

Linux ALSA编程

一.音频的概念

- 1.声音的三要素
- 2.音频的概念
- 3.音频相关的术语

二.alsa的概念

- 1.ALSA架构介绍
- 2.ALSA移植
 - 2.1 ALSA库下载
 - 2.2 ALSA lib编译
 - 2.3 ALSA Util编译
 - 2.4 ALSA库和工具移植入嵌入式平台

三.alsa工具应用

- 1.节点信息
- 2.工具使用
 - 2.1 alsamixer
 - 2.2 aplay
 - 2.3 arecord
 - 2.4 补充问题点

四.alsa编程

- 1.Audio使用的伪代码为
- 2.最小的playback使用程序
- 3.最小的capture程序

版本	日期	作者	变更表述
1.0	2022/12/15	于忠军	文档创建

一.音频的概念

1.声音的三要素

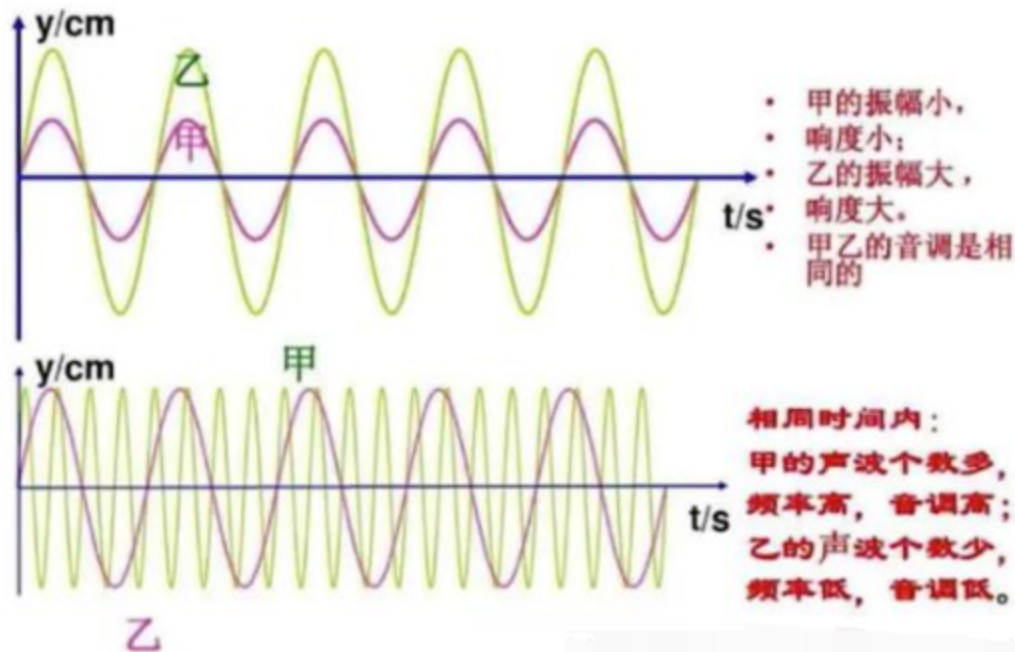
声音的三要素：频率、振幅、波形。

频率：声波的频率，即声音的音调，人类听觉的频率（音调）范围为 20Hz—20KHz

振幅：即声波的响度，通俗的讲就是声音的高低，一般男生的声音振幅（响度）大于女生。

波形：即声音的音色，同样的频率和振幅下，钢琴和小提琴的声音听起来完全不同的，因为他们的音色不同。波形决定了其所代表声音的音色。音色不同是因为它们的介质所产生的波形不同。

声波波形图



2.音频的概念

音频数据的承载方式最常用的是脉冲编码调制 脉冲编码调制，即 PCM。

在自然界中，声音是连续不断的，是一种模拟信号，那怎样才能把声音保存下来呢？

那就是把声音数字化，即转换为数字信号。

我们知道声音是一种波，有自己的振幅和频率，那么要保存声音，就要保存声音在各个时间点上的振幅。而数字信号并不能连续保存所有时间点的振幅，事实上，并不需要保存连续的信号，就可以还原到人耳可接受的声音。

根据奈奎斯特采样定理：为了不失真地恢复模拟信号，采样频率应该不小于模拟信号频谱中最高频率的 2 倍。

根据以上分析，PCM 的采集步骤分为以下步骤：

模拟信号 模拟信号 -> 采样 -> 量化 -> 编码 -> 数字信号

音频到底是什么？

音频这个专业术语，人类能够听到的所有声音都称之为音频，它可能包括噪音、声音被录制下来以后，无论是说话声、歌声、乐器都可以通过数字音乐软件处理。把它制作成 CD，这时候所有的声音没有改变，因为 CD 本来就是音频文件的一种类型。而音频只是储存在计算机里的声音。演讲和音乐，如果有计算机加上相应的音频卡——就是我们经常说的声卡，我们可以把所有的声音录制下来，声音的声学特性，音的高低都可以用计算机硬盘文件的方式储存下来。反过来，我们也可以把储存下来的音频文件通过一定的音频程序播放，还原以前录下的声音。

3.音频相关的术语

响度和强度：声音的主观属性响度表示的是一个声音听来有多响的程度。响度主要随声音的强度而变化，但也受频率的影响。总的说，中频纯音听来比低频和高频纯音响一些。

采样率(Sample Rate)：采样率，即采样的频率。

上面提到，采样率要大于原声波频率的 2 倍，人耳能听到的最高频率为 20kHz，所以为了满足人耳的听觉要求，采样率至少为 40kHz，通常为 44.1kHz，更高的通常为 48kHz。

注意：人耳听觉频率范围[20Hz, 20KHz]。

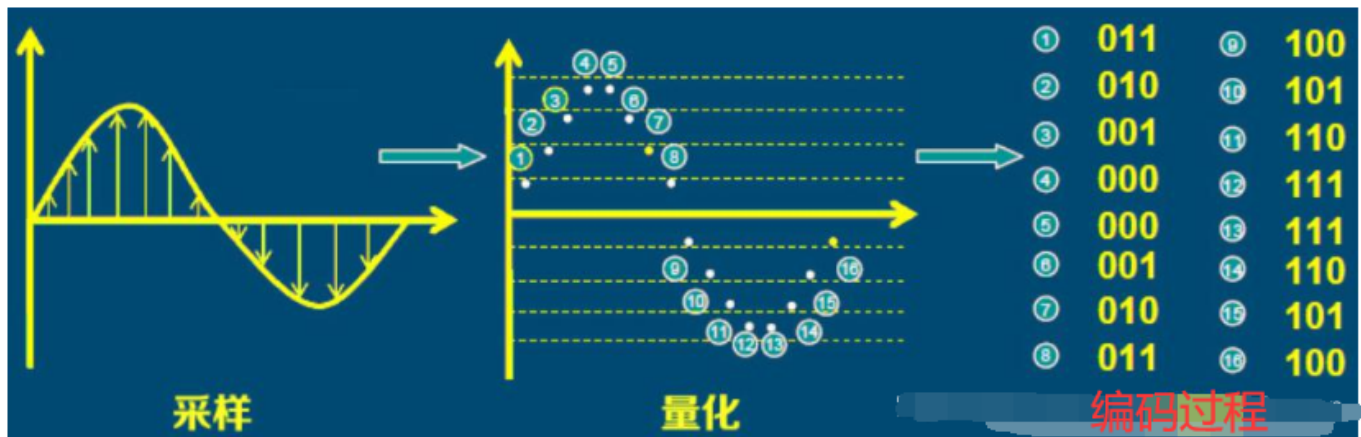
采样位数：涉及到上面提到的振幅量化。波形振幅在模拟信号上也是连续的样本值，而在数字信号中，信号一般是不连续的，所以模拟信号量化以后，只能取一个近似的整数值，为了记录这些振幅值，采样器会采用一个固定的位数 采样器会采用一个固定的位数来记录这些振幅值，通常有 8 位、16 位、32 位。

