AlmusVCU User Manual

version 0.7

25th August 2002

Preamble

This is the user manual for AlmusVCU v0.7, which is a work in progress. This manual itself is also a work in progress, which means that some sections may be a bit fuzzy, and others may be missing.

The current state of the software is very unstable. This release is from an early development stage, and not all functionality has been tested (and is therefore probably broken). The software may work for you or it may not. Use it at your own risk.

In this manual, ASCII art is currently used for the pictures of the text-based user interface. A figure can for example look like this:

```
+--------------------+
|                    |
|                    |
|  <ok >0.1  ^+ v-   |
|v Pick number [3.4] |
+--------------------+
```

The characters `<`, `>`, `^` and `v` are supposed to look like arrows pointing left, right, up and down respectively. The surrounding `+---+` and `|` is simply meant to look like a rectangle which defines the size of the display.

Anders Torger, August 2002
torger@ludd.luth.se

Contents

Chapter 1

Introduction

1.1 AlmusVCU files

This manual assumes that the following components are available:

- `almusvcu-0.7.tar.gz`, the source code for AlmusVCU.
- `brutefir-0.99a.tar.gz`, the source code for the convolution engine BruteFIR.
- `almusvcu-filter-programs-0.5.tar.gz`, standard filter programs for AlmusVCU. A filter program is a generic name for the three types reverb, crosstalk cancellation and equalisation programs which AlmusVCU supports.
- `almusvcu-cd-0.7.iso`, ISO image of the AlmusVCU bootable CD. Only needed if the bootable CD is to be used.

The latest versions of the files and this manual can be found at:

`www.ludd.luth.se/~torger/almusvcu.html`

1.2 What is AlmusVCU?

AlmusVCU is an open-source software which makes a computer equipped with a multi-channel sound card into a real-time versatile convolver unit. The main area of use is as a surround sound processor in multi-channel high fidelity systems such as:

- Ambiophonics
- Ambisonics
- Auralisation/binaural/Ambisonics hybrid systems

The software aims at being useful for music lovers interested in the superior Ambiophonics and Ambisonics surround formats. However, rather than being tailored for a few specific uses, AlmusVCU is feature rich and very flexible, and also tries to

be a valuable tool for audio hobbyists, students and researchers, and even recording engineers. AlmusVCU can be configured to collaborate with other audio processors, be used as a reverb processor in recording post-production, or simply be used as a generic convolver unit for arbitrary audio convolution tasks. However, this documentation concentrates on its use as an Ambiophonics or Ambisonics surround sound processor, by providing tutorials on how to build such systems from scratch.

AlmusVCU runs on Linux-based operating systems. It can either be installed and run on an ordinary Linux workstation, or on what it is designed for, a purpose-made embedded computer platform, forming a HiFi-component of the highest grade, at an unbeatable price.

The power of a modern standard computer platform allows AlmusVCU to outperform most custom-made DSP platforms in terms of throughput. At the cost of a small I/O-delay (˜3 - ˜200 ms depending on application, configuration and hardware), AlmusVCU gives the audio processing power and sound quality of commercial hardware solutions costing 10 times more than the computer it requires. The software itself is free.

1.2.1 What is convolution?

AlmusVCU uses convolution to achieve its magic.

Convolution is a mathematical operation for applying one signal's characteristics to another. The signal whose characteristics is applied is often called "impulse response". In the world of digital audio the impulse response is a normally a very short signal that describes the characteristics of a filter. When the impulse response is convolved with recorded sound, its characteristics is applied to the recording. Thus if it holds the characteristics of a low pass filter, the result of the convolution will be a low pass filtered recording.

With all the processing power a modern computer provides, convolution can be used to do more than simple filtering. For example, the impulse response can represent the room acoustics of a concert hall. Thus, if it is convolved with a recording, the output will sound as a recording made in that concert hall.

As most mathematical operations, convolution can be found in nature. For example, convolution is the way reverberation is created in the real world. A concert hall is acting as a gigantic analog computer, which convolves the sound sources within it, through its many reflective surfaces. The convolved result is perceived as reverberated sound for a listener seated in the hall. This is why reverb simulation by employing convolution sounds so realistic, because it is done like "the real thing".

1.2.2 Features

Powerful convolution engine

The underlying convolution engine in AlmusVCU is BruteFIR, which employs partitioned convolution, meaning that long convolution can be performed while maintaining reasonable low I/O-delay.

More information about BruteFIR is available at:

`www.ludd.luth.se/~torger/brutefir.html`

Real reverb reproduction

AlmusVCU can be loaded with "reverb programs", which are 3D impulse response recordings of fine concert halls. A 3D impulse response of a concert hall is simply a recording of its acoustics. The impulse responses have been measured in the best seat of the hall, where one 3D impulse response has been stored for an impulse launched from the left part of the stage, and one from the right part.

The resulting reverb program is used by AlmusVCU to convolve the left and right channels of stereo recordings in order to produce a realistic reverberation to a speaker array surrounding the listener.

Since a reverb program contains information in all three dimensions, AlmusVCU knows how a reflection from any direction would sound in the original hall when an instrument is played on the stage. Hence, AlmusVCU is not limited to fixed speaker positions or a fixed amount of speakers; the reverb array may consist of few or many speakers placed in any positions.

Cross-talk cancellation

AlmusVCU can be loaded with a cross-talk cancellation program, which is assigned to a pair of narrowly spaced speakers placed in front of the listener. This speaker pair replaces the stereo triangle of an ordinary stereo system. The cross-talk cancellation program acoustically cancels out the speaker cross-talk, meaning that sound from the left speaker reaches only the left ear, and correspondingly for the right side. In ordinary stereo, sound from each speaker reaches both ears which causes various distorsion effects, and limits the stage width to the space between the speakers, which is not the case for cross-talk cancelled stereo.

The cross-talk cancelled stereo setup is most commonly referred to as "Stereo Dipole". However, since AlmusVCU can be used for Ambiophonics, the cross-talk cancelled stereo pair is referred to as "Ambiopole", which works according to the same principle as the Stereo Dipole, but the cross-talk cancellation is optimised only for the pinna-less head-shadow. In plain English this means that the ambiopole is less sensitive to variations in listeners' shapes of head and ears, and is only used for reproducing sounds originating from about a 150 degree arc in front.

The design of the loaded cross-talk cancellation program which contains filters for direct-path(s) and cross-path(s) specifies exactly how the cross-talk cancelled stereo pair will operate. Naturally, it is also possible to load classical Stereo Dipole filters as well.

AlmusVCU can do processing for more than one cross-talk cancelled speaker pair at once, to accomodate special applications such as car-audio installations and Panorambiophonics.

Ambisonics decoder

AlmusVCU can decode both first and second order Ambisonics to any regular speaker array, and also supports shelf filtering of the inputs.

Supports hybrid systems

The reverb engine, Ambisonics decoder and cross-talk cancellation can be combined in any way, forming new hybrid sound systems.

Customisable equalisation

Each channel can be equalised digitally with custom equalisation programs. This way AlmusVCU can digitally equalise speaker and/or room problems. Several filters can be cascaded into a single equaliser using the built-in equaliser designer.

Dynamic equaliser

The built-in equaliser designer allows design of dynamic linear-phase equalisers whose magnitude responses can be changed in runtime.

A/B testing capabilities

Apart from being able to switch reverb programs in runtime, cross-talk cancellation programs and sets of equalisation programs can also be switched. This feature can be used to do A/B testing to compare the performance of cross-talk cancellation programs or different equalisation settings.

Customisable pass-through channels

Configure "passthru channels", that is copy, mix, scale, delay and equalise any input channel and mix them to any output channel. This can be used to connect the AlmusVCU machine to another external unit, for example another cross-talk cancellation unit. Other possible uses are to extract low-frequency channels and mix in direct sound channels into reverb speakers.

Speaker feeds and internal channels work independently from each other. This means that several channels can be mixed into a single speaker.

Trim delay and trim volume for each channel

Each internal channel has a trim delay and trim volume setting. Trim volume can be used to compensate for heterogeneous speakers and amplification power. Trim delay can for example be used to compensate for varying speaker distances, or I/O-delay of cascaded audio processors.

Support for several sample rates

Any reverb, cross-talk cancellation or equalisation programs loaded will be resampled if necessary, using highest quality band-limited interpolation as the resampling algorithm.

AlmusVCU also allows to support for instance 44.1 and 48 kHz at the same time, so the source clock rate can be changed without a need to reconfigure.

Built-in benchmarking

The software has a built-in benchmarking engine which is used to recognise what the current hardware is capable of with the current AlmusVCU configuration, so the user gets alternatives for I/O-delay/reverb length trade-off to choose from.

1.2.3 User feedback

AlmusVCU is developed in a research manner, meaning that it is more important to have unique features than being error free. It is still a high priority to fix any known bugs, but it is not a high priority to search for them, since that would consume valuable time rather spent on improving or adding features. This means that the user should be prepared for that bugs may appear, and if they do, reporting them back to the author, so they will be fixed. Patient users reporting bugs are important for improving the overall software quality.

Any success stories, complaints or suggestions are also welcome, as well as questions regarding how to use the software and feedback on this documentation. Questions concerning the installation or use of Linux-based operating systems should however rather be directed to the appropriate forums.

1.2.4 Acknowledgements

This section will be written when the project is more ready, that is when one might not dislike to be associated to it.

1.2.5 Legal notices

License

Disclaimer

Trademark notices

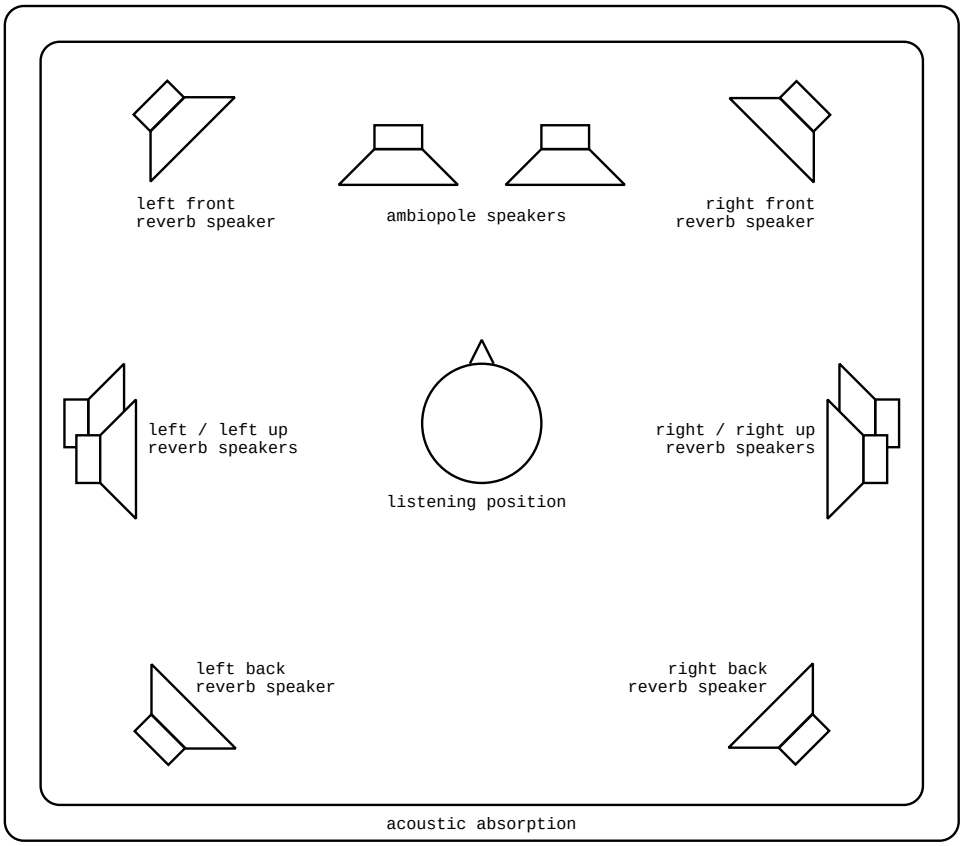

Figure 1.1: a typical Ambiophonics system, with an ambiopole and eight reverb speakers.

Audio Solutions. Ambisonics is a registered trademark of Nimbus Communications International. Linux is a registered trademark of Linus Torvalds. Pentium is a registered trademark of Intel Corporation. Athlon is a registered trademark of Advanced Micro Devices, Inc. Other product and company names mentioned herein may be trademarks and/or service marks of their respective owners.

1.3 What is Ambiophonics?

Ambiophonics is a multi-channel sound reproduction method that employs well documented but under-appriciated psycho-acoustic principles to create believable concert hall sound fields for one or two listeners in a dedicated listening room. The method was designed to reproduce recordings of acoustic music in a way that is perceived as real.

While other multi-channel reproduction methods usually require specifically made recordings to work well, Ambiophonics reproduces the LPs and CDs of today and yesterday, as well as the DVDs and SACDs of tomorrow. Ambiophonics is not merely “stereo compatible”; it reproduces stereo recordings better than stereo itself!

Better in the world of Ambiophonics is “more realistic”, that is closer to a live production. Of course, it is up to the individual to decide if more realistic is to be considered as “better”. For acoustic music, few disagree.

A full Ambiophonics system consists of a reverb array of eight or more speakers which reproduces the original concert hall reverberant field, and an ambiopole (cross-talk free stereo) which reproduces the sound coming directly from the stage. This way the listener gets a "you-are-there" experience, instead of the "they-are-here" which ordinary stereo systems are limited to, since they lack the hall acoustics reproduction. Compared to the stereo triangle, the ambiopole adds among other things accuracy in localisation, and an overall sharper sound stage.

It is not strictly necessary to build a true Ambiophonics system to get some of its advantages. It is possible to use fewer reverb channels, only two for example. Also, if one does not have speakers or environment suitable for the ambiopole, one can stick with the stereo triangle and just add a reverb array to it.

Ambiophonics, which is a public domain technology, has not yet got widely adopted in part because the signal processing hardware for reproducing reverberation realistically has been very expensive. AlmusVCU provides this for free.

Further information on Ambiophonics can be found at:

`www.ambiophonics.org`

1.4 What is Ambisonics?

Despite the similarity in names, Ambisonics has nothing to do with Ambiophonics.

Ambisonics is a technology that allows capturing the full 3D sound field in a single point with a specially made microphone, and then reproduce it over a loudspeaker array. The signal from the microphone is called B-format, a four channel format containing a first-order representation of the recorded sound field. The four channels, labelled W, X, Y and Z, are not actual speaker feeds, instead they are fed through a decoding matrix adapted for the speaker array used. The speaker array can contain any amount of speakers and be of any form, although arrays with a regular form are the most common.

To successfully reproduce all three dimensions, a minimum of eight speakers is recommended. However, height is often omitted, and then four speakers, suitably placed in a square around the listener, is enough for reasonable good results.

The recorded sound field is recreated in the center of the speaker array, through superposition (the sum) of all emitted sound from the loudspeakers. The more speakers, the higher precision of the reproduction. Higher precision (improved localisation) on the recording side can be achieved by employing second-order Ambisonics, which requires nine channels. Currently, second-order recordings can only be created artificially, since there is no working second-order microphone.

Ambisonics has had only limited commercial success, not because of the technology itself, but rather due to bad timing. It came at a time (in the seventies) when there was no good format for distributing multi-channel recordings, and when the failure of (the unrelated) quadraphonic surround was fresh in memory. Ambisonics supporters now hope for a renaissance with the arrival of the new DVD-audio and SACD multi-channel distribution formats.

The amount of commercial available true B-format recordings is very limited, and even more limited for second-order material. However, there are quite many UHJ-encoded CDs available. UHJ is a matrixed format, where the W, X and Y components are matrixed into the two channels of a CD. Actually UHJ supports matrixing

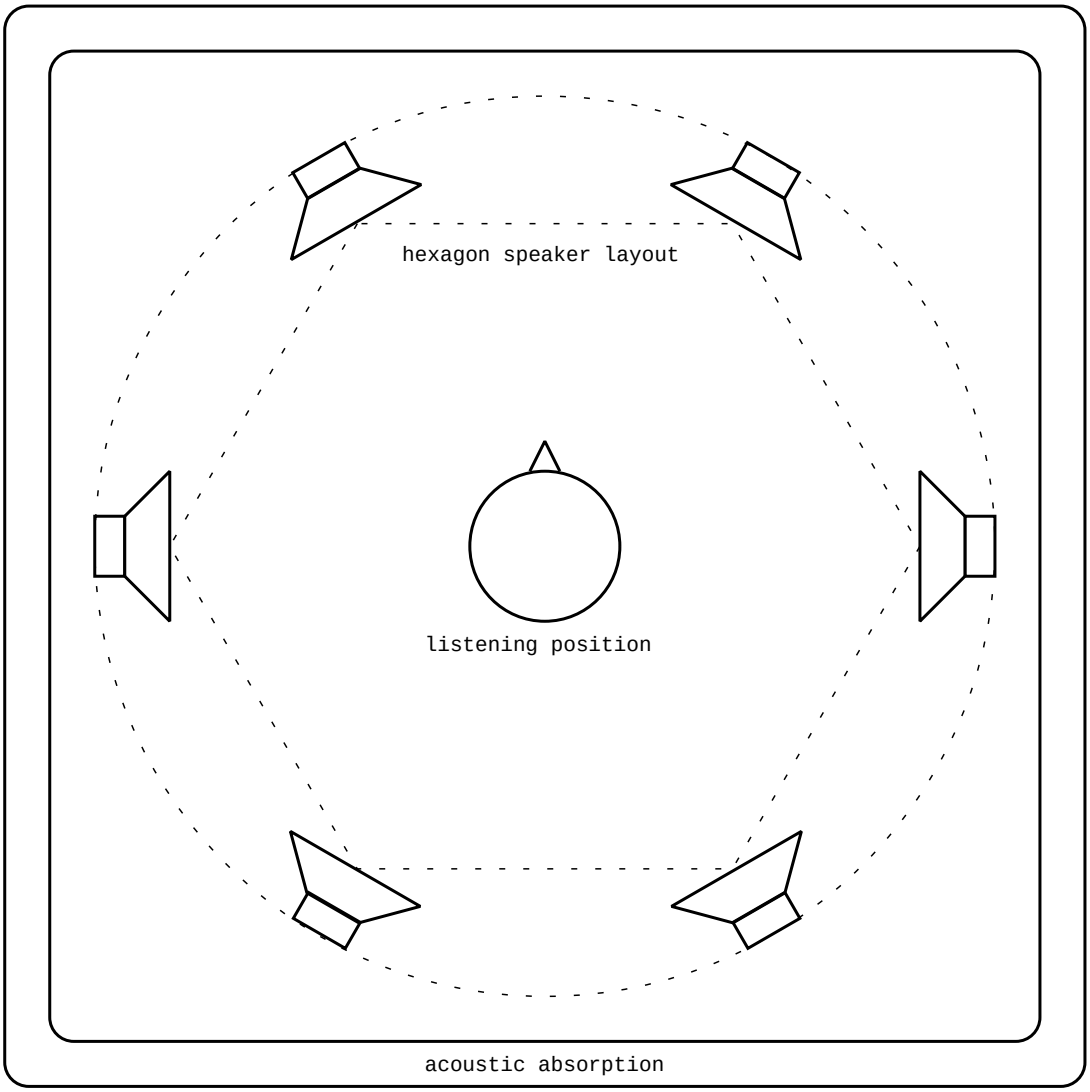

Figure 1.2: a typical Ambisonics system, employing a horisontal hexagon speaker layout.

with more channels involved as well, but since only two-channel material has appeared commercially, UHJ has become synonymous with the two-channel variant.

UHJ-encoded CDs lack the Z component so height cannot be reproduced, and the channel separation is of course not very good, since it is a matrixed format.

There is also a commonly used stereo compability mode for Ambisonics, which is used to reproduce stereo recordings, and allows the stage width to be modified.

Further information on Ambisonics can be found at:

`www.ambisonic.net`

1.5 Ambiophonics vs Ambisonics

Both Ambisonics and Ambiophonics sonically outperforms the mass-market standards stereo and 5.1, but how to they compare to eachother? To make a long story short, Ambiophonics is the system with the higher fidelity, thanks to the use of the ambiopole, which provides superior localisation and focus.

