Saracon

Ultra High-Quality Audio-File And Sample-Rate Conversion Software

Manual

Please see page two for version of this manual.

Weiss Engineering Ltd.
Florastrasse 42, 8610 Uster, Switzerland
Phone: +41 44 940 20 06, Fax: +41 44 940 22 14
Email: weiss@weiss.ch, Websites: www.weiss.ch or www.weiss-highend.com

This is the manual for **Saracon** on Macintosh: © Weiss Engineering LTD. September 16, 2026

Typeset with LaTeX 2_ε.

Author: Uli Franke
Acknowledgements: Daniel Weiss, Rolf Anderegg, Andor Bariska, Andreas Balaskas, Alan Silverman, Kent Poon, Helge Sten, Bob Boyd, all the beta-testers and all other persons involved.

Saracon Version: 01 . 64 - 01
Manual Revision: 00.03

Legal Statement

The software (**Saracon**) and this document are copyrighted. All algorithms, coefficients, code segments etc. are intellectual property of Weiss Engineering LTD.. Neither disassembly nor re-usage or any similar is allowed in any way. Contravention will be punished by law.

Information in this document is provided solely to enable the user to use the **Saracon** software from Weiss Engineering LTD.. There are no express or implied copyright licenses granted hereunder to design or program any similar software based on the information in this document. Weiss Engineering LTD. does not convey any license under its patent rights nor the rights of others.

Weiss Engineering LTD. reserves the right to make changes without further notice to any products herein. Weiss Engineering LTD. makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Weiss Engineering LTD. assume any liability arising out of the application or use of any part of this software or manual, and specifically disclaims any and all liability, including without limitation consequential or incidental damages.

Contents

1 Introduction **7**
1.1 Different Editions of **Saracon** and this Manual . . . 7

2 Conversion Mode **7**
2.1 PCM to PCM - Mode . . . 7
2.2 PCM to DSD - Mode . . . 7
2.3 DSD to PCM - Mode . . . 7
2.4 Selecting the Mode . . . 7

3 Configurations **7**
3.1 Loading and Saving Configurations . . . 10
3.2 Common Settings . . . 10
3.2.1 Source files . . . 10
3.2.2 Batch Modes . . . 10
3.2.3 Gain . . . 10
3.2.4 Sample Rate . . . 10
3.2.5 Destination Folder . . . 10
3.2.6 Postfix . . . 11
3.3 P2P and D2P Settings . . . 11
3.3.1 Format . . . 11
3.3.2 Dither . . . 11
3.3.3 Number Format . . . 11
3.3.4 Gain during D2P . . . 11
3.4 P2D Settings . . . 11
3.4.1 Format . . . 11
3.4.2 Modulator Type . . . 11
3.4.3 Channel Mode/Channel Strings . . . 11
3.4.4 Stabilizer . . . 11
3.4.5 Dither . . . 11
3.4.6 Metering/History . . . 12

4 Conversions **12**
4.1 Job . . . 12
4.2 Running Conversions . . . 12
4.3 Controlling Conversions . . . 12
4.4 Conversion Progress . . . 12

5 Post Processing **12**
5.1 Automatic Variables . . . 13
5.2 Examples . . . 13
5.2.1 Create a MP3-Encoded Copy . . . 13
5.2.2 Conversion to FLAC with Preservation of Metadata . . . 13
5.2.3 Complex Renaming of Output Files . . . 14
5.2.4 Parallel Transfer of Tags . . . 14
5.3 General Notes . . . 14
5.3.1 A Word on the System-Interface . . . 14
5.3.2 Debugging Post Processing Commands . . . 14
5.4 Clear Inputs . . . 14

6 Super Batch **14**

7 Log-Window **14**

8 Drag and Drop **15**

9 Preferences **16**
9.1 Processing Buffer Size . . . 16
9.2 Templates . . . 16
9.3 Generating/Setting a Template . . . 16

10 Signal Generator **16**

11 Command Line **17**
11.1 Signal Generator . . . 17
11.2 Conversion . . . 17
11.3 Batch-Processing of Configurations . . . 18
11.4 Open Configurations . . . 18
11.5 Other Options/Switches . . . 18
11.5.1 Regression Test Mode . . . 18
11.5.2 Tolerant Mode . . . 18
11.5.3 Log Message Verbosity . . . 18
11.5.4 Keep Console . . . 18
11.6 Requirements . . . 18
11.7 Examples . . . 19
11.7.1 Signal Generator Example . . . 19

12 Modulation Level History **20**
12.1 Graphs . . . 20
12.1.1 Modulation Level . . . 20
12.1.2 Modulator Input Level . . . 20
12.1.3 Reset Count . . . 20
12.2 Window Elements . . . 20
12.2.1 Channel ID . . . 20
12.2.2 Time Line . . . 20
12.2.3 Level Indicators . . . 20
12.3 Loading Histories . . . 20
12.4 Saving Histories . . . 20
12.5 Scrolling . . . 20
12.6 Horizontal Zoom . . . 20
12.7 Vertical Zoom . . . 21
12.8 Graph Control . . . 21
12.9 Finding Maxima . . . 21
12.10 Hiding the History . . . 21
12.11 Experience . . . 21

13 Licensing **23**
13.1 Updating Licenses . . . 23
13.1.1 Updating License Files . . . 23
13.1.2 Updating Dongle Licenses . . . 23

14 Technical Data **25**
14.1 PCM to PCM Technical Data . . . 25
14.1.1 Sonic Performance . . . 25
14.1.2 PCM to PCM Features . . . 25
14.2 PCM to DSD Technical Data . . . 25
14.2.1 Sonic Performance . . . 25
14.2.2 PCM to DSD Features . . . 25
14.3 DSD to PCM Technical Data . . . 28
14.3.1 Sonic Performance . . . 28
14.3.2 DSD to PCM Features . . . 28

14.4 Speed Benchmarks . . . 29
14.5 Multiprocessor Support . . . 29
14.6 System Requirements . . . 30
14.6.1 Macintosh . . . 30
14.7 Program Data . . . 30
14.7.1 Settings . . . 30
14.7.2 Log & License Files . . . 30

A Installation **31**
A.1 Macintosh . . . 31
A.1.1 Dongle Drivers . . . 31
A.1.2 **Saracon** . . . 31

B Release Notes **32**
B.1 1.50 . . . 32
B.1.1 Incompatibilities . . . 32
B.1.2 New/Changed Features . . . 32
B.2 1.60 . . . 32
B.2.1 New/Changed Features . . . 32
B.3 1.60-13 . . . 33
B.3.1 New/Changed Features . . . 33
B.3.2 Bug Fixes . . . 33
B.4 1.60-14 . . . 33
B.4.1 New/Changed Features . . . 33
B.4.2 Bug Fixes . . . 33
B.5 1.61-17 (Beta) . . . 33
B.5.1 New/Changed Features . . . 33
B.5.2 Bug Fixes . . . 34
B.5.3 Incompatibilities . . . 34
B.6 1.61-36 . . . 34
B.6.1 New/Changed Features . . . 34
B.7 1.61-37 . . . 34
B.7.1 Bug Fixes . . . 34
B.8 1.61-38 . . . 34
B.8.1 New/Changed Features . . . 34
B.9 1.61-39 . . . 34
B.9.1 Bug Fixes . . . 34
B.10 1.62-01 . . . 34
B.10.1 New/Changed Features . . . 34
B.11 1.63-01 . . . 34
B.11.1 New/Changed Features . . . 34
B.12 1.64-01 . . . 34
B.12.1 New/Changed Features . . . 35
B.13 Known Issues . . . 35

C SACD Specifications **36**
C.1 Audio Level measuring condition (D.1) . . . 36
C.2 Zero dB Audio Reference Level (D.2) . . . 36
C.3 Maximum Audio Peak Level (D.3) . . . 36
C.4 High Frequency DSD Signal and Noise Level (D.4) . . . 36

D FAQ **37**

References **39**

E Contact 39

Index 40

1 Introduction

Saracon is an audio file format conversion software targeted on professional audio-workstations. It incorporates proven and new cutting-edge Weiss-algorithms and combines them with the convenient data handling of workstations. Professional monitoring tools give insight into the conversion processes and give full control to the audio engineer. It has been designed to provide the audio engineer with a software audio file conversion tool of outstanding quality.

Saracon comes with an easy to use interface with lots of features including: Monitoring, log files, full-featured DSD converter with a graphical history, test signal generator and more.

Thank you for purchasing this software.

If you should ever miss a feature or feel uncomfortable with the ergonomics or are dissatisfied with the sonic integrity or anything else please do not hesitate to contact us (see the contact information provided in appendix E).

1.1 Different Editions of Saracon and this Manual

Saracon is available in different editions[1]. Some sections become irrelevant for the Standard edition. Those parts of the manual which are relevant to the DSD edition only are marked with a ♣.

2 Conversion Mode

Saracon operates at a single *mode* at a time. Each mode provides a specific kind of audio data/file conversion. There are three modes available:

2.1 PCM to PCM - Mode

The PCM[2] to PCM (P2P) audio conversion allows conversion from and to diverse PCM audio formats, sample rates, number formats, using sophisticated sample rate conversion and word length reduction algorithms.

2.2 PCM to DSD - Mode ♣

The PCM to DSD[3] (P2D) audio conversion accepts all PCM formats which are supported by the PCM to PCM conversion. The output is in DSDIFF[4] or DSF[5] format.

[1] DSD and Standard, see section 13
[2] Pulse Code Modulation
[3] DSD: *Direct-Stream-Digital*

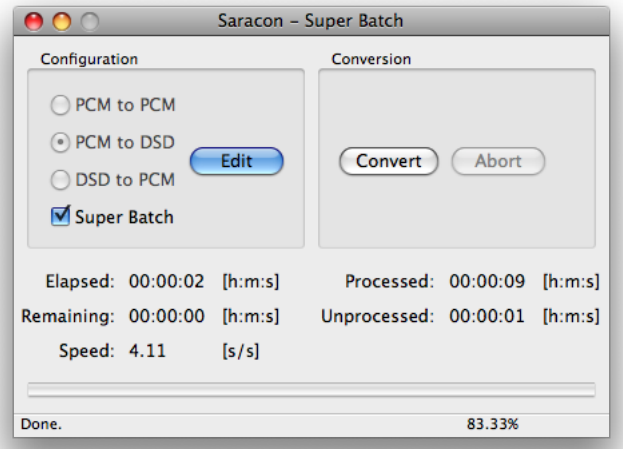

Figure 1: **Saracon**'s main window.

2.3 DSD to PCM - Mode ♣

The DSD to PCM (D2P) audio conversion accepts DSD data in the DSDIFF or DSF file format. The PCM output can be any file format which is supported by the PCM to PCM conversion.

2.4 Selecting the Mode

The mode can be selected in the main window – see figure 1, item 1 – or in the main menu *Edit::Set Configuration::...*.

3 Configurations

Each specific conversion requires precise information about the data to be converted, e.g. input files, output files, sample rate settings, quantization settings, batch mode .... This collected information is called a *configuration*. Configurations can be edited in their dedicated *configuration editor* – see figures 2 and 3.

To launch the configuration editor for the current conversion mode click *Edit* in the main window, the main menu item found under *Edit::Configuration* or hit the shortcut ⌘+Y.