注意：位数越多，记录的值越准确，还原度越高。但是占用的硬盘空间越大。

比特率：表示经过编码（压缩）后的音频数据每秒钟需要用多少个比特来表示，单位常为 kbps。

音频编码：由于数字信号是由 0，1 组成的，因此，需要将幅度值转换为一系列 0 和 1 进行存储，也就是编码，最后得到的数据就是数字信号：一串一串 0 和 1 组成的数据组成的数据。

过程如下：



声道数(Channel): 声道数, 是指 支持能不同发声 (注意是不同声音) 的音响的个数。

- **单声道: 1个声道**
- **双声道: 2个声道**
- **立体声道: 默认为2个声道**
- **立体声道 (4声道) : 4个声道**

单声道的声音只能使用一个扬声器发声, 或者也可以处理成两个扬声器输出同一个声道的声音, 当通过两个扬声器回放单声道信息的时候, 我们可以明显感觉到声音是从两个音箱中间传递到我们耳朵里的, 无法判断声源的具体位置。

双声道就是有两个声音通道, 其原理是人们听到声音时可以根据左耳和右耳对声音相位差来判断声源的具体位置。声音在录制过程中被分配到两个独立的声道, 从而达到了很好的声音定位效果。

记录声音时, 如果每次生成一个声波数据, 称为单声道; 每次生成两个声波数据, 称为双声道 (立体声)。立体声 (双声道) 存储大小是单声道文件的两倍。

码率: 码率, 是指一个数据流中每秒钟能通过的信息量, 单位bps (bit per second) 。

码率 = 采样率 * 采样位数 * 声道数。

音频帧(Frame): 音频跟视频不太一样, 视频的每一帧就是一副图像, 但是因为 音频是流式的, 本身是没有一帧的概念的。而且有些时候确实没有办法说一帧怎么怎么样。比如对于 PCM 流来说, 采样率为 44100Hz, 采样位数为 16, 通道数为 2, 那么一秒的音频固定大小的: $44100 \times 16 \times 2 / 8$ 字节。但是人们可以规定一帧的概念, 比如 amr 帧比较简单, 它规定每 20ms 的音频 的音频是一帧。

关于音频文件大小的计算: 文件大小 = 采样率 * 录音时间 * 采样位数 / 8 * 通道数

PCM 流: PCM 流就是原始收录声音时, 数据会保存到一串 buffer 中, 这串 buffer, 就采用了 PCM 格式存储的。通常把音频采样过程也叫做 脉冲编码调制编码, 即 PCM (Pulse Code Modulation) 编码,

采样值也叫 PCM 值。编码过程：模拟信号 模拟信号-> 抽样-> 量化-> 编码->数字信号 数字信号。在 windows 中，通过 WaveIn 或者 CoreAudio 采集声音，得到的原始数据就是一串 PCM格式的 buffer。

音频格式：是指要在计算机内播放或是处理音频文件，也就是要对声音文件进行数、模转换，这个过程同样由采 样和量化构成，人耳所能听到的声音，最低的频率是从 20Hz 起一直到最高频率 20KHZ，20KHz 以上人耳是听不到 的，因此音频文件格式的最大带宽是 20KHZ，故而采样速率需要介于 40~50KHZ 之间，而且对每个样本需要更多的量化比特数。

音频数字化的标准是每个样本 16 位-96dB 的信噪比，采用线性脉冲编码调制 PCM，每一量化步长都具有相等的长度。在音频文件的制作中，正是采用这一标准。

常见的音频格式有 常见的音频格式有：CD 格式、WAVE (*.WAV) 、 AIFF、MP3、MIDI、 AAC、WMA。

二.alsa的概念

ALSA全称是Advanced Linux Sound Architecture，中文音译是Linux高级声音体系。ALSA 是Linux内核2.6后续版本中支持音频系统的标准接口程序，由ALSA库、内核驱动和相关测试开发工具组成，更好的管理Linux中音频系统。

1.ALSA架构介绍

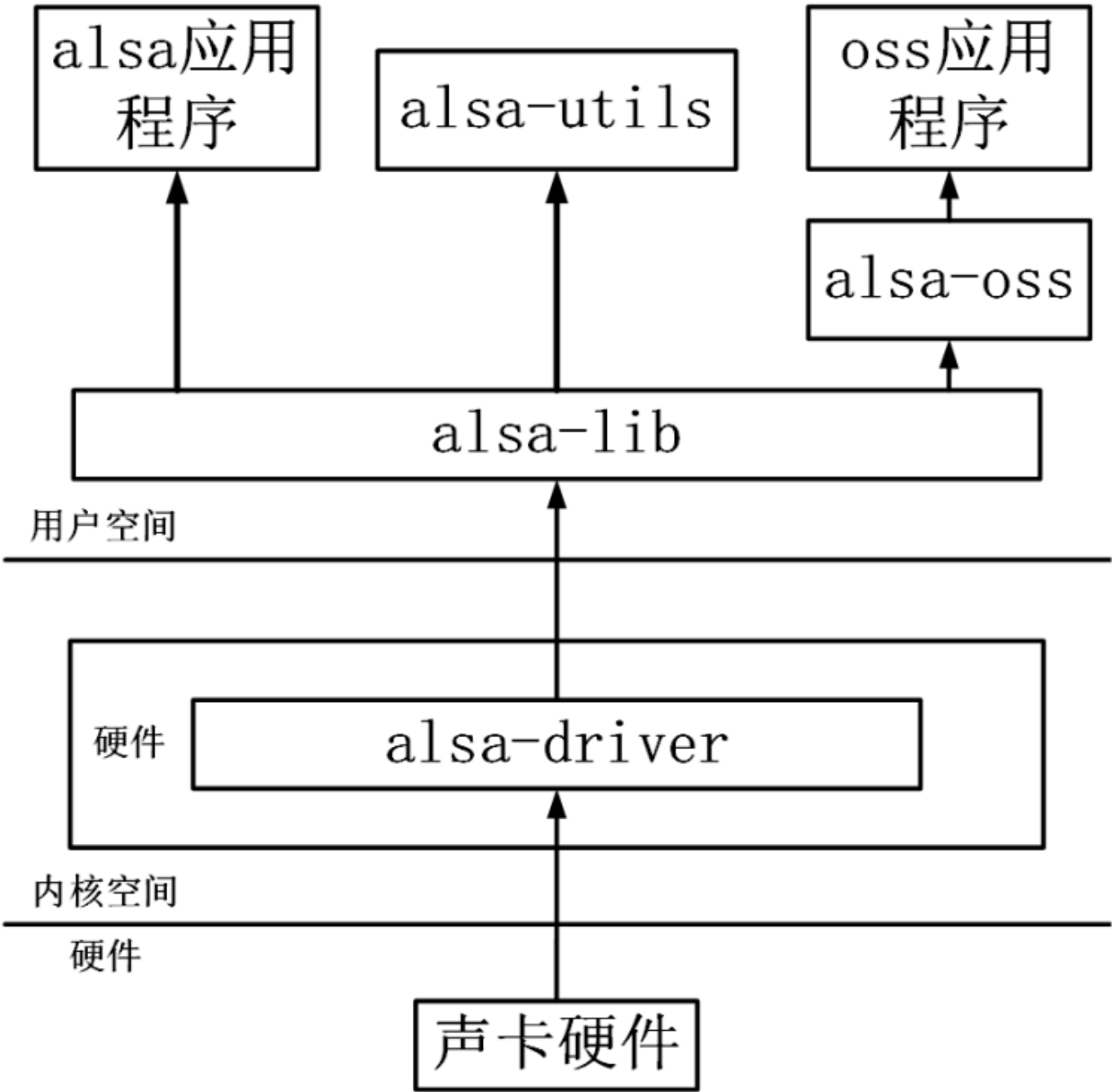
ALSA是Linux系统中为声卡提供驱动的内核组件。它提供了专门的库函数来简化相应应用程序的编写。相较于OSS的编程接口,ALSA的函数库更加便于使用。

