Musical
Carrion

Open Source recording studio

Matthew Geddes

June 2006

abstract:

This paper documents the tools and processes used in a particular recording project. It consists of an overview of the project, as well as detailed technical descriptions of many of the stages of the project.

Web http://www.musicalcarrion.com/
E-mail meat@musicalcarrion.com

Contents

List of Figures

1 Introduction

1.1 Overview

This paper is meant as an overview of a specific recording studio project undertaken this year. It is not supposed to be a recipe to be strictly adhered to, but rather a set of tools and techniques that worked in this specific case. The aim is to provide the following:

- Evidence that it is possible to create decent works using the Linux audio subsystem
- Ideas to be adapted to your specific flavour of recording studio, or
- A list of what not to do in your studio (your choice)
- Enough of a teaser to get you all hooked

1.2 Scope

This paper attempts to avoid suggesting that a particular collection of hardware and/or software is the only way to achieve Musical Nirvana. Instead, it introduces a specific project undertaken and describes the process used to get the end result. It does use specific applications to demonstrate these techniques, but readily admits that other software exists and may even be a better choice in the individual case.

Now that we've covered the crap, on with the content.....

2 The project – The Upside Down

2.1 Overview

The Upside Down is, in their own words:

> The Upside Down are Gini Herrick(vocals) and Elle Taylor(guitar). Finding themselves in Adelaide after a trip around Australia in their beloved Kombi van, the girls decided there was no option but to start a band...
>
> They have recently formed as The Upside Down, drawing from varied influences ranging from Elliot Smith to Janis Joplin. Their music is best described as falling somewhere amongst the genres of blues, rock and alternative.
>
> You can catch them most Tuesdays rocking the Higher Ground, Rundle St., Adelaide.[a]
>
> [a] taken from `http://upside-down.musicalcarrion.com`

They did attempt to record some of their work previously, but had some issues with the results and were keen to make improvements.

2.2 Planning

As usual, forethought and planning was an essential step. Most of the cabling and hardware setup was done well before the dynamic duo arrived. Once drinks had been poured, we set up seating and instruments – looking primarily for comfort. The next few steps were important and were executed at the same time:

1. The band ran through part of each track as a warm-up
2. For each track, we decided how it would be recorded based on the feel during the warm-up. More on this later.
3. Rough levels were set

From this, we created a playlist of tracks and a plan of attack.

2.3 Tracks

2.3.1 Wishing Stone

Wishing Stone is a very bluesy track. It seemed very well suited to doing a live stereo recording of it being jammed out in the studio, rather than giving it a more sterile multi-track sound.

Two microphones were positioned to capture us sitting in a circle playing. They were placed quite far from all sound sources to give that *just jammin' in a room* feel.

The following instruments were used:

- Vocals - *Virginia*
- Rhythm guitar on wood-bodied resonator guitar - *Ellen*
- Second guitar / lead slide guitar - *Me*

The track itself has an interesting set of dynamics ranging between tight rhythmic playing, to the sloppy instrumental sections. For a laugh, the LADSPA LoFi plugin was used to add the crackle, pop and distortion.

2.3.2 Fuselage

Fuselage is another original by The Upside Down. It was better suited to a multi-track recording as opposed to the live technique used in the previous track. A few takes of each instrument were made. To thicken the sound a little, Ellen recorded half of the first guitar track through a different pickup/amp combination.

The following instruments were used:

- Vocals - *Virginia*
- Guitars (Fender Squier Telecaster) - *Ellen*

Due to time restrictions, it wasn't possible to create and sequence a percussion part or add a bass/second-guitar part.

2.3.3 Behind These Walls

Very similar to the previous track. Multi-track was used instead of the live recording. A single guitar part was used this time. Given that multi-track was used, it is possible to add other parts to the track later and remaster it.

