AlmusVCU User Manual

version 0.85

18th July 2004

Preamble

This is the user manual for AlmusVCU v0.85, 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 unstable. This release is from a 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, July 2004
`torger@ludd.luth.se`

Contents

Chapter 1

Introduction

1.1 AlmusVCU files

This manual assumes that the following components are available:

- `almusvcu-0.85.tar.gz`, the source code for AlmusVCU.
- `brutefir-1.0.tar.gz`, the source code for the convolution engine BruteFIR.
- `almusvcu-filter-programs-0.85.tar.gz`, standard filter programs for AlmusVCU. A filter program is a generic name for the three types reverb, cross-talk cancellation and equalisation programs which AlmusVCU supports.

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
- ITU 5.1 or other discrete multi-channel arrangements

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. AlmusVCU can be configured to collaborate with other audio processors, even 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 (about 20 – 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 acts like 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, or other acoustic spaces. 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 one or several representative parts of the stage.

The resulting reverb program is used by AlmusVCU to convolve recordings, typically stereo recordings from an ordinary music CD, 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.

For increased compability with existing impulse response recordings, AlmusVCU can also handle simpler discrete non-3D reverb programs (common in commercial convolution-based reverberators), and properly pre-processes them in order to allow free speaker placement. AlmusVCU provides reasonable good results even when the reverb program only consists of a single mono impulse response.

Cross-talk cancellation

AlmusVCU can be loaded with a cross-talk cancellation program, which is assigned to a pair of narrowly spaced speakers (usually) 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 in the literature most commonly referred to as “Stereo Dipole”. However, since AlmusVCU can be used for Ambiophonics, the cross-talk cancelled stereo pair is here 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 (which aims at being able to reproduce sounds from any direction).

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

Ambisonics decoder

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

Hybrid systems support

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

Switchable channel modes

AlmusVCU can be configured for several different types of decoding, which then can be switched in runtime. For example, there could be separate modes for Ambiophonics, Ambisonics and 5.1 home theater adapted to a single speaker setup, and depending on the recording played, the proper mode is chosen.

Scalable user interface design

Whether AlmusVCU is used in a 20 speaker Ambiophonics system, or in a plain stereo system, the user interface is suitable adapted to fit the given purpose. If the reverb engine is not used, there is no trace of reverb engine controls in the running user interface, if only two channels of 20 possible are used, only those two will be shown in the interface, and so forth.

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 for example be used for A/B testing 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.

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.

Flexible channel model

Speaker feeds (physical channels) and internal channels (logical channels) work independently from each other. This means that several channels can be mixed into a single speaker. It also makes trim volume, delay and equalisation manangement very flexible. For example, each speaker feed could be corrected if necessary with proper equalisation and trim volumes and delay, regardless how the logical channel configuration will be. On top of that, the logical channels (reverb, ambiopole, ambisonic and passthru channels) can have their own trim volumes, delays and equalisers.

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 the need to reconfigure.

High audio signal resolution

Per default, AlmusVCU makes all its sound processing in 32 bit floating point, which is high performing and with a sound quality good enough for most needs. However, should a superior signal quality be desired, 64 bit floating point resolution can be used internally. It allows for 24 bit dithered linear PCM on the outputs.

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 a set of I/O-delay/reverb length trade-off alternatives to choose from.

Saving/loading configuration

Configurations can be saved to be recalled at any time. This is useful for backup purposes.

Open filter program formats

The filter programs, that is reverb, equalisation and cross-talk cancellation programs, is stored in a simple format, described in this manual. Thus it is easy for the user to create own filter programs, for example using third party room equalisation software to generate an equalisation filter which then can be loaded into AlmusVCU.

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 can 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 Design goal

As said, AlmusVCU mainly aims to be a stand-alone surround sound processor for Ambiophonics and Ambisonics and hybrid formats. This means that when set up, it should be easy to use to play back recordings in a home multi-channel HiFi system, operating it with a remote control. However, rather than being designed as a slick commercial product aimed at a specific narrow use, AlmusVCU is made to be flexible to accomodate many user interests, where one of the main is to allow experimentation with hybrid surround formats. Its users specify in which direction AlmusVCU will be developed.

In any case, AlmusVCU will due to its flexibility and many features, be harder to set up than a typical commercial surround sound processor product. The goal is however that once set up, it should be easy to use.

Since it is designed to work in a stand-alone embedded platform where AlmusVCU is the only application running, lots of typical features found in computer workstation sound applications is not to be found in AlmusVCU. For example, a playlist system for playing audio files from hard disk, or a fancy graphical user interface. AlmusVCU and the computer it runs on should be regarded as a single unit, which should be compared to hardware reverb processors and convolver units rather than audio processing software for PCs.

1.2.5 Acknowledgements

The author wants to thank Angelo Farina for all thorough and educating answers to the many questions concerning general acoustics, Ambisonics and binaural recording and reproduction theory, signal processing, psycho-acoustics and more. He has also helped to make this project included in official research, which has allowed deeper study, and provided access to serious hardware.

Ralph Glasgal's encouraging support and enthusiasm is one of the main reasons why this project started in the first place. He is the orignal inventor of the Ambiophonics concept, and through his Ambiophonics Institute, research like this has got financial support, and access to one of the finest HiFi audio labs. Furthermore, he has technically supported the project by sharing his knowledge in psycho-acoustics in general and Ambiophonics in particular.

Robin Miller has during development been the most active user of AlmusVCU, and provided valuable feedback. He has really pushed limits of what AlmusVCU can

do through his highly advanced hybrid sound format configurations used in his own research.

1.2.6 Legal notices

1.2.6.1 Licenses

The AlmusVCU software is free, and its source code is available to the public under the Open Software License version 2.0. In short it means that anyone is allowed to copy, distribute and modify the software, as long as the terms are not changed. In practice this means that any program using AlmusVCU source code must also be covered by the Open Software License (OSL) or compatible license. The full license is included in the source code archive. The underlying convolution engine BruteFIR is also covered by the OSL.

This documentation is also covered by the Open Software License version 2.0.

For the filter programs, the terms is given for each program, found in the filter program archive. The minimum requirement for a filter program to be distributed with AlmusVCU is however that it is free for non-commercial use, and can be freely distributed.

1.2.6.2 Disclaimer

There is no warranty. The full terms is in the Open Software License. The AlmusVCU software and this documentation is used at the user's own risk.

1.2.6.3 Trademark notices

All trademarks, registered or otherwise, are the property of their respective owners. ADAT is a registered trademark of Alesis Corporation. S/PDIF and SACD are registered trademark of Sony Corporation. DVD is a registered trademark of Toshiba Corporation. RME and Hammerfall are registered trademarks of RME Intelligent 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!

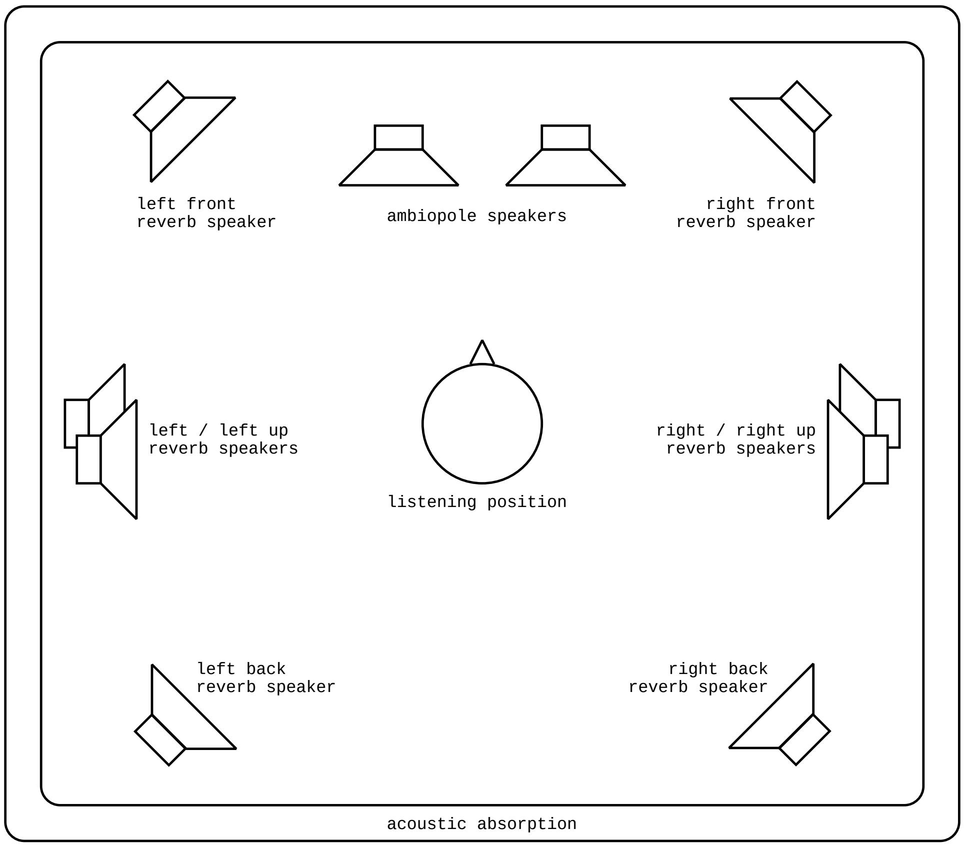

Figure 1.1: a typical Ambiophonics system, with an ambiopole and eight reverb speakers.

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`

The overall design goal of Ambiophonics is to make a realistic reproduction of acoustic music. In recent times, other experimental formats with this goal, which require purpose-made recordings, has been introduced by the Ambiophonics research group. This have created a family of formats, which all relate in one way or another to the original Ambiophonics approach. All these new formats have their own names, but are said to belong to the "ambiophonic family". However, when Ambiophonics is mentioned alone as a format, what is referred to is the classical approach as described above. Time will tell if any of the new ambiophonic formats will see any commercial success (that is commercially available recordings).

1.4 What is Ambisonics?

Despite the similarity in names, Ambisonics has nothing to do with Ambiophonics. Note that Ambisonics lacks the 'o' after "ambi", so it is Ambi-sonics, and Ambio-phonics, a fact which make the two names a bit easier to tell apart than it may seem at first glance.

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. It is also possible to create a B-format signal by artificial means (just like making a stereo mix out of several mono signals). The four B-format 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 (the Z channel is not used), 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 superpositioning (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 are only created artificially, since there is no commercially available second-order microphone. However, there are working prototypes for second-order and even higher order microphones available (the higher the order, the better, but more media channels are required). Time will tell if these prototypes can be made into working commercial grade microphones.

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

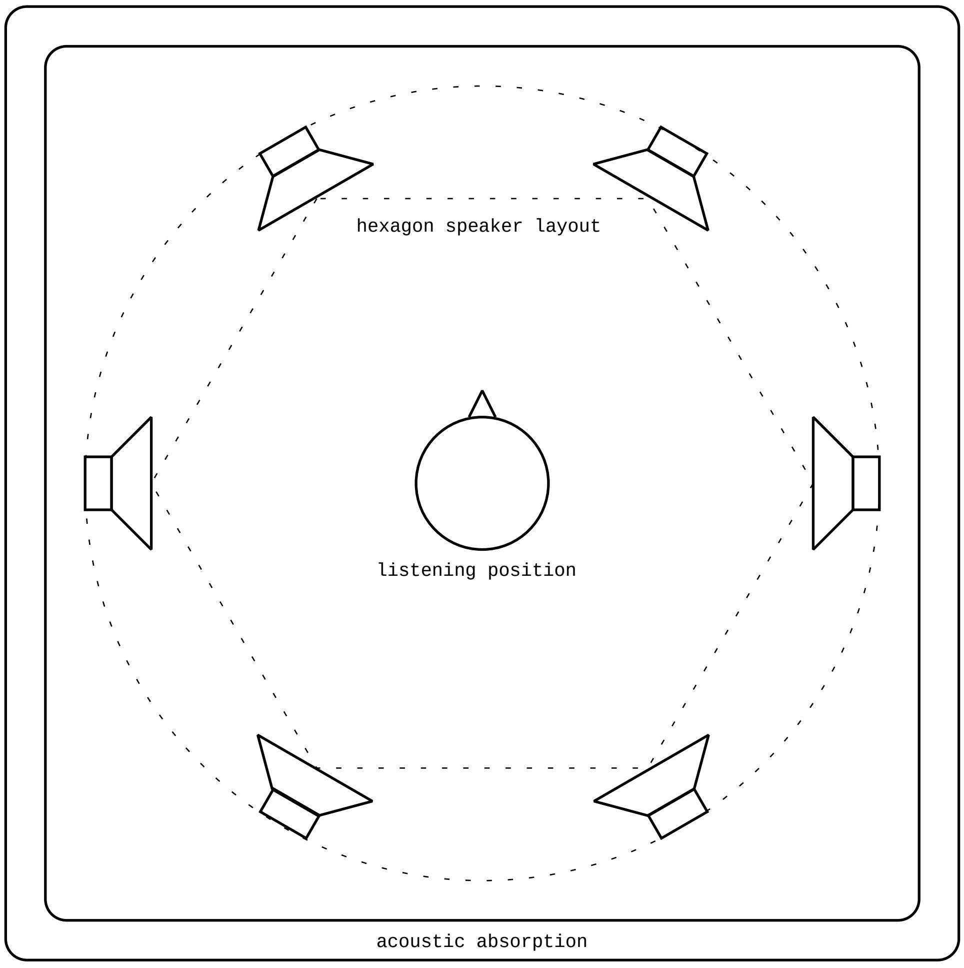

Figure 1.2: a typical Ambisonics system, employing a horisontal hexagon speaker layout.

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 makes three channels out of two.

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 by the user.

Further information on Ambisonics can be found at:

`www.ambisonic.net`

1.5 Ambiophonics vs Ambisonics

If set up well, both Ambisonics and Ambiophonics sonically outperforms the mass-market standards stereo and 5.1, but how do 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. Higher order Ambisonics may put a different light on this, but its availability is very limited, and currently there is no way to make true acoustic recordings of good quality. Here we assume that Ambisonics is first order.

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 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.
- For Ambisonics to work well, it requires that all speakers and amplification are the same. Ambiophonics allows for asymmetric systems, a well-dimensioned system has highest quality speakers in the ambiopole, and cheaper speakers in the reverb array.
- Ambisonics requires precise positioning of all speakers, while Ambiophonics requires extremely precise positioning for the ambiopole speakers, but is very adaptable on reverb speaker placement.
- Ambiophonics is much more computationally demanding than Ambisonics. The latter 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.

Experiments of combining Ambiophonics and Ambisonics into a single system have been conducted by the Ambiophonics research group. 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 be combined.

Chapter 2

Installation

2.1 Introduction

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

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

Note that any specific hardware or software suggestions may be out of date if this manual is old.

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 may be hard to achieve on a workstation used for other tasks. The software is also designed to be built into an embedded hardware platform, forming a HiFi component, with an LCD and a 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.

2.2.1.1 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. Thermal throttling can also often be turned of in the BIOS, but that could of course put the processor at risk, if the cooling is not efficient enough.

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.

2.2.1.2 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.

2.2.1.3 Sound card

The software supports in principle all sound cards supported by ALSA (Advanced Linux Sound Architecture), but it is strongly recommended to use a high quality digital sound card, and employ 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 is available on the output. If the sound card supports synchronous 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. At the time of writing, these cards has gone out of production and been replaced by RME Audio HDSP which works well too. However, it is often better to try to get hold of older hardware than the newest, since older drivers tend to be more stable.

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.

2.2.1.4 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 works as well). For a programmer it is however very easy to add support for other displays to AlmusVCU, as described in section 10.2. 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.

2.2.1.5 Harddisk

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.

Note that some reverb programs can be very large, sometimes several hundreds of megabytes, and in order to store them on the harddisk, or being able to cache them (optional), it must be able to store the corresponding amount.

2.2.1.6 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. The user interface is specially adapted for a scroll wheel for the UP/DOWN controls (scrolling up/down in menus, increasing/decreasing volume etc).

For a programmer it is simple to add support for new user input devices to AlmusVCU, as described in section 10.2.

Apart from LEFT, RIGHT, UP and DOWN there are optional keys which can be used by input devices with more than four keys, such as a standard computer keyboard. These optional keys include direct access to master volume, muting, reverb program selection and more.

2.2.2 Operating system

Any recent Linux distribution can be used as the base operating system. With the proper hardware and a tight 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. In any case, it is however not likely that the system will work reliably with the lowest configurable latency 3 ms, since Linux is not a hard realtime operating system.

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. Searching the Internet with the key words "linux low latency howto" is a good start to find information on how to achieve the best low latency performance. 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.

2.2.2.1 Low-latency kernel

The old 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.

The new Linux 2.6 kernel has a compile-time option called "Preemptible Kernel", which roughly corresponds to what the 2.4 low-latency patches do, and thus should be activated.

It is recommended to use the newest Linux kernel available.

2.2.2.2 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.

2.2.2.3 Swap disk

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

2.2.2.4 Hardware interrupt issues

For the best low latency performance, the sound card should be assigned the highest interrupt priority, and any unnecessary hardware (that may generate interrupts) should be removed from the machine. If a mouse is used as remote control, and low latency is strived for, it is recommended to remove or fix the mouse ball (or corresponding mechanism), so no unnecessary interrupts is generated by it.

Before taking any drastic measures (like removing the ethernet board or graphics board), make sure that there really is a problem. The system may work at the desired latencies without modification.

2.2.2.5 Reboot/shutdown

When the command "shutdown -r now" is issued by root, the system should shut down nicely and reboot, and when the command "shutdown -h now" is issued, the system should shut down and power off. Only then the reboot and shutdown functions in the AlmusVCU user interface will work as expected.

2.2.2.6 File system

If the AlmusVCU machine will be stand-alone, it is recommended to use a journaling file system, such as ext3. A journaling file system will properly and quickly recover from a sudden power loss, such as when the computer is turned off directly without a proper shut down procedure. This will make the machine work more like a real HiFi component, where the only shut down procedure is to press the power button.

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

The channel index listing after the channel count, allows for custom reordering of the channels, if the default as provided by the driver is not pleasing. Note that the static configuration starts with channel index 0, while in the program the first channel index is 1. In the case of RME Audio Hammerfall card example above, the inputs and outputs will be the following in the program:

- channels 1 - 8: ADAT 1
- channels 9 - 16: ADAT 2
- channels 17 - 24: ADAT 3
- channels 25 - 26: S/PDIF

The input/output formats and devices are directly passed to the convolver engine BruteFIR, and documentation for those parameters can thus be found in the BruteFIR documentation.

As mentioned in section 2.5.1, the sound card must be configured to work as clock slave if any of its inputs are 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: {
    "twki_curses" {
        key_left: "key_left";
        key_right: "key_right";
        key_up: "key_up";
        key_down: "key_down";
        key_mvol_up: "p";
        key_mvol_down: "P";
        key_gvol_up: "key_f1";
```

```
            key_gvol_down: "key_f2";
            key_reverb_prg_next: "n";
            key_channel_mode_next: "m";
            key_group_next: "g";
            key_master_mute: "key_enter";
            key_group_mute: "s";
        },
        "tw_matrix_orbital" {
            device: "/dev/ttyS0";
            nicefont: true;
        },
        "ki_mouse" {
            device: "/dev/psaux";
            type: "imps2";
        },
        "textgui" {
            rom1_path: "/cdrom"; # root path for the device
            rom1_name: "CD-ROM"; # name in user interface
            rom1_device: "/dev/cdrom"; # device to mount
            rom1_filesys: "iso9660"; # mount filesystem
            rom1_mount: true; # requires mounting
            rom1_write: false; # allow write to device
            rom2_path: "/floppy";
            rom2_name: "floppy";
            rom2_device: "/dev/fd0";
            rom2_filesys: "msdos";
            rom2_mount: true;
            rom2_write: true;
            rom3_path: "/usr/local/almusvcu/programs";
            rom3_name: "harddisk";
            rom3_mount: false;
            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, they will follow directly after within braces. In the above example the modules `twki_curses`, `tw_maxtrix_orbital`, `ki_mouse` 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 `twki_curses`, which takes input from the keyboard, and `ki_mouse`, which takes input from the mouse. Note that both can be used at the same time.

Modules with the prefix `tw_` provides text window output for the textgui module, modules with the prefix `ki_` provides user input. The prefix `twki_` indicates that the module is a combined output and input module.

2.4.2.1 Mouse input

For taking input from the mouse, there are two modules, the `ki_mouse` and `ki_gpm` module. The latter takes its input through the GPM daemon, which is a popular general purpose mouse daemon which is delivered with most Linux distributions. The `ki_gpm` module needs no configuration, but a GPM supporting the mouse wheel must be installed and correctly configured. The `ki_mouse` module on the other hand, has the mouse driver built in, and connects directly to the mouse on the lowest level. Therefore it cannot be used at the same time as another software is using the mouse, such as GPM or X-windows. In general, `ki_mouse` is suitable for embedded installations, and `ki_gpm` on workstation installations. For `ki_mouse`, path to the device the mouse is connected to, and the type of mouse protocol used must be specified. Currently, only ps/2 mice which work with the imps2 mode in GPM are supported by `ki_mouse`.

2.4.2.2 Keyboard input

As seen in the example, the `twki_curses` module allows for setting more keys than LEFT, RIGHT, UP and DOWN. If no configuration is given to the module, it will use the defaults which is to map the four direction keys to the cursor keys of the keyboard, and leave the other functions unused.

The other configurable keys are only used in runtime, and some are only usable in more advanced configurations where several channel modes are used:

- `key_mvol_up/down` – increase/decrease master volume.
- `key_gvol_up/down` – increase/decrease channel group volume.
- `key_reverb_prg_next` – select the next reverb program (rotate).
- `key_channel_mode_next` – select the next channel mode (rotate).
- `key_group_next` – select the next channel group (rotate).
- `key_master_mute` – mute/unmute all channels.
- `key_group_mute` – mute/unmute the channel group channels.

A detailed description of each function these keys provides direct access to is found in section 3.5. The name of the keyboard key to bind to is associated to the label. The name is for alphanumeric keys simply the character it produces. Shift is allowed, that is "p" and "P" are not the same, the first means pressing the "p" key only, and the second means pressing shift and "p". Special keys like backspace, enter, home etc has corresponding names, "key_backspace", "key_enter" and so forth. The listing of all key names is too long to be shown here, but can be found in the documentation for the ncurses library, or simply by looking in the source code of the `twki_curses` module.

2.4.2.3 Screen output

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

Matrix Orbital LCD2041 (or VFD2041) 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 Matrix Orbital module has an option called `nicefont`, if that is set to true, some characters are modified to be better looking than the standard. The VFD model however, lacks one pixel line on the character elements and on that this option does not work properly.

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.

2.4.2.4 Textgui modules overview

- `ki_mouse` – low level mouse input
- `ki_gpm` – mouse input through GPM daemon
- `tw_matrix_orbital` – output to Matrix Orbital LCD2041 or VFD2041.
- `twki_curses` – keyboard input and screen output through libncurses (text output).
- `twki_svga` – keyboard input and screen output through svgalib (graphical output, emulates a VFD2041 on full screen). Runs in SVGA mode per default (640x480x256), if `vga` is set to true, it will instead run in VGA mode, (640x480x16).

2.4.2.5 Read-only devices

The user interface provided by `textgui` allows loading filter programs from read only devices. This is configured with the `romX_*` settings, which in the example is adapted for a CD-ROM, floppy disk and a directory on a hard disk. There can be up to eight units.

The name rom (read only memory) is historical, functions for storing logs and configurations require write permission. It the example configuration, the floppy has been given write permission and will thus be listed as an alternative to save to for functions that require a place to store data. The setting `romX_write` is set to true to allow writing. If the setting is omitted, the device is considered read only. Writing will only work if the device can be mounted with write permission.

If the device requires temporary mounting (floppies, CD-ROMs), file system (`romX_filesys`) and device (`romX_device`) must be configured, and `romX_mount` must be set to true. For permanently mounted devices, these settings should not be set.

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. In the case of a turntable, it must first be pre-amplified, in order to deliver the proper analog format to the A/D converter.

2.5.1.1 New 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 (or 96 if the higher sample rate really is desired), and then feeding it to the AlmusVCU machine through S/PDIF.

If the player has a S/PDIF digital output, it should be verified which format it has. If it is 24 bits / 48 kHz it will outperform any A/D converter at the same sample rate. Also, there are some rare DVD-Audio players which actually provide 24 bits / 96 kHz S/PDIF output.

2.5.1.2 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 drift 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 generally considered to 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 at 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.

Another more powerful multi-channel transmission format which is gaining in popularity is MADI, which allows for 64 channels at 24 bit on a single wire.

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 can simply be 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 are self-made after finding the suitable resistance after some experimentation, or variable audio attenuators can be used. The resistance can be connected in series, or a 10 kohm voltage divider can be built. The advantage of the voltage divider is that it is not dependent on the input impedance of the power amplifier.

2.5.3.1 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 at least theoretically improves 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 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 or D/A converters. It should be verified how the volume is handled though, a digital power amplifier might require a matching pre-amplifier to work at all.

Instead of a D/A converter, it may be necessary to convert multichannel outputs from the AlmusVCU sound card (typcially ADAT), to S/PDIF for the digital amplifiers. For this, there are a few splitters on the market (for example splitting one ADAT output into four S/PDIF outputs).

