# JACK (JACK Audio Connection Kit) and related tools

Vicente González Ruiz

July 21, 2016

## Contents



# Chapter 1
# The server

### 1.1 What is JACK?

- JACK is a sound system (audio server) for POSIX operating systems (such as Linux, FreeBSD, MacOS X and Windows XP).
- It is a software layer between the kernel (ALSA) and the applications that requires to access to the sound hardware.
- Licensed under LGPL.

### 1.2 Functions

- Controllable latency (suitable real-time audio applications).
- Allow to define the connections (audio flows) between JACK-client applications (like in a real mixer desk).
- Transport control synchronization. JACK can send timming events (using the JACK transport control system) to the rest of clients in order to synchronize them.
- MIDI capabilities.

### 1.3 Installation

1. Install the program `qjackctl`. This a graphical front-end for controlling the audio server `jackd2`. During the installation of `jackd` you must decide to enable the realtime process priority (necesarry for real-time processing in slow systems). The priority is defined in the file:

```
$ cat /etc/security/limits.d/audio.conf
# Provided by the jackd package.
#
# Changes to this file will be preserved.
#
# If you want to enable/disable realtime permissions, run
#
```

```
#    dpkg-reconfigure -p high jackd

@audio   -  rtprio     95
@audio   -  memlock    unlimited
#@audio   -  nice      -19
```

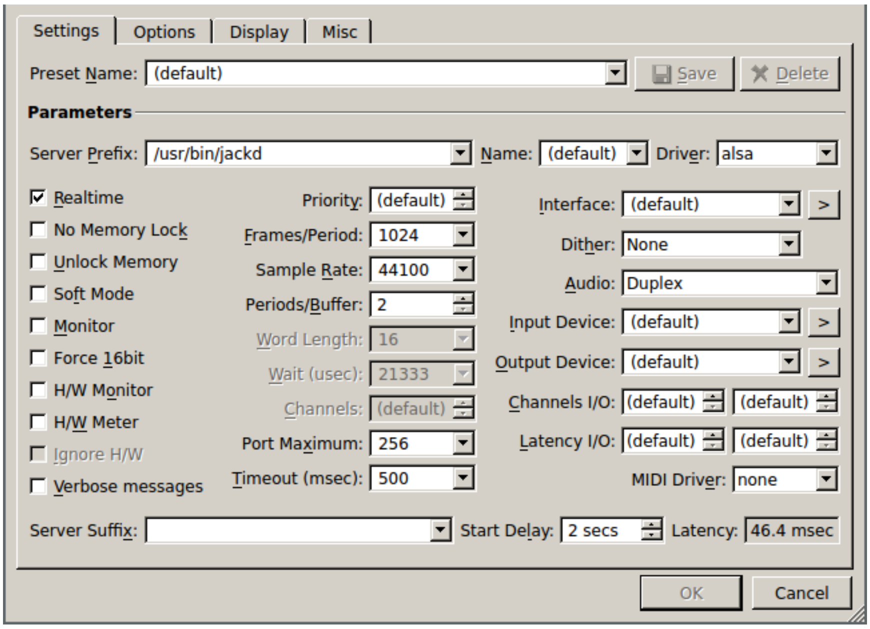
### 1.4 JACK graphical front-end: qjackctl

- Provides a simple GUI dialog for setting several JACK daemon parameters, which are properly saved between sessions, and a way control of the status of the audio server daemon. Run it with:
  `$ qjackctl &`


- Control actions:
  1. `Start:` the server.
  2. `Stop:` the server.
  3. `Quit:` kill the server.
  4. `Messages:` from the server.
  5. `Session:` show/hide the session manager window.
  6. `Setup:` the server.
  7. `Connect:` the audio applications. Notice that all connections made in the Connections interface are kept as long you don't power-cycle the JACK server (jackd). That is, all connections will be lost when the JACK server or any of the client application programs are closed or terminated.
  8. `Patchbay:` will keep all declared connections automatically, as long as QjackCtl is kept alive. Moreover, you can declare typical connection configuration that are carried out when the related clients are executed.
  9. Play: start transport timming events.
  10. Pause: stop transport timming events.
  11. Forward: transport timming events.
  12. Backward: transport timming event.
  13. Rewind: transport timming event.

#### 1.4.1 Server configuration

- Select the Setup form from the `qjackctl` application. Clicking in the "Settings" tab, you will obtain something like:




- Among others, the Setup window allows you to configure important settings:
  1. **Driver:** the physical audio driver. On Linux, the default option is ALSA.
  2. **Realtime:** although this option is selected as the default, it requires a Real-time capable system to use. Unless you have configured your operating system for Real-time, uncheck this option.
  3. **Interface:** select the physical audio device that you would like the Jack server to communicate with (for example, a FireWire or USB interface or the built-in audio of your computer). Currently, JACK can only communicate with one hardware audio device at a time.
  4. **Frames/Period:** choose your desired audio buffer size (in samples=frames). This paremeter controls the Latency.
  5. **Sample Rate:** choose your desired sample rate for the Jack server and the rest of clients.
- Once you have chosen your settings, click on OK to exit the Setup window. Note that however the new settings will be saved, you will have to restart the JACK Server for the changes to take effect.

### 1.5 Minimizing the number of XRUNs

- When we play live music with a computer, it is very important a low latency. This aspect can be controlled in the Setup section of JACK modifying the buffer size.
- However, Linux is a multi-task OS and JACK competes for the same resources that other applications. Therefore, when Jack does not gets enough CPU a XRUN is provoked and the playback stutters.
- In order to minimize the number of XRUNs we should realise that:

  1. Run only the critical applications.
  2. The JACK server is running with the highest priority[1] . The current priority can be determined by:

```
X=‘ps –C jackd –o pid=‘; ps –o nice –p $X
```

And incremented with:

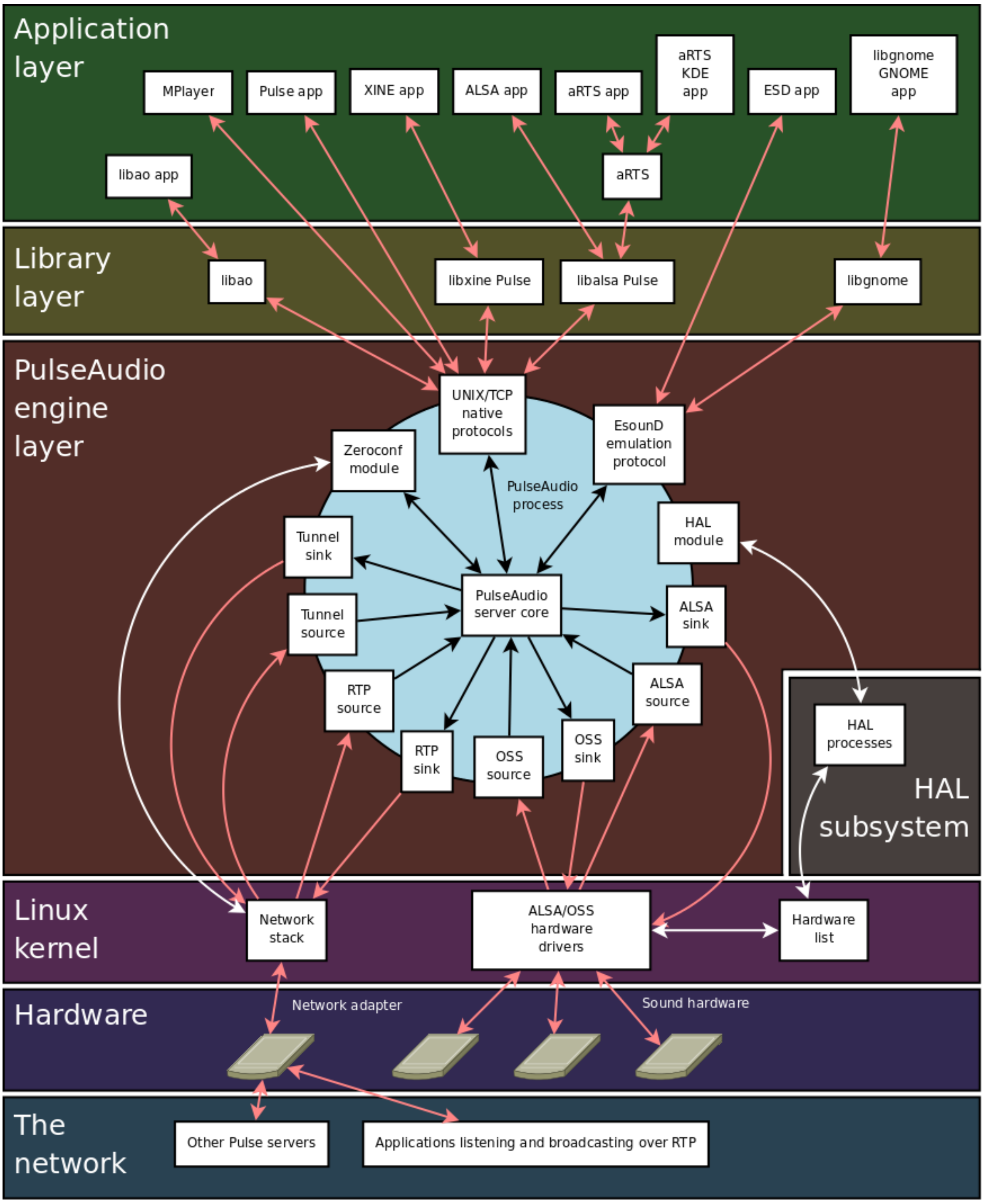
```
X=‘ps -C jackd -o pid=‘; sudo renice -20 -p $X
# Notice that renice only runs with administrator privileges!
```