对应用程序而言ALSA无疑是一个更佳的选择,因为它具有更加友好的编程接口,并且完全兼容于OSS。

ALSA系统包括7个子项目：

- 驱动包alsa-driver
- 开发包alsa-libs
- 开发包插件alsa-libplugins
- 设置管理工具包alsa-utils
- OSS接口兼容模拟层工具alsa-oss
- 特殊音频固件支持包alsa-firmware
- 其他声音相关处理小程序包alsa-tools

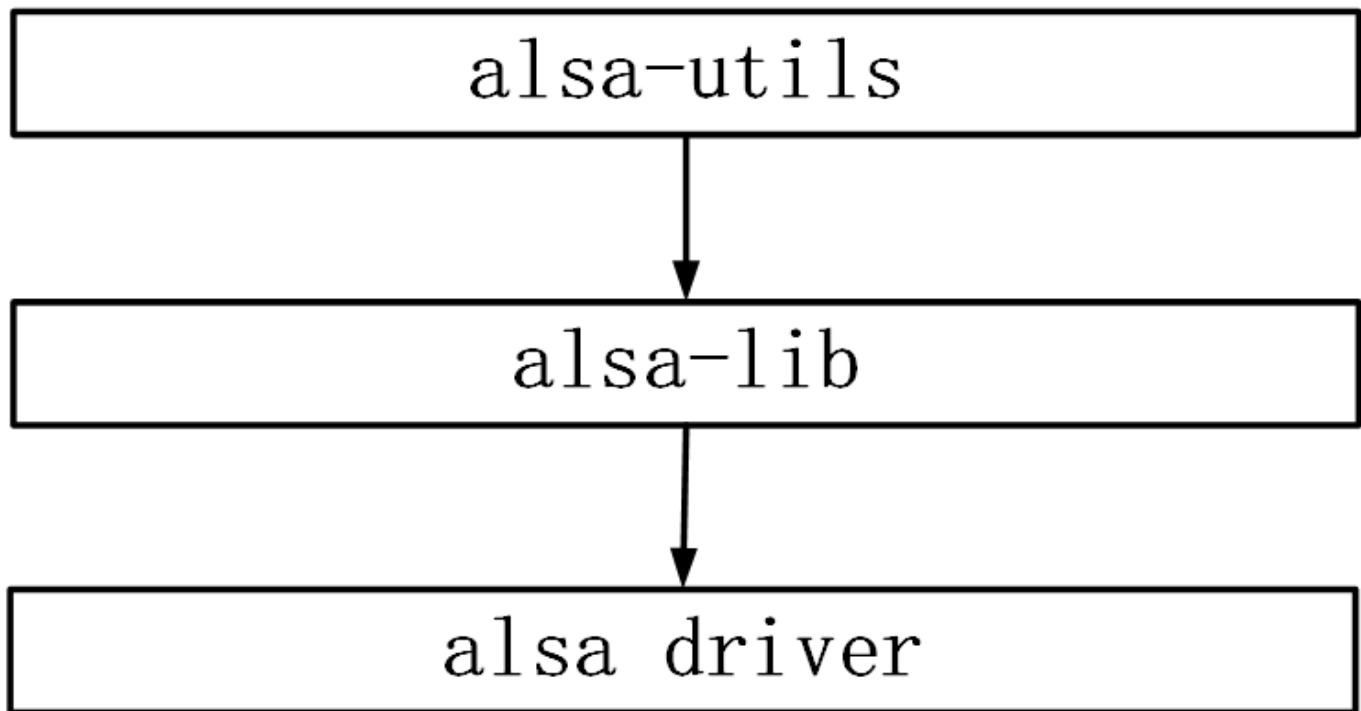
ALSA声卡驱动与用户空间体系结构交互如下图所示：



2.ALSA移植

移植ALSA主要是移植alsa-lib和alsa-utils。


- **alsa-lib**：用户空间函数库, 封装驱动提供的抽象接口, 通过文件libasound.so提供API给应用程序使用。
- **alsa-utils**：实用工具包,通过调用alsa-lib实现播放音频(aplay)、录音(arecord) 等工具。



ALSA Util是纯应用层的软件，相当于ALSA设备的测试程序，ALSA-Lib则是支持应用API的中间层程序，ALSA-Util中的应用程序中会调用到ALSA-Lib中的接口来操作到我们的音频编解码芯片的寄存器，而lib中接口就是依赖于最底层驱动代码，因此移植ALSA程序的顺序就是先后移植Driver,Lib,Util。

2.1 ALSA库下载

LSA首先需要在ALSA的官网上下载官网https://www.alsa-project.org/wiki/Main_Page下载alsa-lib和alsa-utils。



[Main Page](#)
[Discussion](#)

Advanced Linux Sound Architecture (ALSA) project homepage

Main Page

- Developer Zone
- Download...
- SoundCards
- Applications
- Documentation
- Mailing-lists
- Bug Tracker
- Help to debug
- Links


wiki

- About
- Community portal
- Current events
- Recent changes
- Random page
- Help

Tools

- What links here
- Related changes
- Special pages
- Printable version
- Permanent link
- Page information

Advertisements



新一代
VPN连接
技术

Introduction

The Advanced Linux Sound Architecture (ALSA) provides audio and MIDI functionality to the Linux operating system. ALSA has the following significant features:

- Efficient support for all types of audio interfaces, from consumer sound cards to professional multichannel audio interfaces.
- Fully modularized sound drivers.
- SMP and thread-safe design ([PLEASE READ THIS](#)).
- User space library (alsa-lib) to simplify application programming and provide higher level functionality.
- Support for the older Open Sound System (OSS) API, providing binary compatibility for most OSS programs.

[ALSA Project on GitHub \(no releases there\)](#)

Download a package

Current versions

- stable linux kernel
- linux firmware files (GIT)
- alsa-firmware-1.2.4 (sig)
- alsa-lib-1.2.8 (sig)**
- alsa-ucm-conf-1.2.8 (sig)
- alsa-topology-conf-1.2.5.1 (sig)
- alsa-utils-1.2.8 (sig)**
- alsa-tools-1.2.5 (sig)
- alsa-plugins-1.2.7.1 (sig)
- alsa-oss-1.1.8 (sig)
- pyalsa-1.2.7 (sig)
- tinycompress-1.2.8 (sig)

Please, help us to provide better content and organisation on this wiki. Read the [Community Portal](#) for more details.

New ALSA Users

We need users to use, test and provide feedback on drivers and applications using ALSA. If you are interested, please subscribe to a mailing list.

- Is my soundcard supported?
- How do I test my soundcard?