2.5.3.2 Loudspeakers with built-in amplifier

There are a number of loudspeakers on the market which have their amplifier built in, and in some cases even the D/A may be inside. Using such speakers reduces the amount of boxes needed significantly. However, since the availability of such speakers are less than conventional equipment, it is less likely to find the best price/performance ratio among those products.

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. At the top of the diagram is the sound input device (which usually is the computer's sound card), and from that the sound flows through AlmusVCU down to the sound output device. There are four distinct audio 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.

3.1.1.1 Input and output channel matrices

AlmusVCU has two types of channels, physical and logical channels. The physical channels are the sound card channels, while the logical are the audio processing block channels, that is the ambiopole, reverb ambisonic and passthru channels. Both these types of channels are tied together in the input and output channel matrices, where for example several physical inputs can be mixed into one logical, or the other way around.

3.1.1.2 Standard functions per channel

To all physical and logical input and outputs there are a set of standard functions associated. These are volume/mute, delay and equalisation, which all can be controlled in runtime, either individually or in larger groups.

The equalisation is simply generic convolution. The convolution filters can be created with the built-in equalisation designer, or be loaded from disk.

3.1.1.3 Ambiopole

The ambiopole audio processing block provides cross-talk cancellation for one or more ambiopoles. Standard Ambiophonics or Stereo Dipole installations use only

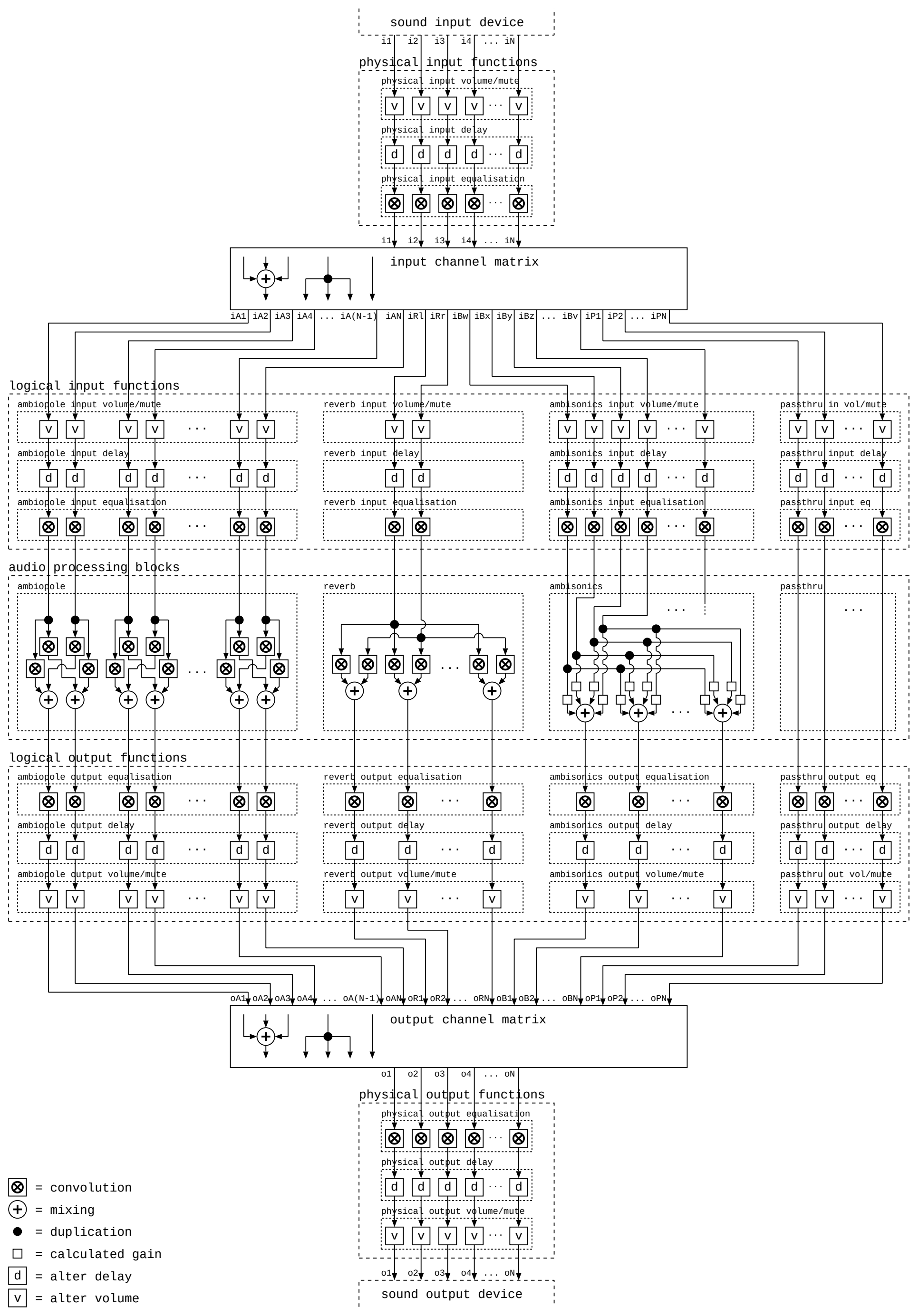

Figure 3.1: AlmusVCU block diagram

one ambiopole, which is the normal case. However, special applications such as PanAmbio, multi-room or car-audio installations require more than one.

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.

3.1.1.4 Reverb

The reverb audio processing block provides feeds for a reverb speaker array. The convolution filters in the reverb block are derived from a given reverb program. 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 block in common configurations has the most channels and the longest filters, it is normally the most processing intensive part of AlmusVCU.

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. It is also possible to configure the reverb audio processing to use only one input (corresponding to the center part of the stage), or with three inputs (left, right and center) and then the corresponding amount of filters per output will be used. Thus, reverb for three inputs is three times as demanding as one input.

The amount of reverb output channels is configurable.

3.1.1.5 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. However, more advanced Ambisonics decoders include psycho-acoustic filters on the input channels, which also AlmusVCU allows (through logical input equalisation), even allowing them to be modified in runtime. 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.

3.1.1.6 Passthru

The passthru audio processing block does as the name states, it simply passes through any signal that comes into it. The amount of passthru channels is configurable (there are none in the default configuration). Should any equalisation, delay or trim volumes be desired, these are applied through the standard input and output functions. Passthru channels could for example be used to provide a straight stereo or 5.1 mode, or sub-woofer channels.

3.1.1.7 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 automatically 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. Note that sample alignment is not shown in the block diagram.

3.1.1.8 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 (physical inputs).
- 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, iRc; left, right and center 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.
- o1, o2, o3, ... oN; the output channels of the sound card (physical outputs).
- 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 keys, 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). In runtime, some input devices allows for additional keys (see section 2.4.2), but all functions can be accessed using only the four basic direction keys, and will be the only ones referenced in this documentation.

3.1.2.1 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 one alternative. By pressing RIGHT the current 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 bottom one. The next section of the menu is then shown if pressing DOWN when pointing at the bottom 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.

```
+--------------------+
|  Load reverb prg...|
| >Del reverb prg... |
|  Tune 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.
- Enter a string. The current string is within brackets, and thus looks the same as the cycle alternative. However, when selected, an edit string mode is entered. A list of letters and numbers are scrolled through with UP and DOWN, and the current selected character is put to the tail of the string with RIGHT. Characters are deleted from the tail with LEFT. To finish or cancel, the list of letters is scrolled to one of its ends, and there the capital letters A for Accept and C for Cancel can be selected.

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

3.1.2.2 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 input to the user interface has stopped. 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 activate, it is left by pressing any of the four buttons (LEFT, RIGHT, UP or DOWN). The single 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?  |
|  Master [-12.0]dB  |
|  Master mute [off] |
|  Load reverb prg...|
|  Del reverb prg... |
|  Tune reverb prg...|
|  Signal setup...   |
|  Channel setup...  |
|  Eq setup...       |
|  Ambiopole setup...|
|  Reverb defaults...|
|  Convolver setup...|
|  User intf setup...|
|  System info...    |
|  Maintenance...    |
+--------------------+
```

In the following sections, each sub-menu in the setup menu pictured above is described. The menus are described in the same order as found in the setup menu. However, if a system is setup freshly from start, it is most natural to start with the channel setup menu (and then possibly the eq setup menu).

3.2.1 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 AlmusVCU is completely configured, which is not the first time it is started. If there is some configuration missing and the alternative is attempted anyway, an error message indicating the problem will appear.

If the convolver is configured for a new block size, or a new signal resolution, it may take some time before the convolver starts, due to optimisation activities.

3.2.2 Master volume control

The master volume control sets the overall output level of all channels. All other volumes (trim volume, reverb program volumes, channel group volumes etc) are relative to the master volume. There is also a mute control.

The master volume can of course also be changed in runtime. The reason for having it available in the setup menu is to make it possible to avoid any nasty surprises in the form of too high output volume when the convolver is started.

3.2.3 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.

3.2.4 Tune reverb program

The tune reverb program menu contains settings for each reverb program, used by the reverb engine. More information on how the reverb engine works is found in chapter 4.

The menu first displays all loaded reverb programs. When one of it is selected, the following menu is shown. In this example, the name of the reverb program is “The Berwald Hall”, which is displayed at the top of the menu. All reverb settings in this menu are saved uniquely per reverb program.

```
+--------------------+
|The Berwald Hall    |
| >Length [100]%     |
|  Predelay [0.0]ms  |
|  Volume [+0.0]dB   |
|  [Stage adapted]   |
| [Decorr as needed]|
| Force LRdelay[off]|
| Force decay [on]  |
| Mix with dry [off]|
| Dry vol [+0.0]dB  |
| Xbalance [+0.0]dB |
| Ybalance [+0.0]dB |
| Zbalance [+0.0]dB |
| B-format dir [1.4]|
| Properties...     |
+-------------------+
```

3.2.4.1 Length

Some reverb programs are recordings of venues with very long reverb time. For example, a cathedral can have more than 10 seconds long impulse responses. Sometimes it is desirable to shorten the reverb time, either because the computer with the current settings cannot handle the full reverb time and therefore must truncate the impulse responses, or that the reverb time is simply too long to sound good in the given application. It is almost always preferable to shorten cathedral reverb programs.

The length setting specifies the length in percent of full length. What the full length is can be seen in the "Properties" sub menu.

The shortening is done for each derived impulse response the following way: the volume of the original impulse response at the tail is measured, and then the response is truncated and a exponential decay is applied such as the new tail gets the original tail volume.

3.2.4.2 Predelay

This setting will add extra delay to the reverb program, previous to its direct sound. Normally, the proper delays are stored in the reverb program and handled by the reverb engine, so it should not need to be adjusted.

3.2.4.3 Trim volume

If a reverb program appears louder or attenuated compared to other loaded reverb programs, the trim volume setting can be used to adjust the overall volume of the program.

A properly constructed reverb program should have a reasonable suitable volume per default. Note that reverb programs with longer reverb times usually sound louder than those with short. This is also the case in reality.

The trim volume can be adjusted in runtime.

3.2.4.4 Early response setting

An impulse response recorded in a hall consists of three components, first the direct sound, followed by early reflections, and then the reverb tail.

The direct sound has the highest intensity and is what arrives first, and tells the listener the direction of the sound source. After that, distinct reflections from the boundaries of the hall arrives with a bit lower intensity and quite low density. These are the early reflections, and arrives during the first 50 ms. The early reflections give some idea of the room, and the distance to the source.

After 50 ms the density of reflections is high, and they arrive seemingly chaotically. This is the reverb tail which decays exponentially. The reverb tail of the impulse response is what is heard in between sound events, and gives the impression of a room.

The first 50 ms of the impulse response, that is the direct sound and early reflections can be altered the following three ways:

- Keep the full response. No alteration of the early response.
- Mute direct sound. Mutes the ten first milliseconds, with a soft fade-in at the end.
- Mute early response (direct sound and early reflections). Mutes the 50 first milliseconds, with a soft fade-in at the end.
- Stage adapted. The early response is muted for any reverb speaker which is within any of the defined stages. For the other speakers, the full response is kept. Stages are speaker placement angle ranges defined in the channel setup menu, and typically corresponds to an ambiopole.

Some reproduction formats, such as Ambiophonics, has specific reproduction of the stage (or stages if there is more than one), including direct sound and early reflection pattern (and reverb tail). In such cases it is important that the reverb array does not reproduce its early response in those directions. This can be achieved in two ways. The most obvious is not to place any reverb speakers within the stage, that is within a 150 degree arc if the stage is reproduced by an ambiopole. The other is to allow placement of reverb speakers within the stage, but set the early response setting to “stage adapted”. Then the early response will be muted for the speakers within the stage. Although duplication of the early response is avoided, the reverb tail will be reproduced by both the reverb speakers on stage, and the stage reproduction speakers. Although not strictly “correct” it is often beneficial, it blends the reverb array and stage reproduction together to a unity, even if there is not a 100% perfect match between reverb program acoustics and the one recorded from stage. This setting is discussed further in the Ambiophonics tutorial, in chapter 5. As noted, in order for this setting to work, a stage must be defined, which is done in the channel setup menu, described in section 3.2.6.

For recording post-production, it may be desirable to keep the full response, or perhaps more likely, mute the direct sound, and mix in the dry signal to avoid any colouring the impulse response direct sound may cause.

3.2.4.5 Decorrelation setting

If two mono impulse responses are recorded in a hall, a few meters apart, they will sound nearly the same. However, although they will have about the same magnitude responses throughout the full length of the responses, the phase responses will differ more and more farther back in the impulse responses.

When impulse responses are assigned to reverberation channels, it is very important that all impulse responses are decorrelated in the way described above. If not, strange directional effects could be heard.

If the reverb program used does not have a full 3D recording of the acoustics, there is a risk that the same impulse response must be used for more than one reverb channel. In an extreme case there may be tens of reverb channels, but only one recorded impulse response in the reverb program.

AlmusVCU therefore has a decorrelation algorithm which can decorrelate a single impulse response infinite number of times, and this way create new impulse responses. The algorithm tries to mimic nature, for example by increasing the degree of decorrelation farther back in the impulse response. However, the decorrelation algorithm cannot be as good as the real thing, so the more impulse responses there are in the original reverb program the better.

This decorrelation algorithm is controlled by three settings:

- Decorrelate none. Do not employ the decorrelation algorithm. This setting is normally only useful to be able to see what improvement decorrelation does with a specific reverb program.
- Decorrelate as needed. Only decorrelate when needed. This means “not at all” when the reverb program contains 3D impulse responses, and “whenever an impulse response must be re-used” for traditionally recorded reverb programs.
- Decorrelate all. Decorrelate all impulse responses. This could be used to force decorrelation onto a 3D impulse response reverb program.

Normally, this setting should be set to “Decorrelate as needed”.

3.2.4.6 Force left/right delay difference

This setting is only shown if the reverb engine is configured with at least two inputs (left/right), and if the reverb program has more than one recorded source (left/right).

If a stereo input signal to the reverb engine is playing in the left channel only, the reverb should typically be heard as a bit heavy to the left, that is there should be some directionality in the reverb program. However, if the early response has been muted (due to the early response setting), the directionality will typically be very weak, which can make the reverb program blend poorly with a direct sound stage. If this setting is activated, artificial delay difference between left and right is introduced, which makes the directionality of the reverb program re-appear.

If the reverb program is mono (only one recorded source position) it will have no directionality, even if the full early response is kept. If the reverb engine then is configured with at least two inputs, this setting is automatically activated, and not shown in the menu.

3.2.4.7 Force decay

Some reverb programs have a poor signal-to-noise ratio, which in practice means that the resulting reverb does not fade out well. If the force decay setting is activated, the signal-to-nosie ratio of the reverb program is found out, and a proper decay is applied, meaning that the natural exponential decay found in the reverb program is extended to either -160 dB or until the impulse response ends, whatever comes first.

Since a real world reverb impulse response far back in the tail basically is noise with exponential decay, this setting will produce very good sounding reverb even if the original reverb program has very poor signal-to-noise ratio, as long as the reverb program is long enough to fit the full decay, instead of being truncated when the noise floor was hit.

The force decay setting will not notably affect a reverb program which already has good signal-to-noise ratio, so it is recommended to keep this setting on (which is the default). The only real reason for not using it is to listen to the actual recording quality of the reverb program.

3.2.4.8 Dry volume settings

The dry volume settings are generally only usable for recording post-production. When a dry signal is to be reverberated in the direct sound channels, it is often desirable not to alter the timbre of the original recording.

The direct sound in a recorded reverberation impulse response is always a bit coloured, typically with some high frequency roll-off, since high frequencies are absorbed in the air. To avoid that colouration one can mute the direct sound (or the early response), and mix in the dry signal as the new direct sound. The standard setting with 0 dB attenuation of the dry signal should match the reverb program.

Note that the dry volume settings cannot be changed in runtime. If that is desired, the dry volume could be provided through a passthru channel mixed to the same outputs as the corresponding reverb channels.

3.2.4.9 Balance settings

The X, Y and Z balance settings are used to balance a reverb program which exhibit static directional effects in a reverb array. A positive/negative value on X will re-balance the reverb sound field to the front/back in the reverb array. Y balance concerns left/right, and Z up/down. The overall volume is not altered.

The balance settings are automatically adapted to the reverb array, which may be irregular. However, on a planar array, the Z setting will do nothing, just as the Y setting will do nothing on an array with all speakers on one side, and the X setting with speakers only in the front or back.

The Y (left/right) setting is typically the one with the strongest audible effect, since it affects the interaural level difference (volume difference between left and right ear).

When the early response is muted in a reverb program, the balance of the reverb sound field is often shifted back, and one may therefore want to apply a positive X setting.

These settings can be adjusted in runtime.

3.2.4.10 B-format directivety factor

The B-format directivety factor can only be adjusted for B-format-based reverb programs. It specifies the mix between the omni-directional W and the directional

X, Y, Z signals. If it is set to 0.0, only the omni-directional part is used, and thus all derived impulse responses will be equal. If it is set to 2.0, the omni-directional part is removed. The default value, which is 1.4, should provide a suitable mix in most cases.

Translating it to microphone terms, 0.0 represents omni-directional, 0.5 subcardioid, 1.0 cardioid, 1.5 hyper-cardioid, and 2.0 represents a figure-of-eight microphone response. An impulse response for a reverb speaker is derived from the B-format reverb program by calculating the impulse response which a microphone in the direction of the speaker, and with the given directivety, would get.

3.2.4.11 Properties menu

The properties menu lists properties of the current reverb program:

```
+--------------------+
| Orig length: 4.3s  |
| Curnt length: 4.3s |
| Orig RT60: 2.1s    |
| Curnt RT60: 2.2s   |
| Orig tail: -119dB  |
| Curnt tail: -120dB |
| Type: traditional  |
| Source: stereo     |
| Recv 1: 45/0       |
| Recv 2: -45/0      |
| Recv 3: 135/0      |
| Recv 4: -135/0     |
| Rate: 44.1kHz      |
| Format: S24_LE     |
+--------------------+
| .
| .
| Type: b-format     |
| Source: stereo     |
| Rate: 44.1kHz      |
| Format: FLOAT_LE   |
+--------------------+
```

First a set of original and current values are listed. The original length is the reverb impulse response length stored in the reverb program file, the current length is the length used in the reverb engine. The current is shorter if the reverb program has been truncated, or the length setting is lower than 100 percent.

RT60 is the time it takes for the reverb to decay 60 dB. It is calculated for the original unmodified reverb program, and the actually used in the reverb engine. These two can differ slightly even if the length has not been altered, due to decorrelation. If the value is 0.0, it means that the impulse response does not decay 60 dB before it has ended. This usually means that the reverb has been truncated, or that the original reverb program has poor signal-to-noise ratio.

If the force decay setting is activated, the current tail volume is typically considerably lower than the original, if not activated the current and original tail volumes are about the same. However, if the reverb program has been truncated, the current volume will be higher than the original. The current tail volume should be at -100 dB or below in order to sound well.

After the original/current values, there is a listing of reverb program recording properties. The type is either “traditional” or “b-format”. Detailed information on these reverb program types is found in chapter 4.

3.2.5 Signal setup

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

```
+--------------------+
| >Input [sound card]|
|  PCM file setup... |
|  [16] bit output   |
|  [32] bit internal |
|  Dither [on]       |
|  Watch clock [on]  |
|  Allow 44.1kHz[on] |
|  Allow 48.0kHz[on] |
|  Suggest [44.1]kHz |
+--------------------+
```

3.2.5.1 Input selection

Normally, the input to AlmusVCU should be the statically configured sound card. However, due to popular demand, a rather hackish feature has been implemented – it is also possible to play from a PCM file.

AlmusVCU is currently not designed as a media player for stored files, so the function to play from a file is not meant to be very polished or practical, just a fallback option when the sound source cannot be delivered into the sound card.

3.2.5.2 PCM file setup

The PCM file setup menu is used to select a raw PCM file when it is used as input instead of the sound card.

```
+--------------------+
|"schnittke.pcm"     |
| >Select file...    |
|  Format [S16_LE]   |
|  Channel count [2] |
|  Loop [off]        |
+--------------------+
```

The format of the raw PCM file must be specified, which is done with the format and channel count settings. The PCM file must have interleaved layout of the channels.

A description of the sample format abbreviations is found in section 10.1.

Selectable files are found on the configured rom devices, and the file name must have the suffix `.pcm` or `.raw`. The current selected file will show at the top of the menu, in the example above, the file `schnittke.pcm` has been selected.

The channels in the PCM file will be mapped to a virtual sound card with the same amount of channels as the actual configured sound card. If there are fewer channels in the file, the channels will be mapped to the lower indexes, and the remaining will generate zeroed signals. If there are more channels in the file, only the channels up to the amount of the configured sound card will be used.

If the loop option is on, the input file will in runtime loop over and over again until the convolver is manually stopped. If the loop option is off, the convolver will exit back to the setup menu when the file has been played once.

3.2.5.3 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.

3.2.5.4 Internal resolution

The convolver makes all its calculation with floating point values. These can either be 32 bit (single precision) or 64 bit (double precision). Double precision makes filter programs use up twice the amount of memory compared to single precision. The convolver will also consume about twice the amount of memory, and run considerably slower. Thus, employing 64 bit internal resolution is costly in terms of memory and throughput, and is generally not necessary. Although not audible for most people, the calculations will be more precise with 64 bit resolution, and it allows adding dither to 24 bit output (which is not possible if internal resolution is 32 bit).

3.2.5.5 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.

This setting is only available if it is possible to set dither. If the output is set to be 24 bit and the internal resolution is 32 bit, dither cannot 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.

3.2.5.6 Watch sample clock

If the statically configured sound card is using a digital input, the sample clock will be decided by the digital source. This means that the sample clock could be changed in runtime. For example, the digital source could be playing a CD at 44.1 kHz when the convolver starts, but later change to a DAT recording at 48 kHz, and thus change the sample clock. If there are any filters loaded, such as equalisers, crosstalk cancellation and reverb programs, they are sampled for a specific sample rate. Thus, if the sample rate change, the filters will be in error. If the sample rate change is quite small, such as from 44.1 to 48, it may be bearable.

The perfectionist should however activate the "watch clock" setting, which will cause the convolver to abort if the sample rate is changed, it can then be restarted with filters for the proper sample rate loaded. If the new sample rate is supported in the configuration, the convolver will automatically restart, thus there will be only a couple of seconds of interruption at sample rate change.

3.2.5.7 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 or DAB receiver at 48 kHz.

If the sound card input is analog, that is it is sampled with an A/D converter, it normally makes little sense to select more than one sample rate. However, if the sample rate on the input should be changed often, and one want to avoid reloading of filter programs (which can take a few minutes in worst cases), one should select the sample rates with interest.

Note that it is not possible to de-select all sample rates, at least one will be set to active at all times.

3.2.5.8 Suggested sample rate

This setting is only available if the sound card supports more than one sample rate, and more than one sample rate has been selected. The suggested sample rate is what AlmusVCU will try to run the convolver in. However, if the input is digital (such as S/PDIF or ADAT) and there is a signal on it, the suggested sample rate will be ignored, and the sample rate will be what the sample clock is on the input instead (the sound card must be configured to operate in slave mode to behave as expected).

For analog inputs, the sample rate will always be set to the suggested.

3.2.6 Channel setup

The channel setup menu is used to configure how many logical output channels that should be used, to which sound card output they should be linked and so on. Together with the equaliser setup menu this menu defines what AlmusVCU will actually 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    |
|  Reverb setup...   |
|  Ambisonic setup...|
|  Input matrix...   |
|  Output matrix...  |
|  Speaker layout... |
|  Stage layout...   |
|  Channel groups... |
|  Channel modes...  |
|  Trim volume in... |
|  Trim delay in...  |
|  Trim volume out...|
|  Trim delay out... |
+--------------------+
```

3.2.6.1 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. The total amount of channels cannot exceed 64, so if the number of ambiopoles are maximised (that is set to 32, two channels per ambiopole), the other types will be set to zero automatically.

3.2.6.2 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 physical 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.

The total amount of channels cannot exceed 64, so if the number of reverb channels are maximised, the other types will be set to zero automatically.

3.2.6.3 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.

The total amount of channels cannot exceed 64, so if the number of ambisonic channels are maximised, the other types will be set to zero automatically.

3.2.6.4 Passthru channel amount

This is the number of passthru channels, seen in the “passthru” block in the block diagram in figure 3.1. This can be set to zero, which corresponds to removing the whole passthru block from the block diagram.

The total amount of channels cannot exceed 64, so if the number of passthru channels are maximised, the other types will be set to zero automatically.

3.2.6.5 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]|
+--------------------+
```

In a B-format signal, the W channel is traditionally attenuated 3 dB compared to X, Y and Z, and AlmusVCU expects that on the input. The reason for having this attenuation is historical; it made it possible to make better use of the available dynamic range in the first microphones. Although this is no longer necessary with the technology of today, the 3 dB attenuation has been kept for backwards compability, so B-format material produced today is coded the same way, and thus the AlmusVCU user does not need to care about it. This 3 dB W attenuation is also used in second order Ambisonics.

