How to create music with GNU/Linux

Emmanuel Saracco

emmanuel@esaracco.fr

How to create music with GNU/Linux

by Emmanuel Saracco

How to create music with GNU/Linux

Warning

WORK IN PROGRESS

Revision History

Revision 0.0 2009-01-30 Revised by: es
Not yet versioned: It is still a work in progress.

Dedication

This howto is dedicated to all GNU/Linux users that refuse to use proprietary software to work with audio.

Many thanks to all Free developers and Free composers that help us day-by-day to make this possible.

Table of Contents

List of Tables

Forword

Working with audio on GNU/Linux can be really complicated, but most of the time it is just a matter of compiling a kernel, installing, settings some packages and also using software the right way :-)

It is still a work in progress. If you have comments, experience to share... do not hesitate to send me an E-Mail and I will update this documentation.

Also keep in mind that it exists many different ways to succeed in setting up, installing and using software, and I do not know them all. So I will describe what I use and how I do, but not how you *must* do.

> **Warning**
>
> I warn the readers that I am French and that my English is poor: so I do appreciate any correction!

Chapter 1. System settings and tuning

Audio creation requires a lot of resources. It requires also specific kernel configuration.

> **Warning**
>
> First of all, check that users who want to work with audio are in the `audio` group. For example, for a user `youruser`, just execute the following command as `root`: **`addgroup youruser audio`**.

1.1. My Studio

I am using a Dell Dimension 8400.

Table 1-1. My Studio

Type	Information
Soundcard	1 RME Hammerfall HDSP 9632 (http://www.rme-audio.com/english/hdsp/hdsp9632.htm) (+ the AEB 4/I expansion board)
RAM	3GB
Hard Disk	1 SATA 250GB
CPU	1 P4 3.6GHz EM64T with HyperThreading activated
Amp simulator	Line6 POD XT (http://line6.com/podxt/)
Mic	audio-technica AT 4041 (http://www.audio-technica.com/cms/wired_mics/dd13750b31887ecb/index.html), Apex 170
Mic preamp	SM Pro Audio TB101 (http://www.smproaudio.com/index.php?option=content&task=view&
Headphone	Sennheiser HD 600 (http://www.sennheiser.com/sennheiser/icm_eng.nsf/root/04465)
GNU/Linux distro	Debian (http://www.debian.org)/sid, 64-bit

1.2. File system

My filesystem type is `ext3`, mounted with the `noatime` option. Here is a sample of my `/etc/fstab` file:

```
/dev/sda7 / ext3 defaults,errors=remount-ro,noatime 0 1
```

If you have a IDE hard disk you can also try to optimize it by using hdparm. I will show you some basic tips, but you should refer to hdparm documentation to adapt it to your own hardware.

```
# hdparm -i /dev/hda1                                  ❶
# hdparm -I /dev/hda                                   ❷
# hdparm -c3 -m16 -d1 -A1 -X69 /dev/hda1               ❸
# htparm -tT /dev/hda1                                 ❹
```

❶ Show you a brief summary of your disk's characteristics.

❷ Same thing than the `-i` option, but more verbose and better structured.

❸ A common way for hard disk tuning.

❸ A common for testing hard disk IO capacity.

1.3. Linux Kernel

I use a 2.6.26.5 (see my .config (http://www.esaracco.fr/downloads/config-2.6.26.5)). In fact, I upgrade the kernel each time a new release is out. I will assume you are confident in building/installing a Linux kernel. With 2.6.x it is generally no more than doing:

```
$ make menuconfig
$ make
$ su
# make modules_install
# make install
# vi /boot/grub/menu.lst
```

But to build a Linux Audio Kernel there is some more work to do.

1.3.1. Realtime Preemption patch

Realtime Preemption Ingo Molnar's Linux kernel patch which tries to improve realtime performance of the Linux kernel.

> **Note:** With my configuration and the last Linux kernel (2.6.26.5) I work without needing this patch.

Always download the last release here (http://www.kernel.org/pub/linux/kernel/projects/rt/). Then apply it:

```
$ cd /usr/src/
$ wget http://people.redhat.com/mingo/realtime-preempt/patch-2.6.26.5-rt9
$ rm linux
$ mv linux-2.6.26.5 linux-2.6.26.5-rt9
$ ln -s linux-2.6.26.5-rt9 linux
$ cd linux/
$ patch -p1 < ../patch-2.6.26.5-rt9
```

Patch should apply without fatal problems. Messages like `Hunk #3 succeeded at 652 (offset 1 line).` are ok. Search for outputs like `Failed`. The most common failure is on the `Makefile` file. Just edit it by hand and add **`-rt9`** at the end of the `EXTRAVERSION` variable.

1.3.2. Configuration

Warning

Do not forget to adapt those samples to your own configuration (especially for the "PCI devices" section).

The most important items for a vanilla Linux kernel (not patched with Ingo's RT patch) are:

```
Processor type and features --->
```

```
  Preemption Model (Preemptible Kernel (Low-Latency Desktop)) --->
  [*] Preempt The Big Kernel Lock
  Timer frequency (1000 HZ) --->

Device Drivers  --->
  Character devices  --->
    <M> Enhanced Real Time Clock Support
    <M> Generic /dev/rtc emulation
    [*]   Extended RTC operation
  <M> Real Time Clock  --->
    [*]   /sys/class/rtc/rtcN (sysfs)
    [*]   /proc/driver/rtc (procfs for rtc0)
    [*]   /dev/rtcN (character devices)
    <M>   PC-style 'CMOS'
  Sound  --->
    <M> Sound card support
      Advanced Linux Sound Architecture  --->
        <M> Advanced Linux Sound Architecture
          <M>   Sequencer support
          <M>     Sequencer dummy client
          <M>   OSS Mixer API
          <M>   OSS PCM (digital audio) API
          <M>   RTC Timer support
          [*]     Use RTC as default sequencer timer
        Generic devices  --->
          <M> Virtual MIDI soundcard
        PCI devices  --->
          <M> RME Hammerfall DSP Audio

Security options --->
  [*] Enable different security models
  <M>   Default Linux Capabilities
```

For a kernel patched with Ingo's RT patch the differences are:

```
General setup  --->
  [*] Enable concurrent radix tree operations
  [*]    Enabled optimistic locking

Processor type and features  --->
  [*] Enable priority boosting of RCU read-side critical sections
```

Also for RT patch you must uncheck the folloging options:

```
Processor type and features --->
  [ ] Enable tracing for RCU - currently stats in debugfs

Device Drivers  --->
  Character devices  --->
    [ ] Real Time Clock Histogram Support
    < > Parallel Port Based Latency Measurement Device

Kernel hacking  --->
  [ ] Wakeup latency timing
  [ ] Non-preemptible critical section latency timing
  [ ] Interrupts-off critical section latency timing
```

1.3.3. Installation

Install modules and kernel and edit the `/etc/modules` file to add the following content, depending on your hardware:

```
rtc
snd-rtctimer
snd-hdsp
snd-hwdep
snd-page-alloc
snd-pcm
snd-rawmidi
snd-seq
snd-seq_device
snd-seq_midi
snd-seq_midi_event
snd-timer
```

Most of these modules depend on others that will be automatically loaded.

Most of time you would also raise the value of the `max-user-freq` kernel value. To do so you can either:

- Use **sysctl** by editing `/etc/sysctl.conf` and adding the following line:

  ```
  dev.rtc.max-user-freq=1024
  ```

Once the file is updated, reload the configuration by running **`sysctl -p`** as root user.

- Directly write the value into `/proc/sys/dev/rtc/max-user-freq` at boot time:

```
echo 1024 > /proc/sys/dev/rtc/max-user-freq
```

Now just reboot on your new kernel.

1.4. Obtaining Reatime privileges

There is actually three ways to obtain root privileges for Realtime. The first way is to use the "Realtime LSM" module. But this way is pretty deprecated and should not be used nowdays. The second way is to use a little tool named set_rlimits. This way is ok, but it should only be used when the PAM library does not support "rlimits" operations. The third way is the right way: it consist on using PAM.

1.4.1. Realtime LSM

This way is not recommended.

The Realtime Linux Security Module (LSM) is a loadable extension for Linux 2.6 kernels. It selectively grants realtime permissions to specific user groups or applications.

First, retreive the last CVS version:

```
$ cvs -d:pserver:anonymous@realtime-lsm.cvs.sourceforge.net:/cvsroot/realtime-lsm login
$ cvs -z3 -d:pserver:anonymous@realtime-lsm.cvs.sourceforge.net:/cvsroot/realtime-lsm co -P realtime-
```

The apply the `rt-lsm-kernel.patch` kernel patch on you Linux kernel:

```
$ cd /usr/src/linux/
$ patch -p1 ../realtime-lsm/rt-lsm-kernel.patch
$ make menuconfig
$ make
$ su -c "make modules_install"
$ su -c "make install"
```

Your Security options kernel configuration section should be like the following:

```
Security options --->
  [ ] Enable access key retention support
  [*] Enable different security models
  [ ]   Socket and Networking Security Hooks
  <M>   Default Linux Capabilities
  < >   Root Plug Support
  < >   BSD Secure Levels
  <M>   Realtime Capabilities
```

If you do not see the Realtime Capabilities item, then check the end of the kernel's `security/Kconfig` file and try to fix it (it can have a issue with the `depends` option).

Reboot with your new kernel.

> **Warning**
>
> Do not forget to repeat this step every time you rebuild a new kernel.

Then if you do not care about security, just do **`modprobe realtime any=1`** to load the module and to allow its use for every users of your system. If you want to allow only the `audio` group to access realtime capacities, check that your current user is in the audio group and load realtime module by passing it the `gid` of the group to allow:

```
# addgroup youruser audio
# modprobe realtime gid=$(grep audio /etc/group | cut -d':' -f3) any=1
```

If you want to load this module at boot time on Debian, do the following:

```
$ su
# echo "options realtime any=1" > /etc/modprobe.d/realtime
# echo "realtime" >> /etc/modules
```

1.4.2. Using set_rlimits

This way is not recommended.

set_rlimits is a small wrapper program to allow people to take advantage of he realtime resource limit extensions available in Linux kernels 2.6.12 and later without having to resort to using a PAM module. This method was ok when PAM was not supporting kernel's rlimits operations.