The following instruments were used:

- Vocals - *Virginia*
- Guitar (Fender Squier Telecaster) - *Ellen*

3 Studio setup

3.1 Hardware and signal flow

Figure 1 shows the signal flow of devices in the studio. A description follows:

Figure 1: Signal flow

1. Noise source (guitar, vocalist, cat, etc)
2. Mixer input to control levels

3. Mixer FX inserts into FX host
4. Real-time FX processing and dumped into other mixer inputs
5. Levels and EQ adjusted
6. Stereo signal into record host
7. Signal back into mixer and out to phones and monitors

3.1.1 Real-time effects processing

Real-time effects processing is handled by a separate host. Given that this host (an Apple PowerBook G4 running Mac OS X) has a single stereo input and single stereo output, only two mono streams can be processed at a time. Luckily this is enough for many situations.

Splitting a stereo stream into two mono streams, processing them using separate sets of effects plugins and mixing them back into a stereo stream is not as simple as it could be, but can be achieved and replicated. Figure 2 shows this concept in a simplified form.

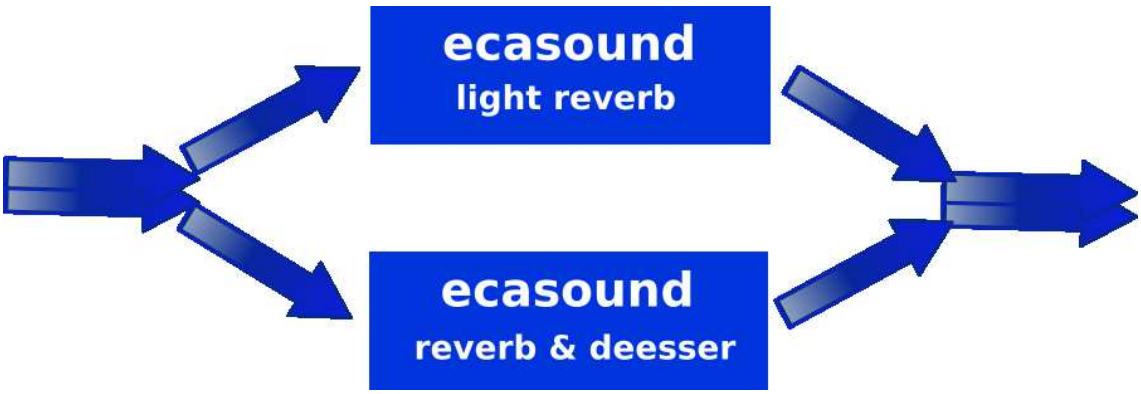

Figure 2: Real-time effects split

3.2 Software used in the project

3.2.1 Multitrack recording

There are a number of multitrack recorders available. The most popular seems to be Ardour[1]. Any multitrack recorder could have been used, but Ecasound[2] was selected. It sports a command-line interface, which lends itself very well to automation and replication.

JACK[3] was used to handle audio I/O.

3.2.2 Effects processing

Effects processing could have been done using any pseudo real-time LADSPA host. Ecasound was again selected for the reasons given previously. JACK was used again too.

[1] `http://ardour.org/`
[2] `http://www.eca.cx/`
[3] `http://jack-it.sf.net/`

3.2.3 Mastering

Once all takes and tracks had been recorded, masters were mixed and tweaked using Ecasound once again. Audacity[4] was also used for the occasional touch-up job on small sections of some parts.

3.2.4 Drivers / Kernels

A late 2.6 kernel was obtained from `kernel.org` and patched with the preemptible kernel patch and the ALSA sound system was used. JACK was also patched to support big-endian 24-bit samples, as supported by some of the hardware used.

The Mac OS X operating system and accompanying drivers/kernel was used on the real-time effects processing host.

3.3 Software used, but not for this project

3.3.1 Hydrogen

A highly capable drum sequencer, Hydrogen[5] is used whereever drums are needs and a drummer cannot be found.

3.3.2 Lilypond

Lilypond is used both for music notation and MIDI sequencing. The latter is still missing a few features, but it does the job.