However, while Ambiophonics is limited to staged music, Ambisonics can reproduce sound from any direction. It could be said that Ambiophonics is highly specialised on its task of reproducing staged music and has avoided solutions which would extend flexibility at the expense of fidelity, while Ambisonics can do anything.

Here follows some other differences:

- Ambisonics is based on solid and elegant mathematics, while Ambiophonics takes a pragmatic psycho-acoustic approach.
- Ambisonics decoding is quite well-defined, while Ambiophonics is more of a collection of ideas of how to build a reproduction system using cross-talk cancelled stereo and a reverb array.
- Ambisonics aims at wide adoption on the mass-market, while Ambiophonics has a more elitist approach, aiming at precision listening installations only.
- Ambiophonics is less dependent on specially made recordings than Ambisonics is, meaning that Ambisonics needs mass-market adoption more than Ambiophonics does.
- Ambiophonics is much more computationally demanding than Ambisonics, which in its basic form only requires mixing. Only recently the very demanding reverberation simulation required by Ambiophonics has become practical.
- Ambiophonics has a sharper sweet spot (or rather sweet line) than Ambisonics. While Ambiophonics allows only for two or three listeners at a time (sitting along a line), Ambisonics sweet spot can be diffused to support a larger area and more listeners.

For the sake of honesty, it should be noted that the author is involved in developing Ambiophonics, and thus this comparison is probably a bit biased. Fortunately, AlmusVCU provides the opportunity to try out both formats, so the reader can decide for herself/himself.

Recently, experiments of combining Ambiophonics and Ambisonics into a single system have been conducted. The most advanced has a front and a rear ambiopole and

an Ambisonics array, meaning eight recorded channels, two for the front ambiopole, two for the back, and four for the B-format for the Ambisonics array. While this type of recordings probably never will get commercially available in great numbers, they show that the Ambisonic and Ambiophonic concept can indeed be combined.

Chapter 2

Installation

2.1 Introduction

To be honest, for the user inexperienced with computers in general and Linux in specific, installing the AlmusVCU software will not be easy. The documentation here assumes that the reader has some basic knowledge in both computers and Linux.

Fortunately, there is the alternative to use the AlmusVCU bootable CD, which is extremely easy to use, and requires no software installation. It is described in chapter 6.

This chapter also provides some tips and suggestions about the audio equipment which should be connected to the AlmusVCU machine.

2.2 Preparing the platform

2.2.1 Hardware

The computer should be a standard PC compatible machine. You can either choose to install AlmusVCU on an ordinary Linux workstation, or build a purpose-made computer which will only run the AlmusVCU software.

The recommend way is to build a computer dedicated to AlmusVCU, since it will need a carefully configured operating environment for reliable operation, which is not achievable on all workstations. The software is also designed to be built into an embedded hardware platform, forming a HiFi component, with LCD and remote control

In any case, the largest problem will propably be the noise of the computer. There are some near-silent computers on the market (unfortunately not all with proper cooling), and with some effort, one can also be built. The easiest way is however to put the computer in a neighbouring room, and use long cables for the display and remote control unit.

Processor (CPU)

An Intel Pentium III, Pentium 4 or AMD Athlon running at 1 GHz or more is recommended as the CPU. The faster the processor, the more channels with longer

reverb times in the reverb engine can be used. For low end systems with few reverb channels (four or less) or if the reverb engine is not to be used at all such as in an Ambisonics system, a 500 MHz processor may be enough.

If a Pentium 4 or other processor with thermal throttling is used, that is its performance is reduced when the processor gets too hot, make sure it has proper cooling so throttling never occurs.

Any other runtime performance altering technologies must be avoided, since the AlmusVCU convolution engine will load the processor constantly at high load. Any performance drop will cause buffer underflow and stop the audio processing.

AlmusVCU supports multiple processors. However, not all filtering tasks can successfully be distributed onto multiple processors. The reverb engine configured to use many channels and use a unique output for each reverb channel, will be the best suited for distribution. The reverb engine configured with many channels is also the part of AlmusVCU that uses the most processor time.

A large processor cache is beneficial, since AlmusVCU is very memory intensive.

Memory (RAM)

256 megabytes of RAM or more is strongly recommended. The more RAM, the more filter programs can be kept simultaneously in memory. Reverb programs is typically the filter programs that uses the most memory.

If AlmusVCU will run with few channels, and/or without the reverb engine, less than 256 megabytes can be used, but less than 128 megabytes is not recommended.

The faster memory the better, since AlmusVCU is both very processor and memory intensive. For high-performance systems, fast memory is as important as a fast processor.

Sound card

The software supports in principle all sound cards supported by ALSA (Advanced Linux Sound Architecture), but it is strongly recommended to use high quality digital sound card, and use external D/A converters (and A/D converters if necessary), since a computer is not a good environment for analog audio signals. Since AlmusVCU most often is used in multi-channel applications, the sound card must support at least as many channels as the number of loudspeakers in the system. A suitable digital multi-channel standard is ADAT, which is a format for transferring eight channels of digital audio in a single cable.

The digital transmission format for stereo called S/PDIF is the most common digital output format used by CD players and similar devices, so it is recommended that the sound card can accept that as input.

Another important feature of the sound card is that it must be full duplex, that is it must be able to receive input at the same time as it generates output. It is possible to use one sound card for input, and another for output, but it is not recommended, mainly since the I/O-delay cannot be guaranteed with such configuration (and sample clock synchronisation may be problematic). I/O-delay is the time it takes from the first sample is received on the input until its processed version(s) are available on the output(s). If the sound card supports synchroneous starting of input and output, then I/O-delay will be fixed. This means that the AlmusVCU machine can

be sample-aligned with other devices if necessary. It is also good if the sound card can operate with short interrupt cycles, since it allows for shorter I/O-delay.

Currently, the RME Audio Hammerfall sound card is recommended (full or light version), since it supports all features mentioned above, and has been tested together with the software.

If the input is digital, the sound card must be configured to work as clock slave, that is adapt to the clock signal received on the input. This is normally achieved using the ALSA configuration tools.

Display

AlmusVCU can produce output to a regular computer console, or to an LCD.

Currently, the only supported LCD is Matrix Orbital LCD2041 (the VFD version should work as well). For a programmer it is however very easy to add support for other displays to AlmusVCU, as described in section 8.2. The requirement for the display is that it presents text and is at least 20 columns wide and at least 2 rows high. The optimal size is 20 columns and 4 rows. Using the LCD instead of a regular computer screen makes it possible to build a stand-alone box with built-in display.

Hard disk

The hard disk is used to store configuration and cached copies of loaded filter programs. For silent operation a solid state disk can be used. However, since the harddisk is not accessed in runtime (when no settings are being altered), a standard harddisk with which is configured to turn off when it is not accessed work as well.

It is good to have about 50 megabytes or more available to AlmusVCU, but smaller sizes is also possible as long as it is larger than about 10 megabytes.

User input

Navigating the user interface is done with only four buttons, LEFT, RIGHT, UP and DOWN. This can be mapped to the directions keys of a standard computer keyboard, or a computer mouse with scroll wheel (left/right buttons = LEFT/RIGHT, scroll wheel = UP/DOWN, the pointing function is not used). For remote control a cordless computer mouse is suitable.

For a programmer it is simple to add support for new user input devices to AlmusVCU, as described in section 8.2.

2.2.2 Operating system

Any recent Linux distribution can be used as the base operating system. With the proper hardware and light configuration, AlmusVCU allows for as low I/O-delay as 3 ms. In order to avoid buffer underflows, it is necessary to tune the operating system to be able to deal with the low latencies required.

Buffer underflow means that the sound processing fails to fill the sound card output buffer in time, and then the convolution engine will stop with an error, and must be restarted. Thus, nothing bad happens apart from that the processing stops, so if

several hours or days of uninterrupted operation is not necessary, one can do with a standard installation.

To make a system that can handle the lowest latencies is a hard task, and there is no definite guide on how to do it. However, if not the lowest latencies are strived for, a standard installation may work reliably. It is recommended to use a low-latency kernel though, and avoid running processes that periodically wake up and execute (demanding) tasks.

Low-latency kernel

The Linux (2.4) kernel in its standard form may not meet the latency requirements, and should therefore be patched with a suitable low-latency patch. There are many low-latency patches in circulation, and which one is most suitable for the moment must be found out at the time of installation.

Periodic processes

Daemons such as cron which wake up and run tasks on a regular basis should not be installed, or be configured not to run periodically. These could steal processor time when it is most needed, and cause a buffer underflow.

Swap disk

The system should not be configured to use a swap disk, since any swapping may introduce latencies causing buffer underflow.

Further requirements

The source code archives contain information on which libraries that are required, and any other requirements on the operating environment that may exist.

2.3 Compiling

After the computer platform is ready, and a Linux-based operating system has been installed on it, it is time to compile, that is turning the human-readable source code into binary code runnable by the computer. The programs to compile are AlmusVCU and its convolution engine BruteFIR. Instructions for compiling is contained in the source code archives.

2.4 Initial configuration

The AlmusVCU configuration is stored in text format configuration files. There is one static configuration file, `/etc/almusvcu_static_config`, and one user configuration file, `almusvcu_user_config`. Additionally, there are separate configuration files for each reverb program loaded. AlmusVCU creates all configuration files itself, apart from the static configuration file, which must be written by the user.

Before AlmusVCU can start and enter the setup menu, it must know which sound card hardware to use, how much memory to allocate and other static settings. These are put in the static configuration file.

The standard path to the static configuration file can be overridden by giving an argument to AlmusVCU, the first will be interpreted as the filename.

With the source code archive follows a sample static configuration file, which is richly commented, and most settings does not need additional documentation.

2.4.1 Sound card

For an RME Audio Hammerfall card, the input/output device settings should be the following:

```
input_device: "alsa"/'{ param: "hw"; }';
input_channels: 26 / 24, 25,
     0,  1,  2,  3,  4,  5,  6, 7,
     8,  9, 10, 11, 12, 13, 14, 15,
    16, 17, 18, 19, 20, 21, 22, 23;
input_formats: "s24_4le", "s16_le";
output_device: "alsa"/'{ param: "hw"; }';
output_channels: 26 / 24, 25,
     0,  1,  2,  3,  4,  5,  6, 7,
     8,  9, 10, 11, 12, 13, 14, 15,
    16, 17, 18, 19, 20, 21, 22, 23;
output_formats: "s24_4le", "s16_4le";
```

If instead the card is an RME Audio Hammerfall Light:

```
input_device: "alsa"/'{ param: "hw"; }';
input_channels: 18 / 16, 17,
    0,  1,  2,  3,  4,  5,  6,  7,
    8,  9, 10, 11, 12, 13, 14, 15;
input_formats: "s24_4le", "s16_le";
output_device: "alsa"/'{ param: "hw"; }';
output_channels: 18 / 16, 17,
    0,  1,  2,  3,  4,  5,  6,  7,
    8,  9, 10, 11, 12, 13, 14, 15;
output_formats: "s24_4le", "s16_4le";
```

Some Hammerfall Light cards are identified as full versions and will require to be configured with the full 26 channels, channels with indexes 16 - 23 will not be usable though.

Note the special ordering of the channels in this static configuration. Channels 24, 25 (16, 17 for Hammerfall Light) are the S/PDIF channels, and by listing them first they will be mapped to the two first channels in AlmusVCU. The following channels are ADAT, three times eight for the full version, two for the light version.

The static configuration starts with channel index 0, while in the program the first channel index is 1. The above static configuration will thus yield the following in runtime:

- channels 1 - 2: S/PDIF
- channels 3 - 10: ADAT 1
- channels 11 - 18: ADAT 2
- channels 19 - 26: ADAT 3

This counts for both inputs and outputs. ADAT 3 is not available on the Hammerfall Light version.

As mentioned in section 2.5.1, the sound card must be configured to work as clock slave if any of its inputs is used.

2.4.2 User interface

The `ui_modules` setting which defines the user interface requires some explanation. It may look like this:

```
ui_modules:
    "tw_curses",
    "tw_matrix_orbital" {
        device: "/dev/ttyS0";
    },
    "ki_curses",
    "ki_gpm",
    "textgui" {
        rom_path: "/cdrom";
        rom_name: "CD-ROM";
        rom_device: "/dev/cdrom";
        rom_filesys: "iso9660";
        rom_mount: true;
        display_size: 20, 4;
    };
```

The setting describes which user interface modules that should be loaded. Without these modules AlmusVCU cannot be used. The modules define how to take input from the user, and how to present output to the user. This manual describes the user interface as presented with the standard modules delivered with the software, which is a text interface adapted to fit small LCDs.

As seen in the example, the name of the user interface module is within quotes, and if there are any settings for the module, the will follow directly after within braces. In the above example the modules `tw_curses`, `tw_maxtrix_orbital`, `ki_curses`, `ki_gpm` and `textgui` are loaded. Some modules depend on others, and may also require to be loaded in a specific order (modules will be loaded in the order they are listed). In this case, `textgui` is the master module providing the user interface for AlmusVCU. However, it needs other modules to get input from the user, in this case `ki_curses`, which takes input from the keyboard, and `ki_gpm`, which takes input from the mouse. Note that both can be used at the same time.

The `textgui` module also needs modules for outputting the characters forming the user interface. This is provided by the modules `tw_curses`, which puts the text to the computer screen, and `tw_matrix_orbital`, which puts the text to a

Matrix Orbital LCD2041 display attached to the given serial port (`/dev/ttyS0` in the example). As for the input modules, several can be used at the same time.

The user interface provided by `textgui` allows loading filter programs from a read only unit. This is configured with the `rom_*` settings, which in the example is adapted for a CD-ROM, which is probably the way to configure for an embedded system. A workstation installation would probably rather point out a directory on the harddisk as the read only unit. For a harddisk, `rom_mount` should be set to false, and then `rom_device` and `rom_filesys` do not need to be set. An example:

```
"textgui" {
    rom_path: "/usr/local/almusvcu/programs";
    rom_name: "harddisk";
    rom_mount: false;
    display_size: 20, 4;
};
```

The last setting `display_size` defines the size of the text display, the number of columns and rows. The `textgui` module is adapted to 20 columns width, but accepts wider (no narrower though). The height must be at least two, and is optimised for four.

2.5 Surrounding equipment

AlmusVCU is typically used in a home HiFi sound system as a surround decoder. Thus, it needs a signal source connected to it, and the signals it delivers must be amplified and delivered to loudspeakers. This section gives tips and suggestions on how AlmusVCU should be connected to the surrounding equipment in a home HiFi setup.

2.5.1 Signal source

The signal source is preferably connected digitally directly to the AlmusVCU machine. The most common signal source today in a home HiFi sound system is a CD player. It should be connected through the digital S/PDIF interface, which supports stereo 44.1 - 48 kHz (some extend it outside its specification to support up to 96 kHz), and is transported through a coaxial or optical cable.

If the signal source is analog, such as a turn-table for vinyl recordings, it is necessary to have analog-to-digital conversion before feeding it to AlmusVCU. Preferably this is done in a separate A/D converter with S/PDIF output.

Modern digital sources

The new audio standards SACD and DVD audio provide higher resolution than CD, typically corresponding to 24 bit at 96 kHz. The real improvement is in the dynamic range (24 bits instead of 16) rather than the increased sample rate, which only provides a very minor improvement detectable by few. Anyway, the problem for AlmusVCU is to get the signal digitally from the the signal source.

Most players have only analog outputs, and those with digital most often provide a down-sampled S/PDIF version on the output. That there is no widely established

standard for transferring these new digital standards is one problem, another is that when it is established, the source material will probably be encrypted and thus be unavailable to open-source projects such as AlmusVCU.

The practical solution today, and will probably hold tomorrow as well, is to connect the DVD or SACD player through its analog outputs to a high-quality A/D converter running at 24 bits / 48 kHz, and then feeding it to the AlmusVCU machine through S/PDIF.

Master sample clock

A common mistake is to have more than one master sample clock in a fully digital system. Unfortunately, it is also hard to detect. The system will seem to work fine, but at some point sample dropouts occur (might be heard as occasional clicks in the sound), or AlmusVCU processing will stop.

The problem is more easily understood through an example: the CD player provides its master clock, but the sound card in the AlmusVCU machine is configured to work as clock master. In this case the sample clock for the CD player runs at 44.1 kHz, and probably provides it through the S/PDIF interface. However, the AlmusVCU will provide its own clock on its output, and clocks are never exactly the same. The CD player clock may be 44.1001 kHz, and the sound card clock in the AlmusVCU machine may be running at 44.0999 kHz, causing the input and output of the AlmusVCU to slowly slip apart, and at some point processing will stop with an error.

The correct way is to make sure all digital devices use the same sample clock, usually of that of the CD player or other digital source. Thus the sound card in the AlmusVCU machine must be set to work as clock slave, if it has digital input that is. This cannot be done through the AlmusVCU interface, nor in its static configuration, but must be configured with the tools coming with ALSA.

2.5.2 A/D and D/A conversion

It is strongly recommended that the D/A and possible A/D conversion is not made inside the AlmusVCU machine, but instead in external dedicated converters. The analog performance is considered to generally be better in external converters than those that need to cope with the rather hostile environment (for analog technology) inside a computer.

Common digital standards for digital transmission are S/PDIF and ADAT. S/PDIF supports stereo 44.1 - 48 kHz, and ADAT supports eight channels in 44.1 - 48 kHz. ADAT is transported over an optical link, S/PDIF through a coaxial cable or an optical link. There exists a wide range of S/PDIF D/A converters on the market, typically targeted on the audiophile market. S/PDIF A/D converters are less common though. ADAT converters are often used in the professional recording studio, and are thus often combined A/D and D/A converters. There is a wide range of ADAT converters on the market to choose from.

2.5.3 Amplification

The signals from the D/A converters need to be amplified. There are multi-channel amplifiers on the market, made for multi-room applications or home cinema, which can be used for multi-channel applications. However, ordinary stereo amplifiers can

sometimes give just as good value for money, and an advantage with them is that one can expand the system in smaller steps, only two channels at a time.

Since the volume is controlled in the digital domain by AlmusVCU, the volume settings for the amplifiers are fixed, typically set to their highest level when the noise floor is still acceptable. For integrated amplifiers this means that the volume knob is put in a suitable position. If power amplifiers are used instead, there must be attenuators which set this volume level. An attenuator is simply a resistor of a suitable value. Proper pre-amplifiers can be used instead, as they serve the same purpose, with the difference that they allow to change the volume setting (and active pre-amplifiers may provide some sonic improvement).

If the choice is to use attenuators, these must be self-made after finding the suitable resistance after some experimentation, or variable audio attenuators can be bought.

Digital amplifiers

There are some amplifiers on the market which accept digital input directly, and has a volume control which alters the reference voltage rather than the digital input signal. This improves at least theoretically the signal-to-noise ratio. Currently AlmusVCU does not support this operation. Instead it alters the volume by scaling the digital signal. However, the signal-to-noise ratio of a 24 bit digital signal outperforms any analog amplifier of today, so controlling the volume by scaling is only a minor or no problem in practice.

Some of the digital amplifiers allow changing the volume through a serial interface, which could be connected to the AlmusVCU machine, so in future releases the volume control of some digital amplifiers might be supported directly.

In any case, digital power amplifiers (those with digital input) should not need any attenuators. It should be verified how the volume is handled though, a digital power amplifier might require a matching pre-amplifier to work at all.

Chapter 3

Using AlmusVCU

3.1 Concepts of operation

3.1.1 Block diagram

The block diagram in figure 3.1 shows the runtime audio signal flow through AlmusVCU. There are four distinct convolution processing blocks: ambiopole, reverb ambisonics and passthru. All of them are optional (at least one must be activated though), thus the block diagram is just an example showing the signal flow when all processing types are in use.

Ambiopole

The ambiopole convolution process block provides cross-talk cancellation for one or more ambiopoles. Standard Ambiophonics or Stereo Dipole installations use only one ambiopole, which is the normal case. However, special applications such as Panorambiophonics, multi-room or car-audio installations require more than one.

Optionally, equalisation can be applied to the outputs, as shown in the "ambiopole eq" block. Convolution in that block will only be performed if equalisers are assigned.

The convolution elements in the cross-talk cancellation block uses filters loaded from cross-talk cancellation programs. Cross-talk cancellation is however optional on an individual basis; if disabled, two direct passthrough channels replace the four cross-talk cancellation filters and their mixers.