Download it here (http://www.physics.adelaide.edu.au/~jwoithe/).

First, edit the `Makefile` file and change the `PREFIX` variable to `/usr/local` to `/usr/bin`. Then uncompress it, rebuild it and install it:

```
$ make clean
$ make
$ su -c "make install"
```

Once it is installed, edit the `/etc/set_rlimits.conf` file and add the following line at the end:

```
@audio  /usr/bin/qjackctl nice=-1 rtprio=80
```

Now you will be able to execute **qjackctl** (see Section 3.2) with all necessary privileges, and it will be able to execute **jackd** (see Section 3.1) with realtime privileges [1].

1.4.3. Using PAM

This is the right way :-)

PAM configuration is the better choice. Basically it only consists on updating the PAM limits configuration in `/etc/security/limits.conf`[2] to fit your needs:

```
@audio  -  rtprio   100
@audio  -  nice     -10
@audio  -  memlock  unlimited
```

This will allow all software executed by audio group members to access realtime and to consume memory without restriction.

Done!

1.5. PCI latency

To display information about all PCI buses in your system and all device connected to them, use the **lspci** utility :

```
$ lspci
[...]
00:1f.2 SATA controller: Intel Corporation 82801FR/FRW (ICH6R/ICH6RW) SATA Controller (rev 03)
00:1f.3 SMBus: Intel Corporation 82801FB/FBM/FR/FW/FRW (ICH6 Family) SMBus Controller (rev 03)
01:00.0 VGA compatible controller: nVidia Corporation NV41.1 [GeForce 6800] (rev a2)
02:00.0 Ethernet controller: Broadcom Corporation NetXtreme BCM5751 Gigabit Ethernet PCI Express (rev
04:00.0 Multimedia audio controller: Creative Labs SB Audigy (rev 04)
04:00.1 Input device controller: Creative Labs SB Audigy Game Port (rev 04)
04:00.2 FireWire (IEEE 1394): Creative Labs SB Audigy FireWire Port (rev 04)
04:01.0 Multimedia audio controller: Xilinx Corporation RME Hammerfall DSP (rev 97)
```

To obtain information about latency for each bus, use the `-v` option:

```
# lspci -v
[...]
04:00.0 Multimedia audio controller: Creative Labs SB Audigy (rev 04)
        Subsystem: Creative Labs Unknown device 2006
        Flags: bus master, medium devsel, latency 176, IRQ 16
        I/O ports at dcc0 [size=64]
        Capabilities: [dc] Power Management version 2
[...]
04:01.0 Multimedia audio controller: Xilinx Corporation RME Hammerfall DSP (rev 97)
        Flags: bus master, medium devsel, latency 255, IRQ 17
        Memory at dcdf0000 (32-bit, non-prefetchable) [size=64K]
```

You can see here that the card I mainly use to work with audio (the RME) has the maximum latency priority.

To modify the priority of a PCI bus, use the **setpci** command. Pass it the device you want to control (to control my RME latency for example I use `04:01.0`), followed by the `latency_timer` option and the value in hexadecimal notation.

To give the maximum priority to a bus, just do as `root`:

```
# setpci -v -s 04:01.0 latency_timer=ff
```

You can use a script like this one (http://www.esaracco.fr/downloads/pcilatency) to automate this task. Just download it and put it in your `/etc/init.d/` directory after updating it to fit you needs. On Debian, you can do the following :

```
# wget http://www.esaracco.fr/downloads/pcilatency
# vim pcilatency
# chmod 755 pcilatency
# mv pcilatency /etc/init.d/
# cd /etc/init.d/
# update-rc.d pcilatency defaults
```

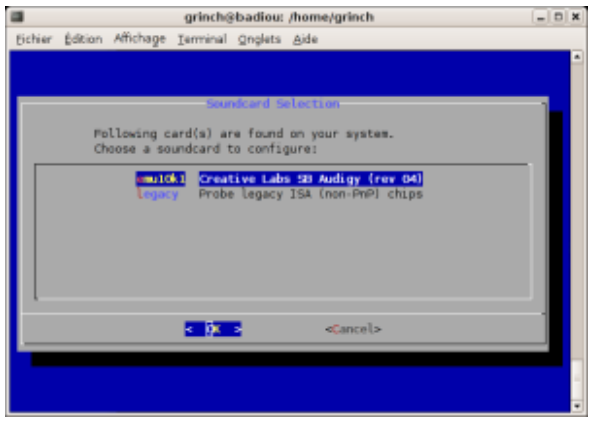
1.6. ALSA

ALSA stand for the Advanced Linux Sound Architecture provides audio and MIDI functionality to the Linux operating system. See the ALSA homepage (http://www.alsa-project.org/) for more information.

See Section 1.3 to know how to build a Linux kernel with ALSA support.

Once you have rebuilt your kernel, install `alsa-base` and `alsa-utils` packages. Then as root user execute **alsaconf**.

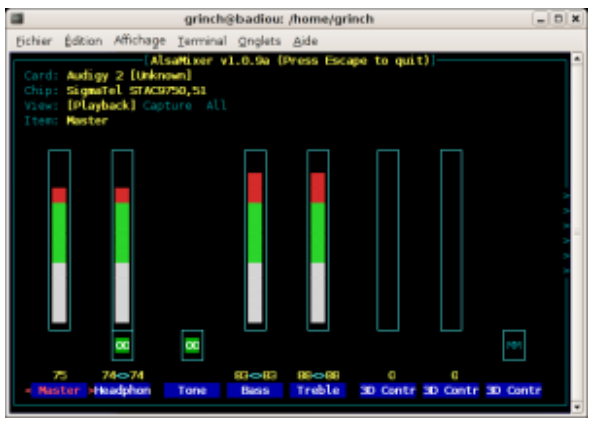
Figure 1-1. ALSA configuration



Choose your soundcard driver. Alsaconf will set up your system to load this driver at boot.

Next execute **alsamixer** to suit your needs. However your hardware must contain conventionnal mixer. If not, try to find specific Free projects that support your card (see Section 2.1.3.2).

Figure 1-2. alsamixer



Notes

1. Execute it like that: **`set_rlimits -r /usr/bin/qjackctl`**

2. You should be able to use a more distro friendly solution by creating a directory called `/etc/security/limits.d/` and write your configuration in a file called `audio.conf`.

Chapter 2. Hardware

This section will explain some specific hardware installation. Please, feel free to send me documentation patches for this section with your own experimentation.

2.1. RME Hammerfall HDSP 9632

A great professionnal soundcard that works just fine with GNU/linux once you have understood its philosophy and avoid traps. We will describe here how to install and configure it.

2.1.1. Firmware

Thanks to Paul Davis and others it exists some kernel stuff for RME HDSP support. However not all firmware versions are supported by old ALSA drivers. ALSA drivers (>= `1.0.14rc1`) are ok[1].

Depending on your firmware and ALSA version you will sometime need to downgrade the firmware of your card (but, I repeat, *this operation is not needed anymore with ALSA drivers* >= *`1.0.14rc1`*). To do this, download a older firmware from the RME website. You will need the 1.51 version. You will find it on the RME archives (http://www.rme-audio.com/english/download/drivers_archive.htm) (under the "HAMMERFALL DSP DRIVERS"). You must download a file named fut_win_dsp.zip (http://www.rme-audio.com/download/treiber_archiv/fut_win_dsp.zip).

Once you have downloaded this file, save it on a USB key and reboot on Windows. Then just unzip the file and click on the `hdsp_fut.exe` file. Do not take in account Windows alerts and click on the update button. Then reboot.

If all was fine, your firmware version should be `1.51` (or `151`) on Windows and **lspci** should output `rev97` on GNU/Linux:

```
$ lspci | grep -i rme
04:01.0 Multimedia audio controller: Xilinx Corporation RME Hammerfall DSP (rev 97)
```

2.1.2. Linux Kernel

First, take a look at Section 1.3 if you are not confident with kernel compilation.

You will need to compile the `snd-hdsp` ALSA module.

```
Device Drivers  --->
  Sound  --->
    <M> Sound card support
      Advanced Linux Sound Architecture  --->
        <M> Advanced Linux Sound Architecture
        PCI devices  --->
          <M> RME Hammerfall DSP Audio
```

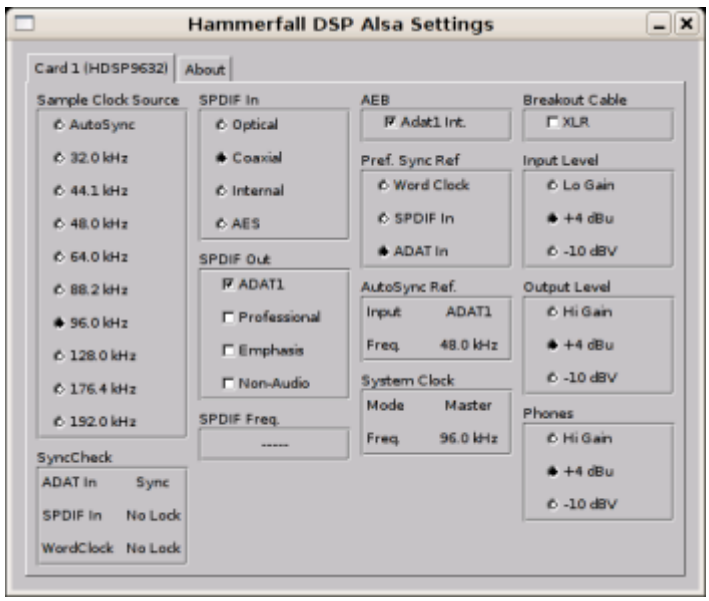
Then reboot or load the new module as root with **modprobe snd-hdsp**.

2.1.3. HDSP utilities

Thanks to Thomas Charbonnel and others RME HDSP soundcards have special mixer and configurator GUI. Those tools are in the alsa-tools-gui archive downloadable from the ALSA Web site or packaged for you own GNU/Linux distro.