3.2.6.6 Input matrix

The input matrix menu maps directly to the “input channel matrix” block found in the block diagram in figure 3.1. In this menu it is specified how the logical input channels should be mapped to the physical input channels.

The menu will show one entry for each logical input channel available in the current configuration. In the example below, there are two ambiopole channels, and one passthru channel. After the logical channel name, the physical inputs that are linked to the logical channel are listed within parentheses. If the number of assigned physical channels exceed six, only the number of links are stated though. There is always at least one link to a physical channel for each logical channel.

```
+--------------------+
| >iA1 (i1,i3)...    |
|  iA2 (i1)...       |
|  iP1 (8 links)...  |
+--------------------+
```

As indicated by the trailing punctuation marks, each entry is itself a menu, in which the links can be set:

```
+--------------------+
| >link iA1-i1 [on]  |
|  link iA1-i2 [off] |
|  link iA1-i3 [on]  |
|  link iA1-i4 [off] |
|  .
|  .
```

The menu contains one entry for each available physical input. Each entry corresponds to a link between the given logical channel (iA1 in the example) and the physical channel, which can be turned on and off. Thus, it is possible to link just one or every physical channel to the logical. However, not more links than actually needed should be established, since it consumes resources.

If the total amount of links gets too high, the convolver may fail with a “too complex configuration” error when it is attempted to be started. If that happens, the number of links must be reduced. The limit is quite high though, so for any normal configuration the error should not occur.

3.2.6.7 Output matrix

The output matrix menu corresponds to the “output channel matrix” block found in figure 3.1. Here each logical output channel is linked to one or more physical output channels. The menu works the same as the input matrix, described in section 3.2.6.6.

3.2.6.8 Speaker layout

The speaker layout menu specifies where the speakers are situated relative to the listening position. It is important to specify the layout correctly, since it is used in calculation of the actual impulse responses 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?     |
|  Specify layout... |
+--------------------+
```

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), and deactivate each position.

The specify layout sub-menu looks like this:

```
+--------------------+
| >o1 position [on]  |
|  o1 a[-45.0]deg    |
|  o1 e[30.0]deg     |
| o1 d[130.0]cm     |
| .
| .
```

The speaker layout identifies the physical output channels, it is thus assumed that there is a speaker attached to each output. The first entry for each physical channel indicates if the speaker position is active or not. If there is no reason to specify a speaker position for the output, it could be deactivated to indicate that (although an active unused speaker position does no harm). Settings for angle (a), elevation (e) and distance (d) follows. The 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.

3.2.6.9 Stage layout

Some sound systems have specific directions where direct sounds will be reproduced from. This could be called a stage, since it often is, if the reproduced sound is music. Ambiophonics is such a system, where the ambiopole reproduces the frontal stage. The reverb engine can take advantage of knowing where the stage (or stages) is, as described in section 3.2.4. Thus, if the reverb engine is used, it is a good idea to specify any stages the sound reproduction system has.

The center of the stage is given through azimuth and elevation (same interpretation as in the speaker layout menu), and range is given through width and height angles. The stage angles are used by AlmusVCU to check if a speaker is within a stage or not. A speaker is defined to be within the stage if its specified position is within or on the edge of the stage range. Thus, if the stage center is azimuth and elevation zero, and width is 150 degrees and height 40 degrees, any speakers with azimuth from -75 to +75 degrees and elevation -20 to +20 degrees are considered to be within the stage.

The width and height of the stage should roughly correspond to the actual range where the system can produce direct sounds. The recommended stage size for an ambiopole is 150 degrees wide and 40 degrees high (the documentation for the assigned crosstalk cancellation program may suggest another value though). For plain stereo or other phantom-imaging setups, the stage width should be as wide as the speakers are placed, that is 60 degrees for stereo, and height 40 degrees.

```
+--------------------+
| >S1 azim [0]deg    |
|  S1 elev [0]deg    |
|  S1 width [150]deg |
|  S1 height [40]deg |
|  Add stage?        |
|  Delete last stage?|
+--------------------+
```

The menu will list all available stages (may be none) with their four configurable entries, azimuth, elevation, width and height. The first stage is named “S1”, the second “S2” and so on. At the end of the menu, there is an entry to add another stage, which is available as long as the stage table is not full (there cannot be more than eight stages), and to delete the last stage in the list, which is available if there is one or more stages available.

3.2.6.10 Channel groups

AlmusVCU has an overall master volume control, and trim volume settings for each logical and physical input and output. However, for almost all configurations it is worthwhile to have separate volume/mute controls for larger groups of channels. One example is that for Ambiophonics it is very useful to be able to control the reverb volume separately.

Channels are grouped together for volume control in the channel groups menu. Channel groups only concern logical channels, so physical channels cannot be grouped together, and there is no reason to.

The channel groups menu shows a listing of all currently available channel groups (may be none), an option to create a new group, and an option to reorder the groups. The listing order of the groups defines in which order the channel groups will be presented throughout the user interface, so it may be desirable to change the order. In the example menu below, the two channel groups called “a-pole” and “reverb” has been created and are thus available for editing.

```
+--------------------+
| >Edit "a-pole"...  |
|  Edit "reverb"...  |
|  New channel grp...|
|  Reorder groups... |
+--------------------+
```

A channel group has a name, a volume and mute setting, and a list of channels it controls. Controlling a channel means applying the mute and volume settings the group has to the channel. The group name is what will be visible in the user interface in runtime, and must be short, only up to seven characters are allowed, to fit into the compact text user interface. The volume and mute setting can be controlled in runtime.

Editing/creating a channel group

Below is the menu shown when editing or creating a new channel group.

```
+--------------------+
| >Name [a-pole]     |
|  Volume [-12.0]dB  |
|  Mute [off]        |
|  Inputs (0)...     |
|  Outputs (2)...    |
|  Delete this group?|
+--------------------+
```

If a new group is created, the group sub menu ends with the entry `Create this group?`, if an existing is edited it is replaced with `Delete this group?`. The numbers in the inputs and outputs entry shows how many channels that are controlled, in the example above there are zero inputs and two outputs controlled. The inputs and outputs sub menus simply show a list of all available logical channels, where one can turn on or off control for each:

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

Recommendations

There is total freedom how to group channels together, however, to avoid chaos and confusion, there are a few recommendations to follow:

- It is common that a channel group exactly matches one audio processing block, that is for example all reverb channels. If that is the case, the recommended name of the group is for ambiopole “a-pole”, reverb “reverb” ambisonics, “a-sonic”, and passthru “p-thru”.
- Control output channels only, if there is no special reason to control input channels. Since the processing is done in floating point internally, there is no risk for overflow, thus there is generally no reason to control the input instead of outputs.
- There should be enough groups so that all logical channels belong to one group, even if some groups will generally not be used for volume control. If a channel group is not desired to be directly available for volume control, it control can be deactivated in the user interface menu described in section 3.2.11.
- Overlapping groups and giving the same name to more than one group only belong in advanced channel mode arrangements (further described below).
- Avoid situations where one group controls a logical output, and another the corresponding logical input.

Overlapping groups

There is nothing that stops configuring groups that overlaps in channel assignment, that is two groups that both control the same channel. It is also possible to assign the same name to several groups. This can be useful in special channel mode arrangements (see the documenation for channel modes), but can cause great confusion if not used properly. If two channel groups control one channel (and both are active in the current channel mode), both their volumes will be added together, and only one group need to be muted to mute the channel.

3.2.6.11 Channel modes

For some HiFi systems, it is desirable to change between vastly different configurations by just pressing a key. For example, one may want to switch to an Ambiophonics configuration for music listening, and to a 5.1 home theater configuration for watching movies. Another example is if AlmusVCU is used in research, and one want to be able to do immediate A/B testing of different sound reproduction formats, for example to compare Ambisonics and Ambiophonics.

To achieve this, AlmusVCU is then configured to run all setups at the same time, which usually means that some channels are mixed to the same speakers (for example reverb and ambisonic channels in case of concurrent Ambiophonics and Ambisonic configurations). Then, to select one configuration, all channels belonging to the other configuration(s) are muted. This would be cumbersome if the mute channels menu would be used, where one by one channel is muted. Instead, the channel modes menu exists solely for this purpose.

Note that if AlmusVCU only is supposed to run one configuration, it is not necessary to set up any channel modes at all.

In the channel modes menu, there is a listing of available channel modes (may be none), an entry which shows the current active channel mode, an option to create a new channel mode and an option to reorder the modes. The order of the channel mode listing is used throughout the user interface, therefore it may be worthwhile to reorder them. To use channel modes, there must be at least two of them available.

```
+--------------------+
| >Mode [ambisonics] |
|  Edit "ambisonics".|..
|  Edit "ambiophonics|...
|  New mode...       |
|  Reorder modes...  |
+--------------------+
```

Editing/creating a channel mode

A channel mode is simply a listing of which channel groups that should be active, and a name of the channel mode (max 20 characters). Channel modes only have effect in runtime. Then, when a specific channel mode is active, only the groups included in it will be visible in the user interface, and all other groups are muted. This allows to effectively switch between for example an Ambiophonics and an Ambisonics decoding mode. Below is an example of a channel mode sub menu, in a configuration which has the channel groups “a-sonic”, “a-pole” and “reverb”. The channel mode is named “ambisonics”.

```
+--------------------+
| >Name [ambisonics] |
|  "a-sonic" [on]    |
|  "a-pole" [off]    |
|  "reverb" [off]    |
|  Delete this mode? |
+--------------------+
```

Note that if the current mode is set to off, all channel groups will be active at all times, and it is then not possible to activate a channel mode in runtime. If there is a current channel mode, the active groups will in runtime be as those specified by the current channel mode, and the channel mode can be changed in runtime, but it is not possible to turn them off (setting current mode to off).

Recommendations

Just as for channel groups, there is total freedom how to set up channel modes, but to avoid chaos, there are a few recommendations:

- Name the channel mode to the surround format it represents, which for AlmusVCU typically is “Ambiophonics”, “Ambisonics”, “ITU 5.1” and more.
- Do not use the same name for more than one channel mode.
- Strive to have more than one channel group in a channel mode. For example an ITU 5.1 channel mode could contain separate channel groups for stereo, center, subwoofer and surrounds. However, in some cases it is not natural having multiple channel groups, for in an Ambisonics array, and then one group per channel mode is ok.
- It may be desirable to use one channel group in more than one channel mode, but avoid having the exact same channel group configuration in two separate channel modes. This would only mean to have a redundant channel mode.
- Some channel mode configurations may require overlapping channel groups. However, avoid having overlapping channel groups within one channel mode.
- Only set up separate volume controls for the same group of channels if strictly necessary (further described below).

Using overlapping channel groups

In some situations it may be desirable to have different channel group volume controls in different channel modes. For example, for a channel mode called “Perambio” which uses an ambisonic array it may be suitable to set that to -6 dB, while for a pure “Ambisonics” channel mode it is suitable to have it at 0 dB. If both channel modes use the same channel group for the ambisonic array, there will be a need to change its volume each time the channel mode is changed, which is cumbersome.

To avoid this, the recommended way is to create two identical channel groups, with the same name, and assign one to each channel mode. In the example, the name of the ambisonic array will then stay the same for both “Perambio” and “Ambisonics” channel modes, thus avoiding confusion, and there will be separate volume controls. However, while easy to use once set up, it may be confusing while setting it up. Luckily, in most cases, it is not necessary to have separate volume controls for the same group of channels in different channel modes, and then only one channel group is needed.

3.2.6.12 Input trim volume

The input trim volume menu allows setting offset volumes and muting of individual physical and logical 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 volume/mute blocks in the logical and physical input functions blocks.

The intended use for the trim volume is to compensate for heterogenous source signal levels. It is typically tuned once and then not changed. Muting of indivudal channels is most often used only temporarily, for example to help identify specific inputs.

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

```
+--------------------+
|Trim volume in      |
| Logical (0)...     |
| >Physical (3)...   |
+--------------------+
```

The first menu entered only gives the choice if physical or logical channels should be accessed. It also gives an overview of how many channels that are muted or have a non-zero offset volume. In the example above, three physical channels are either muted or have a non-zero offset volume (or both), and there are no trim volume settings or muting for logical channels. The physical and logical sub menus work the same way. Below is an example of the physical sub menu.

```
+--------------------+
| >mute all?         |
|  unmute all?       |
|  i1 [+0.0]dB       |
|  i1 mute [off]     |
|  .
|  .
```

The mute all and unmute all options are macro operations to activate/deactivate muting for all channels in the list. The list contains all (relevant) physical channels, and a trim volume and mute setting for each of them.

If mute is set to on for a specific channel, it will produce a silent input signal. Thus it will still be active, so the processing time required for the channel is not altered by muting it.

3.2.6.13 Input trim delay

The input trim delay menu allows setting the delay of individual logical and physical input 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 input trim delay functionality is shown in the "delay" blocks of the logical and physical input functions blocks.

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.

```
+--------------------+
|Trim delay in       |
| >Logical (2)...    |
|  Physical (0)...   |
+--------------------+
```

The first menu entered only gives the choice if physical or logical channels should be accessed. It also gives an overview of how many channels that are have a non-zero delay. In the example above, there are no physical delay settings, and there are two logical input channels with non-zero delay. The physical and logical sub menus work the same way. Below is an example of the physical sub menu.

```
+--------------------+
| >i1 [0.00]ms       |
|  i2 [1.22]ms       |
|  .
|  .
```

The sub menu simply contains a listing of all relevant channels and their current delay setting.

3.2.6.14 Output trim volume

The output trim volume menu controls trim volume and mute settings for individual physical and logical output channels. In the block diagram in figure 3.1, the output trim volume functionality is seen in the “volume/mute” blocks of the logical and physical output functions blocks.

The menu is operated the same way as the input trim volume menu, which is described in section 3.2.6.12.

3.2.6.15 Output trim delay

The output trim delay menu sets trim delays for individual physical and logical output channels. The function is seen in the block diagram in figure 3.1, as the “delay” blocks within the logical and physical output functions blocks.

The menu is operated the same way as the input trim volume menu, which is described in section 3.2.6.13.

3.2.7 Equaliser setup

The equaliser setup menu is used to design and and assign equalisers to channels. In the block diagram in figure 3.1, the equalisation functionality is seen as the equalisation blocks in physical and logical input and output functions blocks. Per default, equalisation is disabled, which equals to removing the equalisation blocks from the block diagram.

Equalisers can be used for many purposes, such as bass management, room and speaker equalisation, or any other generic filtering task.

```
+--------------------+
| >Eq design...      |
|  Eq modes...       |
|  Load custom...    |
|  Del custom...     |
+--------------------+
```

3.2.7.1 Load/delete custom equalisation program

The `Load custom` and `Del custom` menus allow to load and delete custom equaliser programs respectively, which can used when designing equalisers. A custom equaliser program is simply a FIR (Finite Impulse Response) filter in the AlmusVCU filter program file format. Custom equalisers could for example be room equalisation filters designed in another application. For basic filters, the built-in equaliser designer could be used instead of loading custom equalisers though.

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.

3.2.7.2 Equaliser design

The equaliser design menu is used to create equalisers, that is FIR filters, that can be assigned to the logical and physical input and output channels. The assignment itself is done in the equaliser modes menu. If there are no equalisers, the will be only one alternative in the menu, `New equaliser` which is used to create a new equaliser. If there are one or more equalisers, there are two entries for each of them, one to edit the equaliser, and one to view its properties. An additional entry is added to the bottom, `Copy equaliser` which is used to make a copy of an existing equaliser.

```
|  .
|  .
|  Edit "eq1"...     |
|   properties...    |
| >New equaliser...  |
|  Copy equaliser... |
+--------------------+
```

When the new equaliser alternative is chosen, a menu with the available filter types to choose from is opened:

```
+--------------------+
|  Lowpass...        |
|  Highpass...       |
|  Bandpass...       |
|  Bandstop...       |
|  Allpass...        |
|  Random phase...   |
| >Custom...         |
|  FFT static...     |
|  FFT dynamic...    |
+--------------------+
```

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 static and dynamic FFT-designed equaliser. The dynamic type can be modified in runtime. For further details on the different filter types and their options, see section 3.4.

Once a filter type has been chosen, and its parameters are set and the new equaliser is created, the user interface returns to the main equaliser design menu, at the entry for editing the newly created equaliser. Each equaliser has a name, which is set to “eqX” per default, where X is an index.

Equaliser stack

Each equaliser can be built from a multitude of filters, for example two highpass filters can be added together to form a shelfing effect. To allow maximum flexibility, the equaliser has a stack of separate filters and operations, which are combined through Reverse Polish Notation to the final filter used in equalisation. A newly created equaliser has only one filter, and that will be the final equalisation filter. However, entering the edit menu for that equaliser will allow adding new filters and operations to the stack. Below is an example of a shelf filter:

```
+--------------------+
|  lowpass 500 Hz (0.|50)...
|  op: multiply (-6.0| +)...
|  highpass 500 Hz (0|.50)...
|  op: add...        |
|  Name [shelf]      |
|  Add a filter...   |
|  Add an op...      |
| >Reorder stack...  |
+--------------------+
```

Reverse Polish Notation is a way of expressing arithmetic expressions that avoids the use of parentheses to define priorities for evaluation of operators. It is used in some scientific calculators and programs. It is easiest explained with the example given above. The stack is read from the top and down, and starts with a lowpass filter, then a multiply operation, then a highpass filter and finally an add operation.

Starting from the top, there is a lowpass filter, and then a multiply operation, which is applied to the result so far on the stack, which simply is that lowpass filter. Thus the new result is a lowpass filter where each sample has been multiplied with -6.0 dB as the operation states. Further down the stack, there is a highpass filter, and then an add operation, which requires two results to be added together, which will be the scaled lowpass filter and the highpass filter. This will be the final result, as the end of the stack is reached.

Reverse Polish Notation (RPN) may seem cumbersome to the beginner, but is very elegant once grasped. The main advantage is that it is never necessary to use parentheses to define which order things should be calculated. An example from the world of mathematics, (3 + 5) * (7 - 2) is written like 3 5 + 7 2 - * in RPN. Note that for each operation, the stack changes, so in the example 3 5 + 7 2 - *, when reaching to the add operation the stack becomes 8 7 2 - *, and then 8 5 * when the subtract operation is evaluated, and then finally the end result 40 afther the final multiplication. The equaliser stack only shows the starting point, so the consecutive changes when the stack is calculated must be kept in the head of the designer.

It is easy to end up with an invalid stack, that is a stack that does not come to an end result. The `Reorder stack` sub menu is used to reorder the stack if it is incorrectly ordered.

The `Add a filter` sub menu brings up the same menu as shown when a new filter is created, that is a list of the available filter types. The new filter will be put to the bottom of the stack.

The `Add an op` sub menu is used to add an operation to the bottom of the stack. There are four operations:

- Convolve, convolves two results together to form a new one. If filter results are seen as numbers, the convolve operation can be seen as the multiply operator in basic mathematics.
- Add, adds two results together to form a new one. The results are sample aligned before added together.
- Subtract, subtracts one result from the other. Like the add operation, the results are sample aligned before subtracted.
- Multiply, multiplies all samples in the result with the given amplification (or attenuation). Optionally, the sign can be changed as well.

Each entry in the stack has a sub menu, which at the very least contains the option to delete the entry, and if there are parameters to set for the operation or filter, those can be changed. Section 3.4 contains further information about the filter design menus.

Properties menu

The equaliser properties menu available in the top level equaliser design menu, one for each equaliser, has no settings in it. It simply shows the delay, peak amplification and amplification in 31 ISO 1/3 octave 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 among the frequency bins of the frequency domain transform of the filter.

The 31 band frequency response gives 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. To give as good view as possible of the filter, there are two modes to show the bands, in min/mean/max mode and in in/thru/out mode. The mode is chosen in an intermediate menu shown when selecting the equaliser properties entry. Below shows a truncated version of the properties menu for a 2 kHz low pass filter with soft rolloff in min/mean/max mode:

```
+--------------------+
| Eq "eq1":          |
|  delay: 0.8ms      |
|  peak amp: +0.1dB  |
|  ISO 1/3 oct bands:|
| band min mean  max |
|   20 +0.1 +0.1 +0.1|
| .
| .
|  630 +0.0 -0.0 +0.0|
|  800 -0.1 -0.1 -0.0|
| 1000-0.3 -0.5 -0.1 |
| 1250-0.9 -1.4 -0.5 |
| 1600-2.5 -3.9 -1.4 |
| 2000-6.4 -9.5 -3.9 |
| 2500-15.7-23.4-9.6 |
| .
| .
| 16k -82.7-86.5-79.3|
| 20k -95.0-lim -86.3|
+--------------------+
```

The frequencies show the center frequency of the band, and the amplification in dB, minimum, mean and maximum over the whole band. The in/thru/out viewing mode has the same layout, but shows the amplification at entry of the band, through the center of the band, and out of the band. If the equaliser has a soft shape, the in/thru/out mode is often more suitable to get an idea of how the equaliser is shaped. If the amplification exceeds plus or minus 99.9 dB, +/-lim is shown instead.

3.2.7.3 Equaliser modes

The equaliser modes menu is used when assigning the designed equalisers to channels. To allow for example A/B testing, "equaliser modes" are specified, which then can be changed in runtime. An equaliser mode is a specification of which equalisers should be assigned to which channels.

Creating/editing an equaliser mode

The top level equaliser mode menu has an entry called `New mode` which is used when creating a new equaliser mode, when it is selected, the following sub menu is shown:

```
+--------------------+
|  Name [unnamed]    |
|  Inputs (0)...     |
|  Outputs (0)...    |
| >Create this mode? |
+--------------------+
```

The name is a short name (max 7 characters) which will identify the equaliser mode in the user interface. The input and output sub menus are used to assign equaliser to actual channels. The numbers within parentheses in the input and output entries are the number of assigned equalisers.

Both the input and output sub menus has an intermediate menu where it is chosen whether to assign equalisers to logical or physical channels:

```
+--------------------+
|Assign equalisers   |
| >Logical (0)...    |
|  Physical (4)...   |
+--------------------+
```

Should any equalisers be assigned, the amount is shown in the number within parentheses. The assignment menu found under the logical and physical entries contains a list of all associated channels, with their equaliser assignments:

```
+--------------------+
| >o1 [off]          |
|  o2 [eq1]          |
|  o3 [shelf]        |
|   .
|   .
```

The above example shows the physical output equaliser assignment menu, where the first output, o1, has no equalisation (assignment set to “off”), the second output is assigned an equaliser named “eq1”, and the third an equaliser named “shelf”. The equalisers available for selection is those created in the equaliser design menu, described in section 3.2.7.2.

Assigning equaliser modes

The assignment of equaliser modes is different depending on the amount of equaliser modes, and if channel modes (described in section 3.2.6.11) are used or not.

When there are no equaliser modes created, there is no mode to assign, and thus it is not possible to activate equalisation. When there is only one equaliser mode, equalisation can either be turned on or off, meaning that either the equaliser mode is applied, or equalisation is not used. The example below shows the eq modes menu when one equaliser mode has been created, given the name “default”:

```
+--------------------+
| >Equalisation [on] |
|  Edit "default"... |
|  New mode...       |
+--------------------+
```

If there is more than one equaliser mode, equalisation can be set to off, or to one of the available equaliser modes. Note that the equalisation setting can be changed in runtime as well.

When there are multiple channel modes, equalisation modes must be assigned per channel mode, so the `Equalisation [on/off]` entry in the example above is replaced with the sub menu `Assign eq modes`, which can look like this:

```
+--------------------+
| >ch modes off...   |
|  "ambiophonics"... |
|  "ambisonics"...   |
+--------------------+
```

In the example above, there are two channel modes, named “Ambiophonics” and “Ambisonics”. The top alternative, `ch modes off`, is for the case when channel modes are turned off. In the `ch modes off` sub menu, the assignment is the same as the case when channel modes are not used, that is equalisation is turned on or off if there is only one equaliser mode, or if there are more than one, the equaliser mode to use is picked.

For the channel mode sub menus however, it is both selected which equaliser mode to use, and which that should be active for that channel mode. The option to select which equaliser modes to use per channel mode, allows deselecting equaliser modes that has no relevance for a particular channel mode. However, the most obvious cases AlmusVCU takes care of automatically, that is if the channels used in the channel mode has no assigned equalisers in an equaliser mode, that equaliser mode is forced to be deselected. Below an example what the menu could look like:

```
+--------------------+
|"ambiophonics"      |
| >Eq [rev-eq]       |
|  "rev-eq" [on]     |
|  "asx-eq" off      |
+--------------------+
```

In this example, the channel mode is named "ambiophonics", and there are two equaliser modes, named "rev-eq" and "asx-eq". The first, "rev-eq", has equalisers assigned to channels that are found in the "ambiophonics" channel mode, and thus it can be selected, which it has in this example (it is set to "on"). However, the "asx-eq" equaliser mode has no assigned equalisers for the channels used by the "ambiophonics" channel mode, and thus there is no point in including it for this channel mode. Therefore AlmusVCU has forced it to be deselected, shown by that there are no brackets around "off", meaning that it cannot be changed to "on".

The current active equaliser mode has in this example been set to "rev-eq", as seen in the first entry in the menu. It is allowed to deselect all equaliser modes for a particular channel mode; that channel mode will then have no equalisation configuration at all.

Note that it is perfectly ok if the equaliser mode has some equalisers assigned to channels that is not used by the given channel mode. Those equalisers will not be used for that channel mode though.

Recommendations

- Often the same effect is achieved if an equaliser is assigned to the corresponding input channel instead of the output channel. The recommendation is to use the output channel in those situations.
- If there will be only one equaliser mode, the recommended name for that mode is "default".
- If channel modes are used, still try to minimise the number of equaliser modes, meaning that if it is practical to use one equaliser mode in several channel modes, it should be done. For example, if no channel mode should have the possibily to change equaliser mode, this means that it is possible to have a single equaliser mode for all channel modes.

3.2.8 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 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 selecting 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 5.7 it is described how to physically set up an ambiopole to match standard cross-talk cancellation programs.

3.2.9 Reverb defaults

When a reverb program is loaded for the first time, a new configuration file for that program is created, which will contain the default settings as specified in this menu. Thus, it is recommended to have the settings matching what is desired for the current use of the reverb engine.