3. Configure, compile and boot a [real-time kernel](#).
4. Improve your sound hardware.
5. Improve your computer hardware.

# Chapter 2 Living with PulseAudio

## 2.1 What is [Pulseaudio?](#)

- Pulseaudio is a sound system (audio server) for POSIX operating systems (such as GNU/Linux, Solaris, FreeBSD, NetBSD, MacOS X, Windows 2000 and Windows XP).
- It is a software layer between the kernel (ALSA) and the applications that requires to access to the sound hardware.
- [Architecture](#):




- Licensed under LGPL 2.1+.
- Currently, PulseAudio (PA) is used in most friendly Linux distributions to multiplex the sound hardware between the applications that use it.
- The server and clients does not need to run in the same host. This means that one application that run in a remote host can play sound in the local host of the user (that runs the server).

### 2.2 Functions

1. Software mixing of multiple audio streams, bypassing any restrictions the hardware has.
2. Network transparency, allowing an application to play back or record audio on a different machine than the one it is running on.
3. Sound API abstraction, alleviating the need for multiple backends in applications to handle the wide diversity of sound systems out there.
4. Generic hardware abstraction, giving the possibility of doing things like individual volumes per application.

### 2.3 Daemon control

- In the unlikely case that PulseAudio is not automatically started upon entering X, it can can be started with:
```
ps aux | grep pulse
...
vruiz     2201  0.0  0.2 484192  5728 ?        Sl   Sep17  32:18 /usr/bin/pulseaudio --start --log-target=syslog
vruiz     2204  0.0  0.0  98392   508 ?        S    Sep17   0:00 /usr/lib/pulseaudio/pulse/gconf-helper
vruiz    20709  0.0  0.0  11748   888 pts/13   S+   17:13   0:00 grep pulse
...

pulseaudio --start   # On the whole system
start-pulseaudio-x11 # In a X session (usually inside ~/.xinitrc)
```
  and stopped with:
```
pulseaudio --kill
```
- A more accurate control can be obtained with `pactl`. More info at `man pactl`.

### 2.4 Pulseaudio and ALSA tools

- By default, in most OSs, ALSA tools (such as `arecord`) has been configured to access to the sound hardware thought Pulseaudio.

### 2.5 Bypassing PA server (and JACK)

```
pasuspender mplayer -ao alsa * &
```

### 2.6 Sound system usage

```
$ fuser -v /dev/snd/*
                     USER        PID ACCESS COMMAND
/dev/snd/controlC0:  vruiz      2201 F.... pulseaudio
                     vruiz      3810 F.... alsamixer
/dev/snd/pcmC0D0p:   vruiz      2201 F...m pulseaudio
```

### 2.7 Configuration

PA is configured by modifiying the files:

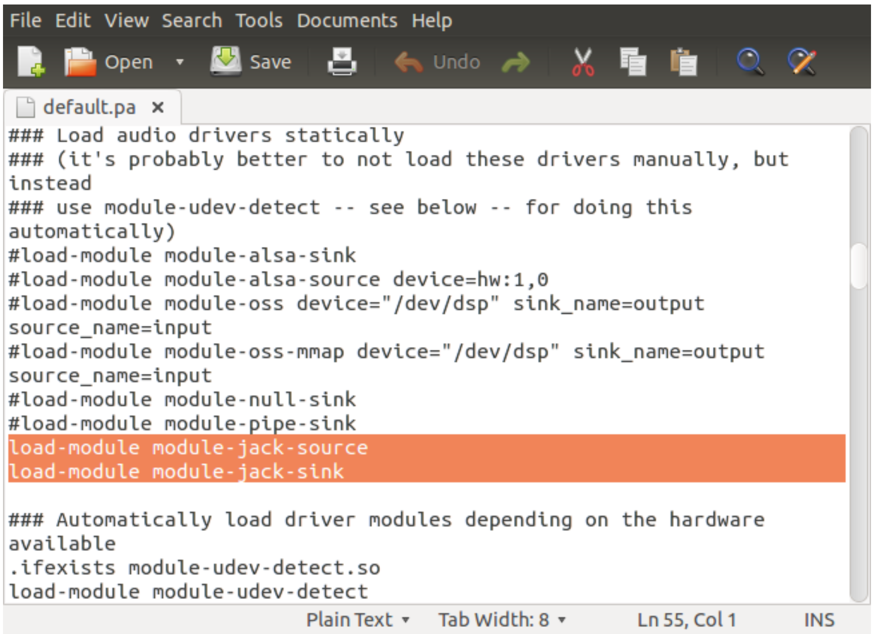
```
/etc/pulse/daemon.conf
/etc/pulse/default.pa
```

### 2.8 Volume control

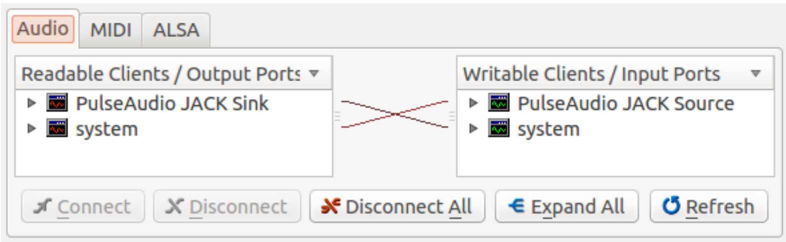
```
$ pavucontrol &
```

### 2.9 Module `pulseaudio-module-jack`

```
# Install the PulseAudio's module for Jack
sudo apt-get install pulseaudio-module-jack

# Go to:
cd /etc/pulse/

# Make a backup of the file:
sudo cp default.pa default.pa.old

# Edit it:
sudo gedit default.pa

# Add:
#
# load-module module-jack-source
# load-module module-jack-sink
#
# after "load-module module-pipe-sink".
#
# Remember to remove the '#' of these lines!!

# Kill and reload JACK.
pulseaudio --kill
```



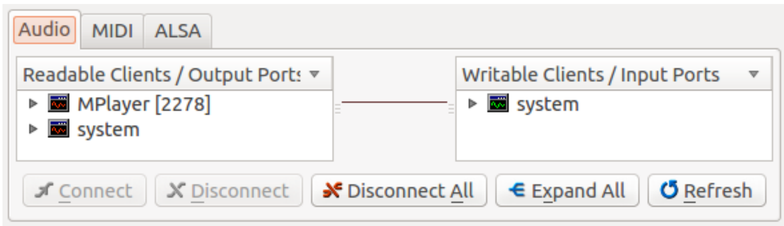

```
### Load audio drivers statically
### (it's probably better to not load these drivers manually, but
instead
### use module-udev-detect -- see below -- for doing this
automatically)
#load-module module-alsa-sink
#load-module module-alsa-source device=hw:1,0
#load-module module-oss device="/dev/dsp" sink_name=output
source_name=input
#load-module module-oss-mmap device="/dev/dsp" sink_name=output
source_name=input
#load-module module-null-sink
#load-module module-pipe-sink
load-module module-jack-source
load-module module-jack-sink

### Automatically load driver modules depending on the hardware
available
.ifexists module-udev-detect.so
load-module module-udev-detect
```