2.1.3.1. hdspconf

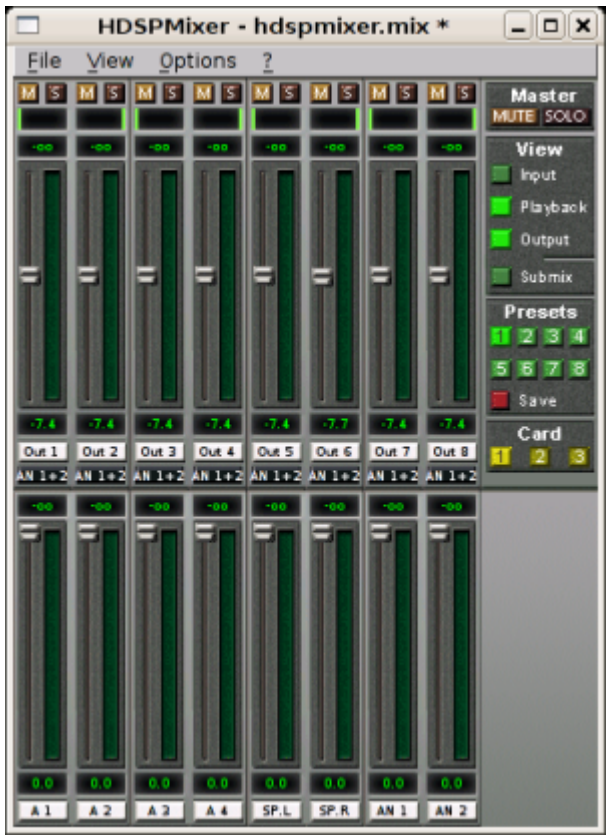
Figure 2-1. hdspconf main window



Use this tool to control the basics of your board. In my case I had to check the Adat1 Int. checkbox and ADAT In radio button in order to have the AEB 4/I expansion fonctionnal.

2.1.3.2. hdspmixer

Figure 2-2. hdspmixer main window



There is no conventional mixer on this card, so alsamixer and other "generic mixer" applications will do nothing. You will have to use hdspmixer every time you will need to capture/produce sound.

Notes

1. Thanks to Andrew Gaydenko for reporting me this great news :-)

Chapter 3. Building required audio software

For convenience[1] I build most of the audio software I am used to work with. You can use any version of gcc >= 3.4[2]. But at this time you will have troubles while compiling the DSSI Hexter plugin with a version >= 4. I mainly use the software described in this chapter, but many others (http://linux-sound.org/) exist. See Appendix A to find useful links.

> **Note:** Please pay attention to applications' dependencies. First check `README` or `INSTALL` files which are commonly found in tarballs archives. Then carefully check the ouput of the configuration process.

3.1. Jack Audio Connection Kit

JACK is a low-latency audio server, written for POSIX conformant operating systems such as GNU/Linux and Apple's OS X. It can connect a number of different applications to an audio device, as well as allowing them to share audio between themselves. Its clients can run in their own processes (ie. as normal applications), or can they can run within the JACK server (ie. as a *plugin*).

I recommand you to always use the SVN version of Jack. Checkout the SVN module as explained here (http://jackaudio.org/download).

Go to the main Jack directory and do:

```
$ ./autogen.sh --prefix=/usr --enable-mmx --enable-sse --enable-dynsimd --enable-optimize --enable-re
$ make
$ su -c "make install"
```

The configuring process should detect automatically a lot of things. The output of this step on my machine is:

```
jack-audio-connection-kit 0.109.0 :

| Build with ALSA support.............................. : true
| Build with old FireWire (FreeBob) support............. : false
| Build with new FireWire (FFADO) support............... : false
| Build with OSS support............................... : true
| Build with CoreAudio support......................... : false
| Build with PortAudio support......................... : false
```

```
| Compiler optimization flags........................... : -DREENTRANT -O3 -fomit-frame-pointer -ffast
| Compiler full flags................................... : -I$(top_srcdir)/config -I$(top_srcdir) -I$
| Install dir for libjack + backends.................... : ${exec_prefix}/lib64/jack
|
| Default driver backend................................ : "alsa"
| Shared memory interface............................... : "POSIX"
| Install prefix........................................ : /usr
```

3.2. 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. With time, this primordial interface has become richer by including an enhanced patchbay and connection control features.

Download it here (http://qjackctl.sourceforge.net/#Downloads).

Qjackctl needs QT stuff to be built. You should also define the QTDIR environment variable.

```
$ ./configure --prefix=/usr
$ make
$ su -c "make install"
```

3.3. LADSPA

LADSPA stands for "Linux Audio Developer's Simple Plugin API".

Many audio synthesis and recording packages are in use or in development on Linux. These work in many different ways. LADSPA provides a standard way for "plugin" audio processors to be used with a wide range of these packages.

For instance, this allows a developer to make a reverb program and bundle it into a LADSPA "plugin library". Ordinary users can then use this reverb within any LADSPA-friendly audio application.

If you wonder if VST plugins can be used on GNU/Linux, the answer is "yes"[3]. Some works and others do not. Please, keep in mind that VST plugins are far to be Free as defined by the Free Software

Fundation. If you want some information about how to use them with GNU/Linux, please refer to the VST on Linux (http://gimpel.gi.funpic.de/Howtos/VST_on_Linux/_pages/1.html) Gimpel's Webspace Howto.

Most of the applications described in this Howto[4] are able to handle LADSPA plugins.

3.3.1. LADSPA plugins

You will find here informations for some well-known plugins. Feel free to visit the LADSPA homepage (http://www.ladspa.org/) to find other ones.

First off all, to compile LADSPA plugins you will need to install the ladspa.h (http://www.ladspa.org/ladspa_sdk/ladspa.h.txt) C header file in `/usr/include/` [5].

I mainly use the following plugins:

- **Steve Harris's LADSPA plugins.** Download them here (http://plugin.org.uk/releases/) [6].

  ```
  $ ./configure --prefix=/usr
  $ make
  $ su -c "make install"
  ```

- **Tom's Audio Processing LADSPA plugins.** Download them here (http://sourceforge.net/projects/tap-plugins). After uncompressing the plugins' archive go to the main directory and edit the `Makefile` file to change the following variables like this:

  ```
  INSTALL_PLUGINS_DIR     =       /usr/lib/ladspa/
  INSTALL_LRDF_DIR        =       /usr/share/ladspa/rdf/
  ```

  Then just compile it like other software:

  ```
  $ make
  $ su -c "make install"
  ```

- **CMT LADSPA plugins.** Download them here (http://www.ladspa.org/download/). After uncompressing the plugins' archive go to the main directory and edit the `src/makefile` file to change the following variable like this:

  ```
  INSTALL_PLUGINS_DIR     =       /usr/lib/ladspa/
  ```

  Then just compile it like other software [7] [8]:

  ```
  $ make
  $ su -c "make install"
  ```

- **CAPS Audio Plugin Suite.** Download them here (http://quitte.de/dsp/caps.html#Download). After uncompressing the plugins' archive go to the main directory and edit the `Makefile` file to change the following variable like this:

  ```
  PREFIX = /usr
  ```

  Then just compile it like other software[9]:

  ```
  $ make
  $ su -c "make install"
  ```

3.4. DSSI

DSSI (pronounced *dizzy*) is an API for audio plugins, with particular application for software synthesis plugins with native user interfaces. DSSI is an open specification developed for use in Linux audio applications, although portable to other platforms. It may be thought of as LADSPA-for-instruments, or something comparable to VSTi.

Download all packages (except the `dssi-vst` package[10]) from here (http://sourceforge.net/projects/dssi/).

> **Note:** Sometime DSSI is not even packaged in common GNU/Linux distros (like Debian for example), so you do not have other choice than building it yourself.

Uncompress each package and execute:

```
$ ./configure --prefix=/usr
$ make
$ su -c "make install"
```

for each one[11]. Just build and install the DSSI package first.

The output of the configuration step on my machine for the `dssi` package is:

```
* ======== DSSI 0.9.1 configured ========
* building jack-dssi-host:            yes
* building DSSI OSC tools:            yes
* building trivial_synth plugin:      yes
* building less_trivial_synth plugin: yes
* building trivial_sampler plugin:    yes
* building less_trivial_synth GUI:    yes
* building trivial_sampler GUI:       yes
```

3.5. Hydrogen

Hydrogen is an advanced drum machine for GNU/Linux. It's main goal is to bring professional yet simple and intuitive pattern-based drum programming.

Download it here (http://www.hydrogen-music.org/?p=download).

```
$ ./configure --prefix=/usr
$ make
$ su -c "make install"
```

The output of the configuration step on my machine is:

```
-----------------------------------------------------------------
 Hydrogen 0.9.2 configuration:
-----------------------------------------------------------------

        Source code location:   .
        Prefix                  /usr
        Data path               /usr/share/hydrogen
        LDFLAGS                  -lsndfile   -lFLAC++ -lFLAC -ljack -lasound -llrdf -lraptor -lxml2

        Debug messages:         no
        FLAC:                   yes
        LRDF:                   yes

        Jack:                   yes
        ALSA :                  yes
        OSS:                    yes
        PortAudio               no
        PortMidi:               no

        Features list =  (FLAC) (Jack) (Alsa) (OSS) (LRDF)
```

If you have problems while using Hydrogen, if it starts to react very sluggish on mouse-clicks and updating of the graphics is also very slow, try to apply this patch (http://www.esaracco.fr/downloads/SongEditor.cpp-hydrogen-0.9.2.patch). It come from the hydrogen-devel (http://lists.sourceforge.net/lists/listinfo/hydrogen-devel) mailing-list (you can read the message here (http://sourceforge.net/mailarchive/message.php?msg_id=12623228)). Apply it like this:

```
$ cd /usr/src/hydrogen-0.9.2/src/gui/SongEditor/
$ patch SongEditor.cpp /usr/src/SongEditor.cpp-hydrogen-0.9.2.patch
```

For those who use the brand new SVN release, use scons instead of make:

```
$ scons prefix=/usr
$ su -c "scons prefix=/usr install"
```

And for those, like me, who bought the wonderfull NS Kit7 (http://www.naturalstudio.co.uk/ns_kit7.html) from Natural Studio, I have developed a Perl script that built a Hydrogen's drumkit XML file from scratch, parsing NS Kit7 audio files. You can download it here (http://www.esaracco.fr/downloads/buildNSKit7.pl).

