CDP Multi-Channel Toolkit

~ by R W Dobson ~

Multi-Channel File Handling Functions

INTRODUCTION
About the Multi-Channel Toolkit, 'surround sound' and Ambisonic B-Format

ABFPAN
Apply a fixed or orbiting 1st order B-Format pan to mono soundfile
ABFPAN2
Apply a fixed or orbiting 2nd order B-Format pan to mono soundfile
CHANNELX
Extract all or selected channels from a multi-channel soundfile
CHORDER
Reorder soundfile channels in multi-channel soundfile
CHXFORMAT
Modify WAVE_EX header to change GUID and/or speaker positions
COPYSFX
Copy soundfiles / convert from one format to another
FMDCODE
Decode 1st or 2nd order B-Format soundfile to a choice of speaker layouts
INTERLX
Interleave mono or stereo files into a multi-channel file
NJOIN
Concatenate multiple soundfiles into a single file, with optional CUE list for CD burning
NMIX
Simple mix of two multi-channel soundfiles, with optional offset
PAPLAY
Playback of multi-channel soundfiles
RMSINFO
Scan file and report RMS and average power level statistics
SFPROPS
Display soundfile details, with WAVE_EX speaker positions

Introduction

About the Multi-Channel Toolkit

The CDP MULTI-CHANNEL TOOLKIT comprises a suite of command-line programs (for Windows, OS X and Linux) to assemble and manipulate the channels of multi-channel soundfiles. It supports the standard PCM WAVE and AIFF file formats, with a special emphasis on Microsoft's **WAVEFORMATEXTENSIBLE** (WAVE_EX) file format.

The WAVE_EX format incorporates two features of special interest - the ability to associate one of eighteen named speaker positions with a channel (e.g. for 5.1 surround sound), and support for third-party custom file formats. In this respect the TOOLKIT supports and demonstrates the new **AMB file format** for Ambisonic B-Format surround audio, based on WAVE_EX. An important use of the TOOLKIT is to assemble individual audio channels into a B-Format file (see **INTERLX**) for use with one of the many decoders now available as standalone applications or plugins. It will also decode a B-Format file in WAVE_EX format (see **FMDCODE**).

The TOOLKIT programs do not perform any signal processing on the audio (except where B-Format encoding or decoding is applied). A comprehensive suite of tools for processing multi-channel files can be found in CDP's **MULTICHANNEL** program group.

To get reasonable use out the programs a **multi-channel soundcard** is required - i.e., one that provides a 'device' which can accept a single file up to the number of channels the card supports. Some of the programs have valuable use even without a multi-channel card. For example, **PAPLAY** can mix a quadrophonic soundfile down to stereo, while **COPYSFX** is the CDP's "Swiss-army knife" for file-format conversion and will convert between a 'standard' multi-channel file and a WAVE_EX file.

The following sample formats (WAVE, WAVE_EX, AIFF, AIFF_C) are supported:

- 16-bit integer
- 32-bit integer
- 32-bit floating-point (AIFF written in AIFF-C format)
- 24-bit ('packed')

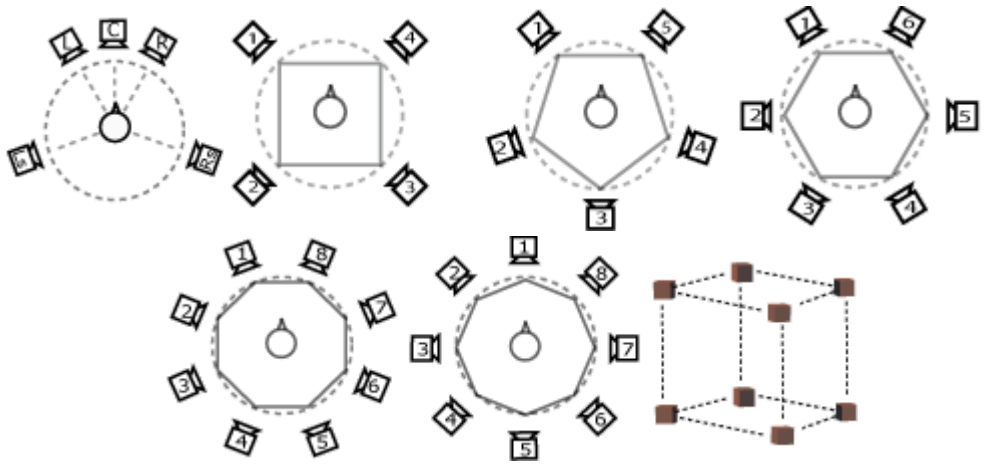
Note also:

- All programs support the maximum 4GB files sizes available in the WAVE and AIFF file formats.
- OS X: for all programs the minimum OS version supported is now 10.5 (Leopard). The programs are Intel only.
- All programs read and write the new **PEAK chunk**, and in many cases this information is displayed on the console.
- All programs can recognize a Windows 'shortcut' to a soundfile, so long as it is in the form "soundfile.wav.lnk" - i.e the WAV or AIF extension is there. Shortcuts to CDP soundfiles can be created with the System Utility program, **ALIAS.**
- Sources for the programs (including Linux) are now available as part of the **CDP source release** under the LGPL licence. (No binary downloads are available for Linux.)

Surround Sound

Interest in **'surround sound'** is growing rapidly, and composers using computer-based digital equipment are now extending their use of PAN into the various types of 'surround': adding 'depth' in the horizontal plane, truly vertical and fully spherical dimensions, ideally in a way which is interactive in the performance situation.