Other information

- Unofficial wiki

Alsa Project News

- 2022-10-25** alsa-lib,alsa-ucm-conf,alsa-utils,tinycompress 1.2.8 release | [Changes v1.2.7.2 v1.2.8](#)
- 2022-07-08** alsa-lib,alsa-ucm-conf 1.2.7.2 release | [Changes v1.2.7.1 v1.2.7.2](#)
- 2022-06-17** alsa-lib,alsa-ucm-conf,alsa-plugins 1.2.7.1 release | [Changes v1.2.7 v1.2.7.1](#)
- 2022-05-31** alsa 1.2.7 release | [Changes v1.2.6.3 v1.2.7](#)
- 2021-12-17** alsa-ucm-conf 1.2.6.3 release | [Changes v1.2.6.2 v1.2.6.3](#)
- 2021-12-10** alsa-ucm-conf 1.2.6.2 release | [Changes v1.2.6.1 v1.2.6.2](#)

Advanced ALSA Users

We need users to use, test and provide feedback on drivers and applications [list](#).

ALSA Developers

We need application developers who choose to use ALSA as the basis for their programs, programmers to work on low level drivers, writers to extend

如上图所示我们下载的版本为：

- alsa-lib-1.2.8.tar.bz2
- alsa-utils-1.2.8.tar.bz2

2.2 ALSA lib编译

ALSA Lib移植不需要修改源码，只需要重新编译库代码以支持自己的平台。

```
tar -xvf alsa-lib-1.0.27.2.tar.bz2
cd alsa-lib-1.0.27.2
CC=arm-none-linux-gnueabi-gcc
./configure --host=arm-linux --prefix=/home/m/3rd/alsa/install/
make
```



```
make install
```

在上述命令中./configure配置的几个重要的配置选项解释如下：

- --host指定编译器，这里指定为交叉编译器，运行本配置命令前务必保证编译器已经可以在Shell下可以直接执行了。
- --prefix指定编译后文件的安装路径，这样安装命令就还会指定的这个目录中创建lib和include两个目录。

2.3 ALSA Util编译

ALSA Util可以生成用于播放，录制，配置音频的应用可执行文件，测试驱动代码时用处很大，编译过程如下：

```
tar -xvf alsa-utils-1.0.27.2.tar.bz2
cd alsa-utils-1.0.27.2
CC=arm-none-linux-gnueabi-gcc
./configure --prefix=/home/m/3rd/alsa/install/ --host=arm-linux --with-alsa-inc-
prefix=/home/m/3rd/alsa/install/include --with-alsa-prefix=/home/m/3rd/alsa/install/lib --
disable-alsamixer --disable-xmlto --disable-nls
make
```

2.4 ALSA库和工具移植入嵌入式平台

ALSA库和测试工具的移植就是将相应库文件和可执行文件放在目标板上，以下文件 必须被拷贝至对应位置：

- (1) ALSA Lib文件，放在/lib/中。
- (2) 配置文件放在/usr/local/share中，与编译时指定的目录相同。
- (3) 测试应用文件，ALSA Util能产生aplay、amixer、arecord，我们可以把这些可执行文件放在/usr/sbin中。
- (4) 内核目录中保证有/dev/snd/目录，这个目录下存放controlC0，pcmC0D0，/usr/sbintimer，timer这些设备文件，如果这些设备文件已经在/dev目录下，可手动拷贝到/snd目录中。

在Linux系统中，每个设备文件都是文件。音频设备也是一样，它的设备文件被放在/dev/snd目录下，我们来看下这些设备文件：

```
ls /dev/snd -l crw-rw----+ 1 root audio 116, 2 5月 19 21:24 controlC0 用于声卡的
```

```
crw-rw-----+ 1 root audio 116, 4 6月 6 19:31 pcmC0D0c
crw-rw-----+ 1 root audio 116, 3 6月 11 11:53 pcmC0D0p
crw-rw-----+ 1 root audio 116, 33 5月 19 21:24 timer
```

- (1) controlC0: 音频控制设备文件，例如通道选择，混音，麦克风的控制等；
- (2) pcmC0D0c: 声卡0设备0的录音设备，c表示capture；
- (3) pcmC0D0p: 声卡0设备0的播音设备，p表示playback；
- (4) timer: 定时器设置。

三.alsa工具应用

1.节点信息

alsa本身提供的基本文件查看路径：

- /proc/asound/cardX/（其中X是声卡号，从0到7）：cardX系统知道的每个声卡都有一个目录：有关此目录内容的信息，请参见下文。
- /proc/asound/cards（只读）：已注册卡的列表
- /proc/asound/dev/：一个目录，列出如果系统使用devfs则程序用于声音操作的特定设备文件，该目录将存在：如果您的系统不使用devfs（从2006-06开始，大多数不使用）：该文件要么不根本存在，或者仅仅是与之的符号链接 /dev/snd
- /proc/asound/devices（只读）：已注册的ALSA设备列表（主设备号= 116）
- /proc/asound/hwdep（只读）：hwdep（硬件依赖）控件的列表未在所有系统上出现（这是否仍然存在？）
- /proc/asound/meminfo（只读）：内存使用情况信息，此proc文件仅在使用内存调试（或完整）选项构建alsa驱动程序时才会显示：当前在内核空间上分配的内存。
- /proc/asound/modules（只读）：已注册的ALSA声卡驱动程序列表，这不是ALSA加载的所有内核模块，这只是：硬件驱动程序的列表。对于使用中的每个声卡，期望在此处看到一行。
- /proc/asound/oss/：包含有关oss仿真的信息的目录，有关此目录内容的信息，请参见下文。
- **/proc/asound/pcm**（只读）：分配的pcm流的列表，请注意，这（可能）并不表示活动流的列表，而是设备的列表。这对于找出hw: 0,0样式的设备非常有用：像aplay这样的命令需要的名称。

- /proc/asound/seq/ : 包含有关音序器信息的目录, 有关此目录内容的信息, 请参见下文。
- /proc/asound/timers (只读) : 类似于/proc/asound/pcm, 它是ALSA知道的计时器列表, 并且描述了: 在该时刻实际使用了哪些计时器。
- /proc/asound/version (只读) : ALSA子系统模块 (或内核) 的版本和日期

比较常见的有以下几个:

1) PCM信息: cat /proc/asound/pcm

```
zhongjun@SH-QTK-D-001757b:/proc/asound$ cat pcm
00-00: ALC3234 Analog : ALC3234 Analog : playback 1 : capture 1
00-02: ALC3234 Alt Analog : ALC3234 Alt Analog : capture 1
00-03: HDMI 0 : HDMI 0 : playback 1
00-07: HDMI 1 : HDMI 1 : playback 1
00-08: HDMI 2 : HDMI 2 : playback 1
00-09: HDMI 3 : HDMI 3 : playback 1
00-10: HDMI 4 : HDMI 4 : playback 1
```

举一个例子, 其中是00-00代表节点名称是: hw:0,0,附带有playback以及capture功能

2) 硬件信息: cat /proc/asound/devices

```
zhongjun@SH-QTK-D-001757b:/proc/asound$ cat devices
1:          : sequencer
2: [ 0- 0]: digital audio playback
3: [ 0- 0]: digital audio capture
4: [ 0- 2]: digital audio capture
5: [ 0- 3]: digital audio playback
6: [ 0- 7]: digital audio playback
7: [ 0- 8]: digital audio playback
8: [ 0- 9]: digital audio playback
9: [ 0-10]: digital audio playback
10: [ 0- 0]: hardware dependent
11: [ 0- 2]: hardware dependent
12: [ 0]   : control
33:       : timer
```