Reverb

The reverb convolution process block provides feeds for a reverb speaker array. The convolution filters in the reverb engine block are derived from a given reverb program. As seen in the block diagram, two filters are employed per reverb channel, one each for a left and a right input signal. These filters are each as long as the reverb time of the given reverb program, which typically is a couple of seconds. Since the reverb simulation block in normal configurations has the most channels and the longest filters, it is normally the most processing intensive part of AlmusVCU.

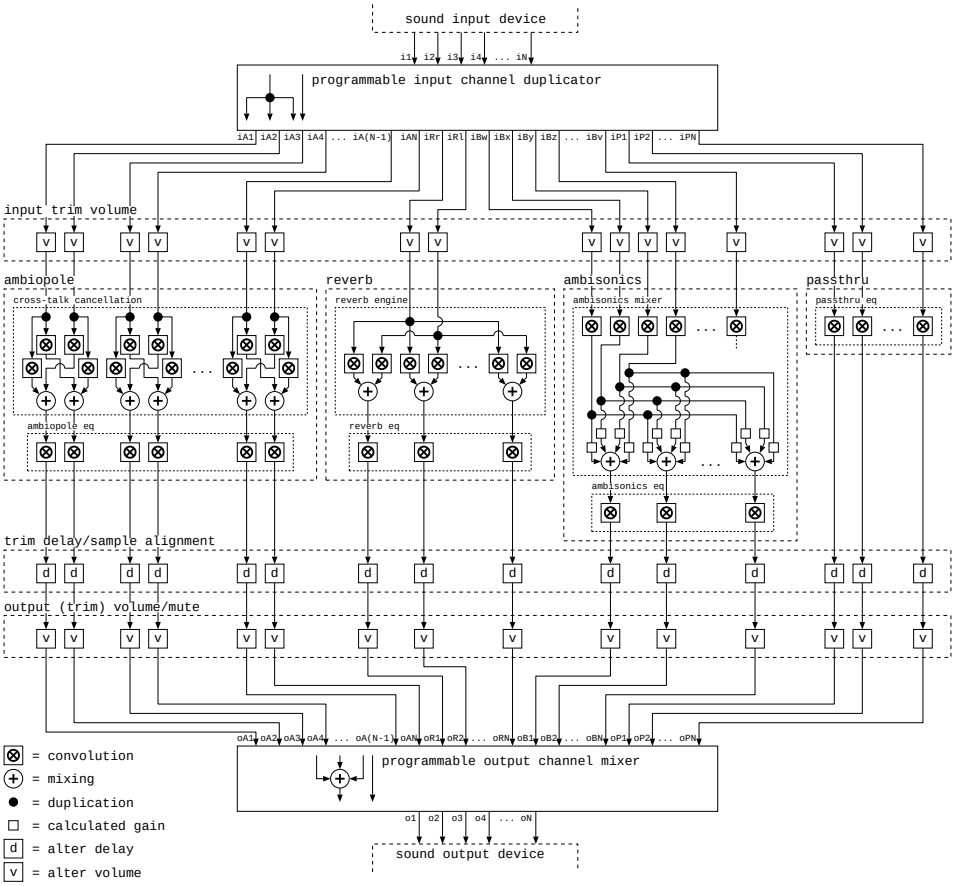

Figure 3.1: AlmusVCU block diagram

The left input signal should contain sound which is supposed to originate roughly from the left part of the stage, and the right input signal should contain sound originating roughly from the right part. In cases where the input signal is mono, the reverb engine should be configured to use that signal for both left and right input.

The amount of reverb channels is configurable. Optionally, the reverb equalisation block can be activated and then equalisation filters can be assigned to each channel. This does however introduce extra processing requirement for each channel, even those which do not have equalisers assigned to them.

Ambisonics

Actually, Ambisonics decoding does not require convolution, it only needs to apply gains and mix. Thus, for very basic Ambisonics decoding, AlmusVCU would seem quite inefficient, since it has some overhead associated to convolution even when it is not performing any convolution at all. However, more advanced Ambisonics decoders include psycho-acoustic filters on the input channels, which also AlmusVCU does, even allowing them to be modified in runtime. AlmusVCU also adds the possibility to apply equalisation filters to the output channels. The number of output channels is configurable.

The gains before mixing is automatically calculated by AlmusVCU from the given speaker layout.

Both first and second order Ambisonics decoding is supported.

Passthru

This block provides ways of passing through audio signals through AlmusVCU without special ambisonic, reverb or ambiopole processing. It is entirely configured by the user, which also decides what to use the passthru channels for. One use could be to provide sub woofer channels. The amount of passthru channels is configurable (there are none in the default configuration), and equalisers can be applied on an individual basis. Convolution will only be performed on the passthru channels with an active equaliser.

Sample alignment

The actual filters used in the convolution processing may introduce delay. For example, cross-talk cancellation filters typically introduce a 10 - 20 ms delay, and if any equalisation filters are used, they may introduce delay as well. AlmusVCU finds out which channel that has the largest flow-through delay through the convolution processing, and then delays all other with the proper amount in order to get exact sample alignment for all channels.

Channel delay and volume settings

Configurable trim delays and volumes can be applied on an individual basis to all channels produced by the four convolution processing blocks. These trim settings can be changed in run-time.

There are also overall volume slave and master volume settings, which are normally the only volume controls accessed in a readily configured system. Each internal channel is either associated to slave or master volume.

Channel duplicator and mixer

The internal ambiopole, reverb ambisonic and passthru channels work independently from the sound card mounted in the AlmusVCU machine. The user specifies how to distribute the sound card input channels to the convolution processing blocks. One sound input channel can be reused several times, that is be duplicated.

On the output side, the user specifies to which sound card output channels the internal channels should be routed. Several of the internal channels can be put to a single sound card output channel, that is be mixed.

Channel names

The user interface has short labels for all channels, which also can be seen in the block diagram. These are the following:

- i1, i2, i3, ... iN; the input channels of the sound card.
- iA1, iA2, iA3, ... iAN; the input channels for the ambiopoles. Each ambiopole takes two inputs, left and right, for the first ambiopole the input channels are thus iA1 and iA2, the second iA3 and iA4 and so on.
- iRl, iRr; left and right input for the reverb engine.
- iBw, iBx, iBy, iBz, iBr, ... iBv; the ambisonic input channels, iBw - iBz is for first order Ambisonics, and for second order the five extra channels iBr - iBv are added.
- iP1, iP2, iP3, ... iPN; the passthru input channels.
- oA1, oA2, oA3, ... oAN; the ambiopole output channel. Each ambiopole has two outputs, left and right. For the first ambiopole the output channels are thus oA1 and oA2, for the second oA3 and oA4 and so on.
- oR1, oR2, oR3, ... oRN; the reverb output channels.
- oB1, oB2, oB3, ... oBN; the ambisonic output channels.
- oP1, oP2, oP3, ... oPN; the passthru output channels.

3.1.2 Navigating the user interface

The AlmusVCU user interface is presented on a text display, with 20 columns width and at least two rows high, preferably four. The text display used can be a computer screen, or an LCD.

Navigating the user interface is done with only four buttons, LEFT, RIGHT, UP and DOWN. These can be mapped to the directions keys of a standard computer keyboard, or to a computer mouse with scroll wheel (left/right buttons = LEFT/RIGHT, scroll wheel = UP/DOWN, the pointing function is not used).

Menus

The basic element of the user interface is a menu. A menu contains a list of alternative actions. With UP and DOWN a pointer is moved to point out the alternative to choose. By pressing RIGHT the alternative is chosen. If there are more alternatives than can be shown on the display, an arrow pointing downwards in the lower left corner indicates that there are further alternatives below the lowest. The next section of the menu is then shown if pressing DOWN when pointing at the lowest alternative. If the next section of the menu is entered, an arrow pointing upwards will then be shown in the upper left corner to indicate that there is a previous section of the menu, which is reached by pressing UP when pointing at the uppermost alternative.

```
+--------------------+
|  Start convolver?  |
| >Load reverb prg...|
|  Del reverb prg... |
|v Signal setup...   |
+--------------------+
```

Each menu alternative describes what is done when it is selected (that is when RIGHT is pressed when the pointer is in front of it). This is done with the textual description together with some special character arrangements.

- Enter a sub-menu. The name of the menu is followed by three full stops. A sub-menu is exited by pressing LEFT.
- Perform a direct action. These alternatives is formed as a question with a question mark at the end.
- Cycle through a set of alternative values for a setting. The setting is described in text, and the currently selected alternative is within brackets.
- Pick a number. The current number is within brackets, and thus look the same as the cycle alternative. However, when selected, a number chooser mode is entered, where the value is increased by UP and decreased by DOWN. A multiplier to increase or decrease the incremental step of which the value is changed with UP/DOWN is cycled through with RIGHT. When the value is what is desired, LEFT is used to return to the menu.

```
+--------------------+
|  Sub-menu...       |
|  Perform action?   |
|  Cycle [a]         |
|v Pick number [0.1] |
+--------------------+
+--------------------+
|                    |
|                    |
|  <ok >0.1  ^+ v-   |
|v Pick number [3.4] |
+--------------------+
```

Screen saver

Some displays, such as VFDs (Vacuum Fluorescent Displays), may suffer burn-in effects if the same static screen is left on for several hours. Since it is common for AlmusVCU to be left on to process sound for long periods of time without user interaction, there is a built-in screen saver which will be automatically activated a while after the user has stopped giving input. It will simply clear the screen and move around the text "AlmusVCU" randomly on it, thus reducing the risk of burn-in effects.

When the screen saver is activated, it is left by pressing any of the four buttons (LEFT, RIGHT, UP or DOWN). The key press used to leave the screen saver is not passed along to the waiting user interface, so there is no risk to execute a command without seeing it.

3.2 Setup menu

The setup menu is the first shown when AlmusVCU is started for the first time. When in the setup menu, AlmusVCU is not in "runtime", that is the convolver (the convolution engine, BruteFIR) is not running, and no audio signals are being processed.

If there is no previously made configuration there is much work to do in the setup menu. The settings made in the menu defines what AlmusVCU should do.

```
+--------------------+
| >Start convolver?  |
|  Load reverb prg...|
|  Del reverb prg... |
|  Signal setup...   |
|  Channel setup...  |
|  Eq setup...       |
|  Ambiopole setup...|
|  Convolver setup...|
|  System info...    |
|  Maintenance...    |
|  Screen dimmer [2] |
|  Autostart [off]   |
+--------------------+
```

In the following sections the top level alternatives are first presented, and then follows one section for each sub-menu.

3.2.1 Top level alternatives

Start convolver

The alternative `Start convolver?` will start the convolver, that is the audio processing, and sound will come out through the speakers, and the runtime main screen will be entered. This is only possible to do if there is a valid configuration. If there is some configuration missing and the alternative is attempted anyway, an error message indicating the problem will appear.

Screen dimmer

The screen dimmer setting is dependent on the display output used. If the display used and its driver supports dimming, it will be affected by this setting, if not, it will do nothing.

Autostart

If Autostart is activated, the convolver will start directly when AlmusVCU is started, bypassing the setup menu. If by some reason the convolver cannot be started, an error message will appear, and after it has been acknowledged the setup menu will be entered.

3.2.2 Load/delete reverb program

Load and delete reverb programs can only be done if AlmusVCU is configured to have at least one reverb channel.

Loading and deleting reverb programs work the same as for custom equalisation and cross-talk cancellation programs, and can be read about in section 3.3.

3.2.3 Signal setup

The signal setup menu concerns signal quality properties of the input and output signals.

```
+--------------------+
| >Soft clip [off]   |
|  Dither [on]       |
|  [24] bit output   |
|  Allow 44.1kHz[on] |
|  Allow 48.0kHz[off]|
|  Suggest [44.1]kHz |
+--------------------+
```

Soft clip

This is not yet implemented, so the setting will do nothing. Anyway, when/if it is implemented it will do the following:

When a signal is passed through a digital filter such as one for revererberation, some frequencies may be amplified. This together with an aggressive volume setting may cause overflow in the digital domain. Overflow occurs if the signal is amplified such as the peak value need to be larger than the one possible for the output format. If soft clip is off, the output will simply truncate these peaks to the maximum possible value. This clean clipping of peaks will be heard as clicks in the resulting sound.

Instead, if soft clip is activated, the convolution engine does its best to hide clipping by compressing the peak to fit the available space. For sporadic minor clipping, the distorsion introduced by the compression will not be percievable. The drawback with soft clipping is that it introduces some extra processing time, typcially in the area of 15 extra percent.

Dither

If dither is activated, the output will be dithered. Dither is used to hide quantisation distorsion and increase resolution by randomising the error through adding noise, which is shaped in the frequency domain so it is not easily detected by the human ear. All modern digital recordings are dithered. However, when they are processed in the digital domain as in AlmusVCU, the signal need to be re-quantisised, and thus new dither must be applied to keep the effect.

If the output is set to be 24 bit and the resolution of the system is 32 bit floating point (which is the default compile-time setting of the software), dither will not be applied, since the internal resolution is not higher than the output. The improvement of dithering of a 24 bit output signal is however extremely hard to detect, maybe even impossible for the human ear.

For 16 bit output though, dither may provide a detectable improvement. The cost of adding dither is typically around 10 percent extra processing time.

Output resolution

The output resolution setting is only available if the statically configured sound card supports more than one resolution. Typically one can choose between 16 and 24 bit output. Normally, the highest resolution is the best choice, since there is no performance penalty.

Select sample rates

It is only possible to select sample rates if the statically configured sound card supports more than one sample rate.

If more than one sample rate is selected, the filter programs will be loaded with all sample rates, and thus occupy more RAM. The purpose of allowing more than one sample rate as input is to adapt to different signal sources, for example a CD player at 44.1 kHz and a DAT player at 48 kHz.

It makes no sense to select more than one sample rate if the sound card input is analog, that is the input signal is sampled with an A/D converter.

Suggested sample rate

This setting is only available if the sound card supports more than one sample rate. The suggested sample rate is what AlmusVCU will try to set the input signal to. For digital sound cards, the suggested sample rate is ignored, and the sample rate will be what the sample clock is on the input (the sound card must be configured to operate in slave mode to behave as expected). For analog inputs, the sample rate will be set to the suggested.

3.2.4 Channel setup

The channel setup menu is used to configure how many output channels that should be used, to which sound card output they should be mapped and so on. Together with the equaliser setup menu this menu defines what AlmusVCU actually will do, that is exactly how a block diagram such as in figure 3.1 will look.

```
+--------------------+
| >[1] (A)mbiopole   |
|  [8] (R)everb      |
|  [0] am(B)isonics  |
|  [0] (P)assthru    |
|  Ambisonic setup...|
|  In duplicator...  |
|  Out mixer...      |
|  Speaker layout... |
|  Slave channels... |
|  Mute channels...  |
|  Trim delays...    |
|  In trim volume... |
|  Out trim volume...|
+--------------------+
```

Number of ambiopoles

This is the number of ambiopoles that should be processed, that is how many cross-talk cancellation pairs as seen in the "ambiopole" block in the block diagram in figure3.1. Since cross-talk cancellation is optional for each ambiopole, they can act as regular stereo pairs as well.

The amount can be set to zero, then the whole ambiopole block will be disabled.

Reverb channel amount

This is the amount of channels that will be assigned to the reverb engine, seen as the "reverb engine" block in the block diagram in figure 3.1. The channel amount can be set to zero, then all reverberation functionality will be disabled, equal to removing the whole reverb block in the block diagram.

Since the reverb channels can be mixed to any sound card output channels, the amount does not necessarily need to match the amount of speakers that will be used in the reverb array, although it makes most sense to do so.

Ambisonics channel amount

This specifies how many channels the Ambisonics block seen in the block diagram in figure 3.1 will handle. This can be set to zero, which corresponds to removing the whole ambisonics block from the block diagram.

Ambisonics decoding will only work properly in the playback system if there is one speaker associated to each Ambisonics output channel.

Passthru channel amount

This is the number of passthru channels.

Ambisonic setup

In the ambisonic setup menu it is specified if first or second order Ambisonics decoding should be employed. The setting is ignored if there are no configured ambisonic channels.

First order Ambisonics will activate the input channels iBw, iBx, iBy and iBz, and second order Ambisonics adds the five additional channels iBr, iBs, iBt, iBu and iBv.

```
+--------------------+
| >Format [1st order]|
+--------------------+
```

Input duplicator

The in duplicator menu maps directly to the "programmable input channel duplicator" block found in the block diagram in figure 3.1. In this menu it is specified how the configured amount of ambiopole, reverb, ambisonic and passthru channels should be mapped to the sound card input channels.

As seen in the example below, one sound card input channel can be used more than once, that is be duplicated. Only the channels available in the current configuration will be shown in the menu.

```
+--------------------+
| >get iA1 from [i1] |
|  get iA2 from [i2] |
|  get iRl from [i1] |
|  get iRr from [i2] |
|  get iBw from [i3] |
|  get iBx from [i4] |
|  get iBy from [i5] |
|  get iBz from [i6] |
|  get iP1 from [i1] |
|  get iP2 from [i2] |
+--------------------+
```

Output mixer

The out mixer menu corresponds to the "programmable output channel mixer" block found in figure 3.1. Here each internal ambiopole, reverb, ambisonic and passthru channel is mixed to a corresponding sound card output channel.

As seen in the example below, several of the internal channels can be mixed to a single sound card output channel. Only the channels available through the current configuration will be shown in the menu.

```
+--------------------+
| >mix oA1 to [o1]   |
|  mix oA2 to [o2]   |
|  mix oR1 to [o1]   |
|  mix oR2 to [o2]   |
```

```
|  mix oB1 to [o3]   |
|  mix oB2 to [o4]   |
|  mix oP1 to [o1]   |
|  mix oP2 to [o2]   |
+--------------------+
```

Speaker layout

The speaker layout menu specifies where the speakers are situated relative to the listening position. It is important to give correctly, since it is used in calculation of the actual FIR filters derived from the reverb programs, and in the calculation of gains in the Ambisonics mixer. A specified speaker layout which does not match the reality can result in poor sound.

```
+--------------------+
| >Dist. comp. [off] |
|  Clear layout?     |
|  Apply preset...   |
|  Per speaker...    |
+--------------------+
```

There are a number of presets to choose from, and the speaker positions can be set individually per speaker as well. If the distance compensation option is activated, delay alignment will be performed in order to get the speakers sample aligned in the listening position. Note that the same can be achieved manually by using trim delays.

The clear layout alternative will set the speaker layout to all speakers centered at the listening position (that is angles and distances are set to zero for all speakers).

[FIXME: the apply preset menu is not yet documented, since it will probably change or be removed, it is not very useful in the current implementation].

The per speaker sub-menu looks like this:

```
+--------------------+
| >o1 play sound?    |
|  o1 a[-45.0]deg    |
|  o1 e[30.0]deg     |
|  o1 d[130.0]cm     |
|  .
|  .
```

The speaker layout identifies the sound card output channels, it is thus assumed that there is a speaker attached to each output. The “play sound” function generates a noise from the given channel [FIXME: not implemented], so the speaker easily can be identified. Settings for angle (a), elevation (e) and distance (d) follows. Angle is ranging from +180 to -180 degrees, where 0 is straight ahead, 90 is left and -90 is right. The valid range for elevation is from +90 (straight up) through 0 (in the plane) down to -90 (straight down). The distance is given in centimeters.

Slave channels

AlmusVCU has the concept of master and slave channels for convenient volume management. The internal ambipole, reverb, ambisonic and passthru channels are either associated to the master or the slave group.

In runtime, when the "slave" is muted, all slave channels will be muted. There is also a separate volume control for the slave group. Master volume and mute will instead affect all channels. A typcial use for this feature is to give have an easily adjustable offset volume (slave) for a set of speakers, for example for all reverb speakers in an Ambiophonics system.

Only the channels used will be visible in the menu.

```
+--------------------+
| >oA1 [master]      |
|  oA1 [master]      |
|  oR1 [slave]       |
|  oR2 [slave]       |
+--------------------+
```

Mute channels

The mute channel menu allows muting of individual ambiopole, reverb, ambisonic and passthru channels. Only the channels used will be shown in the menu. In the example below there are two ambiopole channels, two reverb channels and one passthru. In the block diagram in figure 3.1, this functionality belongs to the "(trim) volume/mute" block.