How you organise your loudspeaker layout is up to you. For the beginner, matching channels to speakers can be quite confusing. The MULTI-CHANNEL TOOLKIT supports a wide variety of speaker layouts (see **COPYSFX** and **INTERLX**; **PAPLAY** and **FMDCODE**), which include the following:



However, many of the CDP processes in the **MULTICHANNEL group** tend to assume a clockwise ring numbering (not directly supported by the TOOLKIT):

For these programs, if you require a different layout, the TOOLKIT's **CHORDER** can be used to convert the channel numbering.

One simple solution is to use "generic" WAVE_EX files, in which channels are matched to no specific speaker layout, but are sent to the soundcard in ascending order. You can then hook up the channel outputs to your favoured speaker layout. Alternatively, you might use a popular speaker layout, such as 5.1 or 7.1 cinema surround, and write WAVE_EX files that support this layout. A third option is to experiment with Ambisonics.

Ambisonic B-Format

Unlike other multi-channel surround formats, Ambisonics' transmission channels do not carry speaker signals. Instead, they contain a speaker-independent representation of a sound field called **B-format**, which is then decoded to the listener's speaker setup (Wikipedia).

Ambisonics can be encoded within either .WAV or .AIFF formats; however, there was initially no way of distinguishing between Ambisonic B-format files and plain multi-channel files. For this reason, CDP's Richard Dobson has defined a simple B-Format specialization of WAVE_EX that supports up to third-order B-Format streams (i.e. up to sixteen channels). It is now widely known and used as the **AMB** format. The CDP MULTI-CHANNEL TOOLKIT includes support for AMB files. A B-Format stream can be decoded to a wide range of speaker layouts (using **FMDCODE**), including the ubiquitous but otherwise unambitious 5.1 cinema surround arangement.

For more information on Ambisonics, see the **Wikipedia article** on the subject.
Read more about the AMB format in Richard's article **The AMB Ambisonic File Format**.

ABFPAN – Apply a fixed or orbiting 1st order B-Format pan to mono soundfile

Usage

abfpan [**-b**] [**-x**] [**-o***N*] *infile outfile startpos endpos*

Parameters

infile – standard **mono** .wav (any of the sample formats listed above)
outfile – Default: create a standard 4-channel **.wav** soundfile. Or use **-o** flag to write B-Format.
startpos – rotation start position (Range: greater than 0.0 and less than 1.0. 0.0 and 1.0 = Centre Front)
endpos – determines the nature of the rotation:

- < 0.0 gives anticlockwise rotation (negative numbers)
- > 0.0 gives clockwise rotation
- units give number of revolutions
- fractions give final position
- set *endpos* = *startpos* for a fixed pan

-b – write output as horizontal Ambisonic B-Format (**.amb**): the standard format
-x – write B-Format (WAVE_EX) format file. This requires an **.amb** or a **.wav** file extension.
Default: write a standard multi-channel file (WAVE, AIFF).

Note that ABFPAN requires the **-x** flag to create a WAVE_EX file. This flag should always be used when writing a **.wav** format file. Use the **-b** flag to write the B-Format channels. *Both flags are needed to create an **AMB** format file.* Such a file can then be decoded using **FMDCODE** to a choice of speaker layouts.

-o*N* – the number of B-Format output channels: *N* = 3 or 4 only.
(Default: write a standard 4-channel .wav file.)

About ABFPAN

ABFPAN generates a simple periodic rotating pan around the listener, using 1st-order B-Format (horizontal) encoding. The *startpos* and *endpos* values determine the number of revolutions performed over the duration of the file.

- Identical values will result in a fixed radial position.
- Integer positions define Centre Front.
- A negative *endpos* value generates an anti-clockwise pan.
- Pans a sound ambisonically in a circle round the listener, between given start and end positions.
- Multiple rotations are possible.
- Output is either a standard (decoded) quad soundfile, or Horizontal B-Format (standard WAVE file).
- The latter can be converted to the WAVE_EX format by **COPYSFX**.

End of ABFPAN

ABFPAN2 – Apply a fixed or orbiting 2nd order B-Format pan to mono soundfile

Usage

abfpan2 [**-g***gain*] [**-w**] [**-p**[*deg*]] *infile outfile startpos endpos*

Parameters

infile – standard **mono** .wav (any of the sample formats listed above)
outfile – a 2nd-order B-Format soundfile (use **.amb** extension). Note that the default .wav output is in WAVE_EX format. Or use the **-w** flag to write a plain WAVE format file.
startpos – rotation start position (Range: greater than 0.0 and less than 1.0. Both 0.0 and 1.0 = Centre Front: i.e., they are the same place.)
endpos – determines the nature of the rotation:

- < 0.0 gives anticlockwise rotation (negative numbers)
- > 0.0 gives clockwise rotation
- units give number of revolutions
- fractions give final position
- set *endpos* = *startpos* for a fixed pan

-g*gain* – scale *infile* amplitude by a factor of *gain* amount. *Gain* must be > 0.
-w – write standard (plain) WAVE format multi-channel soundfile (**.wav** / **.aiff**).
Default: write WAVE_EX B-Format. The plain WAVE format file may be needed by some legacy applications.
-p[*deg*] – write a full 9-channel (periphonic) B-Format soundfile.
Default: write a 5-channel (2nd-order horizontal) file.
deg sets an optional fixed height argument, in degrees. Range: -180 to +180, where -90 = nadir, +90 = zenith (directly above). Default: *deg* = 0, with which the height channels (Z, R, S, T) will be empty.