3.3.3 Timidity++

Timidity++ is used for converting MIDI tracks into PCM audio using SoundFonts.

3.4 Notes on software

This should not be taken as an exhaustive list of audio software available under Linux (and similar) operating systems. There are many other applications available and may be just as suitable or even better in other circumstances. These applications were selected because they were the right tool for the specific circumstances presented.

4 Software specifics

4.1 Overview

This section aims to describe in detail how the software was used for the project.

[4] `http://audacity.sf.net/`
[5] `http://hydrogen.sf.net/`

4.2 Starting JACK

4.2.1 Effects processing host

The host used for effects processing had JACK (actually, JACK OS X) started with two input and output channels (using CoreAudio) with the a sample rate of 44100kHz. The actual command used to start `jackd` is shown in Figure 3.

```
/usr/local/bin/jackd \
    -R \
    -d coreaudio \
    -p 64 \
    -r 44100 \
    -i 2 \
    -o 2 \
    -n AppleDBDMAAudioDMAEngine:0
```

Figure 3: Command used to start jackd on Mac OS X

4.2.2 Recording host

The host used to record and master audio tracks had JACK started in a similar fashion, and also used `hdparm` to set disk parameters before starting JACK. Figure 4 shows the parameters passed to `hdparm` and to `jackd`. The `hdparm` parameters suit the host they're used on, but you should check them against your hardware before using them.

```
hdparm -X66 -d1 -u1 -m16 -c3 /dev/hda

/opt/jack-bswap/bin/jackd -R -d alsa -C hw:1,1 -P hw:1,0 -p 64 -n 2
```

Figure 4: Command used to start jackd on Linux

4.3 Using ecasound

4.3.1 Ecasound overview and concepts

Ecasound is a command-line multitrack recorder and audio processing tool. It can act as a JACK client and a LADSPA host and has been written with performance in mind. Central to ecasound is the concept of chains. A chain can be thought of a performing a similar function to a patch lead in an analogue studio – audio travels through it from one side to another and is used to route audio to and from specific components.

4.3.2 Recording a track

To record a single track or take, we need a chain that is connected to the audio input (from JACK) and to a file on the filesystem. We probably also want to turn off hardware monitoring[6]

[6]The feature some sound cards possess that routes all input directly to its output. It is lower latency than using software to monitor audio, but it does allow you to hear anythin your audio applications may do to colour

and create another chain that takes audio in and sends it back out to JACK. To summarise:

Chain	Input	Output
1	JACK	PCM Wave file
2	JACK	JACK

The syntax for the is shown in Figure 4.3.2 and can be broken down thusly:

```
ecasound -c -b:64 -r \
    -a:1,2  -i jack_auto \
    -a:1    -o bass-take1.wav \
    -a:2    -o jack_auto
```

Figure 5: Recording a single stereo track using ecasound

1. `-c` - start in interactive mode, rather than record straight away
2. `-b:64` - set the buffer size to 64 samples. A larger number of buffers will introduce more latency, but prevent dropouts.
3. `-r` - raise the priority of ecasound with the kernel. Often on by default
4. `-a:1,2 -i jack_auto` - input for both chains 1 and 2 come from JACK
5. `-a:1 -o bass-take1.wav` - output for chain 1 goes to a file called `bass-take1.wav`
6. `-a:2 -o jack_auto` - output for chain 2 goes to JACK

Once ecasound has initialised, it will prompt for a set of commands. `t` to start recording, `s` to stop recording. The manual contains many other commands, such as `setpos`[7].

4.3.3 Adding another track

Once the first track is recorded, we often want to add other tracks. We also want to keep these separate from each other to make it easier to mix-and-match different takes and to post-process individual tracks. The chains used might look as follows:

Chain	Input	Output
1	JACK	PCM Wave file
2	JACK	JACK
3	PCM Wave file (first track)	JACK

The command-line options used are quite similar to the previous example. Figure 4.3.3 shows an example command.