```
+--------------------+
| >[Stage adapted]   |
|  [Decorr needed]   |
|  Force LRdelay[off]|
|  Force decay [on]  |
|  Mix with dry [off]|
|  Dry vol [+0.0]dB  |
|  B-format dir [1.4]|
+--------------------+
```

The settings correspond to those that can be tuned per reverb program, and those are described in section 3.2.4.

3.2.10 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 components 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. 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 large concert hall is typically around five seconds. When a reverb program is loaded, and it is longer than the configured reverb time, it is truncated to the maximum length allowed.

The task of finding the right I/O-delay can be made manually, or by using an automatic benchmark which finds out what the hardware can handle with the current configuration. The benchmark and manual settings exist 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.

```
+--------------------+
| >Power save [on]   |
|  Benchmark (off)...|
|  Manual (on)...    |
+--------------------+
```

The power save setting will make the convolver consume very little processor time if there is no input to it. One of few reasons to not have it active would be to test if the computer running AlmusVCU can stand the power consumption for long periods of time, without the need to feed it with a signal.

3.2.10.1 Benchmarking with reverb

In the benchmark menu, AlmusVCU can run a benchmark with the current configuration, to find out which I/O-delay/reverb time alternatives that exist. 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, but 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.

The maximum length value is the maximum reverb time the benchmark will try to achieve.

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. A processor with thermal throttle and which is overheated can also trigger a buffer underflow (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 underflow”.

```
+--------------------+
| >Max load [85]%    |
|  Max length [24.0]s|
|  Low latency [off] |
|  Run benchmark?    |
+--------------------+
```

When the benchmark is run, screens with the progress will be shown. These screens exist in a four-row version and two-row version 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.

If there was not enough memory for a certain I/O-delay reverb time combination, it will be printed instead of the processor load.

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 the given max length, or the longest possible reverb time for the given combination of configuration, processor and memory has been found. Then a screen will indicate that the benchmarking is complete, and if it was limited by processor, memory (either the hardware, or statically configured) or a hard limit (minimum block size on the sound card for example).

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

```
+--------------------+
|  Max load [85]%    |
|  Max length [24.0]s|
|  Low latency [off] |
|  Power save [on]   |
| >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, each get their own I/O-delay/reverb time alternatives.

If AlmusVCU is restarted, only the currently selected alternative will be remembered, so when the benchmark menu 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]%    |
|  Max length [24.0]s|
|  Low latency [off] |
| >Run benchmark?    |
|  44kHz 370ms,6.5s  |
+--------------------+
```

3.2.10.2 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.

3.2.10.3 Setting I/O-delay and reverb length manually

Expert users may want to control the I/O-delay and reverb length setting manually. Then it is possible to precisely control the amount of processor time used by trial-and-error. Manual fine-tuning is more precise than the automatic benchmarking. The CPU monitor found in runtime is a valuable tool for finding the longest possible reverb length with a corresponding I/O-delay (note that a reverb program that fills out the full reverb length must be used in such tests).

In the "set manual" menu, the current I/O-delay and, if applicable, reverb time is shown for each sample rate. The sample rate is stated in kHz, rounded to nearest whole kHz in front of each line. In the example below, there is only one sample rate, which is 44.1 kHz.

The I/O-delay and reverb time settings can be altered, and when the bottom alternative "Apply the above" is selected, the values will be set to the nearest above valid values, and those will be used by the convolver. Note that the I/O-delay is in samples in powers of two, starting at 128, so the valid values may seem a bit far apart. The reverb time will be set to an even multiple of half the I/O-delay.

```
+--------------------+
|  44 delay [372]ms  |
|  44 reverb [6.1]s  |
| >Apply the above?  |
+--------------------+
```

AlmusVCU will accept any manual setting just as if it was delivered from the automatic benchmark, so if it is too demanding, the convolver cannot start, but will instead exit with a buffer underflow or a memory error.

3.2.11 User interface setup

The user interface setup menu contains settings which specify the behaviour of some parts of the user interface. The settings mostly affect the runtime main screen (described in section 3.5) which is the screen most time is spent in when AlmusVCU is readily configured.

Some entries are always available in the menu, while other depends on the names and availability of channel groups (see section 3.2.6.10), and if channel modes (3.2.6.11) or equaliser modes (3.2.7.3) are used.

In the example menu shown below, there are three channel groups called “a-pole”, “reverb” and “a-sonic”, for which there are volume selection settings, and both channel modes and equaliser modes are used, so a selection setting exists for both of them.

```
+--------------------+
|  Screen dimmer [1] |
|  Autostart [off]   |
|  Volume step [3.0] |
|  Ch mode [sel on]  |
|  Eq mode [locked]  |
|  Master vol [on]   |
|  A-pole vol [off]  |
|  Reverb vol [on]   |
|  A-sonic vol [off] |
+--------------------+
```

3.2.11.1 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.

3.2.11.2 Autostart

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

3.2.11.3 Volume step

In the runtime main screen (see section 3.5), different sound volume settings can be increased and decreased in one step at a time. The volume step setting defines how large each step is, in dB. There are five presets to choose from, 6.0, 3.0, 1.0, 0.5 and 0.1 dB per step.

3.2.11.4 Mode selection settings

If channel modes or equaliser modes are used (see section 3.2.6.11 and 3.2.7.3 respectively), there will be a selection setting for those. It is one of three values, "locked", "sel on" or "sel off". If it is set to locked, it means that the mode cannot be changed in runtime. If it is set to "sel on" (short for "select on"), it will be possible to select mode directly in the runtime main screen, and if set to "sel off", changing in runtime is only possible from the runtime channel control menu (see section 3.6). The idea to have the setting only availble in the runtime channel control menu is to avoid putting too many settings into the runtime main screen, which will make it cumbersome to operate.

3.2.11.5 Volume selection settings

At the bottom of the user interface setup menu, are all volume selection settings, one entry per channel group (see section 3.2.6.10), plus the master volume. Each setting is either set to "on" or "off". If it is set to on, it will be possible to change the volume the entry represents directly in the runtime main screen (described in section 3.5), or else it will only be possible to change in runtime through the runtime channel control menu (see section 3.6).

The idea of these settings is too allow the user to make a clean runtime main screen, with only the most commonly accessed volume settings available.

If the settings are made such that there is no volume control active at all, the master volume will be made available, so there is always at least one volume control in the runtime main screen.

Should there only be one active channel group (in the current channel mode, if channel modes are used), it will not be accessible in the runtime main screen, regardless the setting in the user interface setup menu. Instead, the master volume will be accessible. This is because if there is only one channel group, it is expected to cover all channels, and thus be equal to the master volume. Should the channel group cover only a subset of channels, it is recommended to change the channel group setup according to the recommendations found in section 3.2.6.10.

3.2.12 System info

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

```
+--------------------+
| Version: 0.85      |
| 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).

3.2.13 Maintenance

In the maintenance menu, the whole AlmusVCU configuration can be saved and loaded, which allows to easily switch between several different configurations.

It also allows for storing debug logs, install software updates and reboot or shutdown the machine.

```
+--------------------+
| >Load config...    |
|  Save config...    |
|  Del config...     |
|  Reset config...   |
|  Flush cache junk? |
|  Debug mode [off]  |
|  Flush logs?       |
|  Save logs...      |
|  Update software...|
|  Restart AlmusVCU? |
|  Reboot machine?   |
|  Shutdown machine? |
+--------------------+
```

3.2.13.1 Load configuration

The whole configuration, that is the static configuration, all specific reverb program settings, and the user configuration can be saved, and loaded.

In the load configuration menu, a device to load from is chosen, and then which of the user, static and reverb configuration components that should be loaded. In the bottom is a list of the available configuration names to load (in the example below there is one configuration to load called “ambio 1”).

```
+--------------------+
| >Load user [on]    |
|  Load reverb [off] |
|  Load static [off] |
|  Load "ambio 1"?   |
+--------------------+
```

The user configuration are all menu settings, except reverb program specific settings. The reverb configuration are all reverb program specific settings, and the static configuration is the manually created static configuration made at installation.

Normally only the user configuration is loaded. Loading an incompatible static configuration will leave AlmusVCU unusable (the original static configuration must then manually be restored). A use of loading the static configuration could for example be to switch between different hardware setups (using different sound cards).

Loading user and reverb configurations are safe, as long as the loaded configuration file format is correct.

When the configuration has been loaded, the program is restarted, employing the new configuration.

3.2.13.2 Save configuration

The current configuration can be saved to be recalled later. All configuration (user, static and reverb) will be saved to an archive (in tar format). In the save configuration menu, first the device to save to is selected, and then there is the choice to name the configuration, or replace an existing if available on the selected device. In the example screen below there is one configuration to replace, called “ambio 1”.

```
+--------------------+
| >New []            |
|  Replace "ambio 1"?|
+--------------------+
```

The file name on disk is picked to be as close as possible to the name picked in the user interface, but truncating it to 8+3 characters (to be compatible with primitive file systems).

The archive contains all configuration files, and an index file called `config.txt`, which contains further configuration information, and the full name.

3.2.13.3 Delete configuration

This menu is used to delete any stored configuration archives, and is straight-forward to use.

3.2.13.4 Reset configuration

If it is desired to start over with new configuration, this menu is used. It allows to reset the user and reverb program configurations. It simply removes the configuration and restarts the software.

3.2.13.5 Cache junk

When a reverb program is removed from the cache, its configuration file is still kept, so it can be re-used when the program is loaded again. However, if one would want to flush all those orphan configuration files, this is done with the "flush cache junk" function.

3.2.13.6 Debug logs

When debug mode is turned on, various logging will be activated, which is helpful in order to find and fix any problem that might occur.

In the "save logs" menu the current logs can be saved to a compressed tar archive on disk, which is useful for sending bug reports to the author. The name of the archive will be `vculogs.tgz` (if it exists, it will be replaced) and contains all AlmusVCU logs and configuration plus some information about the computer.

If debug mode is active, the logs can grow large, and should be periodically flushed. This is done by selecting the "flush logs" alternative. However, logs are normally automatically rotated (that is flushed), as configured in the static configuration file, so it is generally not necessary to do it manually.

3.2.13.7 Update software

When the AlmuVCU software should be updated, it can manually be installed on the computer as described in the installtion chapter. However, for an embedded runnning system, it may be easier to update the AlmusVCU software by loading all binaries from a compressed tar archive. The update software menu provides just that function. In the menu a device is chosen, and there all tar archives, compressed or not, are listed (`*.tar`, `*.tar.gz` and `*.tgz`).

When an archive is selected, it will be scanned through for binary files belonging to AlmusVCU and its convolution engine BruteFIR (that is `almusvcu` `*.vcu`, `brutefir`, `*.bfio` and `*.bflogic`), and those are installed. If any of the AlmusVCU binaries are replaced, the program will restart after installtion is finished. The target installation directories will be the ones used by the currently running AlmusVCU.

3.2.13.8 Restart AlmusVCU

The restart alternative restarts the AlmusVCU software. This should normally not be necessary, but if AlmusVCU seems to behave strangly, perhaps due to a bug, it may be worth just trying to restart the software.

3.2.13.9 Restart/shutdown

The restart and shutdown alternatives makes the computer restart or shutdown the way computers like, that is with synching and unmounting all file systems. For a computer which runs a journaling file system, it is not necessary to use these settings, then the computer can be turned off directly with its power button.

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 for example be a CD-ROM, DVD-ROM or a 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 means that the size in RAM can be much different from the size on disk.

When a program is loaded from a removable media such as a DVD-ROM, it is also copied to a local cache on the hard disk (if there is space left). 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. If a filter program is so large that it does not fit in the disk cache, which could happen for extreme reverb programs as they can be several hundreds of megabytes large, it will be pre-processed and loaded to RAM anyway, however not stored to the cache, so when AlmusVCU is restarted the reverb program must be manually reloaded from the source.

If the filter program resides on a permanently mounted media, such as a local hard disk, a cached copy is not created, instead a reference to the program is stored so it can be reloaded directly from the source when AlmusVCU is restarted.

A filter program can be removed from RAM to leave space for other programs. The possibly 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 Loading 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 "harddisk"), a list of the available files are given, stating the name of each filter program. 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.

3.3.1.1 Reverb program copies

For reverb programs, it is possible to load an extra copy of one which is already loaded. The copy will get an index number added to the end of the name so it can be differed from the original. Each copy gets its own reverb program settings, so they can be tuned separately. This way it is possible to for example make instant A/B testing on different reverb program settings.

3.3.2 Deleting and listing 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 3.2.7.2. It is used for creating equalisers that can be assigned to the any of the physical and logical input and output channels in AlmusVCU.

The filter types that can be designed with the built-in filter designer are lowpass, highpass, bandpass, bandstop, all-pass, FFT magnitude, and random phase filters. The FFT magnitude filter exists in two versions, one static and one dynamic. The dynamic version is a special case 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.

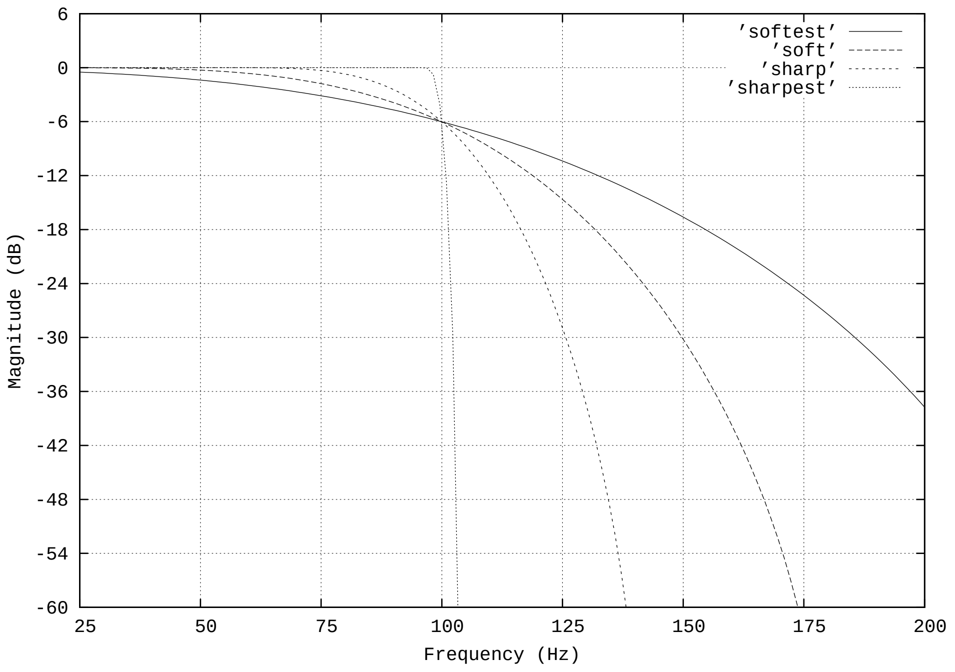

Figure 3.2: 100 Hz lowpass filters with sharpness values 0.00, 0.50, 0.75 and 1.00.

3.4.1.1 Slope

Cutoff frequencies and a rough slope indication is given by the user. The slope is how sharp the attenuation slope should be around the cutoff frequency. The slope will always go through -6 dB at the cutoff frequency. How far before the cutoff frequency the slope starts thus defines how soft the slope will be, and this is specified with a number between 0.00 and 1.00, called the sharpness value.

For lowpass filters, the sharpness value is multiplied with the cutoff frequency to specify where the filter should start to roll off. Thus, a sharpness of zero will make the filter start rolling off at 0 Hz, while a value of 1.00 will make the filter as sharp as possible. A value of 0.50 will make a 100 Hz lowpass filter start rolling of at 50 Hz. For highpass filters (or the higher cutoff of bandpass and bandstop filters), the rolloff will end at cutoff plus the cutoff times the given sharpness value. Hence, the slope will be a perfect mirror of the corresponding slope for a lowpass filter.

Figure 3.2 shows four 100 Hz lowpass filters with sharpness values 0.00, 0.50, 0.75 and 1.00. Usually, sharpness values between 0.50 - 1.00 are the most useful.

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.

The highpass filter is designed by creating a lowpass filter at the given cutoff frequency, and then change polarity of it and add an impulse. This results in a high pass filter that if added together with a lowpass filter at the same cutoff frequency and slope will create a perfect flat response. This is a property useful in for example bass management configurations.

3.4.1.2 Delay and processing requirement

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 (slope value 1.00) is about 740 ms long at 44.1 kHz sample rate, while a 2 kHz soft sloping filter (slope value 0.50) is only about 1.6 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. In most cases it is also motivated from a sonic point of view to use soft sloping filters.

Note that AlmusVCU automatically compensates for the delay in the filters, that is all outputs will be correctly sample aligned.

3.4.1.3 Design menu

The design menu for the filters are invoked from the equaliser design menu, described in section 3.2.7.2. The example below shows the bandpass design menu.

```
+--------------------+
|Bandpass settings:  |
| >Cutoff 1 [2000]Hz |
|  Cutoff 2 [3000]Hz |
|  Sharpness [0.50]  |
|  Create filter?    |
+--------------------+
```

A filter will only be created if the `Create filter` alternative is selected. If the menu is accessed from an equaliser stack, 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, 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 3.2.7.2 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 filter 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.

3.4.3.1 Design menu

The design menu for the random phase filter is invoked from the equaliser design menu, described in section 3.2.7.2.

```
+--------------------+
|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 FFT magnitude equalisers

The FFT equalisers are linear phase equalisers with configurable bands, ISO octave bands existing as presets. Both the static and dynamic version is designed the same way. 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. Design-wise, the static and dynamic versions are exactly the same.

Since the 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 is never shorter than half 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 3.2.7.2 will display what the actual delay is for the filter.

Note that there is no extra processing cost in runtime for the dynamic filter compared to the static version.

3.4.4.1 Design menu

The choice of frequency bands is definite. 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 up to 32 frequencies 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 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 screen

The runtime main screen is the first shown when the convolver is started, and is the screen where most time is spent when AlmusVCU is in runtime. It exists in a main four-line version, and in a two-line version for small displays.

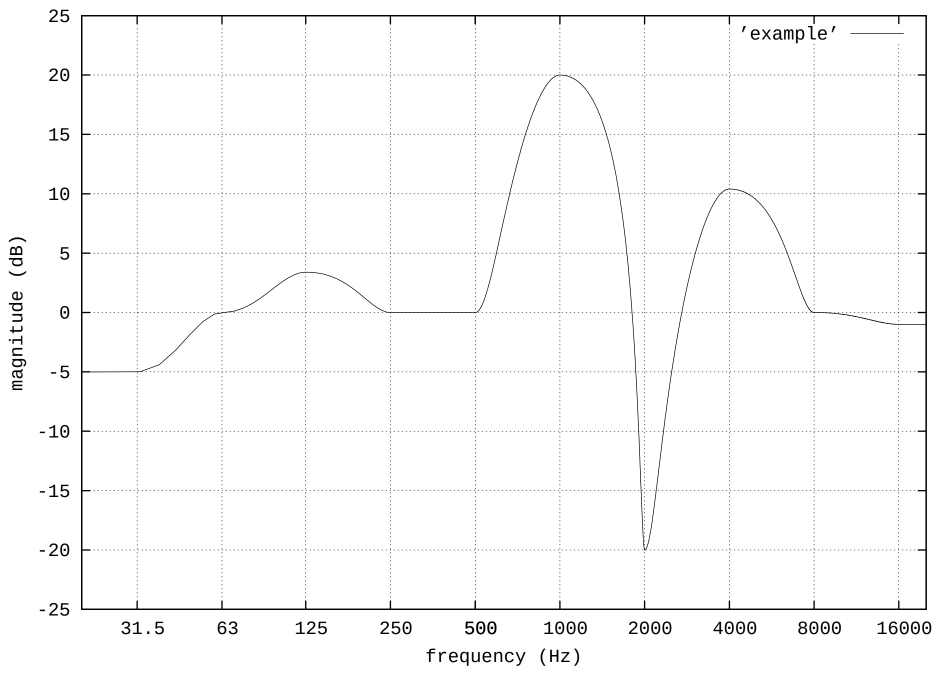

Figure 3.3: An ISO octave filter designed with the dynamic equaliser

The layout is not as a standard menu, and it is not navigated as such. Instead, it is a single screen packed with the most relevant information for runtime operation, and provides access to the most common commands used in runtime. As in the standard menu parts of the user interface, the only keys needed to navigate are UP, DOWN, LEFT and RIGHT.

The exact layout of the screen is decided from how AlmusVCU is configured, thus the examples below are just that, examples of how the runtime main screen may look like:

```
+--------------------+
|    BERWALD HALL    |
| [pick reverb prog] |
|pk-88.3  master-12.0|
|69/44.1  reverb-10.0|
+--------------------+

+--------------------+
|    BERWALD HALL    |
|p-88.3 m-12.0 s-10.0|
+--------------------+
```

3.5.1 Command availability

The commands available in the runtime main screen are dependent on the configuration of AlmusVCU. The possible commands are the following:

- Changing reverb program. This is only available if the reverb engine is used.

- Changing/muting master volume. This is only available if AlmusVCU has been configured to have it available, which is done in the user interface setup menu described in section 3.2.11.
- Changing/muting volume for each channel group. Which channel group volumes that should be accessible in the runtime main screen is configured in the user interface setup menu.
- Changing channel mode. This is only available if channel modes are active, and if it is configured to be selectable in the user interface setup menu.
- Changing equaliser mode. This is only available if equaliser modes are active, and if it is configured to be selectable in the user interface setup menu.
- Changing menu. Exit to the setup menu, or change into the runtime channel control menu.

3.5.2 Using the commands

With RIGHT, the type of command is cycled through. In the four-line version the current command name is printed within brackets on the second line, and is one of the following:

- `[pick reverb prog]`. Change the reverb program. The reverb programs are cycled through with UP and DOWN. Only the reverb programs currently loaded are available for selection. The name of the currently selected reverb program is shown on the top line of the screen.
- `[alter master vol]`. Change the master volume with up and down, or toggle mute by pressing LEFT. For each time pressing UP or DOWN, the volume is increased or decreased respectively with one step. The size of the step is defined in the user interface setup menu. All other volumes (group volumes, trim volumes etc) relate to this master volume.
- `[alter <group name> vol]`. The <group name> is replaced with the name of the group. This is operated the same way as the master volume command.
- `[pick channel mode]`. Change the channel mode. Channel modes are cycled through with UP and DOWN. The name of the currently selected channel mode is shown on the top line of the screen.
- `[toggle eq]` or `[pick eq mode]`. Change equaliser mode. If there is only one equaliser mode, the text string is `[toggle eq]`, and means that equalisation can be toggled with UP and DOWN. The top line of the screen reads "equalisation on" or "equalisation off". If there are more than one equaliser mode, the name of the equaliser mode is printed on the top line of the screen, and the command name is `[pick eq mode]`. The equaliser modes are then cycled through with UP and DOWN, including the special mode "equalisation off" when all equalisation is turned off.
- `[select menu]`. Select menu. The RIGHT button is used for changing to the selected menu. The three menus to select are "channel control", "main screen" (this screen), and "setup", which will exit back to to the setup menu.

In the more compact two-line version, there is unfortunately no place to write the current command selected. Instead, the current command is highlighted through which part of the interface that is in uppercase letters.

3.5.3 Four-line version contents

The top line of the screen contains one of the following:

- The name of the reverb program. This will only be shown if the reverb engine is active, and if the current command selected does not use the top line.
- The name of the channel mode. This is shown if channel modes are used, and if the current selected command does not use the top line.
- The name of the equaliser mode or "equalisation on" or "equalisation off". These strings are shown when the change equaliser mode command is active.
- The name of the menu to select is shown if the select menu command is active.
- The string "AlmusVCU" is shown if none of the above applies, which is only for the most primitive configurations.

The second line contains the name of the currently selected command.

The third line shows the peak volume directly after the abbreviation `pk`. The peak volume is the currently highest volume on any physical output (usually speaker feed) so far, since the main screen was entered, or since it was reset through a change in master or group volume. If the peak is a positive value, it means that the signal has clipped with the given amount of decibels, thus is 0.0 dB the maximum value that can be represented. The value -inf means negative infinity, that is there is only zero output. Note that if the output signal is dithered, the peak volume will never be -inf since there is at least the dither background noise.

At the lower right part of the screen, the master volume and channel group volumes are shown, if applicable. The volume is in dB, and the sign (plus or minus) is always shown. If the volume is muted, the string "mute" is displayed instead of the dB offset. If there are multiple channel groups, the channel group volume shown is the current or last that was selected.

At the lower left of the four-line version are two numbers separated with a slash (/). The first is how many percent of the processor time which is used, and the second is the sample rate in kHz.

3.5.3.1 Examples

Below are four examples of how the four-line version of the runtime main screen can look like in different situations:

```
+--------------------+   +--------------------+
|    BERWALD HALL    |   |      Perambio      |
| [pick reverb prog] |   | [alter master vol] |
|pk-88.3  master-12.0|   |pk+5.3   master:mute|
|69/44.1  reverb-10.0|   |20/48.0 a-sonic+1.0 |
+--------------------+   +--------------------+

+--------------------+   +--------------------+
|     AMBISONICS     |   |      Perambio      |
|[pick channel mode] |   |[alter a-sonic vol] |
|pk-88.3  master-12.0|   |pk-inf   master-10.0|
|34/44.1             |   |20/48.0 a-sonic-6.0 |
+--------------------+   +--------------------+
```

Upper left example

The name of the currently selected reverb program is shown on the top line, which is “Berwald Hall”. The selected command is “pick reverb program”, and thus UP and DOWN will cycle through any other loaded reverb programs. The peak volume so far is -88.3 dB, and the master volume is set to -12.0 dB. The channel group called “reverb” has an offset volume of -10.0 dB. Channel group volumes are relative to the master volume, so the actual volume of the reverb channel group is thus -22.0 dB. The current processor load is 69%, and the sample clock is 44.1 kHz.

Upper right example

In the AlmusVCU configuration shown here, channel modes are used, and the name of the current mode is “Perambio”, which is printed on the top line. The master volume has been muted (using the LEFT key), meaning that there are no sound output currently. Since the current command is “alter master vol”, the master volume can be unmuted by pressing LEFT. The channel group named “a-sonic” has an offset volume of +1.0 dB.

The peak volume is at +5.3 dB which means that the signal has been clipped with as much as 5.3 dB since the peak meter was last reset. The current processor load is 20%, and the sample clock is running at 48.0 kHz.

Lower left example

In this example, the currently active command is “pick channel mode”, where channel mode is selected with UP and DOWN. The current channel mode is called “Ambisonics”, and is capitalised to emphasize that channel mode is currently being selected. For this channel mode there is only one channel group, and thus its volume is not controlled, and therefore there is an empty space in the lower right corner.

Lower right example

In the lower right example, the current command is “alter a-sonic vol”, which means altering the volume for the channel group named “a-sonic”, whose volume currently is -6.0 dB. The peak level is currently at -inf, which means that there is no output whatsoever. This means that dither is not activated (which deliberately produces some low-level output noise), and there is no signal on the input.

3.5.4 Two-line version contents

The two line version does not have the name of the currently selected command on it. Instead, the selected command is indicated through which part of the screen that is in uppercase letters. Another limitation compared to the four-line version is that the channel group names are not printed, instead the name is represented with the single character ’g’. This means that it is not very suitable to have multiple channel group volumes available in the runtime main screen if using the two-line version, because all groups will be represented by the character ’g’.

The clock rate and CPU utilisation numbers are not available in the two-line version, but can instead be viewed in the runtime info screen found in the runtime channel control menu, described in section 3.6.

Below are four examples of how the two-line version of the runtime main screen can look like in different situations:

```
+--------------------+   +--------------------+
|    BERWALD HALL    |   |      Perambio      |
|p-88.3 m-12.0 g-10.0|   |p+5.3  M:mute g+1.0 |
+--------------------+   +--------------------+

+--------------------+   +--------------------+
|     AMBISONICS     |   |      Perambio      |
|p-88.3        m-12.0|   |p-inf  m-10.0 G-6.0 |
+--------------------+   +--------------------+
```

The examples correspond exactly to the four-line version examples described earlier in section 3.5.3.1.

Obviously, it is not recommended to have equaliser modes, channel modes or reverb programs named the same, since then it will not be possible to sort out what the currently active command is. In the above examples, the active command on the upper left is "pick reverb program", since "Berwald Hall" is a reverb program, and it is shown in capital letters. In the lower left example, the active command is "pick channel mode", since the name "Ambisonics" is a name of a channel mode, and it is capitalised.

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 channel modes have been configured, there will be no channel mode menu entry. The trim volume and trim delay menus are the same as found in the channel setup menu, and is documented in section 3.2.6. There is a small difference though, only the channels used are listed in the menus, and therefore the amount of channels are often less than in the versions found in the channel setup menu.

```
+--------------------+
| >Master [-12.0]dB  |
|  A-pole [+0.0]dB   |
|  Reverb [-24.0]dB  |
|  Master mute [off] |
|  A-pole mute [off] |
|  Reverb mute [off] |
|  Channel mode...   |
|  Eq mode...        |
|  A-pole setup...   |
|  Reverb setup...   |
|  Trim volume in... |
|  Trim delay in...  |
|  Trim volume out...|
|  Trim delay out... |
|  Runtime info...   |
+--------------------+
```

A the top of the menu, there are volume and mute settings. The master entries are always shown, the others are for the channel groups. In the example above, there are two channel groups named “a-pole” and “reverb”. If channel modes are used, on the currently active channel groups are listed.

If there is only one channel group, its volume settings is put away into a sub menu called `Offset volume`. This is made because the channel group volume will in those cases control all used channels (if the channel group is configured according to the recommendations), and thus becomes a setting similar to the master volume.

3.6.1 Channel mode

The channel mode menu only exists if the configuration uses channel modes, as described in section 3.2.6.11. The menu displays the name of the currently active channel mode, and allows cycling through them.

```
+--------------------+
|Ambiophonics        |
| >Next channel mode?|
+--------------------+
```

In the example above, the currently selected channel mode is called “Ambiophonics”.

3.6.2 Equaliser mode

The equaliser mode menu (`Eq mode`) is only shown if AlmusVCU is configured to use at least one equaliser mode. If channel modes are used, the menu will only be available if the current channel mode uses equaliser modes.

The menu displays the name of the current equaliser mode (or “equalisation on” if there is only one mode), or “equalisation off” if equalisation is turned off.

```
+--------------------+
|Default             |
| >Next eq mode?     |
|  Dynamic eq...     |
+--------------------+
```

In the current equalisation mode contains dynamic equalisers, the sub menu `Dynamic eq` will be available.

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. 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 be added to the redesign delay.

The redesign is done in the background with the available 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.

Designing the equaliser is done the same as in the equaliser design menu, and is described in section 3.4.4.

3.6.3 Ambiopole setup

The ambiopole setup menu is only available if there is one or more ambiopoles configured. If there is more than one, 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.

3.6.4 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 however 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 |
|  Xbalance [+0.0]dB |
|  Ybalance [+0.0]dB |
|  Zbalance [+0.0]dB |
+--------------------+
```

The remaining entries, delay, trim volume and balance settings are the same as found in the tune reverb program menu in setup, which can be read about in section 3.2.4.

3.6.5 Runtime info

The runtime info screen exists in one version for four-line displays, and a more compact version for two-line displays. It contains five values.

- CPU load. The number of percent of the time the processor is busy. This counter is global for the machine, so if there are other programs running (not recommended), they will impact the CPU load. If the machine has multiple processors, the percent used of the most busy processor will be shown.
- Memory use in percent. A global view on how much memory is left on the machine to be used by programs. This will include any swap space. Note that it is recommended to run AlmusVCU on a machine without swap memory (RAM paged to hard disk), since swapping can give unpredictable performance. With the reverb engine active and many channels, AlmusVCU will typically contribute to the major part of the memory used.
- CPU Temperature. The temperature in degrees Celcius of the CPU. This only works if the machine has been configured to use the temperature sensors, and AlmusVCU has been configured to use them. If they are not active, this value will be zero.
- Motherboard temperature. As the CPU temperature, but the motherboard sensor is used here.
- Sample rate. The current sample rate.

```
+--------------------+
|Load CPU/Mem 69%/76%|
|Temp CPU/MB 66C/60C |
|Sample rate 44.1kHz |
|                    |
+--------------------+

+--------------------+
|CPU 69%/66C  MB 60C |
|Rate 44.1kHz Mem 76%|
+--------------------+
```

In the above example screens, the CPU load is 69%, memory usage 76%, CPU and motherboard temperature 66 and 60 degrees Celcius (both rather high).

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 10.3.

Chapter 4

The reverb engine

4.1 Introduction

The AlmusVCU real-time reverb engine can be used in Ambiophonics or other applications to drive a reverb loudspeaker array, or be used as a reverb processor, that is to reverberate a dry recording in recording post-production.

It uses a one to three channels source signal as input, and produces a configurable amount of reverberated outputs, based on a loaded reverb program. Several reverb programs can be loaded at the same time allowing switching between them in realtime.

4.2 How it works

4.2.1 Reverb programs

A reverb program is a data file archive containing the recorded acoustics of a venue, typically a concert hall. From a reverb program, filters for each channel to be reverberated are derived.

The acoustics is recorded into a set of impulse responses in or around a listening position. The sound source is placed on the stage (if there is one). Thus the captured acoustics will match the reverberation heard by a listener sitting in the recording position, while listening to a sound source which is in the same position as in the recording. A reverb program can store impulse responses for a source placed on the center of the stage, or specific sets for the left and right parts of the stage, or sets for all three positions left, right and center. The reverb engine can be configured to work with one to three inputs, that is center, left/right or left/right/center (lrc). For best results, the reverb programs used should have at least as many source positions as the number of inputs used by the reverb engine.

AlmusVCU supports two types of reverb programs:

- B-format. This type of reverb program contains the impulse respons(es) in B-Format, and thus a full 3D representation (with the limits the B-format implies).

- Traditional. This type of reverb program contains various amounts of mono impulse responses. Typically there are four cardiod impulse responses per source location, measured around a listening center with a few meters spacing between the microphones, which typically are aimed +/- 45 and +/- 135 degrees in the plane. However, there impulse response may be recorded with omni-directional microphones, there may be more or less than for, only one omni-directional impulse response is possible for example. Some reverb programs has several decorrelated sets of impulse responses for the same directions.

Of the two reverb program types, the B-format is the more “correct” one, since it stores a full 3D representation. For few amount of reverb channels, or fixed position which fairly well matches the given reverb program, the traditional type can however produce better results. The sound quality of the acoustic recording itself is generally more important than the reverb program type for a good end result.

The traditional reverb program type is commonly used in commercial reverb processors aimed at recording post-production, and in that case there is one impulse response dedicated for each channel to be convolved.

Note that the B-format reverb program also may have several impulses per source location, for example along a circle around the listening position. These solutions provide for better decorrelation, and makes it possible for wave-front synthesis renderings (not supported by AlmusVCU), but usually makes the reverb programs very large, potentially several hundreds of megabytes.

4.2.2 Deriving convolution filters

Before entering runtime mode, AlmusVCU will derive convolution filters from the reverb program to match the given channel configuration. Each reverb channel has an assigned direction, through the speaker layout configuration. Thus, the task for AlmusVCU is to find or derive a suitable impulse response for those directions. This is how it is done:

1. Pick the best matching impulse response. The reverb program contains one or more impulse responses recorded in specified directions out from the listening center. Ideally, these directions should match reverb speakers directions, which they often do not, and then the following rules are used to get a best match. This is done for both B-format and traditional reverb programs, but has most relevance for the latter, since the former is full 3D and thus often has only one response per source position.

    - Reverb speakers are first sorted after closest to vector 0/0, meaning that frontal speakers will get priority in finding the best suitable impulse response. This is because any error would be most obvious in that direction.
    - Least difference in angle. A small difference between a given reverb program impulse response angle and the reverb channel angle is preferred.
    - If the reverb program only contain impulse responses with angles to one side (must be left), the directions are mirrored when comparing.
    - For side and back reverb channels, where azimuth is equal to or larger than +/- 80 degrees, a back impulse response is rather assigned than a front one, if available. This is to avoid getting a “back-heavy” reverb, that is getting too much reverb energy from the back.

- If not very large angle differences, it is preferred to use all impulse responses instead of always picking the closest one.
- If the impulse response has been used already, it is picked from another set within the reverb program, if available.

2. Compensate for source layout mismatch if necessary. The reverb engine can be configured for one to three sources (center, left+right or left+right+center), and the reverb program can provide one to three sources. If there is a mismatch, it must be compensated. This is done for both B-format and traditional reverb programs.

    - One reverb engine source, three reverb program sources – only the center source is used.
    - One reverb engine source, two reverb program sources – left and right sources are mixed into a new fake center.
    - Two reverb engine sources, three reverb program sources – only the left and right sources are used.
    - Two reverb engine sources, one reverb program source – reverb program source used as fake left, and a decorrelated version as fake right, and an artificial left/right delay difference is applied to provide some directionality. For elevated speakers, the left/right delay difference is reduced.
    - Three reverb engine sources, one reverb program source – reverb program source used as fake left, and decorrelated versions as fake right and center, and an artificial left/right delay difference is applied to provide some directionality. For elevated speakers, the left/right delay difference is reduced.
    - Three reverb engine sources, two reverb program sources – left and right in the reverb program are used for left and right in the reverb engine, and a fake center is derived by mixing left and right and decorrelating the result.

3. Apply reverb engine configuration parameters, as described in section 3.2.4. Note that some reverb configuration parameters are dynamic and can be changed in runtime. Only the static ones are applied here.

    - Depending on the early response setting, mute an early part of the impulse and fade it in. Any previously introduced artificial left/right delay will be kept if the "force left/right delay difference" setting is active, in order to keep the sense of directionality.
    - The decorrelation policy configuration is followed, that is "none", "needed" or "all". For "needed", if an impulse response is re-used, it will be decorrelated using a time-variant decorrelation algorithm (greater degree of decorrelation furher back in the response). A B-format impulse response can be re-used without decorrelation though, as long as another direction is extracted from it.
    - If configured to do so, mix in the dry signal.

4. If necessary, reduce early response reverb energy for directional mismatch compensation. This is only necessary for traditional reverb programs where there is a large mismatch between the reverb program impulse response direction and the speaker direction.

- If there is more than a 90 degree azimuth difference between reverb program impulse and speaker direction, and the impulse is in the front half (more reverb energy), the early part is compensated through direction-dependent attenuation.
- If there is a large elevation difference between speaker and its assigned impulse response, it is compensated for, by possibly applying early attenuation.

Note that due to the inherent fuzziness of reverb, there can be quite large mismatches in direction without them being noticed.

4.2.3 User settings

The user can tune each reverb program, by setting another decay time, muting the direct sound or early response, and specifying decorrelation mode among other things. In section 3.2.4 all these settings are described in detail.

4.3 Use in Ambiophonics and similar applications

The Ambiophonics tutorial in chapter 5, contains a step by step setup of the reverb engine for Ambiophonics, which could be applied in other applications as well, where a reverb speaker array is needed to supplement a direct-sound stage.

4.4 Use as a dedicated reverb processor

AlmusVCU and its reverb engine is not primarily designed to be used in recording post-production, and therefore does not have a user interface strictly adapted for that task. However, the patient recording hobbyist can tame AlmusVCU to become a very competent unit in this area as well, and of course at an unbeatable price. The sonic performance is decided by the reverb programs used, and can thus be as good as or better than commercial units. As long as AlmusVCU is run on a reasonable up-to-date computer, the performance in terms of throughput is generally much better than the best commercial hardware reverb processors (which on the other hand almost always have very low or no I/O delay, something AlmusVCU cannot match).

When AlmusVCU is used as a dedicated reverb processor, the tools used are the reverb engine and the passthru functionality. To use familiar terms in reverb post production, the passthru is used to deliver the “dry” sound, and the the reverb is the “wet” sound.

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.

Note that the following short descriptions are not meant to be a complete manual on how to use reverb in recording post-production, but rather a quick insight on how to use AlmusVCU for that purpose.

4.4.1 Stereo reverberation

To reverberate a stereo recording, AlmusVCU is configured to have two reverberation channels, and two passthru channels. There is only need for two input and two output channels, so the AlmusVCU computer could be equipped only with a simple stereo full duplex sound card.

The following description of how to configure AlmusVCU is compact and only shows the most important parts. For a complete tutorial, see the Ambiophonics tutorial in chapter 5, and then modify its settings according to the instructions given here.

4.4.1.1 Channel configuration

The "Channel setup" menu should look like this:

```
+--------------------+
| >[0] (A)mbiopole   |
|  [2] (R)everb      |
|  [0] am(B)isonics  |
|  [2] (P)assthru    |
|  Reverb setup...   |
|  Ambisonic setup...|
|  Input matrix...   |
|  Output matrix...  |
|  Speaker layout... |
|  Stage layout...   |
|  Channel groups... |
|  Channel modes...  |
|  Trim volume in... |
|  Trim delay in...  |
|  Trim volume out...|
|  Trim delay out... |
+--------------------+
```

The "Reverb setup" menu like this:

```
+--------------------+
|  Layout [stereo]   |
+--------------------+
```

The input and output matrix menus should look like this:

```
+--------------------+
| >iRl (i1)...       |
|  iRr (i2)...       |
|  iP1 (i1)...       |
|  iP2 (i2)...       |
+--------------------+
```

```
+--------------------+
| >oRl (o1)...       |
|  oRr (o2)...       |
|  oP1 (o1)...       |
|  oP2 (o2)...       |
+--------------------+
```

The speaker layout should be configured with o1 as 30 degree azimuth, and o2 as -30 degree azimuth, that is the stereo triangle configuration. The speaker distance should be set to zero. The speaker layout will be used by the reverb engine, to derive proper filters from the given reverb program.

A channel group for the reverb should be created so its volume can be controlled separately.

4.4.1.2 Reverb program configuration

The reverb defaults menu specifies the default reverb program settings which is applied when a reverb program is loaded for the first time.

```
+--------------------+
| >[Mute direct snd] |
|  [Decorr as needed]|
|  Force LRdelay[off]|
|  Force decay [on]  |
|  Mix with dry [off]|
|  Dry vol [+0.0]dB  |
|  B-format dir [1.4]|
+--------------------+
```

The direct sound will be delivered by the passthru channels, so it should be muted, and the mixing with dry is not necessary, since that is done already with the passthru channels. If the recording to be reverberated is not totally dry, it may be wise to mute the early response and not only the direct sound.

Reverb programs with stereo source should be preferred. For tradititional reverb programs, there should be at least two receivers, one pointing approximately to the left speaker and the other approximately to the right. Reverb programs based on the B-format will generally always work well (if they are of good quality, and has at least a stereo source layout that is), since from them it is possible to generate proper impulse responses for all directions.

The reverb programs can of course be individually tuned, for example to shorten reverb time, in the “Tune reverb programs” menu.

4.4.1.3 Runtime tuning

The reverb volume controls the “wet” volume, that is the reverb volume, while the master volume is the “dry” volume, that is the volume of the passthru channels. For normal use, these are the only settings that need to be accessed in runtime, apart from possibly changing reverb program that is.

If desirable, it is possible to assign separate equalisers to the dry and wet channels, and if these are dynamic, they can be controlled in runtime.

4.4.2 5.1 reverberation

AlmusVCU supports LRC reverberation, and thus can be used for 5.1 reverb. This can for example be used to expand a stereo recording to a five channel recording with surround channels, or enhance reverb on existing five channel recordings.

4.4.2.1 Stereo to five channel

It is easy to add the reverberated surrounds. Use stereo layout for the reverb engine, make a five channel layout corresponding to the ITU 5.1 standard, and do reverb.

The hard part is to derive the center channel. Many techniques exists for that, and works well or not depending on the source material. While the goal is to enhance the stereo image stability, the derived center channel can in many cases intsead lead to fuzzier phantom imaging. The easy way out, which has been used in many upmixes, is simply to put a L+R mix 20 dB down in the center channel. The level of the center channel is then so low that it is psychoacoustically the same as having it muted, but some sound is put there so the listener does not think that there is something wrong with the center channel.

4.4.2.2 True five channel recordings

The principle for adding reverberation to a true five channel recording is exactly the same as for stereo to stereo reverb described in a previous section. The difference is that a LRC reverb engine layout is used, and for the surrounds, mixing between recorded and derived reverb is employed (since the recorded surround channels are not used as input for the reverb engine).

4.4.2.3 The LFE channel

For music recordings, many think that the LFE channel should not be used, but should instead be left to the user to do possible bass management on the playback side. In any case, the five main channels should be kept full range.

4.5 Making your own reverb programs

Currently, AlmusVCU does not support sampling reverb programs through it. This might be supported in future releases. Currently, these have to be made manually, using other tools.

In section 10.1 the reverb program file format is described.

4.5.1 How a reverb program should sound

The quality of a reverb program is easiest verified in a proper reverb array, such as one in an Ambiophonics system.

If the input to the reverb engine is to one side only (only left or only right), the diffuse reverberant sound could be heard as coming more from the corresponding side. This can be beneficial, but is not a property heard in all good reverb programs, and is in any case reduced if the early response is muted.

If the reverb program is used without muting its direct sound, and the reverb array has frontal speakers, the sound could have some direction towards the front, but not necessarily. It should however not be heard as if coming from the back. If the early response is muted, the reverb program may start to sound a bit like coming from the back, and should sound more diffuse than with direct sound used.

With the early response muted, 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, if the reverb array 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.

If the sound seems to be very close to the head, there is probably some phase problem with the reverb program.

Chapter 5

Ambiophonics tutorial

5.1 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 and elaborated information can be found at:

`www.ambiophonics.org`

5.2 System design tips

5.2.1 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 time on a music recording is most often much longer than for an ordinary living room, and then the reverberation of the living room can be masked. For successful masking, the recorded reverberation time 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 reverberation 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 at the recording studio market, although they usually are the ones that look the best (but not necessarily those with the best absorption).

The bass frequency range is always very difficult to absorb. So called “bass traps” help some, but do 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, 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 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 reasonable silent. There is quite much noise on recordings as well though, since microphones, recording equipment and recording venue are not perfect, and that noise can mask minor background noise in the room. However, in silent passages of music playback, an audible ventilation system or other noise source will make the listener aware of the listening room acoustics, and thus 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. Putting the electronics outside of the listening room also helps to solve or reduce thermal problems, which are common in small rooms with much absorption.

5.2.2 The ambiopole

The ambiopole consists simply 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. Here are a few tips to consider:

- 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 / mp3 on the web].

5.2.3 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 about the same volume.

The secret is that the sole purpose of the reverb array is to reproduce a rather 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 mostly 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. It is good to experiment with different speaker placements before settling with a layout.

- The amplification should produce as little noise as possible. 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: how important is good dynamic range? Can one let the ambiopole dominate at high sound levels?]

Of course, in a high-end system, the reverb speakers should not be of low quality, but if they are as good as the ambiopole speakers, then the system is not well balanced. A rough rule is to spend as much on the whole reverb array as on the ambiopole.

5.3 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 5.1 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.

5.4 Installing AlmusVCU

In this tutorial it is assumed that the sound card hardware in the AlmusVCU machine 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.

5.5 Configuring AlmusVCU

5.5.1 Static configuration

The static configuration is a text file created manually once at the installation, and specifies things such as which input device(s) to use (a mouse, a keyboard etc), which sound card(s) and so on. Chapter 2 describes the static configuration and how it is set up.