About ABFPAN2

This new extended version of ABFPAN employs 2nd-order B-Format encoding, with optional fixed elevation. Output is in **AMB** format, and there is no decoded output option. A horizontal-only output comprises five channels, and one with height ('periphonic') comprises 9 channels.

ABFPAN2, like ABFPAN, generates a simple periodic rotating pan around the listener, but this time using 2nd-order B-Format encoding. The pan arguments (*startpos* and *endpos*) are the same as in ABFPAN. The *startpos* and *endpos* values determine the number of revolutions performed over the duration of the file.

- Identical values will result in a fixed radial position.
- Integer positions define Centre Front.
- A negative *endpos* value generates an anti-clockwise pan.
- Pans a sound ambisonically in a circle round the listener, between given start and end positions.
- Multiple rotations are possible.
- Output is either a 2nd-order B-Format multi-channel soundfile, or a standard (decoded) quad soundfile (**-w** flag).
- The latter can be converted to the WAVE_EX format by COPYSFX.

End of ABFPAN2

CHANNELX – Extract all or selected channels from a multi-channel soundfile

Usage

channelx [**-o***basename*] *infile chan_no* [*chan_no...*]

Parameters

-o*basename* – base name (with extension) of outfiles (appended *_cN for ch N)

NB: the channels of WAVE_EX files are written as standard soundfiles.

infile – input multi-channel soundfile
chan_no – channel number(s) of channels to extract

List a single integer to extract one channel, or several integers separated by spaces to extract several channels, up to the number of channels in the file.

About CHANNELX

- CHANNELX extracts all or selected channels of a multi-channel soundfile file to separate output mono soundfiles.
- The channels are counted from 1.
- By default the output soundfiles are named as *infile_c1*, *infile_c2* etc., with the source file extension.
- Alternatively, you can use the **-o** flag to define a custom name.

ALSO SEE: **HOUSEKEEP CHANS 2** – extracts all channels from a soundfile.

End of CHANNELX

CHORDER – Re-order soundfile channels in a multi-channel soundfile

Usage

chorder *infile outfile orderstring*

Parameters

infile – input multi-channel soundfile

Infile channels are mapped in order as a = 1, b = 2 ... z = 26.
For example, the channels in a 4-channel file are represented by the characters **abcd**.
Any other character is an error.

outfile – output multi-channel soundfile with re-ordered channels.

- The **.amb** extension is supported for the outfile.
- Note that the *infile* (WAVE_EX) speaker positions are discarded.

orderstring – the order string is made up of any combination of characters, a to z inclusive, e.g. **dcba**

- Characters are confined to those of the input channels and MUST be lower case.
- Characters may be used more than once. Duplicate characters duplicate the corresponding input channel.
- The zero character ('0') may be used to set a silent channel.
- A maximum of 26 channels (a-z) is supported for both input and output.

About CHORDER

CHORDER expedites many of the tasks previously requiring a combination of CHANNELX (extract channels) and INTERLX (interleave channels). Channels may be re-ordered while retaining the number of them. The program also supports both extracting a subset of input channels (e.g., 1st-order channels from a 2nd or 3rd order file), and augmenting the channel count by duplicating input channels and/or defining a number of empty (silent) channels. All **.wav** output files are in WAVE_EX format – the speaker positions (and/or the GUID) can be changed using **CHXFORMAT**:

Example 1: Convert anti-clockwise quad layout to standard surround quad. The command lines are:

```
chorder inregular.wav outquad.wav abcd
chxformat -s0x33 outquad.wav
```

Example 2: Convert 5.0 surround file to 5.1 (with silent LFE channel). The command lines are:

```
chorder in50.wav out51.wav abc0de
chxformat -s0x3f out51.wav
```

ALSO SEE: FMDCODE and **CHXFORMAT**.

End of CHORDER

CHXFORMAT – Modify WAVE_EX header to change GUID and/or speaker positions

WARNING: this program is destructive - it modifies the header of the infile.
File system errors are always a danger. Do not use with files that cannot be replaced - use **COPYSFX** instead. It can also be used safely in test mode to report relevant header properties. In this release this program is marked as beta. While extensively tested (it employs full header parsing, it does not take any "short cuts"), as a destructive program it comes with no guarantees, nor warranty.
Note that only the file header is modified; the audio data itself is untouched.

Usage

chxformat [**-m**] | [[**-t**] [**-g***GUID*] [**-s***mask*]] *filename*

Parameters

filename – input multi-channel soundfile in the WAVE_EX format
-m – display list of WAVEX mask values. This option is not to be used in combination with other options.
-t – test only: do not modify file.

- If only *filename* is given (the infile), the program prints the current GUID type and speaker mask.
- In test mode, the program checks *filename*, prints the current channel mask as binary, and, if the **-s** option is set, the new mask in binary.

-g*GUID* – change GUID type between PCM and AMB.

- *GUID* = **1** is for Plain WAVEX
- *GUID* = **2** is for AMB

-s*mask* – change speaker position mask.