If mute is set to on for a specific channel, it will not produce any output. It will still be active though, so the processing time required for the channel is not altered.

This menu is also accessible in runtime from the channel control menu.

```
+--------------------+
| >oA1 mute [off]    |
|  oA2 mute [off]    |
|  oR1 mute [off]    |
|  oR2 mute [off]    |
|  oP1 mute [on]     |
+--------------------+
```

Trim delay

The trim delay menu allows to set the delay of individual internal channels. Only the channels used will be shown in the menu. The smallest step is 0.01 milliseconds, but the delay will be truncated to fit an exact number of samples for the given sample rate. For example, at 44.1 kHz, the delay 2.72 ms is 119.952 samples which will be truncated to 119 samples.

In the block diagram in figure 3.1, the trim delay functionality is shown in the "trim delay/sample alignment" block.

A typical use for setting individual delays is to compensate for different speaker distances, or to compensate for I/O-delay of another audio processor connected the AlmusVCU machine.

This menu is also accessible in runtime from the channel control menu.

```
+--------------------+
| >oA1 [0.00]ms      |
|  oA2 [0.00]ms      |
|  oR1 [3.14]ms      |
|  oR2 [2.72]ms      |
|  oP1 [0.00]ms      |
+--------------------+
```

Input trim volume

The input trim volume menu allows to set offset volumes of individual ambipole, reverb, ambisonic and passthru input channels. Only the channels used will be shown in the menu. In the block diagram in figure 3.1, this functionality corresponds to the "input trim volume" block.

The intended use for this feature is to compensate for heterogenous signal source levels. These settings are typically tuned once and then not changed.

This menu is also accessible in runtime from the channel control menu.

```
+--------------------+
| >iA1 [+0.0]dB      |
|  iA2 [+0.0]dB      |
|  iRl [-3.5]dB      |
|  iRr [+6.0]dB      |
|  iP1 [+0.0]dB      |
+--------------------+
```

Output trim volume

The output trim volume menu allows to set offset volumes of individual ambipole, reverb, ambisonics and passthru output channels. Only the channels used will be shown in the menu. In the block diagram in figure 3.1, this functionality belongs to "(trim) volume/mute" block.

The intended use for setting individual offset volumes on the outputs is to compensate for heterogenous amplification and speaker sensitivities. These settings are typically tuned once and then not changed.

This menu is also accessible in runtime from the channel control menu.

```
+--------------------+
| >oA1 [+0.0]dB      |
|  oA2 [+0.0]dB      |
|  oR1 [-3.5]dB      |
|  oR2 [+6.0]dB      |
|  oP1 [+0.0]dB      |
+--------------------+
```

3.2.5 Equaliser setup

The equaliser setup menu is used to design and and assign equalisers to channels.

```
+--------------------+
| >Ambiopole eq [off]|
|  Reverb eq [on]    |
|  Ambisonic eq [off]|
|  Passthru eq [off] |
|  Load custom...    |
|  Del custom...     |
|  Eq design...      |
|  Assign eq...      |
|  Eq groups...      |
+--------------------+
```

Enable/disable equalisation

Equalisation can be enabled or disabled for ambiopole, reverb, ambisonics and passthru output channels. Enabling it for reverb or ambisonic channels will add considerable amount of extra processing time required for each reverb/ambisonic channel, even if equalisers are not assigned. For the ambiopole and passthru processing blocks however, enabling it makes no difference in processing requirement, there the actual assigned equalisers makes the difference.

Enabling or disabling equalisation works like a mute setting, it does not alter what equalisers that are loaded or assigned, they will just not be used if equalisation is disabled.

In the block diagram in figure 3.1, ambiopole, reverb ambisonics and passthru equalisation is shown in their own blocks.

Load/delete custom equalisation program

Load and delete custom equalisation programs which are used when designing equalisers. A custom equaliser program is simply a FIR (Finite Impulse Response) filter in the AlmusVCU filter program file format.

Loading and deleting custom equalisation programs work the same as for reverb and crosstalk cancellation programs, and can be read about in section 3.3.

Equaliser design

The equaliser design menu is used to create equalisers, that is FIR filters, that can be assigned to the internal channels. The assignment itself is done in the assign equaliser menu. If there are no equalisers, the will be only one alternative in the menu, `New equaliser?` which will create a new equaliser. This alternative is always found at the bottom of the menu.

Each existing equaliser has three alternatives in the menu, (1) delete the equaliser, (2) enter its filter list menu, and (3) view its properties.

```
|  .
|  .
|  Delete E3?        |
|  E3 filters...     |
|  E3 properties...  |
| >New equaliser?    |
+--------------------+
```

When a new equaliser is created, the add filter menu is automatically opened, since an equaliser need at least one filter.

```
+--------------------+
|  Lowpass...        |
|  Highpass...       |
|  Bandpass...       |
|  Bandstop...       |
|  Allpass...        |
|  Random phase...   |
| >Custom...         |
|  Dynamic...        |
+--------------------+
```

In that menu, there is a set of different standard filter types to choose from. First there are the sinc-based lowpass, highpass, bandpass and bandstop filters, then there is an all-pass filter, a random phase filter, and "custom", which will pick the filter from a previously loaded custom equaliser program. Last is a dynamic equaliser which can be modified in runtime. For further details on the different filter types, see section 3.4.

The filter list menu will show the current list of filters belonging to the equaliser, and how they are combined to form the resulting filter. In the example below, a lowpass filter with cutoff at 450 Hz is added with a highpass filter with cutoff at 300 Hz, and the result is then convolved with a 10 ms random phase filter. Filter names that do not fit in the display width are scrolled sideways.

Each filter component can be scaled with a specified amount of decibel, and be added, subtracted or convolved with the below component. The list is strictly processed from up and down, thus convolution (can be seen as multiplication) is not necessarily calculated before addition and subtraction.

```
+--------------------+
|  Lowpass 450 Hz (so|ft)...
|  scale [+0.0]dB    |
|  [add] w/below     |
|  Highpass 300 Hz (s|oft)...
|  scale [-1.5]dB    |
|  [convolve] w/below|
|  Random phase 10ms.|..
|  scale [+1.5]dB    |
| >Add a filter...   |
+--------------------+
```

At the bottom of the filter list, the alternative to add another filter is present, which will bring up the add filter menu. If a new filter is created, it will be added last to the list. If a filter in the list is selected, the corresponding filter design menu will be shown, where settings can be modified, or the filter be deleted. Section 3.4 contains information about the filter design menus.

The equaliser properties menu has no settings in it, it simply shows the delay, peak amplification and amplification in ten bands of the equaliser. The delay is calculated by scanning for the largest sample in the filter, or by telling the predefined value if a such exists. The peak amplification value is the largest amplification found in among the frequency bins of the frequency domain transform of the filter.

The ten band frequency response should give a rough view on what the filter looks like, but it is by no means an exact tool, and can sometimes be a bit misleading due to its coarse resolution. The frequencies show the center frequency of the band, and the amplification in dB is the mean over the whole band. The example below shows the properties of a 2 kHz low pass filter.

```
+--------------------+
| Equaliser 1:       |
|  delay: 2.9ms      |
|  peak amp: +0.1dB  |
|   31.5 Hz: +0.0dB  |
|   63.0 Hz: +0.0dB  |
|  125.0 Hz: +0.0dB  |
|  250.0 Hz: -0.1dB  |
|  500.0 Hz: -0.0dB  |
|   1.0 kHz: -0.0dB  |
|   2.0 kHz: -22.7dB |
|   4.0 kHz: -54.4dB |
|   8.0 kHz: -67.5dB |
|  16.0 kHz: -75.3dB |
+--------------------+
```

Assign equalisers

The assign equaliser menu is used to assign the designed equalisers to ambiopole, reverb, ambisonic and passthru channels. Only channels with equalisation enabled are shown in the menu. However, if Ambisonics is activated, the ambisonic input channels will always be available.

It contains one entry for each channel, and when selected, a sub-menu is entered where equalisers are assigned to four groups.

```
+--------------------+
| >oA1 grp 1 [E1]    |
|  oA1 grp 2 [E10]   |
|  oA1 grp 3 [off]   |
|  oA1 grp 4 [off]   |
+--------------------+
```

In runtime, equalisers are assigned to the channels by choosing one of the four groups, not individual equalisers. The reason for grouping together equalisers into groups this way, is to make sure that there is enough processing power available in runtime to accomodate all possible combinations. If groups did not exist it would be necessary to allocate processing time for the most intensive equaliser to all channels. With the group concept, the user can specify a limited amount of combinations.

If there is no interest in changing equalisers in runtime, only one group need to be assigned, which should be the first.

Note that for an ambiopole channel pair which has cross-talk cancellation activated, equalisers must be equal at least in the cancellation range of the associated program, or else the cancellation performance will be degraded.

Equaliser groups

This menu is used to choose which equaliser group that will be used in runtime when the convolver is started. To see and alter which equaliser a specific group matches for a specific channel, the assign equaliser menu is used. If the choice is set to dash ('-') there is no group assigned, thus no equaliser.

Only channels with equalisation enabled are shown in the menu. However, if Ambisonics is activated, the ambisonic input channels will always be available.

This menu is accessible in runtime.

```
+--------------------+
| >A1 eq grp [-]     |
|  A2 eq grp [1]     |
|  R1 eq grp [2]     |
|  R2 eq grp [3]     |
|  P1 eq grp [4]     |
+--------------------+
```

3.2.6 Ambiopole setup

In the ambiopole setup menu, cross-talk cancellation programs can be loaded and deleted, and cross-talk cancellation can be activated/deactived for each ambiopole (if there are any).

If cross-talk cancellation is not activated, the given ambiopole will work just as a stereo pair, with straight passthrough from the left/right input signal. If it instead is activated, a cross-talk cancellation program must be loaded. Loading and deleting cross-talk cancellation programs work the same as for reverb and custom equalisation programs, and can be read about in section 3.3.

Under each ambiopole cross-talk cancellation activation alternative, there is a submenu (`Select xtc prg...`) which allows to select which cross-talk cancellation program to assign to the ambiopole. Cross-talk cancellation programs can also be changed in runtime, allowing A/B-testing.

```
+--------------------+
| >Load xtc prg...   |
|  Del xtc prg...    |
|  A-pole 1 xtc [on] |
|  Select xtc prg... |
|  A-pole 2 xtc [off]|
|  Select xtc prg... |
+--------------------+
```

The cross-talk cancellation program defines how an ambiopole will behave. In section 4.7 it is described how to physically set up an ambiopole to match standard cross-talk cancellation programs.

3.2.7 Convolver setup

In the convolver setup menu, the I/O-delay of the convolver is decided. I/O-delay is the time from when input comes into the AlmusVCU until it produces output.

In practice this is the time from pressing "play" on the remote control, until sound comes out from the speakers (delay can also exist in other places of the system though). Changes in volume and similar will also lag as long as the I/O-delay is.

The I/O-delay produced by the convolver which is tuned here, is normally the largest part of the total I/O-delay, where the additional components are possible filter program I/O-delay, and any trim delay settings made by the user.

The goal is to have as short convolver I/O-delay as possible, and the problem is that the shorter the I/O-delay, the more processing power is required (within some limits). The problem gets an extra dimension if the reverb engine is used. When it is used, a trade-off between convolver I/O-delay and reverb time must be made.

The reverb time is simply the length of the reverberation, that is how long time a sudden sound can be heard in the hall before it has faded away completely. The reverb time of a normal concert hall is typically around two seconds. When a reverb program is to be loaded, its reverb time is listed, and if it is longer than the configured reverb time here, the it is truncated to the maximum length allowed.

The task of finding the right I/O-delay is done with help of an automatic benchmark which finds out what the hardware can handle with the current configuration. It exists in one version when the reverb engine is activated which takes reverb time into consideration, and one version used when the task is only to find the shortest I/O-delay possible.

Benchmarking with reverb

To be able to decide which I/O-delay/reverb time alternatives that exist, AlmusVCU runs a benchmark with the current configuration. A benchmark run takes a few minutes. Before it is run, the max load limit should be set. It specifies how many percent of the available processor time that may be used by the convolver. The benchmark is not exact, and a marigin is also needed so the processor is able to handle other tasks. The default value of 85% should be safe, a well-tuned machine can handle a few percent more. If dynamic equalisers will be used and it is important that they are updated quickly on a change, it may be wise to reduce the default value. This will provide more spare processor time to the dynamic equaliser redesign process.

There is also a setting if low latency alternatives should be tried or not. If this is activated, the benchmarking will start trying out with I/O-delays as low as 128 samples (3 ms in 44.1 kHz), without this activated, the start I/O-delay will be about 4096 (93 ms in 44.1 kHz). Only very well-tuned machines can handle low latency configuration in runtime. The sound card hardware can also limit the minimum I/O-delay, if it cannot be configured to generate interrupts at the rate required for low latency configurations.

If, when in runtime, the convolver exists with the failure "buffer underflow", it most likely means that the max load value is set too high, or a low latency setting was used which the machine could not handle. It could also mean that the processor has a thermal throttle and is overheated (if that is the case, the cooling of the computer must be improved).

Note that the benchmark itself will not detect latency problems in the operating system environment, so if there are some, they will only appear in runtime, through the failure "buffer overflow".

```
| >Max load [85]%    |
|  Low latency [off] |
|  Run benchmark?    |
+--------------------+
```

When the benchmark is run, screens with the progress will be shown. These screens exists in four-row versions and two-row versions for small displays.

```
+--------------------+
|Last settings:      |
| D,Rt: 92.9ms,10.2s |
| 44.1 kHz, 121% load|
|Now testing new...  |
+--------------------+
+--------------------+
|Last: 44.1kHz 121%  |
| D,Rt: 92.9ms,10.2s |
+--------------------+
```

The screens show the result of the last test run in a series of test runs that make up the benchmark. The numbers after `D,Rt` is the I/O-delay and reverberation time respectively, in the example above the I/O-delay is 92.9 milliseconds, and the reverb time is 10.2 seconds. However, to achieve this a total of 121% of the specified processor load was needed, so the next run will try a shorter reverberation length.

Benchmarking starts at the lowest I/O-delay (4096 samples normal, 128 samples in low latency mode). The longest reverb time is then found for that I/O-delay, whereafter the I/O-delay is doubled (I/O-delay must be doubled due to the design of the convolver engine) and the maximum reverb time is found for the new I/O-delay, and so it continues. Benchmarking stops when the reverb times are longer than around 20 seconds, or the longest possible reverb time for the given combination of configuration, processor and memory has been found.

When finished, the user can choose between the available alternatives in the convolver setup menu:

```
+--------------------+
|  Max load [85]%    |
|  Low latency [off] |
| >44kHz[370ms,6.5s] |
|  Rerun benchmark?  |
+--------------------+
```

The first number in the I/O-delay/reverb time alternative is the I/O-delay in seconds, and after that the corresponding reverb time in seconds. If AlmusVCU is configured to use several sample rates, all of them get their own I/O-delay/reverb time alternatives.

If AlmusVCU is restarted, only the currently selected alternative will be remembered, so when the convolver setup is entered it will just show the selected alternative. Benchmarking must be run again in order to retreive the old ones. Also, whenever a configuration change which affects performance is done, the I/O-delay/reverb time setting will be dropped, and benchmarking must be run again before starting the convolver.

```
+--------------------+
|  Max load [85]%    |
| >Run benchmark?    |
|  44kHz 370ms,6.5s  |
+--------------------+
```

Benchmarking without reverb

When benchmarking is run on a configuration without reverb channels, it will only search for the lowest possible I/O-delay. Apart from that, it works the same as the case when the reverb engine is used.

3.2.8 System info

The system info screen has no settings in it, just information on the current system.

```
+--------------------+
| Version: 0.7       |
| Uptime: 22days/2h  |
| 44.1kHz I/O-delay: |
|   static: 92.9ms   |
|    4096 samples    |
|   align: 92.9ms    |
|    4096 samples    |
|   total: 185.8ms   |
|    8192 samples    |
| Cache free: 36MB   |
| Filter mem: 50MB   |
| Total mem: 249MB   |
| CPU: AMD Athlon(tm)|
| CPU count: 1       |
| CPU speed: 995MHz  |
+--------------------+
```

The following information is available:

- AlmusVCU version.
- Uptime, that is how long the computer has been up and running (not necessarily running AlmusVCU all the time though).
- I/O-delay for all allowed sample rates. This is only shown when the convolver is ready for runtime. The static part of the delay is the convolver I/O-delay, and the align part is the sample alignment compensation delay, made due to delay in filters used, the total part is the both added together. The numbers are given both in milliseconds and the exact number of samples.
- Cache free. How many megabytes storage space left in the filter program cache on the hard disk.
- Filter memory. How many megabytes memory that has been assigned to store filter programs.

- Total memory. The total amount of memory the machine has.
- CPU. The processor type and model.
- CPU count. The amount of processors. AlmusVCU and its convolver engine BruteFIR supports multiple processors.
- CPU speed. The approximate clock speed of the processor(s).

3.2.9 Maintenance

```
+--------------------+
| >Debug mode [off]  |
|  Reboot machine?   |
|  Shutdown machine? |
+--------------------+
```

The maintenance menu so far allows for rebooting and shutting down the machine, as well as turning on debug mode, which will start various logging helpful in finding a fixing any problem that might occur.

3.3 Filter program management

Filter programs (reverb, cross-talk cancellation and equalisation programs) are loaded from a "read-only" device, which depending on the static configuration could be a CD-ROM or maybe the hard disk.

When the filter program is loaded to memory it is pre-processed to be used with the convolution engine. For reverb programs this typically means that the space occupied in RAM is much larger than on disk.

When a program is loaded, it is also copied to a local cache on the hard disk. The cache is used when AlmusVCU is restarted, all loaded filter programs will the be reloaded from the cache into memory, without the need of providing the original source. A file in the cache will not be flushed until it is full, and then the oldest and not currently used filter program will be flushed first.

A program can be removed from RAM to leave space for other programs. The cached copy will not be removed from the hard disk, so it will be available for reloading. It is however possible to manually remove filter programs from the cache, before they are automatically removed when the cache is filled.

3.3.1 Load filter programs

Below is the menu as shown when a filter program is to be loaded (AlmusVCU has been configured to load from "harddisk").

```
+--------------------+
|Space left: 7.0s    |
| >From harddisk...  |
|  From cache...     |
+--------------------+
```

The "Space left" indicates how many seconds of filter programs of the given type can be stored in RAM before it is full. All filter programs share the same memory, so a loaded reverb program will reduce the space for equalisation and cross-talk cancellation programs and the other way around. Since there is usually many reverb channels, the amount of space left for reverb programs is fewer seconds than for equalisation programs (which is only one channel).

When choosing to load from cache or the read-only device (in the example "hard-disk"), a list of the available files are given, simply stating the name of each filter program, followed by the length in seconds within parentheses. If a name does not fit in the 20 column space of the user interface, it will automatically side-scroll to show one part of the name at a time.

3.3.2 Delete and list programs

The following menu is shown when a filter program is to be deleted. It can be deleted either from RAM or cache. When one of the alternatives is chosen, a list of currently available programs is shown, and if any of them is selected (by pressing RIGHT) it is deleted.

```
+--------------------+
|Delete program      |
| >From memory...    |
|  From cache...     |
+--------------------+
```

Since the delete filter program menu shows all available programs in memory or in cache, it also functions as a "list all programs" menu.

3.4 Built-in filter designer

The built-in filter designer is accessed from the equaliser design menu, described in section 3.2.5. It is used for creating equalisers that can be assigned to the any of the internal channels in AlmusVCU.

The filter types that can be designed with the built-in filter designer are lowpass, highpass, bandpass, bandstop all pass and random phase filters. Additionally there is a special case, the dynamic equaliser, which can be modified in runtime, but not be combined with any other filters.

3.4.1 Low/high/bandpass and bandstop

The low/high/bandpass and bandstop filters are designed using Kaiser-windowed sinc signals. This design method gives the filters certain properties:

- Flat passband, that is there is no ripple. This is a property achieved by using the Kaiser window.
- Linear phase. This equals a filter which does not alter phase, but introduces some delay.