3) 查看cardx的信息, 我们拿hw:0,0节点来说明

查看播放状态, 在没有播放的时候显示这样

```
zhongjun@SH-QTK-D-001757b:/proc/asound$ cat card0/pcm0p/sub0/status
closed
```

在播放的时候显示这样

```
zhongjun@SH-QTK-D-001757b:/proc/asound$ cat card0/pcm0p/sub0/status
state: RUNNING
owner_pid   : 1553
trigger_time: 705336.581116395
tstamp      : 705340.020058659
delay       : 2786
avail       : 85449
avail_max   : 85449
-----
hw_ptr      : 151688
appl_ptr    : 154439
```

查看硬件参数

```
zhongjun@SH-QTK-D-001757b:/proc/asound$ cat card0/pcm0p/sub0/hw_params
access: MMIO_INTERLEAVED
format: S16_LE
subformat: STD
channels: 2
rate: 44100 (44100/1)
period_size: 44100
buffer_size: 88200
```

查看软件参数

```
zhongjun@SH-QTK-D-001757b:/proc/asound$ cat card0/pcm0p/sub0/sw_params
tstamp_mode: ENABLE
period_step: 1
avail_min: 87319
start_threshold: 18446744073709551615
stop_threshold: 6206523236469964800
silence_threshold: 0
silence_size: 0
boundary: 6206523236469964800
```

另外还有一个路径来查看信息

```
zhongjun@SH-QTK-D-001757b:/proc/asound$ ll /sys/class/sound/
total 0
drwxr-xr-x 2 root root 0 4月 2 2020 ./
drwxr-xr-x 77 root root 0 10月 9 2018 ../
lrwxrwxrwx 1 root root 0 12月 6 10:04 card0 -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/
lrwxrwxrwx 1 root root 0 12月 6 10:05 controlC0 -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/controlC0/
lrwxrwxrwx 1 root root 0 12月 6 10:05 ctl-led -> ../../devices/virtual/sound/ctl-led/
lrwxrwxrwx 1 root root 0 12月 6 10:05 hwC0D0 -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/hwC0D0/
lrwxrwxrwx 1 root root 0 12月 6 10:05 hwC0D2 -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/hwC0D2/
lrwxrwxrwx 1 root root 0 12月 6 10:05 pcmC0D0c -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/pcmC0D0c/
lrwxrwxrwx 1 root root 0 12月 6 10:05 pcmC0D0p -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/pcmC0D0p/
lrwxrwxrwx 1 root root 0 12月 6 10:05 pcmC0D10p -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/pcmC0D10p/
lrwxrwxrwx 1 root root 0 12月 6 10:05 pcmC0D2c -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/pcmC0D2c/
lrwxrwxrwx 1 root root 0 12月 6 10:05 pcmC0D3p -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/pcmC0D3p/
lrwxrwxrwx 1 root root 0 12月 6 10:05 pcmC0D7p -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/pcmC0D7p/
lrwxrwxrwx 1 root root 0 12月 6 10:05 pcmC0D8p -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/pcmC0D8p/
lrwxrwxrwx 1 root root 0 12月 6 10:05 pcmC0D9p -> ../../devices/pci0000:00/0000:00:1f.3/sound/card0/pcmC0D9p/
lrwxrwxrwx 1 root root 0 12月 6 10:05 seq -> ../../devices/virtual/sound/seq/
lrwxrwxrwx 1 root root 0 12月 6 10:05 timer -> ../../devices/virtual/sound/timer/
```

后缀名c代表capture，后缀名p代表playback

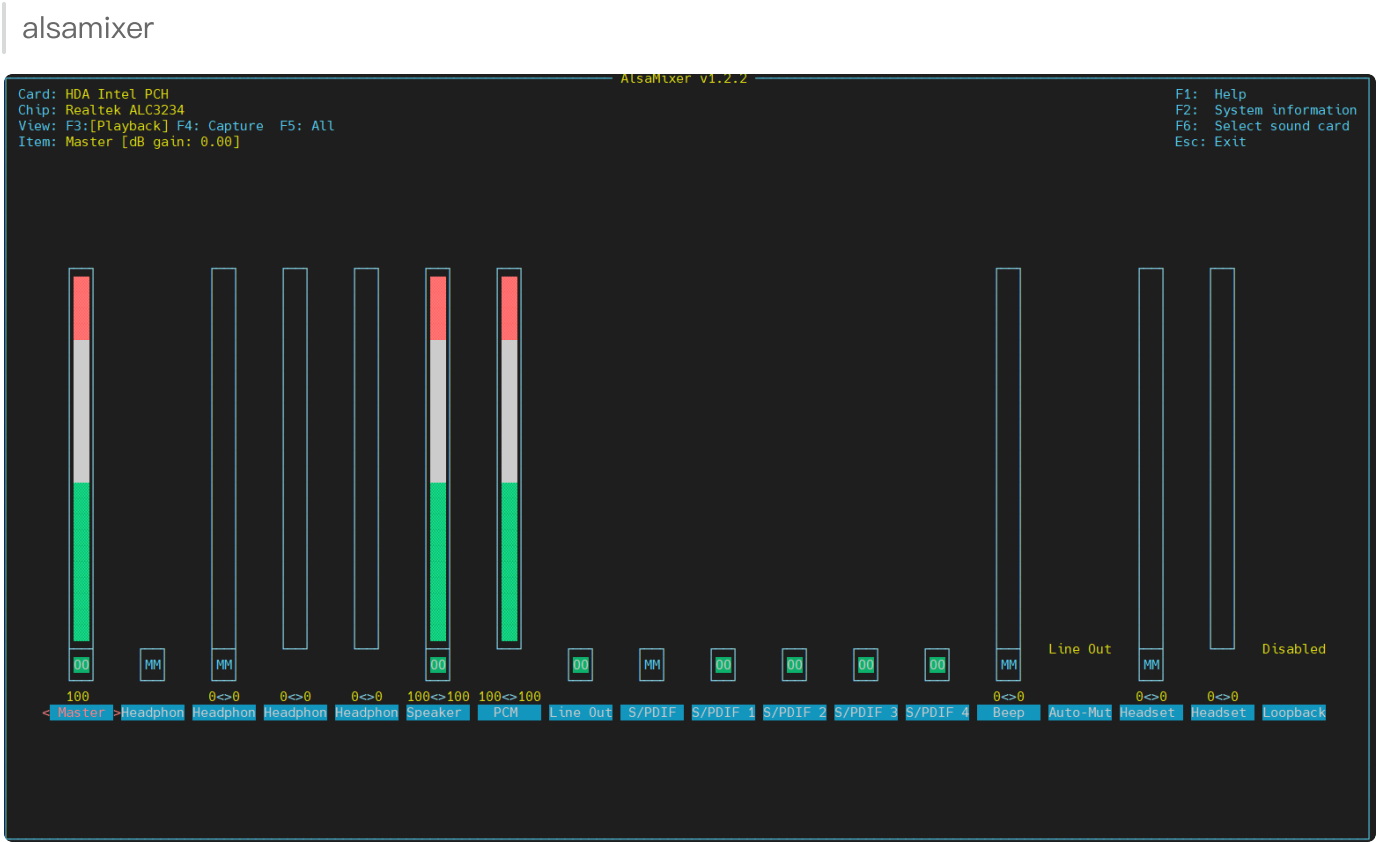
所以hwC0D0c代表是hw:0,0是capture功能，hwC0D0p是hw:0,0是playback功能

