UmDevAudio, Tutorial and examples

Davide Berardi

14 settembre 2013

1 Introduction

UmDevAudio is a UmView[1] module compatible with tinyalsa[2][3].
it can be used for "extending" the actual audio processor like pulseaudio and jack do, but with more control over the real sound card.
for example we can use it to record the stream of data directed to the sound-card; so we can record an audio stream which is playing inside a browser.

2 UmDevAudio modules

UmDevAudio is made of two modules, the control module (umdevaudioCtl) and the pcm module (umdevaudioPcm);
the first module is used to control the mixer and the various card-relative options.
the second one is used to control the real data streams and the playing/recording relative options.

3 Preparing the environment

first of all you'll need UmDevAudio.

```
you@pc:~$ git clone umdevaudio
```

[1] http://wiki.virtualsquare.org/wiki/index.php/UMview
[2] ALSA and OSS support is still experimental
[3] http://github.com/tinyalsa/tinyalsa

then you must cd inside the new directory (for example UMDEVAUDIO)

```
you@pc:~$ cd ~/UMDEVAUDIO
```

then you must place the umview source code inside a directory called "umview" inside the root of the program.
a simple entry in the Makefile is thinked to get the last experimental branch

```
you@pc:~/UMDEVAUDIO$ make getumview
```

(or you can just modify the variable UMVIEW in the Makefile) then you can continue with a simple make all (the makefile is a wrapper that will do the autoreconf -i; ./configure; make all cycle)

```
you@pc:~/UMDEVAUDIO$ make all
```

the next step is installing umview in the system, that is simply accomplished by using again the wrapper Makefile

```
you@pc:~/UMDEVAUDIO$ sudo make install
```

4 Mounting and executing

in order to mount and using our virtual sound card we first need to spawn a virtual environment using

```
you@pc:~$ umview bash
```

and to add the umdev service

```
you@pc:~$ um_add_service umdev
```

now we can mount our modules on the designated card with

```
you@pc:~$ mount -t umdevaudioCtl [-o card:<cardNo>] none\
        /dev/snd/controlC<cardNo>
you@pc:~$ mount -t umdevaudioCtl [-o card:<cardNo>] none\
        /dev/snd/pcmC<cardNo>D<deviceNo><'p'or'c'>
```

for example we can mount on card 2 with the command

```
you@pc:~$ mount -t umdevaudioCtl -o card:2 none\
        /dev/snd/controlC2
you@pc:~$ mount -t umdevaudioCtl -o card:2 none\
        /dev/snd/pcmC2D0p
```

umdevaudio have some options to configure the virtual soundcard like:

- card:cardNo (like card:2)
- version:soundProtocolVersion (like version:2.0.10, for alpha test for soundcards)
- controls:number (like controls:4 for 4 mixers)
- playcap:'p'or'c' (like playcap:p for playback devices)
- configrc:path (like config: /.umaudiorc for configure the min/max framerate etc)
- file:path (like file: /test.raw to set the input/output file)

5 VirtualSuite

Virtual Suite is a simple script to simplifying the mount process of all required modules of umdevaudio.
to use it simply use

```
you@pc:~/UMDEVAUDIO$ ./VirtualSuite [cardno][options]
```

where options can be one or more of the followings

- -v / –protocol sound protocol p.e. 2.0.2
- -m / –mixer-size number of control's mixer
- -c / –control control module mount path
- -p / –pcm-path pcm module mount path
- –virtual-conf virtual configuration file (contains options like min/max framerate etc)
- -P / –playcap pcm is mounted as playback or capture
- -o / –outfile path to the file on which the stream is recorded