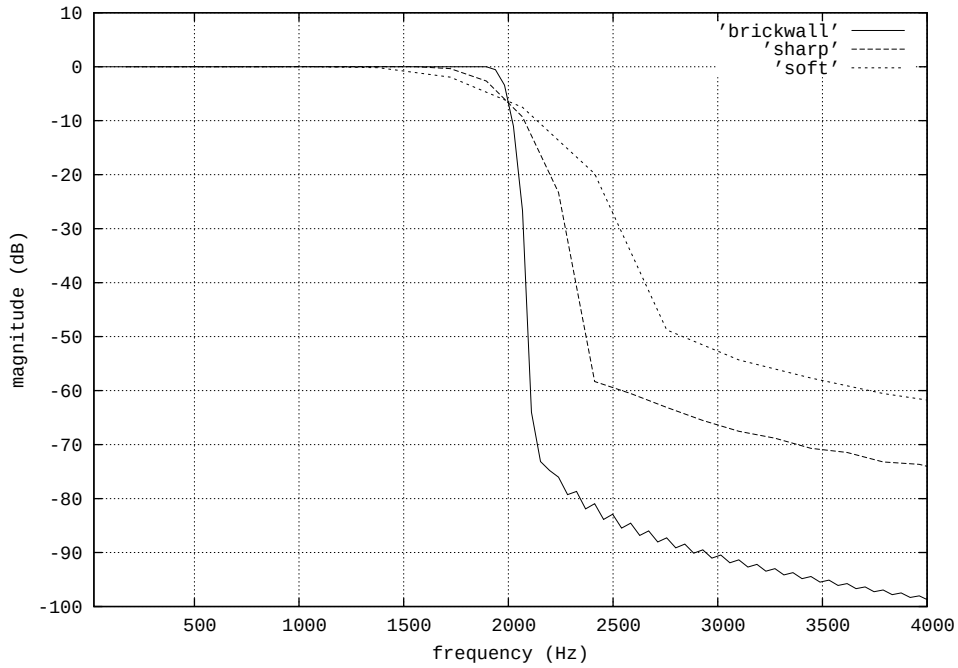

Figure 3.2: 2000 Hz lowpass filter with three different slopes

Slope

Cutoff frequencies and a rough slope indication is given by the user. The slope is how fast the filter should attenuate after the cutoff frequency, and can be "soft", "sharp" or "brickwall". These are quite rough specifications, and are not to be seen as exact dB/octave slopes. Figure 3.2 shows a 2000 Hz lowpass filter with the three different slopes.

The bandpass filter consists of a low and and a highpass filter convolved together, and the bandstop are low and high pass filters added together. The reason for employing this method instead of creating the sincs directly, is to get the same slope at both low and high cutoff frequency.

Delay and processing requirement

As said, the filters are linear phase. This means that they introduce some delay, and the longer the filter is, the longer the delay, which is exactly half the length of the filter. The user does not specify the filter length directly, it is instead calculated from the given slope and cutoff frequencies. A sharper slope, means that a longer filter is required. Also, the lower the (low) cutoff frequency, the longer the filter must be. For example, a 200 Hz brickwall lowpass filter is about 190 ms long at 44.1 kHz sample rate, while a 2 kHz soft sloping filter is only 3 ms, so the differences can be huge.

The length of the filter does not only increase delay, it also increases the processing time requirement. Thus it is good to use soft sloping filters whenever possible, to reduce delay and processing time requirement.

Note that AlmusVCU automatically compensates for the delay in the filters.

Design menu

The design menu for the filters are invoked from the equaliser design menu, described in section 3.2.5. The example below shows the bandpass design menu.

```
+--------------------+
|Bandpass settings:  |
| >Cutoff 1 [2000]Hz |
|  Cutoff 2 [3000]Hz |
|  Slope [soft]      |
|  Create filter?    |
+--------------------+
```

A filter will only be created if the `Create filter?` alternative is selected. If the menu is accessed from an equaliser filter list menu, the create filter alternative is replaced with `Delete filter?` which will delete the filter. If the menu is left the normal way (by pressing LEFT), the filter design is aborted, or the current settings accepted if the design menu is accessed from an equaliser filter list menu. Cutoff 1 and 2 is the low and high cutoff frequency respectively. The valid range for each cutoff frequency is 5 to 20000 Hz

The design menus for the other filter types in this class work the same as this one, and is therefore not further described here.

3.4.2 All pass

The all pass filter is simply a zero delay dirac pulse, and will not modify the signal in any way, if it is not scaled, and then of course only the volume is changed.

The filter may be useful in combination with other filters, for example to create shelf filters. Section 3.2.5 describes how different filter types can be combined.

3.4.3 Random phase

The random phase filter is a white noise filter, that is an all-pass filter (no change in magnitude), but the phase (delay) for every frequency is random. The filter is not minimum phase, that is each frequency can be delayed more than one period. No frequency can be delayed longer than the length of the filter though, which is specified by the user.

The filter is treated as a zero-delay filter, thus there is no delay compensation made for it (it is not feasible or meaningful to do when the phase for all frequencies are different and random).

An important property of the generated filters is that if several are generated, they will all be decorrelated, that is the phase is randomised differently for each filter.

An equaliser that includes a random phase filter may look a bit chaotic in the equaliser properties screen.

Design menu

The design menu for the random phase filter are invoked from the equaliser design menu, described in section 3.2.5.

```
+--------------------+
|Random phase:       |
| >Length [10]ms     |
|  Create filter?    |
+--------------------+
```

A filter will only be created if the `Create filter?` alternative is selected. The length is specified in milliseconds. The valid range is 1 - 10000 milliseconds.

3.4.4 Dynamic equaliser

The dynamic equaliser is a special case, in the way that it cannot be combined with any other filter, and that the filter design can be modified in runtime. It is a linear phase equaliser with configurable bands, ISO octave bands existing as presets.

Since a dynamic equaliser is linear phase, it will introduce some delay (which AlmusVCU automatically compensates for). The delay is exactly half the filter length, and the filter length is indirectly specified by one of the following parameters:

- Lowest frequency band center frequency. The lower the frequency, the longer the filter.
- Convolver I/O-delay. The filter can never be shorter than the I/O-delay.
- Minimum center frequency distance. The smaller the distance, the longer the filter.

The parameter that will yield the longest filter in the current configuration will decide the filter length. The equaliser properties menu described in section 3.2.5 will display what the actual delay is for the filter.

The dynamic equaliser can be used to design magnitude-shaped linear phase filters even if it is not interesting that it can be changed in runtime, since there is no extra processing cost compared to an ordinary static filter.

Design menu

The choice of frequency bands is static, and cannot be changed in runtime. There are two ISO octave bands presets, ISO whole octave bands (10 bands), or ISO 1/3 octave bands (31 bands).

```
+--------------------+
|  ISO oct bands?    |
|  ISO 1/3 oct bands?|
|  Custom bands...   |
+--------------------+
```

There is also the alternative to choose a custom frequency bands. Choosing that alternative will display the menu below. There must be at least two center frequencies specified, and more can be added.

```
+--------------------+
|  Band1 [200]Hz     |
|  Band2 [2500]Hz    |
|  Add band?         |
|  Accept current?   |
+--------------------+
```

If the `Accept current?` alternative is selected, the magnitude response menu will be entered, which also is accessible in runtime. Below is an example of how it looks for an ISO octave filter:

```
+--------------------+
|  31.5Hz [-5.0]dB   |
|  63.0Hz [+0.0]dB   |
|  125.0Hz [+3.4]dB  |
|  250.0Hz [+0.0]dB  |
|  500.0Hz [+0.0]dB  |
|  1000Hz [+20.0]dB  |
|  2000Hz [-20.0]dB  |
|  4000Hz [+10.4]dB  |
|  8000Hz [+0.0]dB   |
|  16000Hz [-1.0]dB  |
+--------------------+
```

The valid decibel range for each frequency is from -20.0 dB to +20.0 dB. The edge frequencies are set to the same as the edge center frequencies. That is, in the ISO octave filter example above, the amplification at 0 Hz will be -5.0 dB (same as at 31.5 Hz), and at the maximum frequency it will be -1.0 dB (same as at 16000 Hz). Figure 3.3 shows the magnitude response of the given example.

3.5 Runtime main mode

The runtime main mode is the first shown when the convolver is started. It exists in a main four-line version, and in a two-line version for small displays.

```
+--------------------+
|    BERWALD HALL    |
| [pick reverb prog] |
|peak-144.5 mvol-12.0|
|44.1kHz    svol-10.0|
+--------------------+

+--------------------+
|    BERWALD HALL    |
|p-144.5m-12.0 s-10.0|
+--------------------+
```

It shows the peak volume, the master volume and the slave volume in decibel (slave volume is only displayed if there are any slave channels). At the top is the short name of the currently loaded reverb program ("BERWALD HALL" in this example), or simply the text "AlmusVCU" if the reverb engine is not used.

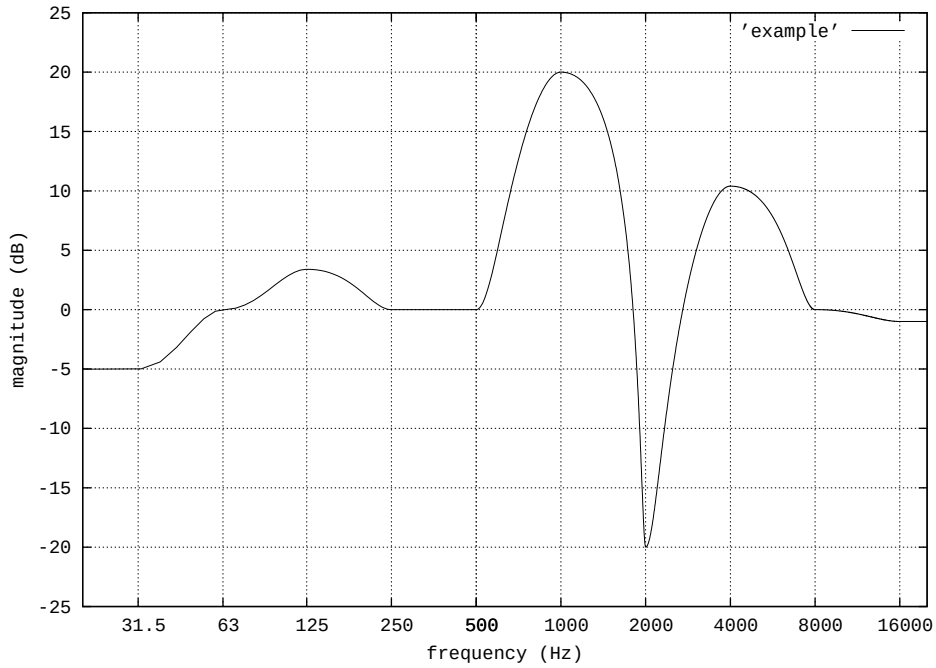

Figure 3.3: An ISO octave filter designed with the dynamic equaliser

The peak volume is the currently highest volume on any sound card output (speaker feed) so far, since the main screen was entered, or since it was reset through a change in volume. If the peak is a positive value, it means that the signal has clipped with the given amount of decibel, thus is 0.0 dB the maximum value that can be represented.

The master volume affects all channels, the range is from -48.0 to +48.0 dB. The level 0.0 means that the volume is not altered from the input, apart from amplification/attenuation caused by filter programs, and possible mixing and trim volume settings.

The slave volume is relative to the master volume, and affects all slave channels. The range is -48.0 to +48.0 dB.

If LEFT is pressed, all channels are muted, and the uppermost line in the display is replaced with the text "All channels muted". If LEFT is pressed again the channels are unmuted.

The RIGHT button is used to cycle through the four select modes: "pick reverb program", "alter master volume", "alter slave volume" and "select menu". The "pick reverb program" mode is only available if the reverb engine is active, and the "alter slave volume" mode is not available if there are no slave channels.

If in the "select menu" mode, and UP or DOWN has been used, the RIGHT button is used for changing to the selected menu. The buttons UP and DOWN are used to operate the current select mode: cycle through reverb programs forward/backward, increase/decrease master volume, increase/decrease slave volume, cycle trough menus forward/backward respectively.

The three menus to select are "channel control", "main mode" (this mode), and last "setup", which will exit back to to the setup menu.

In the two-line version of the main screen, the current select mode is indicated with what part of the screen that is in uppercase letters.

3.6 Runtime channel control

The runtime channel control menu presents all settings that can be changed in runtime. Only the relevant entries will be shown, for example if no passthrus have been configured, there will be no passthru menu entries. The menus “Mute channels”, “Eq groups”, “Trim delays” and “Trim volumes” are the same as found in the channel setup menu, and is documented in section 3.2.4.

```
+--------------------+
| >Master [-12.0]dB  |
|  Slave [-10.0]dB   |
|  Master mute [off] |
|  Slave mute [off]  |
|  A-pole mute [off] |
|  Reverb mute [off] |
|  A-sonic mute [off]|
|  P-thru mute [off] |
|  Volume step [3.0] |
|  Master eq grp [-] |
|  A-pole eq grp [-] |
|  Reverb eq grp [-] |
|  A-sonic eq grp [-]|
|  P-thru eq grp [-] |
|  Eq groups...      |
|  Dynamic eq...     |
|  Reverb setup...   |
|  A-pole setup...   |
|  Mute channels...  |
|  Trim delay...     |
|  In trim volume... |
|  Out trim volume...|
+--------------------+
```

The master and slave volume setting is the same as accessed in the runtime main mode. The mute settings for the groups master (all channels), slave, ambiopole, reverb, ambisonic and passthru work independently from the individual mute settings found in the “Mute channels” menu.

The volume step setting defines how much the volumes should be altered in the main mode when UP or DOWN is pressed for altering slave or master volume.

The equaliser group settings for master, ambiopole, reverb, ambisonic and passthru, changes the equaliser group selection for all channels within the given group. The master setting will override all others, it will actually change all other settings to the same as it is set to. For individual channel equaliser group control, the “Eq groups” menu is used. Any equaliser group settings made in the channel control menu will be transferred to the “Eq groups” menu.

3.6.1 Dynamic equaliser

The dynamic equaliser menu will lead to the design menu for the active dynamic equalisers, where the magnitude response can be changed in real-time. The response will be changed while altering the delay, however since it is a quite time-consuming task to redesign the filter, the change will be delayed somewhat. The I/O-delay of the system will of course be added to the redesign delay.

The redesign is done in the background with the spare processor time. Thus, to increase the redesign performance, a less demanding convolver configuration can be selected, so there is more spare processor time left.

3.6.2 Reverb setup

The reverb setup menu is used to tune settings for each loaded reverb program. The name of the current reverb program is shown at the top (in this example "The Berwald Hall"). The entry `select next prg?` will select the next reverb program from the loaded, and present its settings in this menu. It is only present if more than one reverb program has been loaded.

```
+--------------------+
|The Berwald Hall    |
| >Select next prg?  |
|  Delay [45.0]ms    |
|  Trim vol [+0.0]dB |
|  Reset delay?      |
+--------------------+
```

The delay setting is how many milliseconds the reverb program is delayed. A reverb program does not store any predelay in its filters, instead they have a recommended delay value which is the default here, and is restored by `Reset delay?`. For a concert hall reverb program the delay corresponds to the time it takes for sound to travel from the stage, to wall or ceiling and then to the listener's seat. A delay of 45 milliseconds thus corresponds to about 15 meters distance (0.045 times 343 which is the speed of sound in meters per second).

If the reverb program seems to be too loud or too silent compared to the other reverb programs, its volume can be adjusted with the trim volume setting.

The delay and trim volume setting is stored per reverb program.

3.6.3 Ambiopole setup

The ambiopole setup menu is only available if there is one ambiopole or more configured. If there is more than one ambiopole, there is first a menu where the ambiopole to configure is selected.

```
+--------------------+
| >Ambiopole 1...    |
|  Ambiopole 2...    |
+--------------------+
```

Currently, the only operation that can be performed in this menu is cycling through the cross-talk cancellation programs for the selected ambiopole.

```
+--------------------+
|XTC Basic           |
| >Select next prg?  |
+--------------------+
```

The heading shows the name of the currently loaded cross-talk cancellation program, and by selecting `Select next prg?` the next program will be selected.

If there is only one cross-talk cancellation program loaded, or the selected ambiopole does not have any cross-talk cancellation activated, a message saying that there is nothing to configure will appear.

3.7 Trouble shooting

3.7.1 Debug logs

A common problem is that the convolver refuses to start due to a mistake made during installation or some other problem. This is hard to debug from within AlmusVCU through its sparse text interface, but by activating the debug logging, the problem will be easier to find. Debug logging is activated in the maintenance menu. A description of the debug logs and what they contain can be found in section 8.3.

Chapter 4

Ambiophonics tutorial

4.1 Introduction

This chapter describes how to set up a typical 10 channel Ambiophonics system built around AlmusVCU. Since both AlmusVCU and Ambiophonics is very flexible in how it is configured, this should only be seen as an example of what a system can look like.

For the user who wants a basic but full-featured Ambiophonics system, the system presented here is a suitable example.

It is assumed that the reader knows what Ambiophonics is and what the components are, and thus the information here concerning Ambiophonics technology is rather brief and concentrated. More detailed information can be found at:

`www.ambiophonics.org`

4.2 System design tips

4.2.1 The listening room

The room should be acoustically dead, and silent. For Ambiophonics, the goal is to remove the acoustics of the listening room and replace it with the acoustics reproduced by the reverb array. This means that the only room treatment needed is absorption. Diffusion which is common on the back wall for ordinary stereo systems, is better replaced with absorption. In theory, the best room would be anechoic, but that can only be achieved in purpose-built laboratories.

Fortunately, one gets very far even with limited absorption. The reason for this is that the reverberation on a music recording is most often much longer than for an ordinary living room, thus the acoustics of the living room will be masked. For successful masking, the recorded reverberation should be about five times longer than for the listening room. This means that the reverb part of Ambiophonics works very well most of the time even in an untreated room.

However, for high performance, it is still recommended to significantly lower the reverb time of the listening room by means of absorption. Making your own absorption panels out of fiber glass or similar is the lowest cost solution. If you do not want to make your own, buying absorption panels meant for general acoustic

treatment in public rooms is usually much cheaper than panels targeted for the recording studio market, although they usually are the ones that look the best.

The bass frequency range is always very difficult to absorb. So called "bass traps" helps some, but does not solve the problem as good as normal acoustic treatment does in higher frequency ranges. In the bass, standing waves is the largest problem. With a large room this problem is minimised, but that is seldom available, and then one may need to resort to digital room equalisation. Currently, the AlmusVCU allows for some minimum room compensation features, which hopefully will be extended in the future.

Room absorption is not only needed for lowering the reverberation time of the room, it is also needed to remove harmful early reflections from the ambiopole. Sound radiated from the ambiopole which is reflected from the walls, ceiling and floor in front of the listener will diffuse the sound stage, and reduce, or even ruin the cross-talk cancellation. If you plan to use only minimal sound absorption, at least put a thick carpet on the floor between the listening seat and the ambiopole, and absorptive panels at the "mirror points" on the walls and ceiling. The mirror points are the places where if you put a mirror there will see the ambiopole from the listening position.

Ambiophonics does not actually require room treatment any more than ordinary stereo does (although it could be argued that the ambiopole is slightly more sensitive to early reflections than the stereo triangle). However, Ambiophonics is about realistic sound reproduction, and to achieve that proper room treatment is important. In a really low cost or casual setup, it could be skipped though, just like for ordinary stereo. In such cases it is good to at least have some soft furniture in the room.

The room should also be silent. Any audible noise in the room will harm the feel of realism. For really high end installations, all sound equipment except the speakers should be put in a neighbouring room, since anything with AC input usually produces some noise. There is quite much noise on recordings as well though, since microphones and recording equipment is not perfect, and that noise can mask minor background noise in the room.

4.2.2 The ambiopole

The ambiopole simply consists of a pair of regular loudspeakers normally used for stereo, which are carefully placed to fit the cross-talk cancellation program used.

In theory, the best ambiopole speakers are point sources, meaning small two-way speakers. These are a good choice for low cost systems, but to achieve full dynamics and frequency range of, for example a symphony orchestra, larger speakers is necessary.