- *mask* = **0** unsets the channel mask, otherwise set to *mask*.
- *mask* may be given in decimal or hexadecimal (prefix '0x') – note examples given in **CHORDER** section.
- Note that speaker positions are only supported for WAVEX PCM files.
- If *GUID* is set to **2**, the **-s** option should not be used.
- Any existing speaker positions will be set to zero.
- Type `chxformat -m` at the console to see a list of WAVEX *mask* values.

About CHXFORMAT

CHXFORMAT operates exclusively on files in the WAVE_EX format. Some familiarity with this format is advisable before using this program.

The **-g** option can be used to change a file from AMB format to **.wav** or *vice versa*, e.g., to be able to load a B-Format file into an editor or DAW that does not know about the AMB format. The program automatically assigns the appropriate GUID according to whether the samples are integer or floating-point.

The **-s** option is used the change the contents of the WAVE_EX speaker position flags, either to zero (generic WAVE_EX) or to some other layout. The flag value can be entered in either decimal or hexadecimal form.

Use the **-m** option on its own to see the list of available positions, together with examples of the most common layouts.

Use the **-t** option to see the current speaker layout of the file, and to verify the chosen speaker layout. The layout value is printed in binary, to facilitate checking against the position list.

Finally, use CHXFORMAT with just a file name to see the relevant properties of *filename*. For a comprehensive general format report, use SFPROPS.

ALSO SEE: examples of use with CHORDER.

End of CHXFORMAT

COPYSFX – Copy/convert soundfiles

Usage

copysfx [**-d**] [**-h***N*] [**-s***N*] [**-t***N*] *infile outfile*

Parameters

infile – a **wav** or **aif** soundfile
outfile – a **wav** or **aif** soundfile.
Default in all cases: *outfile* has the format of the *infile*.

-d – apply TPDF (2-LSB triangular) dither to a 16-bit *outfile*
-h*N* – write minimum header:

- **0** = minimum (no extra data: this avoids incompatibility with software which cannot read the extra data)
- **1** = PEAK data only
- Default (flag not used): include PEAK data.

-s*N* – force output sample type to type *N*

Available sample types:

1. 16-bit integers (shorts)
2. 32-bit integer (longs)
3. 32-bit floating-point
4. 24-bit integer 'packed'

Default (flag not used): format of *infile*

-t*N* – write *outfile* format as type *N*

Possible formats:

0. – (default) standard soundfile (**.wav**, **.aif**, **.afc**, **.aifc**)
1. – generic WAVE_EX (no speaker assignments)
2. – WAVE_EX mono/stereo/quad (LF, RF, LR, RR) – *infile* number of channels must match
3. – WAVE_EX quad surround (L, C, R, S) – *infile* must be quad
4. – WAVE_EX 5.1 format surround – *infile* must be 6-channel
5. – WAVE_EX Ambisonic B-Format (W, X, Y, A ...) – the **.amb** extension is supported
6. – WAVE_EX 5.0 surround – *infile* must be 5-channel
7. – WAVE_EX 7.1 surround – *infile* must be 8-channel
8. – WAVE_EX cube surround – *infile* must be 8-channel
9. – WAVE_EX 6.1 Surround – *infile* must be 7-channel

NB: Types 1 to 8 are for WAV format only.

About COPYSFX

COPYSFX is the primary workhorse program in the TOOLKIT for copying and converting files from one format to another.

By default COPYSFX writes a PEAK chunk to the output soundfile. This stores the position and value of the first absolute maximum sample in each channel, together with a timestamp. Almost all the TOOLKIT programs print the PEAK data to the screen on completion. It can also be shown via **SFPROPS**. For integer formats, the value will of necessity be clipped to 1.0. For floating-point formats the value reflects the true peak values in the file.

It is still common for many applications to assume a fixed length header, and be broken by any file with additional chunks such as the PEAK data. For such applications, use the **-h** flag to write a "minimal" header with no extra chunks.

Note that while **CHXFORMAT** can save considerable disk space and time by changing the header of a file directly, it is inherently risky. COPYSFX is non-destructive and, if space and time permit, is still the recommended tool for format conversion. CHXFORMAT may however be required where it is desired to set an unusual or experimental speaker mask not supported by **COPYSFX**.

ALSO SEE further explanations in the **System Utilities** Manual.

End of COPYSFX

FMDCODE – Decode 1st or 2nd order B-Format soundfile to a choice of speaker layouts

Usage

fmdcode [**-x**] [**-w**] *infile outfile layout*

Parameters

infile – input 1st or 2nd order B-Format soundfile in WAVE_EX format (including **.amb**).
outfile – appropriately decoded multi-channel soundfile in one of the WAVE formats

-w – write plain WAVE outfile format.
Default **.wav** output format is generic WAVE_EX (channels delivered to soundcard in ascending order).
-x – write standard WAVE_EX speaker positions to the header. This applies to compatible layouts only and requires the **.wav** extension.