The only difference from the previous example is that we have another chain (number 3) to read from the first track and write to JACK. We also gave a unique filename to the output of chain 1. The end result is you will hear all tracks playing as the second is recorded.

the sound

[7] `setpos 0` is great when you screw up a take and want to start from the top without keeping the broken take.

```
ecasound -c -b:64 -r \
    -a:1,2  -i jack_auto \
    -a:3    -i bass-take1.wav \
    -a:1    -o guitar-take1.wav \
    -a:2,3  -o jack_auto
```

Figure 6: Overdubbing a second track

4.3.4 A third track - spot the pattern

Adding subsequent tracks is a case of building upon what we already have, as Figure 4.3.4 confirms.

```
ecasound -c -b:64 -r \
    -a:1,2        -i jack_auto \
    -a:3        -i bass-take1.wav \
    -a:4        -i guitar-take1.wav \
    -a:1        -o vocal-take1.wav \
    -a:2,3,4        -o jack_auto
```

Figure 7: Adding a third track

At this point, we have each of three tracks stored in their own separate audio files, which may be manipulated separately and then mixed together as a single stereo master. Speaking of which....

4.3.5 Mixing tracks together

Creating a basic mix is quite straightforward. It's a case of setting up a chain for each input track and sending all of their outputs to JACK (if we want to listen), or an audio file (if we want to save a copy to disk).

```
ecasound -c -b:64 -r \
    -a:1        -i bass-take1.wav \
    -a:2        -i guitar-take1.wav \
    -a:3        -i vocal-take1.wav \
    -a:all        -o jack_auto

or

ecasound -c -b:64 -r \
    -a:1        -i bass-take1.wav \
    -a:2        -i guitar-take1.wav \
    -a:3        -i vocal-take1.wav \
    -a:all        -o rough-mix.wav
```

Figure 8: Mixing all tracks

4.3.6 Post-processing individual tracks

It's not often that we record a take of a particular track perfectly in the perfect aural environment. Most tracks will require some kind of touching up or post-processing. Ecasound has a few efffects built in, but can also act as a host for LADSPA plugins. Most multi-track recorders can do the same. The example in Figure uses the built-in compressor and a LADSPA reverb effect to add a little colour and stability to a vocal track.

```
ecasound -i vocal-take1.wav \
    -eca:30,0.2,0.50,0.50 \
    -eli:1123,0,0.5,0.2,0.1,0.5,0.5 \
    -o vocal-take1-r1.wav
```

Figure 9: Post-processing a single take

This can be broken down thusly:

- `-i vocal-take1.wav` - the existing track to use as input. Note that we only have a single chain, hence not specifically adding one using the `-a` option used previouly.
- `-eca:30,0.2,0.50,0.50` - use one of the built-in compressors[8].
- `-eli:1123,0,0.5,0.2,0.1,0.5,0.5` - use LADSPA effect 1123 (Freeverb) reverb plugin.
- `-o vocal-take1-r1.wav` - the new file to write the processed output to

4.3.7 Creating a final mix

Using concepts already covered, we can create a stereo master mix that has had more fine-tuning than the crude mix we created before. Details such as panning and levels relative to other tracks can make all the difference. Figure 4.3.7 shows an example mixing three tracks to a single PCM file.

```
ecasound \
    -a:1 -ea:90  -epp:50 -i vocal-take1r1.wav \
    -a:2 -ea:120 -epp:65 -i guitar-take1.wav \
    -a:3 -ea:120 -epp:35 -i guitar2-take2.wav \
    -a:all -o mixr1.wav
```

Figure 10: Creating a more complex mix

The above command may be broken down thusly:

- `-a:1 -ea:90 -epp:50 -i vocal-take1r1.wav` - create a chain with vocal track as input, de-amplified by 10% and panned dead-centre.