[4] *DSD Interchange File Format*
[5] *DSD Stream File*

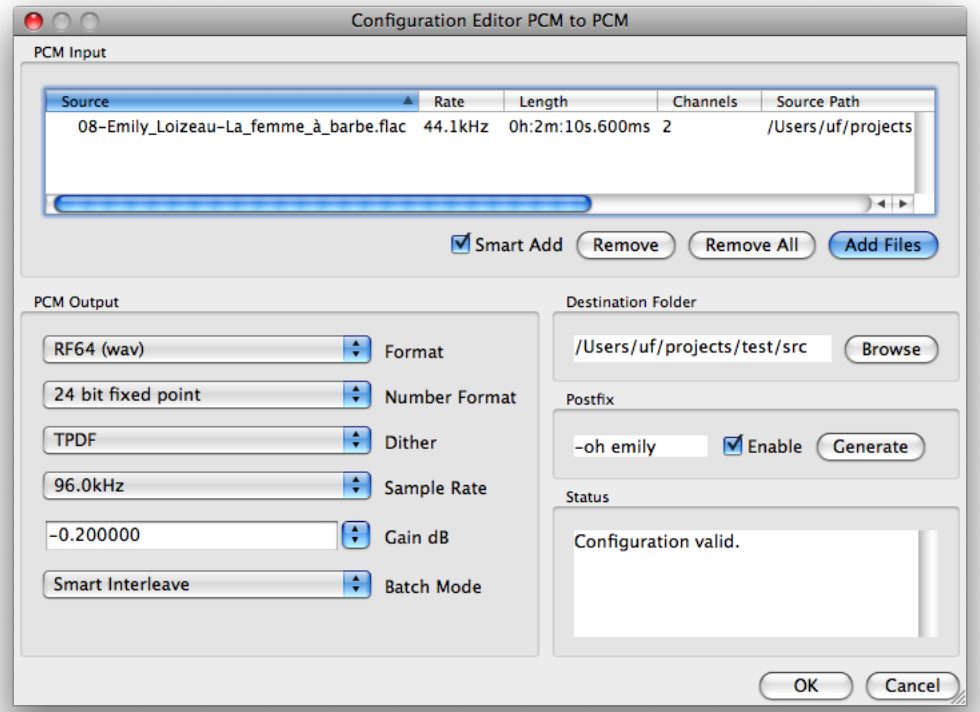

Figure 2: P2P and D2P configuration editor (the D2P configuration differs only in the input file information).

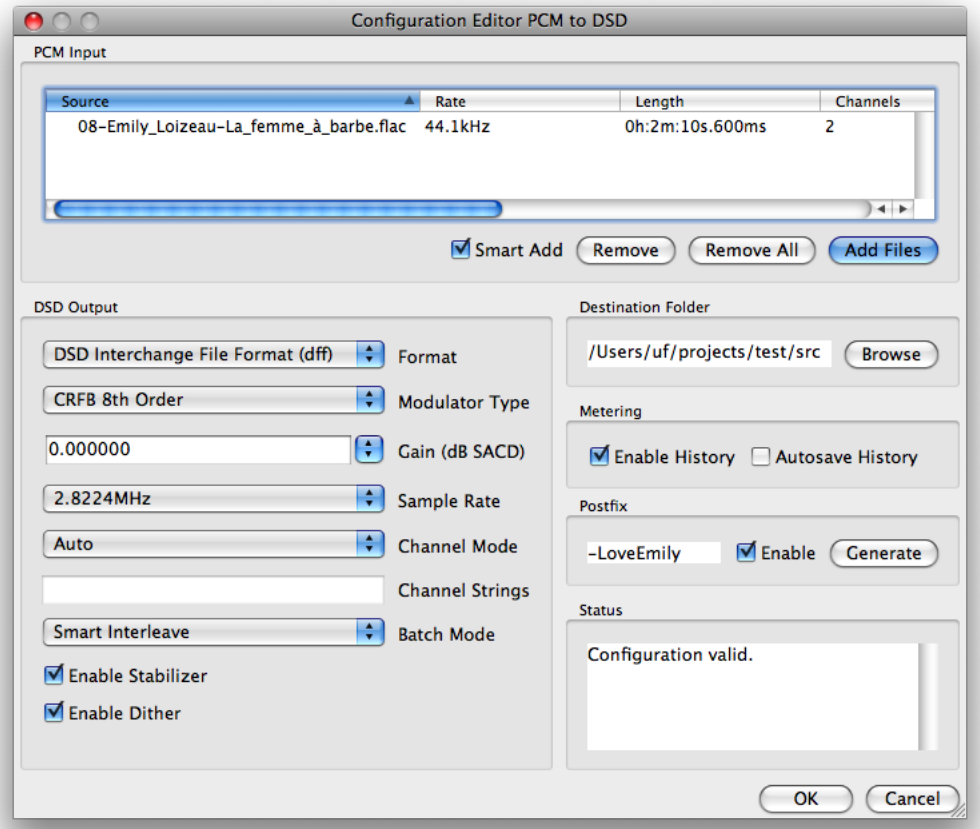

Figure 3: P2D configuration editor.

3.1 Loading and Saving Configurations

Configurations can be saved to and loaded from files. All configurations use the same file-format, the *.src format. This file-format uses plain text which makes it human-readable[6] and portable between applications/platforms.

To save a configuration to a file, select *File::Save Configuration* from the main menu or hit ⌘+S.

To load a configuration select *File::Open Configuration* from the main menu or hit ⌘+O. *File::Save As* saves the current configuration at a different place.

The 10 most recent configurations are directly accessible through the main menu (*File::Recent Configurations*).

The currently active configuration can be flushed to default values (empty) by clicking *File::Load Blank Configuration* or hitting ⌘+B.

All input files are saved as absolute paths in the configuration files. If the source files are moved, the paths in the configurations must be adapted either by manually editing the .src file or by loading the configurations and re-adding the source files.

3.2 Common Settings

All modes share some common settings which are explained here. The mode specific settings are explained in sections 3.3, 3.4 and 3.3 respectively.

3.2.1 Source files

Each configuration editor provides an *Add Files* button (or *insert* key on the keyboard) which allows the user to browse for files and select multiple input files to a configuration. Depending on the mode the file filters are initialized differently. If the file should have a different extension, the *.* filter can be selected.

The *Add Folder* button adds all supported files (in terms of extensions) of the selected directory recursively to the current configuration. *Smart Add* is engaged according to the check box' setting.

The *Remove* button (or *delete* key on the keyboard) removes the file which is currently selected in the input file list, the *Remove All* button removes all input files.

The *Smart Add* check box defines how additional input files are searched and added to support to the so called *Smart Interleave* batch mode (see section 3.2.2): When adding a single or multiple input files, **Saracon** searches for other channels of these input files based on the names of the selected files and adds them to the source files if found. Typical channel indications (here examples for the left channel) are `<name>.L.wav`, `<name>.L`, `<name>.1.wav`, `<name>.1`.

When adding files via drag and drop, *Smart Add* is always enabled.

3.2.2 Batch Modes

The batch mode determines which list of output files is generated from a list of input files. The user can select from four different batch modes:

Normal In this batch mode each input file generates exactly one output file.

Smart Interleave During Smart Interleave **Saracon** scans the input files for files which seem to be of the same group and interleaves them to a single file whereas the channel order follows the input file name convention. Example: If the source file list contains `file.x.L.wav`, `file.x.R.wav`, `file.y.L.wav`, `file.y.R.wav` and `file.z.wav`, the resulting files would be `file.x.wav`, `file.y.wav` and `file.z.wav` whereas `file.x.wav` and `file.y.wav` are interleaved files containing the channels of their `*.L.wav` and `*.R.wav` source files. File extensions which are recognized by Smart Interleave are `.L`, `.R`, `.1`, `.2`, ..., `.8` and the same extensions with the file format extension appended (as in the example above).

Interleave All In this mode all channels from all input files are interleaved to a single output file. The file/channel order corresponds to the order in the source file list.

Split In Split mode all channels of all input files end up in separate output files.

3.2.3 Gain

Defines the gain which is applied during the conversion in decibel (dB).

3.2.4 Sample Rate

Determines the sample rate of the target files.

3.2.5 Destination Folder

Determines the location of the target files. You can either set the location with the directory browser

[6] This can be useful if anything has to be edited by hand if something really goes wrong

which is opened with the *Browse* button or by editing the destination manually and confirming with *Enter*. When editing manually the path must exist exept for the last directory which is then created automatically.

3.2.6 Postfix

The postfix is a user defined string which is appended to the file name just before the file extension. If the source file is for instance `my-file.x.aiff` and you convert to FLAC, then, with the postfix set to `-mypfx`, the resulting file name would be `my-file.x-mypfx.flac`. When you edit the postfix you have to commit the changes with *Enter*.

3.3 P2P and D2P Settings

3.3.1 Format

Determines the output file format. The supported formats are listed in section 14.1.2 of the appendix.

3.3.2 Dither

Determines which kind of dither should be applied during word length reduction. Floating point output formats (see number formats below) are **not** dithered.

3.3.3 Number Format

Determines the number format of the output file. For certain output formats some number formats are not available. `ogg` files for example support *Vorbis* as number format only.

3.3.4 Gain during D2P

The gain during D2P conversion defaults to +6dB because the SACD/DSD encoding's maximal amplitude corresponds to -6dB in the PCM domain (those -6dB are usually referred to as *0dB SACD*, see section C of the appendix).

> **Note 1** *It must be payed attention though when reinflating the amplitude back to 0dB PCM because the DSD signal can contain amplitudes greater than those -6dB PCM*

3.4 P2D Settings

3.4.1 Format

Determines the output file format. The supported formats are listed in section 14.2.2 of the appendix.

3.4.2 Modulator Type

Determines the modulator which generates the one bit stream. Three models are available *CRFB6*, *CRFB8* and *CRFB10*. The SNR (signal to noise ratio) improves from *CRFB6* to *CRFB10* but the sensitivity to overload (instability) increases as well. The *CRFB8* is a good choice because it is quite resistant to overloads but provides more than 24 bit resolution (SNR > 147 dB). The *CRFB6* is the most stable modulator and uses less CPU than the others. The *CRFB10* provides the best SNR with tradeoff in stability and CPU, but for material which isn't compressed/limited to very high loudness (e.g. classical music) this modulator is the best choice.

3.4.3 Channel Mode/Channel Strings

DSDIFF files support certain predefined channel setups. If the channel mode is set to *Auto*, **Saracon** tries to match your channels to a predefined setup (2 channels → Stereo, 5 channels → 5-channel surround, 6 channels → 5.1 surround). If no default setup is found, **Saracon** set the channel strings to "CHxx" for channels 0 to 99 (xx corresponds to the channel number) and to "Cxxx" for channels 100 to 999.

For those who require different mappings **Saracon** provides the custom channel mode where the user can set his own channel strings. Channel strings are four characters long and must be separated by a comma. Confirm with *Enter*. When in custom channel mode all target files must have the same number of channels and must match the channel count defined by the custom channel strings.

3.4.4 Stabilizer

The stabilizer takes care of modulator overloads and resets the modulator if a certain modulation level is reached. This modulation level is defined by the maximum modulation level as outlined in appendix C.

3.4.5 Dither

The modulators can be dithered which reduces the distortions. Due to stability issues modulators can not by fully dithered but this dither is a good compromise.

Note 2 *Note that dither reduces the maximum input amplitude. It is advisable to reduce the conversion gain when enabling the dither. A good value to start with is -1.0 dB. The modulator resets can be controlled within the log window when the stabilizer is enabled (suggested) or within the metering/history. If resets occur, the gain must be reduced further.*

3.4.6 Metering/History

Enable metering. If enabled the RMS of the input signal, the modulation level at the modulator output and the number of resets can be monitored in real time. See section 12 for more information.

If the *Autosave History* button is enabled the modulation level history is saved automatically to the destination directory. The file name is the same as the first target file of the corresponding job with a `.mlh` extension.

4 Conversions

A conversion is a process which is defined by a configuration. Since batch modes can be employed a conversion can consist of multiple independent jobs.

4.1 Job

4.2 Running Conversions

A conversion can be run when its configuration has been determined to be valid. In the main window this is reflected through the *Convert* button which is enabled on valid configurations.

To launch a conversion, click the *Convert*-button in the main window or the *Process::Convert* entry in the main menu or hit the shortcut ⌘+*X*. You are probably prompted for file overwrites.

When successfully launched, the *Convert*-button becomes the *Pause*-button.

Should a conversion exit because of an error, the log window (7) will inform about the reasons.

4.3 Controlling Conversions

Suspending a conversion can be useful to assign the CPU-resources temporarily to another application. Resuming returns the CPU-power back to **Saracon**. You can suspend and resume conversions by clicking the *Convert/Pause/Continue* button in the main window (see figure 1), by clicking the *Process::Pause/Continue* item in the main menu or through the ⌘+*W* shortcut.

A conversion can be aborted by clicking the *Abort* button in the main window, through the *Process::Abort* main menu item or with the ⌘+*C* shortcut.

The already computed data will be written to the destination file(s) and the file will be closed properly and will be a valid file which can be opened with **Saracon** or other applications.