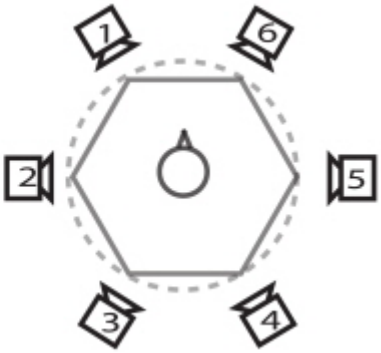
layout – one of the choices below.
The output channel order is **anticlockwise** from Centre Front except where indicated.
Layouts marked with an asterisk (*) are compatible with WAVE_EX speaker position order.
Available speaker layouts:

1. – * mono (= W signal only)
2. – * stereo (quasi mid/side, = W +- Y)
3. – square
4. – * quad (FL, FR, RL, RR order)
5. – pentagon
6. – * 5.0 surround (WAVE_EX order)
7. – * 5.1 surround (WAVE_EX order, silent LFE)
8. – hexagon
9. – octagon 1 (front pair, 45°)
10. – octagon 2 (front centre speaker)
11. – cube (as **3**, low-high interleaved, *Csound* compatible)
12. – cube (as **4**, low quad followed by high quad)

About FMDCODE

With the exception of layouts **6** and **7**, this program implements the widely used Furse-Malham decoding coefficients for 1st and 2nd order B-Format. These coefficients are also employed in the *bformenc1* and *bformdec1* opcodes for *Csound*. Layout option **11** (cube) corresponds to the one employed by *bformdec1*, while layout option **12** is (approximately) WAVE_EX-compatible.

In all cases (as in *Csound*) the 'controlled opposites' (also known as 'in-phase') decoding solutions are employed, without either shelf filters or distance compensation. These features may appear in a future release. These solutions apply to *regular* geometric layouts – that is, where speakers are evenly distributed around the circle, as in the diagrams below:



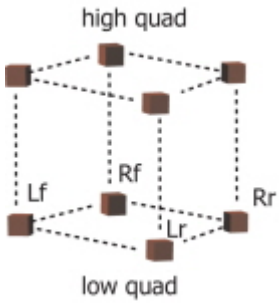
Type 3: square **Type 5: pentagon** **Type 8: hexagon**

In options **9** and **10**, two alternative octagonal layouts are defined, depending on the use of either one speaker at centre front, or two left and right of centre front, as in the diagram below:

Type 9: octagon 1 **Type 10: octagon 2**

As the figures show, in all these cases the *outfile* channels (speaker feeds) are ordered ***anti-clockwise*** relative to centre front. The opposite order is also widely demonstrated in the literature. Should this order be required, the program **CHORDER** can be used to reverse the sequence.

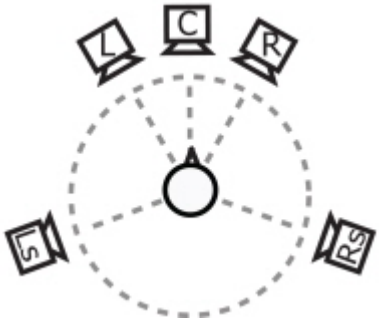
The principle of B-Format encoding and reproduction is predicated on such regular layouts, which extend to the periphonic case ('with-height' reproduction) using shapes such as the simple cube and the dodecahedron. The MULTI-CHANNEL TOOLKIT does not currently address layouts involving more than eight speakers. A diagram for a basic cube layout is given below.



Typical cube periphonic layout

About WAVE_EX-compatible speaker layouts

As the usage message indicates, the majority of these regular layouts are not compatible with the WAVE_EX speaker positions. Despite the fact that they are described by name rather than by precise co-ordinates, their positions are in general *assumed*, e.g., by reference to standard irregular layouts such as the ITU 5.1 surround layout as shown below (sub-woofer omitted):



ITU 5.1 surround layout

The order of the speaker position flags in WAVE_EX is based on alternating left/right pairs (e.g., layout **4** – rectangular quad). We can use the applicable WAVE_EX positions to represent this layout only with the understanding that the front pair are spaced to subtend *not* the conventional stereo 30° angles left and right of centre front, *but* 45° angles, such that the four speakers form an exact square.

With the same understanding, the cube layout type **12** can also be presented in WAVE_EX, as that includes support for 'TOP' positions matching the usual 'LOW' positions. However, even here we must take care: the purely horizontal layouts naturally address speakers arranged at head height. In contrast (and as with all such layouts), the B-Format cube layout places the listener in the *centre* of the cube, so that speakers are equidistantly *below* and *above* the listener. To achieve such a layout in a practical context, these low speakers are typically placed at floor level.

As WAVE_EX does not define much in the way of regular layouts, by default the program sets the speaker position flags to zero. This signifies a 'generic' layout, in which the channels are delivered to the soundcard in simple ascending order. It is then up to the user to route those outputs appropriately. One advantage of this approach is that it bypasses attempts by the operating system to re-map the signals to whatever layout it thinks you have! To write such layouts with the appropriate WAVE_EX speaker mask, use the **-x** flag. Advanced users can use CHXFORMAT to experiment with speaker masks for the other regular layouts. Attempting to play such files directly using standard Windows players such as *Media Player* is very likely to give unintended results. See the description of **PAPLAY** for more information regarding this issue.

About 5.1 layouts, types 6 and 7

As noted above, these layouts are not derived from the Furse-Malham specifications. The coefficients for the 5.0 and 5.1 layouts (using the so-called 'velocity' decode option) have been kindly supplied by David Moore as a result of recent research employing advanced High-Performance Audio Computing (HiPAC) techniques. They are a result of a research project employing advanced high-performance computing techniques (HPC) to find optimal coefficients through the use of exhaustive search algorithms.

For B-Format decoding, the LFE (low-frequency effects) channel (the '1' of '5.1') is not used. Accordingly, two decode options are provided, supporting five- and six-channel files. The latter thus includes an empty LFE channel. When using the WAVE_EX speaker option (**-x** flag) you may need to use the six-channel option to ensure your OS (and hardware) can render the file correctly.