3.6. QSynth

QSynth is a Fluidsynth GUI front-end application written in C++ around the Qt3 toolkit using Qt Designer. Eventually it may evolve into a softsynth management application allowing the user to control and manage a variety of command line softsynth but for the moment it wraps the excellent FluidSynth. FluidSynth is a command line software synthesiser based on the Soundfont specification.

Download it here (http://sourceforge.net/projects/qsynth).

It could be useful to use soundfonts if you do not have a Soundblaster card. See Section 4.2.2.2.

```
$ ./configure --prefix=/usr
$ make
$ su -c "make install"
```

3.7. SpiralSynthModular

SpiralSynthModular is an object orientated music studio with an emphasis on live use. You can use it in a fairly straight forward way to make tracks with, or get very experimental. Audio or control data can be freely passed between the plugins. Data can also be fed back on itself for chaotic effects.

SpiralSynthModular is not under active development. You can give a try to Om (http://www.nongnu.org/om-synth/) if you want.

You can dowload SpiralSynthModular here (http://sourceforge.net/project/showfiles.php?group_id=62620). However I advise you to work with the last CVS version.

```
$ cvs -d:pserver:anonymous@spiralmodular.cvs.sourceforge.net:/cvsroot/spiralmodular login
$ cvs -z3 -d:pserver:anonymous@spiralmodular.cvs.sourceforge.net:/cvsroot/spiralmodular co -P ssm
```

Then build it with[12]:

```
$ ./autogen.sh --prefix=/usr
$ make
$ su -c "make install"
```

3.8. ZynAddSubFX

> **Warning**
>
> ZynAddSubFX will not be able to read neither its own configuration file nor instruments patches with 2.3 MiniXML (mxml) library (http://www.minixml.org) release (but it is ok with >= 2.4). If you can not upgrade to 2.4 you can apply this patch (http://www.esaracco.fr/downloads/mxml-file.c.patch) on the mxml `mxml-file.c` file to fix the problem.

ZynAddSubFX is a opensource software synthesizer capable of making a countless number of instruments, from some common heared from expensive hardware to interesting sounds that you'll boost to an amazing universe of sounds.

You can either download it here (http://sourceforge.net/projects/zynaddsubfx), or decide to build it from the CVS: see here (http://sourceforge.net/cvs/?group_id=62934) for information. To get the current CVS HEAD, do the following:

```
$ cvs -d:pserver:anonymous@zynaddsubfx.cvs.sourceforge.net:/cvsroot/zynaddsubfx login
$ cvs -z3 -d:pserver:anonymous@zynaddsubfx.cvs.sourceforge.net:/cvsroot/zynaddsubfx co -P zynaddsubfx
```

Then do the following:

Then go to the `src/` subdirectory and edit the `Makefile.inc` file. Then uncomment/comment lines to suit your needs. Significant lines are:

```
OS_PORT=LINUX
FFTW_VERSION=3
ASM_F2I=YES
DISABLE_GUI=NO
LINUX_MIDIIN=ALSA
LINUX_AUDIOOUT=OSS_AND_JACK
LINUX_DSSI=NO
```

Then make sure that the following packages are installed on your system: `fluid`, `libfltk` and `libmxml`.

To build and install ZynAddSubFX, just do [13]:

```
$ make
$ su
# cp zynaddsubfx /usr/bin/
# cd ..
# mkdir -p /usr/share/zynaddsubfx
# /bin/cp -a banks presets /usr/share/zynaddsubfx/
```

You will find up-to-date instruments banks here (http://zynaddsubfx.sourceforge.net/doc/instruments/).

3.9. Rosegarden

Rosegarden is a professional audio and MIDI sequencer, score editor, and general-purpose music composition and editing environment.

If you want to use it efficiently, D. Michael McIntyre (aka. Silvan) wrote a *The Rosegarden Companion* (http://rosegarden.sourceforge.net/tutorial/), a great little book about Rosegarden, and also a lot of very interesting and good written other supplemental tutorials from Silvan.

> **Note:** Most of the time Rosegarden which is distributed with common GNU/Linux distros was not built with DSSI support. Manually rebuilding it is the only way to have DSSI plugins working with it.

You can either download the stable archive, or decide to build it from the SVN: see here (http://www.rosegardenmusic.com/getting/) for information. To get the current SVN trunk, do the following:

```
svn co https://svn.sourceforge.net/svnroot/rosegarden/trunk/rosegarden rosegarden
```

I use the SVN trunk of Rosegarden. And when things go wrong I edit Rosegarden files (`.rg`) manually to repair them. Rosegarden's files are XML files compressed with gzip[14].

```
$ cmake . -DCMAKE_INSTALL_PREFIX=`kde-config --prefix`
```

With my configuration this step produces:

```
-- Found Qt3 (version 3.3.7)
-- Found KDE3 include dir: /usr/include/kde
-- Found KDE3 library dir: /usr/lib
-- Found KDE3 dcopidl preprocessor: /usr/bin/dcopidl
-- Found KDE3 dcopidl2cpp preprocessor: /usr/bin/dcopidl2cpp
-- Found KDE3 kconfig_compiler preprocessor: /usr/bin/kconfig_compiler
-- Program pkg-config found (/usr/bin/pkg-config)
-- Found LADSPA (/usr/include)
-- Program msgfmt found (/usr/bin/msgfmt)
-- KDE3HTMLDIR : share/doc/kde/HTML
```

```
-- KDE3DATADIR : share/apps
-- KDE3ICONDIR : share/icons
-- KDE3MIMEDIR : share/mimelnk
-- KDE3MENUDIR : share/applications/kde
-- KDE3L18NDIR : share/locale
-- KDE3EXECDIR : bin
-- Program kde-config found (/usr/bin/kde-config)
-- Program meinproc found (/usr/bin/meinproc)
-- Rosegarden 1.7.0-svn will be built for install into /usr
-- Configured to compile for release without debug information

Installation Summary
--------------------

Install Directory             : /usr
Build type                    : Release
Use Qt/KDE precompiled headers: FALSE

Xft notation font support     : TRUE
No LIRC support configured.

ALSA MIDI support             : TRUE
JACK audio support            : TRUE
LADSPA plugin support         : TRUE
DSSI synth plugin support     : TRUE
Custom OSC plugin GUI support : TRUE
Audio timestretching          : TRUE
LRDF plugin metadata support  : TRUE

-- Configuring done
-- Generating done
-- Build files have been written to: /usr/src/audio/rosegarden
```

Then build and install it. The process can take some time, so be patient :-)

```
$ make
$ su -c "make install"
```

3.10. LMMS - Linux Multimedia Studio

LMMS aims to be a free alternative to popular (but commercial and closed-source) programs like FruityLoops, Cubase and Logic giving you the ability of producing music with your computer by creating cool loops, synthesizing and mixing sounds, arranging samples, having more fun with your MIDI-keyboard and much more....

You can either download the stable archive here (http://sourceforge.net/project/showfiles.php?group_id=105168&package_id=113209), or decide to build it from the SVN: see here (http://lmms.sourceforge.net/download.php) for information. To get the current SVN trunk, do the following:

```
svn co https://svn.sf.net/svnroot/lmms/trunk/lmms
```

Then do a **`autoreconf -is`** followed by a **`./configure --prefix=/usr`** and take a look to the output. If you see something like that:

```
[...]
=========================
 ===  LMMS - WARNING  ============================================================
 =========================
 =
 = You don't seem to have SDL_sound-library installed and/or SDL_sound-
 = development-package is missing. This means that you can only load samples
 = within LMMS if you have libsndfile and libvorbis installed!
 = Consider installing the missing packages for using the full power of LMMS.
 =
[...]
```

Install the `libsdl-sound1.2` and `libsdl-sound1.2-dev` packages.

Then do the following:

```
$ make
$ su -c "make install"
```

3.11. Ardour

Ardour is a digital audio workstation (DAW). You can use it to record, edit and mix multi-track audio. You can produce your own CDs, mix video soundtracks, or just experiment with new ideas about music and sound.

You can either download the stable archive, or decide to build it from the SVN : see here (http://www.ardour.org/download_full) for information.

Download it from here (http://www.ardour.org/download_full), then :

```
$ scons PREFIX=/usr
$ su -c "scons install"
```

3.12. JAMin

JAMin is the JACK Audio Connection Kit (JACK) Audio Mastering interface. JAMin is an open source application designed to perform professional audio mastering of stereo input streams. It uses LADSPA for digital signal processing (DSP). JAMin is licensed under the GPL.

You can either download the stable archive here (http://sourceforge.net/project/showfiles.php?group_id=78441), or decide to build it from the CVS: see here (http://sourceforge.net/cvs/?group_id=78441) for information. To get the current CVS HEAD, do the following:

```
$ cvs -d:pserver:anonymous@jamin.cvs.sourceforge.net:/cvsroot/jamin login
$  cvs -z3 -d:pserver:anonymous@jamin.cvs.sourceforge.net:/cvsroot/jamin co -P jamin
```

Then do the following:

```
$ ./autogen.sh --prefix=/usr
$ make
$ su -c "make install"
```

3.13. Jack Timemachine

JACK audio recorder for spontaneous and conservatory use.

Download it here (http://plugin.org.uk/timemachine/).

I use it very often to capture Rosegarden output and to convert in WAV format, before encoding it with Oggenc or Lame. It writes WAV files with a sample format of 32bit float, which is not appropriate to my configuration. So I must use Rezound to transform it to a **signed** sample format with a sample width of **24**.

You can capture all that Jack can manage. It is especially great when you just want to grabe what actually sounds on your soundcard (audio CD, sounds of video DVD...).

```
$ ./configure --prefix=/usr
$ make
$ su -c "make install"
```

3.14. Rezound

ReZound aims to be a stable, open source, and graphical audio file editor primarily for but not limited to the Linux operating system.

Download it here (http://sourceforge.net/project/showfiles.php?group_id=5056).