[8]It appears as though this is a fairly-heavy handed use of the effect, but there was a lot of headroom above the signal peaks

- `-a:2 -ea:120 -epp:65 -i guitar-take1.wav` - main guitar track, amplified and panned slightly to the right.
- `-a:3 -ea:120 -epp:35 -i guitar2-take2.wav` - second guitar track, amplified and panned slightly to the left.

Notice that we didn't set the buffer size this time, nor did we care about controlling the starting and stopping of recording in interactive mode. There are no realtime sources, so latency doesn't matter – we're more interested in getting the job done.

4.3.8 Post-processing the final mix

Once we have the levels and panning right, we can perform any final touching up the track may need. This may involve adding a little reverb, removing the peaks using a little compression and then normalising the track, or it may consist of using effects such as the LADSPA LoFi plugin to destroy the track. These tasks are performed in the same way that we processed individual tracks in section 4.3.6.

4.3.9 Recording a software source

JACK can be used to route audio between applications. A good example of this might be recording a drum track created using the Hydrogen drum machine. Whilst we didn't use this technique in this project, it is probably still worth mentioning. The following steps may be used to record a software source:

1. Configure the application (Hydrogen) to use JACK for its output
2. Have ecasound record from JACK, such as shown in Figure 4.3.9. The `-G` option tells ecasound to wait for the start signal sent through JACK when the play button is hit in Hydrogen.
3. Hit play in Hydrogen[9].

```
ecasound -i jack_auto \
    -o drum_track.wav \
    -G jack,ecasound,recv
```

Figure 11: Recording a software source through JACK

Because a software source is unable to react to subtle timing changes, it is wise to record software sources first and then overdub real-world tracks.

[9]It can take a second or so for ecasound to initialise once receiving the start command through JACK. I like to leave a couple of seconds of silence at the start of drum tracks to ensure that nothing is missed

4.3.10 Splitting streams for real-time effects processing

We discussed earlier the idea of taking two mono streams from the mixing desk and routing them through a stereo port on a host to perform real-time effects processing. This involves a more complex set of ecasound chains. Figure 4.3.10 shows a set of possible commands, including the commands necessary to map JACK ports to ecasound chains. Figure 2 shows the concept.

```
ecasound -c -b:64 \
    -a:1,2 -i jack_auto \
    -a:1 -f:16,1,44100 -erc:1,1 \
    -eli:2147,1,4500,0,0 \
    -o jack_generic,right \
    -a:2 -f:16,1,44100 -erc:2,1 \
    -eli:1123,0,0.5,0.2,0.1,0.5,0.5 \
    -o jack_generic,left

jack_connect ecasound:left_2 coreaudio:Built-in\ Audio:in2 &&
jack_connect ecasound:right_1 coreaudio:Built-in\ Audio:in1
```

Figure 12: Processing two channels of a stereo input separately

The commands in figure 4.3.10 may be broken down thusly:

1. `-c -b:64` - we're back to real-time processing, so we want interactive mode to stop/start processing and we want our buffer size to be as small as possible.
2. `-a:1,2 -i jack_auto` - both chains (1 and 2) get their input from JACK
3. `-a:1 -f:16,1,44100 -erc:1,1` - set chain 1 format to CD format, but mono and route channel 1 of our input to this chain
4. `-eli:2147...` - use LADSPA plugin 2147 to process this stream
5. `-o jack_generic,right` - send output to a specific JACK port
6. `-a:2 -f:16,1,44100 -erc:2,1` - set chain 2 format to CD format, but mono and route channel 2 of our input to this chain
7. `-eli:1123...` - use LADSPA plugin 1123 to process this stream
8. `-o jack_generic,left` - send output to a specific JACK port
9. The `jack_connect` commands just attach ecasound ports to hardware ports.

5 Closing notes

5.1 Reproducing genius

If you're using a command-line tool such as ecasound, it is a wise idea to use something like a Makefile or shell script to keep track of the individual command-line options used for each stage of the process. It'll save you a great deal of time when attempting to record a new take or track, as well being easier to retrace your steps.