6 Examples

6.1 record the audio stream in a raw file

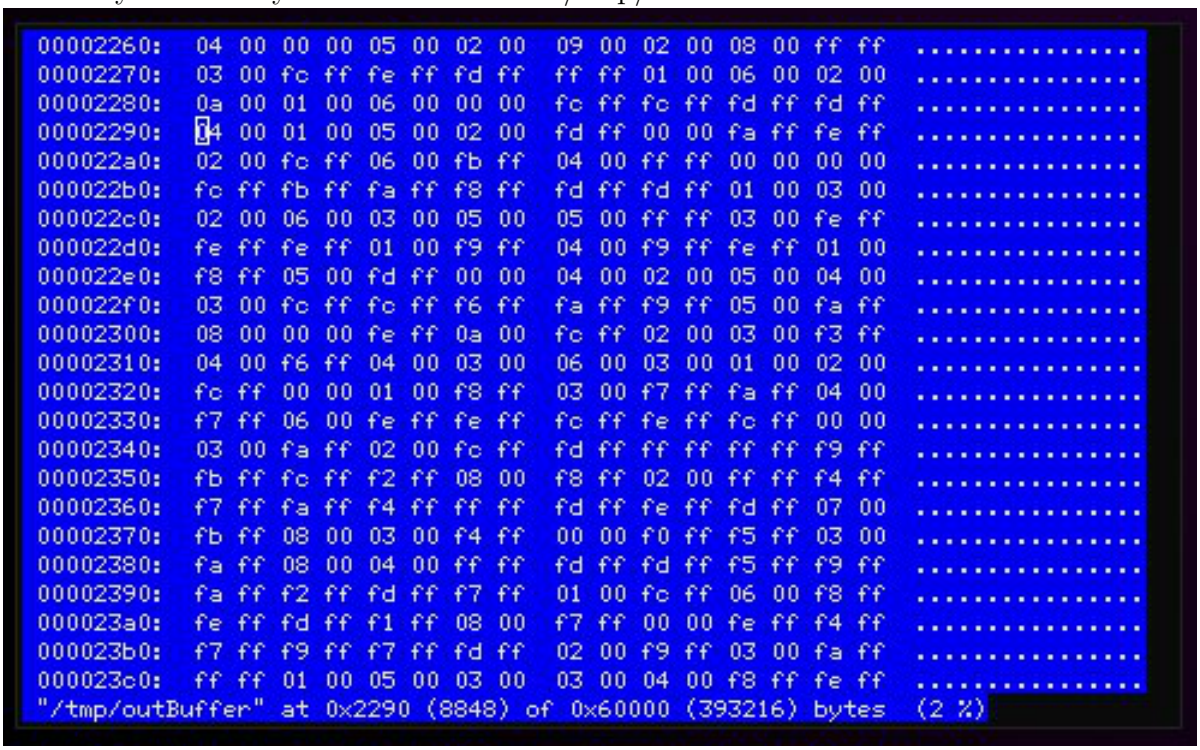
you'll start with initializing the environment with mounting our virtual card as the card 2 (for example)

```
you@pc:~/UMDEVAUDIO$ ./VirtualSuite 2 -P p
you@umview ~/UMDEVAUDIO =>
```

next you can play the stream using tinyplay

```
you@umview ~/UMDEVAUDIO => tinyplay <wavfile> -D 2 0
```

now you'll have your recorded file on /tmp/outBuffer



so you can just use aplay (outside the virtual machine, or on your real card!) to play it

```
you@pc:~$ aplay -M -t raw -f cd -r 44100 /tmp/outBuffer
```

6.2 put a file as the sound recorded from the mic

you'll start with initializing the environment with mounting our virtual card as the card 2 (for example)

```
you@pc:~/UMDEVAUDIO$ ./VirtualSuite 2 -P c
you@umview ~/UMDEVAUDIO =>
```

next you can "record" the input raw file (by default /tmp/outBuffer) with tinycap

```
you@umview ~/UMDEVAUDIO => tinycap <out wav file> -D 2 -d 0
```

so now you'll have your output file recorded.
you can test it with aplay (on your real card!)

```
you@pc:~$ aplay <out wav file>
```