I use it as often as I use Timemachine to adapt audio files in a suitable format and to add some final effects [15].

```
$ ./configure --prefix=/usr --enable-largefile
$ make
$ su -c "make install"
```

3.15. Lame

LAME is an LGPL MP3 encoder. The Open source development model allowed to improve its quality and speed since 1999. It is now an highly evolved MP3 encoder, with quality and speed able to rival state of the art commercial encoders.

Download it here (http://lame.sourceforge.net/download/download.html).

```
$ ./configure --prefix=/usr
$ make
$ su -c "make install"
```

Notes

1. Most of the time, GNU/Linux distros are not up-to-date, and audio software teams are often very active. Building those software by yourself enables you firstly to have a better understanding of their use and their interactions and secondly to experiment most advanced features and to fix bugs quickly.
2. To use another compiler than your system default, just update the following symbolic links: `/usr/bin/gcc` and `/usr/bin/g++` to point on, respectively (to use gcc 4.x): `/usr/bin/gcc-4.x` and `/usr/bin/g++-4.x`.
3. Using WINE (http://www.winehq.com/).
4. Like Rosegarden, Hydrogen, Rezound, Ardour...
5. Of cource, you will have to rename it from `ladspa.h.txt` to `ladspa.h`
6. If you have problem on `x86_64` architecture, just try to replace all `-march=x86_64` by `-march=x86-64`" after executing **configure**, it should do the trick.
7. Note that those plugin will not build with a g++ version >= 4. To fix the problem, just edit the `src/analogue.cpp` and add **`a = b = c = 0;`** to the line `264` (just after variables declaration).
8. If you experience problem on `x86_64` architecture, just add `-fPIC` in the `Makefile` file, line `48`: **`$(CC) $(CFLAGS) -fPIC -I/usr/local/include -c $<`**
9. Note that with versions < 0.3.0 this plugin will not build with a g++ version >= 4. To fix the problem, download the following Mario Lang's patch (http://quitte.de/dsp/caps-0.2.3+gcc-4.diff) and apply it like that:

   ```
   $ cd /usr/src/
   $ wget http://quitte.de/dsp/caps-0.2.3+gcc-4.diff
   $ cd caps-0.2.3/
   $ patch -p1 < ../caps-0.2.3+gcc-4.diff
   ```

10. I will not tell you how to work with VST plugins in this tutorial. For details about VST support, see Section 3.3
11. The Hexter plugin will not compile if your gcc version is >= 4.

12. Note that if you use a `x86_64` arch you will need to apply a patch (http://www.esaracco.fr/downloads/LADSPAInfo.C.patch) in order to have LADSPA plugins working.

```
$ cd ssm/SpiralSound/Plugins/LADSPAPlugin/
$ wget http://www.esaracco.fr/downloads/LADSPAInfo.C.patch
$ patch LADSPAInfo.C LADSPAInfo.C.patch
```

13. If you experiment problems with CVS version of ZynAddSubFX and SVN version of Jack (>= 998), try to apply the following patch (http://www.esaracco.fr/downloads/zyn-jack-midi-adjust.patch).
14. You will find tools like **rgedit** and **rgview** (to manipulate `.rg` files) in the `scripts/` subdirectory of the CVS tree.
15. Rezound will failed to build on 64-bit arch. To build it with success, apply this patch (http://www.esaracco.fr/downloads/rezound-0.12.2beta-64-bit.patch) (taken from this thread (http://sourceforge.net/mailarchive/forum.php?thread_id=8925105&forum_id=18767)) like that:

```
$ cd /usr/src/
$ wget http://www.esaracco.fr/downloads/rezound-0.12.2beta-64-bit.patch
$ cd rezound-0.12.2beta/
$ patch -p0 < ../rezound-0.12.2beta-64-bit.patch
```

Chapter 4. Soundfonts

Just look at the Wikipedia soundfont definition (http://en.wikipedia.org/wiki/Soundfont) before reading this chapter.

Once you have software to make some noise, you must use some soundfont that contains those great noises you want :-) If you have a Soundblaster card you can even directly load those soundfonts on the card. Others will use a soft synthetizer and work with soundfonts this way.

4.1. Getting soundfonts

See Section A.3 for non-commercial soundfonts web sites links.

I mainly use 2 of them. You will find them here:

- PC51f (ftp://ftp.personalcopy.net/pub/PC51f.sf2.gz) (compressed - 51M)- The one I use mainly. Good, except for violins.
- Fuild (release 3) (http://www.hammersound.net/cgi-bin/soundlink.pl?action=view_download_page;ID=699) (compressed - 68M) - Good for orchestral strings (especially violins and violas).

You will find a lot of soundfonts archives compressed with the sfArk (http://melodymachine.com/sfark.htm) tool. You can download it from here (http://melodymachine.com/files/sfarkxtc_lx86.tar.gz).

To uncompress a file with it, just uncompress it with **`tar zxvf -C /usr/local/bin/`** and execute: **`sfarkxtc FluidR3\ GS.sfArk /tmp/fluidr3.sf2`**.

4.2. Using soundfonts

Soundfonts can be loaded on soundcard, or used with a soft synthetizer.

4.2.1. With Soundblaster soundcard

If you have a Soundblaster soundcard, install a package named `awesfx`. This package should contain the asfxload application. This little tool will help us to load the soundfonts in the soundcard.

Once you have downloaded and uncompressed your soundfont, just execute something like **`asfxload /tmp/fluidr3.sf2`**, and wait for a while.

A software like Rosegarden is able to load sounfount for you at startup (see Section 5.4).

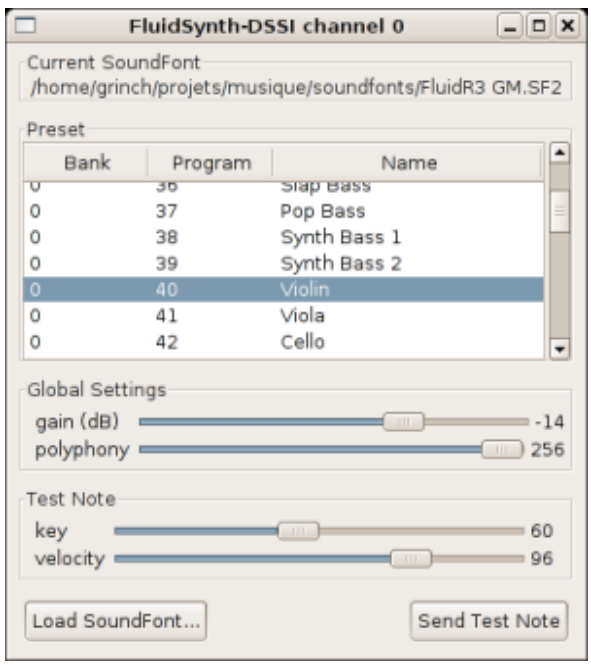
4.2.2. With other soundcards

It exists a lot of pretty good synthetizers. Some of them have more features than others, but most of them allow you to load your own soundfonts. In those examples we will use one of the DSSI plugins we have built: `fluidsynth-dssi.so` (see Section 3.4), and QSynth (see Section 3.6).

4.2.2.1. Testing with DSSI plugins

To use directly DSSI plugins you must at first launch Jack server. To do that, just execute Qjackctl as normal user (see Section 5.1). Then we use the Jack DSSI host program to load and execute the Fluidsynth DSSI plugin with **`jack-dssi-host fluidsynth-dssi.so`**. You will see a little graphic interface.

Figure 4-1. Fluidsynth DSSI plugin graphic interface

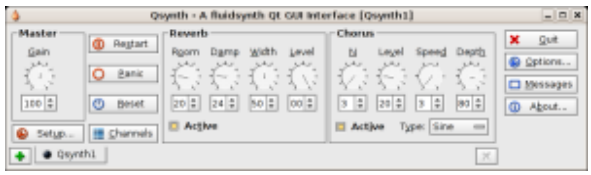


Click on the Load Soundfont... button to load a soundfont, choose any instrument and click on Send test note to hear its sound.

4.2.2.2. Testing with QSynth

First, execute Qjackctl as normal user (see Section 5.1) to launch Jack server. Then execute QSynth. In fact, QSynth, like other audio software, can run with or without Jack, directly using OSS, ALSA etc.

Figure 4-2. QSynth graphic interface



Click on the Setup button, go to the Soundfonts notebook and load at least one soundfont. Now QSynth is ready for giving us sound when we will need it (see Section 5.4.2).

Chapter 5. Create music

Now that we have built most needed software, we can begin to discover the GNU/Linux sound creation world. First I must say that I systematically use Jack to interconnect sotwares between them, even if most of them can run without Jack. It is a useful application that permits to play with almost everything that claim to be a Jack client (and all software here are Jack-compliant).

5.1. Execute Jack at first

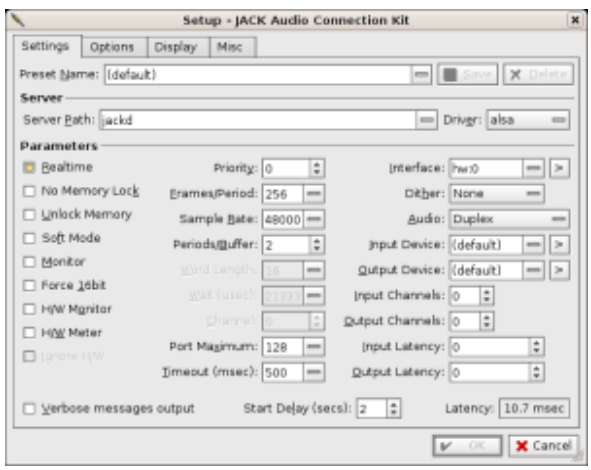
Note: To know how to build Jack server, see Section 3.1.

To know how to build Qjackctl, see Section 3.2.

Setting Jack can be a nightmare, but sometimes it works more or less as you expected it to work (that is to say without too much `xruns`). We will not explain here what `xruns` are, it is enough to notice that they are bad. They must not appear while applications are running. But they can sometimes appear when you shutdown them. Not really bad in this case.

You can either execute Jack server as command line or use Qjackctl. I recommand using Qjackctl, for it makes connection/disconnection between Jack clients more simple. It you have chosen to use set_rlimits instead of Realtime LSM or PAM to manage realtime privileges access (see Section 1.4), do not forget to execute those applications prefixing them with **`set_rlimits -r`**.