2.工具使用

一共有4个工具：alsamixer/amixer/aplay/arecord

2.1 alsamixer

这个主要是UI工具，可以针对调整音量,直接在命令行敲



2.2 aplay

使用方法是在命令行中直接敲aplay

```
zhongjun@SH-OTK-D-001757b:~/project/pcm_music$ aplay
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
-s, --samples=#          interrupt after # samples per channel
-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 PCM stop is # microseconds from xrun
-v, --verbose            show PCM structure and setup (accumulative)
-V, --vumeter=TYPE       enable VU meter (TYPE: mono or stereo)
-I, --separate-channels  one file for each channel
-i, --interactive        allow interactive operation from stdin
-m, --chmap=ch1,ch2,...  Give the channel map to override or follow
--disable-resample       disable automatic rate resample
--disable-channels       disable automatic channel conversions
--disable-format          disable automatic format conversions
--disable-softvol        disable software volume control (softvol)
--test-position          test ring buffer position
--test-coef=#            test coefficient for ring buffer position (default 8)
                        expression for validation is: coef * (buffer_size / 2)
--test-nowait            do not wait for ring buffer - eats whole CPU
--max-file-time=#        start another output file when the old file has recorded
                        for this many seconds
--process-id-file        write the process ID here
--use-strftime            apply the strftime facility to the output file name
--dump-hw-params          dump hw_params of the device
--fatal-errors           treat all errors as fatal

Recognized sample formats are: S8_U8 S16_LE S16_BE U16_LE U16_BE S24_LE S24_BE U24_LE U24_BE S32_LE S32_BE U32_LE U32_BE FLOAT_LE FLOAT_BE FLOAT64_LE FLOAT64_BE IEC959_SUBFRAME_LE IEC959_SUBFRAME_BE MU_LAW A_LAW TMA_APCM M
PEG_GSM S20_LE S20_BE U20_LE U20_BE SPECIAL S24_3LE S24_3BE U24_3LE U24_3BE S20_3LE S20_3BE U20_3LE U20_3BE S18_3LE S18_3BE U18_3LE U18_3BE G723_24 G723_24_1B G723_40 G723_40_1B DSD_U8 DSD_U16_LE DSD_U32_LE DSD_U16_BE DSD_U
32_BE
Some of these may not be available on selected hardware
The available format shortcuts are:
-f cd (16 bit little endian, 44100, stereo)
-f cdr (16 bit big endian, 44100, stereo)
-f dat (16 bit little endian, 48000, stereo)
zhongjun@SH-OTK-D-001757b:~/project/pcm_music$
```

常见的选项有：

选项	功能
-D,--device	指定声卡设备, 默认使用 default
-l,--list-devices	列出当前所有声卡
-t,--file-type	指定播放文件的格式, 如 voc,wav,raw, 不指定的情况下会去读取文件头部作识别
-c,--channels	指定通道数
-f,--format	指定采样格式
-r,--rate	采样率
-d,--duration	指定播放的时间
--period-size	指定 period size
--buffer-size	指定 buffer size

举例：

```
| aplay -D hw:0,0 -f S16_LE -r 44100 -c 2 1.pcm
```

2.3 arecord

使用方法是在命令行中直接敲arecord

```
zhongjun@SH-QTK-D-001757b:~/project/pcm_music$ arecord
Usage: arecord [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
-e, --samples=#           interrupt after # samples per channel
-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 PCM stop is # microseconds from xrun
-v, --verbose             show PCM structure and setup (accumulative)
-V, --vumeter=TYPE        enable VU meter (TYPE: mono or stereo)
-I, --separate-channels   one file for each channel
-i, --interactive         allow interactive operation from stdin
-m, --chmap=ch1,ch2,...   Give the channel map to override or follow
--disable-resample        disable automatic rate resample
--disable-channels        disable automatic channel conversions
--disable-format          disable automatic format conversions
--disable-softvol         disable software volume control (softvol)
--test-position           test ring buffer position
--test-coef=#             test coefficient for ring buffer position (default 0)
                           expression for validation is: coef * (buffer size / 2)
--test-nowait             do not wait for ring buffer - eats whole CPU
--max-file-time=#         start another output file when the old file has recorded
                           for this many seconds
--process-id-file         write the process ID here
--use-strftime            apply the strftime facility to the output file name
--dump-hw-params          dump hw_params of the device
--fatal-errors            treat all errors as fatal

Recognized sample formats are: S8 U8 S16_LE S16_BE U16_LE U16_BE S24_LE S24_BE U24_LE U24_BE S32_LE S32_BE U32_LE U32_BE FLOAT_LE FLOAT_BE FLOAT64_LE FLOAT64_BE IEC958_SUBFRAME_LE IEC958_SUBFRAME_BE MU_LAW A_LAW IMA_ADPCM M
PEG GSM S20_LE S20_BE U20_LE U20_BE SPECIAL S24_3LE S24_3BE U24_3LE U24_3BE S20_3LE S20_3BE U20_3LE U20_3BE S18_3LE S18_3BE U18_3LE U18_3BE G723_24 G723_24_1B G723_40 G723_40_1B DSD_U8 DSD_U16_LE DSD_U32_LE DSD_U
32_BE
Some of these may not be available on selected hardware
The available format shortcuts are:
-f cd (16 bit little endian, 44100, stereo)
-f cdr (16 bit big endian, 44100, stereo)
-f dat (16 bit little endian, 48000, stereo)
```

常见的选项有：

选项	功能
-D,--device	指定声卡设备, 默认使用 default
-l,--list-devices	列出当前所有声卡
-t,--file-type	指定播放文件的格式, 如 voc,wav,raw, 不指定的情况下会去读取文件头部作识别
-c,--channels	指定通道数
-f,--format	指定采样格式
-r,--rate	采样率
-d,--duration	指定播放的时间
--period-size	指定 period size
--buffer-size	指定 buffer size

举例：

```
| arecord -Dhw:0,0 -f S16_LE -r 16000 -c 2 -d 5 ./1.wav
```

2.4 补充问题点

此小节通过一个问题点来引入问题：

问题：为什么节点是default可以播放/录音呢?在设备信息中也没有看到default节点呢?

```
zhongjun@SH-QTK-D-001757b:~/project/pcm_music$ aplay -D default -f S16_LE -r 44100 -c 2 1.pcm
Playing raw data '1.pcm' : Signed 16 bit Little Endian, Rate 44100 Hz, Stereo
```

答案：这个是alsa plugins的概念，因为我们在 /usr/share/alsa/alsa.conf 配置了default的节点，详细信息参照https://www.alsa-project.org/alsa-doc/alsa-lib/pcm_plugins.html,

```
defaults.pcm.card 0  
defaults.pcm.device 0  
defaults.pcm.subdevice -1
```

These defaults can be freely overwritten in local configuration files.

