Source'

; type=ENUMERATED,access=rw------,values=1,items=3

; Item #0 'Mic 1'

; Item #1 'Mic 2 A'

; Item #2 'Mic 2 B'

: values=0

alsa\_amixer controls

numid=30,iface=MIXER,name='Headphone Playback ZC Switch'

numid=4,iface=MIXER,name='Headphone Playback Switch'

numid=3,iface=MIXER,name='Headphone Playback Volume'

numid=6,iface=MIXER,name='PCM Playback Volume'

numid=5,iface=MIXER,name='Line In Volume'

numid=7,iface=MIXER,name='Mic 1 Volume'

numid=8,iface=MIXER,name='Mic 2 Volume'

numid=85,iface=MIXER,name='Mic A Source'

numid=84,iface=MIXER,name='Mic B Source'

numid=9,iface=MIXER,name='Mic Boost (+20dB) Switch'

numid=10,iface=MIXER,name='Mic Headphone Mixer Volume'

numid=47,iface=MIXER,name='Aux Playback Headphone Volume'

numid=48,iface=MIXER,name='Aux Playback Master Volume'

numid=49,iface=MIXER,name='Aux Playback Mono Volume'

numid=67,iface=MIXER,name='Mono Mixer Aux Playback Switch'

numid=69,iface=MIXER,name='Mono Mixer Bypass Playback Swit'

numid=70,iface=MIXER,name='Mono Mixer Mic 1 Sidetone Switc'

numid=71,iface=MIXER,name='Mono Mixer Mic 2 Sidetone Switc'

numid=65,iface=MIXER,name='Mono Mixer PC Beep Playback Swi'

numid=68,iface=MIXER,name='Mono Mixer PCM Playback Switch'

numid=66,iface=MIXER,name='Mono Mixer Voice Playback Switc'

alsa\_alsactl store：该命令生成/etc/asound.state文件，该文件显示当请codec的状态，可以根据该文件检查codec的状态是否正确。

# cat /etc/asound.state

state.SMDK6400 {

control.1 {

comment.access 'read write'

comment.type INTEGER

comment.count 2

comment.range '0 - 31'

iface MIXER

name 'Speaker Playback Volume'

value.0 31

value.1 31

}

control.2 {

comment.access 'read write'

comment.type BOOLEAN

comment.count 2

iface MIXER

name 'Speaker Playback Switch'

value.0 false

value.1 false

}

control.3 {

comment.access 'read write'

comment.type INTEGER

comment.count 2

comment.range '0 - 31'

iface MIXER

name 'Headphone Playback Volume'

value.0 26

value.1 26

}

control.4 {

comment.access 'read write'

comment.type BOOLEAN

comment.count 2

iface MIXER

name 'Headphone Playback Switch'

value.0 true

value.1 true

}

control.5 {

comment.access 'read write'

comment.type INTEGER

comment.count 2

comment.range '0 - 31'

iface MIXER

name 'Line In Volume'

value.0 23

value.1 23

}

使用amixer命令可以参照alsa\_amixer contents的内容；编写asound.conf，可以参照alsa\_alsactl生成的/etc/asound.state。

下面是我的asound.conf的一段，其它的pcm.AndroidPlayback\_xxx可以写法一样，就是hook\_argsp[]中的内容根据自己的情况设置。

##

## Mixer Devices

##

ctl.AndroidPlayback {

type hw

card 0 # Can replace with driver"s name from /proc/asound/cards

}

ctl.AndroidRecord {

type hw

card 0 # Can replace with driver"s name from /proc/asound/cards

}

##

## Playback Devices

##

pcm.AndroidPlayback {

type hooks

slave.pcm {

type hw

card 0

device 0 # Must be of type "digital audio playback"

}

hooks.0 {

type ctl\_elems

hook\_args [

{

name 'Master Playback Switch'

value true

}

{

name 'Master Playback Volume'

value.0 51

value.1 51

}

{

name 'Phone Playback Switch'

value false

}

{

name 'Phone Playback Volume'

value.0 0

value.1 0

}

{

name 'Mic Playback Switch'

value false

}

{

name 'Mic Playback Volume'

value.0 0

value.1 0

}

{

name 'Mic Boost (+20dB)'

value false

}

{

name 'Line Playback Switch'

value false

}

{

name 'Line Playback Volume'

value.0 0

value.1 0

}

{

name 'PCM Playback Switch'

value true

}

{

name 'PCM Playback Volume'

value.0 51

value.1 51

}

{

name 'Capture Source'

value.0 Mic

value.1 Mic

}

{

name 'Capture Switch'

value true

}

{

name 'Capture Volume'

value.0 0

value.1 0

}

]

}

}

呵呵，这个帖子是第一次在linux forum中写总结，自问自答，希望对调音频的兄弟有点帮助。

编辑者： hpyu (10-05-18 10:49)