- The most expensive components of a well-balanced Ambiophonics system should be the ambiopole speakers. Note that it is more demanding for a speaker to be used in an ambiopole with a cross-talk cancellation program than in a regular stereo triangle.
- Both speakers must have equal phase and amplitude response. Fortunately, only very poor speakers have significant differences in amplitude response. However, some special speaker types instead have random phase response, and those will not work with a cross-talk cancellation program.

- Very wide speakers will probably not work well, for example wide planar speakers.
- The bass should be powerful, that is provide good dynamics. Simply put, it should be able to play loud without sounding bad. The reason for this is that a cross-talk cancellation program will put quite much extra energy into the bass, which will be cancelled out in the air.
- Always test before you buy. [FIXME: provide pre-processed test CD image on the web].

4.2.3 The reverb array

The reverb array is the place to save money in your Ambiophonics system. You can employ much lower quality components in the reverb array than the ones found in the ambiopole, without compromising the overall system sound quality. You can even have different speakers and amplifiers within the reverb array, as long as they can be tuned to the same volume reasonable well.

The secret is that the sole purpose of the reverb array is to reproduce a diffuse sound field caused by hall reflections of sound produced on the stage. There is no direct sound reproduced, and therefore we can do several compromises without impacting overall sound quality.

- Low frequencies can be reproduced by the ambiopole instead, since they are hard to localise anyway. In small rooms, room mode problems will be exaggerated if the reverb array reproduces low frequencies (without room equalisation). AlmusVCU allows for applying a high pass filter to avoid this problem.
- Most concert halls start to roll off at around 3 kHz, that is high frequencies are absorbed by the walls and air. This means that there is not much high frequency information in the reverb sound field, and therefore the reverb array does not need to be good at reproducing that.
- Correctness in phase response is meaningless, because we want to have a diffuse sound field, and one property for that is a randomised phase response.
- Minor errors in magnitude response ("frequency response") of the speakers are not harmful. These will only correspond to a slight change in how the original concert hall behaved acoustically, for example if there was a curtain on the wall or not, or if there was a large audience in the hall or not.
- It is better to have many lower quality speakers than few higher quality speakers. A minimum amount for better-than-conventional-surround results is four speakers, eight is recommended for a full-fledged system, but it should not be interpreted as an upper limit. The speakers should be placed at the sides of the room, in the back, over the head, and in the front, in that order of importance. With only four speakers, the sides and back can be covered. Of course, experimentation is good.
- The amplification should be low noise. One problem with having many speakers is that if the amplification is noisy, it will be even more evident. Since the amplifiers must be set to full output per default (the volume is controlled by the level of the output signal from AlmusVCU), there can be a problem with excessive background noise if the amplifiers are of low quality. [FIXME: analog coil filter between amplifier and speaker to reduce noise at the expense of high frequencies, which are not important anyway?].

- [FIXME: how important is good dynamic range? Can one let the ambiopole dominate at high sound levels?]

4.3 Connecting all components

Although the first thing to do probably will not be to connect all components together, knowing what is needed is necessary. Figure 4.1 shows AlmusVCU in the typical Ambiophonics setup with an ambiopole and eight reverb channels.

The reason for using S/PDIF for the ambiopole and ADAT for the reverb array as in the example, is simply to give the possibility to differ in quality. It is wise to put most money into the ambiopole. In this case one can assign a high quality S/PDIF D/A converter for the ambiopole, and a low-cost ADAT D/A converter for the reverb array.

4.4 Installing AlmusVCU

In this tutorial it is assumed that the sound card hardware in the Almus VCU machine is an RME Audio Hammerfall sound card, which provides both ADAT and S/PDIF input/output simultaneously.

Information on which hardware to choose and how to install the software is found in chapter 2. For the configuration given in this tutorial, eight reverb channels with an ambiopole, at least a Pentium III at 1 GHz or corresponding is recommended.

4.5 Configuring AlmusVCU

4.5.1 Static configuration

This tutorial assumes that an RME Audio Hammerfall sound card is used, and that it is configured in the way described in chapter 2.

4.5.2 Setup menu

After the static configuration has been completed, and AlmusVCU is started for the first time, the setup menu is entered:

```
+--------------------+
| >Start convolver?  |
|  Load reverb prg...|
|  Del reverb prg... |
|  Signal setup...   |
|  Channel setup...  |
|  Eq setup...       |
|  Ambiopole setup...|
|  Convolver setup...|
|  System info...    |
|  Maintenance...    |
|  Screen dimmer [2] |
|  Autostart [off]   |
+--------------------+
```

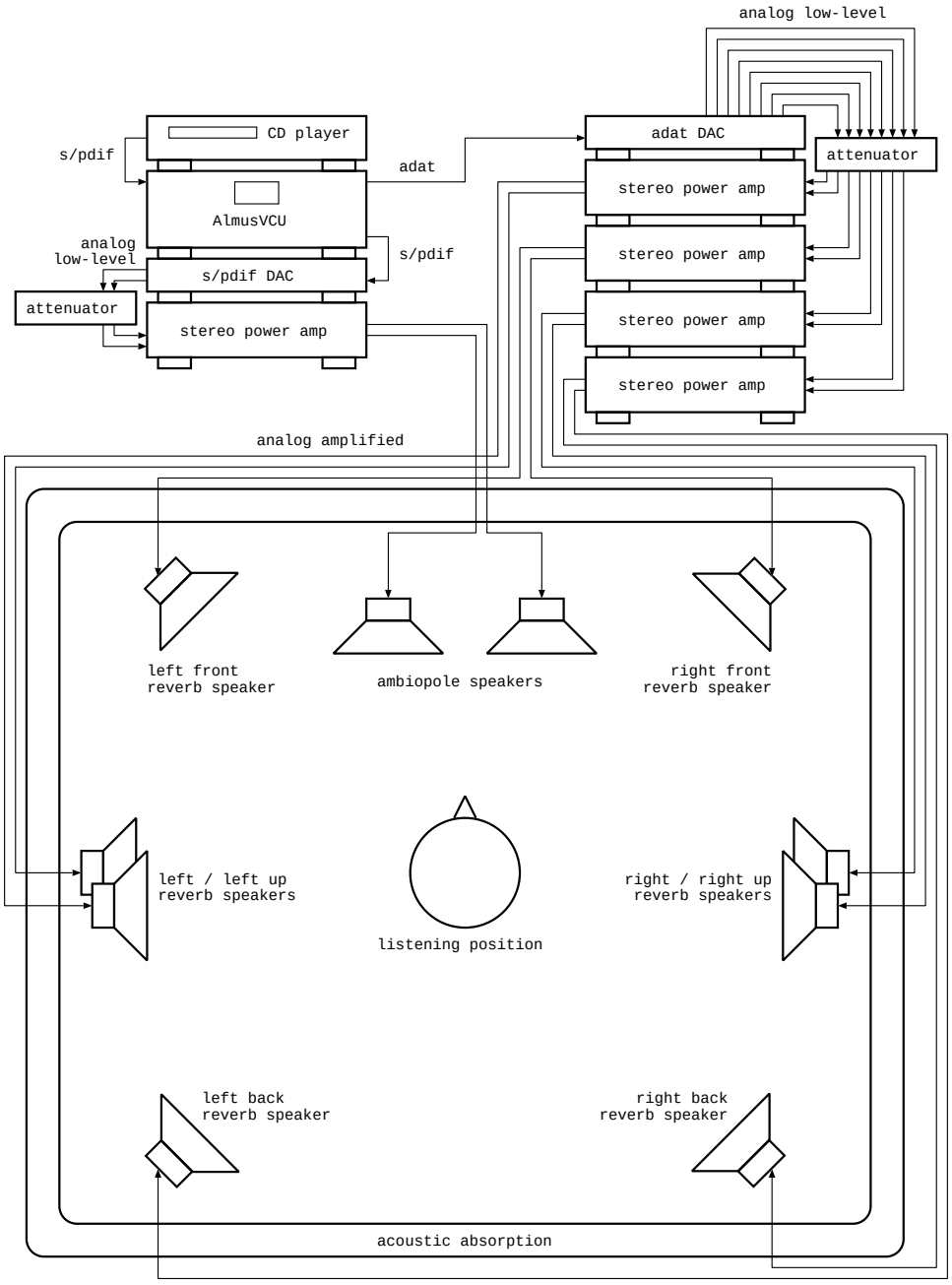

Figure 4.1: a typical Ambiophonics system, consisting of an ambiopole and eight reverb channels.

Information on how to navigate the menu system is found in section 3.1.

Signal setup

First skip down to the "Signal setup" menu and enter it:

```
+--------------------+
| >Soft clip [off]   |
|  Dither [on]       |
|  [24] bit output   |
|  Allow 44.1kHz[on] |
|  Allow 48.0kHz[off]|
|  Suggest [44.1]kHz |
+--------------------+
```

Assuming that a CD player is the signal source, these default settings are exactly what is needed, so nothing need to be changed. It is good to check them anyway, since defaults may change faster than this manual does.

Channel setup

Next, back up to the setup menu, and enter the "Channel setup" menu:

```
+--------------------+
| >[1] (A)mbiopole   |
|  [8] (R)everb      |
|  [0] am(B)isonics  |
|  [0] (P)assthru    |
|  Ambisonic setup...|
|  In duplicator...  |
|  Out mixer...      |
|  Speaker layout... |
|  Slave channels... |
|  Mute channels...  |
|  Trim delays...    |
|  In trim volume... |
|  Out trim volume...|
+--------------------+
```

The amount of reverb channels should be set to eight, and there should be one ambiopole. Passthru and ambisonic channel amount should be set to zero.

The input duplicator menu should look like this:

```
+--------------------+
| >get iA1 from [i1] |
|  get iA2 from [i2] |
|  get iRl from [i1] |
|  get iRr from [i2] |
+--------------------+
```

The default settings should be ok. Both inputs for the ambiopole and the reverb engine should point at i1 and i2 that is the S/PDIF input (see static configuration, section 5.5.1). The inputs should be connected directly to the CD player.

In the output mixer menu, the outputs for the ambiopole should be set to o1 and o2, that is the S/PDIF output. The reverb channels should be put to their own outputs in ADAT 1, that is o3 - o10. The output mixer menu should look like this when left:

```
+--------------------+
| >Mix oA1 to [o1]   |
|  Mix oA2 to [o2]   |
|  Mix oR1 to [o3]   |
|  Mix oR2 to [o4]   |
|  Mix oR3 to [o5]   |
|  Mix oR4 to [o6]   |
|  Mix oR5 to [o7]   |
|  Mix oR6 to [o8]   |
|  Mix oR7 to [o9]   |
|  Mix oR8 to [o10]  |
+--------------------+
```

The speaker layout (`Speaker layout`) should be set to exactly match the actual layout of the speakers. If the speakers has not been put in place yet, this can wait until later.

In the slave setup menu, the ambiopole outputs should be associated to master, and the reverb channels to slave. This way it will be possible to adjust the reverb volume relative to the ambiopole in runtime in a convenient way.

If there are different speaker or amplifier types in the reverb array, output trim volumes may need to be adjusted to match the given setup. This is however better left until later, when running with sound.

Ambiopole setup

Next leave the channel setup menu, skip equalisation setup (`Eq setup`) and enter the ambiopole setup menu. Activate cross-talk cancellation, and load a cross-talk cancellation program. However, if an ordinary stereo triangle is to be used instead of the ambiopole, keep the cross-talk cancellation de-activated and do not load any cross-talk cancellation program.

Convolver setup

Finally, the convolver setup menu is entered, and there the benchmark is run. If the computer is well-performing latency-wise, the `Max load` setting could be increased to 90% from the default conservative 85%. The low latency setting is usually not useful when the reverb engine is active, since it will require an extremely fast computer to get long enough reverb times.

After the benchmark has been run, pick an alternative which has at least 3.0 seconds reverb length. It could look like this before the menu is left:

```
+--------------------+
|  Max load [90]%    |
|  Low latency [off] |
| >44.1 [186ms,4.6s] |
|  Rerun benchmark?  |
+--------------------+
```

Load reverb program

Before being able to start the convolver, a reverb program must be loaded. This is done in the `Load reverb prg` menu.

Starting the convolver

Now the system should be ready to start processing, and this is done by selecting the alternative `Start convolver?` in the setup menu. To avoid nasty surprises, one can start with playing some soft music, and have the amplifier volume controls turned down (if there are any).

Before the sound in the system will be any good, the reverb array and ambiopole must be properly set up, as described in the following sections.

4.6 Setting up the reverb array

4.6.1 Speaker layout

The reverb speakers should be placed as far away as possible (up to a practical limit) from the listening position, this way a large sweet area with realistic reverberation will be obtained. In small spaces where long distances to the reverb speakers are not possible, the reverberation will only sound at its best in the listening position.

The minimum distance possible from the reverb speaker depends on the speaker model, but a general rule is to try to keep it no less than 1.5 meters. If the speakers are too close, the reverb array is heard as a set of individual speakers rather than believable reverbation.

With the suggested 8 reverb speakers, it is recommended to place one pair to the sides (+/- 90 degrees), one pair in the back (+/- 135 degrees) one pair at the sides and a bit overhead, and the last pair in front (+/- 45 degrees). The suggested directions are only approximate and may be modified after some experimentation. Here are two basic rules to follow though, concerning symmetry:

- Keep the array symmetric, that is the layout of the left side should be the same as the right side.
- Think in pairs, and have them equal. This means if there are multiple speaker models in the reverb array, the should be matched pair-wise, that is the side pair should be the same model, the front pair the same etc.

In theory there is no need to have the array symmetric as suggested here, but in practice it is important in order to avoid a reverberation that sounds directional (that is the sounds comes from a specific direction).

Suggested layouts for fewer speakers

Theoretically the front speakers are the least important (the ambiopole is there), but in practice they may be important to provide a good blend between sound from the reverb array and the sound originating from the ambiopole, as well as to provide a good front-back balance. If there is less than eight reverb speakers, the overhead or front pair should be dropped first, followed by the front/overhead, then the back pair and last the side pair. Common sense might say that the back pair should be more important than the side pair, but psycho-acoustic research has shown that speakers placed at the sides provide better envelopment.

[FIXME: to provide better advice for few-speaker layouts than given here, some experiments need to be conducted.]

4.6.2 Listening to the reverb array

It is very important that the speaker layout AlmusVCU knows about is the same as the actual physical layout. This is because the reverb programs are processed to fit that layout, and if the one programmed into AlmusVCU does not match reality, it may result in poor sound. However, the directions of the reverb speakers need not to match exactly to the degree (+/- 5 degrees is perfectly ok), and the speaker distance information is not used (by the reverb engine that is). This may change in the future however, so it is a good habit to try to match the AlmusVCU speaker layout with reality as exactly as possible.

When the speaker layout has been correctly configured in AlmusVCU, it is a good idea to listen to music only through the reverb array to see if there are any remaining problems. A good reverberation should be diffuse. This means that it should not be possible to hear a direction from where the sound is coming, it should come from all directions at the same time. If the sound field does not seem to be diffuse, it is likely due to irregularities in the speaker array. Perhaps one speaker is closer and therefore sound louder, or is amplified more compared to the others.

If the front speaker pair is missing, the reverberation will probably sound as a bit heavy to the back, but combined with the ambiopole, that problem is non-existent.

Detecting a poor reverb program

A poor reverb program can cause some unwanted phase effects. Hopefully poor reverb programs will not be common, but it may be good to know how to recognise them anyway, if you have made an own reverb program for example.

To verify that the reverb program is of good quality, try to listen at one pair of the reverb array at a time. There should not be any stereo imaging between the speaker pair, the sound should be totally diffuse. When moving the head a bit from side to side, the sound should continue to be diffuse. However, as mentioned earlier, if the speakers are really close, a speaker may break through as a clear individual sound source when moving the head closer to it even when seated in the listening position. Also, if the sound seems to be very close to the head, there is probably some phase problem with the reverb program.

4.6.3 Room equalisation

If the reverb array is placed in a small room and the speakers are capable of reproducing bass, there may be a problem of excitation of room modes, that is it simply

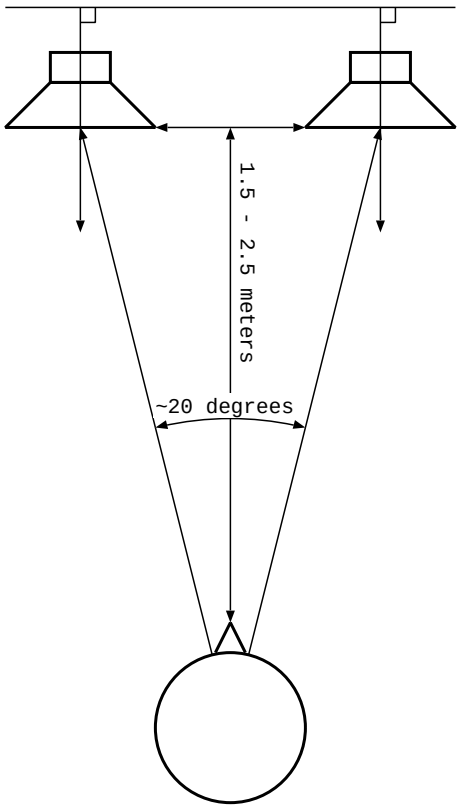

Figure 4.2: ambiopole layout

becomes too much bass in the room.

This can be solved by letting only the ambiopole reproduce bass, and simply high-pass filter all reverb channels. The cut-off frequency with a soft roll-off is suitably put around 100 - 200 Hz (near the Schroeder frequency for the room). The AlmusVCU equalisation capabilities can be used to achieve this:

1. Enter the setup menu, and in that select `Eq setup`.
2. Turn on reverb equalisation (`Reverb eq`).
3. Design an equaliser (`Eq design`), containing a high-pass filter which is designed using the built-in equaliser designer.
4. Assign the designed equaliser to all reverb channels (`Assign eq`), using the same group (group 1 for example).
5. Choose the given equaliser group (`Eq groups`).

4.7 Setting up the ambiopole

This section provides a guide of how to physically set up an ambiopole to match a (classical) cross-talk cancellation program.

The ambiopole needs very careful tuning, or else it will not work at all. The problem is to get the two ambiopole speakers positioned exactly at the right place for the

cross-talk cancellation to work the best. A misplacement of only five millimeters is a clear audible degradation, in the form of reduced stage width. Therefore it is very important to be patient when setting up the ambiopole.

4.7.1 Basic positioning rules

Figure 4.2 shows a schematic of the ambiopole layout. A cross-talk cancellation program is preset for a specific angle, which usually is around 20 degrees as seen from the listener. The listener should sit exactly in between the speakers, some distance back. This means that if two listeners should listen to the ambiopole, they must sit along a line.

If there is any documentation delivered with the cross-talk cancellation program, read that first, as it may contain tips on how to set up the amibopole speakers for that specific program.

The speakers should not be rotated, they should be placed along a perfectly straight line and point 90 degrees from that, as shown in the figure. The reason for this is simply because it is very hard to rotate to speakers exactly the same amount, and any difference in rotation, if only a few millimeters, will degrade the cross-talk cancellation performance.

To ease the work of tuning the positioning of the speakers, it is a good idea to paste a ruler to the floor in front of the speakers, so one can exactly measure the speaker spacing, and move them one centimeter at a time. The ruler also helps in keeping the speakers perfectly aligned.

4.7.2 Hearing cross-talk cancellation performance

The cross-talk cancellation performance is simply decided by how wide the sound stage seem to be. If there is no cross-talk cancellation at all, as for regular stereo, the sound-stage is not wider than the speakers are spaced. If the performance is good, sound sources can be heard far to the sides. Exactly how far to the side a sound can be heard is dependent on the source material, and to some extent the listener. This means that during tuning the same source material should be used, and the same listener should do the listening.

A good source material is a left/right channel pink noise. The farther out to the sides the pink noise is heard, the better is the cancellation performance. The problem with pink noise is that compared to natural recordings it shows great differences between individuals (one listener may hear the noise at +/- 45 degrees, another at +/- 60), and that it may have some unpredictable effects (hearing the sound from within the head for example). However, if it is shown to work for a specific setup, it is probably the fastest source material to use. Instead of pink noise, and ordinary recording containing anything could be used, with only one channel connected either to the left or right input channel.