Figure 5-1. Jack server main setup window



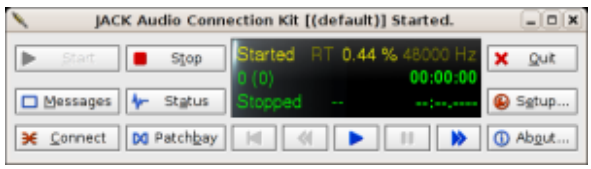
Things like `/usr/bin/jackd -R -dalsa -dhw:0 -r48000 -p128 -n2` should be reasonable. Adapt values to your hardware's configuration.

Read carefully the **jackd** manpage and do not hesitate to try different values, especially for the "Frame/Period" parameter (`-p` command line argument):

```
-p, --period int
              Specify the number of frames between JACK process() calls.  The
              default  is 1024.  If you need low latency, set -p as low as you
              can go without seeing xruns.  A larger period size yields higher
              latency,  but makes xruns less likely.  The JACK capture latency
              in seconds is --period divided by --rate.
```

Now Jack is up and running.

Figure 5-2. Jack server main window



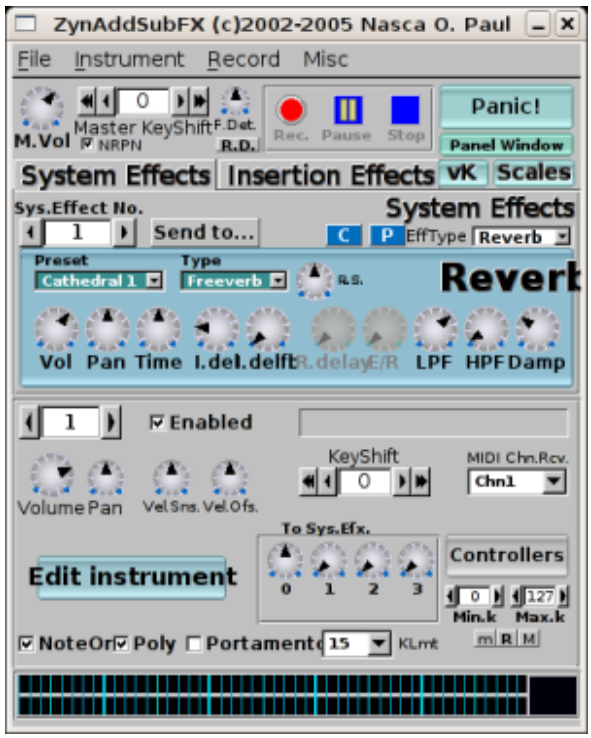
5.2. Playing with ZynAddSubFX

> **Note:** To know how to build ZynAddSubFX, see Section 3.8.

You can use ZynAddSubFX alone to understand what it is and how it works before using it with an audio sequencer like Rosegarden. We will only see here the basics of this powerful software. For more information, see its Homepage (http://zynaddsubfx.sourceforge.net/).

> **Note:** If the internal samplerate of your Soundward is different than `44100` (it is to say `48000` for example), launch it like that: **`zynaddsubfx --sample-rate=48000`**.

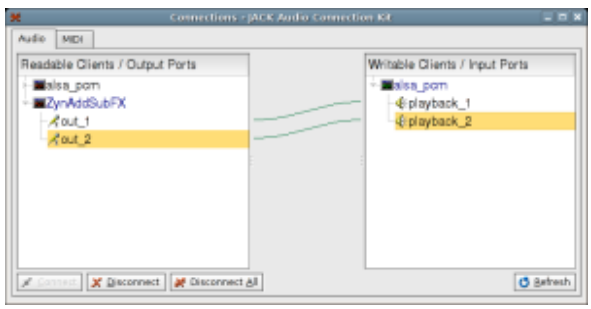
Figure 5-3. ZynAddSubFX main window



Graphic interface is nice, but fonts are always a little bit to big for widgets. I do not know how to reduce them [1].

Anyway, once you are here you should see ZynAddSubFX line in the Readable Clients/output Ports part of the connexion manager screen.

Figure 5-4. Jack connection manager/ZynAddSubFX client



Do appropriate connections between ZynAddSubFX outputs and your soundcard inputs, as seen in screenshot Figure 5-4 [2].

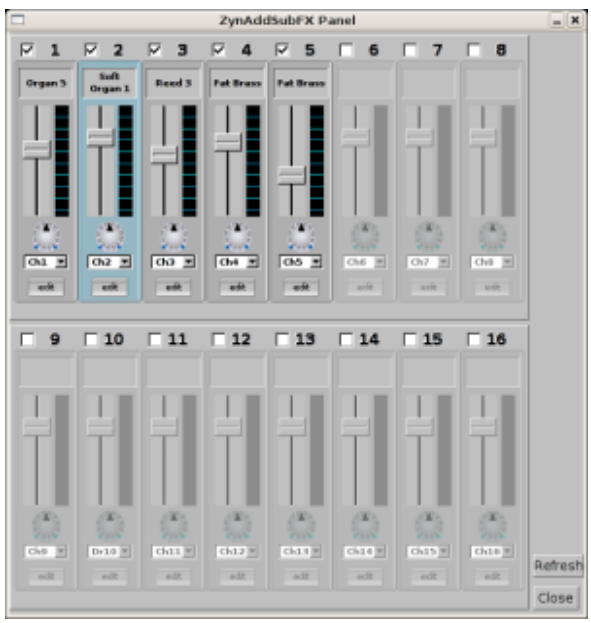
Now go to the main ZynAddSubFX screen, open the Instrument⟶Virtual keyboard... menu.

Figure 5-5. ZynAddSubFX virtual keyboard

If you either click on the virtual synthetizer keys or press your computer keyboard keys you should hear some sound. You are hearing the default song of the ZynAddSubFX bank. To change instrument, open the Instrument⟶Show instrument bank menu. Then choose a bank and click on a patch. You should see the patch name displayed in the main window. But for now you are just working with 1 track.

To add other tracks and control their volume in an easy way, just click on the panel Window button in the main window. You will see a mixer like this:

Figure 5-6. ZynAddSubFX multitracks mixer



You can do a lot of things with ZynAddSubFX. Play with it a while.

5.3. Playing with Hydrogen

Note: To know how to build Hydrogen, see Section 3.5.

Hydrogen is a good alternative to GM drum kits. It is really specialized in percussions pattern creation.

Figure 5-7. Hydrogen main window

This application is simple to use. Just create patterns in the "Pattern editor" and chain them in the "Song editor".

5.4. Playing with Rosegarden

Note: To know how to build Rosegarden, see Section 3.9.

I really appreciate Rosegarden. It is a really amazing software, and it evolves in a good way. Its developers are kind and do a great job. Thanks to them.

I will neither show you here all features, nor explain you in details how to use it. I will just try to show you the way to begin. For more informations, take a look at this page (http://rosegarden.sourceforge.net/tutorial/).

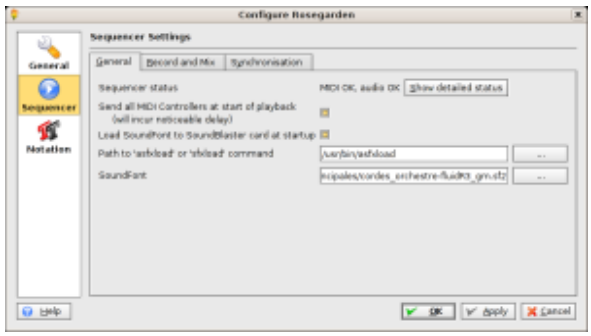
I mainly use Rosegarden to do MIDI compositions, and use its great notation editor. But you can do a lot of things with it:

- Connect external MIDI instruments, record them and see the corresponding notation.
- Connect accoustic instruments, record them and apply effects.
- ...

5.4.1. Rosegarden settings

We will just see here the main settings of Rosegarden, that is to say *Sequencer* settings.

Figure 5-8. Rosegarden General settings



Click on the Settings⟶Configure Rosegarden... menu to raise the windows settings. Then click on the Sequencer button.

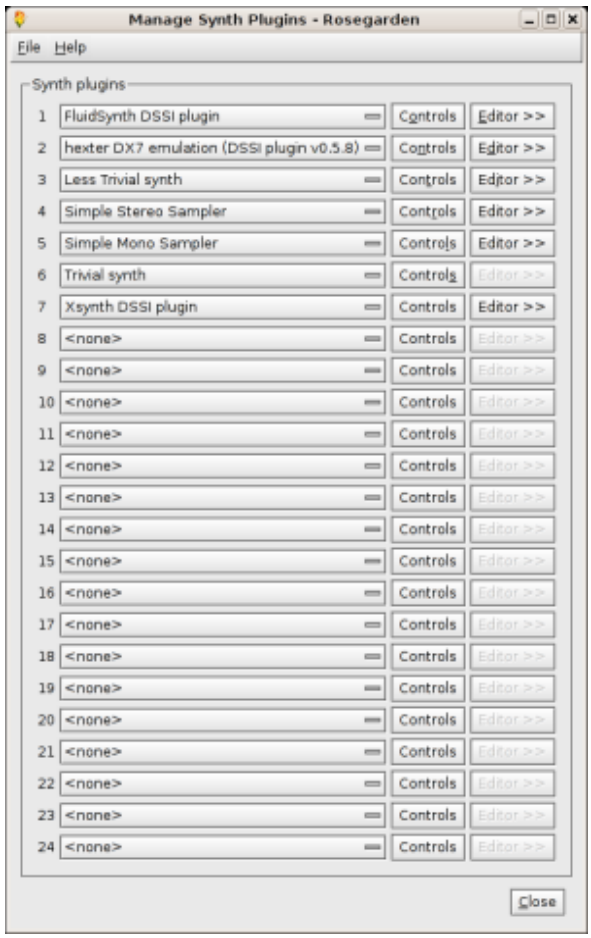
To load a specific Soundfont at Rosegarden startup, just click to the Load SoundFont to SoundBlaster card at startup radio button, enter the path to the **asfxload** command and select a Soundfont to load. See Chapter 4 for Soundfonts informations. Save your choices.