Note 3 *If the processing buffer size is set to large values a noticeable delay between abortion an actual process termination can occur because the current buffer will be finalized.*

4.4 Conversion Progress

The progress of a conversion is displayed through several indicators in the main window. Beneath the progress bar which shows the progress of each job. The status bar displays how many jobs are left. If a super batch is processed, the amount of remaining conversions is displayed as well. Some other elements:

Elapsed: Elapsed time since the last sub-conversion has been started.

Remaining: Estimate for the time required to complete the current sub-conversion.

Speed: Current processing-speed in "seconds of output material" produced every "real-time second".

Processed: Seconds of audio-data converted.

Unprocessed: Seconds of audio-data still to be converted.

5 Post Processing

Post processing allows the user to run arbitrary commands/programs on the results/output files. Such commands could be used to

- Run additional conversions
- Perform complex renaming tasks
- Perform transfer of meta-data which is not supported by **Saracon**
- Upload files to a webserver after conversion

- Integrate **Saracon** into complex workflows

The post processing is launched directly after each job if enabled but will run parallel to the next job in the background which can be useful if many CPUs are available.

In order that a conversion is able to terminate when all regular jobs have been processed, all post processing has to terminate as well. Sometimes it will become necessary to manually kill the spawned process if the post processing command is formulated in a wrong way.

Note 4 NERD ALERT: *Note that the post-processing feature is quite powerful. On the other hand the first steps are not easy. So do not expect it to be "userfriendly" in the sense that some options can be easily clicked together. Don't whine – you have been warned.*

5.1 Automatic Variables

Saracon makes available to the user most of the source and target file properties in terms of so called automatic variables which are substituted before the command is executed. Automatic variables are very powerful since it will allow the formulation of rather complex commands. Table 5.1 lists all available variables. In case of interleaving, the values of the input related variables are taken from the first file of the input files which are interleaved.

Note 5 *All input related values are lower case, all output related variables have capital letters*

Note 6 *if arguments contain spaces it can become necessary to enclose them in quotes. For instance if a path could contain spaces it should be quoted like this:*

"%P/%F"

5.2 Examples

5.2.1 Create a MP3-Encoded Copy

We often get inquiries to incorporate mp3 encoder support into **Saracon**. This is in preparation but with the post processing option this is almost obsolete since the mp3 encoding process could be formulated as simple as

Conv	Name	Description
	%G	Conversion gain (as float value) in dB
	%S	Postfix
	%r	Input sample rate
	%R	Output sample rate
	%p	Source path
	%P	Destination path
All	%n	Input file without file extension (if interleave or interleave-all the first of all files to be interleaved is selected)
	%f	Input file with file extension
	%N	Output file without file extension
	%F	Output file with file extension
	%x	Input file extension
	%X	Output file extension
	%m	Input format string
	%M	Output format string
	%D	Dither type string
P2P		None
P2D	%U	Modulator type string
D2P		None

Table 1: Post processing variables.

```
lame --preset 192 "%P/%F"
```

This creates a mp3 encoded copy of the target with an average bitrate of 192kb. [LAME] has to be installed on the system of course and its binary has to be on the system's path.

5.2.2 Conversion to FLAC with Preservation of Metadata

The internal conversion of **Saracon** does not transfer custom metadata from wav or aiff files to a flac file. The factory flac encoder however can do this (in a proprietary way, so you'll need this encoder to get the data back). We can now use the post processing feature to preserve for instance BWF metadata for archiving purposes. For this reason we have to set up our conversion and select BWF as output format. The post processing command line would then look as following:

```
sh -c 'flac -f --keep-foreign-metadata
"%P/%F"; rm -f "%P/%F"'
```

In other words: We flac encode **Saracon**'s output file with preservation of its metadata (the -f ignores if the target file already exists) and after the flac conversion we delete **Saracon**'s output file.

5.2.3 Complex Renaming of Output Files

Sometimes it can come in handy to reflect the conversion process in the file name. The post processing feature can be used to name your files uniquely. If you have a file with the name

`Å-Ä-Ö.flac`

the command

```
mv "%P/%F" "%P/%n_%m-%r_to_%M_%R.%X"
```

will name the output to

`Å-Ä-Ö_24bit_96.0kHz-to-16bit_44.1kHz.wav`

.

5.2.4 Parallel Transfer of Tags

We can get even more complicated. The post processing command can be used to transfer tag information to the target file. We could for instance enhance the MP3 encoding example from above with tag transfer in case we have flac files as input files:

```
sh -c 'FLAC="%p/%f";MP3="%P/%F.mp3";metaflac
-export-tags-to=- "$FLAC" | sed
"s/=\\(.*\\)/=\\"\\1\\"/" > tmp.tmp;.
./tmp.tmp;rm tmp.tmp;lame -abr 192 -tt
"$TITLE" -tg "$GENRE" -ty "$DATE" -ta
"$ARTIST" -tl "$ALBUM" -add-id3v2 "%P/%F"
"$MP3";rm "%P/%F"'
```

For this command the [FLAC] tools and [LAME] must be installed on the system. This example works on UNIX based systems only (Mac OSX, GNU/Linux).

5.3 General Notes

5.3.1 A Word on the System-Interface

The command the user can pass to the system through the post processing feature is interpreted by the system directly. Usually this interface is not a shell, so concatenation or piping of commands is not possible, but it is possible to launch a shell from this interface and pass a chain of commands to it as we did in some examples from above.

5.3.2 Debugging Post Processing Commands

Post processing commands can get rather complex and all the escaping syntax requires some interactive debugging possibility. All command line based tools will output their output to the command line from which **Saracon** has been launched. Just launch **Saracon** from the command line (the Terminal application) like this (assuming **Saracon** has been installed in the applications folder):

```
./Applications/Saracon.app/Contents/MacOS/Saracon
```

The post processing command is logged to **Saracon**'s log window from which it could be copied into a terminal/console program for fine tuning.

You can as well output debug information manually by injecting echo commands (`echo %P/%F`) into your post processing command, dump partial results to files for examination (`sed "s/=\\(.*\\)/=\\"\\1\\"/" > tmp.tmp`) or dump temporary files to the console (`cat tmp.tmp` on UNIX based systems, `type tmp.tmp` on Windows).

5.4 Clear Inputs

To speed up certain workflows you may want to clear the list of input files after each conversion. Check the *Input Clear* option in the configuration editor.

6 Super Batch

The *Super Batch* functionality allows the user to group multiple configurations/conversions which are then run all subsequently. With this tool the user can keep his workstation busy the whole night or even weekend without having to set up and run the individual conversions.

To set up a super batch, the user has to prepare several configurations (even multiple of the same mode, one of P2P, P2D, D2P), engage the super batch mode in the main window (check the *Super Batch* check-box), open the super batch editor by clicking on the *Edit* button (or the main menu as if you edit configurations) and add all prepared configuration files to the super batch (use the *Add* button in the super batch editor).

The super batch conversion can then be run as you would run a regular conversion.

7 Log-Window

The log window logs all runtime messages (events, errors, summaries, confirmations, etc.). All log mes-

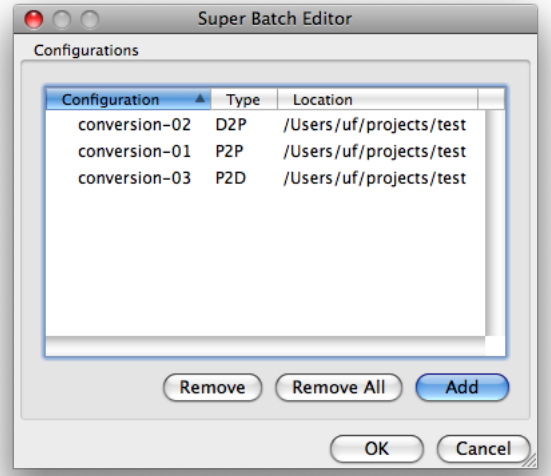

Figure 4: Super batch editor.

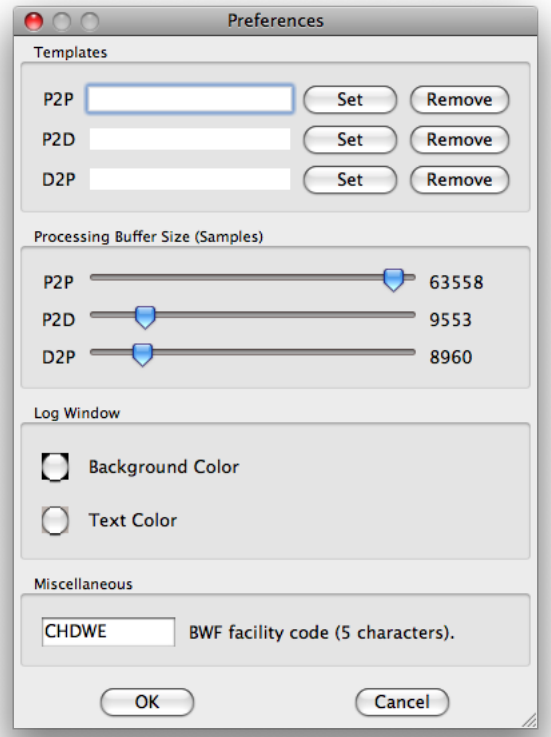

Figure 6: **Saracon**'s preferences dialog.

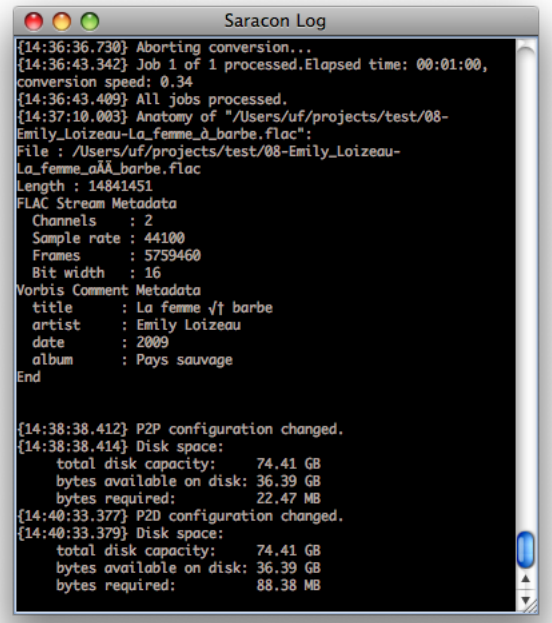

Figure 5: **Saracon**'s log window.

sages are logged to `cout.log` and `cerr.log` located in `/Users/<account>/Library/Application Support/Saracon` at run-time. When reporting bugs it makes sense to attach these files to your report.

The log window can be opened and closed with ⌘+*L* or through the *View::Show Log-Window* main menu item.

8 Drag and Drop

Source files for every conversion mode (P2P, P2D, D2P) can be added to the currently active configuration by dragging the files onto the main window's title bar when the configuration editor is closed. If the input files are not compatible with the current conversion mode **Saracon** tries to find adequate mode.

When the configuration editor is open input files can be dragged to its title bar .

9 Preferences

9.1 Processing Buffer Size

Here you can set the approximate processing buffer size in samples for each channel. Depending on your system adapted adjustments can improve speed.

> **Note 7** *In P2P mode the best buffer sizes to start with are the largest buffer sizes. In P2D and D2P mode start with smaller buffer sizes.*

9.2 Templates

Saracon's preferences support the definition of templates for all types of configurations. The templates will be loaded at application startup as default configurations.

> **Note 8** *Templates enable the user to start with customized configurations each time* **Saracon** *is launched what can eliminate some time-consuming steps setting up similar conversions.*

9.3 Generating/Setting a Template

A template can be any configuration which has been previously saved. Follow this guide to setup you own template:

1. Select your conversion mode (P2P, P2D, D2P).
2. Open the configuration editor.
3. Set the output format options, postfix, output folder. Even the input files can be set (if the input is always the same).
4. Close the configuration editor by clicking *Ok*. Do not confirm if you are asked to correct inconsistencies in your current configuration since you don't want to run a conversion – it's just a template which doesn't have to be consistent.
5. Save this configuration at your preferred template location.
6. Open **Saracon**'s preferences and set this configuration as template for the mode you prepared it for.