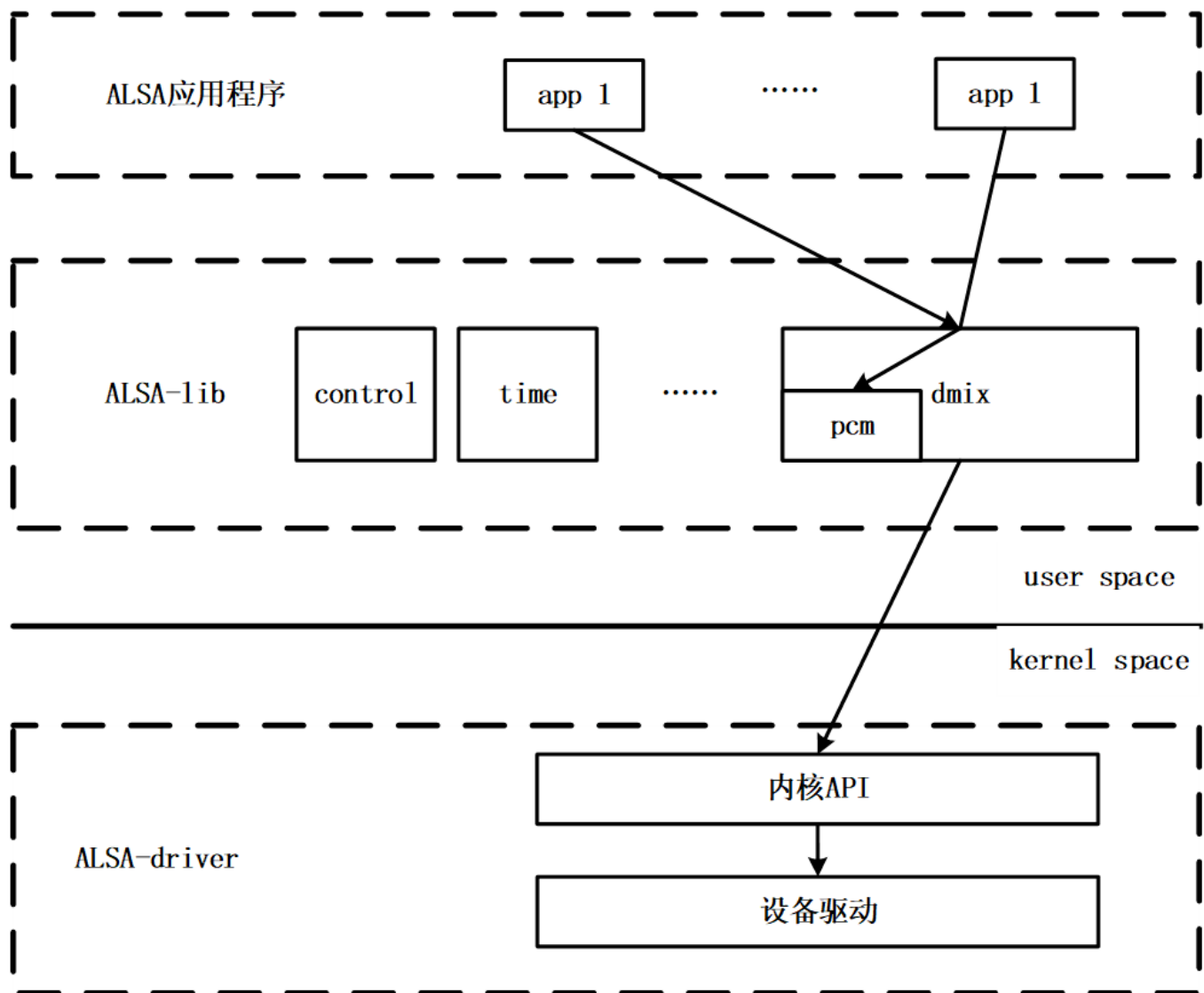
我们来看下alsa.conf文件：

```
defaults.ctl.card 0  
defaults.pcm.card 0  
defaults.pcm.device 0
```

四.alsa编程

alsa app编程的参考网站是：https://www.alsa-project.org/alsa-doc/alsa-lib/group___p_c_m.html

从代码角度体现了alsa-lib和alsa-driver及hardware的交互关系。用户层的alsa-lib通过操作alsa-driver创建的设备文件/dev/snd/pcmC0D0p等对内核层进行访问。内核层的alsa-driver驱动再经由sound core对硬件声卡芯片进行访问。



主要牵扯到接口为：

函数名	解释
snd_pcm_open	
snd_pcm_info	
snd_pcm_hw_params_any	
snd_pcm_hw_params_set_access	
snd_pcm_hw_params_set_format	
snd_pcm_hw_params_set_channels	
snd_pcm_hw_params_set_rate_near	
snd_pcm_hw_params_set_buffer_size_near	
snd_pcm_hw_params	
snd_pcm_sw_params_current	
snd_pcm_sw_params	
snd_pcm_readi	
snd_pcm_writeti	
snd_pcm_close	

我们就不来一一列出API的使用方法了，自己看我们前面的链接也能查到，我们来直接说下使用

1.Audio使用的伪代码为

C | 复制代码

```

1      open_the_device();
2      set_the_parameters_of_the_device();
3  while (!done) {
4          /* one or both of these */
5          receive_audio_data_from_the_device();
6          deliver_audio_data_to_the_device();
7      }
8      close the device
9

```

2.最小的playback使用程序

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <alsa/asoundlib.h>
4
5  int main (int argc, char *argv[])
6  {
7      int i;
8      int err;
9      short buf[128];
10     snd_pcm_t *playback_handle;
11     snd_pcm_hw_params_t *hw_params;
12
13     if ((err = snd_pcm_open (&playback_handle, argv[1], SND_PCM_STREAM_PLA
YBACK, 0)) < 0) {
14         fprintf (stderr, "cannot open audio device %s (%s)\n",
15                 argv[1],
16                 snd_strerror (err));
17         exit (1);
18     }
19
20     if ((err = snd_pcm_hw_params_malloc (&hw_params)) < 0) {
21         fprintf (stderr, "cannot allocate hardware parameter structure (%
s)\n",
22                 snd_strerror (err));
23         exit (1);
24     }
25
26     if ((err = snd_pcm_hw_params_any (playback_handle, hw_params)) < 0) {
27         fprintf (stderr, "cannot initialize hardware parameter structure
(%s)\n",
28                 snd_strerror (err));
29         exit (1);
30     }
31
32     if ((err = snd_pcm_hw_params_set_access (playback_handle, hw_params, S
ND_PCM_ACCESS_RW_INTERLEAVED)) < 0) {
33         fprintf (stderr, "cannot set access type (%s)\n",
34                 snd_strerror (err));
35         exit (1);
36     }
37
38     if ((err = snd_pcm_hw_params_set_format (playback_handle, hw_params, S
ND_PCM_FORMAT_S16_LE)) < 0) {
39         fprintf (stderr, "cannot set sample format (%s)\n",
40                 snd_strerror (err));
```