An alternative is to use a two-channel natural recording, preferably made with a microphone which preserves natural binaural ques (a sphere microphone for example), containing sounds at the sides. These are often a bit more cumbersome to use for tuning than the pink noise, but is a good alternative.

[FIXME: provide test recordings on the web.]

4.7.3 Environment considerations

Sound from the ambiopole reflected from walls, floor, ceiling and nearby furniture between the listener and the speakers will harm the cross-talk cancellation performance. Thus, reflections should be avoided by placing the ambiopole speakers far from side walls, and for best performance the walls and ceiling should be covered with absorption. A thick carpet between the speakers and listener is also recommended.

Late reflections, for example coming from the wall behind the speakers and listener, are not harmful. Perfectly symmetric reflections are not harmful either, for the cross-talk cancellation that is, they are still harmful for the focus of the sound stage. Anyway, it may prove beneficial to have similar walls/furnituring to the left and right of the ambiopole, and of course the distance to the side walls must be the same to enjoy the possible advantages.

4.7.4 Distance

The distance to the ambiopole should be as short as possible. The closer one is to the ambiopole, the less interference with room reflections that harms the performance of the cross-talk cancellation.

The shortest distance possible is decided by roughly setting up the ambiopole, and while playing music move closer to the ambiopole until the speakers themselves are heard as the sound source. Then move back a decimeter for safety, and set that as the ultimate listening distance. This distance is dependent of the speaker type, but is usually closer than the distance would be for a stereo triangle. A typical distance would be 1.5 meters, for small speakers it may be less, for larger more.

When sitting further back than the optimal distance, the cross-talk cancellation performance will gracefully degrade, and which shows as a gracefully reduced stage width. This can be seen as a feature, since moving further back one meter feels just like moving 10 seats back in a real concert hall, thus one can choose stage perspective. In combination with the reverb array, this effect becomes very realistic.

4.7.5 Spacing

The spacing between the speakers is the hardest to tune, and requires patience. There is a risk of getting stuck when doing it, and then it is a good idea to take a break and come back later.

The trick here is to use the tuning source material and optimise spacing so the virtual sound sources can get as far out to the sides as possible.

Start with the speakers clearly too wide apart, then perform the following until the optimal spacing has been found:

While a virtual sound source to the left side is playing, sit in the listening position and move the head slowly sideways to the left up to a decimeter or so. If the sound moves to the right while doing it, the speakers are too wide apart, if it moves to the left, the speakers are too close. The same test is also performed for the right side. If the source material is symmetric, the left and right sound source should be heard at the same distance to the left and right. After each try, move the speakers to increase/decrease the spacing a couple of centimeters, and reasonable soon the optimal spacing will be found.

4.7.6 Trouble shooting

Cross-talk cancellation programs are still an area of research, so some problems can be noticed with them. Which the possible problems are, is specific per cross-talk cancellation program and should be mentioned in their documentation.

Anyway, a stage width of at least 90 degrees should be expected after performed tuning. If the stage is asymmetric, that is wider to one side, it is probably due to asymmetric reflections in the room, for example one side wall has much more absorption than the other, or due to asymmetry in the loudspeaker positioning.

If it does not seem to work at all (that is the sound stage gets no wider than the speakers are placed), it may be the case that one of the speakers has been connected with inverted polarity. It could also be the case that the speakers have different phase response (not common) and therefore cannot work as ambiopole speakers.

If equalisation is applied to the ambiopole, it is important that the equalisation filters are exactly the same for both speakers, at least in the active cross-talk cancellation range, or else performance will be degraded.

It is a good idea to do some further fine-tuning of the positioning a week or so after the first, when one has got used to listening to the ambiopole.

Chapter 5

Ambisonics tutorial

5.1 Introduction

This chapter contains a tutorial on how to set up a typical six channel Ambisonics system using AlmusVCU. Ambisonics supports any amount of channels though, and AlmusVCU is also very flexible, so this should only be considered as an example.

For the user interested in a small but full-featured Ambisonics system (without height though), the system here is a suitable example.

It is assumed that the reader is familiar with Ambisonics, thus the information here concerning the technology is rather brief and concentrated. More detailed information can be found at:

`www.ambisonic.net`

5.2 System design tips

5.2.1 The listening room

The room should be acoustically dead, and silent. A complete Ambisonics system reproduces the full sound field, and should not need any reinforcement of listening room reflections. This means that the only room treatment needed is absorption. In theory, the best room would be anechoic, but that can only be achieved in purpose-built laboratories.

Most tips about the listening room for Ambiophonics, which can be read about in section 4.2.1, is also applicable to Ambisonics.

5.2.2 The speaker array

Loudspeakers and amplification

Since Ambisonics creates the desired sound field by adding sound from all speakers in the air, it is important that magnitude and phase response of all speakers in the array are the same. The simplest and recommend way to achieve this is to have identical speakers and amplifiers.

The best speaker type is in theory a small speaker acting as a point source, meaning a small two-way speaker in practice. For three-dimensional arrays, point sources is strongly recommended. A problem with small speakers is that dynamics and frequency range tend to be rather limited. So, if the array is horisontal only, it is instead better to employ larger floor-standing speakers, since it is not necessary to have point source speakers in such a layout.

If the system will be listened to outside the sweet spot, or even outside the array, it is recommended to use wide dispersion speakers. It is then more important to have wide dispersion at high frequencies.

The amplification should be low noise. One problem with having many speakers is that if the amplification is noisy, it will be even more evident. Since the amplifiers must be set to full output per default (the volume is controlled by the level of the output signal from AlmusVCU), there can be a problem with excessive background noise if the amplifiers are of low quality.

Layout

Ambisonics supports any form of layout. However, angular irregular arrays (such as the standard 5.1 layout) require special decoding techniques, which is currently not supported by AlmusVCU, mainly because the most common technique is patented. The main advantage of irregular arrays is that it is possible to give preference to a certain direction by having the speaker array more dense in that direction, usually the front. A natural Ambisonics speaker array is however regular, which also is the most common form used.

Differences in distance to the speakers can be compensated for, but to avoid problems and to get potentially better performance, having the same distance to all speakers is recommended.

If the speaker array is horisontal only, it will not reproduce the vertical component of the recordings. However, since it is quite hard to mount elevated speakers properly (especially if the speakers are large), and the vertical component is often not recorded, the most common ambisonic arrays are horisontal only.

Figure 5.1 shows three horisontal layouts, square, pentagon and hexagon, and one three-dimensional layout, the dual square. Another popular three-dimensional layout when having eight speakers, is a cube, that is the speakers are in the corners of a cube. However, the dual square layout is by most considered to be the best three-dimensional layout for eight speakers.

Note that the layouts shown in the figure are suggestions only, more speakers can be employed. When designing own layouts, the only rule is to keep them regular.

5.2.3 Sweet area

The sweet spot, or sweet area, is where it is suitable to listen to the system, which is exactly in the center of the speaker array for Ambisonics. If the listener moves out of the sweet spot, localisation of sound sources is distorted. However, much work has been put into making Ambisonics suitable for multiple concurrent listeners. There are even Ambisonics systems used in large auditoria. For this, special decoding techniques to enlarge the sweet area exists, and some layouts suits better than others to achieve this goal.

Compared to the normal case with a small sweet spot in the center, a large sweet area merely means that localisation performance is worse, but equal in a larger area.

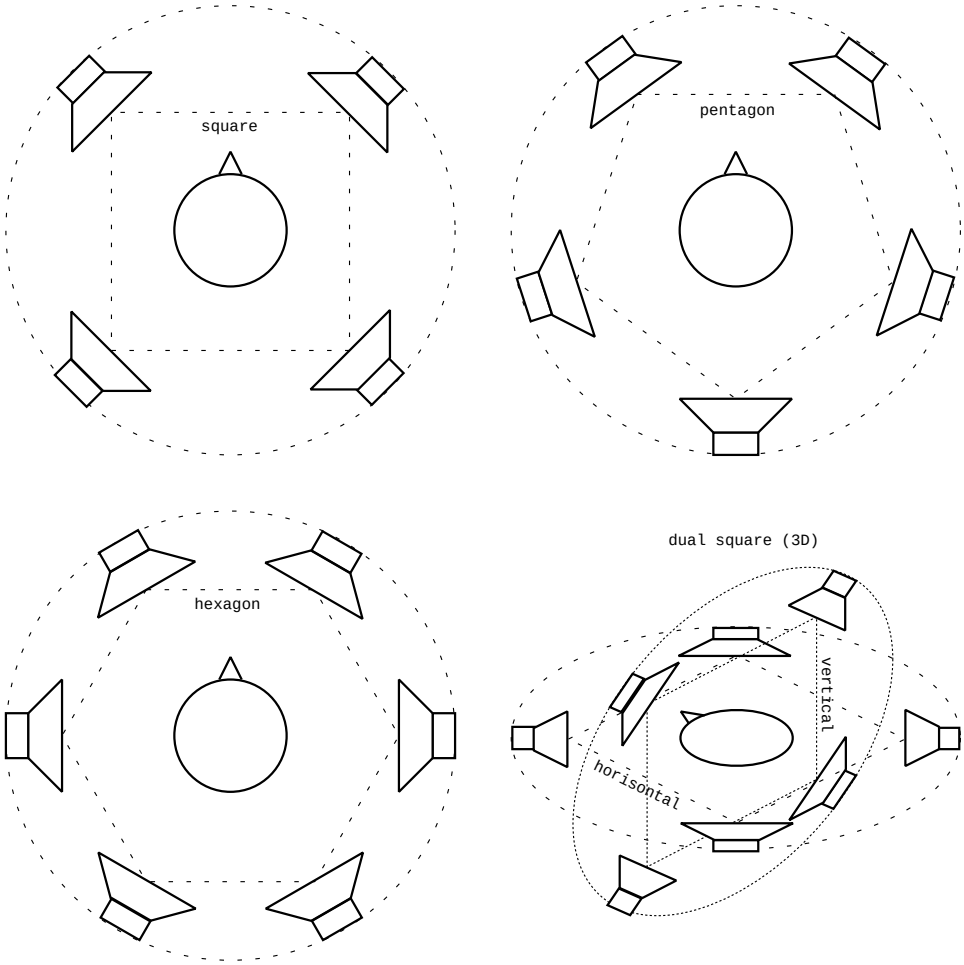

Figure 5.1: common layouts for Ambisonics.

To support a large sweet area, it is also necessary that no listener gets too close to the speakers, meaning that the speaker distance from the center should probably be at least two meters.

Currently, AlmusVCU only supports normal decoding, which produces a clear sweet spot in the center of the array, suitable for one listener. The localisation performance degrades gracefully though, so for casual listening, more than one listener is still possible.

5.3 Connecting all components

Ambisonics requires four input channels for first order Ambisonics (B-format), and nine channels for second order. In figure 5.2, it is suggested to use ADAT to transfer the four channels to AlmusVCU. ADAT supports eight channels per connection, so if second order Ambisonics is employed, it is necessary to have two ADAT links.

It is an interesting question what the signal source machine should be. It is probably most likely that it is a computer. In the future, it may appear B-format recordings on the new multi-channel formats SACD and DVD, and then such a player with an A/D conversion step could be used.

Currently, AlmusVCU does not support UHJ recordings, which is B-format without the vertical component matrixed into two channels. There exists quite many commercial UHJ recordings distributed on CD. When UHJ support is implemented in AlmusVCU (it is planned), the signal source could be a CD player, and the link would then be S/PDIF.

If the signal source is a computer, it could be the same computer AlmusVCU is running on, if it has two sound cards and a loop-back ADAT connection. One sound card is then used by AlmusVCU, and the other by the playback software. In future versions, support for playing back local files might be implemented, but since AlmusVCU is mainly designed to be embedded in a stand-alone machine, it is not of high priority.

There is also the possibility to use the hack described in section 7.5, which fools AlmusVCU to use a locally stored file as input instead of a sound card.

5.4 Installing AlmusVCU

It is assumed that the sound card in the AlmusVCU machine is an RME Audio Hammerfall sound card. Information on which hardware to choose and how to install the software is found in chapter 2. For the configuration given in this tutorial, six ambisonic channels, a 300 - 500 MHz processor or faster is recommended.

5.5 Configuring AlmusVCU

5.5.1 Static configuration

This tutorial assumes that an RME Audio Hammerfall sound card is used, and that it is configured in the way described in chapter 2.

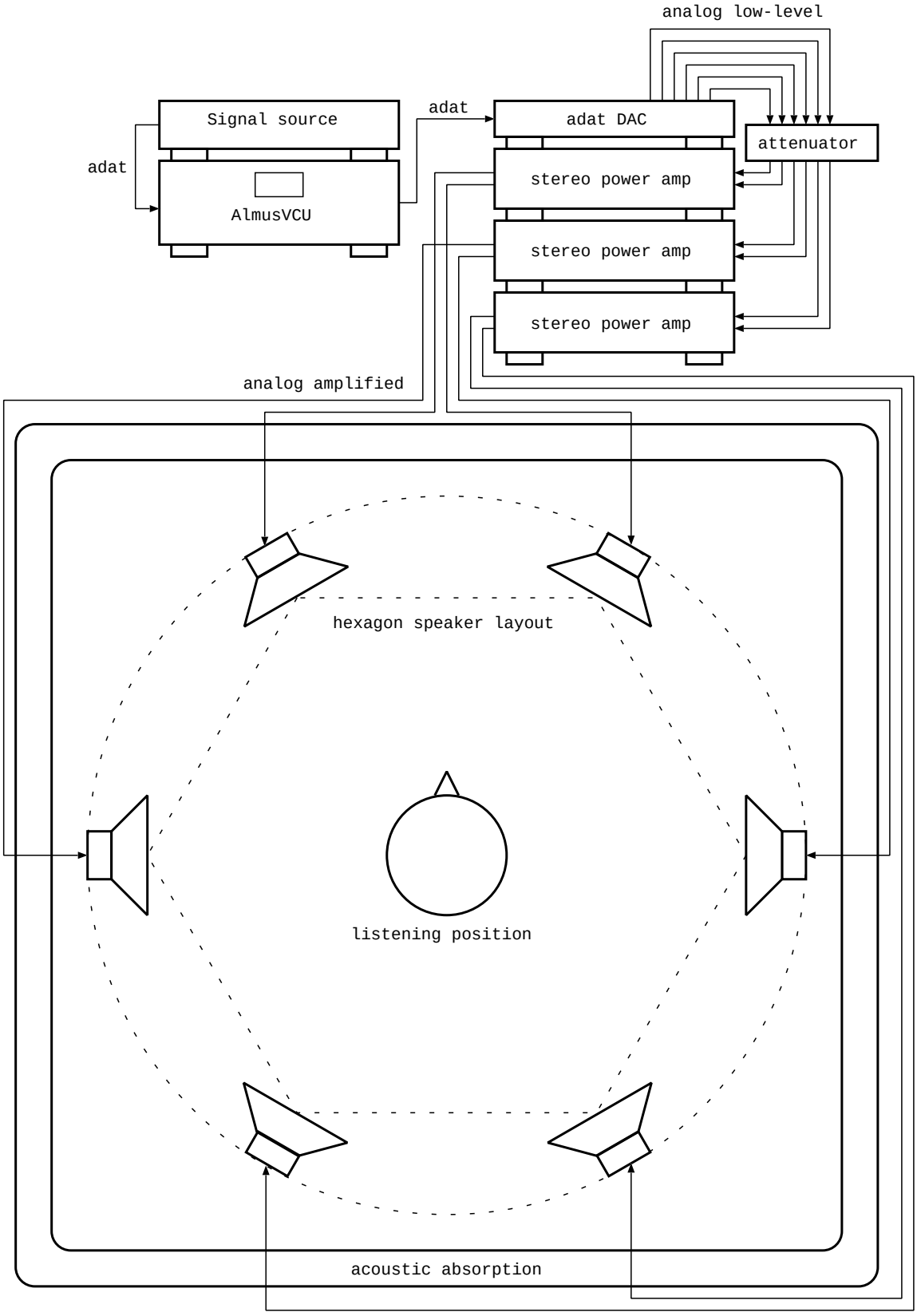

Figure 5.2: a typical Ambisonics system, with a horisontal hexagon speaker layout.

5.5.2 Setup menu

After the static configuration has been completed, and AlmusVCU is started for the first time, the setup menu is entered:

```
+--------------------+
| >Start convolver?  |
|  Load reverb prg...|
|  Del reverb prg... |
|  Signal setup...   |
|  Channel setup...  |
|  Eq setup...       |
|  Ambiopole setup...|
|  Convolver setup...|
|  System info...    |
|  Maintenance...    |
|  Screen dimmer [2] |
|  Autostart [off]   |
+--------------------+
```

Information on how to navigate the menu system is found in section 3.1.

Signal setup

First skip down to the "Signal setup" menu and enter it:

```
+--------------------+
| >Soft clip [off]   |
|  Dither [on]       |
|  [24] bit output   |
|  Allow 44.1kHz[on] |
|  Allow 48.0kHz[off]|
|  Suggest [44.1]kHz |
+--------------------+
```

The defaults should be ok, the only decision here is if we should support both 44.1 and 48 kHz. 44.1 kHz is probably the most common sample rate, but since B-format recordings is not distributed on audio CD, there may be some 48 kHz recordings as well. Both sample rates can be supported at the same time, but will make AlmusVCU use more memory, so if it is not necessary, it is better to configure to use only one sample rate. In the above example, only 44.1 kHz is allowed.

Channel setup

Next, back up to the setup menu, and enter the "Channel setup" menu:

```
+--------------------+
| >[0] (A)mbiopole   |
|  [0] (R)everb      |
|  [6] am(B)isonics  |
|  [0] (P)assthru    |
```

```
|  Ambisonic setup...|
|  In duplicator...  |
|  Out mixer...      |
|  Speaker layout... |
|  Slave channels... |
|  Mute channels...  |
|  Trim delays...    |
|  In trim volume... |
|  Out trim volume...|
+--------------------+
```

The amount of ambisonic channels should be set to the desired, which is six in this example, the rest (ambiopole, reverb and passthru) should be set to zero. Next the Ambisonic setup menu should be entered, and there the choice of using first order or second order source signal is made. In this example, first order is chosen.

```
+--------------------+
| >format [1st order]|
+--------------------+
```

Next, the input duplicator should be configured like this:

```
+--------------------+
| >get iBw from [i3] |
|  get iBx from [i4] |
|  get iBy from [i5] |
|  get iBz from [i6] |
+--------------------+
```

The input channels i3 - i6 is the first four channels of the first ADAT input (see static configuration, section 5.5.1).

In the output mixer menu, the ambisonic outputs should be put to the the first ADAT output, that is o3 - o10, in this case with only six outputs, the last two ADAT channels are not used.

```
+--------------------+
| >Mix oB1 to [o3]   |
|  Mix oB2 to [o4]   |
|  Mix oB3 to [o5]   |
|  Mix oB4 to [o6]   |
|  Mix oB5 to [o7]   |
|  Mix oB6 to [o8]   |
+--------------------+
```

The speaker layout (`Speaker layout`) should be set to exactly match the actual layout of the speakers. A common mistake is to assume that a certain AlmusVCU output belongs to a specific speaker, without verifying it first. In this example, the distance and angles for outputs o3 to o8 should be configured.

If there is any difference in distances between the speakers, it must be compensated for, either manually with trim delays, or by using the distance compensation feature (only compensates delay, not volume) found in the speaker layout menu.

In the slave setup menu, all channels should be associated to master, since there is no use for a slave volume when only ambisonics decoding is employed.

Output trim volumes may need to be adjusted manually to compensate for room problems or speaker distances. This is however better left until later, when running with sound.

Convolver setup

Before starting the convolver, the computer must be benchmarked, which is done in the convolver setup menu. If the computer is really well-tuned, the low latency setting could be tried, and the max load setting could be increased to 90% from the default conservative 85%. For basic Ambisonics setups without equalisers, the computer is normally only under light load in runtime though.

After the benchmark has been run, it could look like this:

```
+--------------------+
|  Max load [90]%    |
|  Low latency [on]  |
|  44.1 2.9ms        |
| >Rerun benchmark?  |
+--------------------+
```

Starting the convolver

After benchmarking in the convolver setup menu, the system is ready to enter runtime, and this is done by selecting the alternative `Start convolver?` in the setup menu. For safety, it is a good idea to start with playing some soft music, and have the amplifier volume controls turned down (if there are any) the first time.