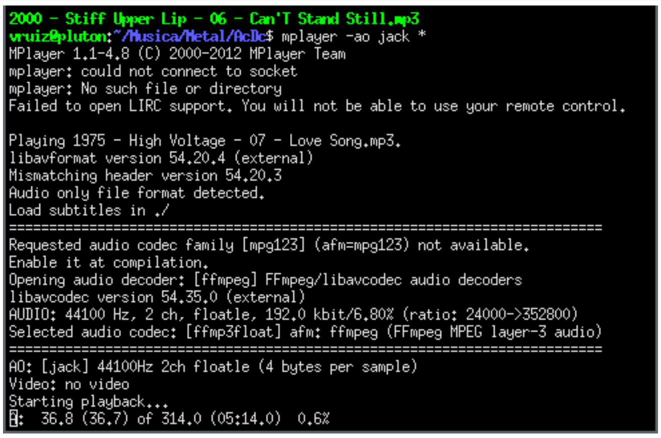



# Chapter 3
# Media players

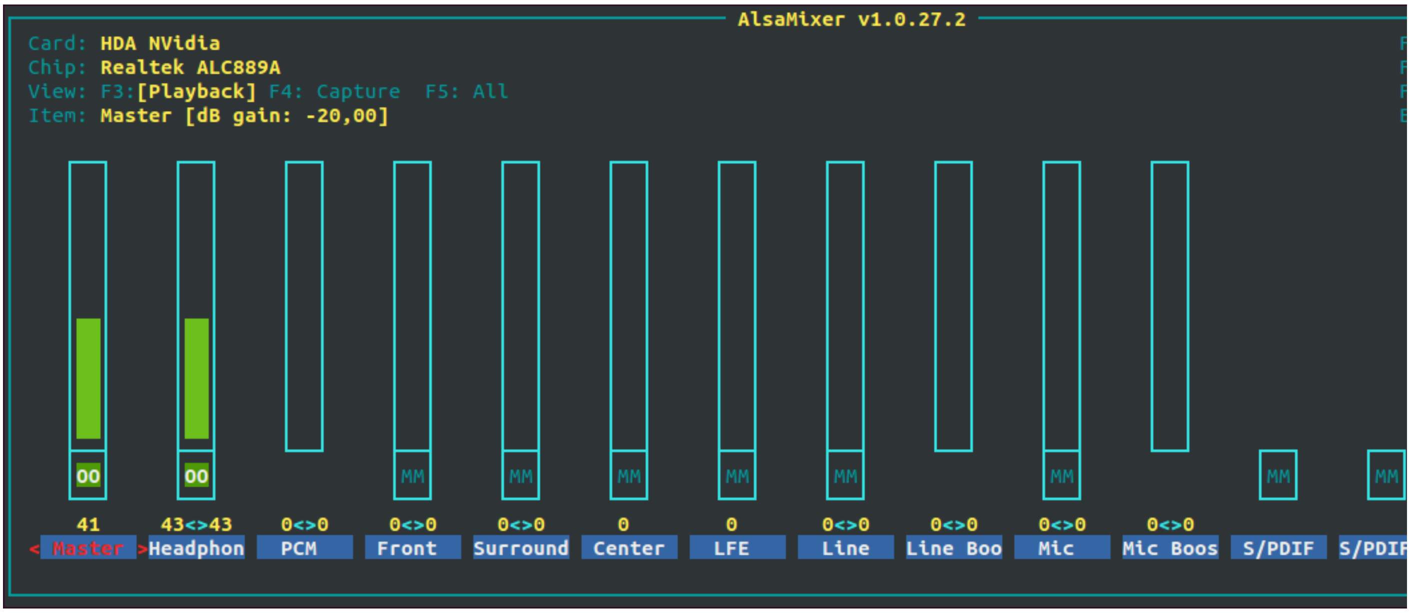
### 3.1 Media player: [MPlayer]

```
$ mplayer -ao jack * &
```



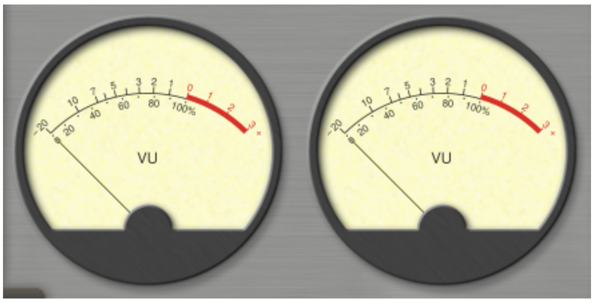
```
2000 - Stiff Upper Lip - 06 - Can'T Stand Still.mp3
vruiz@pluton:~/Musica/Metal/AcDc$ mplayer -ao jack *
MPlayer 1.1-4.8 (C) 2000-2012 MPlayer Team
mplayer: could not connect to socket
mplayer: No such file or directory
Failed to open LIRC support. You will not be able to use your remote control.

Playing 1975 - High Voltage - 07 - Love Song.mp3.
libavformat version 54.20.4 (external)
Mismatching header version 54.20.3
Audio only file format detected.
Load subtitles in ./
==========================================================================
Requested audio codec family [mpg123] (afm=mpg123) not available.
Enable it at compilation.
Opening audio decoder: [ffmpeg] FFmpeg/libavcodec audio decoders
libavcodec version 54.35.0 (external)
AUDIO: 44100 Hz, 2 ch, floatle, 192.0 kbit/6.80% (ratio: 24000->352800)
Selected audio codec: [ffmp3float] afm: ffmpeg (FFmpeg MPEG layer-3 audio)
==========================================================================
AO: [jack] 44100Hz 2ch floatle (4 bytes per sample)
Video: no video
Starting playback...
A:  36.8 (36.7) of 314.0 (05:14.0)  0.6%
```



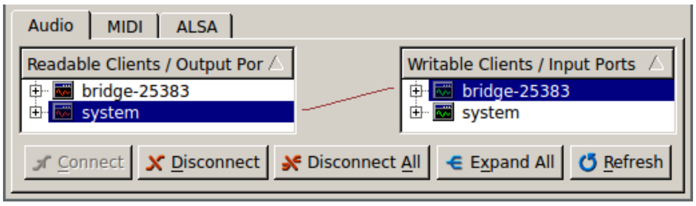

# Chapter 4
# Meters

### 4.1 A VU (Volume Unit) meter: [meterbridge]

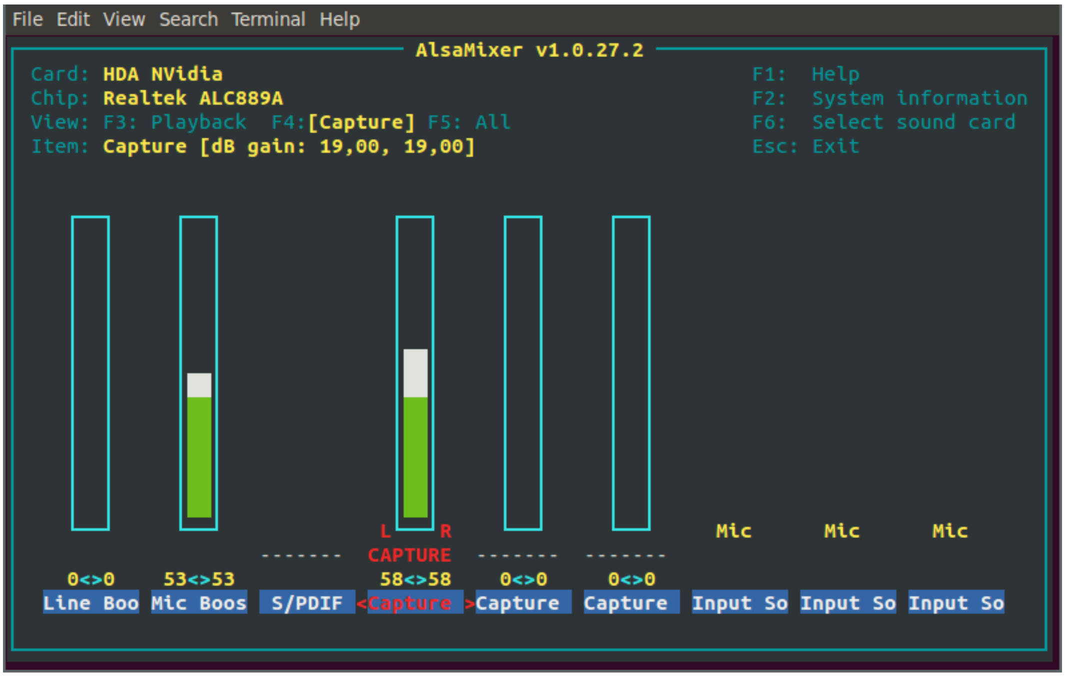
1. Run:
```
$ meterbridge -t vu alsa_pcm:capture_1 alsa_pcm:capture_2 &  # Or ...
$ meterbridge -t ppm alsa_pcm:capture_1 alsa_pcm:capture_2 & # Or ...
$ meterbridge -t dpm alsa_pcm:capture_1 alsa_pcm:capture_2 & # Or ...
$ meterbridge -t jf alsa_pcm:capture_1 alsa_pcm:capture_2 &  # Or ...
$ meterbridge -t sco alsa_pcm:capture_1 alsa_pcm:capture_2 & # Although this is preferable
```



