

BeSweet Commandline Reference

Document version 2005-03-28

Based on German version 2005-03-28

BeSweet version 1.5b29

Table of Contents

1 Introduction	2
2 BeSweet's Syntax	2
3 Global BeSweet Configuration	3
3.1 The <code>-core()</code> Section	3
3.1.1 Basic Parameters	3
3.1.2 Switches for VOB Input	5
3.1.3 Destination Format Switches	5
3.1.4 Tweaking Switches	6
3.2 The <code>-split()</code> Section	7
3.3 The <code>-plugin()</code> Section	7
4 Decoder and Sound Manipulation	8
4.1 The <code>-ota()</code> Section	8
4.1.1 Normalising Switches	8
4.1.2 Delay Switches	9
4.1.3 Switches for Sampling Frequency and Framerate	9
4.2 The <code>-soundtouch()</code> Section	10
4.3 The Sections <code>-shibatch()</code> and <code>-ssrc()</code>	11
4.4 The <code>-boost()</code> Section	11
4.5 The <code>-azid()</code> Section	12
4.5.1 Normalisation and Dynamic Compression	12
4.5.2 Downmix Switches	13
4.5.3 Switches for Controlling Individual Channels	15
5 Encoders	16
5.1 The <code>-ogg()</code> Section	16
5.2 The <code>-bsn()</code> Section	17
5.3 The <code>-lame()</code> Section	19
5.4 The Section <code>-mp2enc()</code> or <code>-toolame()</code>	20
5.5 The <code>-ac3enc()</code> Section	21
6 Credits	21

1 Introduction

This document contains an overview of all of BeSweet's commandline parameters and their interaction. It is a translation of the German "BeSweet-Kommandozeilenreferenz", which I started because of the lack of good German documentation for BeSweet. But even in English there is no single document summarising BeSweet's abilities. That is why I decided to do a translation.

The Commandline Reference is based on BeSweet version 1.5 beta 29 available from DSPguru's homepage <http://besweet.notrace.dk>. To avoid problems with missing files you should also download the stable version 1.4 and install that one first.

Found an error? Got a suggestion? Contact me via PM on Doom9's forum. Please don't ask me things about using BeSweet. The audio section of Doom9's forum is a better place for that kind of questions. Make sure to read the BeSweet FAQ before posting: <http://forum.doom9.org/showthread.php?s=&threadid=15738>

Brother John

2 BeSweet's Syntax

Syntax

```
BeSweet.exe -core() [-split()] [-plugin()] [-ota()] [-soundtouch()]
[-shibatch()|-ssrc()] [-boost()] [-azid()] [-ogg()|-bsn()|-lame()|
-mp2enc()|-toolame()|-ac3enc()]
```

Sections

Only the `-core()` section is mandatory. All the others may be omitted or combined in almost any way, except for the encoder sections. Only one of those may be used.

Here is the shortest possible BeSweet commandline:

```
BeSweet.exe -core( -input <source> -output <destination> )
```

BeSweet will create a 128 kbit/s CBR stereo MP3.

Spaces

Sections are separated with a space each. Additionally a space must be there after every opening bracket and before every closing bracket. Switches inside a section must be separated by a space, too. An example (red dots • mean spaces):

```
-core(•-input•source.ac3•-output•dest.wav•-2ch•)-ota(•-d•80•)
```

Decimal delimiter

Regardless of system settings BeSweet always uses a dot as decimal delimiter, never a comma. I. e. $\frac{1}{2}$ always must be written as 0.5, never as 0,5.

Default values

Some switches have a default value (common with `-azid()`). BeSweet uses the default value, if such a switch is omitted. If you want to, for example, tell Azid to perform a Dolby Pro Logic downmix, you don't have to specify the appropriate parame-

ter (`-s surround`) explicitly because BeSweet automatically assumes `-s surround`, if `-s` is not present. All functions without a default value are only active if given explicitly. If you omit OTA's delay parameter (`-d`), no delay will be applied to the output file.

Case sensitive switches

Switches of most sections are case sensitive. E. g. `-6ch` and `-6CH` (`-core()` section) are two different switches. That's why you should always use switches exactly as given in this reference.

3 Global BeSweet Configuration

3.1 The `-core()` Section

Syntax

```
-core( -input -output [-logfile|-logfilea] [-seek|-substream] [-noid3]
[-be] [-2ch|-6ch|-6CH|-6chfloat|-6chwav|-5chaiff|-6chaiff|-6chliff|
-6chogg|-wavmp3|-ddwav|-payload|-6chnothing|-2chnothing] )
```

This section contains BeSweet's basic configuration, including definitions of source, destination and log files as well as switches for output formats not needing an encoder section.