Next go to the Composition⟶Edit Document Properties... menu. Then click on the Audio button and choose the directory where Rosegarden will save your audio tracks. Audio tracks are tracks on which you have recorded external instruments. For the moment, just choose a base directory on a well dimensioned partition (audio files can be pretty large). Later we will adapt subdirectories for each of our songs. Save your choices.

Now that we have customized settings we can tell Rosegarden to save them as the default for all futur new projects by clicking on the Composition⟶Studio⟶Save Current Document as Default Studio menu.

To check that all DSSI plugins we have compiled have been loaded by Rosegarden, click on the  icon on the main menu. You should see some lines and have the choice to select plugins in combo boxes. If not, go to Section 3.4 and check what is wrong in you configuration.

Figure 5-9. Rosegarden synth plugins manager



5.4.2. Using soft synthetizers

Using Rosegarden with a soft synthetizer is as easy as executing both your soft synthetizer and Rosegarden at the same time. As a example we will use the fuidsynth graphic frontend: QSynth.

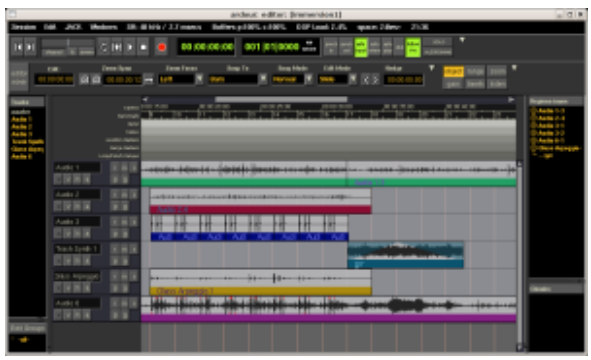
5.4.3. Recording audio tracks

5.5. Playing with Ardour

Note: To know how to build Ardour, see Section 3.11.

Ardour is a rather complex application, so I advise you to refer to its documentation (http://ardour.org/files/manual/index.html).

Figure 5-10. Ardour main window



5.6. Playing with JAMin

Note: To know how to build JAMin, see Section 3.12.

Masterisation is a complex step and it exists a lot of books and tutorial.

Figure 5-11. JAMin main window

Here is the advices of one of the Ardour's developers on using JAMin with Ardour:

1. Open up either the editor mixer for the master bus or use the mixer window
2. Right click on the black box above the master fader to open the context menu
3. Select New Insert (it will automatically be a stereo insert)
4. "Edit" the insert either via right clicking on the insert and selecting Edit, or "edit click" (**CTRL**-**Right** click) the insert
5. Connect JAMin inputs and outputs to the send outputs and inputs
6. "Activate" the insert by either right clicking on the insert and selectiong Activate or middle-click on the insert

Notes

1. Note from Ismael Cortes: "The only way I have found to get rid of that is to recompile FLTK (and maybe ZynAddSubFX, if it was compiled with FLTK statically) without the xft option configured (i.e. `--disable-xft` or just omitting `--enable-xft`, since it defaults to off). That seems to get the fonts were they should."
2. If you are lazy like me and do not want to manually connect it every time you launch it, just uncomment the 2 following lines in `src/Output/JACKaudiooutput.C`:

```
/*
  jack_connect(jackclient,jack_port_name(outport_left),"alsa_pcm:out_1");
  jack_connect(jackclient,jack_port_name(outport_right),"alsa_pcm:out_2");
*/
```

and replace `out_1` by `playback_1` and `out_2` by `playback_2`. Then rebuild ZynAddSybFX as explained in Section 3.8.

Chapter 6. Capturing and encoding

Once we have created something and that we want to share it with others, we must capture and encode it. The capture can be easily done in WAV format with Timemachine or in a more complex way with Ardour. Then we will encode it in other formats to share it.

6.1. Playing with Timemachine

Note: To know how to build Timemachine, see Section 3.13.

Timemachine allows you to capture everything that can be understood by Jack server.

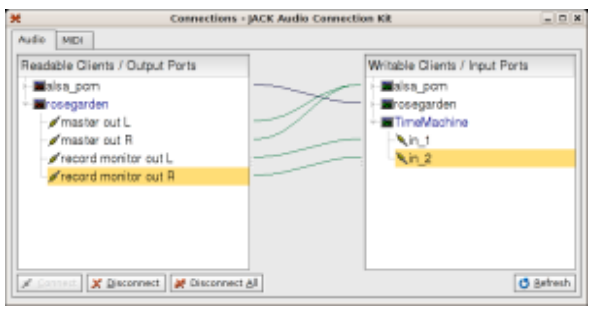
The first step is to launch Timemachine with some arguments to tell it to capture in WAV format and to manage its files in our Desktop directory:

```
timemachine -f wav -p /home/youruser/Desktop/
```

Just replace the **`youruser`** string by your current Unix user.

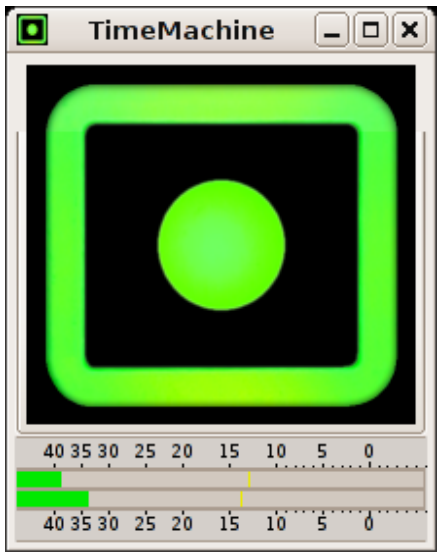
Once Timemachine is started it is necessary to manually connect the required Jack's clients outputs to Timemachine's inputs. Saying you want to record a Rosegarden composition, just connect Rosegarden's output 1 record monitor out L to Timemachine's input in_1 and Rosegarden's output 2 record monitor out R to Timemachine's input in_2, as seen in Figure 6-1.

Figure 6-1. Jack connection manager/Timemachine client



Now you can click on the main Timemachine green button, and play your Rosegarden song.

Figure 6-2. Timemachine



6.2. Playing with Rezound

Note: To know how to build Rezound, see Section 3.14.

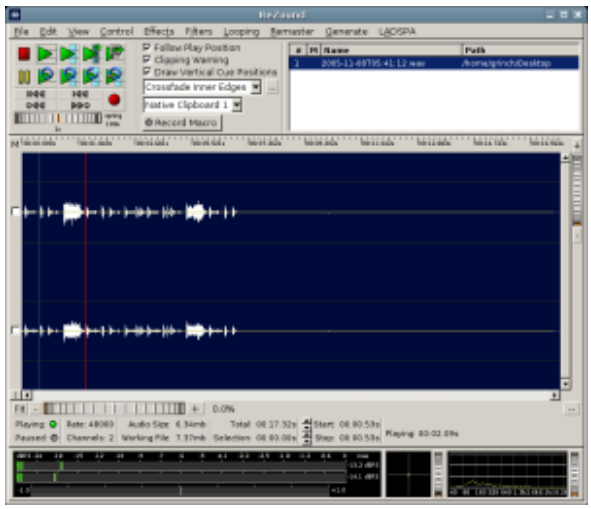
The WAV file produced by Timemachine must be edited with Rezound to remove extra parts, control volume level or apply effects.

To launch Rezound as a Jack client do:

```
rezound --audio-method=jack
```

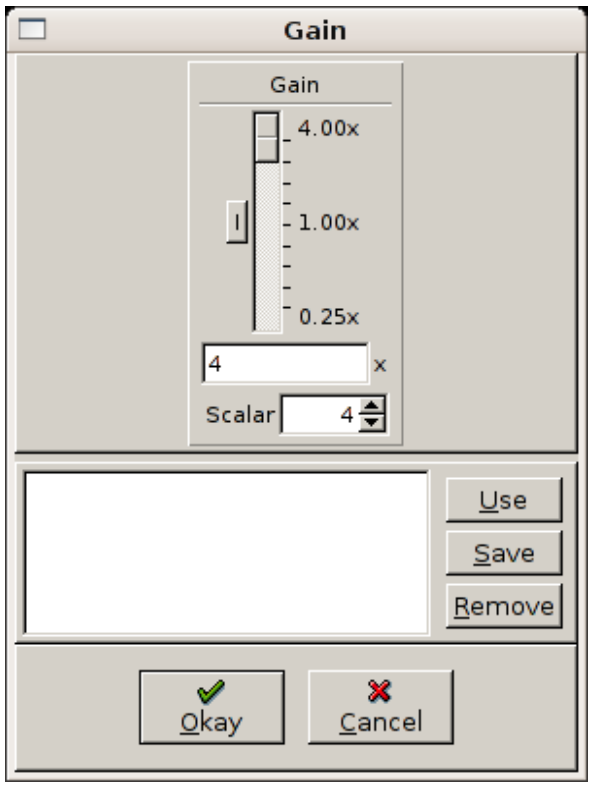
Once your file is loaded, cut the beginning and the ending to fit your needs.

Figure 6-3. Rezound main window



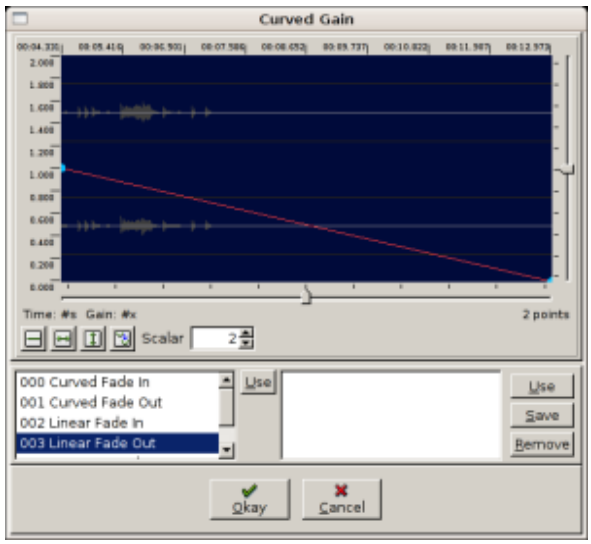
I am used to raise the volume level. To do that, select the entire song with **CTRL**-**A** and click on the main menu to Effects⟶Gain. Then enter a scalar value of `3` or `4`, raise the Gain level and click on the Okay button to apply changes on the current file.