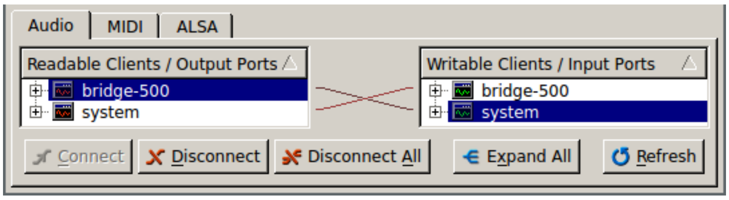

2. Connect (route) it:

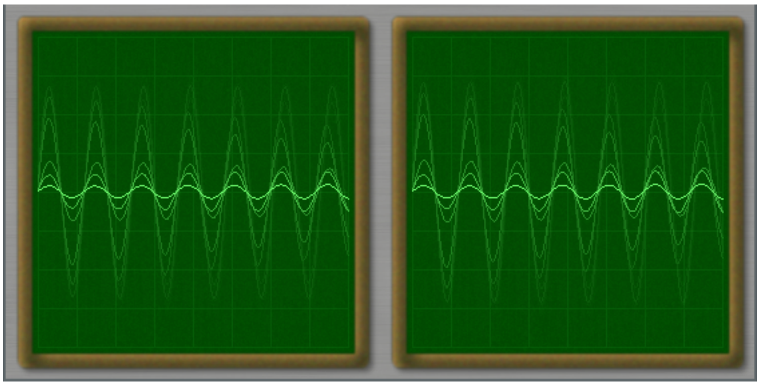



3. But, I don’t listen myself! Of course, you have to route the `system` readable client in the Connections/Audio window towards the `system` writable client in order to provide this functionality. But be careful with the output volume of the speakers (Master and PCM) and the microphone gain (Mic and Mic Boost) mixer controls, specially is you are done this in a laptop, because the microphone can be feedbacked by the speakers, producing an annoying sound coupling. See below:



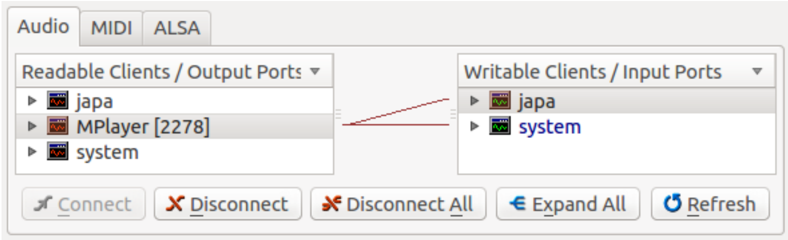


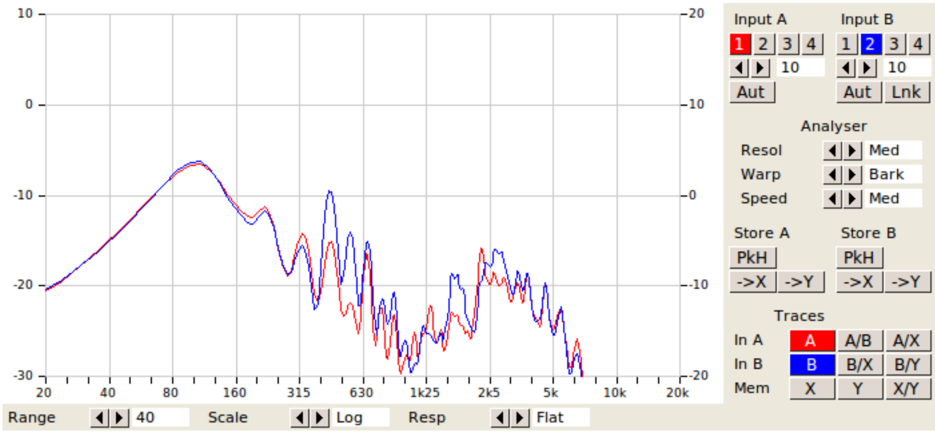


## 4.2 [Japa (JACK Perceptual Analyzer)](https://w3.ual.es/~vruiz/Docencia/Apuntes/Multimedia/Sound/JACK/index.html)

- Run as (while the JACK server is running):
  `$ japa -J &`

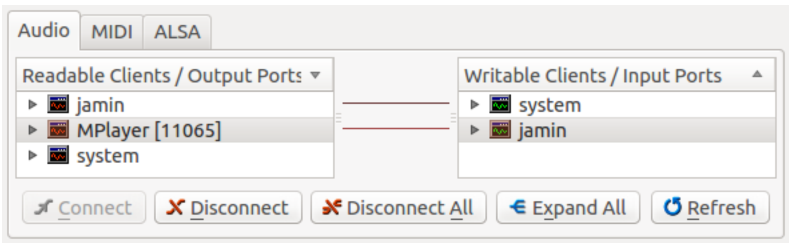


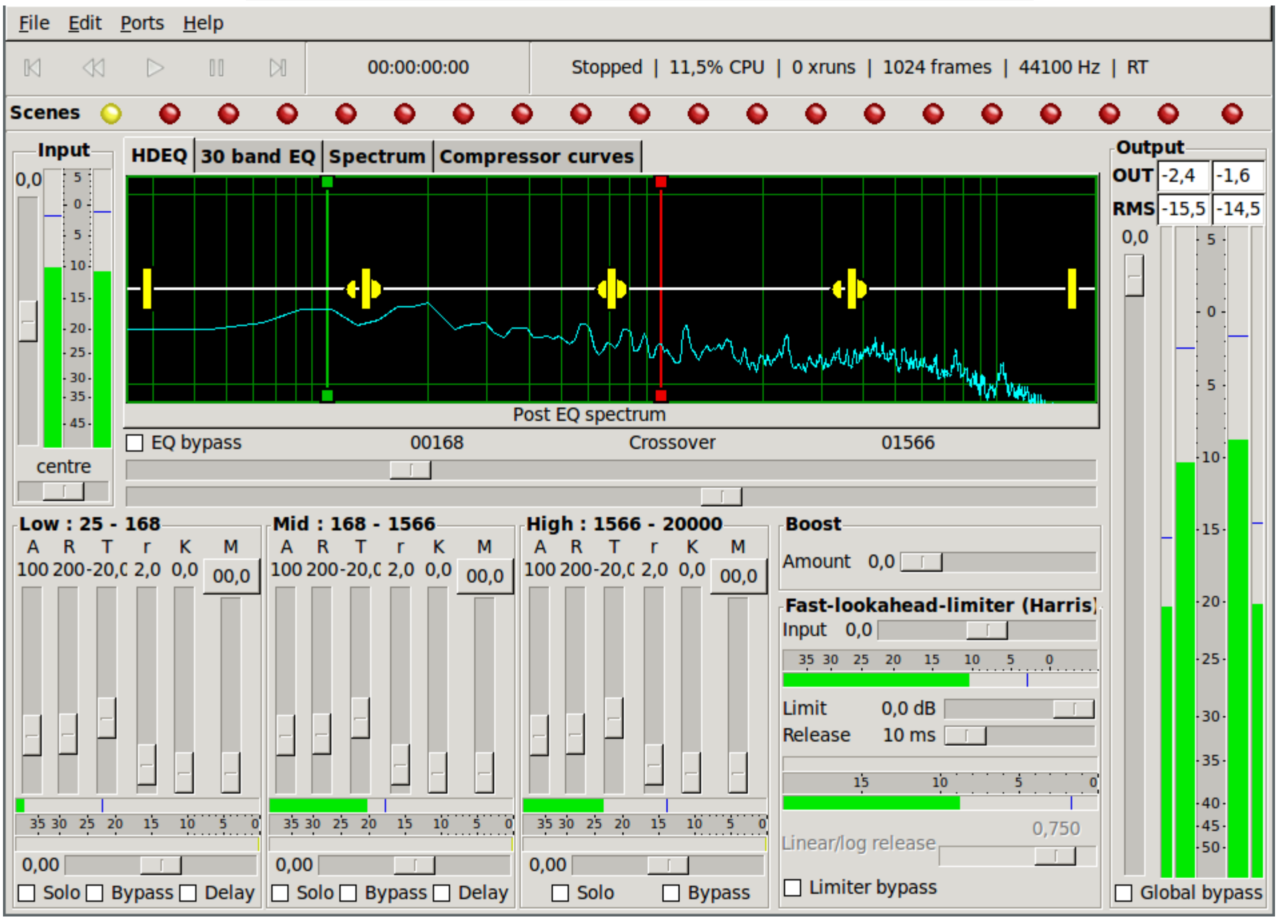




# Chapter 5
# Effects

## 5.1 [JAMin (JACK Audio Mastering interface)]()

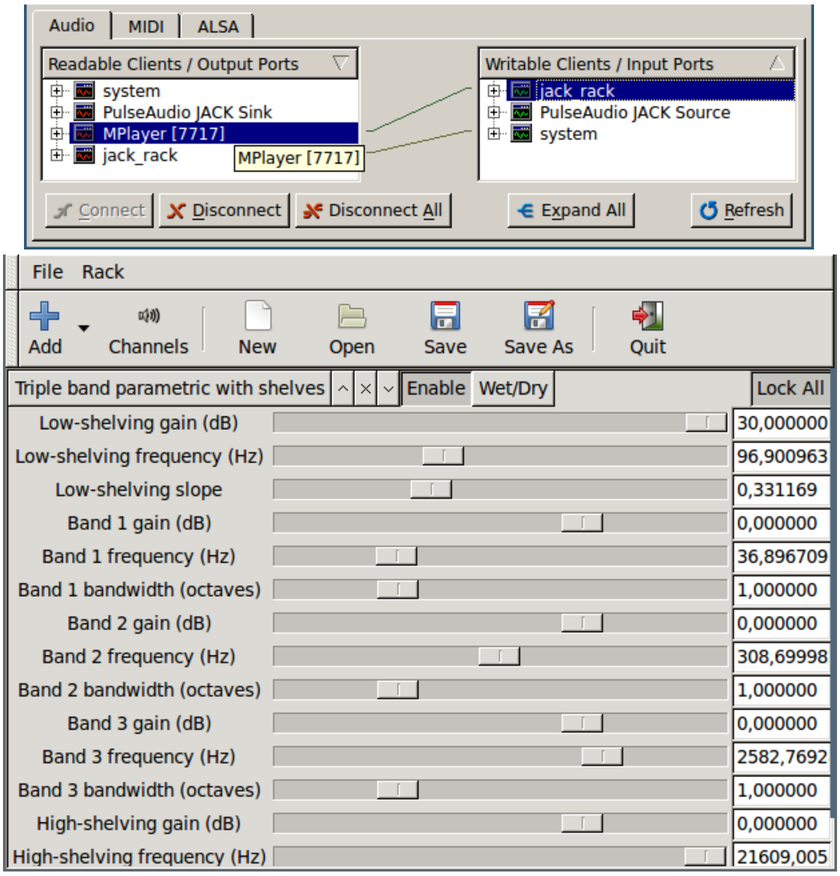
```
$ jamin &
```



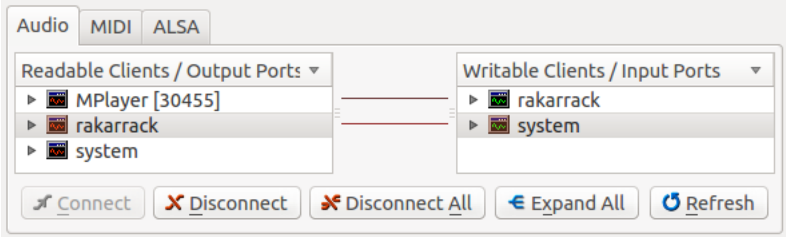
### 5.2 [JACK Rack]