3.1.1 Basic Parameters

Switches `-input` and `-output` are both required. All the others are optional.

`-input <source file>`

Path and name of source file. Of course, relative paths are possible. Paths containing spaces need to be enclosed in quotes according to Windows conventions. File names containing wildcards (`*` and `?`) are not allowed.

```
-input "D:\Audio\stream 1.ac3"
```

BeSweet accepts these types of source files:

MPEG-Audio (*.MP3, *.MP2, *.MPA)

MPEG-1 Layer II and Layer III in mono and stereo.

Dolby Digital (*.AC3)

Mono, stereo and multichannel.

Ogg Vorbis (*.OGG)

Mono, stereo and multichannel.

Wave (*.WAV)

Mono, stereo and multichannel waves.

16 bit waves only up to 2 Gbytes. 32 bit floating point waves without a size limit.

Raw PCM stream (*.PCM)

“headerless waves”

BeSweet assumes 16 bit, stereo, 48 kHz. Different sampling rates may be given using `-ota( -fs <Hz> )`.

AVIs (*.AVI)

AVI files containing AC3, MPEG or PCM audio streams.

PCM may be 8 bit or 16 bit with mono, stereo or multiple channels.

VOBs (*.VOB)

DVD video object files containing AC3 and/or PCM audio. PCM may be 8 bit or 16 bit with mono, stereo or multiple channels. 20 bit and 24 bit are not supported.

BeSweet listfile (*.LST)

Plaintext file containing paths to a set of source files (one path per line). Those files will be encoded to one output file.

BeSweet assumes 48 kHz sampling frequency. Different frequencies may be given using `-ota( -fs <Hz> )`.

BeSweet muxing file (*.MUX)

Plaintext file similar to the listfile. A muxing file contains paths for up to six mono waves (one path per line), that will be processed like one multichannel wave. The order of the file names defines the output file's channel order.

-output <destination file>

Path and name of destination file. Of course, relative paths are possible. Paths containing spaces need to be enclosed in quotes according to Windows conventions. File names containing wildcards (* and ?) are not allowed.

```
-output "D:\Audio\stream 1 transcoded.ogg"
```

Given file extension does *not* define the output format, which is handled by destination format switches or an encoder section. BeSweet does not correct wrong extensions. This means it is (not very useful but) perfectly possible to create e. g. an Ogg Vorbis file with extension .ac3.

Warning: Already existing output files will be overwritten without confirmation!

-logfile <logfile name>

Path and name of logfile, into which BeSweet writes information about the transcoding process. Without this switch BeSweet will output to the screen.

```
-logfile "D:\Audio\Logfile.log"
```

-logfilea <logfile name>

Similar to -logfile, but appends new logs to an existing file instead of overwriting.

```
-logfilea "D:\Audio\Logfile.log"
```

3.1.2 Switches for VOB Input

`-seek`

Displays stream numbers (see `-substream`) for every audio track contained inside a VOB. Does not transcode anything. This switch only requires a source file given using `-input`. Additional switches will be ignored.

`-substream <stream number>`

The audio track with given stream number will be transcoded. Numbers are hexadecimal values. A DVD normally contains AC3 inside streams 0x80 to 0x88, PCM inside 0xA0 to 0xA7 and MPEG audio inside 0xC0 to 0xC7. Streams 0x89 to 0x8F are reserved for optional audio formats like DTS. Standalone players without a proper decoder either ignore those or just output the digital stream.

If input file is a VOB and `-substream` omitted, BeSweet automatically decodes the file's first stream.

```
-substream 0x83
```

3.1.3 Destination Format Switches

Apart from exceptions (`-6chogg`, `-wavmp3`, `-ddwav`) these switches are only needed when no encoder section is defined. In most of those cases BeSweet won't be used for transcoding, but for decoding or sound processing only.

`-2ch`

Creates a 16 bit stereo wave. Channel order is: FL, FR.

`-6ch`

Creates six 16 bit mono waves, one for every source channel. If the source lacks a channel, the corresponding wave will contain silence.

Assumed channel order is: FL, FR, C, LFE, SL, SR.

`-6CH`

Creates six 16 bit mono waves, one for every source channel. If the source lacks a channel, the corresponding wave will contain silence.

Assumed channel order is: FL, C, FR, SL, SR, LFE.

`-6chfloat`

Creates six 32 bit floating point mono waves, one for every source channel. If the source lacks a channel, the corresponding wave will contain silence.

Assumed channel order is: FL, FR, C, LFE, SL, SR.

`-6chwav`

Creates a 16 bit multichannel wave.