Figure 6-4. Rezound Gain Control window



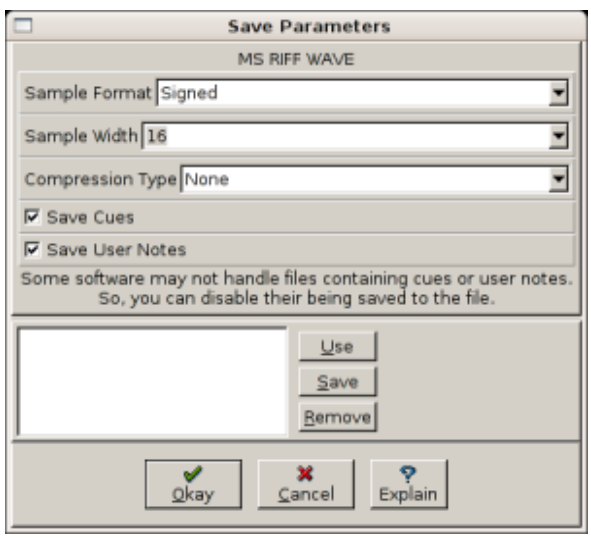
You can also apply a fade out at the end. Select the region you want to apply a fade out, then click on the main menu to Effects⟶Curved gain. Choose Linear Fade Out or anything else by double-clicking on it and click on the Okay button.

Figure 6-5. Rezound Curved Gain window



Then save your file. When saving the file, Rezound will popup a dialog that will allow you to change properties of the WAV encoding. With my configuration I must save WAV with a **`signed`** sample format and a width format of **`24`**. If I do not do that, Oggenc OGG encoding will turn in nothing else than a noisy file [1].

Figure 6-6. Rezound save window



6.3. Share your work

Note: To know how to build Lame, see Section 3.15.

Now that we have our song in WAV format, we will convert it in OGG and MP3 format to share it more easily.

oggenc and **lame** allow us to specify some extra attributs for our songs, like its title or its genre (see manpages). Usually I only specify title and genre.

To obtain a list of genres, pass the **`--genre-list`** argument to Lame. It will output the following:

Example 6-1. Music genre's list for OGG and MP3 encoding

```
123 A Cappella
 34 Acid
 74 Acid Jazz
 73 Acid Punk
 99 Acoustic
 20 Alternative
 40 Alt. Rock
 26 Ambient
145 Anime
 90 Avantgarde
116 Ballad
 41 Bass
135 Beat
 85 Bebob
 96 Big Band
138 Black Metal
 89 Bluegrass
  0 Blues
107 Booty Bass
132 BritPop
 65 Cabaret
 88 Celtic
104 Chamber Music
102 Chanson
 97 Chorus
136 Christian Gangsta Rap
 61 Christian Rap
141 Christian Rock
 32 Classical
  1 Classic Rock
112 Club
128 Club-House
 57 Comedy
140 Contemporary Christian
```

```
  2 Country
139 Crossover
 58 Cult
  3 Dance
125 Dance Hall
 50 Darkwave
 22 Death Metal
  4 Disco
 55 Dream
127 Drum & Bass
122 Drum Solo
120 Duet
 98 Easy Listening
 52 Electronic
 48 Ethnic
 54 Eurodance
124 Euro-House
 25 Euro-Techno
 84 Fast-Fusion
 80 Folk
115 Folklore
 81 Folk/Rock
119 Freestyle
  5 Funk
 30 Fusion
 36 Game
 59 Gangsta Rap
126 Goa
 38 Gospel
 49 Gothic
 91 Gothic Rock
  6 Grunge
129 Hardcore
 79 Hard Rock
137 Heavy Metal
  7 Hip-Hop
 35 House
100 Humour
131 Indie
 19 Industrial
 33 Instrumental
 46 Instrumental Pop
 47 Instrumental Rock
  8 Jazz
 29 Jazz+Funk
146 JPop
 63 Jungle
 86 Latin
 71 Lo-Fi
 45 Meditative
142 Merengue
  9 Metal
 77 Musical
```

```
 82 National Folk
 64 Native American
133 Negerpunk
 10 New Age
 66 New Wave
 39 Noise
 11 Oldies
103 Opera
 12 Other
 75 Polka
134 Polsk Punk
 13 Pop
 53 Pop-Folk
 62 Pop/Funk
109 Porn Groove
117 Power Ballad
 23 Pranks
108 Primus
 92 Progressive Rock
 67 Psychedelic
 93 Psychedelic Rock
 43 Punk
121 Punk Rock
 15 Rap
 68 Rave
 14 R&B
 16 Reggae
 76 Retro
 87 Revival
118 Rhythmic Soul
 17 Rock
 78 Rock & Roll
143 Salsa
114 Samba
110 Satire
 69 Showtunes
 21 Ska
111 Slow Jam
 95 Slow Rock
105 Sonata
 42 Soul
 37 Sound Clip
 24 Soundtrack
 56 Southern Rock
 44 Space
101 Speech
 83 Swing
 94 Symphonic Rock
106 Symphony
147 Synthpop
113 Tango
 18 Techno
 51 Techno-Industrial
```

```
130 Terror
144 Thrash Metal
 60 Top 40
 70 Trailer
 31 Trance
 72 Tribal
 27 Trip-Hop
 28 Vocal
```

A basic OGG encoding could be done with:

```
$ oggenc -a"Emmanuel Saracco" -t"Blackbird" -G"Fusion" -d2005-08-23 blackbird.wav
```

Where `-a` is for author name, `-t` for title, `-G` for genre, and `-d` for entire date.

A basic MP3 encoding could be done with:

```
$ lame -s 48 --ta "Emmanuel Saracco" --tt "Blackbird" --tg"Fusion" --ty "2005" blackbird.wav blackbir
```

Where `--ta` is for author name, `--tt` for title, `--tg` for genre, and `--ty` for year.

> **Note:** If you need to convert a old MP3 file to OGG, you can use **ecasound** like the following:
>
> ```
> $ ecasound -i blackbird.mp3 -o blackbird.ogg
> ```

Notes

1. However Lame MP3 encoding can be done on WAV in `signed 32` format without any problem.

Appendix A. Useful Audio Links

A.1. Mailing-lists

Linux Audio Users list (http://music.columbia.edu/mailman/listinfo/linux-audio-user)

Rosegarden users list (http://lists.sourceforge.net/lists/listinfo/rosegarden-user)

Ardour users list (http://lists.ardour.org/listinfo.cgi/ardour-users-ardour.org/)

Hydrogen users list (http://lists.sourceforge.net/lists/listinfo/hydrogen-users/)

A.2. Web sites

Dogmazic.net, Free music (http://www.dogmazic.net/)

Elody, a music composition environment (http://www.grame.fr/Elody/)

Ubuntu Studio (http://ubuntustudio.com/)

MAO Libre [fr] (http://www.linuxmao.org/)

Sound & MIDI Software For Linux (http://linux-sound.org/)

The Freesound Project (http://freesound.iua.upf.edu/)

A multi-lingual dictionary of musical terms (http://www.cadenza.org/glossary/)

Musique et audio [fr] (http://logiciels-libres-cndp.ac-versailles.fr/rubrique.php3?id_rubrique=16)

Creative Commons License for Audio work (http://creativecommons.org/audio/)

Rosegarden (http://www.rosegardenmusic.com/)

Jack Audio Connection Kit (http://jackaudio.org/)

QJackctl (http://qjackctl.sourceforge.net/)

DSSI (http://dssi.sourceforge.net/)

LADSPA (http://www.ladspa.org/)

Steve Harris's plugins (http://plugin.org.uk/)

Tom's Audio Processing plugins (http://tap-plugins.sourceforge.net/)

Hydrogen (http://www.hydrogen-music.org/)

QSynth (http://qsynth.sourceforge.net/qsynth-index.html)

ZynAddSubFX (http://zynaddsubfx.sourceforge.net/)

Ardour (http://ardour.org/)

Jack Timemachine (http://plugin.org.uk/timemachine/)

Rezound (http://rezound.sourceforge.net/)

Audacity (http://audacity.sourceforge.net/)

EasyTAG (http://easytag.sourceforge.net/)

How to create a Audio CD (http://gentoo-wiki.com/HOWTO_Create_an_Audio_CD)

A.3. Soundfonts

PC51f soundfont (ftp://ftp.personalcopy.net/pub/PC51f.sf2.gz)

SGM180 soundfont (ftp://sf2midi.com/sgm128/SGM180v1.5.zip)

The Freepats project (http://freepats.opensrc.org/)

sf2 MIDI
(http://www.sf2midi.com/index.php?searchword=&filetype=sf2&typesearch=files&page=search)

Hammersound (http://www.hammersound.net/)

HomeMusician (http://www.homemusician.net/soundfonts.php)

SoundFonts.it (http://www.soundfonts.it/?a=soundfonts)

A.4. Sound effects

Sound-fishing (bruitages gratuits) [fr] (http://www.sound-fishing.net/bruitages.htm)

Appendix B. My work

This is the *real* thing, the one that made you working hard to set up you GNU/Linux box: Creating Music.

My own audio work is available on my Website (http://www.esaracco.fr/musique/), Dogmazic (http://www.dogmazic.net/Emmanuel_Saracco), Last.fm (http://www.lastfm.fr/music/Emmanuel+Saracco) or Jamendo (http://www.jamendo.com/artist/Emmanuel_Saracco).

> **Note:** Most of this material is intended to be executed by *real* instruments, so MIDI should sound a bit strange. In this case, better listening with headphones...

Colophon

This book is written in DocBook (http://www.docbook.org/) XML on GNU (http://www.gnu.org/)/Linux (http://www.kernel.org/) Debian (http://www.debian.org/) *sid* (amd64) system.

Outputs were produced with **db2html** and **db2pdf**.

Stylesheet for HTML output have been stolen and adapted from here (http://www.karakas-online.de/myLinuxTips/css-for-docbook.html).