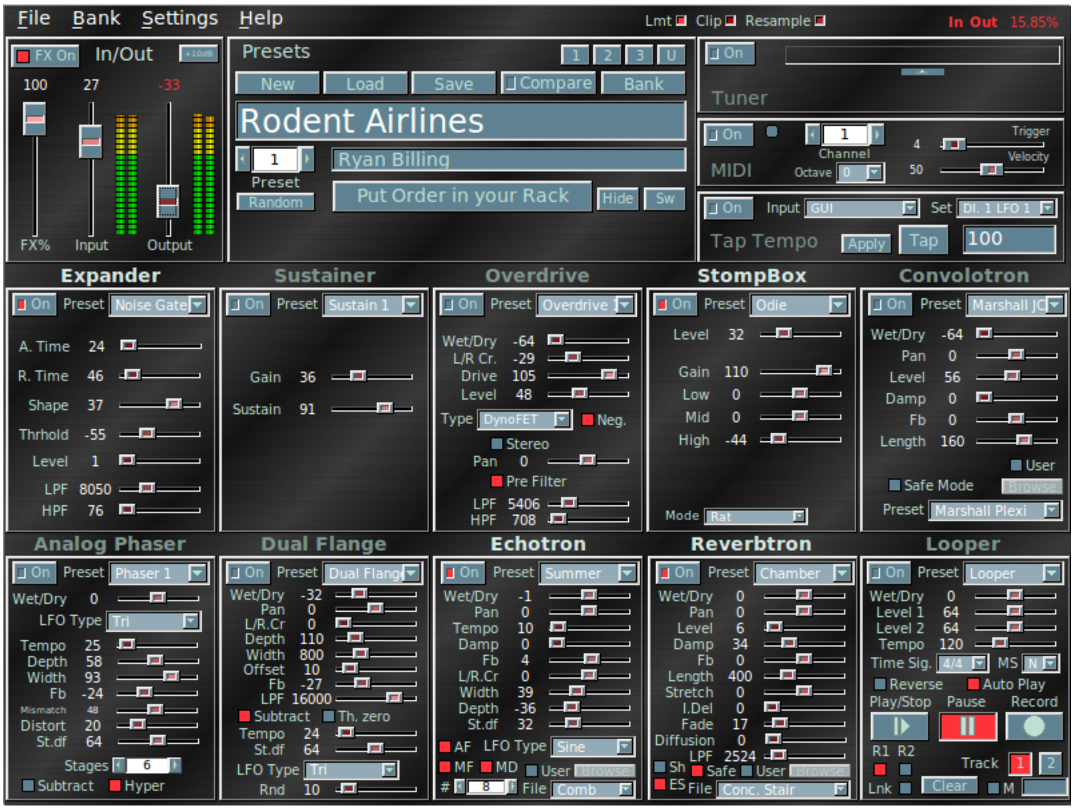
```
$ jack-rack &
```




### 5.3 [Rakarrack]

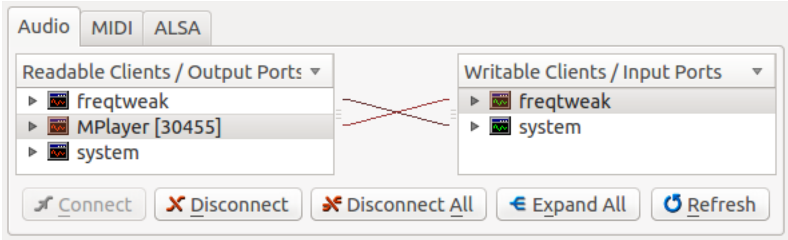
```
$ rakarrack &
```

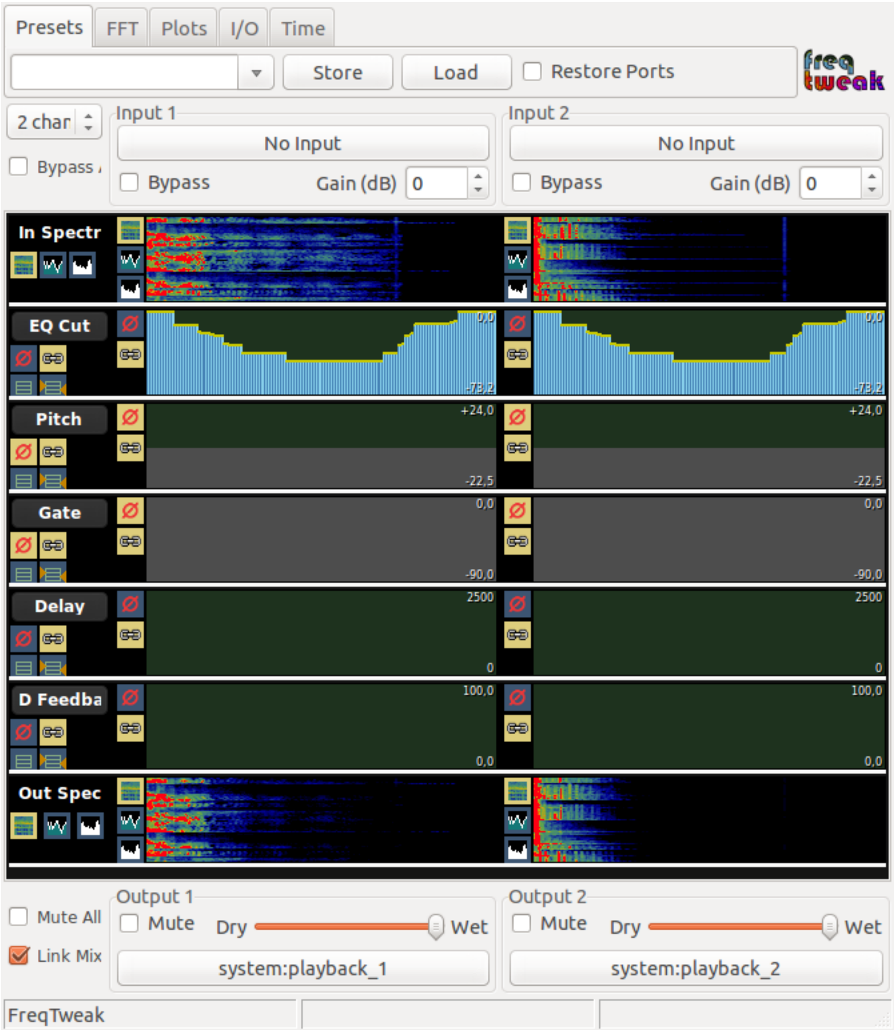



## 5.4 [Freqtweak]()

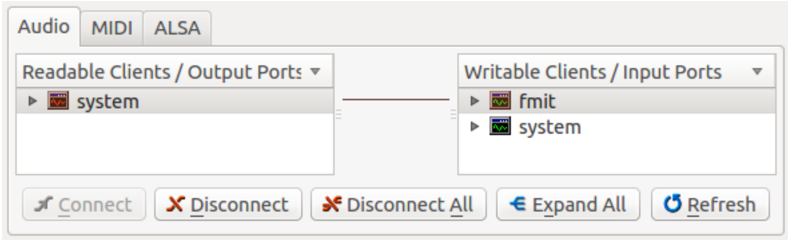
```
$ freqtweak &
```

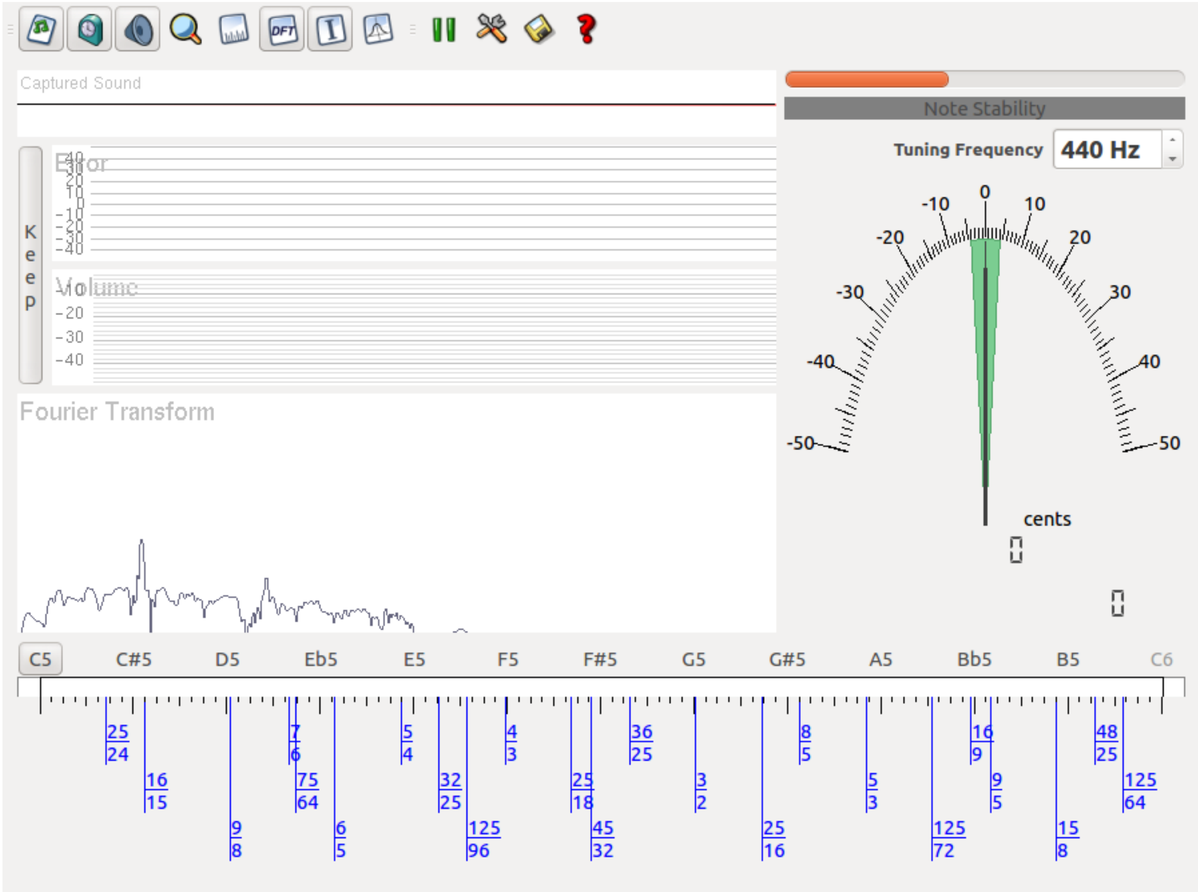