To further improve sound source localisation, shelf filters can be applied, as described in the following section.

5.5.3 Shelf filters

Localisation performance can be slightly improved with shelf filters applied to the B-format inputs. They are however not mandatory, and many ambisonic decoders does not use them at all. Anyway, for very large arrays (larger than say four meters in radius), shelf filters should not be applied.

The idea with applying shelf filters is to enhance inter-aural time difference (ITD) cues in low frequencies. This will make the system interact better with the human hearing system.

Since the shelf filters are psycho-acoustic, there are no “right” filters, and some even claim that they should be different depending on layout. The general design idea is clear, but which exact crossover frequency that should be used, and what slopes is not equally clear.

Hardcore Ambisonics users may want to be able to tune them in runtime, and then they can be designed as dynamic equalisers. If that is not necessary, it may be wiser to design them using static low pass and high pass filters instead, since the flow-through delay for them will probably be shorter. Both methods will produce linear phase filters.

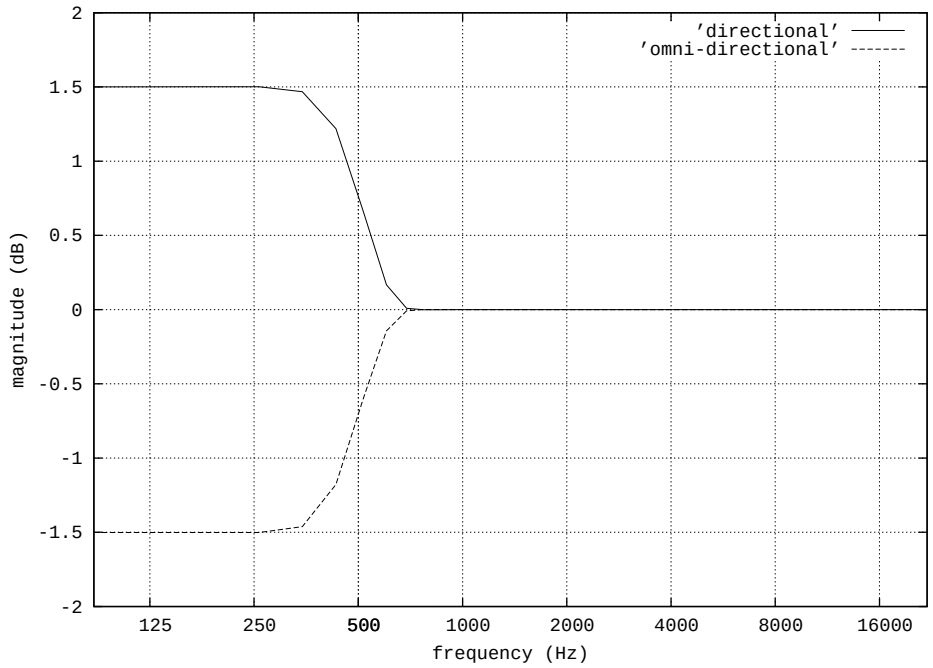

Figure 5.3: Shelf filters for the omni-directional and directional components.

The shelf filter for W (omni-directional) is usually -1.5 dB below 500 Hz, and for the directional components (XYZ) they are instead +1.5 dB below 500 Hz. The magnitude response of such filters can be seen in figure 5.3.

Dynamic equaliser

Enter the equalisation setup menu (`Eq setup`), and then the equaliser design menu, and create a new dynamic equaliser, with custom bands at for example 450 and 550 Hz. Since the dynamic equaliser does not allow to change frequency bands in runtime, one should either make several dynamic equalisers with different bands and assign them to different groups, or create several bands in one equaliser. If there is no interest in altering the cutoff frequency in runtime, this is not necessary of course.

Two dynamic equalisers should be created, one for iBw (W, omni-directional), and one for the remaining inputs. Below is an example on how the dynamic equaliser design menu for iBw could look.

```
+--------------------+
|  450.0Hz [-1.5]dB  |
|  550.0Hz [+0.0]dB  |
+--------------------+
```

Static equaliser

The static equaliser for iBw (that is W, the omni-directional component) is created with a 500 Hz soft-sloping low pass filter scaled to -1.5 dB, and to that a 500 Hz soft-sloping high pass filter is added. The equaliser for the directional components

is made in the same way, but scaling the low pass filter to +1.5 dB instead. The filter list menu for the directional component equaliser is shown below, and in figure 5.3 are the magnitude responses shown.

```
+--------------------+
|  Lowpass 500 Hz (so|ft)...
|  scale [+1.5]dB    |
|  [add] w/below     |
|  Highpass 500 Hz (s|oft)...
|  scale [+0.0]dB    |
|  Add a filter...   |
+--------------------+
```

The flow-through delay for an equaliser designed like the above will be as low as 5.8 milliseconds at 44.1 kHz.

Chapter 6

The AlmusVCU bootable CD

6.1 Introduction

The AlmusVCU bootable CD does not exist yet. Here follows a description of what it will be:

Even for the experienced Linux user, setting up an AlmusVCU machine can be quite time-consuming, and for a user not familiar with Linux and computers, it is honestly a very hard task. Therefore there exists an alternative for those users that want to try out AlmusVCU for Ambiophonics *now*, without the hassle of building and installing the software first. It is the AlmusVCU bootable CD, available as a small ISO image which can be burnt onto a CD-ROM.

The CD-ROM is simply put into the CD tray of a computer equipped with a supported sound card and powered on. The CD will boot Linux and start AlmusVCU which is pre-configured for 10-channel Ambiophonics, not all 10 channels need to be connected though. A few reverb programs and a standard cross-talk cancellation program is also loaded.

The only limitation compared to a real AlmusVCU installation is that any changes made to the configuration will be lost when rebooting or turning off the machine. This is due to that the CD is read-only, so there is no space to save the configuration to. It is worth noting that the CD will not modify anything on the computer it is run.

6.2 Prerequisites

[FIXME: to be done]

6.3 Configuration

[FIXME: to be done]

Chapter 7

Tips and tricks

7.1 Using AlmusVCU with other processors

AlmusVCU can be used together with other audio processors, for example a separate cross-talk cancellation processor. The problem is to match I/O delays of the processors. Most commercial hardware processors has no or very low I/O-delay, while AlmusVCU has significant due to it is based on a standard computer. If the AlmusVCU is equipped with a properly designed sound card its I/O-delay will be fixed, and can be seen in the system info menu. Below is an example of what it can look like:

```
+--------------------+
| Version: 0.7       |
| Uptime: 22days/2h  |
| 44.1kHz I/O-delay: |
|   static: 92.9ms   |
|    4096 samples    |
|   align: 7.3ms     |
|     324 samples    |
|   total: 185.8ms   |
|    4420 samples    |
| Cache free: 36MB   |
| Filter mem: 50MB   |
| Total mem: 249MB   |
| CPU: AMD Athlon(tm)|
| CPU count: 1       |
| CPU speed: 995MHz  |
+--------------------+
```

The total I/O-delay in this example is 4420 samples, and thus any other audio processors must be delayed with the same amount in order to sample align them. This can be achieved using a hardware delay line, or simply by routing all signals through the AlmusVCU processor. Then the passthru functionality are used to provide outputs which are connected to the other processor(s). If the other processor(s) have I/O-delay, delay alignment can be performed within AlmusVCU using the trim delay settings.

7.2 Multiple machines for the reverb engine

The reverb engine is often configured to supply many channels, and sometimes a single computer is not powerful enough to handle as many channels as desired. Fortunately, it is rather easy to use several AlmusVCU machines in parallel for the reverb engine, since there is only two input channels that need to be distributed to all of them. In a parallel configuration, it is important that the I/O-delay for all AlmusVCU machines is the same. The exact I/O-delay is looked up in the system information menu, and chosen in the convolver setup menu.

All AlmusVCU machines must receive the same sample-aligned source signal. Some CD players have both a coaxial and optical S/PDIF output, and both can normally be used at the same time. In this case, two AlmusVCU machines could be connected, one to the coaxial and the other to the optical output. A Y-cable for S/PDIF might work in some cases, but it is better to use a hardware digital audio distributor.

It also possible to send the input signal only to one AlmusVCU machine, which uses a passthru channel to send it along to the next machine, combined with proper delay alignment configuration. With this solution, the total I/O-delay will be the sum I/O-delay of all AlmusVCU machines, which can be large if there are many machines.

Currently, it is not possible for AlmusVCU to act as slave for another, so volume and reverb program settings and other run-time settings must be done on all machines, which can be a bit cumbersome if there are many. If there is a common demand for distributed reverb processing, the possibility to run AlmusVCU as a "reverb slave" to a master machine may be implemented.

7.3 Combining the ambiopole with subwoofers

[FIXME: this should be verified with practical experiments. What about cross-over frequencies, and low-pass filter shapes?]

A cross-talk cancellation program uses up a part of the bass dynamics of the ambiopole speakers. Due to this, or simply due to a general limited bass performance of the ambiopole speakers, it may be desirable to strengthen the bass with dedicated subwoofers.

A single one could be placed between the ambiopole speakers, but it is best to have stereo subwoofers, placed in a stereo triangle, or directly to the sides.

The signal feed to the subwoofers should be configured using the passthru functionality of AlmusVCU. A low-pass filter could be assigned if necessary, and left and right could be added together to a mid channel if only one subwoofer is used.

7.4 Ambisonics subwoofer rig

If the Ambisonics rig consists of small two-way speakers there may be insufficient bass performance to get a realistic sound. It is then possible to build a subwoofer rig which runs in parallel. This is done simply by adding more Ambisonics channels, and specifying the speaker layout for the new channels separately from the previous ones. This way a separate rig is created, which uses the same input as the first.

However, in the basic case, when there is only one subwoofer, it may not be desirable to perform Ambisonics decoding for it. Instead, a passthru for the omi-directional input (iBw) could be passed to the subwoofer via a low pass filter.

With two subwoofers, a two-speaker rig with the speaker placed directly to the sides could be specified.

7.5 Using a file as input

AlmusVCU is designed to operate on a stand-alone computer, and has therefore no interface to play locally stored sound files. However, in some cases it is strongly desirable to do so, and then there is a work around.

The I/O devices specified in the static configuration file, `almusvcu_static_config`, is passed directly without modification to the BruteFIR convolution engine. Thus, any I/O module available in BruteFIR can be specified, for example the file I/O module. The documentation of which parameters for the modules, and which modules that are availble, is found in the BruteFIR documentation.

Below an example from the static configuration file when the input is a four channel B-format file :

```
input_device: "file"/'{ path:"/home/anders/bformat/Aria_B.wxyz"; }';
input_channels: 4;
input_formats: "s16_le";
```

Of course, the output device can also be set to any supported by BruteFIR, so the output could be put to a file for example.

When an input file comes to an end, BruteFIR will stop running, and that means that AlmusVCU will silently exit runtime mode and return back to the setup menu.

7.6 AlmusVCU as a dedicated reverb processor

AlmusVCU is not a strict Ambiophonics or Ambisonics processor, it could for example also be used as a traditional reverb processor, that is to add reverberation to a recording in post-production.

It could for example be used to create surround channels to make a 5.1 recording out of a stereo recording, or simply to add reverberation to a single channel before mixing.

The speaker location settings in AlmusVCU will be fundamental to control how the reverberation is performed. The direction of the speaker location compared to the listening position specifies the direction from where the reverberation should come. Thus, if a speaker location is put in the back, the reverberation for a reverb channel associated to that speaker location will sound as it would from the back in the original hall, if the sound source was on the stage.

7.7 Ambiophonic reproduction of 5.1 recordings

[FIXME: To be done, need to do some practical experiments here]

Passthru channels for surround, drop center channel, alternatively passthru for that too. Add reverb array to that.

7.8 Using AlmusVCU for Panorambiophonics

Panorambiophonics, or PanAmbio for short (and a bit easier to pronounce), is the concept of using one ambiopole in the front, and one in the back, to create a near-360 degree sound-stage with accurate localisation.

It is still an experimental format which require specially made four-channel recordings, and while working very well for some, the sides and the back are a bit more sensitive to variations in listeners' hearing system and can therefore work less well for some individuals.

Configuring AlmusVCU for PanAmbio is very easy. Just configure it to use two ambiopoles, one which will be placed in front, and one in back. For extra immersion and some height information, a reverb array can be added to it. The reverb engine should then be configured to use the left and right front channels as input.

Further reading on Panorambiophonics [FIXME: to be done].

7.9 Ambiophonic/Ambisonic hybrid systems

It is possible to combine the reverb array for Ambiophonics with the main array for Ambisonics. Concerning speakers, Ambisonics reproduction is much more demanding that reverb reproduction, so the surrounding speaker array should be designed for Ambisonics. It will then also work fine for reverb reproduction, being a bit better than it needs to be.

One potential problem is the front stage, which is reproduced with an ambiopole in Ambiophonics. The Ambisonics array might have speaker very close to that, and for reverb reproduction it is not desirable or at least unnecessary to have reverb speakers say within 40 degrees from the ambiopole. Either an Ambisonic array such as the dual square layout which is not dense in the front is chosen, or the speakers closest to the ambiopole is not used or attenuated in when reproducing reverb.

The dual square layout matchs quite well speaker placement recommendations for a reverb array as well.

7.10 Higher sample rates

Throughout this manual, sample rates 44.1 and 48 kHz is used. However, AlmusVCU is not limited to that, it can also work on 88.2 and 96 kHz. However, the author does not find any great improvement in the increased sample rate, and convolution performance will be much worse due to the many more samples to process, therefore it is not recommended to use them.

If the RME Audio Hammerfall card is used, 88.2 and 96 kHz can be supported, but not at the same time as 44.1 and 48 kHz, since the amount of channels are reduced. Each ADAT channel will only transport four channels instead of eight. Specially made D/A-convertes supporting this sample-split mode must be employed as well.

Chapter 8

Developer information

8.1 Filter program file format

8.1.1 General description

A filter program file is a GNU tar archive which contains a human-readable index file (named `filter.txt`), and one or more bzip2-compressed raw data files containing filter samples with one channel per file. The index file must be first in the tar archive, and the order of the raw data files must be the same as specified in the index file.

The filter program may contain filters in several sample rate versions.

Here follows an example `filter.txt` for a reverb program, however most fields are applicable to equalisation and cross-talk cancellation programs as well:

```
type: "stereo impulse response b-format";
comment: "";
name: "Comun Fe";
longname: "Comunale Ferarra";
rates: 44100;
format: "float_le";
length: 2.722;
predelay: 0.045;
files: "comun-fe-left-w.pcm.bz2",
       "comun-fe-left-x.pcm.bz2",
       "comun-fe-left-y.pcm.bz2",
       "comun-fe-left-z.pcm.bz2",
       "comun-fe-right-w.pcm.bz2",
       "comun-fe-right-x.pcm.bz2",
       "comun-fe-right-y.pcm.bz2",
       "comun-fe-right-z.pcm.bz2";
```

All channels should be of equal length, and should be given in seconds in the `length` field. The convention is to specify the length to be a bit longer than the files are to avoid that samples are dropped when converting from seconds to samples (where truncation is used instead of rounding). The format of the files is specified in the `format` field, and is one of the following:

- `float_le`, 32 bit floating point, little endian

The sample rate(s) is specified in in the `rates` field. If there are more than one rate, they should be ordered in raising order, and the files should be ordered with all files for the lowest sample rate first, and so on.

The name of the filter program is specified in the two fields `longname` and `name`. The `name` field can only contain 20 characters and is required, while the optional `longname` can specify a longer name, up to 255 characters.

The optional comment field will be ignored, and thus could contain any (valid) string.

The type field specifies the type of the filter program.

8.1.2 Reverb program

A reverb program is simply two B-format impulse responses recorded in a good position of the acoustical place to reproduce. The sound source was placed at the left side of the stage for the first impulse response, and at the right side of the stage for the second impulse response.

The filter files must be ordered with left impulse response w, x, y, z channels and then right w, x, y, z channels, and should not contain any unnecessary null samples in the beginning or the end of the files. The suggested predelay is put in the `predelay` field, the unit is seconds.

The amplificiation of the impulse responses should be scaled to be about 0 dB.

The type field is set to (exactly) `"stereo impulse response b-format"`.

8.1.3 Cross-talk cancellation program

The cross-talk cancellation program stores the four FIR filters in this order: left direct path, left cross path, right direct path, right cross path. Alternatively if the cross-talk cancellation is symmetric, only two filters are stored, first the direct path then the cross path.

AlmusVCU calculates the delay of the filters by finding the largest sample (absolute value) of the direct path(s). The predelay field is not used.

The type field is set to (exactly) `"crosstalk cancellation"`.

8.1.4 Equalisation program

The equalisation program stores simply a single channel FIR filter. If the predelay field exists, it specifes the delay of the filter which will be used by the AlmusVCU sample alignment. If the predelay field does not exist, AlmusVCU will specify the delay as the position of the largest sample (absolute value), which work for most filters with reasonable phase response, and is thus the preferred method.

Predelay is truncated when transformed into sample delay, thus if the delay is 0.00272 at 44.1 kHz, which is 119.952 samples, the sample delay is truncated to 119 samples.

The type field is set to (exactly) `"equalisation"`.

8.2 Adding support for new devices

User input and display output devices for the `textgui` user interface module can easily be added by a programmer. Looking at the source code files of the modules delivered with the source archive should be enough. These are:

- `keyin_gpm.c`, the `ki_gpm` module which uses the GPM mouse server to get user input from the mouse.
- `keyin_curses.c`, the `ki_curses` module which uses ncurses to get user input from the keyboard.
- `tw_curses.c`, the `tw_curses` module which uses ncurses to put the user interface to the computer console.
- `tw_matrix_orbital.c`, the `tw_matrix_orbital` module which uses a Matrix Orbital LCD2041 display to present the text interface.

Modules with the prefix `ki_` are assumed by textgui to be user input modules, and those with `tw_` prefix to be text output devices. The makefile included in the source archive shows how to compile the modules correctly. The resulting files should be named as `<module name>.vcu`.

8.3 AlmusVCU files

Apart from the static configuration file (`/etc/almusvcu_static_config`) and binaries, AlmusVCU has a storage directory which will contain a set of files:

- The user configuration file, named `almusvcu_user_config` and `almusvcu_\ user_config.new`. The last is a backup file used to avoid data loss if the machine loses power exactly when writing to the file. The contents of the configuration file is all user settings, and the format is the same as in the static configuration file, however with other parameters.
- Reverb program settings. These files are named with an MD5 sum followed by `.revprg`, for example `575613aec3b9e4994c262889bbed47ff.revprg`. The MD5 sum uniquely identifies the reverb program the configuration file is associated to. It contains user-specified volume and delay settings for the reverb program.
- Cached filter programs. These are named with a type prefix followed with an MD5 sum which uniquely identifies the program, and followed by `.tar`. These are simply copies of loaded filter programs. `eqr.*.tar` identifies equalisation programs, `xtc.*.tar` cross-talk cancellation programs, and `rev.*.tar` identifies reverb programs. The MD5 sum for the reverb program matches the one found in the corresponding `.revprg` file.

In the static configuration, there is a logging path specified, and there the following files will appear, if logging is activated:

- The AlmusVCU log, named `almusvcu.log`, normally useful only for debugging.

- A set of BruteFIR logs, normally only used for debugging. However, if the convolver refuses to start, these logs can provide some help for normal users as well. The logs are `brutefir_console_output`, `brutefir_stderr_output` and `brutefir_last_config`. The logs contain the CLI output, standard error output and the configuration file used at start respectively. The standard error output file is the one which can be useful for normal users, since if the convolver fails to start, the reason is usually logged there.

- If the equaliser properties menu is opened, the filter viewed will be dumped to disk (not dynamic equalisers though). The file names will be `equaliser_mag` for the magnitude response, and `equaliserX` where `X` is replaced with the sample rate, thus if there is more than one sample rate, all sample rate versions will be logged. The format is text, in one column for the filters, and two columns (frequency in Hz and magnitude in dB) for the magnitude response. The magnitude response is calculated on the lowest sample rate version. When equaliser properties is called again, any previous logged filters will be overwritten.