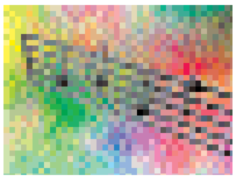
Out of the box SOUND RESEARCH



BY CHRISTIAN PERLE

There are thousands upon thousands of tools and utilities for Linux. This is great, but it makes finding the real gems rather difficult. To save you time and effort, »Out of the box« does the job for you, each month highlighting a particularly interesting or useful program that you might otherwise have overlooked. This month we'd like to introduce you to the SpiralSynth synthesiser.

When Kraftwerk – one of the pioneers of electronic music – was formed, a synthesiser was still a huge and clumsy analog device overloaded with knobs and slide controls and costing tens of thousands of pounds. Synthesisers are much more compact and affordable these days, of course, and with the computing power available today it is now even possible to simulate such devices on a relatively basic PC. Indeed, this is exactly what *SpiralSynth*, developed in the UK by Dave Griffiths, does.

Prerequisites

A graphical user interface of some form is required in order to give the *SpiralSynth* user the ability to change its settings and configure its options, as well as to provide a visual display of the audio signal created. *SpiralSynth* makes use of the **FLTK Library** for this purpose, so before you do anything else you'll have to obtain and install this. We got our FLTK (Version 1.0.9 or higher is required) from *http://www.fltk.org/* and *SpiralSynth* itself from *http://www.blueammonite.f9.co.uk/SpiralSynth/*. Be sure to download version 0.1.5 rather than a more current version such as 0.1.6, however. There is no functional difference between these two versions, but they do use different file formats. You'll discover the importance of this a little later on.

Library: *A file containing a collection of useful C-functions for specific purposes. Examples include libm, which provides mathematical functions, and libXt, which contains functions for programming the X11 window system. Libraries are often shared by several programs.*

FLTK: *The »Fast Light ToolKit« (pronounced: »Fulltick«) is a very compact library for easy programming of the X11window system.*

Compiling: *In its source text form, a program is not usually executable by the operating system. It is only by compiling (converting) this source code that it can be turned into something that can be executed by a PC's processor.*

RPM: *With the »Red Hat Package Manager« software packages can be quickly and easily installed or removed.*

Installation

Once you have the two necessary components on your hard disk, it's time to **compile**. FLTK needs to be compiled and installed first, the latter step requiring *root* privileges:

```
tar xzf fltk-1.0.9-source.tar.gz
cd fltk-1.0.9
./configure
make
su  (enter root-password)
make install ; exit
```

Anyone who wants to avoid compiling FLTK can install the **rpm** package version. To do this you need two files which can be found at *ftp://rpmfind.net/linux/Mandrakedevel/7.2beta/i586/Mandrake/RPMS/fltk-1.0.9-2mdk.i586.rpm* and *ftp://rpmfind.net/linux/Mandrakedevel/7.2beta/i586/Mandrake/RPMS/fltk-devel-1.0.9-2mdk.i586.rpm*.

The installation of the *rpm* packages should be done as follows:

```
su  (enter root-password)
rpm -Uvh fltk-1.0.9-2mdk.i586.rpm
rpm -Uvh fltk-devel-1.0.9-2mdk.i586.rpm
exit
```

Now it's the turn of the actual *SpiralSynth* program itself:

```
tar xzf SpiralSynth-0.1.5.tar.gz
cd SpiralSynth-0.1.5
```

```
./configure
make
su  ( enter root-password)
make install ; exit
```

If errors arise when compiling *SpiralSynth* you can try installing a pre-compiled version. The program's author recommends *http://www.blueammonite.f9.co.uk/SpiralSynth/dload/SpiralSynth-i386Linux-0.1.5.gz*. There is actually very little you need to do in order to install this version:

```
gunzip SpiralSynth-i386Linux-0.1.5.gz
chmod 755 SpiralSynth-i386Linux-0.1.5
su  (enter root password)
cp SpiralSynth-i386Linux-0.1.5 ↩
/usr/local/bin/SpiralSynth
exit
```

Sound off!

So much installation work should be rewarded. From the terminal emulator of your choice (*xterm*, *kvt* or *Gnome-Terminal*, for example) start the program with the command *SpiralSynth &*, at which point a window similar to that shown in Figure 1 should appear.

The window is split into three main areas. On the left are the oscillators, on the right the mixers and effects devices and at the bottom the knobs with which stored settings (*Patches*) can be retrieved. There is also a graphical display of the audio signal (*Scope*).

Each of the three basic oscillators has the same setting options: wave form (square wave, sawtooth or noise), pulse width (*PW*), noise generator setting (*SH*), **Portamento** (*PM*) and controllers to tune and adjust the modulation depth.
On top of everything else, an *Envelope* can be adjusted for each oscillator. This determines the attack and fade-out behaviour of the signal.
Using the two mixers, signals from the oscillators are linked together. As with all other settings in *SpiralSynth* the word here is Experiment! Nothing bad will happen.
Finally, you can give a tone a »finishing touch« with the *LFG* (»Low Frequency Generator«), the low-pass filter and the delay effect *Delay*).
Anyone who finds this is all too much trouble can press the *Rand* button in the *Patch Bank* area, which sets all the controls randomly.
The actual triggering of the sounds occurs via the keyboard, where the rows of keys *y* to *m* and *q* to *p* are assigned as »white keys«, and the rows *a* to *j* and *2* to *0* as »black keys«.
Once you have created an interesting sound, you can save it by clicking on the *Save* button and then selecting one of the blue shaded buttons in the *Patch Bank*. In the *Output* field, the sound produced can be saved in a **WAV** file by clicking on *Record*. Brilliant!

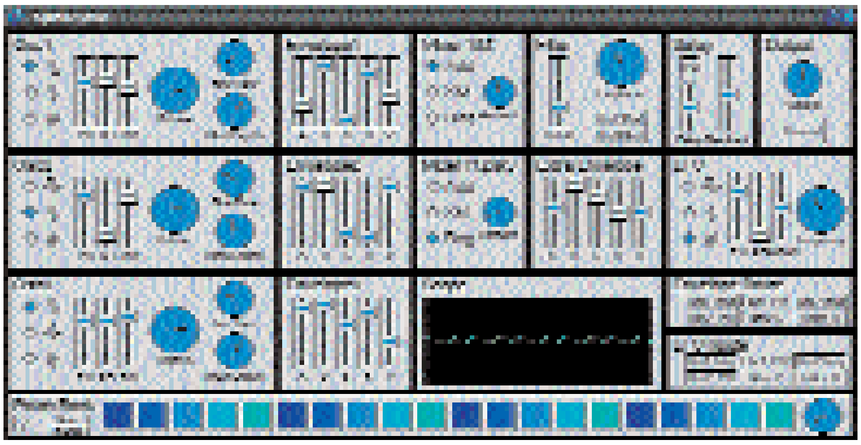


Fig. 1: SpiralSynth main window

Tuning

SpiralSynth will create two hidden files in your home directory, in which the basic settings (*.Spiralrc*) and the stored patches (*.SpiralPatches.bank*) are located. Let's just take a closer look at the first file. In order to do this you'll first have to shut *SpiralSynth* down. Having done so, fire up your favourite text editor and point it at *.Spiralrc*.

Not everyone has a **MIDI** keyboard. The *WantMidi* entry can be set to zero if this applies to you. The *KeyMap* entry contains a listing of the keyboard keys *SpiralSynth* uses, and can be changed if required – essential if you aren't using a standard UK or US keyboard. Listing 1 shows an example *.Spiralrc* file with MIDI disabled.

Listing 1:.Spiralrc with MIDI disabled.

```
SpiralSynth resource file
BufferSize      = 512
Samplerate      = 44100
WantMidi        = 0
FilterGranularity = 50
Output          = /dev/dsp
Midi            = /dev/midi
WantRealtimeOut   = 1
KeyMap          = ↩
zsxdcvgbhnjmq2w3er5t6z7ui9o0p[
```

Sound samples

If, despite the very useful random function, you are still unable to produce any interesting sounds from *SpiralSynth*, all is not lost. Our coverdisc this month contains a pre-set patch bank called *mypatches.bank*, which you can rename as *.SpiralPatches.bank* and copy into your home directory. This file can also be found on the Web at *http://home.tu-clausthal.de/~incp/mypatches.bank*.

There shouldn't be anything standing between you and some exciting synthesiser sound research, unless you have installed Version 0.1.6 (against our advice). This uses a binary format for the patches file that is incompatible with the format used in version 0.1.5, which is what we used to create the coverdisc file. ■

Portamento: *With this effect the pitch selected is not attained immediately, but is »drawn out« from the previous sound.*
WAV: *A common and usually uncompressed audio format, first implemented in Windows.*
MIDI: *»Musical Instruments Digital Interface«, a standard for controlling electronic musical instruments.*
■