### 5.5 [FMIT (Free Music Instrument Tuner)]

```
fmit &
```

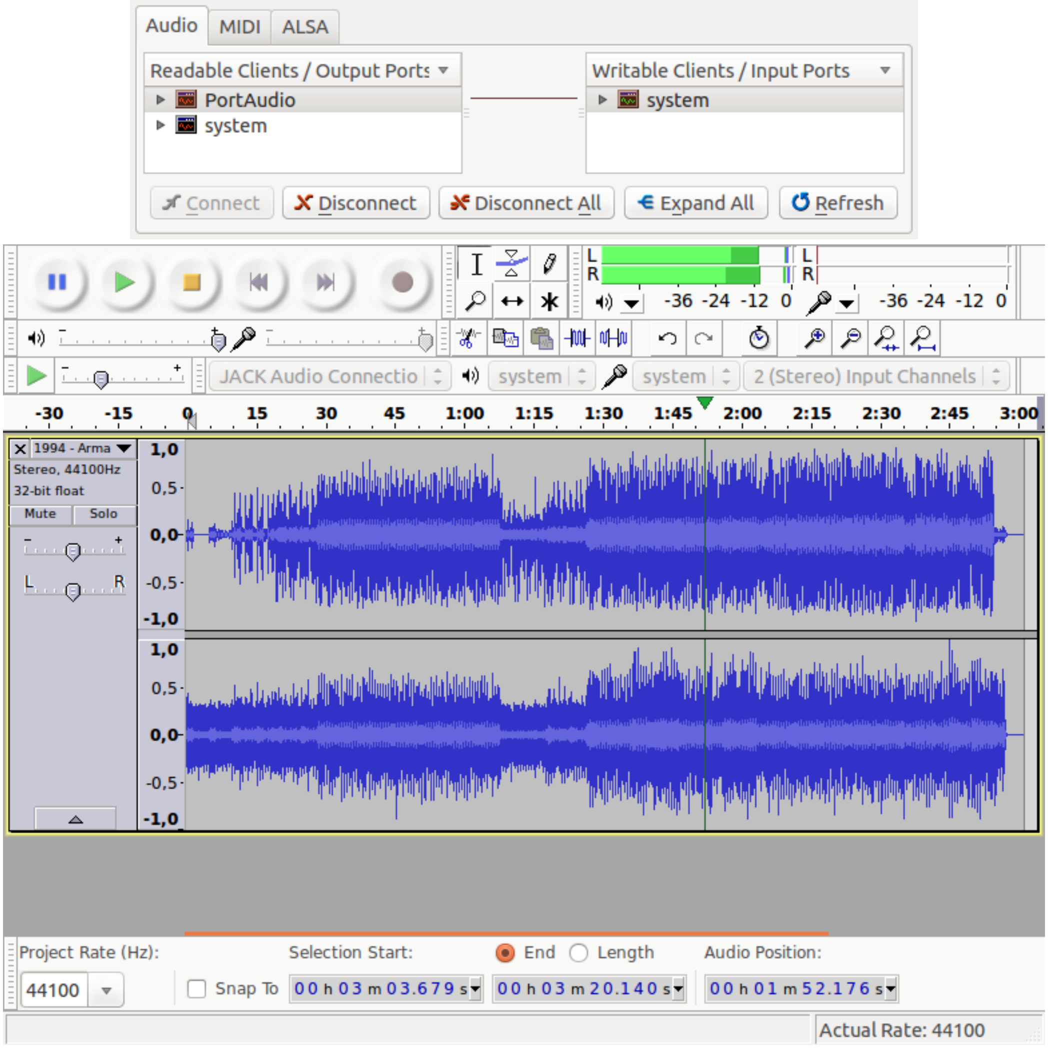



## Chapter 6
## Edition

### 6.1 [Audacity](): capture and edit audio

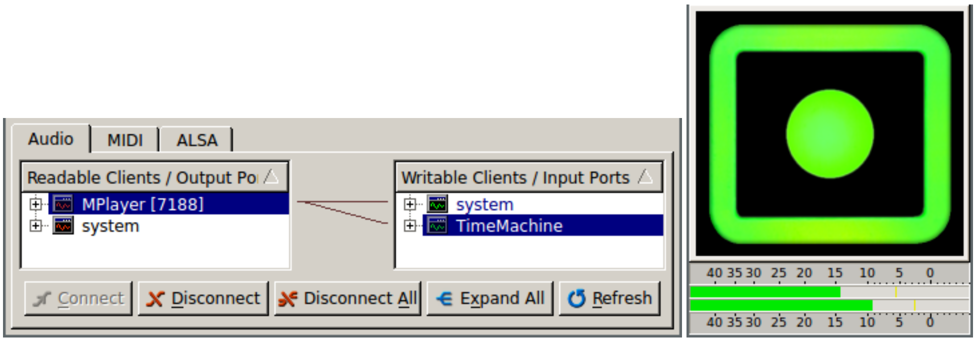
```
# Note: usually audacity establish the connections when playing.
$ audacity &
```