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:

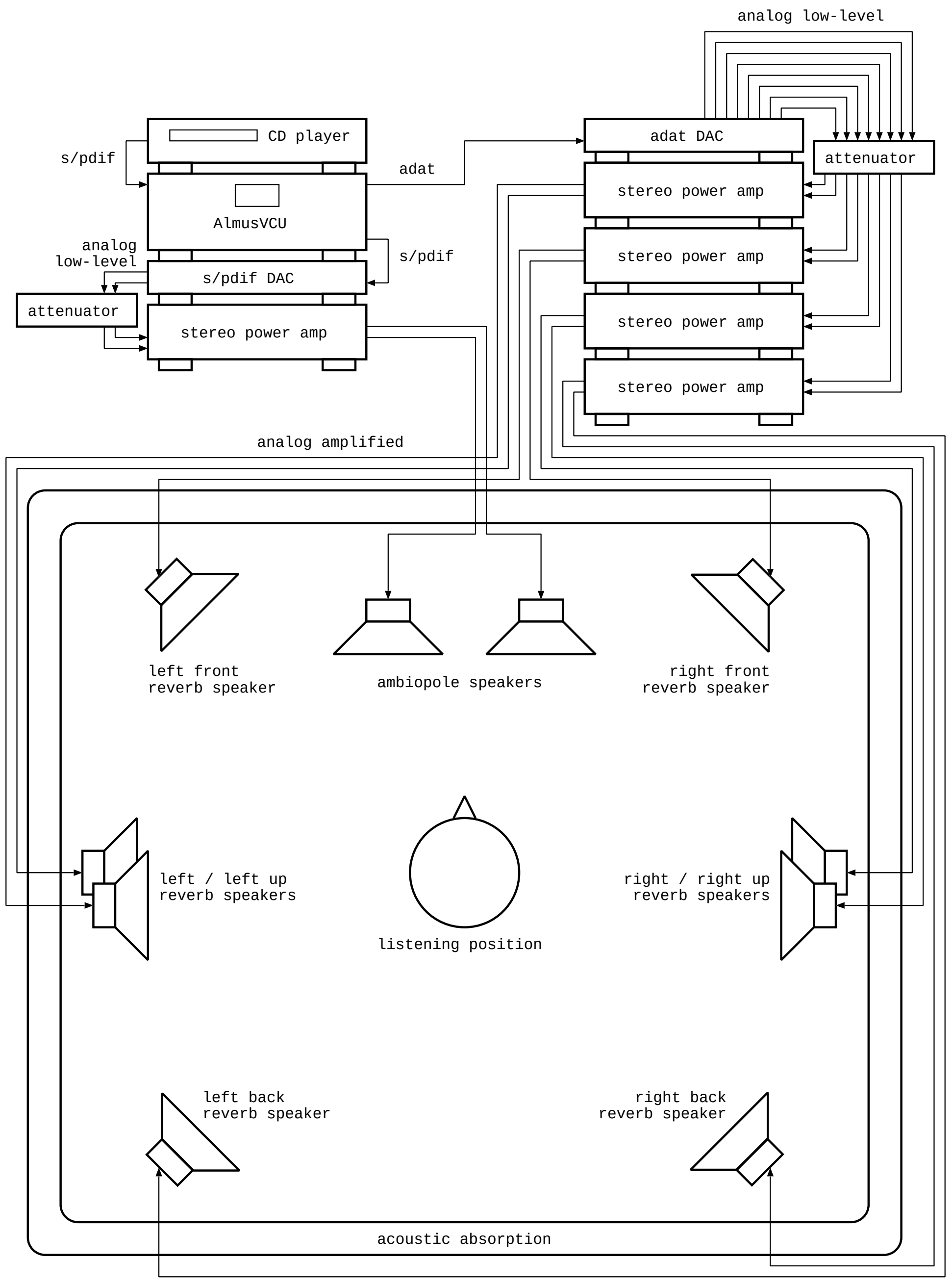

Figure 5.1: a typical Ambiophonics system, consisting of an ambiopole and eight reverb channels.

```
+--------------------+
| >Start convolver?  |
|  Master [-48.0]dB  |
|  Master mute [off] |
|  Load reverb prg...|
|  Del reverb prg... |
|  Tune reverb prg...|
|  Signal setup...   |
|  Channel setup...  |
|  Eq setup...       |
|  Ambiopole setup...|
|  Reverb defaults...|
|  Convolver setup...|
|  User intf setup...|
|  System info...    |
|  Maintenance...    |
+--------------------+
```

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

5.5.2.1 Signal setup

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

```
+--------------------+
| >Input [sound card]|
|  PCM file setup... |
|  [24] bit output   |
|  [32] bit internal |
|  Dither off        |
|  Watch clock [off] |
|  Allow 44.1kHz[on] |
|  Allow 48.0kHz[off]|
+--------------------+
```

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 user manual does.

Also note that some settings depend on the hardware configuration. In the example above, the hardware supports both 44.1 and 48.0 kHz, and that is thus listed in the menu. For further details on the signal setup menu, read section 3.2.5.

5.5.2.2 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    |
|  Reverb setup...   |
|  Ambisonic setup...|
|  Input matrix...   |
|  Output matrix...  |
|  Speaker layout... |
|  Stage layout...   |
|  Channel groups... |
|  Channel modes...  |
|  Trim volume in... |
|  Trim delay in...  |
|  Trim volume out...|
|  Trim delay out... |
+--------------------+
```

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.

Reverb setup

The reverb engine should be left at its default, that is stereo source layout:

```
+--------------------+
| >Layout [stereo]   |
+--------------------+
```

I/O matrices

The input matrix should be configured to point out the S/PDIF input. Assuming that those are i9 and i10, the menu should look like this:

```
+--------------------+
| >iA1 (i9)...       |
|  iA2 (i10)...      |
|  iRl (i9)...       |
|  iRr (i10)...      |
+--------------------+
```

Both inputs for the ambiopole and the reverb engine should be linked to the S/PDIF input, meaning that the inputs should be connected directly to the CD player.

In the output matrix menu, the outputs for the ambiopole should be set to the S/PDIF output, which in this example is assumed to be o9 and o10. The reverb channels should be put to their own ADAT outputs, assumed to be o1 - o8 in this example. The menu should then look like this when left:

```
+--------------------+
| >oA1 (o9)...       |
|  oA2 (o10)...      |
|  oR1 (o1)...       |
|  oR2 (o2)...       |
|  oR3 (o3)...       |
|  oR4 (o4)...       |
|  oR5 (o5)...       |
|  oR6 (o6)...       |
|  oR7 (o7)...       |
|  oR8 (o8)...       |
+--------------------+
```

Speaker layout

The speaker layout (configured in the `Speaker layout` menu) 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.

The position of the ambiopole should be used as the reference for the front direction, that is the point between the ambiopole speakers should be set to azimuth 0.

Stage layout

In the stage layout menu, a 150 degree wide stage, 40 degrees high should be specified for the ambiopole. The stage layout menu should look like this when left:

```
+--------------------+
| >S1 azim [0]deg    |
|  S1 elev [0]deg    |
|  S1 width [150]deg |
|  S1 height [40]deg |
|  Add stage?        |
|  Delete last stage?|
+--------------------+
```

The stage layout is used by the reverb engine to make rendering decision based on from where direct sounds are generated.

Channel groups

Channel groups define group of channels which get their own name and volume control. For Ambiophonics, there are two clear compononents, the reverb array and the ambiopole. Thus, two channel groups should be created, one named “a-pole” which controls the ambiopole outputs, and one named “reverb” which controls the reverb outputs. The volume of both should be set to +0.0 dB.

```
+--------------------+
|  Name [a-pole]     |
|  Volume [+0.0]dB   |
|  Mute [off]        |
|  Inputs (0)...     |
|  Outputs (2)...    |
| >Create this group?|
+--------------------+
```

Channel modes

Channel modes are only used when creating a configuration supporting several decoding schemes, such as Ambiophonics and Ambisonics in the same configuration. This setup will only support Ambiophonics, so the channel modes menu is skipped.

Trim volumes and delays

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.

5.5.2.3 Ambiopole setup

Next leave the channel setup menu, skip equalisation setup (the `Eq setup` menu) and enter the ambiopole setup menu. Load a crosstalk cancellation program and activate crosstalk cancellation for the ambiopole. If an ordinary stereo triangle is to be used instead of the ambiopole, the cross-talk cancellation should be kept de-activated.

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

5.5.2.4 Reverb defaults

The reverb defaults menu specifies the default reverb program settings which is applied when a reverb program is loaded for the first time.

```
+--------------------+
| >[Stage adapted]   |
|  [Decorr as needed]|
|  Force LRdelay[off]|
|  Force decay [on]  |
|  Mix with dry [off]|
|  Dry vol [+0.0]dB  |
|  B-format dir [1.4]|
+--------------------+
```

For Ambiophonics, the reverb array should not reproduce any direct sound or early reflections if any speakers are put inside the ambiopole reproduction range, since those are reproduced by the ambiopole. Thus, the early response setting is set to “Stage adapted”.

The reverb engine should decorrelate impulse responses when needed, and there should be no dry signal mixing. The B-format directivity factor is suitably set to 1.4, which is the default.

It is a matter of taste if the force left/right delay setting should be activated, it is off per default. It is described in detail in section 3.2.4.

5.5.2.5 Convolver setup

Next, the convolver setup menu is entered, and there the benchmark menu is chosen, since setting the convolver settings manually is a bit cumbersome.

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, 6.0 seconds or more is considered good. It could look like this before the menu is left:

```
+--------------------+
|  Max load [90]%    |
|  Max length [12.0]s|
|  Low latency [off] |
| >44.1 [186ms,4.6s] |
|  Rerun benchmark?  |
+--------------------+
```

5.5.2.6 User interface setup

For Ambiophonics, it is typical to adjust the reverb array volume relative to the ambiopole. Therefore, it is good if the reverb offset volume is easily accessed from the runtime main screen. However, it is not necessary to adjust the ambiopole level, if that is not at the correct volume, the reverb level should be adjusted instead.

Thus, in the user interface setup menu, the reverb volume and master volume controls are activated, and the ambiopole volume control is de-activated

```
+--------------------+
|  Screen dimmer [1] |
|  Autostart [off]   |
|  Volume step [3.0] |
|  Master vol [on]   |
|  A-pole vol [off]  |
|  Reverb vol [on]   |
+--------------------+
```

5.5.2.7 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.

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

5.6 Setting up the reverb array

5.6.1 Speaker layout

The reverb speakers should be placed far away from the listening position, usually near the listening room walls, if not the listening room is very large. This way a large sweet area with realistic reverberation will be obtained. In small spaces where long distances to the reverb speakers is 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 amount of 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 often important in order to avoid a reverberation that sounds directional (that is the sounds comes from a specific direction). In general, the farther away from the listening position the speakers are placed, and the more there are, the less need to be cared about symmetry.

5.6.1.1 Speakers within the ambiopole stage

The suggested front placement of +/- 45 degrees is not entierly uncontroversial. The ambiopole stage width can be up to 150 degrees wide (sometimes even more), and thus +/- 45 degrees is often within the stage the ambiopole produces. This means that those reverb speakers would conflict with the sound produced from the ambiopole speakers. However, if the reverb is set to "Stage adapted" and the ambiopole has a defined stage, as suggested in this tutorial, the front speakers will not reproduce any direct sound or early reflections, leaving only the diffuse reverb tail. Thus, the directional information is removed, and will not conflict with the ambiopole speakers.

Still, there are reverb tails reproduced both from the ambiopole speakers and the reverb. This will make the stage slightly more wet than the original. However, in cases where the reverb program acoustics does not exactly match the recorded, the front speakers within the stage will seamlessy blend the stage and reverb array together into a unity.

There are different opinions about the suitability of this approach, so feel free to experiment.

5.6.1.2 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.]

5.6.2 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 (+/- 10 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.

Note that for sounds at the extreme sides, for example left or right only, there may be some directional sense in the sound field (good quality reverb usually have some directionality), so for testing, the best is to use a mono recording, which has the same signal both in left and right channel.

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 usually disappears.

5.6.3 Simple room compensation

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 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`.

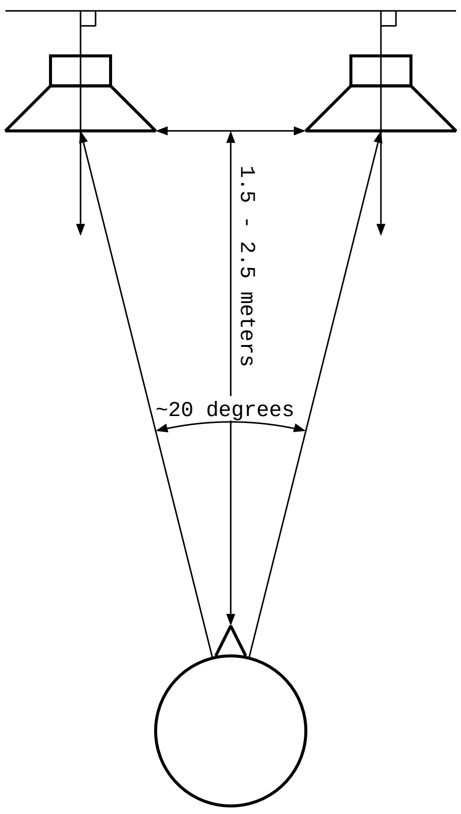

Figure 5.2: ambiopole layout

2. Design an equaliser (`Eq design`), containing a high-pass filter with a soft rolloff (sharpness 0.50) which is designed using the built-in equaliser designer.

3. Create a new equaliser mode (`Eq modes`), and put the equaliser on the physical outputs connected to the reverb speakers. Name the mode "default".

4. Turn on equalisation (done in the `Eq modes` menu).

5.7 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 task 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 can lead to a clear audible degradation, in the form of reduced stage width. Therefore it is very important to be patient when setting it up.

5.7.1 Basic positioning rules

Figure 5.2 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, one behind the other.

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 one degree, can 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.

5.7.2 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 most efficient 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.]

5.7.3 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.

5.7.4 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 or so 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 is 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, 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.

5.7.5 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 one side is playing (input in only one channel to the ambiopole), sit in the listening position and move the head slowly sideways up to one decimeter or so in both directions. If the sounds moves farther to the side when moving the head the opposite direction, the speakers are too narrowly spaced. If it moves farther to the side when moving the head the same direction, the speakers are too widely spaced.

The same test should be performed for both left and 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 quite soon the optimal spacing will be found.

5.7.6 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, or simply because the listener is not sitting exactly on the line between the 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.

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.

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.

5.7.7 Equalisation

Music recordings often have too much high frequency content to sound realistic. One reason for this could be that microphones are usually placed much closer to the musical instruments during recording, than a listener sits in a concert hall. High frequencies are absorbed when transported through the air more quickly than lower frequencies, so the listener in the concert hall will get less high frequencies than what is registered by the microphones.

Another issue is that loudspeakers are much closer to the listener in the Ambiophonics system, than the musical instruments are in a concert hall, and thus there is further exaggeration of high frequencies.

Many audiophiles has learnt to like this exaggeration, the high frequency content gives “air” and “detail” to the recording and improves envelopment. However, an Ambiophonics system has a reverb array, so the air and envelopment is provided the right way.

If one wants to equalise the ambiopole or not, will be a matter of personal taste, but also how evident it is with the given equipment, some speakers roll off more than enough in the high frequency range.

Anyway, the AlmusVCU equalisation capabilities can be used to achieve this:

1. Enter the setup menu, and in that select `Eq setup`.
2. Design an equaliser (`Eq design`), containing a dynamic equaliser, for example a ISO 10-band octave equaliser, with -0.5 decibel at 125 Hz with additional 0.5 dB attenuation for each octave, ending at -4.0 dB at 16 kHz.
3. Create a new equaliser mode (`Eq modes`), and put the equaliser on the physical outputs connected to the ambiopole speakers. Name the new mode “default”. If there already exists an equaliser mode, complement that with the new equaliser
4. Turn on equalisation (done in the `Eq modes` menu).

The equalisation curve suggested above,is only an example, and may be a bit extreme for many speakers.

5.8 Listening

For normal stereo listening, one have to adjust the volume for each recording. Ambiophonics adds the extra task of choosing a matching reverb program and adjust its volume. At first, it may feel a bit cumbersome.

A good habit is to take a small piece of paper and write down master volume, reverb volume and reverb program and store it along with the recording, so next time the recording is listened to, one can apply the settings directly, before starting to listen.

5.8.1 Setting the master volume

When a recording is listened at for the first time, mute the reverb array (can be done in the channel control menu in runtime), and just listen to it through the ambiopole.

The task is here to set a suitable master volume. For recordings of music which normally is amplified when performed live, the level could be any, as long as it is loud enough. However, for recordings of acoustic music, it is beneficial to be more careful when setting the master volume, if one is interested in getting a realistic sound.

Listen to the recording, get a feel of the distance, are the musicians close, or far away? The farther away, the lower the volume. Generally, users tend to set too high volume on small ensemble recordings, and too low volume on orchestral music. When a symphony orchestra plays at full volume in real life, the sound pressure level is often louder than 100 dBA. However, recordings are generally a bit compressed dynamically, either intentionally by using compressors, or by limitations in the recording equipment. If the audio equipment used in the Ambiophonics system is not of the best quality, the amplifiers and/or speakers probably further compresses the dynamics, so it is generally wiser to tune the master volume during normal levels than during peaks in the music.

Setting the master volume is no more dramatic or difficult than doing it for an ordinary stereo system. However, with Ambiophonics, the potential of getting a realistic sound is so good that it is rewarding to spend a few more seconds to choose a realistic volume.

There is no need to worry about finding the exact "correct" volume, since there is no such thing. Within the same concert hall, the sound pressure level can differ several decibels in different seats, and it is even not necessarily closest to the musicians where the sound is the loudest.

5.8.2 Choosing reverb program

Many standard music recordings contain some added reverberation (artificially through reverb processors, or by mixing in sound from hall microphones) in order to sound more interesting on ordinary stereo systems. This may seem like a problem for Ambiophonics. Fortunately, the effect is not very dramatic, since reverberation arrive from the front even in the live situation. The added reverberation simply corresponds to a stage that reflects more reverberation than the original.

The reverberation in the recording will be reverberated by the reverb program. Thus the resulting reverb coming from the reverb array will be the original in the recording modulated with the reverb program. Again, this could at first glance look as a serious problem. However, for relatively dry recordings the effect is neglible, and for recordings with rich reverb, it modulates the given reverb program to produce a better match.

Actually, it is possible to match almost all recordings with a suitable reverberation using only three reverb programs, one small hall, one large hall and one church. It is by no means necessary to have a reverb program recorded in the same venue as the musical recording to make a seamless match, which comes as a pleasant surprise to most new users. The primary reason for this is the combination of that the reverb program both masks the reverberation on the recording and gets modulated by it.

There is really only one rule when matching reverb program, which is that the reverb time of the reverb program should be similar or longer than the reverb in the recording. A reverb program containing a small hall or studio acoustic should not be combined with a music recording made in a cathedral. Although the result probably will not be that bad, it certainly will not sound as if the recording was made in that small acoustic space. The longer reverberation takes the upper hand. There is one exception to this rule though, for recordings made with really long reverberation, the matched reverb program should be shorter. Thus, it is suitable if the church acoustics reverb program does not have extremely long reverb time.

To decide what reverb program is suitable for a specific music recording, play it with the reverb array muted (muting is done in the channel control menu), and listen. Does it sound like a small hall, a large hall, or a church? After that classification, pick any hall from the given category, and it will most certainly match very well.

It is generally possible to get very realistic results scaling up the acoustics one size, that is a recording made in a small hall can be combined with an acoustics from a large hall, and then the impression will be that the recording was made in the larger hall. Jumping two sizes, combining a small hall music recording with a cathedral reverb may however be too extreme to sound realistic. If it will work or not strongly depends on the source material, some recordings are more sensitive than others on the choice of reverb program.

The best all-around acoustics is a small hall. Thus, if the Ambiophonics system reproduces a radio program or something else where the recording venues change all the time, a small hall is best (any studio speaker voice will sound a bit funny though).

5.8.3 Setting the reverb volume

A brand new reverb array, is like a brand new subwoofer – the first days one will be tempted to have it on too high volume, just to really hear the effect of the new investment.

The reverb volume should be high enough to add envelopment, but it should not be too obvious that it is there. Instead, it should be obvious that it *was* there, when the reverb array is muted.

When music is playing, through both the ambiopole and the reverb array, start by setting the reverb volume so high that the reverb does stick out. Then lower the volume a few decibel at a time until the reverb starts to disappear. Then try to mute the reverb array, a clear difference should be heard, in the form of loss of envelopment and feel of being there. If it did not make any or very minimal sonic difference, the reverb volume was set too low.

Chapter 6

Ambisonics tutorial

6.1 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, although without height, 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`

6.2 System design tips

6.2.1 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, achievable only in purpose-built laboratories.

Most tips about the listening room for Ambiophonics, which can be read about in section 5.2.1, is also applicable to Ambisonics.

6.2.2 The speaker array

6.2.2.1 Loudspeakers and amplification

Since Ambisonics creates the desired sound field by mixing the sound from all speakers, 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 however, is that dynamic 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.

The amplification should have good signal-to-noise ratio. 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.

6.2.2.2 Layout

Ambisonics supports any form of layout. However, angular irregular arrays (such as the standard ITU 5.1 layout) require special decoding techniques, which is currently not supported by AlmusVCU, mainly because the most common technique is patented, and the patent owner does not allow free software implementations. 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 6.1 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.

6.2.3 Sweet area

The sweet spot, or sweet area, is where it is suitable to listen to the system, which for Ambisonics is in the exact center of the speaker array. 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.

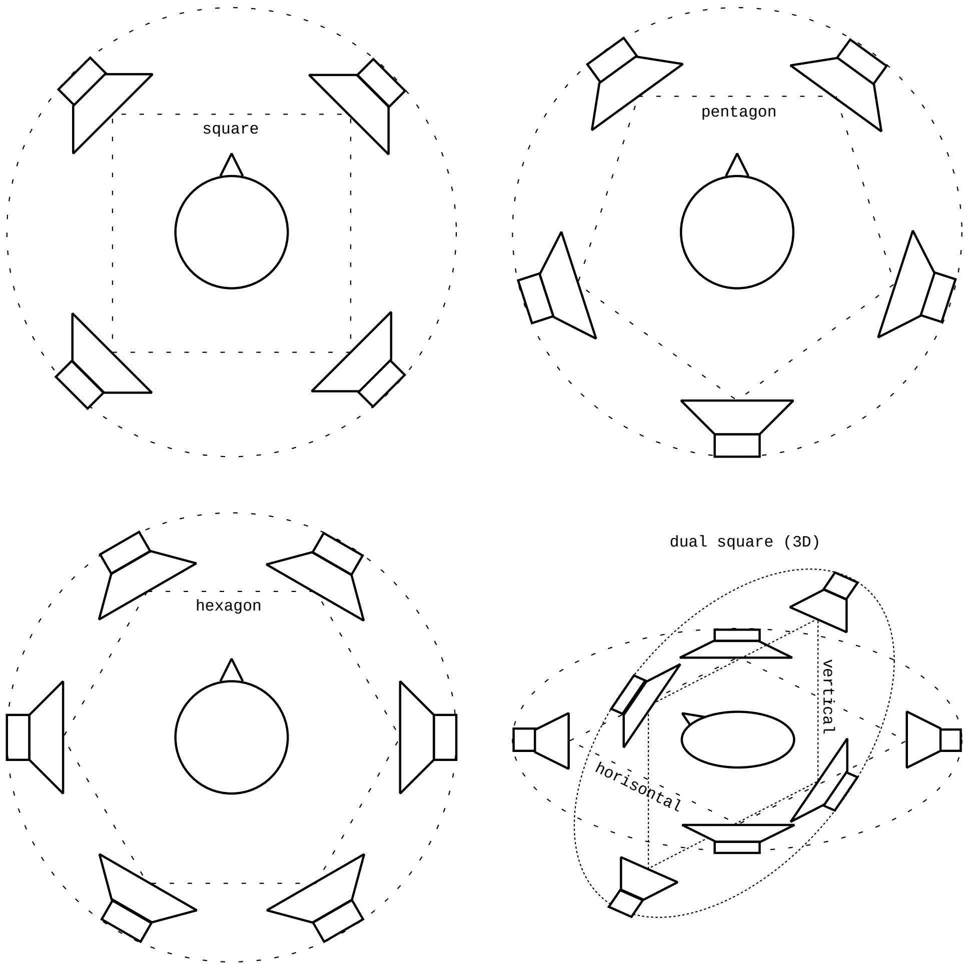

Figure 6.1: common layouts for Ambisonics.

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–three 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.

6.3 Connecting all components

Ambisonics requires four input channels for first order Ambisonics (B-format), and nine channels for second order. In figure 6.2, 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-Audio, 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 are quite many commercial UHJ recordings distributed on CD. When/if UHJ support is implemented in AlmusVCU, the signal source could be a CD player, and the link would then be S/PDIF. Any interest in UHJ support should be reported, so it gets implemented.

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. It is also possible to use the simple file playback feature of AlmusVCU described in section 3.2.5.

6.4 Installing AlmusVCU

In this tutorial it is assumed that the sound card hardware in the AlmusVCU machine provides ADAT input/output.

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.

6.5 Configuring AlmusVCU

6.5.1 Static configuration

The static configuration is described in chapter 2. This tutorial assumes that the sound card inputs 1 – 4 (0 – 3 in the static configuration) are fed with the B-format signal, and that sound card outputs 1 – 6 are connected to speakers.

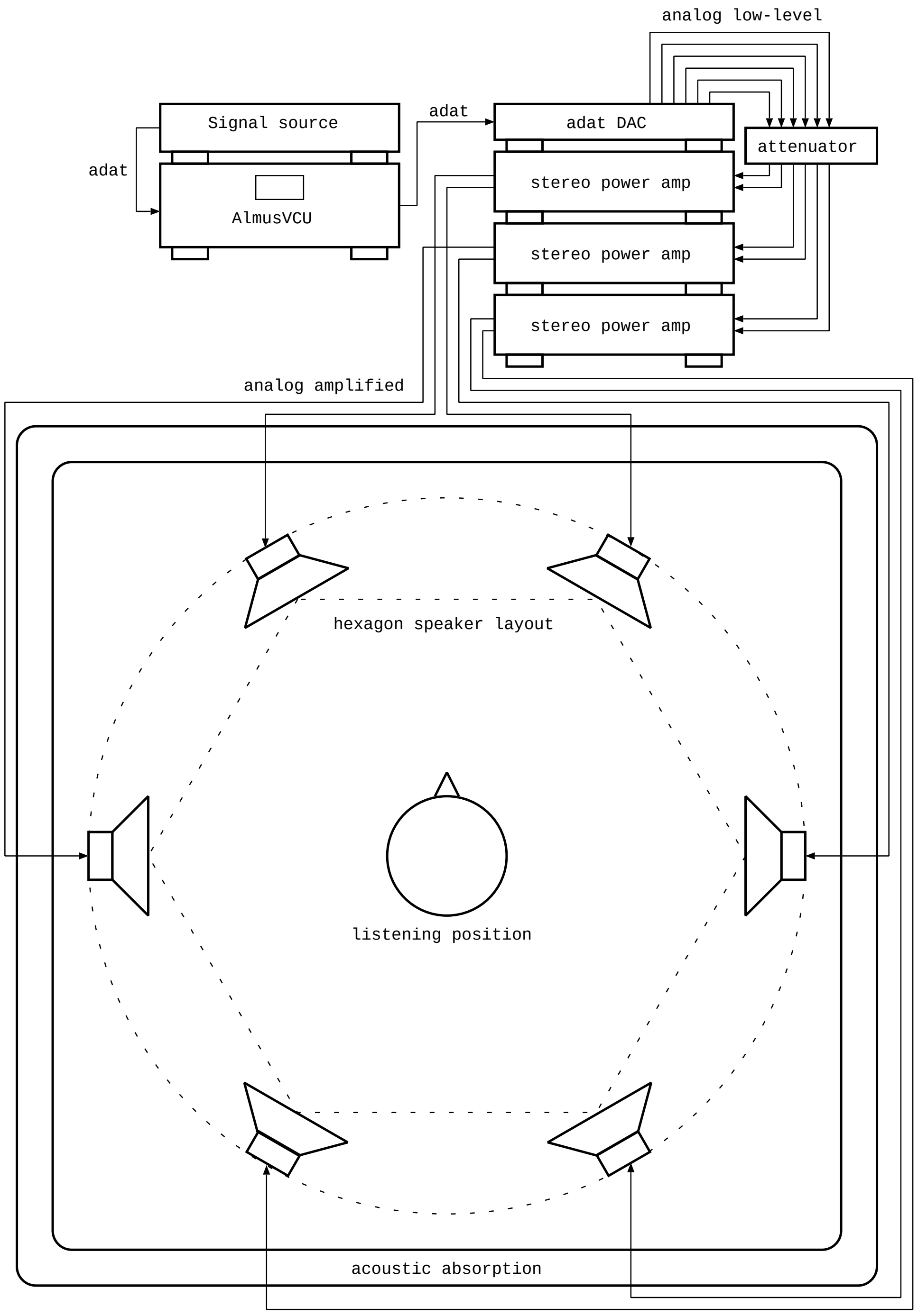

Figure 6.2: a typical Ambisonics system, with a horisontal hexagon speaker layout.

6.5.2 Setup menu

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