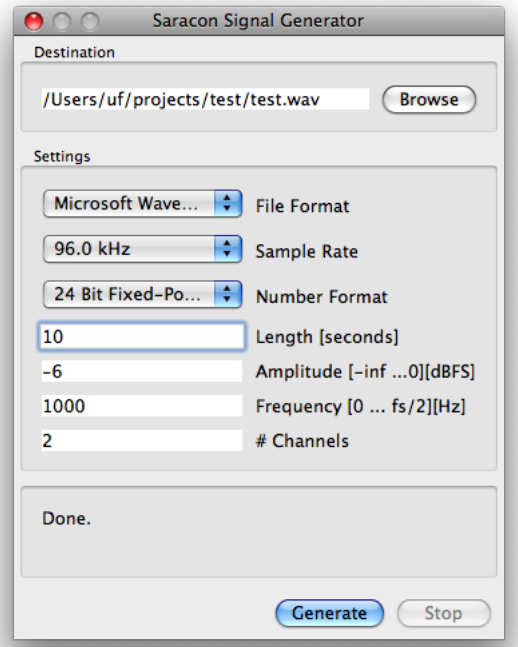

Figure 7: **Saracon**'s Signal generator.

10 Signal Generator

Saracon contains a high precision test signal (single tone sine) generator. In contrast to many other implementations this test signal generator tries to avoid all common digital design pitfalls (quantization/dithering). Test signals in a vast variety of frequencies, amplitudes and formats with an arbitrary number of channels can be generated.

The generator renders its signals in double precision floating-point and uses TPDF-dither (flat dither) when it quantizes the signal to the target number format.

To open the user-interface (shown in figure 7) click the *Tools::Signal Generator* entry in the main menu or use the ⌘+*G* short cut. The signal generator is run by clicking *Generate* and can be canceled by clicking *Stop*.

The signal generator's options:

File Format: Output file format (wav, aiff, au, raw, DSDIFF (♣) and DSF (♣)).

Sample Rate: Sample rate of the test signal.

Number Format: Numeric format of the test signal (irrelevant for DSD output).

Length: Length of the test tone in seconds.

Amplitude: Amplitude of the test signal in dB full scale (dBFS). Zero decibel equals full scale. See the note below.

Frequency: Frequency of the test tone in Hertz.

 # Channels: Number of output channels.

> **Note 9** *The test signal is quantized to the target format using a flat triangular dither. If the test signal's amplitude is set to 0 dB clipping can occur because the dither is added to the internally generated test signal. This is especially important during DSD generation (DSDIFF & DSF output format): To high input levels can cause the modulator to become unstable. Modulator clips/resets are logged in the log window. Therefore the default amplitude is set to -6 dB.*

11 Command Line

Saracon can be operated from the command line. This simplifies the use of **Saracon** in shell scripts for automation purposes. The command line operation can be divided in four fundamental modes

1. Run the signal generator
2. Run a conversion with command line arguments
3. Batch processing of configuration (*.src) files
4. Open the **Saracon** GUI but load the configurations specified on the command line

You can always get help by running

Usage:

```
saracon -h
```

It can happen that this documentation is out of sync with the application, so the command line help is a more accurate source of information.

Options and switches are always in a short and a long form. The short form consists of a single dash and a single letter. If it's an option it is followed by a space which again is followed by the option's value (which must be quoted in case it should contain spaces). Long switches/options start with a double dash and a rather descriptive long name. Long options are directly followed by an equal sign and the value without any spaces.

11.1 Signal Generator

The command line signal generator is run as following

Usage:

```
saracon -s [options]
```

Whereas the `[options]` are explained here:

`-q`	`--s_freq`	Generator frequency in Hz.
`-a`	`--s_amp`	Generator amplitude in dB.
`-l`	`--s_len`	Generator signal length in seconds.
`-o`	`--s_out`	Generator output file name.
`-e`	`--s_rate`	Generator sample rate in Hz. Any when PCM output, one of {2822400, 5644800} when DSD output.
`-m`	`--s_format`	Generator file format, one of {wav, rf64, mat, aiff, raw, au, dff, dsf}.
`-u`	`--s_number`	Generator number format. One of {16bit, 24bit, 32bit, float, double}. PCM only.
`-k`	`--s_channels`	Generator number of channels.

11.2 Conversion

A conversion can be run directly from the command line with

Usage:

```
saracon -c <mode> [options] <source files>
```

Whereas `<mode>` is the conversion mode, i.e. one of `{p2p, p2d, d2p}`. The `<source files>` are the audio files you would like to convert, e.g. `*.wav` or `*.dff`. The `[options]` are explained in the following table

`-r`	`--rate`	Set target sample rate. One of {44100, 48000, 88200, 96000, 176400, 192000, 352800, 384000} for P2P and D2P, one of {2822400, 5644800} for P2D.
`-t`	`--target`	Set target directory.
`-f`	`--format`	Set target file format. One of {wav, aif, rf64, sd2, mat, caf, raw, flac, ogg} for P2P and D2P, one of {dff, dsf} for P2D.
`-n`	`--number`	Set target number format. One of {16bit, 24bit, 32bit, float, double, vorbis}. P2P and D2P only.
`-d`	`--dither`	Set dither. One of {off, tpdf, powr1, powr2, powr3} in P2P and D2P, one of {off, tpdf} in P2D.
`-g`	`--gain`	Set conversion gain in dB.
`-b`	`--batch`	Set conversion batch mode. One of {normal, smart, all, split}.

11.3 Batch-Processing of Configurations

If you prepared several configurations within **Saracon** then you can process them from the command line unsing the following syntax

Usage:

```
saracon -p <configuration files>
```

Saracon then processes each of the specified `*.src` files one after the other.

11.4 Open Configurations

If configurations are passed on the command line without any switch or option, these configurations are loaded in **Saracon**'s GUI as with *File::Open Configuration*.

11.5 Other Options/Switches

11.5.1 Regression Test Mode

Saracon can be set into a regression test mode which ensures that dynamic data (e.g. noise generators and time tags) generate always the same data inbetween multiple program invocations. This is used primarly for testing. The regression test is enabled using the `-R` resp. the `--regression` switch.

11.5.2 Tolerant Mode

When **Saracon** is set to tolerant mode overwrites and collisions are ignored and all command line conversions are processed even if existing files are overwritten or multiple input files generate the same output file. The tolerant mode is enabled with the `-T` resp. the `--tolerant` switch.

11.5.3 Log Message Verbosity

Saracon's command line verbosity can be set with the `-V <level>` resp. `--verbose=<level>` options. `<level>` determines the level of verbosity and can be one of {none, error, warning, all}. *all* enables all output (default), *warning* enables warnings and errors, *error* enables just errors and *none* disables even error printout.

11.5.4 Keep Console

When launching **Saracon** from the console on Windows systems, the console window is hidden, when **Saracon** enters GUI mode. Sometimes it can be useful if the console remains visible, for instance when debugging post processing commands. Use the `-C` resp. the `--keep-console` switch for this very purpose.

11.6 Requirements

To get **Saracon**'s command line working you have to add the path to **Saracon**'s binary to your `PATH` variable. This would be something like

```
/Applications/Saracon.app/Contents/MacOS/
```

11.7 Examples

11.7.1 Signal Generator Example

This example generates ten seconds of 5.6448 MHz DSD (output as DSDIFF file) with a test-sine with an amplitude of -6 dB and a frequency of 1 kHz.

```
>> saracon -s --s_format=dff --s_amp="-6" --s_len=10.0 --s_rate=5644800
{17:30:33.906} Initializing...
{17:30:33.906} Signal generator.
  Frequency:     1000Hz
  Length:        10s
  Amplitude:     -6dBFS
  Sample rate:   5644800Hz
  Format:        DSDIFF
  Target path:   test-signal-(1000.0Hz-5644800Hz-10.0s).dff
{17:30:33.906} Generating...
{17:30:46.187} 10%
{17:30:58.046} 20%
   ...
{17:32:21.890} 90%
{17:32:33.828} 100%
{17:32:33.828} Done.
```

12 Modulation Level History ♣

The *Modulation Level History* is a graphical tool to monitor P2D conversion processes to find the right gain settings for the modulators. It can be enabled in the P2D configuration (see section 3.4.6) and is shown in figure 8.

The history is persistent and can be inspected after completion of the conversion and the conversion's settings can be adapted according to the history. Histories can be saved and loaded later on again.

The modulation level has a reduced resolution to avoid unnecessary amounts of data. A single point in the history corresponds to frame of 65536 DSD samples (bits) which equals 0.02322 seconds.

12.1 Graphs

The following parameters are monitored:

12.1.1 Modulation Level

The maximal modulation level during a frame (for definition of the modulation level please refer to section C). The modulation level is an indicator for the load of the modulator and its instability. The modulation level is displayed in dB SACD. The +3.1dB SACD is the maximum modulation level allowed. When resets are enabled the modulators are reset at modulation levels higher than 3.1dB SACD.

12.1.2 Modulator Input Level

The maximal RMS modulator input level during a frame. The modulator input level is displayed in dB SACD and shows the maximal levels appearing at the input of the modulator after the upsamplers. Different modulators have different maximal allowed peak levels (see Modulators) for their inputs.

> **Note 10** *Since the up-sampling process (prior to the modulation) removes out-of-band signals, higher levels than at the up-sampler-input can appear. Often this is the case when heavily limited/compressed/clipped files are converted, since higher-frequency-"flat top" signals contain out-of-band information.*

12.1.3 Reset Count

The number of resets during a frame. This information is not shown as graph but as bar growing from top to the bottom.

12.2 Window Elements

12.2.1 Channel ID

The *Channel ID* (see figure 9) identifies the **input** channel which is displayed in the corresponding plot.

12.2.2 Time Line

The time line (see figure 9) indicates the time in the *Modulation Level History*. The indicator is placed at every vertical grid line and indicates the time exactly at this position in the form `[minutes]:[seconds]:[milliseconds]`. The resolution depends on the horizontal- (X-) zoom.

12.2.3 Level Indicators

The level indicators and the vertical- (Y-) zoom are arranged such that the top region (0 and 3.1 dBSACD) are always in the plot, since only the peaks are of interest. The level indicators adapt them selves to the vertical- (Y-) zoom.

12.3 Loading Histories

To open an existing modulation level history (e.g. from a prior conversion), click *File::Load History* in the Modulation Level History menu.

12.4 Saving Histories

You can check a check-box in the output tab in the P2D configuration editor to automatically save of the history at the end of a conversion – see section 3.4.6. To save a history manually, go to the Modulation Level History window menu and click *File::Save History As* or *File::Save History*.

12.5 Scrolling

Turning the Mouse-Wheel without any modifier key just scrolls the history. Dragging the scroll bar scrolls as well.

12.6 Horizontal Zoom

To zoom horizontally, keep the *Ctrl*-key pressed and turn the Mouse-Wheel. Turning upwards zooms in, downwards zooms out.

Zooming with the mouse wheel is quantized to larger steps. If fine adjustment is needed, you can

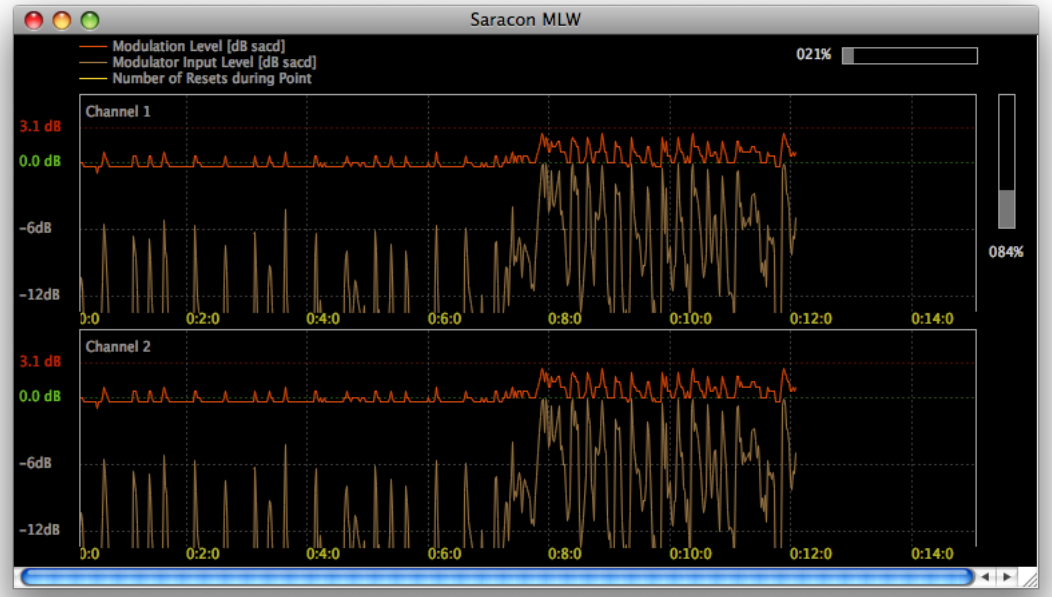

Figure 8: Modulation level history.

click into the *horizontal zoom-bar* (see figure 9) in the plot window and drag the zoom-bar.