```

41         exit (1);
42     }
43
44     if ((err = snd_pcm_hw_params_set_rate_near (playback_handle, hw_params
45 , 44100, 0)) < 0) {
46         fprintf (stderr, "cannot set sample rate (%s)\n",
47             snd_strerror (err));
48         exit (1);
49     }
50
51     if ((err = snd_pcm_hw_params_set_channels (playback_handle, hw_params
52 , 2)) < 0) {
53         fprintf (stderr, "cannot set channel count (%s)\n",
54             snd_strerror (err));
55         exit (1);
56     }
57
58     if ((err = snd_pcm_hw_params (playback_handle, hw_params)) < 0) {
59         fprintf (stderr, "cannot set parameters (%s)\n",
60             snd_strerror (err));
61         exit (1);
62     }
63
64     snd_pcm_hw_params_free (hw_params);
65
66     if ((err = snd_pcm_prepare (playback_handle)) < 0) {
67         fprintf (stderr, "cannot prepare audio interface for use (%s)\n",
68             snd_strerror (err));
69         exit (1);
70     }
71
72     for (i = 0; i < 10; ++i) {
73         if ((err = snd_pcm_writei (playback_handle, buf, 128)) != 128) {
74             fprintf (stderr, "write to audio interface failed (%s)\n",
75                 snd_strerror (err));
76             exit (1);
77         }
78     }
79
80     snd_pcm_close (playback_handle);
81     exit (0);
82 }

```

3.最小的capture程序

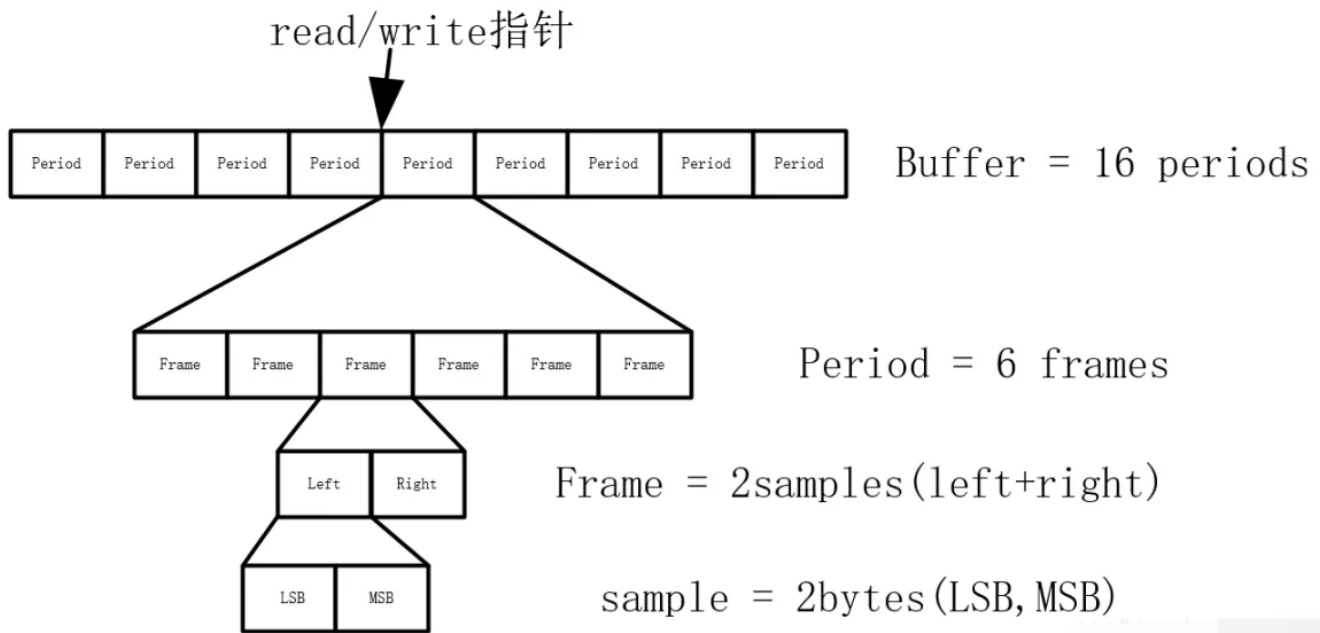
```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <alsa/asoundlib.h>
4
5  int main (int argc, char *argv[])
6  {
7      int i;
8      int err;
9      short buf[128];
10     snd_pcm_t *capture_handle;
11     snd_pcm_hw_params_t *hw_params;
12
13     if ((err = snd_pcm_open (&capture_handle, argv[1], SND_PCM_STREAM_CAPTURE, 0)) < 0) {
14         fprintf (stderr, "cannot open audio device %s (%s)\n",
15                 argv[1],
16                 snd_strerror (err));
17         exit (1);
18     }
19
20     if ((err = snd_pcm_hw_params_malloc (&hw_params)) < 0) {
21         fprintf (stderr, "cannot allocate hardware parameter structure (%s)\n",
22                 snd_strerror (err));
23         exit (1);
24     }
25
26     if ((err = snd_pcm_hw_params_any (capture_handle, hw_params)) < 0) {
27         fprintf (stderr, "cannot initialize hardware parameter structure (%s)\n",
28                 snd_strerror (err));
29         exit (1);
30     }
31
32     if ((err = snd_pcm_hw_params_set_access (capture_handle, hw_params, SND_PCM_ACCESS_RW_INTERLEAVED)) < 0) {
33         fprintf (stderr, "cannot set access type (%s)\n",
34                 snd_strerror (err));
35         exit (1);
36     }
37
38     if ((err = snd_pcm_hw_params_set_format (capture_handle, hw_params, SND_PCM_FORMAT_S16_LE)) < 0) {
39         fprintf (stderr, "cannot set sample format (%s)\n",
40                 snd_strerror (err));
```

```

41         exit (1);
42     }
43
44     if ((err = snd_pcm_hw_params_set_rate_near (capture_handle, hw_params
45 , 44100, 0)) < 0) {
46         fprintf (stderr, "cannot set sample rate (%s)\n",
47                 snd_strerror (err));
48         exit (1);
49     }
50
51     if ((err = snd_pcm_hw_params_set_channels (capture_handle, hw_params,
52 2)) < 0) {
53         fprintf (stderr, "cannot set channel count (%s)\n",
54                 snd_strerror (err));
55         exit (1);
56     }
57
58     if ((err = snd_pcm_hw_params (capture_handle, hw_params)) < 0) {
59         fprintf (stderr, "cannot set parameters (%s)\n",
60                 snd_strerror (err));
61         exit (1);
62     }
63
64     snd_pcm_hw_params_free (hw_params);
65
66     if ((err = snd_pcm_prepare (capture_handle)) < 0) {
67         fprintf (stderr, "cannot prepare audio interface for use (%s)\n",
68                 snd_strerror (err));
69         exit (1);
70     }
71
72     for (i = 0; i < 10; ++i) {
73         if ((err = snd_pcm_readi (capture_handle, buf, 128)) != 128) {
74             fprintf (stderr, "read from audio interface failed (%s)\n",
75                     snd_strerror (err));
76             exit (1);
77         }
78     }
79
80     snd_pcm_close (capture_handle);
81     exit (0);
82 }

```

另外要想更好的设置硬件参数，需要了解一下几个概念



Sample: 样本长度，音频数据最基本的单位，常见的有8位和16位。

Channel: 声道数分为单声道mono和立体声stereo。

Frame: 帧，构成一个完整的声音单元， $\text{Frame} = \text{Sample} * \text{channel}$ 。

Rate: 又称Sample rate，采样率，即每秒的采样次数，针对帧而言。

Interleaved: 交错模式，一种音频数据的记录方式，在交错模式下，数据以连续帧的形式存放，即首先记录完帧1的左声道样本和右声道样本(假设为立体声)，再开始帧2的记录。而在非交错模式下，首先记录的是一个周期内所有帧的左声道样本，再记录右声道样本，数据是以连续通道的方式存储。多数情况下使用交错模式。

Period size: 周期，每次硬件中断处理音频数据的Frame个数，对于音频设备的数据读写，单位是Frame。

Buffer size: 数据缓冲区大小，这里特指runtime的buffer size，而不是snd_pcm硬件定义的buffer_bytes_max。一般来说 $\text{Buffer size} = \text{period_size} * \text{period_count}$ ，period_count相当于处理完一个buffer数据所需的硬件中断次数。单位也是Frames

参考文章：

1. <https://zhuanlan.zhihu.com/p/443728870>
2. https://www.alsa-project.org/alsa-doc/alsa-lib/group__p_c_m.html
3. <http://equalarea.com/paul/alsa-audio.html>

4. <https://blog.csdn.net/tq08g2z/article/details/125023630>
5. https://blog.51cto.com/u_11626714/4969946
6. <https://blog.csdn.net/SlowIsFastLemon/article/details/108777174>