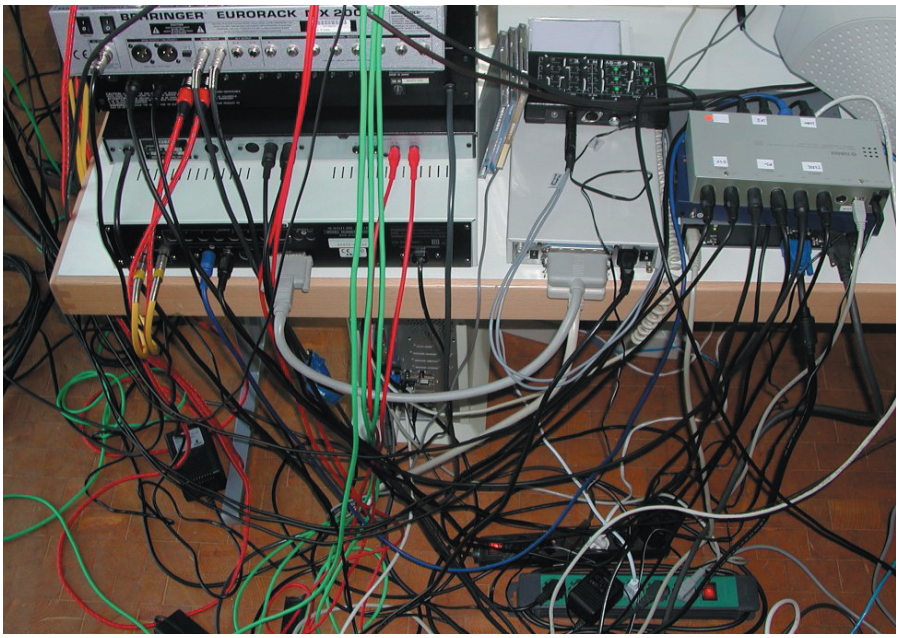
The MusE Audio/MIDI Sequencer

Professional Music Maker

Linux for musicians? No big deal, so far. But the Open Source sequencer, MusE, is poised to change that by handling two major tasks: audio and MIDI processing.

BY FRANK NEUMANN



Some of you might be wondering what a sequencer does. Well, a sequencer is a kind of composition tool that brings scores, pauses and other musical data to the PC, allowing you to replay a composition any time later. It does not make any difference whether you do a "live" recording or use your mouse to click in every single note. The results are typically a multi-track composition that can be played back at the press of a button.

The instruments that produce the actual sound can be external MIDI sources (see Box 1), sound files (in "WAV" format) or software instruments that calculate sounds or noises in real time and output them via a computer's soundcard. External sound sources do not use up system resources as the computer simply needs to send the right score to the source at the right time, but has very little else to do. It takes more than a little skill to assign scores to multiple tracks that all run and produce the desired sounds simultaneously.

Playing WAV files places a heavier load on a machine's resources – particularly on the memory. That does not mean that the CPU can take a break. Mixing multiple audio tracks to create a composite track, and applying effects to audio files in real time, that is while playing back a track, can also be expensive in terms of CPU cycles.

Software instruments (also known as *virtual instruments*) are the most modern variant, but also place the heaviest load on a system. The sequencer sees them as external MIDI devices, as they accept note commands. Internally, their job is to use mathematical formulae to calculate waveforms that the soundcard outputs. This has to be done as quickly and cleanly as possible, and of course it has to be in sync with the rest of the composition, and without any noticeable delay, just in case someone wants to play a virtual instrument during a live performance. This is referred as to minimizing latency.

Before you start

If you want to use MusE (no rhyme intended!), it makes sense to look for a distribution that includes the binary package. Compiling the source code (from [2]) is quite time-consuming (and incidentally, MuSE, the "*Mu*ltimedia *S*treaming *E*ngine", has nothing to do with the sequencer described in this article, despite the similar names).