12.7 Vertical Zoom

To zoom the amplitude range, keep the *Ctrl* and *Shift*-key pressed and turn the Mouse- Wheel. Turning upwards zooms in, downwards zooms out.

Like the horizontal zoom also the vertical zoom can be fine-adjusted if needed. Click into the vertical zoom-bar – see figure 9 – in the plot window and drag the zoom-bar.

12.8 Graph Control

It is possible to enable and disable each graph independently. To enable/disable a graph click the menu item *View* and the corresponding sub-item. The corresponding sub menu item reflects the current state (you can take a look here to check for instance if the "number of resets graph" was disabled or no resets occurred at all). The graphs can be identified using the graph Legend – see figure 9:

Red: Maximal Modulation Level

Brown: Maximal Input Level

Yellow: Number of Resets

12.9 Finding Maxima

This function searches all graphs of all channels for their maximal values. This function can be found in the menu under *Tools*. From release 1.5 on this dialog supports refresh during conversions (click the *Refresh*-button).

12.10 Hiding the History

The modulation level history can be hidden without loosing the current data until a new conversion has been launched or a history has been loaded from file. To hide a Modulation Level History click *File::Hide Window* in the modulation level history.

12.11 Experience

Using **Saracon**'s P2D conversion in conjunction with the Modulation Level History will require some experience to get the right feeling for the relations between input material, gain, input level, modulation level and resets. The best training is to experiment with a lot of different material, settings. Select some short sources and just start some conversions and take a look at the graphs and try to predict the behavior.

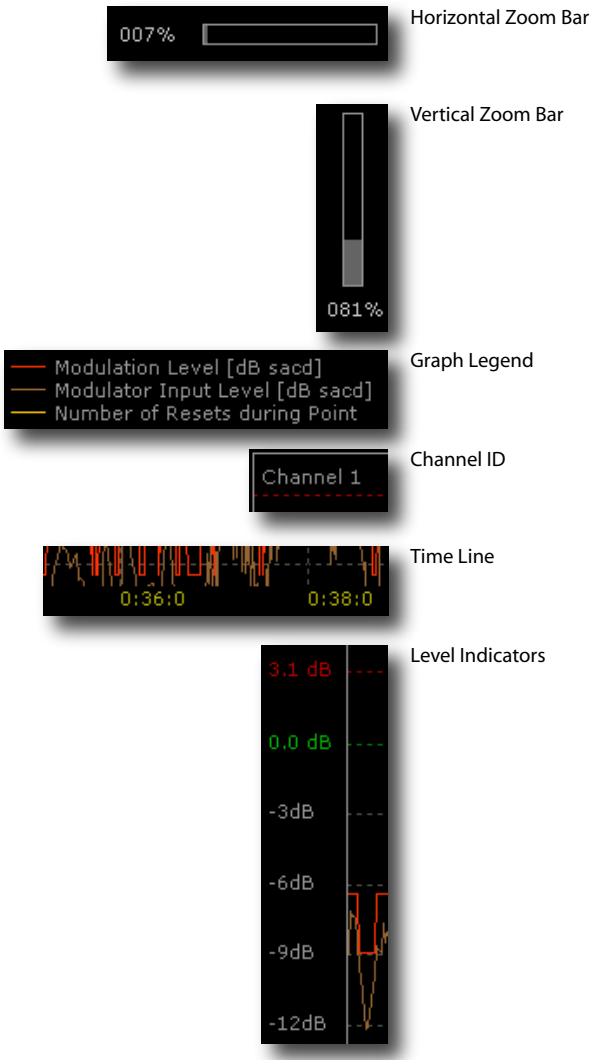

Figure 9: *Modulation Level History* display elements

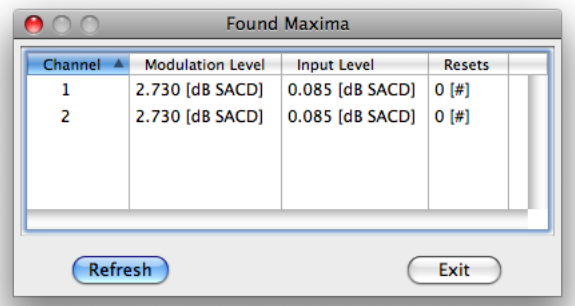

Figure 10: "Find Maxima" dialog.

13 Licensing

To protect the intellectual property of Weiss Engineering LTD and avoid redistribution the **Saracon** is license protected. Two types of protection systems are available (see also sec. A):

license file: based on the Hardware ID of your workstation a license file is provided which contains the **Saracon** license. You will be asked to load this license file by a file dialog when you start **Saracon** after a fresh installation. The license file is not portable to other workstations.

hardware dongle: a tiny USB dongle/stick that contains your personal **Saracon** license and is portable to whatever workstation you want to work on. Note: USB dongle support is no longer supported on macOS (see sec. B.12).

The protection systems support different licenses:

Standard: **Saracon** supports all PCM functionality but no DSD functionality.

DSD: The whole functionality of **Saracon** is at your service.

13.1 Updating Licenses

The license update system of **Saracon** allows change of the licensing mode without installing/un-installing anything and even without restarting the application. This functionality can be accessed from the main-window menu by clicking *Edit::License* and a dialog as shown in figure 11 will appear. Depending on the license protection system you are currently using, the respective other procedures will be grayed out.

13.1.1 Updating License Files

In order to upgrade your **Saracon** license (e.g. from *Standard* to *DSD*) load the newly provided `lic` file by clicking the *Load* button. A downgrade procedure can obviously not be offered with this protection system.

13.1.2 Updating Dongle Licenses

The modification of the licensing information – which is saved in the internal memory of the dongle – requires two steps:

1. Reading the licensing information from the dongle: To generate an updated memory image of your dongle Weiss Engineering needs the former state of the dongle. Therefore the information must be extracted by performing the *Read Dongle*-procedure. The result will be a *customer to vendor* (`c2v`) file which has to be sent to Weiss Engineering LTD (per email or on a floppy using "snail-mail").

2. Burning the updated licensing-information into the dongle: When you received the license-update (a *vendor to customer* (`v2c`) file), you upload it into the dongle by using the *Update Dongle*-procedure. Your new license should become active instantly. In certain cases (i.e. downgrading) a new `c2v`-file will be generated. This file contains the new state of the dongle (including a time-tag) and proves the downgrade. It is located in the same folder as the `v2c`-file and has a ack-postfix.

All data transferred between the dongles/keys and all parties (you and Weiss Engineering LTD) is encrypted and can be decrypted only at our factory or in your dongle and will not be readable by anyone.

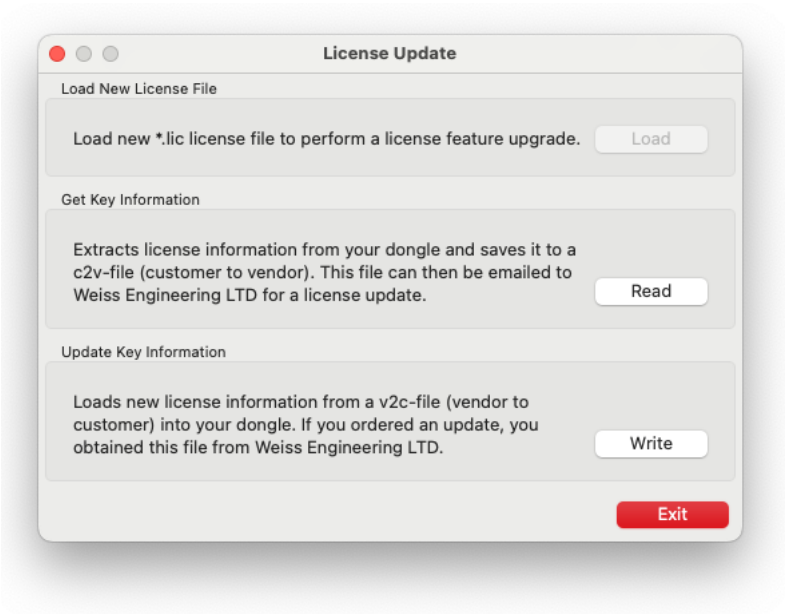

Figure 11: License updating interface.

14 Technical Data

14.1 PCM to PCM Technical Data

14.1.1 Sonic Performance

Table 2 lists the THD+N (Total Harmonic Distortion plus Noise) of all possible PCM to PCM conversions, measured under the following test conditions:

- Not weighted.
- A dithered, 32 bit fixed point, 1kHz test tone converted to a dithered 32 bit fixed-point output.
- 2^{16} samples, not averaged.
- Chebychev window, -300dB side-lobe attenuation.

When converting to 24bit precision data formats (i.e. 24bit fixed-point and 32bit floating-point) and lower (16 and 20 bits) the degradation in comparison to a 24 bit test signal will not be measurable.

Figure 12 shows plots of two conversions[7] with the following properties:

- Coherent double precision test signals
- 2^{14} FFT size
- No window
- 50 times averaged
- THD measured with Chebychev window with 300db side-lobe attenuation
- Output TPDF dithered

14.1.2 PCM to PCM Features

Files and Channels:
Unlimited number of files.
Unlimited number of channels per file.

Input/Output sample rates:
{44.1kHz, 48kHz, 88.2kHz, 96kHz, 176.4kHz, 192kHz, 352.8kHz, 384kHz}

Input/Output numeric formats:
16/20/24/32-bit fixed-point
32/64-bit IEEE floating-point

[7] Plots of all conversions are available. If you like to get a copy, please contact Weiss Engineering.

Input/Output file-formats:
Microsoft-Wave (*.wav),
Sonic Foundry w64 (*.wav),
Audio Interchange File Format (*.aiff),
Sun Audio (*.au),
Broadcast-Wave (BWF),
Sound Designer II (sd2),
Apple Core Audio Files (caf),
Ensoniq PARIS audio file format (paf),
Matlab 4 and 5 files (mat),
.L/.R,
.1/.2,
Header-less raw audio files (*.raw)[8]

Dithering/Re-quantization modes:
Triangular Probability-Density-Function Noise Dithered (TPDF)
Powr 1 (high-pass-shaped noise dither)
Powr 2 (dithered, noise-shaped quantizer)
Powr 3 (dithered, noise-shaped quantizer)

14.2 PCM to DSD Technical Data ♣

14.2.1 Sonic Performance

Table 3 lists the THD+N (Total Harmonic Distortion plus Noise) of PCM to DSD conversions using the CRFB8V2 and CRFB10V2 modulators, measured under the following test conditions:

- Not weighted.
- The input is a dithered, 32 bit fixed point, 1kHz test tone.
- 2^{16} samples, not averaged.
- Chebychev window, -300dB side-lobe attenuation.

Figure 13 shows plots of two conversions using two different modulators.