# Chapter 7 Capture

## 7.1 [JACK Timemachine](): capture audio

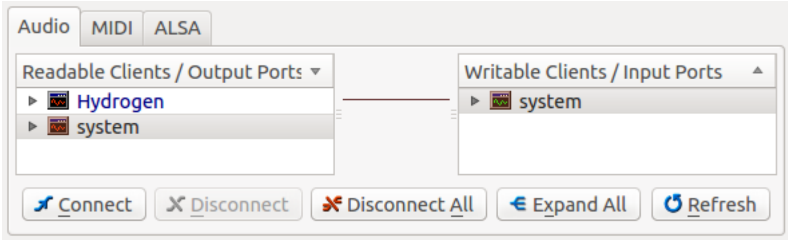
```
$ timemachine &
```

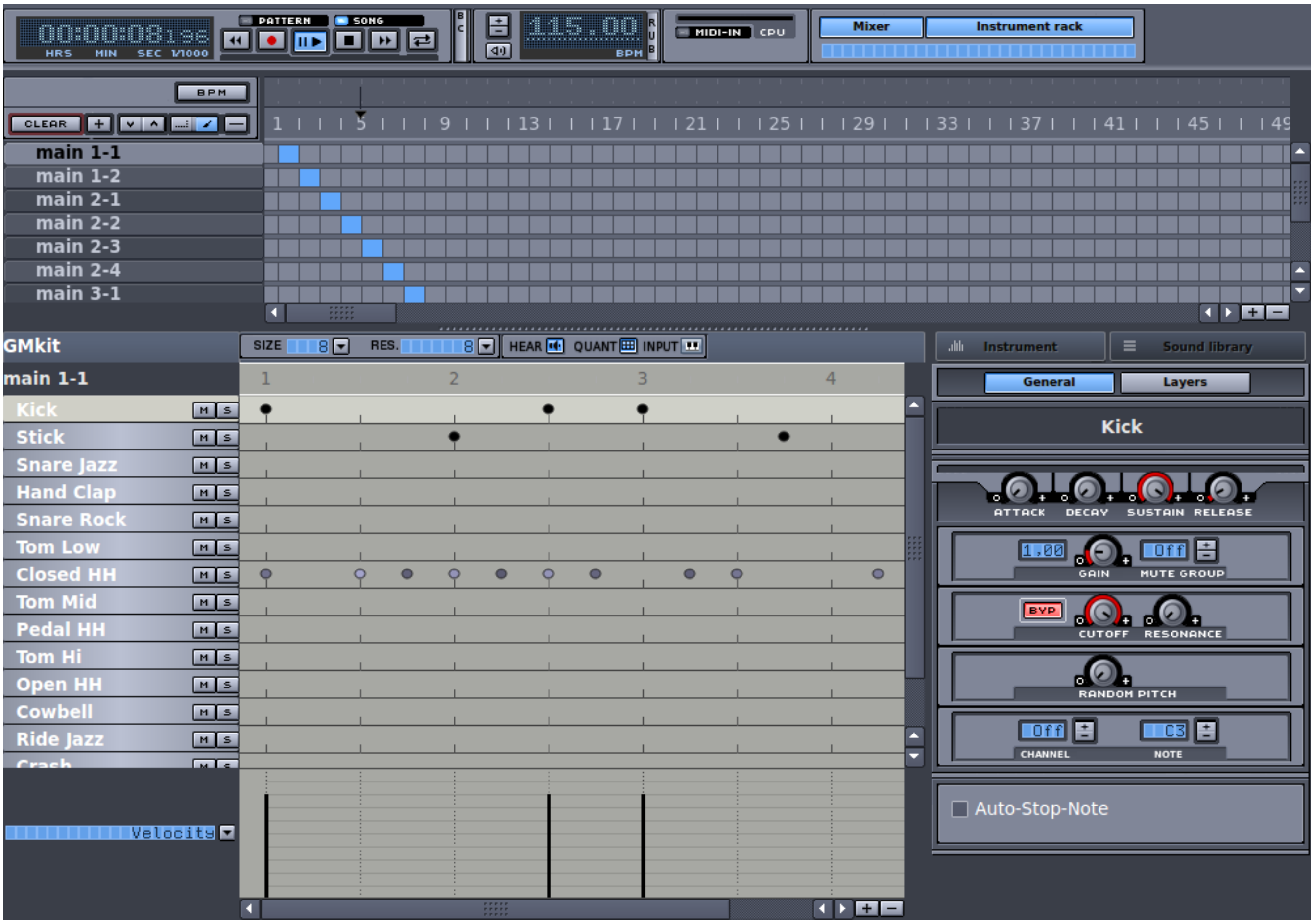


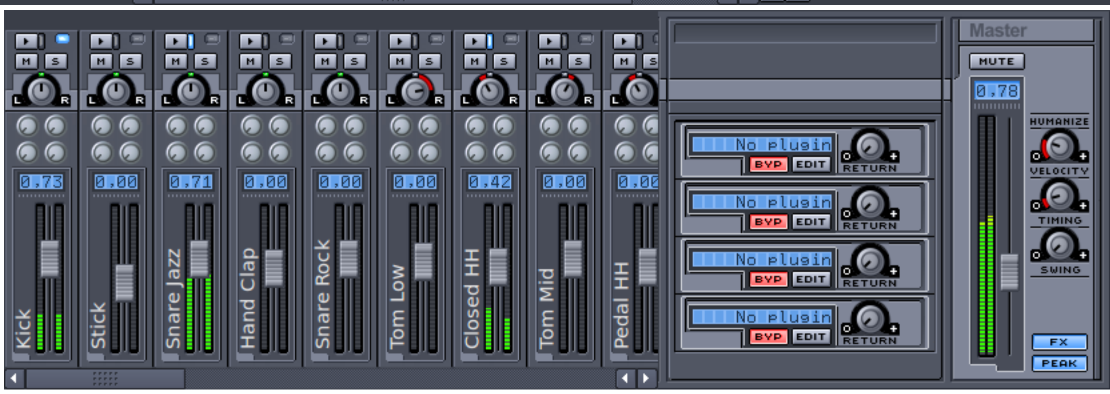

# Chapter 8 Synthesis

## 8.1 The [Hydrogen]() drum machine

```
$ hydrogen &
```

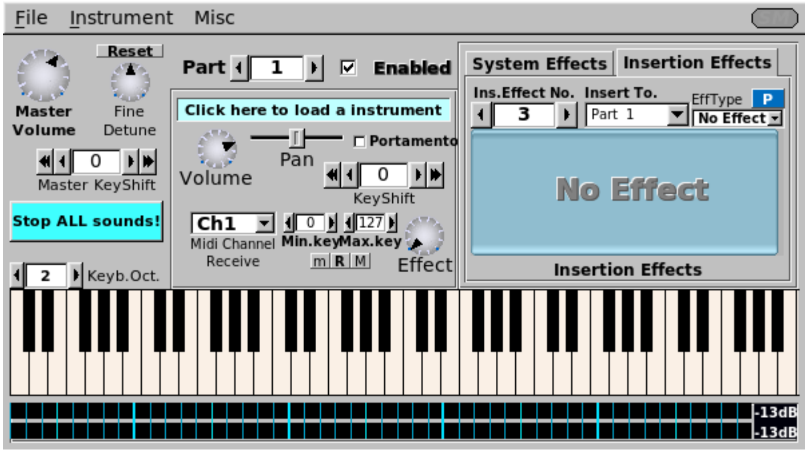


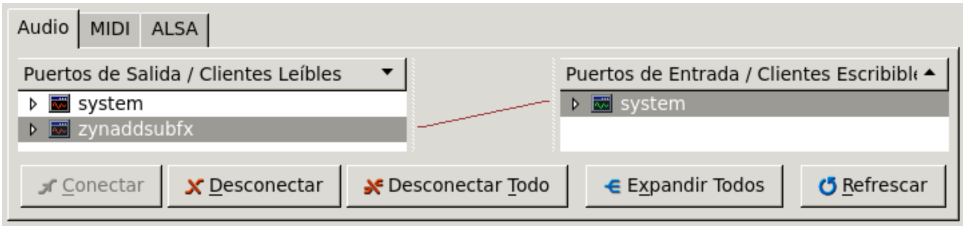




## 8.2 ZynAddSubFX

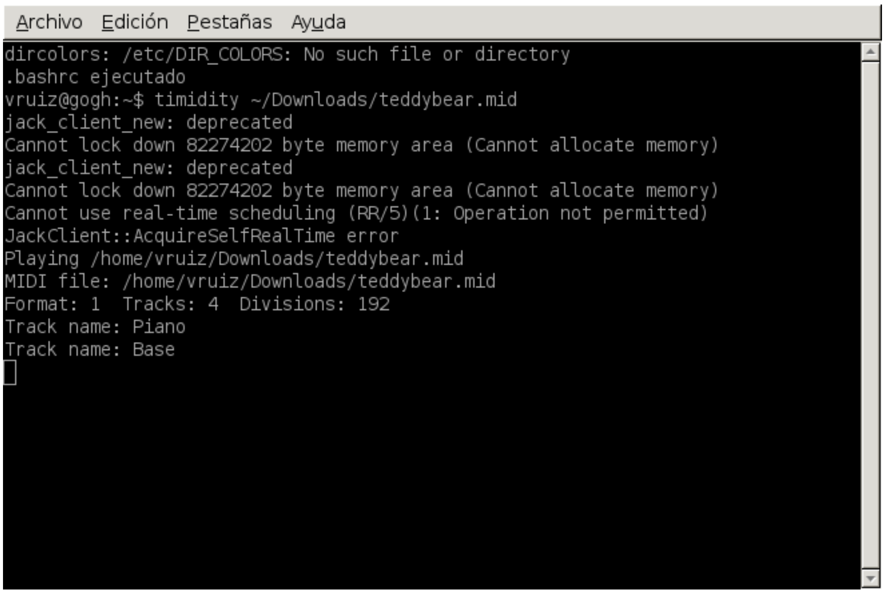
```
zynaddsubfx &
```

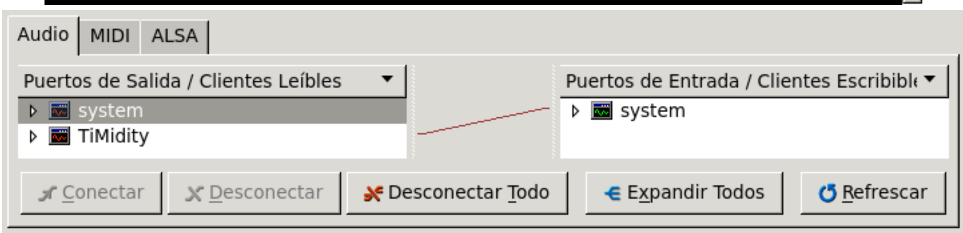



## 8.3 [TiMidity]

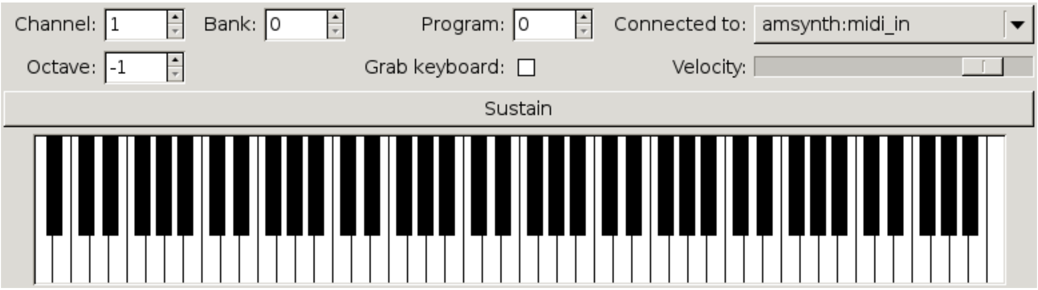
```
timidity ~/Downloads/teddybear.mid
```




## 8.4 [JACK Keyboard] and [AMSynth]

```
jack-keyboard &
amsynth &
```






# Chapter 9
# Remember

- While the modules `module-jack-source` and `module-jack-sink` are loaded into Pulseaudio, the JACK server is automatically started.
- To avoid this, remove these modules from the file `/etc/pulse/default.pa`.