```
+--------------------+
| >Start convolver?  |
|  Master [-48.0]dB  |
|  Master mute [off] |
|  Load reverb prg...|
|  Del reverb prg... |
|  Tune reverb prg...|
|  Signal setup...   |
|  Channel setup...  |
|  Eq setup...       |
|  Ambiopole setup...|
|  Reverb defaults...|
|  Convolver setup...|
|  User intf setup...|
|  System info...    |
|  Maintenance...    |
+--------------------+
```

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

6.5.2.1 Signal setup

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

```
+--------------------+
| >Soft clip [off]   |
|  [24] bit output   |
|  [64] bit internal |
|  Dither [on]       |
|  Watch clock [off] |
|  Allow 44.1kHz[on] |
|  Allow 48.0kHz[on] |
|  Suggest [44.1]kHz |
+--------------------+
```

Ambisonics is not very demanding for the computer to do, so it is probably ok to run with 64 bit internal resolution. Although it probably will not lead to an improvement which is audible in practice, if the processing power is there, one could run in 64 bit just because it is possible.

If the sound hardware supports more than one sample rate, it can be a good idea to configure AlmusVCU to allow them all. This will make AlmusVCU consume more memory, but since there will be no reverb programs in Ambisonics, and those are the main memory consumers, it will probably be within bounds.

Since B-format recordings is not distributed on audio CD, one can expect both 44.1 and 48.0 kHz material.

6.5.2.2 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    |
|  Reverb setup...   |
|  Ambisonic setup...|
|  Input matrix...   |
|  Output matrix...  |
|  Speaker layout... |
|  Stage layout...   |
|  Channel groups... |
|  Channel modes...  |
|  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.

Ambisonic setup

Next, skip down to the Ambisonic setup menu and enter it. There the choice of using first order or second order source signal is made. In this example, first order is chosen.

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

I/O matrices

Next, the input matrix should be configured like this:

```
+--------------------+
| >iBw (i1)...       |
|  iBx (i2)...       |
|  iBy (i3)...       |
|  iBz (i4)...       |
+--------------------+
```

And in the output matrix menu, the ambisonic outputs should be put to the outputs 1 to 6.

Note that since the Ambisonics layout in this example is horisontal only, there is no need for the Z input channel. However, AlmusVCU requires that all four inputs are present. Instead of putting the Z input on its own input, it is possible to put it on any other input though, for example i1, and in that way save one channel. This is possible because the Z input will not lead to any change in the output when the array is horisontal only.

Speaker layout

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 o1 to o6 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 by using delay, not volume) found in the speaker layout menu. It is however strongly recommended to make the physical layout fully symmetrical, so no compensation is necessary.

Stage layout

The stage layout menu has no relevance for Ambisonics, and thus there should be no defined stages in it.

Channel groups

Channel groups define groups of channels which get their own name and volume control. For Ambisonics, it is most natural to control the volume of all speakers at once. Thus, a single channel group should be created, named "a-sonic" which controls all ambisonic outputs. The volume should be set to +0.0dB.

```
+--------------------+
|  Name [a-sonic]    |
|  Volume [+0.0]dB   |
|  Mute [off]        |
|  Inputs (0)...     |
|  Outputs (6)...    |
| >Create this group?|
+--------------------+
```

Channel modes

Channel modes are only used when creating a configuration supporting several decoding schemes, such as Ambiophonics and Ambisonics in the same configuration. This setup will only support Ambisonics, so the channel modes menu is skipped.

Trim volumes and delays

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.

6.5.2.3 Convolver setup

Before starting the convolver, the computer must be benchmarked, which is done in the convolver setup menu. Convolver settings could be could be applied manually, but the benchmark way is recommended to start with.

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.

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

```
+--------------------+
|  Max load [90]%    |
|  Low latency [off] |
|  44.1 92.9ms       |
|  48.0 85.3ms       |
| >Rerun benchmark?  |
+--------------------+
```

6.5.2.4 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 section 6.5.3.

6.5.3 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 do not use them at all. For very large arrays (larger than say four meters in radius), shelf filters should not be applied.

The purpose of the shelf filters is to slightly strengthen 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 is not one “correct” set of 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.

The shelf filter for W (omni-directional) is usually -1.5 dB below 500 Hz, and for the directional components (X, Y, and Z) they are instead +1.5 dB below 500 Hz. The magnitude response of such filters can be seen in figure 6.3.

6.5.3.1 Dynamic equaliser

Enter the equalisation setup menu (`Eq setup`), and then the equaliser design menu, and create a new FFT dynamic equaliser, with custom bands at for example 450 and

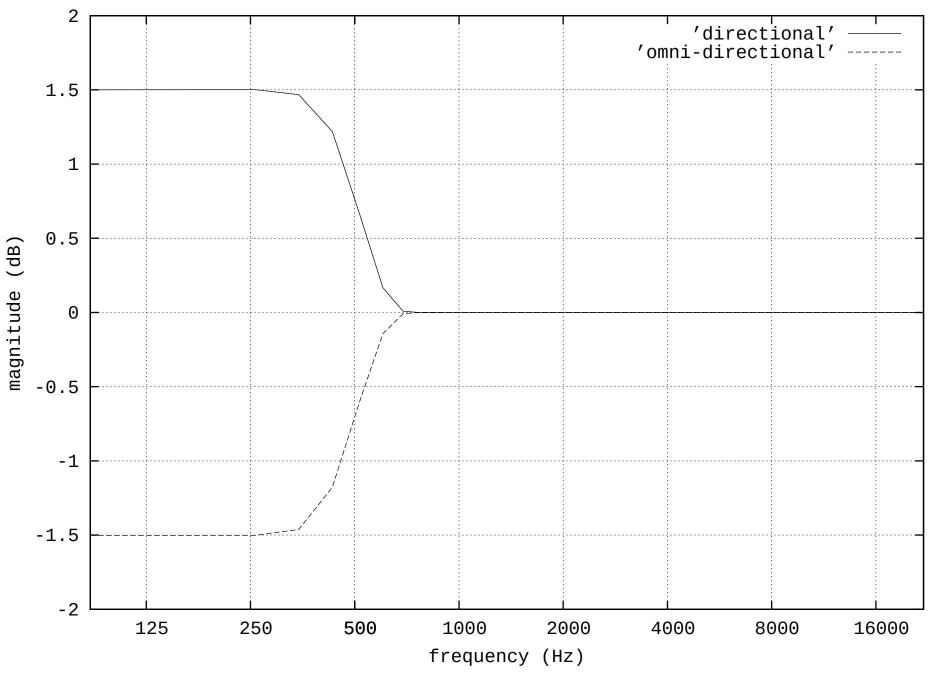

Figure 6.3: Shelf filters for the omni-directional and directional components.

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 equaliser modes, 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  |
|  Delete filter?    |
+--------------------+
```

6.5.3.2 Static equaliser

The static equaliser for iBw (that is W, the omni-directional component) is created with a 500 Hz soft-sloping (sharpness 0.50) 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 6.3 are the magnitude responses shown.

```
+--------------------+
|  lowpass 500 Hz (0.|50)...
|  op: multiply (-1.5| +)...
|  highpass 500 Hz (0|.50)...
|  op: add...        |
|  Name [shelf-dir]  |
|  Add a filter...   |
|  Add an op...      |
|  Reorder stack...  |
|  Delete this eq?   |
+--------------------+
```

The flow-through delay for an equaliser designed like the above will be as low as 3.4 milliseconds at 44.1 kHz.

Chapter 7

Bass management tutorial

7.1 Introduction

A true full-range speaker can reproduce freqencies between 20 - 20000 Hz. One can argue that the system should be able to produce frequencies a few hertz below 20 as well, although it cannot be heard, it can be felt. Only very large and very expensive speakers are truly full-range. For a traditional stereo system it may be worthwhile to buy a pair of those, but for a multi-channel system having ten speakers or more where some should be mounted in elevated positions, it is not feasible to use full-range speakers, at least not for all of them.

Bass management is the task of managing the bass in the HiFi system. The problem with bass is that it requires large speakers and lots of amplification power. Due to this, it is common that separate bass speakers are used, often called sub woofers, which handles all the bass signals that the main speakers cannot handle. The amount of bass speakers is lower than the number of main speakers. Often there are as few as only one, since bass speakers are clumsy and expensive and the human hearing system is not very good at hearing direction in the bass range anyway.

For home theaters, there is a special bass effects channel, called LFE (Low Frequency Effects), which is the "dot one" in 5.1. That can be seen as bass management on the production side, where powerful bass effects is put in a separate channel which drives a dedicated bass speaker, to spare the main speakers. However, providing a separate bass channel on the production side is a quite crude and inelegant way of bass management. The proper way to do it is to include the bass in the main media channels, and do any separation if necessary on the reproduction side (for 5.1 home theater one could argue that having a separate LFE channel is good to have anyway, since many will want to crank up the volume of the bass effects, that is explosions and such, without messing up the frequency response of other sounds).

Doing bass management at the reproduction side means splitting up each full range speaker feed into two, one low-pass filtered signal which is sent to a bass speaker, and the remaining part of the signal is sent to the main speaker. AlmusVCU supports this through its generic equaliser sub-system.

7.2 System design tips

For low-end systems, one can skip the use of bass speakers, even though the main speakers cannot reproduce bass below say 150 Hz. This is because there is not much valuable information in the bass, and that any sounds with frequencies in that range, almost always also extends up over 150 Hz, so they are heard anyway. However, for really good sound, proper bass reproduction is a must.

Proper bass reproduction means that full range is obtained, and any directional information in the bass that can be heard is preserved. How far down in frequency direction can be heard is much debated, 80 Hz seems to be a reasonable limit though, although some claim that it is lower. In any case, it is safe to assume that any direction heard is very rough below 100 Hz. Thus, this means that under 80 Hz, it should be necessary to have only one speaker, and for the higher bass, two speakers should be enough to give some sense of direction. If the cross-over frequency is really high, say 150 Hz or higher, it may be motivated to have more than two bass speakers, if it can help improving the directionality of the bass reproduction.

Note that it is generally better the lower the crossover frequency is. If the crosssover frequency is too high, it may become possible to hear that the bass is detached from the rest of the system.

7.2.1 Ambiopoles

An ambiopole is a crosstalk cancelled stereo pair. At low frequencies, crosstalk cancellation will not be effective, which means that bass will be heard as a mono source coming from the narrowly spaced speakers. Since almost all bass sounds extends over a wider frequency range, the directionality of the sounds is kept intact anyway. However, to improve bass directionality, it can be useful to use a pair of bass speakers placed at +/- 90 degrees, which handles the bass frequencies of the ambiopole.

The wide placement, +/- 90 degrees, that is straight to the sides, improves the channel separation between left and right ear.

Experiments remain to be done on the crossover frequency. It might be worthwhile to use bass speakers that go higher up in range, to say 150 - 200 Hz to use specifically with the Ambiopole. If such bass speakers are used however, they must be complemented with proper sub woofers for high end reproduction, since a woofer that reproduces up to 200 Hz well, cannot reproduce low bass.

7.2.2 Reverb arrays

The reverb array often has overhead speakers and is often used together with another component, such as an ambiopole in the case of Ambiophonics. If that other component reproduces bass well, it is generally not that important that the reverb array reproduces bass. However, for high end systems, it is recommended to place a pair of bass speakers (or use full-range speakers) at +/- 90 degrees, directly to the sides that is, since bass from those directions improves reverb envelompent.

7.2.3 Ambisonic arrays

For Ambisonic arrays, it is a good idea to create a separate ambisonic array for the bass speakers. If the crossover frequency is low, a single bass speaker could be used, which reproduces only the W signal. The higher up the crossover is, the more bass speakers should be employed, but they should always be considerably less than the main speakers. In most cases, more than two bass speakers (placed directly to the sides) should not be needed.

7.2.4 Stereo and 5.1

In stereo an 5.1 systems seldom more than one subwoofer is used. However, nothing stops from breaking this tradition, and use more. For stereo, there could be two, placed as close to the main speakers as possible if the crossover frequency is high, or else they should be placed straight to the sides.

For 5.1, it is usually more important to have full range in the front than in the back. However, a pair of side placed bass speakers will provide a good position for all speakers in the system. In such a case, the low frequency content of the center and LFE channel should be sent to both bass speakers, lowered 3 dB to compensate for that two speakers are used.

7.2.5 Advanced/hybrid setups

AlmusVCU can support several decoding systems in one speaker array, such as stereo, 5.1, Ambisonics and Ambiophonics. Naturally, one want to reuse as many of the main speakers and bass speakers for all channel modes. Fortunately, the two bass speaker placement of +/- 90 degrees work very well for most systems.

In high end setups, where still a high cross-over point is desired for example for an ambiopole, it is perfectly ok to have a separate sub woofer (or more than one) which handles the bottom range, and have other that handle the higher bass range.

7.2.6 Room considerations

The listening room, especially if small, can really mess up the bass reproduction. In many cases, it is more important to relate the bass speaker placement to the room, rather than to what would be most efficient if room effects would not have been there. The smaller the room, and the lower the bass, the more problems can be expected.

[FIXME: extend with tips on how to place bass speakers in small rooms]

7.2.6.1 Room equalisation

It may be worthwhile to equalise the bass for the room. There are some measurement based automatic equalisation systems on the market, and some time in the future, AlmusVCU may include such a feature. Meanwhile, room equalisers must be designed in external systems and be imported into AlmusVCU, or one can do a crude manual equalisation with help of a sine wave generator, a sound level meter and the built-in dynamic equalisation feature.

[FIXME: test this, and make a tutorial on it if it is any good]

7.3 AlmusVCU equalisation for bass management

7.3.1 Bass management filters

To split a signal into a lowpass (bass) and a highpass (mid-range and treble) component, a lowpass and a highpass filter are used. The lowpass and highpass filters found in the built-in equaliser designer (further described in section 3.4) have no phase distorsion, and can make a perfect split of the signal, so when the two components are added together, the exact original is restored, something which is not possible with analog filters used in traditional speaker designs.

7.3.2 Bass management setup

The following list of actions describes the recommended way to setup bass management in AlmusVCU:

1. Make a configuration which ignores bass management alltogether, that is no bass speakers, and full range to all main speakers.
2. Decide which outputs on AlmusVCU to connect the bass speakers to. There must be one free output per bass speaker feed.
3. Create bass signal routes for the bass speakers according to the recommendations found in section 7.3.3.
4. Decide the bass cutoff frequencies for all speakers. If all main speakers are equal, the frequency will be the same everywhere, which will simplify the setup.
5. Create lowpass equalisers for all cutoff frequencies, with corresponding highpass equalisers. The sharpness value should be 0.50 (soft rolloff), and recommended names of the equalisers are "lpX" for lowpass, and "hpX" for highpass, where X is replaced with the cutoff frequency. Section 3.2.7.2 gives further information on equaliser design.
6. Create an equaliser mode for bass management (see section 3.2.7.3).
   (a) Assign the proper highpass equalisers to the physical outputs connected to the main speakers.
   (b) Assign the proper lowpass equalisers to the physical outputs connected to the bass speakers. If a bass speaker needs to reproduce bass signals with different cutoff frequencies, for example when two different main speaker types, one with bass cutoff at 100 Hz, the other at 80 Hz route their bass signals to the same bass speaker, the highest cutoff frequency should be used, in this example 100 Hz, "lp100".
   (c) If necessary, assign lowpass filters on logical channel outputs were needed. This is only necessary if a bass speaker is required to handle bass signals with different cutoff frequencies, as described in the previous bullet. However, for this to work, the logical output must not be routed to a main speaker output, as that require a full range signal (which is cutoff with a highpass equaliser attached to the physical output). This means that in some cases it is required to duplicate bass signal routing.

7.3.3 Bass signal routing

Signals are routed to the bass speakers either by duplicating outputs which go to main speakers to the bass speakers, or by creating new logical channels. Both methods may be used in a complete bass management setup. Which one is chosen is mainly due to what type of logical channel that need bass management.

7.3.3.1 Ambiopole

The bass for ambiopoles should be brought through a separate passthru channel. If there is one bass speaker, the passthru channel should link both ambiopole inputs, and its output should be routed to the physical output the bass speaker is connected to. If there are two bass speakers, there should be two passthru channels, one for left and one for the right.

The reason for using separate passthru channels, instead of just duplicating the ambiopole outputs, is that the ambiopole outputs are normally processed with crosstalk cancelling filters, and thus not suitable for the bass speakers.

7.3.3.2 Reverb

The reverb outputs should simply be duplicated to the physical outputs where bass speakers are connected. Which outputs directed where is decided by the bass speaker placement. If there is only one, all reverb outputs should be linked to the physical output connected to the bass speaker. If there are one left and one right speaker (preferred), the reverb channels corresponding to reverb speakers on the left side of the room should be directed to the left bass speaker and correspondingly for the right.

7.3.3.3 Ambisonic

The ambisonic signal routing could be made in the same way as for the reverb described in section 7.3.3.2. However, the recommended way is to make a separate ambisonic decoding array for the bass speakers. This is done simply by adding one ambisonic channel per bass speaker, and indicating the proper direction.

Should there only be one bass speaker though, a single passthru with a -3 dB trim volume for the W channel input should be made for the bass speaker (which corresponds to one speaker ambisonic decoding, however the built-in ambisonic decoder requires at least two speakers to function properly, thus a passthru channel must be used instead).

7.3.3.4 Passthru

Passthru channels usually correspond to stereo or 5.1 or similar setups with a one-to-one media-to-speaker channel mapping. Bass management for these setups are made just as for reverb, described in section 7.3.3.2.

7.3.4 Performance considerations

To be done

7.4 Bass management example

To be done

Chapter 8

Advanced configuration tutorial

8.1 Introduction

AlmusVCU allows for combining several reproduction systems, and is able to switch between them in runtime. For example, one can combine Ambisonics, Ambiophonics, stereo and ITU 5.1 into a single system. In this tutorial, an example of such a setup will be described, with the additional experimental formats Ambiophonics for 5.1 recordings and Perambio, a hybrid between Ambisonics and Ambiophonics. Apart from using the concepts described in the tutorials in previous chapters, the AlmusVCU features which make it possible to do detailed customisation of combined reproduction systems will be demonstrated. Those features will be the main focus of this tutorial.

8.2 System design

In this example, eight main speakers and two bass speakers are used. As suggested in the bass tutorial chapter, the bass speakers are not integrated into the system until the rest is ready.

Of the four systems in question, all can compromise the speaker layout somewhat, except Ambisonics, which require a precise symmetric layout. Also the ambiopole in the Ambiophonics layout will require precise placement. Thus, a horisontal-only hexagon layout is chosen for Ambisonics, and the remaining two speakers are put up as an ambiopole up front, as seen in figure 8.1.

The ITU 5.1 layout will have to use the ambiopole speakers in combination as the center speaker, and use the +/- 90 degree speakers as the surrounds. The ITU standard places them at +/- 120 degrees, so this is a compromise, however a very good one, since the +/- 90 placement provides for even better envelopment, and is therefore often used even when the standardised +/- 120 placement is possible to use. Left and right speakers for both stereo and ITU 5.1 are at the standardised +/- 30 degrees.

The bass speakers are not shown in the figure. The two experimental formats Ambiophonics for 5.1 recordings, and Perambio both use all available channels (just as Ambiophonics).

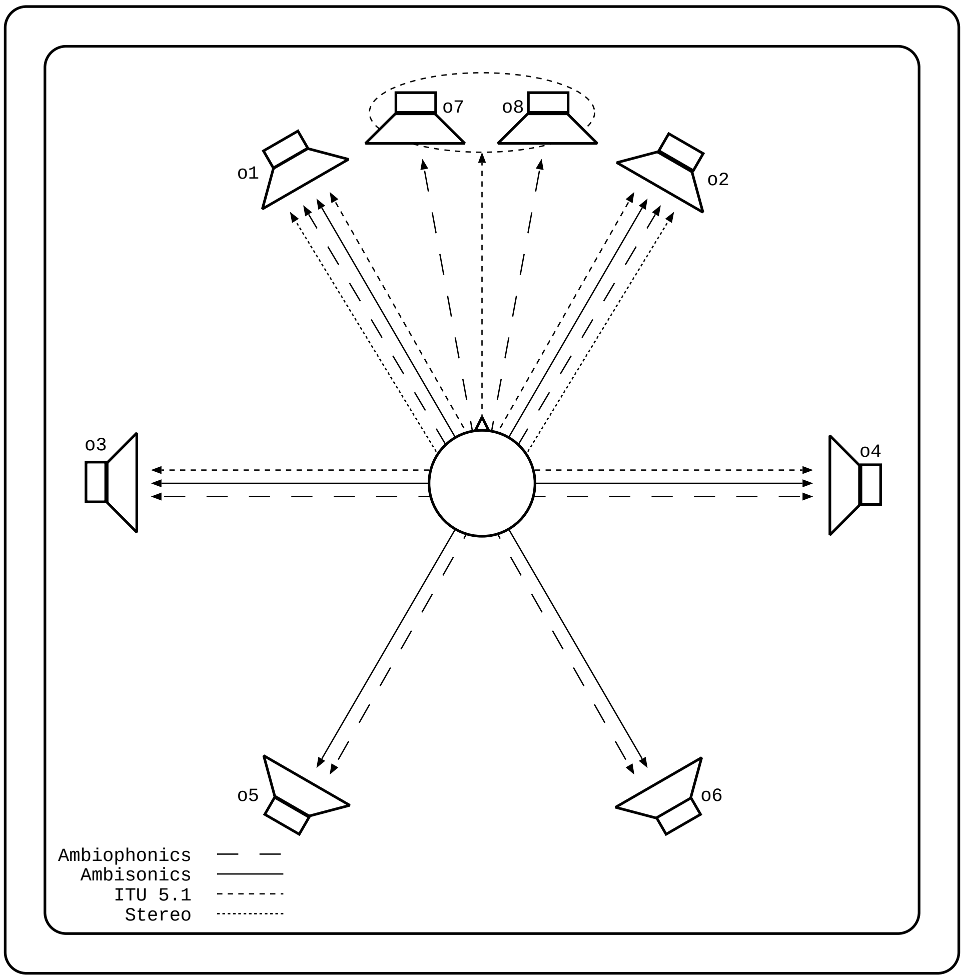

Figure 8.1: Combined Ambiophonics, Ambisonics, ITU 5.1 and stereo speaker layout.

8.2.1 Volume control

A central part of configuring AlmusVCU to be easy-to-use for a multi-format system, is to design the volume controls appropriately. All formats will share the same master volume control and the individual channel trim volumes. However, each format can have their individual volume controls controlling a specified group of channels. These volume controls will be add to the master and trim volumes.

Table 8.1 shows a suggested volume control design for the seven formats. As seen in the table, it does not need to be complicated, most formats can do with only a master volume control. Extra volume controls should only be added if they will be accessed frequently, such as the reverb offset volume for Ambiophonics.

Ambiophonics	Ambiopole bound to master volume, offset volume for the reverb channels.
Ambisonics	All channels bound to master volume.
ITU 5.1	All channels bound to master volume.
Stereo	All channels bound to master volume.
Ambiophonics for 5.1	Ambiopole bound to master volume, offset volume for the reverb channels.
Perambio	Ambiopole bound to master volume, offset volume for the ambisonic channels.