14.2.2 PCM to DSD Features

Input sample rates:
44.1kHz
48kHz
88.2kHz
96kHz
176.4kHz
192kHz
352.8kHz
384kHz

[8] output only

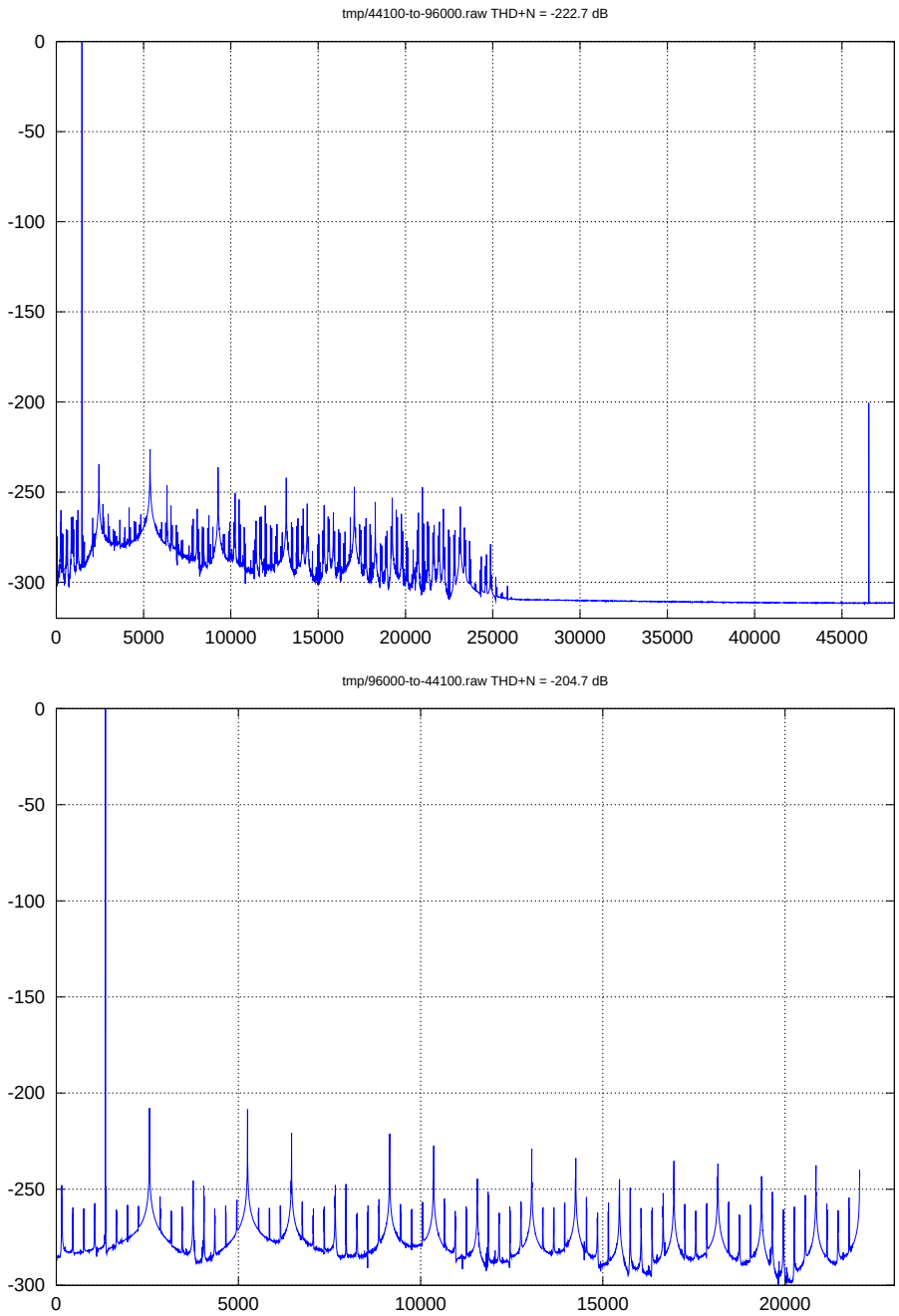

Figure 12: Spectra of PCM to PCM conversions of a coherent 1kHz test tone (sine). **Top:** 44kHz double precision to 96kHz double precision, **Bottom:** 96kHz double precision to 44kHz double precision.

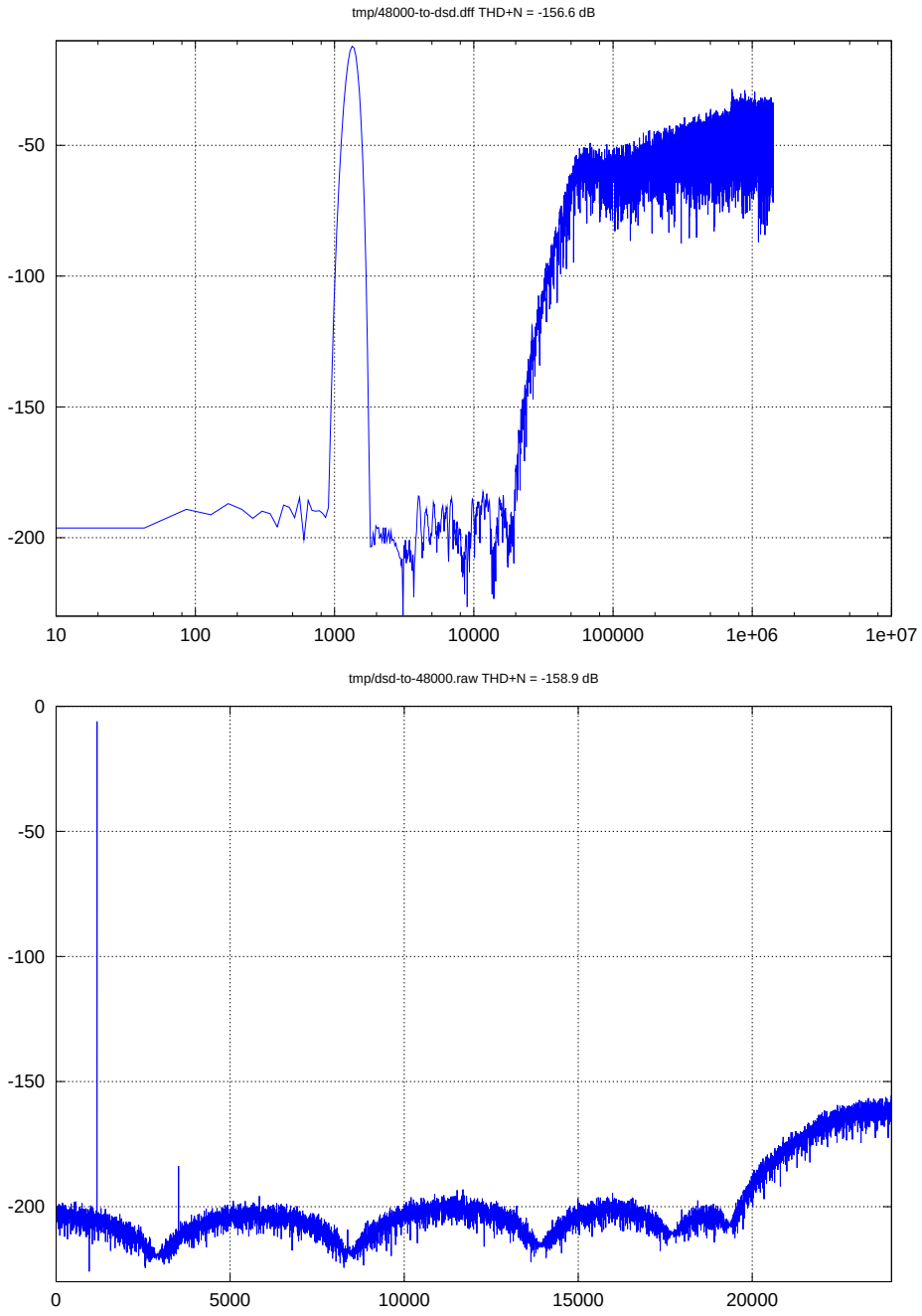

Figure 13: **Top:**PCM to DSD conversion of a 1kHz test tone at a sample rate of 96kHz to DSD (2.8224MHz, 1 bit). **Bottom:**DSD to PCM conversion of a coherent 1kHz DSD test tone (generated from a 352.8kHz source to DSD using Saracon's CRFB10V2 modulator).

↓ From ╲ To →	44.1	48.0	88.2	96.0	176.4	192.0	352.8	384.0
44.1		-189.0	-189.0	-189.2	-190.0	-190.0	-190.4	-191.4
48.0	-188.8		-180.8	-189.5	-189.9	-190.1	-183.6	-190.5
88.2	-191.2	-190.2		-190.1	-190.1	-190.0	-190.5	-190.5
96.0	-190.0	-190.4	-179.6		-190.5	-190.5	-190.3	-190.7
176.4	-190.7	-191.2	-191.3	-187.9		-191.9	-192.0	-192.1
192.0	-190.8	-191.1	-179.6	-183.2	-183.7		-192.4	-192.5
352.8	-191.1	-187.1	-191.9	-188.1	-192.9	-189.2		-193.9
384.0	-191.1	-191.7	-179.7	-192.4	-184.8	-192.4	-185.2	

Table 2: THD+N (in decibel) of all PCM to PCM conversions (input 32bit fixed point, output 32bit fixed point).

Input numeric formats:
16/20/24/32-bit fixed-point
32/64-bit IEEE floating-point

Input file formats:
Microsoft-Wave (*.wav),
Sonic Foundry w64 (*.wav),
Audio Interchange File Format (*.aiff),
Sun Audio (*.au),
Broadcast-Wave (BWF),
Sound Designer II (sd3),
Apple Core Audio Files (caf),
Ensoniq PARIS audio file format (paf),
Matlab 4 and 5 files (mat)

Output file formats:
DSDIFF (*.dff)
DSF (*.dsf)

Upsample filter modes:
Smooth (Slow transition-band and therefore less pre-echo)
No Aliasing

Modulators:

Source	CRFB8V2	CRFB10V2
44.1	-147.2	-159.4
48.0	-147.0	-159.8
88.2	-147.0	-159.6
96.0	-147.2	-159.4
176.4	-147.2	-159.3
192.0	-147.1	-159.5
352.8	-147.4	-159.9
384.0	-147.5	-159.5

Table 3: THD+N (in decibel) of PCM to DSD conversions using interpolating ΔΣ-modulators (CRFB8V2 and CRFB10V2).

Interpolating DS:
CRFB8V2 (8th order IIR noise-shaper without phase distortion).
CRFB10V2 (10th order IIR noise-shaper without phase distortion).

Other:
Dithering option for modulators
Stabilization option for modulators
Monitoring via log-window or modulation-level-history.
Ganging of multi-channel settings
Free input-channel to output-channel routing

14.3 DSD to PCM Technical Data ♣

14.3.1 Sonic Performance

Table 4 lists the THD+N (Total Harmonic Distortion plus Noise) of some DSD to PCM conversions, measured under the following test conditions:

- Not weighted.
- The input is a dithered, 32 bit fixed point, 1kHz coherent test tone converted to DSD with Saracon's *CRFB10V2* modulator.
- 2^{16} samples, 8 times averaged.
- No window.

Figure 13 shows plots of a DSD conversion to 48kHz. The sonic performance

14.3.2 DSD to PCM Features

Input sample rates:
2.8224MHz
5.6448MHz
11.2896MHz

Input file formats:
DSDIFF (*.dff)
DSF (*.dsf)

Output sample rates:
{44.1kHz, 48kHz, 88.2kHz, 96kHz, 176.4kHz, 192kHz, 352.8kHz, 384kHz}

Output numeric formats:
16/20/24/32-bit fixed-point
32/64-bit IEEE floating-point

Output file-formats:
Microsoft-Wave (*.wav),
Sonic Foundry w64 (*.wav),
Audio Interchange File Format (*.aiff),
Sun Audio (*.au),
Broadcast-Wave (BWF),
Sound Designer II (sd2),
Apple Core Audio Files (caf),
Ensoniq PARIS audio file format (paf),
Matlab 4 and 5 files (mat),
Header-less raw audio files (*.raw)[9]

Dithering/Re-quantization modes:
Triangular Probability-Density-Function Noise Dithered (TPDF)
Powr 1 (high-pass-shaped noise dither)
Powr 2 (dithered, noise-shaped quantizer)
Powr 3 (dithered, noise-shaped quantizer)

14.4 Speed Benchmarks

Here are some trivial Benchmarks conducted with the **Saracon** 1.60-14 release on the following three platforms:

A: Win XP 2GHz Core 2 2GB RAM

B: Mac Mini 1.8GHz Core 2 Duo 1GB DDR2 SDRAM

[9] output only

Target	THD+N
44.1	-158.9
48.0	-158.9
88.2	-158.8
96.0	-158.9
176.4	-159.7
192.0	-158.5
352.8	-158.7
384.0	-159.0

Table 4: THD+N (in decibel) of DSD to PCM conversions.