Where possible, a second-order B-Format source should be employed. This is generally held to decode better to the ITU layout than first-order. For casual exploration of B-Format surround, this layout appeals as it corresponds to the expected speaker layouts for home cinema. Note however that for best results the speakers should be identical (or at least very closely matched) and full-range. For a simple test, **ABFPAN2** can be used to pan a sound around a circle, using 2nd-order encoding.

End of FMDCODE

INTERLX – Interleave mono or stereo files to make a multi-channel file

This is the basic program for making multi-channel files.

Usage

interlx [**-t***N*] *outfile infile1 infile2* [*infile3 ... infile16*]

Parameters

outfile – the output format is taken from *infile1*
infile1 infile2 – input mono or stereo soundfiles to interleave. Files must match in sample rate and in number of channels, but may have different sample types.
infile 3 ... infile16 – additional input soundfiles to interleave (up to 16, max.)

-t*N* – write *outfile* format as type *N*
Available speaker layouts:

0. – (default) standard soundfile (**.wav**, **.aif**, **.afc**, **.aifc**)
1. – generic WAVE_EX (no speaker assignments)
2. – WAVE_EX mono/stereo/quad (LF, RF, LR, RR) – *infile* number of channels must match
3. – WAVE_EX quad surround (L, C, R, S) – *infile* must be quad
4. – WAVE_EX 5.1 format surround – *infile* must be 6-channel
5. – WAVE_EX Ambisonic B-Format (W, X, Y, A ...) – the **.amb** extension is supported
6. – WAVE_EX 5.0 surround – *infile* must be 5-channel
7. – WAVE_EX 7.1 surround – *infile* must be 8-channel
8. – WAVE_EX cube surround – *infile* must be 8-channel
9. – WAVE_EX 6.1 Surround – *infile* must be 7-channel
   (B-Format decoding is not yet available for this layout.)

NB: Types **1** to **8** are for WAV format only.
In all cases the *outfile* has the sample format of *infile 1*.

Please also note:

- The same *infile* can be listed multiple times, for example, to write a mono file as stereo, quad, etc.
- To create a silent channel, for *infile2* onwards, use 0 (zero) as the filename. *Infile1* must be a soundfile.
- Speaker-positions in WAVE_EX *infiles* are ignored.
- The **.amb** B-Format extension is supported. The program will issue a warning if the channel count is anomalous. Known B-Format channel counts are: 3, 4, 5, 6, 7, 8, 9, 11, 16.

About INTERLX

This program complements **CHANNELX**, and is the primary workhorse for assembling multi-channel files from multiple input files. As the usage message shows, it serves two key tasks: to create WAVE_EX files in particular formats, and to create AMB B-Format files from mono input files comprising individual B-Format signals. For the latter task, use `-t1` to make a generic WAVE_EX file – no speaker positions are defined for the AMB format.

End of INTERLX

NJOIN – Concatenate multiple soundfiles into a single file, with optional CUE list output for CD burning

Usage

njoin [**-s**secs | **-S***secs*] [**-c***cuefile*] [**-x**] *filelist.txt* [*outfile*]

Parameters

filelist.txt – text file containing a list of soundfiles in order, one complete file path per line.

- Files must match in sample rate and in number of channels, but can have different sample types.
- Channel spec (if present) must be the same, but files with no spec are assumed to be compatible.
- In OS X, use of the ~/ notation for the user's home directory is supported in *filelist*.

outfile – optional output soundfile of joined up (multi-channel) soundfiles. The output sample format is taken from the soundfile with the highest precision. If no *outfile* is specified, the program merely scans the files and prints a report to the console.

-s*secs* – separate files with silence of *secs* seconds
OR
-S*secs* – as above, but with no silence before the first soundfile.

Default: soundfiles are joined with no gap.

-c*cuefile* – if *outfile* is used, generate a cue textfile as the *cuefile*
-x – strict: allow only CD-compatible files: files must be stereo, use a sample rate of 44100 and have a minimum duration of 4 seconds.

About NJOIN

This utility program stands somewhat apart from the others in that it is only indirectly concerned with multi-channel files. Its primary purpose is simply to concatenate multiple files into one output file. Channel counts and sample rates must match. Secondarily the program supports preparation for CD-R burning by means of basic support for the standard CUE list file, used by a number of CD burning applications. Files can be separated (prefixed) by a defined amount of silence.

The program exits with an error message if the total duration exceeds the 4 Gigabyte capacity of the file format. When used without an *outfile* argument, the program computes the total duration and reports to the console.

End of NJOIN

NMIX – simple mix of two multi-channel files, with optional offset

Usage

nmix [**-d**] [**-f**] [**-o***offset*] *infile1 infile2 outfile*

Parameters

infile1 infile2 – input soundfiles to mix

- Files must have the same channel count, but can have different sample rates.
- WAVE_EX files must have the same speaker layout.

-d – apply TPDF dither (16-bit format output only)
-f – set the output sample type to 32-bit floating-point ('floats').
Default: *outfile* type is that of *infile1*.
-o *offset* – start *infile2* at *offset* seconds

About NMIX

There is nothing much to say about this program: it mixes the two named soundfiles together. The channel counts (and speaker layouts, if any) must match. Note that it does not use a CDP *mixfile* as input.

ALSO SEE: SUBMIX MERGE

End of NMIX

PAPLAY – Playback of multi-channel soundfiles

Usage

paplay [**-B***N*] [**-d***N*] [**-g***N*] [**-h***N*] [**-i**] [**-l**] [**-b**[*N*] | **-m**[*S*]] [**-u**] [**-x**] *soundfile* [*from*] [*to*]

Parameters

soundfile – handles a variety of standard and multi-channel soundfiles.
NB: You need to supply the extension explicitly with this program.
from – optional start-time in seconds (Default = 0).
to – optional end-time in seconds (Default= end-of-file)
Enables arbitrary block to be played and looped.

-d*N* – use output Device *N*
-g*N* – apply gain factor *N* to input
-B*N* – set memory buffer size to N frames (default: 32768)
N must be a power of 2 (e.g 4096, 8192, 16384 etc).
-h*N* – set hardware blocksize to N frames (32 < N <= BN/4). Default: from soundcard.
N is recommended to be a power of two size.
Where set, buffer sizes are doubled internally for sample rates >48KHz.
-i – play immediately (do not wait for keypress)
-l – loop file continuously, from start-time to end-time
-m[*S*] – render using channel map string S. (Use -m without parameter for usage.)
S (order string) = any combination of characters a-z inclusive.
No. of characters in S sets no. of output channels – must be supported by selected device.
Further details – see **CHORDER** *orderstring*
NB: -m cannot be combined with -b (B-Format decoding).
If channel mapping is used, a B-format file (AMB) will be rendered without decoding.
-u – suppress elapsed time updates
-x – apply WAVE_EX infile channel mask to Direct Sound audio stream (ignored if -m or -b used)
-b*N* – apply 1st-order B-Format decoding to a standard soundfile, which must have at least 3 channels.
B-Format (**.amb**) files will be decoded automatically.
The output channel order is anticlockwise from Centre Front except where indicated.
Layouts marked with an asterisk (*) are compatible with WAVE_EX speaker position order.
***N* sets the speaker layout to one of the following:**

1. – * mono (= W signal only)
2. – * stereo (quasi mid/side, = W +- Y)
3. – square
4. – * quad (FL, FR, RL, RR order) – default
5. – pentagon
6. – * 5.0 surround (WAVE_EX order)
7. – * 5.1 surround (WAVE_EX order, silent LFE)
8. – hexagon
9. – octagon 1 (front pair, 45°)
10. – octagon 2 (front centre speaker)
11. – cube (as **3**, low-high interleaved, *Csound* compatible)

12. – * cube (as **4**, low quad followed by high quad) NB: no decoding is performed if the -m flag is used. Please also note:

- On Windows, ASIO as well as DirectSound is supported.
- OS X version supports Jack as well as Core Audio.
- Windows only: use the **-x** flag option to send WAVE_EX speaker positions (for starred layouts) to the hardware. The default is to send a zero mask, so that channels map directly to successive hardware outputs.
- PAPLAY reads the PEAK chunk if present to rescale over-range floating-point ('floatsam') files.
- Type PAPLAY without arguments to display the usage message followed by a list of the available audio devices. Note that input and output devices are listed separately, so that the list can become quite long! The current default output is indicated by a leading asterisk before the device number (see example below).
- For ASIO multi-channel, you may need to select the highest device number:

Device	Input	Output	Name
0	2	0	Primary Sound Capture Driver
1	2	0	Microphone (Realtek High Definition Audio)
2	2	0	Monitor (M-Audio Delta Audiophile)
3	2	0	Line 1/2 (M-Audio Delta Audiophile)
4	2	0	S/PDIF (M-Audio Delta Audiophile)
*5	0	2	Primary Sound Driver
6	0	2	Line 1/2 (M-Audio Delta Audiophile)
7	0	6	Speakers (Realtek High Definition Audio)
8	0	2	S/PDIF (M-Audio Delta Audiophile)
9	0	2	Realtek Digital Output (Realtek High Definition Audio)
10	6	4	M-Audio Delta ASIO

About PAPLAY

The same B-Format decoding is employed here as in **FMDCODE**. As in that program, no shelf filters or nearfield compensation are employed.

The **-i** flag is provided primarily to enable the program to be invoked from a scripting language (e.g., TCL/Tk, MatLab, Octave, Tabula Vigilans, etc.) or from batch files, where it is inconvenient to interact with the program via the keyboard. Otherwise, note that when launched, the program *waits for a keypress* to commence playback. As the first block of data is pre-loaded, playback will effectively start immediately.

Testing audio routing

It is strongly recommended that users test their routing before trying to play a multi-channel file. The MULTI-CHANNEL TOOLKIT can itself be used to prepare suitable test materials.

One approach is to create a special file (e.g., of spoken idents, or clearly distinguishable tones), with each channel clearly identified in sequence. The **-l** flag can be used to loop this or any file until stopped via CTRL-C, while any necessary adjustments to routing or levels are made. For B-Format playback, it is especially important that all speakers are as closely matched in level and distance as possible, and that any mixer or other level controls are ganged together.

Such a file is especially easy to create using *Audacity*: record each ident in sequence as a new track. Then use **Export** with **Advanced Mixing Options** enabled, to save each track as a channel of a multi-channel soundfile. You can use **COPYSFX** to convert this file into the required WAVE_EX format. Alternatively, use the **Export Multiple** option to save each track as a separate numbered file, and create the final multi-channel file using **INTERLX**. This approach is indicated, for example, when the same idents can be used, in different combinations, for a variety of multi-channel layouts.

To check that speaker levels are exactly matched, a file with test signals is required, with (ideally) a loudness meter at the listening position. This is especially important if unmatched speakers (and amplifiers) are employed!

OS X

It goes without saying that PAPLAY uses CoreAudio. OS X itself recognises both WAVE_EX files and the most standard speaker layouts and renders them (e.g., via *Quicktime*) appropriately, given multi-channel hardware. PAPLAY simply sends the channels in order to the chosen device.

The historical limitations (and bugs) with respect to the WAVE formats associated with earlier Macintosh systems are long gone, and with the move to Intel processors the case for using WAVE (in general) over the AIFF formats is even stronger. The AIFF format is now arguably all but obsolete – such definitions as exist in the AIFF and AIFFC specifications for multi-channel files are variously obsolete and ambiguous. Apple's replacement for the AIFF family is the new CAF file format. Support for this may appear in later updates to the MULTI-CHANNEL TOOLKIT.

Windows

PAPLAY supports both the DirectSound and the ASIO APIs. Accordingly, the displayed list of devices will be longer than before, with the same hardware identified for each API as appropriate. In Windows, it is not necessary to set a device number if you wish to play the file on the default audio playback device set in Control Panel.

Note the **-x** flag, exclusive to the Windows platform. Windows recognises a very limited set of standard speaker layouts. These can be selected (in this example, in XP) via the **Advanced** button on the Audio page of **Sound and Audio device Properties**. To play a multi-channel WAVE_EX file using **Media Player** or other Windows-standard player, users will need to check which layout is selected. If there is any mismatch, such as trying to play a 5.1 file to stereo speakers, the Windows kernel mixer will attempt to map the audio channels to that layout in some representative way. This has some virtues, such as being able to audition an arbitrary multi-channel file over stereo speakers. Many commercial editors and DAWs perform in much the same way, sending the WAVE_EX channel mask directly to the subsystem. Should the user want to replicate this behaviour using PAPLAY, the **-x** flag can be used. This flag is not available in the OS X version of PAPLAY.

Note, however, that when Windows plays, for example, a quad soundfile to a stereo device, the rear channels are typically rendered at a lower level.

For such tasks as B-Format decoding to relatively unorthodox layouts such as hexagon or octagon, this automatic remapping is not useful beyound a simple integrity check. The default behaviour of PAPLAY is to send a 'generic' mask value of zero to the audio subsystem, so that channels are simply mapped to successive device outputs.

End of PAPLAY

RMSINFO – Scan file and report RMS and average power level statistics

Usage

rmsinfo [**-n**] *infile* [*startpos* [*endpos*]]

Parameters

-n – include equivalent 0dBFS-normalised RMS and AVG levels.
Use this flag to see the equivalent levels if the file is normalised to 0dBFS.
infile – handles a variety of standard and multi-channel soundfiles
startpos – start position in seconds at which to begin the scan
endpos – position in seconds at which to end the scan

To stop a scan early, use CTRL-C. The program will report levels up to that point.

About RMSINFO

Standard output shows:

- RMS level
- Average level
- DC level (bipolar average)

RMSINFO complements **SFPROPS** by determining the overall power level (loudness) of each channel of the input soundfile. It uses two types of computation: RMS (Root-Mean-Square) and raw average (normalised sum of all samples). In addition, it displays the average level of any DC present. All levels are calculated relative to the digital peak at 0dBFS. Thus a full-range sinusoid will have a reported RMS level of -3dB.

For sustained sounds, the RMS and average levels will be very similar. For percussive, speech-based, and other sounds with widely differing amplitudes, they may be markedly different – the average level will be lower than the RMS level. In such cases, the information reported gives an overall indication of the 'peakiness' of a file, and may be a guide to the choice of compressor thresholds.

Note that for *static* B-Format soundfields – i.e., no motion of sources – the information will demonstrate the standard -3dB reduction of the W signal (as employed in all the TOOLKIT programs and in keeping with the Furse-Malham conventions).

For *mobile* sources, e.g., from using **ABFPAN2**, the levels may well be reported as equal, depending on the overall bias (if any) of the panning. The final reported value represents the average over the whole file.

Other conventions for B-Format streams are under consideration within the surround sound community, in particular, normalisation schemes that avoid the traditional -3dB reduction of W. This will in principle lead to RMS values for W that are closely similar to those of other channels.

The program recognises that it can take a long time to scan a large file. The optional arguments *startpos* and *endpos* (which can be invoked separately) can be used to select a selection of a file to be scanned. Also, CTRL-C can be used to terminate the program, in which event the levels *up to that point* will be displayed.

End of RMSINFO

SFPROPS – Display soundfile details, with WAVE_EX speaker positions

Usage

sfprops *infile*

Parameters

infile – handles a variety of standard and multi-channel soundfiles

About SFPROPS

This is a purely informative program: in addition to reporting the standard properties of a soundfile, it gives details of any WAVE_EX speaker positions and identifies the layout when it is one of the standard ones supported by other MULTI-CHANNEL TOOLKIT programs. It prints all PEAK information if present. Use CHXFORMAT to find low-level information on the WAVE_EX speaker mask.

Note that it will identify 7.1 and cube speaker layouts. The PEAK chunk report includes a level report in dB. SFPROPS displays the soundfile properties on the console: duration, number of sample frames, sample rate, etc.

ALSO SEE: SNINDFO PROPS.

End of SFPROPS

Documentation: Richard Dobson, revised R. Fraser