Table 8.1: Volume controls for the formats.

8.2.1.1 Ambiophonics volume control

The ambiopole should be bound to the master volume, the reverb channels should have its own offset volume.

8.2.1.2 Ambisonics volume control

8.3 Connecting all components

This system will require ten outputs, one for each speaker. It is assumed that an ADAT + S/PDIF card is used for this, so there are eight ADAT channels, and two S/PDIF channels. The S/PDIF output is used for the bass speakers, and the ADAT outputs are used for the main speakers.

The input part is a bit more complicated. ITU 5.1 (pronounced “five point one”) require six channels, Ambiophonics and stereo two channels, and the horisontal Ambisonics array four. Actually, a horistontal only Ambisonics layout requires only three channels, since the Z input will be redundant, but AlmusVCU require that all four inputs are present.

It is natural for stereo and Ambiophonics and left/right channels of ITU 5.1 to share the same inputs, and assuming that there are 10 inputs available, eight ADAT and two S/PDIF, with the latter as the two last inputs i9 and i10, the input table seen in 8.2 is suggested.

The S/PDIF input could be connected to a CD player when playing stereo and Ambiophonics. As usual, it may be a bit more complicated to connect sources with ITU 5.1 and Ambisonics material. The problem with Ambisonics is that there is no

	i1	i2	i3	i4	i5	i6	i7	i8	i9	i10
Ambiophonics									L	R
Ambisonics	W	X	Y	Z						
ITU 5.1					C	Ls	Rs	LFE	L	R
Stereo									L	R
Ambiophonics for 5.1					C	Ls	Rs	LFE	L	R
Perambio	W	X	Y	Z					L	R

Table 8.2: Input mappings for the formats.

standard, and the problem with ITU 5.1 is that the coding schemes are closed and not available for open-source projects like AlmusVCU. However, there are hardware converters if a pure digital input is desired, or else the alternative is to go through a A/D converter. Since most ADAT D/A converters are combined with a A/D converter that alternative is the most cost effective, and the signal degradation of that extra step through the analog domain is less than most tend to believe.

In the example, the Ambisonics Z input has got its own channel. This is most practical since one can expect that the source signal will have four channels. However, it could be assigned to any input, since the data received on it will not be used because the Ambisonics layout is horisontal only.

8.4 Installing AlmusVCU

Information on which hardware to choose and how to install the software is found in chapter 2. For the configuration given in this tutorial, a 1000 MHz processor or faster is recommended, together with 256 megabytes or more RAM.

8.5 Configuring AlmusVCU

8.5.1 Static configuration

The static configuration is described in chapter 2. This tutorial assumes that there are ten sound card inputs and outputs, and the channels 1 – 8 (0 – 7 in the static configuration) are ADAT, and the remaining two are S/PDIF.

8.5.2 Setup menu

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

```
+--------------------+
| >Start convolver?  |
|  Master [-48.0]dB  |
|  Master mute [off] |
|  Load reverb prg...|
|  .
|  .
```

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

8.5.2.1 Signal setup

The `Signal setup` menu is the first menu to enter. Usually, the defaults are suitable. If the sound card supports several sample rates, another than the one selected per default may be desired though.

8.5.2.2 Channel setup

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

```
+--------------------+
| >[1] (A)mbiopole   |
|  [6] (R)everb      |
|  [6] am(B)isonics  |
|  [5] (P)assthru    |
|   .
|   .
```

As seen, there should be one ambiopole, six reverb channels, six ambisonics channels, and five passthru channels. There will be extra channels required for the bass speakers, but it is generally better to save bass management to last, or else it can be a bit too much to think about at once.

Reverb setup

In the reverb setup menu, the source layout for the reverb engine is chosen. For standard Ambiophonics, it should be stereo, which is the default. However, with the experimental format Ambiophonics for 5.1 recordings, it is better set to LRC, employing all three stage originating channels for the reverb engine:

```
+--------------------+
| >Layout [lrc]      |
+--------------------+
```

Note that when the system is run in standard Ambiophonics mode, meaning that the center channel will not be available, the reverb engine still will process for three channels meaning that more processor time will be consumed. Thus, if the processor is slow, and Ambiophonics for 5.1 will be seldomly used, it may be better to keep the reverb engine source layout in stereo.

Ambisonic setup

Next, skip down to the Ambisonic setup menu and enter it. There the choice of using first order or second order source signal is made. In this example, first order is chosen.

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

I/O matrices

To match the input mapping table 8.2, the input matrix menu should look like this:

```
+--------------------+
| >iA1 (i9)...       |
|  iA2 (i10)...      |
|  iRl (i9)...       |
|  iRr (i10)...      |
|  iRc (i5)...       |
|  iBw (i1)...       |
|  iBx (i2)...       |
|  iBy (i3)...       |
|  iBz (i4)...       |
|  iP1 (i9)...       |
|  iP2 (i10)...      |
|  iP3 (i5)...       |
|  iP4 (i6)...       |
|  iP5 (i7)...       |
+--------------------+
```

Note that the LFE channel is not included yet. The iP1 – iP3 channels correspond to left, right and center channels for ITU 5.1, and iP4 and iP5 are left and right surround inputs.

The output matrix is configured to match the suggested layout in figure 8.1. Thus, the output matrix menu should look like this when left:

```
+--------------------+
| >oA1 (o7)...       |
|  oA2 (o8)...       |
|  oR1 (o1)...       |
|  oR2 (o2)...       |
|  oR3 (o3)...       |
|  oR4 (o4)...       |
|  oR5 (o5)...       |
|  oR6 (o6)...       |
|  oB1 (o1)...       |
|  oB2 (o2)...       |
|  oB3 (o3)...       |
|  oB4 (o4)...       |
|  oB5 (o5)...       |
|  oB6 (o6)...       |
|  oP1 (o1)...       |
|  oP2 (o2)...       |
|  oP3 (o7,o8)...    |
|  oP4 (o3)...       |
|  oP5 (o4)...       |
+--------------------+
```

Speaker layout

The speaker layout should be set to exactly match the actual layout of the speakers, as seen in figure 8.1, and specified in table 8.3.

	o1	o2	o3	o4	o5	o6	o7	o8	o9	o10
azimuth	-30	30	90	-90	150	-150	10	-10	-	-
elevation	0	0	0	0	0	0	0	0	-	-

Table 8.3: Speaker layout.

The distance to the speakers should also be specified, which should reflect the actual distances, and be the same for all speakers. The exact angle for the ambiopole speakers will be tuned during cross-talk cancellation calibration. An approximate angle such as the suggested +/- 10 degrees, is enough for the speaker layout menu. For the ambisonic speakers however, the layout must be a perfect match.

Stage layout

In the stage layout menu, a 150 degree wide stage, 40 degrees high should be specified for the ambiopole. The stage layout menu should look like this when left:

```
+--------------------+
| >S1 azim [0]deg    |
|  S1 elev [0]deg    |
|  S1 width [150]deg |
|  S1 height [40]deg |
|  Add stage?        |
|  Delete last stage?|
+--------------------+
```

The stage layout is used by the reverb engine to make rendering decision based on from where direct sounds are generated.

Channel groups and modes

The channel groups and channel modes menus are central for a multi-format configuration such as in this tutorial. Channel groups are groups of channels which get their own names and volume controls in the user interface. Channel modes are collections of those channel groups which define one format. Thus the channel modes will directly correspond to Ambiophonics, Ambisonics, ITU 5.1 etc in this example. Sections 3.2.6.10 and 3.2.6.11 describes channel groups and modes in detail, and provide recommendations for how to assign them. Those recommendations and the volume control discussion in section 8.2.1 have been followed in the suggested assignment seen in table 8.4. Note that even if a format only needs the master volume control (the Ambisonics, ITU 5.1 and stereo formats in this example), a channel group including all channels for the format must be created anyway, in order to make it possible to create a channel mode for the format. The volume control of such a channel group will however be redundant, and will not be used in the user interface.

The suggested assignment uses the advanced concept of multiple groups with the same name, in the table indicated with and index within parentheses following the channel group name. The purpose of doubling the “a-sonic” group is to provide separate volume controls for the group in the Perambio and Ambisonics channel modes. Likewise, the “surrnd” group is doubled for separate volume control of the 5.1 surround channels for channel modes ITU 5.1 and Ambiophonics 5.1. However,

Channel mode	Channel groups	Controlled channels
ambiophonics	a-pole(1)	oA1 – oA2
	reverb	oR1 – oR6
ambisonics	a-sonic(1)	oB1 – oB6
itu 5.1	lrc	oP1 – oP3
	surrnd(1)	oP4 – oP5
stereo	stereo	oP1 – oP2
perambio	a-pole(1)	oA1 – oA2
	a-sonic(2)	oB1 – oB6
ambiophonics 5.1	a-pole(2)	oA1 – oA2, oP3
	reverb	oR1 – oR6
	surrnd(2)	oP4 – oP5

Table 8.4: Channel modes and channel groups assignments.

the two "a-pole" groups differ on which channels they control. In the standard Ambiophonics channel mode, only the ambiopole channels (oA1 and oA2) are included in the group, while the 5.1 version of Ambiophonics also includes oP3, which is the center channel in 5.1. Thus, the "a-pole" group in Ambiophonics 5.1 reproduces LRC (left/right/center) of 5.1 by crosstalk-cancellation of the left and right channel, and mixing in the center unprocessed.

Some groups are overlapping, that is they control the same channels. Most obviously, the doubled groups do so. However, within a single channel mode there is no overlap in the groups, and thus the channel group volume controls will function isolated from each other.

The indexes following the names of the doubled channel groups is shown in the table for clarity, but should not (and cannot) be included in the actual names in the AlmusVCU configuration. Thus there will be a few groups which have the same names, so one need to be careful not to mix them up when they are assigned to the different channel modes. In the channel group creation menu below, the "a-pole" group is about to be created:

```
+--------------------+
|  Name [a-pole]     |
|  Volume [+0.0]dB   |
|  Mute [off]        |
|  Inputs (0)...     |
|  Outputs (2)...    |
| >Create this group?|
+--------------------+
```

The volume for all channel groups should be set to +0.0 dB (it is -48.0 dB per default), and only channel outputs should be controlled (those indicated in the table). After all channel groups have been created, the channel group listing should look like this:

```
+--------------------+
|  Edit "a-pole"...  |
|  Edit "reverb"...  |
|  Edit "a-sonic"... |
|  Edit "lrc"...     |
|  Edit "surrnd"...  |
|  Edit "stereo"...  |
|  Edit "a-pole"...  |
|  Edit "a-sonic"... |
| >Edit "surrnd"...  |
|  New channel grp...|
|  Reorder groups... |
+--------------------+
```

The groups have been ordered such that the doubled groups have their version 1 first and version 2 towards the end of the list. These channel groups should then be assigned to channel modes, which is done in the channel modes menu. The names and group assignments should be as indicated in table 8.4. Below the Ambiophonics channel mode is shown, which consists of the “a-pole” and “reverb” channel groups:

```
+--------------------+
|  Name [ambiophonics|]
|  "a-pole"  [on]    |
|  "reverb"  [on]    |
|  "a-sonic" [off]   |
|  "lrc"     [off]   |
|  "surrnd"  [off]   |
|  "stereo"  [off]   |
|  "a-pole"  [off]   |
|  "a-sonic" [off]   |
|  "surrnd"  [off]   |
| >Create this mode? |
+--------------------+
```

After all channel modes have been created, the channel mode listing should look like this:

```
+--------------------+
|  Mode [ambiophonics|]
|  Edit "ambiophonics|"...
|  Edit "ambisonics".|..
|  Edit "itu 5.1"... |
|  Edit "stereo"...  |
|  Edit "perambio"...|
|  Edit "ambiophonics| 5.1"...
|  New mode...       |
|  Reorder modes...  |
+--------------------+
```

The mode is set to off per default, meaning that all channel groups will be active, which would be a incomprehensible cacophony in this case, so one mode should be selected. In the example, the channel mode “ambiophonics” has been selected.

Trim volumes and delays

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.

Note that delay and trim volumes are global settings that will affect all channel modes.

8.5.2.3 Ambiopole setup

Next leave the channel setup menu, skip equalisation setup (the `Eq setup` menu) and enter the ambiopole setup menu. Load a crosstalk cancellation program and activate crosstalk cancellation for the ambiopole.

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

8.5.2.4 Reverb defaults

The reverb defaults menu specifies the default reverb program settings which is applied when a reverb program is loaded for the first time. The settings should be the same as described in the Ambiophonics tutorial, that is the following:

```
+--------------------+
| >[Stage adapted]   |
|  [Decorr as needed]|
|  Force LRdelay[off]|
|  Force decay [on]  |
|  Mix with dry [off]|
|  Dry vol [+0.0]dB  |
|  B-format dir [1.4]|
+--------------------+
```

8.5.2.5 Convolver setup

Next, the convolver setup menu is entered, and there the benchmark menu is chosen. Note that AlmusVCU will run all channel modes in parallel, that is all seven reproduction formats. When a specific channel mode is selected, the channels used by all other channel modes will be muted, but they will still be active. This means that having a single configuration with multiple formats such as in this example will be more demanding on the CPU than having multiple configurations with one format in each (which also is possible). The great advantage of having all formats in one configuration is that the formats can immediately be switched, without delay.

With the powersave option activated, muted channels will not consume much processor time, however, the benchmarking code assumes that all channels will be processed. This means that manual tuning of the convolver parameters can yield considerably better result in some situations. In any case, it is best to start with benchmarking.

If the computer is well-performing latency-wise, the `Max load` setting could be increased to 90% from the default conservative 85%.

After the benchmark has been run, an alternative which has at least 3.0 seconds reverb length should be picked, 6.0 seconds or more is considered good. It could look like this before the menu is left:

```
+---------------------+
|  Max load [90]%     |
|  Max length [12.0]s|
|  Low latency [off] |
| >44.1 [372ms,4.8s] |
|  Rerun benchmark?  |
+---------------------+
```

8.5.2.6 User interface setup

For Ambiophonics, it is typical to adjust the reverb array volume relative to the ambiopole. Therefore, it is good if the reverb offset volume is easily accessed from the runtime main screen. However, it is not necessary to adjust the ambiopole level, if that is not at the correct volume, the reverb level should be adjusted instead.

Thus, in the user interface setup menu, the reverb volume and master volume controls are activated, and the ambiopole volume control is de-activated.

For the other modes, the master volume is the only one necessary to control, except for Ambiophonics 5.1 which uses a reverb offset volume (same as standard Ambiophonics), and Perambio which uses an offset volume for the ambisonic array. Thus the volume control for the second "a-sonic" channel group is activated as well. With this setup, the volume controls matches table 8.1.

```
+---------------------+
|  Screen dimmer [1] |
|  Autostart [off]    |
|  Volume step [1.0] |
|  Ch mode [sel on]   |
|  Master vol [on]    |
|  A-pole vol [off]   |
|  Reverb vol [on]    |
|  A-sonic vol [off] |
|  Lrc vol [off]      |
|  Surrnd vol [off]   |
|  Stereo vol [off]   |
|  A-pole vol [off]   |
| >A-sonic vol [on]   |
|  Surrnd vol [off]   |
+---------------------+
```

Note that the user interface setup menu only specifies which volume controls that will be available in the runtime main screen, where one want no more than the most common controls. The menu also lists all channel groups, but when AlmusVCU is configured with different channel modes like in this tutorial, only the channel groups active in the current channel mode will be accessible, and thus only their volume controls. Thus, for the Ambiophonics channel mode only the master and reverb volume control will be available in the runtime main screen, with the suggested user interface configuration here.

All channel group volumes and trim volumes available in the active mode can be accessed in runtime from the runtime channel control menu.

8.5.2.7 Load reverb program

Before being able to start the convolver, a reverb program must be loaded. This is necessary when there is at least one channel mode that uses the reverb engine, even if no such channel mode is the currently selected.

Reverb programs are loaded in the `Load reverb prg` menu.

8.5.2.8 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.

Below are four examples of how the four-line version of the runtime main screen can look like in different situations with this configuration:

```
+--------------------+   +--------------------+
|    BERWALD HALL    |   |  Ambiophonics 5.1  |
| [pick reverb prog] |   | [alter master vol] |
|pk-88.3  master-12.0|   |pk+5.3   master:mute|
|69/44.1  reverb-10.0|   |20/48.0 a-sonic+1.0 |
+--------------------+   +--------------------+

+--------------------+   +--------------------+
|      ITU 5.1       |   |      Perambio      |
|[pick channel mode] |   |[alter a-sonic vol] |
|pk-88.3  master-12.0|   |pk-inf   master-10.0|
|34/44.1             |   |20/48.0 a-sonic-6.0 |
+--------------------+   +--------------------+
```

8.5.3 Bass management

The current configuration is complete, apart from the bass management.

To be done.

8.6 Checklist for advanced configurations

To be done.

Chapter 9

Tips and tricks

9.1 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: 0.85      |
| 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 is 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.

9.2 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 one - three input channels depending on configuration that need to be distributed to all of them. In a parallel setup, 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 feature to run AlmusVCU as a "reverb slave" to a master machine may be implemented.

9.3 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.

Multi-channel sound transport formats such as ADAT or MADI will only support half amount of channels if used at higher sample rates than 48 kHz. This means that a 16 channel sound card becomes an 8 channel sound card at higher sample rates. AlmusVCU does not support switching the amount of channels of the sound card hardware, so in this case, only one of the lower and higher range of sample rates must be configured in the static configuration.

Chapter 10

Developer information

10.1 Filter program file format

10.1.1 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: "reverb b-format";
name: "Auditorium Paganini";
longname: "Auditorium Niccolo Paganini -
Parma Italy"; source_layout: "stereo"
sets: 1;
scale: 9.0;
rates: 44100;
format: "float_le";
length: 2.600634921;
predelay: 0.0;
files: "parma-aud-left-w.pcm.bz2",
       "parma-aud-left-x.pcm.bz2",
       "parma-aud-left-y.pcm.bz2",
       "parma-aud-left-z.pcm.bz2",
       "parma-aud-right-w.pcm.bz2",
       "parma-aud-right-x.pcm.bz2",
       "parma-aud-right-y.pcm.bz2",
       "parma-aud-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:

- `float64_le`, 32 bit floating point, little endian.
- `float64_be`, 32 bit floating point, big endian.
- `float_le`, 32 bit floating point, little endian.
- `float_be`, 32 bit floating point, big endian.
- `s32_le`, 32 bit signed integer, little endian.
- `s32_be`, 32 bit signed integer, big endian.
- `s24_le`, 24 bit signed integer, little endian.
- `s24_be`, 24 bit signed integer, big endian.
- `s16_le`, 16 bit signed integer, little endian.
- `s16_be`, 16 bit signed integer, big endian.
- `s8`, 8 bit signed integer.

For the floating point formats, zero decibel (no attenuation) corresponds to +/- 1.0. For the integer formats, zero decibel corresponds to the number of bits minus 1, to the power of 2, that is 128 for 8 bit, 32768 for 16 bit, and 8388608 for 24 bits.

The `scale` field specifies how many decibels the channels should be scaled. This is used to scale a recorded impulse response to the appropriate volume, without changing the recorded data.

The `sets` field is only used by reverb programs, and spcifies how many decorrelated version of all impulse responses there are.

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 type field specifies the type of the filter program, which in the example is a reverb program, based on B-format. The remaining fields in the example are specific for reverb programs, and is described below.

10.1.2 Reverb program

A reverb program contains impulse responses recorded in some acoustic venue, typically a high class concert hall.

The impulse responses are recorded in, or around, the best listening position in the acoustic venue, with the sound source located where the sound source naturally should be, which usually is on stage if the venue is a concert hall. A reverb program has a source layout (the `source_layout` field in the index file) which either is “center”, “stereo” or “lrc”. If it is center, it means that only one source position has been used, typically on the center of the stage, and if it is stereo, two source positions have been used, typically to the left and right of the stage. The third variant, “lrc”, which means left/right/center, has three source positions, typically to the left, to the right and in the center of the the stage.

The purpose of having more than one source position is that the recording which is reverberated by the reverb program usually has more than one channel, for stereo recordings left and right, where the left channel of the recording is roughly coming from the left side of the stage, and the corresponding for the right side. Thus a stereo reverb program is made to match a stereo recording.

A reverb program can be recorded with several decorrelated sets as well. All sets should be equal in terms of what they represent, but should be reasonably well decorrelated.

The impulse response files are ordered like this in the reverb program:

- Sample rate 1
  - Set 1
    - Source position 1
      - Receiver 1 impulse responses. A single impulse response for traditional reverb programs, and four (W, X, Y and Z) for B-format reverb programs.
      - Receiver 2
    - Source position 2
      - ...
  - Set 2
    - ...
- Sample rate 2
  - ...

In the simplest case, there is only one sample rate, one set, one source position (“center” source layout) and only one impulse response.

The impulse responses recorded should be full, that is contain the direct sound, and be long enough so the reverb has decayed fully. However, in some cases, for example when sampling a reverb processor as sound source, the direct sound or even the early part of the impulse response may have been cut away. If so, it should be specified how long part of the start of the impulse response has been cut away, which is specified in the `predelay` field, in seconds. The reverb engine will then if necessary recreate a new direct sound for the reverb program.

All impulse responses must have the same time alignment (if not, the `predelay` field must be used to correct for it). It should start at the time when the direct sound reaches the microphone. Thus, the sound propagation delay from source to receiver should not be included.

The volume of the impulse responses should be at the level such as if a dry sound is reverberated with a frontal impulse response for traditional reverb programs, or with the W channel for B-format reverb programs, the reverberated sound should by the ear be heard at the same volume as the dry sound itself. That is, only reverberation should be added when dry sound is convolved with it, the heard volume should not change.

Instead of scaling the recorded samples to a proper volume, instead specify in the reverb program index file how many decibel, positive or negative, the samples should be scaled, by using the `scale` field.

10.1.2.1 Traditional reverb program

For the traditional reverb program, a receiver layout should be specified. This is done with the receiver field, which could look like this:

```
    receivers: 45/0, -45/0, 135/0, -135/0;
```

The above example specifies four receivers, with azimuth directions 45, -45, 135, and -135, all with zero elevation. These directions specify which direction compared to the listening position a given impulse response corresponds to. With four receivers, there are four recorded impulse responses per source position, set and sample rate. Although directions are specified, it is not necessary to use cardiod microphones.

The receiver layout must follow these rules:

- If there is only one receiver, the azimuth/elevation must be 0/0.
- If there are more than one receiver, they must either be symmetrically placed around the X axis, or only be to the left side (positive azimuths).
- It is not allowed to have more than one receiver in the same direction.
- Any receiver with elevation 90 or -90 must have azimuth 0.

To get comparable performance with high quality B-format reverb programs, it is recommended to have at least four receivers evenly distributed around the listening position. It is also recommended to use cardioid microphones at least for rear receivers to get a proper directional response. Note that cardioid microhpones typically needs heavy amplitude equalisation.

To get a good sounding reverb program, it is better the more impulse responses there are, many either through more receiver directions, or recording different sets.

10.1.2.2 B-format reverb program

A B-format reverb program contains B-format impulse responses. Each B-format impulse response consists of four files, the W, X, Y and Z channels separately.

The type field is set to (exactly) `"reverb b-format"`.

Optionally, the B-format program may contain several receivers such as a traditional reverb program.

10.1.3 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"`. Below an example of what an index file may look like:

```
type: "crosstalk cancellation";
name: "XTC Basic";
format: "float_le";
rates: 44100;
length: 0.1;
files: "directpath.pcm.bz2", "crosspath.pcm.bz2";
```

10.1.4 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"`. Below an example for what an index file may look like:

```
type: "equalisation";
name: "Room eq filter 1";
format: "s24_le";
rates: 44100;
length: 0.093;
files: "roomeq1.pcm.bz2";
```

10.2 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_mouse.c`, the `ki_mouse` module which get user input from the mouse, by directly accessing the mouse device.
- `twki_curses.c`, the `twki_curses` module which uses ncurses to put the user interface to the computer console. It is a combined user input and screen output module (hence the twki prefix).
- `twki_svga.c`, the `twki_svga` module which uses svgalib to put the user interface to the full screen of the user interface.
- `tw_matrix_orbital.c`, the `tw_matrix_orbital` module which uses a Matrix Orbital LCD2041 or VFD2041 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, and `twki_` are combined. The makefile included in the source archive shows how to compile the modules correctly. The resulting files should be named as `<module name>.vcu`.

10.3 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.
- If parsing of the user configuration file fails, for example after a software upgrade or a bug, the failed user configuration file will be copied to a backup called `almusvcu_user_config.failed` whereafter the current user configuration is reset to defaults.
- 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:

- The AlmusVCU log, named `almusvcu.log`, normally useful only for debugging. The log file rotates, and saves the old log to the file `almusvcu.log.old`.
- A set of BruteFIR logs, normally only used for debugging (will only appear if debug mode is activated). However, if the convolver refuses to start, these logs can provide some help for normal users as well. The logs are `brutefir_cli_input`, `brutefir_cli_output`, `brutefir_console_output` and `brutefir_last_config`. The respective logs contain the CLI input/output, console output and the configuration file used at start. The console 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.
- A local socket, called `.bfsock`, which is used in inter-process communication between the AlmusVCU process and BruteFIR.
- If the equaliser properties menu is opened and debugging is activated, 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.