C: Mac Powerbook G4 1.5 GHz 1.5 GB RAM

The buffer sizes – which can be set in src's preferences – were set to:

P2P	51200
P2D	8000
D2P	8000

The results are summarized in tables 5 and 6.

From	To	A	B	C
44.1	176.4	16.0	10.0	4.39
176.4	48.0	10.3	10.8	5.58
176.4	44.1	20.5	10.0	2.66
44.1	192.0	10.3	8.99	3.97
176.4	DSD	2.82	0.95	0.50
DSD	176.4	0.80	0.35	0.10

Table 5: **Saracon** 1.60-13

From	To	A	B	C
44.1	176.4	26.5	25.5	4.79
176.4	48.0	11.3	10.5	4.28
176.4	44.1	34.7	33.8	5.21
44.1	192.0	13.7	14.7	3.96
176.4	DSD	2.60	2.25	0.57
DSD	176.4	1.88	1.80	0.23

Table 6: **Saracon** 1.60-14 beta

Conversion speed values are for 2 channel stereo relative to real time. A value 1.88 means that the conversion runs 1.8 times faster than a real time playback or process. This value will drop of course for more channels and will increase for less channels.

Sometimes a restart of a conversion can result in a faster conversion because the operating system reschedules all newly spawned processes.

14.5 Multiprocessor Support

All major processor manufacturers are currently switching from clock frequency increase to parallel processing by means of replicating the number of processor cores on one chip. This feature can only be exerted by specially adapted software which can distribute the processing load onto several cores. For **Saracon** 1.5 multiprocessor support has been added. This guarantees you an efficient use of new hardware.

14.6 System Requirements

14.6.1 Macintosh

- Intel processor
- OS X 10.9
- 256MB RAM (recommended)
- Screen Resolution of 1024x768 (recommended)
- One USB port for Dongle

14.7 Program Data

14.7.1 Settings

Saracon stores its settings between program invocations in the following file

```
~/Library/Application
Support/Saracon/saracon
```

On UNIX/GNU/Linux machines **Saracon** stores the settings in

```
~/.saracon/saracon
```

.

14.7.2 Log & License Files

The installed license file (`key.lic`) and contents of the log window can be found at

```
~/Library/Application Support/Saracon/
```

whereas `cout.log` contains the regular output and `cerr.log` the error output. On UNIX/GNU/Linux machines **Saracon** stores these files under

```
~/.saracon/
```

.

These log files are written even if **Saracon** suddenly crashes. Please attach these files to any problem/bug report.

A Installation

A.1 Macintosh

Download and open the `Saracon-xx.yy-zz.dmg` disk-image from our website (Saracon Downloads).

A.1.1 Dongle Drivers

If you plan to run **Saracon** with the dongle based license system (sec. 13): run the Mac dongle driver first. It is available as `Sentinel_Runtime.dmg` within the **Saracon** installation image or download it separately from our website (Saracon Downloads). Installing this driver is required only once, even if new **Saracon** versions are released afterwards.

Note 11 *Do not plug the dongle before its driver has been installed. Plugging the dongle without drivers installed can cause problems during the plug-and-play recognition.*

A.1.2 Saracon

Drag `Saracon.app` into your "Applications" folder. If desired, create an alias for fast access from your desktop (⌘-/right-click onto `Saracon.app` in your "Applications" folder → "Make Alias".

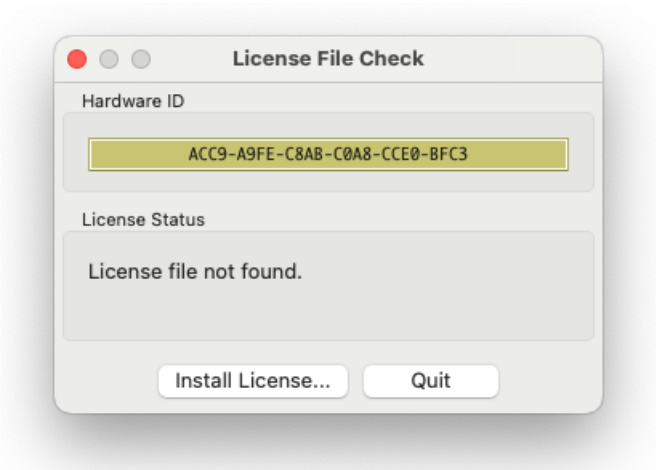

Figure 14: License file dialog.

When invoking **Saracon** you will be asked to install a license file (fig. 14), unless a dongle with a valid license is plugged in.

This license file will be provided to you by Weiss Engineering LTD. Contact us with the Hardware ID that is shown in the file dialog and mention your desired license type. Install the personal license file that is sent to you[10].

I.e. either reinvoke **Saracon** with the dongle plugged in. Or use the license file that was provided to you by Weiss Engineering LTD.

[10] alternatively you may manually copy the license file to the **Saracon** program data directory (sec. 14.7.2) and rename it `key.lic`.

B Release Notes

B.1 1.50

April 2007

B.1.1 Incompatibilities

The 1.5 Release differs from previous versions. In most cases this does not influence the workflow but the configuration files (`*.src`) of previous versions are incompatible to the configurations used in 1.5.

B.1.2 New/Changed Features

Macintosh OS X version:
Saracon is available for Macintosh computers from now on. It features exactly the same functionality as the Windows releases.

Multiprocessor support:
Multi CPU/core support has been added. See section 14.5.

Supported file types:
The number of input (PCM to PCM and PCM to DSD) and output (PCM to PCM) file types has been increased.

Recent configurations:
The 10 most recent configurations are now directly accessible through the main menu. See section 3.

Templates:
The user can now define templates for default configurations loaded at application startup. See section 9.2.

Command line mode:
Saracon now supports a (non GUI) command line mode – see section 11.

Drag and Drop:
Support for adding source files through drag and drop, see section 8.

Split/Merge in P2P
The target can now be re-channelized. The *Split* mode creates one file for each channel in each input file, the *Interleave* mode collapses all channels of all input files into one output file.

Flush Configuration
The currently active configuration can be flushed to default values (empty) from **Saracon**'s main menu (see section 3.1).

Multi-selection of files:
When opening input files in PCM to DSD and PCM to PCM configuration editors now multiple files can be selected at once and an unlimited number of files is supported.

Postfix renaming mode:
The behaviour of the postfix renaming mode for PCM to PCM conversions has been changed such that also a different output folder can be defined and postfixes can be generated automatically..

Find maxima dialog:
The find maxima dialog of the modulation level history has been reworked – see section 12.9.

Log window:
The log window messages have been improved to print out more useful information.

Remove PCM files:
Multiple or all PCM files can now be removed from input section in the P2P configuration editor at once and in a more convenient way.

.1/.2 and .L/.R files
When adding a .1/.2 or .L/.R file all other files with the same filename trunk get added automatically. Furthermore the output file type is preset to the input file type. When in split mode, .1/.2 and .L/.R files can be generated.

"Dither Off" option
Additional to all dither options the output dither can be disabled now (for instance when no sample rate conversion is performed and the output resolution is higher than the input resolution).

Standard Mode
The features of **Saracon** standard are extended to full functionality in P2P mode.

B.2 1.60

September-December 2007

B.2.1 New/Changed Features

D2P Mode:
Saracon supports now conversion from DSD to PCM (D2P).

Custom channel configurations for P2D:
Saracon() supports now arbitrary channel configurations for P2D conversions.

More internationalization on Mac: The internationalization on Mac has been improved (support for uncommon characters in file names etc.).

Improvement for SD2 file support: The SD2 file support has been massively improved.

Support for .1/.2 files: The .1/.2 AIFF files produced by SoundBlade are now supported.

B.3 1.60-13

May 2008

B.3.1 New/Changed Features

Conversion Gain:
The user can now set a conversion gain. It defaults to a save value which should avoid clipping in the quantizer due to sample rate conversion filter ringing. This safety gain can be avoided if you first sample rate convert to a high resolution output format (e.g. 32bit) with enough headroom and normalize the result and then perform the final requantization to your target resolution (e.g. 16 bit).

.2/.4/.5/.6/.7/.8 File Support:
Sonic Studio SoundBlade .3 .. .8 files are supported and handled as multichannel source files (which means that Saracon searches for other channels to add).

Different Source Rates:
Noninterleaving (Split, Normal) batch conversions in P2P with different source sample rates are supported now.

5.6448MHz DSD Support in D2P:
5.6448MHz DSD sample rate is now supported in D2P conversions.

ABSS Chunk in DSDIFF files:
This chunk is now filled with zeros to avoid problems w/ third party tools.

B.3.2 Bug Fixes

Channel Order in P2P Interleaving:
When adding multichannel files (.L/.R) were added to a P2P conversion setup they were added in the wrong order.

Click at End of Conversion:
There was the possibility for a click at the end of the target file if the file was shorter than the conversion buffer size.

Disk space query failure:
Diskspace Query Failed for non ASCII Characters. This prevented certain conversions from running.

BWF Time Reference Transfer:
BWF time references were not transfered properly in P2P conversions.

B.4 1.60-14

December 2008

B.4.1 New/Changed Features

Optimizations:
The natural factor sample rate conversions on Intel platforms have been hand optimized.

Manual:
Finally the manual received its deserved update.

Drag and Drop:
Improved drag and drop behavior. **Saracon** is now asynchronous to the drag source and releases it immediately after adding the files. A little bug in the D2P configuration editor has been fixed.

Large File Handling in D2P:
The handling of large files in D2P conversions has been improved. **Saracon** now complains if the target format is not suitable for a certain amount of data. Furthermore the target disk space check has been added to D2P as well.

B.4.2 Bug Fixes

Multi CPU:
The multiprocessor handling has been improved and some false detections of multi-CPU systems are now history. Because of **Saracon**'s multi threaded architecture the processing load will be distributed on all CPUs available (the quality of this distribution depends only on the capabilities of the operating system).

B.5 1.61-17 (Beta)

September 2009

B.5.1 New/Changed Features

- RF64 file format support
- FLAC support

- Smart interleave batch mode (interleave files based on their file names)
- Batch modes for all conversion types
- 5.6448 P2D support
- New and more intuitive/consistent GUI
- Ogg/Vorbis support (preliminary)
- DSD signal generator
- Multiple signal generators at the same time
- Super batch (a batch of multiple conversions)
- Low order delta sigma modulator for P2D (CRFB-6)
- P2D and D2P can now retain time stamps when using DSDIFF and BWF as IO
- Gain always available even during requantizations.
- Command line operation supports now mostly Saracon's complete functionality
- Native Leopard builds
- Dropped support for OSX 10.4 (Tiger)

B.5.2 Bug Fixes

- GUI lock at small buffer sizes resolved

B.5.3 Incompatibilities

The 1.5 Release differs from previous versions. In most cases this does not influence the workflow but the configuration files (`*.src`) of previous versions are incompatible to the configurations used in 1.5.

B.6 1.61-36

May 2020

B.6.1 New/Changed Features

- Windows 64bit support
- macOS 64bit support
- drop support for OSX 10.8 (Mountain Lion)
- Linux 64bit support (deb, rpm)
- DSF file format support
- upgrade libraries (wxWidgets, libsndfile,...)
- upgrade HASP library & runtime installer
- unicode file paths support
- various bugfixes

B.7 1.61-37

August 2020

B.7.1 Bug Fixes

- fix P2P excessive output file lengths

B.8 1.61-38

November 2023

B.8.1 New/Changed Features

- drop support for macOS $\leq$10.12 (Sierra)
- upgrade libraries (wxWidgets, libsndfile,...)
- update HASP runtime installer

B.9 1.61-39

November 2023

B.9.1 Bug Fixes

- fix P2P & D2P output formats size limitations (e.g. FLAC supports large files)

B.10 1.62-01

April 2024

B.10.1 New/Changed Features

- add protection with Hardware ID based license files

B.11 1.63-01

February 2026

B.11.1 New/Changed Features

- add post conversion input clear option

B.12 1.64-01

September 2026

B.12.1 New/Changed Features

- macOS universal build, supporting ARM64 & Intel x86_64
- drop HASP USB dongle support on macOS, contact Weiss Engineering to receive your Hardware ID based license file

B.13 Known Issues

None at the moment.