Red Hat users should surf to PlanetCCRMA [3], where you will find both MusE and binary packages for numerous other audio programs. SuSE 8.2 includes the fairly ancient 0.6.0 version. Debian *unstable* (or *testing*) both include a current MusE package, as does the Mandrake developer version, Cooker.

But there are some requirements you need to fulfill before you can run the pro-

Box 1: What is MIDI?

The MIDI protocol ("*M*usical *I*nstrument *D*igital *I*nterface") was invented around 1982/83 and defines the transfer of musical data and accompanying information via a 31250 baud serial line between various MIDI-aware devices (keyboards, expanders, sequencers, and computers). Some typical MIDI commands are "NoteOn" (keypress), which produces a three byte bitstream (with the key number, channel, velocity), or "Program Change" (which changes a sound program for a specific channel). MIDI is very easy on resources in comparison to audio transmissions (176400 bytes per second in CD quality), but there is a downside, a legacy of the protocol.

More modern, and faster, protocols (such as mLAN by Yamaha) are being introduced slowly but surely. Having said that, sound devices still provide the typical MIDI In, Out, and Thru jacks. See [1] for more information.

Box 2: Clock stability and kernel patches

MusE needs *root* privileges at runtime, to maintain clock stability, although responsible admins may not be too thrilled with the idea. To assign these privileges, log on as or *su* to *root* and enter

```
chown root /usr/bin/muse
chmod 4755 /usr/bin/muse
```

(you may need to modify the path */usr/bin*), preferably on a standalone machine. Also, the kernel will need to support the "Real-Time Clock" (RTC), which most modern distributions do anyway.

You may want to extend the kernel by applying the *Low Latency* (LL) and *PreEmptive Scheduler* (PE) patches – despite the obvious advantages for MusE, make sure that you know what you are doing, as the steps are non-trivial. Some future 2.6.x kernel versions will have these patches by default.

gram. For example, MusE will refuse to run without

- the qt library version 3.1.0 or newer (version 3.1.1 recommended)
- the ALSA soundcard driver system ALSA, version 0.9.x (0.9.6 is current, see also Box 3) and
- the *libsndfile* library to import and export WAV files and other formats, version 1.0.1 or newer (1.0.4 is current).

The following are recommended (but not mandatory)

- a customized Linux kernel, to provide clock stability (see Box 2),
- the Fluidsynth software synthesizer, which uses soundfonts to emulate an external keyboard,
- external software instruments, like AlsaModularSynth [7] or ZynAdd SubFX [8],
- the "*L*inux *A*udio *D*eveloper's *S*imple *P*lugin *API*" LADSPA (Box 4) and
- JACK ("*J*ack *A*udio *C*onnection *K*it").

The latter creates a background process that exchanges and links audio data between multiple programs retaining sample precision and reducing latency. If you want MusE to send output to other programs, you definitely need JACK. JACK source code is available from [6]. The same site should provide packages for major distributions, or you can use *rpmseek.com* or *rpmfind.net* [4] to locate them – watch out for obsolete versions, in this case.

After installing MusE, the best way to launch the program, is to type *muse -R* in the command line. The *-R* option ensures that the program will be assigned higher priority if required, thus ensuring clock accuracy. *muse -h* shows other possible **flags**.

Typing the following:

```
muse -R name.med
```

loads a composition in MusE's own, XML-based file format. The program stores score data, window positions, the active software synthesizer, color settings and many other things in .med files. Although MusE can import and export standard MIDI files, it makes more sense to use the proprietary format and not risk any data integrity during conversions while creating a composition, before exporting the finished product to a MIDI file.

GLOSSARY

Flag: *An option that affects a command line, in the case of MusE for example we have -R or -h. Flags are traditionally introduced by a minus sign (or two), so you might say that the option was hanging on a flagpole.*

The Arranger

When you launch the program, the first window to appear is the main MusE window, aka the *Arranger*, accompanied by the Transport window (see Figure 1). The latter is a miniature mixing desk: the play, start and wind buttons, two position displays (time and bar), the bpm count, and the current values for the left and right marks, and some additional recording controls are all located here.

Your first task will be to select a synthesizer in the *Arranger* window. To do so select *Config / Soft synthesizer* (see Figure 2) and choose one of the available synthesizers. The left column shows the candidates and the right column the virtual instruments you have actually selected.

You should also take a look at the *Midi Port Table* below *Config / MIDI Ports* (see Figure 3). This window contains a list of MIDI ports, no matter whether they are physical five pin DIN ports on a soundcard, or virtual ports that exist only in computer memory. You need to assign a port to each software synthesizer you launch. To access the GUI, simply click on the small circle in the *GUI* column.

Finally, check that the right driver is selected in *Config / Audio System* (this defaults to *ALSA*), before saving your configuration in the *~/.MusE* file by selecting *Config / Save Configuration*.

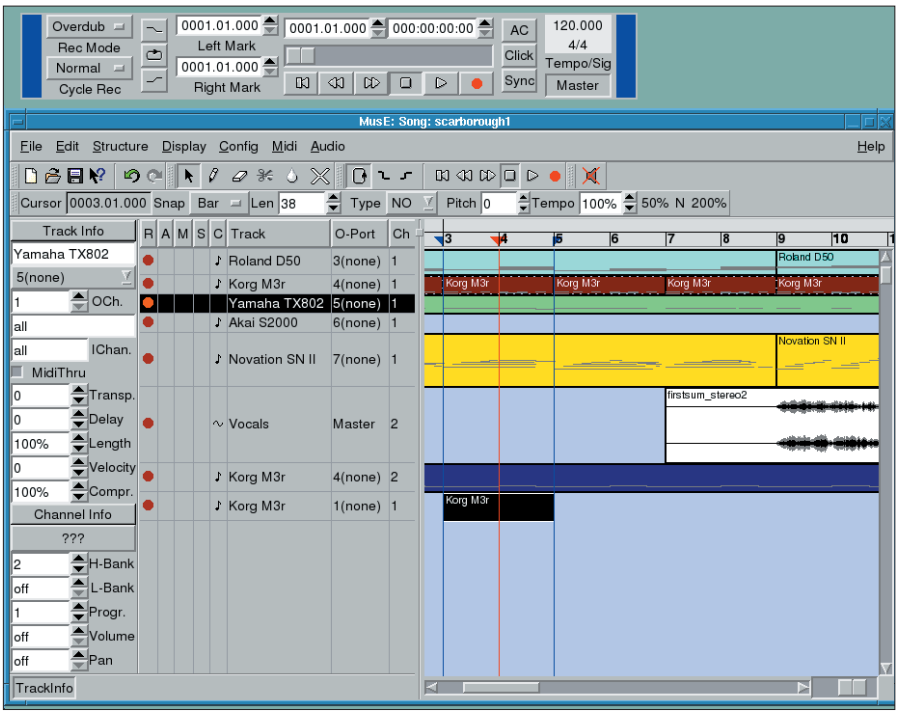


Figure 1: The Transport window and the Arranger

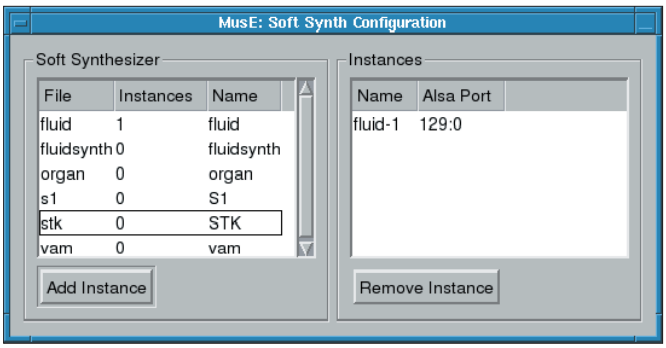


Figure 2: Enabling an external software synthesizer

The individual tracks take up a large part of the Arranger window. Of course there is not a lot to see at first, but you can doubleclick on an empty track to create a new one. The track name (*Track 1*) is editable. You can then assign the MIDI port where this track will output its data in the column to the right of the track name (*O-Port*).

The panel on the right contains the parts that belong to each track. These are scores of various lengths that can be modified or moved independently of each other. Vertically above the part display you will notice two blue lines and a single red line which are capped with small triangles. They indicate the left and right *Locators* as well as the current play position (red line).

To create a new part, hold the center mouse button down for the left locator, or the right button for the right locator, and drag the locator to a start or end position. Then double click the area between the locators. The part is originally assigned the same name as the track it belongs to.

After working with MusE for a while, you will probably start looking around for keyboard shortcuts to speed up common tasks such as playing or re-winding. Table 1 lists some of them. For a more complete overview, refer to the *README.shortcuts* file in the MusE source package – if you are using a precompiled package, it is to be hoped that this file has been installed for you.

Box 3: ALSA and its sequencer concept

When the kernel developers added ALSA, the "*A*dvanced *L*inux *S*ound *A*rchitecture" to the Linux source code for version 2.5.3, it became obvious that the modular soundcard driver system would be replacing the OSS ("*O*pen *S*ound *S*ystem") in the next user kernel generation, 2.6. Many distributions, such as SuSE and Debian, use ALSA by default, or you can customize the OS to do so.

ALSA has a lot of benefits to offer, such as support for modern (multichannel) soundcards, lower latency and its modular approach. The sequencer concept really pays off in the context of MusE. Programs log on as ALSA clients and state their capabilities: for example "This program can transmit MIDI, receive MIDI, or both". Other applications query ALSA's client list when they need to transfer bitstreams between programs.

External MIDI ports, or the virtual GUI keyboard *vkeybd* create MIDI messages. These messages are in turn received by external MIDI ports, software synthesizers such as AlsaModularSynth [7] and ZynAddSubFX [8], or the wavetable synthesizers provided some soundcards that can use soundfonts [11] (see Box 5).

Composing with the Pianola

When you double click a part, the so-called *PianoRoll Editor* appears (the editor derives its name from the fact that it looks just like the rolls of an old pianola). This is where you enter, delete and move notes (see Figure 4). The four tools used for this task are located in the menu bar at the top. The pointer tool, which selects individual notes (or multiple notes for a single modification step), the pencil, which is used to create new notes or change the length of existing notes, the rubber, which deletes notes, and finally the drawing tool ("Draw").

You can use all of these editing functions while replaying a track – this allows you to repeat short passages using the "Loop" function, at the same time adding corrections until you are happy with the results.

You can also change the velocity (that is, the force with which a key was pressed), by clicking on *Ctrl* at the bottom left. When you do so, MusE opens a panel in the lower part of the window (see Figure 4), where a vertical blue line per note represents the velocity of the note. You can use the pencil to change the velocity, or the draw tool to add lines that define the velocity for a multiple selection of notes.

You can use the multiple Undo/Redo function in the *Edit* menu to remove input errors, and the *Quantize* function takes care of incorrect notes (that occur during of live recordings, for example). On the subject of live recordings, to create a live recording, in the Arranger window click in the *R* column of the track that will store the recording. The red Record button in the Transport window starts recording, and clicking on the Play button launches parallel playback and record mode. Don't be surprised that the PianoRoll window does not immediately display the score, it will do so, as soon as you stop recording!

To the left of the Record bar *R*, the Arranger window contains four additional columns called *A*, *M*, *S*, and *C*. *A* refers to the activity in this track. While replaying the track a small black bar appears here. The length of the bar changes constantly to reflect the frequency of notes in the track. *M* stands for *M*ute, and allows you to cut out indi-

Table 1: Keyboard shortcuts

Key	Action
[Enter]	Play
[Space]	Stop (if pressed again, re-wind to left locator)
[C]	Toggle metronome on/off (only the MIDI metronome at present)
[Ctrl-Z]	Undo one step
[Ctrl-Y]	Redo one step
[/] (numerical block)	Toggle loop mode on/off

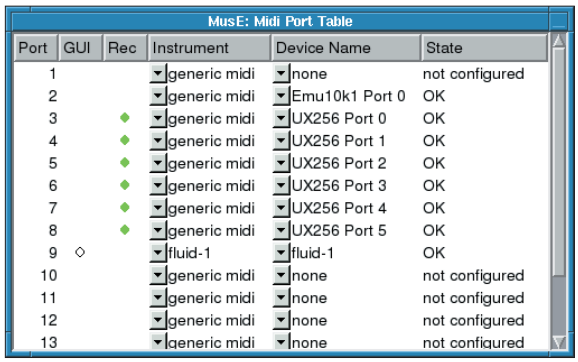


Figure 3: Make sure that you assign a MIDI port

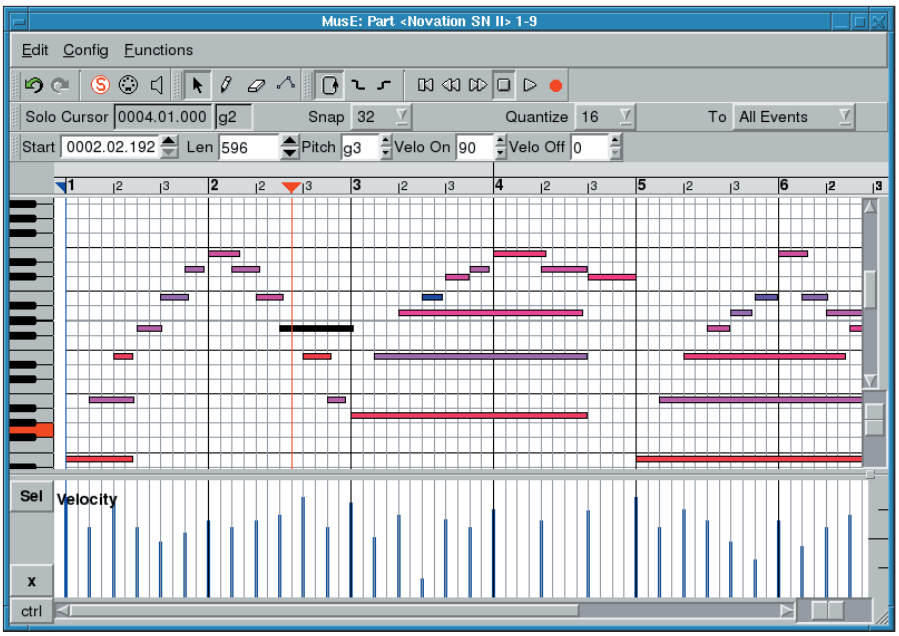


Figure 4: The PianoRoll Editor

vidual tracks when replaying a song so that you can concentrate on the remaining tracks. *S* stands for "Solo" and produces the opposite effect to "Mute". Click on this column to mute all other tracks apart from this one.

The radio button in the *C* column allows you to specify the ("Content") of a track. We have only looked at MIDI tracks so far. But this button allows you to convert a track to an audio track (assuming the track is empty). The *File / Import Wave* menu allows you to convert individual files to *.wav* format, displaying them in the selected audio track and allowing you to move them to the desired position.

Instead of a MIDI track, you could also specify a *Drum* track. When you edit the parts in this track, a different editor appears (see Figure 5). The editor is specially designed for editing drum tracks, as you can assign a name to each key on the keyboard (each key represents a different percussion instrument when you are editing a drum set). The good thing is, when you are working with the drum editor and enter a *Bass Drum* note, you really do get a bass drum and not a tambourine.

GLOSSARY

Panorama Position: *this specifies where a sound should appear to come from in a stereo image – from the left, the right, or the center.*

Mixers and Plugins

As you would expect of a sequencer that can handle audio tracks, MusE also provides an audio mixer. You can select *Audio / Mixer* to access the mixer (see Figure 6). For each audio track, a channel with volume and "**Panorama Position**" controls is displayed. In addition, you can add apply LADSPA effects.

To achieve this, right click on one of the *empty* fields. This should drop down a list of plugins (assuming you have installed them – see Box 4). Select a plugin. As some effects only change the sound very slightly, you can use the *Bypass* function to compare the effect with the original sound.

Select the *Show GUI* menu item to explore a plugin. Figure 7 shows the simple, but fully-functional GUI for a multiband equalizer plugin.

If needed, you can send single tracks directly to the "Master" output using the buttons at the bottom of the screen, or you can assign them to groups. A group is useful if you are working with multiple audio tracks and need to apply the same LADSPA effect to all of them. You could add the effect individually to each channel, but that would be expensive in terms of CPU cycles. Instead, you can tell MusE to combine these tracks to create a subgroup, apply a single instance of the effect, and send the group off to the master. Just like other tracks in the Arranger,

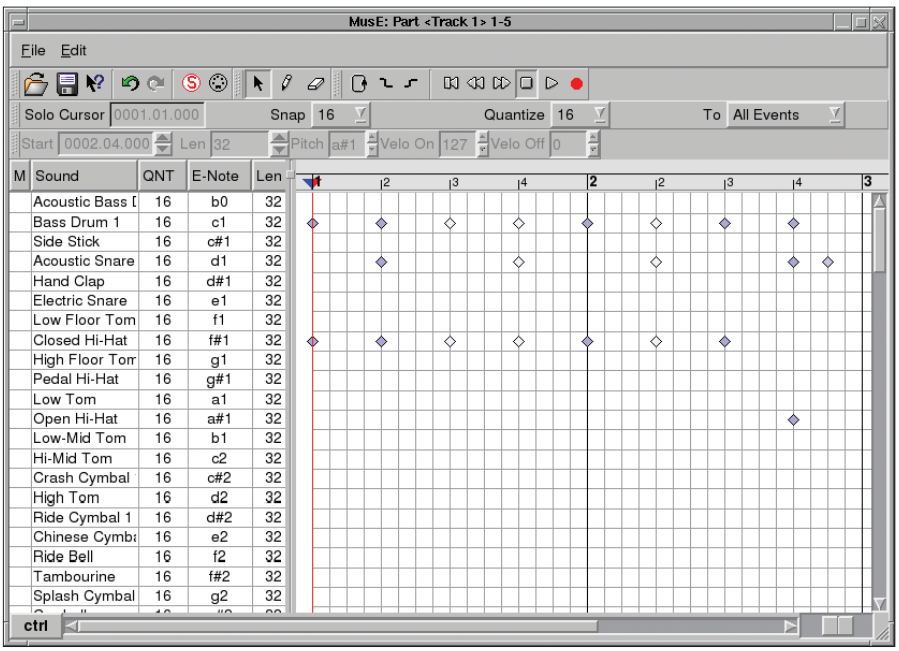


Figure 5: The Drum Editor

Box 4: LADSPA Plugins

Programs like MusE (referred to as "hosts") can load external programming code (so-called "shared objects"), to provide sound effects. The range of effects includes echo, reverb, amplification, compressors, limiters, in fact anything a musician or sound technician could possibly need to provide more body, brilliance, or depth to his or her work – or just make it sound weird. The biggest and most interesting collection of LADSPA plugins [9] is available from [10].

Figure 6: The Audio Mixer

you can mute individual channels (*M*) or switch to solo mode (*S*), or even toggle them on and off completely.

The output from your soundcard is defined by the signal that is sent to the "Master" channel, which has its own volume control. You can use the "*Record Downmix*" function to send this output to a WAV file (look for the red button below the master channel). Of course this only applies to sound sources whose output is actually processed by MusE.

And this discounts external MIDI sound devices, and software synthesizers running on your machine, independently of MusE. However, you can use JACK [6] is to capture these sources. JACK allows you to transfer sound data between various music or sound programs.

Strengths and Weaknesses

Let's be honest about this – at the time of writing MusE cannot hold its own against a Windows software like Steinberg's *Cubase*. But can you expect that of a project powered mainly by a single developer, *Werner Schweer*? Having said that, since Werner released the very first MusE version in January 2000 he has created a wide range of functions that some commercial applications took years to implement.

The upside is MusE' impressive functional range, particularly with respect to MIDI handling, its intuitive user interface that allows you to quickly get on with the job, plus the fact that the sequencer can work both with external and internal sound sources.

The downside is that the program was prone to crash at times, particularly when processing audio files. Some functions, like the *Random Rhythm Generator*, for example, have not been fully implemented, or have not been maintained for a while, like the score editor (the author has moved this to a project of its own, *MScore* [12]). Unfortunately, the complexity of actually compiling the source code is more likely to frighten the majority of potential users away. There are simply too many requirements to fulfill. It makes sense to look around for a precompiled package.

The slightly unusual configuration might also put one or two people off. But the worst thing is the fact that the lack of documentation (at present). The best place to look is the mailing list, which you can access via the homepage.

All of these disadvantages could be resolved by more intensive testing and more bug reports from users, on the one hand, and more co-developers willing to work on the project – and if this sounds like a call for help, it is! And if anyone is interested in my opinion: MusE is well on its way to becoming the killer application for MIDI and sound that GIMP

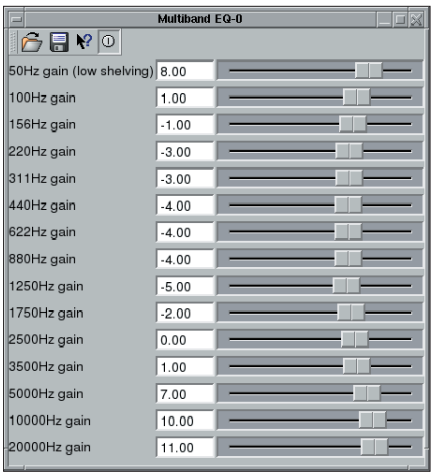


Figure 7: The Multiband Equalizer

Box 5: Soundfonts

A soundfont is a single binary file that contains a range of sounds (for example samples of a piano, a flute, or a percussion instrument), and maps these sounds to a range of keys. Compared with creating computer-generated sounds, playing soundfonts does not have anything like the impact on a computer's performance.

Soundfont files are typically recognizable by the *.sf2* suffix. There are two ways of playing back a soundfont: either the soundcard itself (such as the SoundBlaster AWE64 and Live cards by Creative Labs) is capable of processing soundfonts (this is referred to as *WaveTable synthesis*), or a software application takes over this task.

Fluidsynth [5] is a stable software package for soundfont use. The synthesizer is launched as follows

```
fluidsynth -m alsa_seq
SoundFont.sf2
```

It runs as a standalone ALSA client, in order to be visible and accessible to MusE.
MusE provides a more simple, and older embedded version of Fluidsynth, the advantage being that the sound created by virtual instruments, can be routed through MusE' audio mixer, allowing the composer to add effects before outputting the results on the soundcard.
Soundfonts are available from [11], for example. You might even find some on the Windows driver CD supplied with your soundcard.

INFO

[1] Introduction to MIDI: *http://www.harmony-central.com/MIDI/Doc/doc.html*

[2] MusE homepage: *http://muse.seh.de/*

[3] PlanetCCRMA: *http://www-ccrma.stanford.edu/planetccrma/software/*

[4] RPMFind: *http://rpmfind.net*

[5] Fluidsynth: *http://www.fluidsynth.org/*

[6] JACK Audio Connection Kit: *http://jackit.sourceforge.net/*

[7] AlsaModularSynth: *http://alsamodular.sourceforge.net/*

[8] ZynAddSubFX: *http://zynaddsubfx.sourceforge.net/*

[9] LADSPA: *http://www.ladspa.org/*

[10] swh plugins: *http://plugin.org.uk/*

[11] Soundfonts: *http://www.hammersound.net/*

[12] MScore: *http://muse.seh.de/mscore/*

THE AUTHOR

Frank Neumann had been working with Linux for quite a while (first on the Amiga and later on PCs), before he discovered how well Linux supports his other old hobby, music, about a year ago.