Channel order of this file is: FL, FR, C, LFE, SL, SR.

`-5chaiff`

Creates a 16 bit, 5 channel, big endian AIFF file.
Channel order of this file is: FL, FR, C, SL, SR.

`-6chaiff`

Creates a 16 bit, 6 channel, big endian AIFF file.
Channel order of this file is: FL, FR, C, LFE, SL, SR.

`-6chliff`

Creates a 16 bit, 6 channel, little endian AIFF file.
Channel order of this file is: FL, FR, C, LFE, SL, SR.

`-6chogg`

Together with `-ogg()` creates a multichannel Ogg Vorbis file. If no `-ogg()` section is there, BeSweet encodes with VBR quality 0.800.
Assumed channel order of input file is: FL, C, FR, SL, SR, LFE.

`-wavmp3`

Together with `-lame()` transcodes to MP3 and writes an additional wave header to the file. That is useful for muxing MP3s with programs like VirtualDub (but not VirtualDubMod), that only can handle waves. Only use the switch for those special cases. Without a `-lame()` section an 128 kbit/s CBR stereo MP3 will be created.

`-ddwav`

Together with `-ac3enc()` creates a Dolby Digital wave. Essentially that's an AC3 with additional wave header, similar to `-wavmp3`.
Without an `-ac3enc()` section BeSweet will encode using 640 kbit/s.

`-payload`

Does not decode anything, but copies the source audio unprocessed. Useful for demuxing tracks from VOBs, for splitting files without decoding, for inserting delays etc.

3.1.4 Tweaking Switches

`-6chnothing`

Decodes a 6 channel stream without writing data to disk. Used by AacMachine to normalise tracks from a VOB.

`-2chnothing`

Decodes a stereo stream without writing data to disk. Used by AacMachine to normalise tracks from a VOB.

`-noid3`

Disables the ID3V1 tag that is usually added to the output file.

`-be`

Tells BeSweet to assume a 16 bit big endian wave as source file.

3.2 The `-split()` Section

Syntax

```
-split( [-start] [-end] )
```

This function provides partial transcoding of a source file. You are not forced to use both `-start` and `-end`. Without `-start` starting point is the beginning of the file, without `-end` transcoding continues to the end of the file.

Both values are absolute. E. g. a starting point at 6 seconds and an end point at 10 seconds produces an output file of 4 seconds in length.

`-start <sec>`

Sets the starting point for processing the source file to the given value in seconds. Fractions are allowed. The example below sets the starting point to 10 seconds and 80 milliseconds.

```
-start 10.080
```

`-end <sec>`

Sets the end point for processing the source file to the given value in seconds. Fractions are allowed. The example below sets the end point to 90 seconds and 525 milliseconds.

```
-end 90.525
```

3.3 The `-plugin()` Section

Syntax

```
-plugin( -name -func [-6ch] )
```

Features can be added to BeSweet using plugins, whose functions are invoked using the `-plugin()` section.

`-name <DLL name>`

Sets the plugin's file name.

```
-name MyPlugin.dll
```

`-func <name>`

Executes a function called `<name>` provided by the plugin.

`-6ch`

Inputs 6 channel data to the plugin. Necessary, if the plugin performs downmixing or similar actions.

4 Decoder and Sound Manipulation

4.1 The `-ota()` Section

Syntax

```
-ota( [norm|-g|-G|-hybridgain] [-d] [-e] [-sc] [-fs] [-r] )
```

OTA is an acronym for “overall track adjustment”. Mainly it is used for normalising audio streams and inserting delays.

4.1.1 Normalising Switches

`-norm <level>`

Normalises the file before encoding (pre gain normalisation) to a given percentage. Utilises a two-pass method. The first pass finds the audio stream’s peak level. Using this value and the given `<level>` the appropriate gain value is then calculated and applied in the second pass, where the stream gets encoded to its final format.

`<Level>` is a floating point value between 0 and 1, where 1 means 100 % or maximum possible normalisation.

```
-norm 0.97
```

`-g {<dB>|max|peak}`

Used for raising or lowering audio volume by a fixed number of decibels before encoding (pre gain). E. g. 5db raises every sample by 5 dB, -5dB attenuates by 5 dB.

```
-g -5db
```

`max` uses the highest possible value not distorting the sound, working identical to `-norm 1`.

```
-g max
```

`peak` uses the value stored within the source waves peak chunk. If no such value exists or the source file is not a wave, `peak` works identical to `max`.

```
-g peak
```

Compared to Azid’s `-g` switch, OTA’s `-g` is a bit slower, but also a bit more precise. Furthermore if a listfile is used, it normalises all sources as one, eliminating the problem of volume jumps in the output file.

`-G <level>`

Normalises the output file after encoding (post gain normalisation) to a given percentage. The value is a floating point number between 0 and 1, where 1 means 100 % or maximum possible normalisation. If the destination file is not MP2, MP3 or Vorbis, `-G` works identical to `-norm`. Unfortunately this is not true for AAC, which always needs pre gain (`-g` or `-norm`) specified explicitly.

```
-G 0.97
```

-hybridgain

Combines a fixed pre gain value with post gain normalisation depending on source and destination formats.

source file	Dialog Normali- zation	pre gain for different dynamic compression (Azid's -c)				
		none	light	normal	heavy	inverse
6 channel AC3	on	7 dB	8 dB	10 dB	14 dB	5 dB
Stereo AC3	off	0 dB	0 dB	0 dB	0 dB	0 dB

Above pre gain values will be used for stereo output only. If output is multichannel, pre gain always will be set to 0 dB.

When encoding is done, BeSweet post gain normalises to 97 %. For MP3 lame's scale switch will be set to 1, too.

If source is not AC3 or destination is not MP2, MP3 or Vorbis, -hybridgain works identical to -g max. Unfortunately this is not true for AAC, which always needs pre gain (-g or -norm) specified explicitly.

4.1.2 Delay Switches**-d <ms>**

Inserts given number of milliseconds of silence at start of the file or removes first milliseconds (for negative values).

```
-d 112
-d -80
```

-e <ms>

Appends given number of milliseconds of silence to end of file.

```
-e 1600
```

-sc <resolution>

Sets minimum sample resolution for delay operations. Used by AACMachine to only insert complete AAC frames for delays.

4.1.3 Switches for Sampling Frequency and Framerate**-fs <Hz>**

Skips automatic recognition of sampling frequency, enforcing the given value in hertz.

```
-fs 44100
```

This function does not convert sampling frequencies! BeSweet only without further checks assumes a source file with given frequency. Useful for sources with unknown sampling frequency, e. g. raw PCM files.

`-r <source fps> <destination fps>`

Converts the audio stream to a different framerate changing pitch as well as track length. This switch is especially useful when converting NTSC to PAL or vice versa.

Values are integers. BeSweet interprets the last three numbers as fractions.

Accordingly the following example would convert from 23.976 fps to 25.000 fps.

```
-r 23976 25000
```

Conversion is not fully available for every kind of 6 channel output. Only possibilities are creating 5.1 AC3s and separate mono waves.

4.2 The -soundtouch() Section

Syntax

```
-soundtouch( [-r|-tempo|-pitch|-rate] [-quick] [-naa] )
```

This section provides features to adjust speed and pitch of an audio stream, but only for mono and stereo output. For multichannel output OTA's `-r` might be an alternative.

On the whole `-soundtouch()` output should be of higher quality than `-ota ( -r )`.

`-r <source fps> <destination fps>`

Converts the audio stream to a different framerate without changing pitch. This switch is useful, for example, when converting NTSC to PAL or vice versa.

Values are integers. BeSweet interprets the last three numbers as fractions.

Accordingly the following example would convert from 23.976 fps to 25.000 fps.

```
-r 23976 25000
```

`-tempo <percent>`

Changes audio speed by the given percentage without affecting pitch. Valid values lie between -95 and +5000. Negative values are for slowing down, positive ones for speeding up.

```
-tempo -20
```

`-pitch <halftones>`

Changes pitch by given number of halftones without changing audio speed. Possible values are -60 to +60. Negative values are for lowering, positive ones for raising pitch.

```
-pitch -5
```

```
-pitch +15
```

`-rate <percent>`

Changes audio speed by given percentage, affecting pitch as well as track length. This switch works similar to OTA's `-r`. Values may be given between -95 and +5000, negative values slowing down and positive ones speeding up.

```
-rate -20
```

-quick

Uses a different algorithm for calculations, sacrificing quality for improved speed.

-naa

Disables anti-aliasing to gain speed at the expense of some quality.

4.3 The Sections -shibatch() and -ssrc()

Syntax

```
-shibatch( [--rate] [--normalize] [--att] )
-ssrc( [--rate] [--normalize] [--att] )
```

These sections convert the audio stream's sampling frequency. However, some problems are known. E. g. upsampling from 44.1 kHz to 48 kHz won't always work correctly. For such cases you should use an external program like SSRC.exe.

Both section names, -shibatch() and -ssrc(), are perfectly the same.

--rate <Hz>

Defines output sampling rate in hertz.

--rate 24000

--normalize

Normalises volume to 100 %. You should use OTA's or Azid's normalising instead.

--att <dB>

Attenuates output by given number of decibels.

--att 3

4.4 The -boost() Section

Syntax

```
-boost( {/b|/b2|/b3} [/1] )
```

This section is used for dynamic compression, just like Azid's -c. To avoid interference between the two function resulting in audio artefacts you should always use only one of them.

/b=<level>

Defines the dynamic compression's strength using an algorithm by LigH. Valid arguments range from 1 to 10. With a value around 3 this switch is especially useful for stereo sources.

/b=3

/b2=<level>

Defines the dynamic compression's strength using an algorithm by DSPguru. Valid arguments range from 1 to 10. With a value around 4 this switch is especially useful for 6-channel sources.

/b2=4

/b3=<level>

Defines the dynamic compression's strength using an algorithm by Tera. Valid arguments range from 1 to 10.

/b3=5

/l=<value>

Default value: 0.99

Defines maximum allowed level of the output file. <Value> is a percentage between 0 and 1, so 95 % need to be given as 0.95. No normalisation is done. The switch only defines a level the volume may not exceed.

/l=0.95

4.5 The -azid() Section

Syntax

```
-azid( [-g|--maximize] [-c] [-d] [-s] [-L] [-l] [-C] [-S] [-f] [-n] [-o]
[--ch] )
```

Azid decodes AC3 files and, if desired, performs a downmix from 6 channels. If BeSweet gets AC3 input but no -azid() section is defined, every option's default value will be used. For individual default values refer to the following pages. Default for the -d switch is not fixed but depends on the number of channels expected by the destination format. For MP3 stereo downmix would always be used because MP3 does not support more than two channels. For Vorbis or AAC, which do support multi-channel, -d's default is dependent on the options set in the respective encoder section.

4.5.1 Normalisation and Dynamic Compression

-g {max|<level>}

Performs pre gain normalisation similar to OTA's -g and -norm.

max is the same as --maximize below. The file first gets search for highest possible gain value, which is applied afterwards (2 pass approach like OTA's -norm).

-g max

<Level> may be used in different ways. A value between 0% and 99% works similar to -g max, using not highest possible gain but given percentage of that value.

-g 97%

A negative number followed by db works similar to above percentages. Only now the applied value is highest possible gain minus given number of decibels.

-g -3.2db

A positive number followed by `db` raises the volume by a fixed number of decibels, that is not relative to highest possible gain.

```
-g 4db
```

Compared to OTA's `-g` Azid's `-g` is faster but a little less accurate. Furthermore, when using listfiles Azid normalises every source file individually, possibly leading to volume jumps. To avoid this you should always normalise via `-ota()` when using listfiles.

```
--maximize
```

Performs pre gain normalisation identical to `-g max`. The file first gets search for highest possible gain value, which is applied afterwards (2 pass approach like OTA's `-norm`).

```
-c {none|light|normal|heavy|inverse}
```

Default value: none

Compresses the source file's dynamic range, i. e. reducing the range between loudest and most silent sample. The function has five modes.

none	No dynamic compression.
light	Light compression. This mode's compression is 50 % (-6 dB) lighter than normal.
normal	Normal compression. Decoders in standalone player commonly use this mode.
heavy	Heavy compression. Useful for noisy environments or low quality hifi equipment.
inverse	Inversion of the light-mode, i. e. loud samples become even louder, silent samples become more silent.

4.5.2 Downmix Switches

```
-d <front>/<rear>
```

Default value: see beginning of chapter

Defines for how many destination channels Azid should downmix. Important: This switch's values ignore the LFE channel. For stereo downmixing the `-s` switch is interesting, too.

Values are given as `channels front/channels rear`. Valid are: 1/0, 2/0, 3/0, 1/1, 2/1, 3/1, 1/2, 2/2, 3/2.

```
-d 2/0
```

```
-s {stereo|dpl|surround|dplii|surround2}
```

Default value: dpl

Controls the mode used for stereo downmixing. This switch only has an effect, if `-d 2/0` is selected.

stereo	Normal stereo downmix.
dpl or surround	Downmix compatible with Dolby Pro Logic.
dplii or surround2	Downmix compatible with Dolby Pro Logic II.

4.5 The -azid() Section

- `-L <level>` Default value: 0db
Changes the level used for mixing the LFE channel (bass channel) into left and right front channels. The value is given in decibels.
`-L -3db`
- `-l <level>` Default value: 0db
Changes the level used for mixing the source LFE channel into the destination LFE channel. The value is given in decibels.
`-l -2.5db`
- `-C {<level>|BSI}` Default value: BSI
Changes the level used for mixing the center channel into left and right front channels. The value is given in decibels. `BSI` sets the level according to the values contained in the source AC3's BSI info field.
`-C -2db`
- `-S {<level>|BSI}` Default value: BSI
Changes the level used for mixing the rear channels into left and right front channels. The value is given in decibels. `BSI` sets the level according to the values contained in the source AC3's BSI info field.
`-S BSI`
`-S -1.5db`
- `-f {0|1}` Default value: 0
Filters the rear channels when downmixing to stereo. The filter is a 2nd order butterworth filter providing an increasing phase shift according to frequency. At 7 kHz it is 90 degrees and -3 dB.
Major applications for this filter include ensuring a proper pro logic downmix and/or correcting phasing problems (washy sound) for weirdly mastered audio streams.
`-f 1` turns the filter on, `-f 0` or the switch missing turns it off.
- `-n {0|1}` Default value: 0
Activates "dialog normalisation reduction" (DNR). 5.1 AC3 files have a field in their BSI info stating how far below maximum level the dialog channel's (front center) subjective loudness is. According to that value the `-n` switch changes the dialog level to -31 dB. Normalisation is not affected by this because DNR is applied before any normalisation function.
DNR is only recommended for two scenarios. First: When you transcode several 5.1 AC3s with different dialog levels to one output file. Second: For equalising different dialog levels within a single source AC3 (possibly a TV capture with loud commercials). DNR has no effect when transcoding AC3 input without different dialog levels (should be the case for most DVD audio streams). Additionally AC3s without a center channel (e. g. 2/0) are not affected.
`-n 1` turns the filter on, `-n 0` or the switch missing turns it off.

4.5.3 Switches for Controlling Individual Channels

These switches are not needed for ordinary AC3 transcoding. `-o` becomes important, when changing channel order from input to output file. The `--ch` switches should only be used by experienced users with a good reason for doing so.

`-o <channel order>`

Defines the output file's channel order. This function is only needed when channel order for input and output file is different.

Channels are given as abbreviations and separated by commas. Note that channel names refer to their position in the *input file*.

<code>l</code>	FL channel of the source (front left)
<code>r</code>	FR channel of the source (front right)
<code>c</code>	C channel of the source (front center)
<code>lfe</code>	LFE channel of the source
<code>sl</code>	SL channel of the source (rear left)
<code>sr</code>	SR channel of the source (rear right)

Use this switch, for example, to transcode 5.1 AC3 to 5.1 Vorbis. AC3 has a channel order of FL, FR, C, LFE, SL, SR while Vorbis uses FL, C, FR, SL, SR, LFE.

For this example the `-o` switch should look like this:

```
-o l,c,r,sl,sr,lfe
```

`--ch<number>={light|normal|heavy|inverse},<dB>`

Defines separate values for dynamic compression and gain for individual channels.

`<Number>` stands for the channel's position (0 to 5) in the output file. So you must consider any changes done by `-o`.

The first value (before the comma) sets dynamic compression mode. Same modes as for `-c` are used. The second value sets a fixed gain value in decibels for this individual channel. Write a `db` following the value.

Example: To use light compression and attenuate volume by 2.5 dB for channel number 4 (defaults to SL for AC3) use the switch like this:

```
--ch4=light,-2.5db
```

`--ch<abbreviation>={light|normal|heavy|inverse},<dB>`

This switch works similar to `--ch<number>`, but identifying channels by their position in the source. Abbreviations are the same as for `-o`.

Above example (SL channel with light compression and attenuation of 2.5 dB) now would look like this:

```
--chsl=light,-2.5db
```

5 Encoders

These sections must not be combined, meaning you can only use one of them in a single command line. If no encoder section is defined, you should use one of `-core()`'s destination format switches.

5.1 The `-ogg()` Section

Syntax

```
-ogg( [-q|{-b [-m] [-M]}}] [-6ch] )
```

Encodes to Ogg Vorbis. The BeSweet package does not include a Vorbis encoder. To be able to use `-ogg()` Ogg Vorbis DLLs (download e. g. on www.rarewares.org) must be present in BeSweet's folder or in a folder contained in the path environment variable.

Vorbis is primarily designed to use real VBR in constant quality mode. That is why you should always use the `-q` switch to define output quality and bitrate.

The section may be defined empty as `-ogg()`. Then BeSweet will encode with an average bitrate of 160 kbit/s. (identical to `-ogg( -b 160 )`).

`-q <quality>`

Sets a quality level for encoding the file. Valid values are `-0.100` to `1.000`.

By default Vorbis uses a scale from -1 to 10 being different from BeSweet's default of -0,1 to 1. Converting is easily done by multiplying BeSweet's value by 10 or dividing the Vorbis value by 10. BeSweet's 0.250 would be 2.50 for Vorbis and vice versa.

```
-q 0.200
```

For values below 0 BeSweet's screen output reads `Average Bitrate: 160` (yes, including the typo), but actual transcoding is done with the correct value.

`-6ch`

Creates a 6 channel Ogg Vorbis with channel order FL, C, FR, SL, SR, LFE. If the source's order is different, channels must be redirected accordingly. 6 channel Vorbis is not recommended because it is not tuned and needs high bitrates. For good quality bitrates hardly below typical AC3 are needed.

Switch does not work with `-ssrc()`, `-boost()` and `-ota( -g )`.

`-b <kbps>`

Sets nominal bitrate in kbit/s (ABR mode). Not recommended. For best quality use `-q` instead.

```
-b 128
```

`-m <kbps>`

Sets minimum allowed bitrate in kbit/s (also needs `-b`). Not recommended. For best quality use `-q` instead.

`-m 64`

`-M <kbps>`

Sets maximum allowed bitrate in kbit/s (also needs `-b`). Not recommended. For best quality use `-q` instead.

`-M 160`

5.2 The `-bsn()` Section

Syntax

```
-bsn( [-2ch|-6chold|-6chnew] [-config] [-aacprofile_{he|lc}]
[-cbr|-vbr_{tape|radio|internet|streaming|normal|extreme|audiophile|
transcoding}] [--codecquality_{high|fast}] [-downmix] [-usepns]
[-hintforstreaming] [-muxintovideo] [-exportaac] )
```

This function uses Nero to encode to AAC. To enable `-bsn()` Nero including its AAC encoder must be installed. BeSweet needs Nero's *aac.dll* and *aacenc32.dll*. For older Nero versions possibly *NeroIPP.dll*, too. These files need to be copy to BeSweet's folder from *C:\Program Files\Common Files Ahead\AudioPlugins* and *...Ahead\Lib*.

The section may be defined empty as `-bsn()`. BeSweet will then encode to stereo using HE-AAC and VBR *streaming* preset.

Output file name always must be given with full path or BeSweet will transcode but not write anything to disk.

`-2ch`

Creates a stereo AAC file.

`-6chold`

Creates a 6 channel AAC file when using Nero version 6.0.11 or older.

`-6chnew`

Creates a 6 channel AAC file when using Nero version 6.0.12 or newer.

`-config`

Before encoding starts opens Nero's configuration dialog to set desired options. When using `-config` nothing else needs to be defined on the command line apart from number of channels (see above). If `-config` is omitted as well as all following switches, `-bsn()` will use its default values (see beginning of chapter). Additionally when using `-config` BeSweet automatically resamples to 44.1 kHz, while leaving sampling rate alone without the switch. Both methods won't introduce sync issues.

`-aacprofile_{he|lc}`

Default value: he

Sets the AAC profile for encoding.

LC encodes the complete frequency spectrum. That is what other encoders (MP3, Vorbis etc.) usually do. HE uses a technology called SBR, that does not encode the high frequency range but only stores recovery information. Using those and certain information from the low frequency range a decoder can reconstruct the complete spectrum.

HE is especially useful for low bitrates.

`-aacprofile_he``-cbr <kbps>`

Encodes with constant bitrate. The value is given in kbit/s from 16 to 448. For best possible quality use VBR instead.

`-cbr 96``-vbr_{tape|radio|internet|streaming|normal|extreme|audiophile|transcoding}`

Default value: streaming

Uses VBR and the given preset for encoding. Tape produces lowest bitrate and worst quality, transcoding produces highest bitrate and best quality.

`-vbr_streaming``--codecquality_{high|fast}`

Is meant to define if the encoder should focus on speed (*fast*) or quality (*high*). You should always use *high*, because despite its name *fast* is not faster.

`--codecquality_high``-downmix`

Downmix to mono. Useful only when BeSweet provides stereo input via `-2ch`, but output should be mono.

`-usepns`

Activates PNS, basically removing noise from the source, not encoding it and adding back random noise when decoding. Useful mainly for very low bitrates.

`-hintforstreaming`

Creates a hint track for using the file on streaming servers.

`-muxintovideo`

Used for muxing output into an already existing MP4 video file.

`-exportaac`

Creates ISO 1318-7 AAC output.

5.3 The -lame() Section

Syntax

```
-lame( [--preset] [-m] [-a] [--resample] [--lowpass] )
```

Lame encodes MP3 files. I intentionally did not list all of lame's options supported by BeSweet. Except for very special cases you will not need more than the `--preset` switch to control quality and bitrate.

The section may be defined empty as `-lame()`. Then BeSweet will encode to stereo with a constant bitrate of 128 kbit/s (identical to `-lame( --preset cbr 128 )`).

```
--preset {[fast] medium|[fast] standard|[fast] extreme|
insane}|cbr <kbps>|<kbps>}
```

Defines mode and quality for encoding the MP3. Constant quality is set by `cbr` followed by a value in kbit/s. Valid are 8, 16, 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256 and 320 kbit/s. Encoding to a constant bitrate of 192 kbit/s looks like this:

```
--preset cbr 192
```

Additionally the named preset `insane` uses a constant bitrate of 320 kbit/s. Writing `--preset insane` is the same as `--preset cbr 320`.

An average bitrate (ABR) is set simply by any number in kbit/s between 77 (for lame 3.90 shipped with BeSweet) or 72 (for lame 3.96) and 320. Encoding to an average bitrate of 135 kbit/s looks like this:

```
--preset 135
```

Variable bitrate is set by a named preset. From lowest to highest bitrate (= quality) they are called `medium`, `standard`, `extreme`. Every preset may be preceded by `fast` to use the new VBR mode, that is a little less accurate but a lot faster.

```
--preset standard
```

```
--preset fast medium
```

Instead of `--preset` you may also use `--alt-preset`.

```
--alt-preset standard
```

```
--alt-preset 150
```

```
--resample <Hz>
```

Changes destination sampling frequency to the given value in Hertz. Real resampling should be done by `-shibatch()`. This switch is there to prevent lame's automatic downsampling for low bitrates (below 88 kbit/s). Possible values for `<Hz>` are 8000, 11025, 12000, 16000, 22050, 24000, 32000, 44100, 48000. To keep a source sampling rate of 44100 Hz for an encoding to below 88 kbit/s, the switch would read

```
--resample 44100
```

```
--lowpass <kHz>
```

Defines lowpass frequency in kilohertz. All frequencies above given value will be truncated. Maximum possible value is half the sampling frequency. When using a

48 kHz source that would be 24. Reasonable values are always lower because there cannot be frequencies above 24 kHz in an 48 kHz file.

```
--lowpass 15.5
```

Lame applies lowpass filters automatically, depending on chosen quality setting. Only reasonable use for this switch I can think of: When using very low bitrates you might be better off when sacrificing a part of the high frequencies in exchange for increased bitrate (= quality) for the lower spectrum. You'll have to try if and when exactly this might be true. When in doubt better do not use `--lowpass`.

`-m {s|j|f|m}` Default value: j

This switch is only necessary for downmixing to mono. Use it like this:

```
-m m
```

The other three modes set the method used for encoding stereo channels.

s normal stereo

j joint stereo

f forced joint stereo

The default j is fine for stereo encodings and should not be changed.

5.4 The Section `-mp2enc()` or `-toolame()`

Syntax

```
-mp2enc( [-b|-v] [-m] )
-toolame( [-b|-v] [-m] )
```

By the name of this section you chose, which MP2 encoder BeSweet should use: MP2enc or 2Lame. Switches are the same for both.

The section may be defined empty as `-mp2enc()` or `-toolame()`. Then BeSweet will encode in dual channel mode using 192 kbit/s CBR.

`-b <kbps>` Default value: 192

Uses CBR encoding with given bitrate in kbit/s.

```
-b 192
```

`-m {s|j|d|m}` Default value: d

Sets the stereo mode used. s for normal stereo, j for joint stereo, d for dual channel and m for mono.

```
-m s
```

`-v <quality>`

Uses VBR encoding with given quality level. Possible are values between 0 and 9. Warning: VBR mode is experimental and not guaranteed to work properly.

```
-v 5
```

5.5 The -ac3enc () Section

Syntax

```
-ac3enc ( [-b] [-6ch] )
```

Encodes to AC3. When using `-ac3enc ()` normalisation must be done by Azid. OTA's normalisation settings will be ignored.

The section may be defined empty as `-ac3enc ()`. Then BeSweet will create a 6 channel AC3 with 384 kbit/s.

`-b <kbps>`

Default value: 384

Sets bitrate in kbit/s. By default a video DVD uses 448 and 384 kbit/s for 6 channel AC3 and 192 kbit/s for stereo AC3.

```
-b 448
```

`-6ch`

Creates a 6 channel AC3. Without this switch set output channels will be the same as input channels.

6 Credits

Many thanks to

DSPguru and the coders of the DLLs for the best audio transcoding tool ever.

Doom9 for doom9.org.

Gleitz for the German doom9.org mirror and its great forum.

LigH and *akapuma* for all their feedback on the German version.