C SACD Specifications ♣

This is an excerpt from the Philips Super Audio CD (SACD) System Description (Scarlet-Book), Annex D, Audio Signal Requirements Normative, © by Royal Philips Electronics, March 2003.

C.1 Audio Level measuring condition (D.1)

SACD Audio levels must be measured after a 50 kHz Butterworth 30 dB/Oct low pass filter.

C.2 Zero dB Audio Reference Level (D.2)

The SACD Zero dB Audio Reference Level, referred to as *0dB SACD*, corresponds to a sine wave with a peak amplitude equal to 50% of the theoretical maximum DSD signal level.

C.3 Maximum Audio Peak Level (D.3)

The maximum SACD audio peak level is determined by the maximum allowable DSD modulation level. The DSD Modulation Level is equal to $\frac{\|28-2\cdot N\|}{28}$, where N is the number of bits set to one within any 28 consecutive bits of the DSD stream, and $4 \leq N \leq 24$. The maximum allowed value of the DSD Modulation Level is 20/28. A DSD Modulation Level of 20/28 corresponds to the maximum SACD Audio Peak Level of +3.10 dB SACD. Peak signal levels above +3.10 dB SACD are not allowed.

C.4 High Frequency DSD Signal and Noise Level (D.4)

The accumulated RMS signal + noise level of the DSD signal, measured after a $40kHz$ Butterworth $30dB/Oct$ high pass filter and a $100kHz$ Butterworth $30dB/Oct$ low pass filter, is maximally equal to the RMS level of an input sinewave with a peak amplitude of -20 dB SACD (see D.2). The averaging filter used to calculate the RMS level must be a first order unity gain IIR filter with a coefficient of 1/524288 (2^{-19}), corresponding to an IIR filter with a cutoff frequency of about $0.85Hz$.

D FAQ

Question 1 *Is it possible to convert multi channel projects with* **Saracon** *by processing for instance 5 mono files separately? Can I expect to maintain sample accurate sync if it is done in 5 subsequent processes?*[11]

Answer 1 Sample accurate sync is guaranteed (in the filters and all other structures containing delays). So you don't have to worry when processing surround material with **Saracon** in separate processes.

Question 2 *What format SD2 files need to be in for it to work on a PC?*[12]

Answer 2 The SD2 files are based on the old Macintosh file system which has a so called *resource fork* which keeps file information in addition to the data saved in the *data/main fork*. Since most other file systems (e.g. NTFS or FAT) do not support a resource fork, this information must be saved in a separate file.
Saracon on on Wintel PCs handles this by reading/writing to a file with the same file name as the source/target file but with a `._` prepended. For instance a file called `trio-from-hell-live.sd2` on Macintosh consists of `trio-from-hell-live.sd2` and `._trio-from-hell-live.sd2` on Windows.

Question 3 *How do I up- or downgrade the* **Saracon** *license?*

Answer 3

1. Get into contact with your distributor or Weiss Engineering directly.
2. Weiss Engineering will send you a file which contains the encrypted dongle update data which will be written to the dongle in the subsequent steps.
3. Run **Saracon**.
4. Select from the main menu *Edit::License* to open up the license editor (*see* section 13).
5. Select the *Write* button from the *Update Key Information* section and pass the file which has been send to you by Weiss Engineering.
6. Then click the *Read* button from the *Get Key Information* section, choose a destination (for instance the desktop) and save the new file which has the `c2v` extension.
7. Send the file created during step 6 back to Weiss Engineering that we can verify that the up- or downgrade was successful.

Question 4 *Why are the resultant files "x" samples longer than the originals? There are "y" samples added to the front and "z" samples added to the back.*

Answer 4 Answer 4 The additional samples are because of the FIR filters used in **Saracon**. They add samples to the original file. Of course we could truncate the samples, but it would not be the proper way. All filters have at least a post-ringing ("z"), so they add samples to the end. Linear phase filter have also a pre-ringing ("y"), so they add samples at the beginning as well.

Question 5 *How do I pronounce* Weiss*?*

Answer 5 "W" as in *volume* and "eiss" as *ice*.

Question 6 *What can I do when I'm really happy with this Software?*

Answer 6 You can send the developer a copy of the audio you mastered with it!

Question 7 *From what I understand, when you sample rate conversion there is an upsampling of the material to the least common multiple of the two rates in question. If that's true does not it mean that there is no sonic advantage between converting 88.2 to 44.1 vs. 96 to 44.1?*[13]

[11] Carl Talbot
[12] Jeff Lipton
[13] Barry Wood

Answer 7 Upsampling to the least common multiple is required for rational sample rate conversion only (sample rate conversion with a non integer factor). For integer conversions you can convert directly to the target sample rate. This is valid for both upsampling and downsampling. But: There are many options for those processes. For instance the quality oft the sample rate conversion filters (anti aliasing and anti imaging filters) suffer from extreme rate differences, i.e. they collide with the numerical precision of the CPU. Therefore and for speed improvements it makes sense to split up the conversion in certain cases.

Regarding the sound quality: This depends strongly on the algorithms. You can get the same sound quality in frequency domain (images and aliasing artifacts well below the quantization noise floor) for both. **Saracon** for instance is designed this way. There might be a higher CPU load though. In other products this effect can lead to CPU load optimized designs on which it is possible that 96/44.1 conversions sound different than their 88.2/44.1 equivalents. And: The difference between source and target rate has an influence on the filter cutoff slopes. Therefore upsampling to a higher intermediate rate (rational conversion) has disadvantages compared to a direct (integer) conversion when it comes to time domain effects (filter ringing and preecho) which are not so easy to compensate for.

Question 8 *I have been a user of* **Saracon** *DSD for 3 years. I often convert DSD files to PCM files, in order to mix them in my software (XXX) at a high sample rate (often 32/64 bits and 176.4 khz) and then re-convert them to do a 16/44 CD.*

I have noticed that, when you convert DSD files (5.6 Mhz) to 64 bits float files (at 176.4 khz), the choice of dither disappear: you can't chose any dither.

Is it a bug, or does it mean that you don't need to add dither at this bit rate? Generally speaking, is it better, when you convert 5.6 Mhz DSD files to 32 bits fixed/64 bits float files (at 176.4 khz) to let the files without any dither (off/truncate)?

Indeed, if I add TPDF dither to 176.4 Khz files, for example, I will necessarily have to add another dither to convert the 176.4 Khz to 16 bits - 44 Khz files. So, adding dithering twice will add many extra noise to the original sound. I have also noticed that the sound is more natural and "full-bodied" when you use no dither. Adding dither tends to transform the sound (thinner sound, more treble).

Answer 8 This has nothing todo with the input format but with the output format (float/double): There is no compelling algorithm for dithering floating point output. This is caused by the nature of the floating point number format. There are two (bad) solutions:

1. You could assume the audio level to be at -1./1. bounds and dither with this assumption but with this all other level ranges and dynamic files are excluded (and there for clipped when above these bounds), exactly contrary to the advantages of the floating point format. We once dithered this way but I'm not convinced of this method any more so it had been removed in one of the former releases.

2. The other opportunity is to add a dither which adapts to the signal level. But the you have a very bad decorrelation between signal and dither noise which will degrade the desired effect of the dither.

Conclusion: Since the floating point format uses – due to its nature – always the greatest bit depth for any signal level, the quantization effects are very low. The regular float (32 bit) can hold by default and without bit length reduction 24 bit fixed point signals, double (64 bit) can hold a lot more.

And as a last word of comfort: There are rarely DSD sources with a SNR much above 24 bit (~145 dB) and the theoretical SNR of double (64 bit float) is far beyond the human hearing threshold and when I mean far I'm speaking of decimal powers. If you want dither, than use fixed point output.

References

[LipKoyDither] Lipshitz and Vanderkooy, *Dither Myths and Facts* , AES Preprint 6279, Presented at the 117th Convention, 2004 October 28-31 San Francisco

[BWF] European Broadcast Union, *BWF – a format for audio data files in broadcasting*, Tech 3285 – Technical Specification, Version 1, July 2001

[RF64] European Broadcast Union, *RF64 – An extended File Format for Audio*, A BWF-compatible multichannel file format enabling file sizes to exceed 4 Gbyte, Tech 3306 – Technical Specification, January 2006

[DSDIFFspec] Philips, *Direct Stream Digital Interchange File Format*, Version 1.5, April 2004

[DSFspec] Sony Corp., *DSF File Format Specification*, Version 1.01, November 2005

[LAME] Lame MP3 encoder, http://lame.sourceforge.net

[FLAC] FLAC encoder, http://flac.sourceforge.net

E Contact

Weiss Engineering Ltd.
Florastrasse 42
8610 Uster
Switzerland

Phone: ++41 44 940 20 06
Net: www.weiss.ch
Email: weiss@weiss.ch

Index

⌘
- +B, 10
- +C, 12
- +G, 16
- +L, 14
- +O, 10
- +S, 10
- +Shift
  - +S, 10
- +W, 12
- +X, 12
- +Y, 7

Abort
- Conversions, 12

Agreement, 2
Annex D (SACD Specifications), 36

Benchmarks, 29
Buffer Size, 16

Channel ID, 20
Command Line, 17
- Signal Generator, 17

Configuration, 7
- Editor, 7
- Flushing, 10
- Load Blank, 10
- Loading, 10
- Opening, 10
- Saving, 10
- Saving As, 10

Configurations
- Recently Used, 10

Conversion, 12
- Abort, 12
- Pause, 12
- Progress, 12
- Resume, 12
- Run, 12
- Speed, 12
- Status, 12
- Sub, 12
- Suspend, 12

Conversion Mode, 7
Conversion Type, 7
Conversion:Job, 12
Copyright, 2

D2P
- Technical Data, 28

Disable Graphs, 21
Dongle, 23
- Update, 23

Drag and Drop, 15
DSD Generator, 16
DSD to PCM
- Technical Data, 28

DSD to PCM Features, 28
DSD Version, 23

Editions and Manual, 7
Elapsed Time, 12
Enable Graphs, 21

FAQ, 37
FindMaxima, 21
FLAC, 13
Frequently Asked Questions, 37

Graph Legend, 21

Hide
- Log Window, 14

History, 20
Horizontal Zoom, 20

Installation, 31
- Macintosh, 31

Introduction, 7

Job, 12

Key, 23

Legend
- Graph, 21

Level Indicators, 20
License, 23
- DSD, 23
- Standard, 23
- Update, 23

Load
- Configuration, 10
- Modulation Level History, 20
- v2c, 23

Log & License File Locations, 30
Log Window, 14
- Hide, 14
- Show, 14

Metering, 20
Mode, 7
- DSD to PCM, 7
- PCM to DSD, 7

PCM to PCM, 7
Selecting, 7
Modulation Level History, 20
Channel ID, 20
Disable Graphs, 21
Enable Graphs, 21
FindMaxima, 21
Graph Legend, 21
Level Indicators, 20
Loading Histories, 20
Saving Histories, 20
Time Line, 20
Zoom X, 20
Zoom Y, 21
MP3, 13
Multiprocessor Support, 29

P2D
History, 20
Technical Data, 25
P2P
Technical Data, 25
Pause
Conversions, 12
PCM to DSD
Technical Data, 25
PCM to DSD Features, 25
PCM to PCM
Technical Data, 25
PCM to PCM Features, 25
Post Processing, 12
Examples, 13
FLAC, 13
Renaming Output Files, 14
Tag Transfer, 14
Variables, 13
Preface, 7
Preferences, 16
Processing Buffer Size, 16
Program Data, 30
Progress, 12

Recently Used
Configurations, 10
Remaining Time, 12
Renaming Output Files, 14
Resume
Conversions, 12
Run
Conversion, 12

SACD Specifications (Annex D), 36
Save
c2v, 23
Configuration, 10
Modulation Level History, 20
Settings File Location, 30
Show
Log Window, 14
Signal Generator, 16
Command Line, 17
Sine Generator, 16
Standard Version, 23
Start
Conversions, 12
Status, 12
Sub-Conversions, 12
Super Batch, 14
Suspend
Conversions, 12
System Requirements, 30

Tag Transfer, 14
Technical Data, 25
Time
Elapsed, 12
Remaining, 12
Time Line, 20

Update
License, 23
Upgrade
License, 23

Vertical Zoom, 21

Zoom X, 20
Zoom Y, 21

Notes: