

***TOBYBEAR PRODUCTIONS***  
***IS PROUD TO PRESENT***

# ***MIDIBAG***

**Disclaimer:**

All information in this manual is given at my best knowledge. I cannot gurantee that this plugin is working on every system under every condition. If you encounter a problem, please contact me.  
Features & specifications subject to change without notice

© 2002-2004 Tobybear Productions  
All Rights Reserved.

## **Introduction**

Welcome to MidiBag, a VST MIDI plugin collection from Tobybear!

This collection contains *Arpimedes*, *CC-Control*, *Chordator*, *Harmonisator*, *Humanisator*, *MetroGnome*, *MicroTuner*, *MidiMirror*, *N2CC*, *PeakFreak* and *QuicKeys*.

### **Compatibility issues:**

#### *1. Operating systems:*

This software has been tested under Microsoft Windows 95, 98, ME, 2000 and XP and worked without problems. Should you encounter an OS-related problem, please contact me!

***Note concerning Macintosh version: I don't have a Mac myself, nor do I know someone nearby who has one, nor do I know how to program on a Mac, so it's very improbable that a Mac version of these plugins will come out. Sorry!***

#### *2. Host software:*

I have tested this extensively with "Steinberg Cubase SX 2.0" and "Plogue Bidule", two of the few VST hosts out there that actually support VST MIDI plugins. If YOU encounter any problems relating to these plugins or can't get it to work in your favorite VST host software, don't hesitate to contact me under [tobybear@web.de](mailto:tobybear@web.de) (full contact details at the end of this document).

### **Installation:**

Just extract the archive and copy the DLL files into your preferred VST plugins directory (mostly something like "c:\programs\steinberg\vstplugins" when working with Cubase VST). I decided not to include an installer yet as this would significantly enlarge the archive.

## **General information**

All plugins in this bundle (except for Harmonisator) are VST MIDI plugins, which means they don't do anything to incoming audio and do not output audio at all. Instead, they receive and send VST MIDI information. Unfortunately, not every VST host supports these kind of plugins, the best ones are currently Steinberg Cubase SX 2.0/Nuendo 2.0 ([www.steinberg.net](http://www.steinberg.net)) and Plogue Bidule ([www.plogue.com](http://www.plogue.com)), but it should also work with EnergyXT ([www.webmassiva.com](http://www.webmassiva.com)) and FruityLoops ([www.fruityloops.com](http://www.fruityloops.com)) as well as some other hosts.

Since most hosts cannot handle plugins which do not have any audio inputs or outputs, each plugin has a standard stereo in/stereo out configuration, even though the audio part will simply pass through.

**Note:** These are not MFX MIDI plugins (which for example Cubase can also handle), but true VST plugins! Cubase also has true MIDI plugins (Arpache and others) available as MIDI channel inserts,

Here is an example for setting up VST MIDI plugins in Cubase SX 2.0, in other hosts it might be more complicated (or way easier, eg. in Bidule :-)):

1. Load the desired VST MIDI plugin as an audio insert effect on any channel you want. It does not matter where you put it, as the audio will simply pass through, this just has to be done so that Cubase automatically adds the plugin's MIDI ports to the internal pool of ports. In this example, let's add "Arpimedes".
2. Go to a MIDI track with the input data for the VST MIDI plugin, for example a MIDI track with several chords. In the arrange window, you can choose the MIDI output destination for this track. There should now be a new MIDI port available "Arpimedes". Make this the active port to send data to it.
3. Go to a MIDI track that is supposed to receive the data that the plugin generates. This can of course also be a VST instrument track. Click on the "e" (for edit) box in the arrange window on that track. In the window that now opens up, you can set the MIDI input for that track. Choose your VST MIDI plugin here as well, eg. "Arpimedes".
4. You now have a routing as follows: the MIDI track with the chords gets sent to Arpimedes, who breaks the chords into MIDI arpeggios and sends them to the desired MIDI output port or VST instrument.

Some hosts can only send MIDI data to VST instruments, so each plugin in this bundle has two registry files coming with it, named "xxxxx as FX.reg" and "xxxxx as Instrument.reg".

If you double-click on the FX setting, the plugin identifies itself as a VST effect, if you double-click on the instrument setting, the plugin will show up in the VST instrument list of the host.

In hosts like Plogue Bidule, this does of course not matter, as it treats plugins in a very flexible way.

## Arpimedes



**Arpimedes** is a funky little VST arpeggiator plugin with adjustable pattern rhythm and speed. You can define the number of octaves it should use and the step length of the current pattern. Of course several different arp modes are included. And you can even sync the arpeggiator to your host sequencer's temp

**speed:** controls the speed of the arpeggiated notes, can either be set in milliseconds, or, if **sync** button is on, as multiples of the host tempo.

**octaves:** controls how many octaves the arpeggio uses

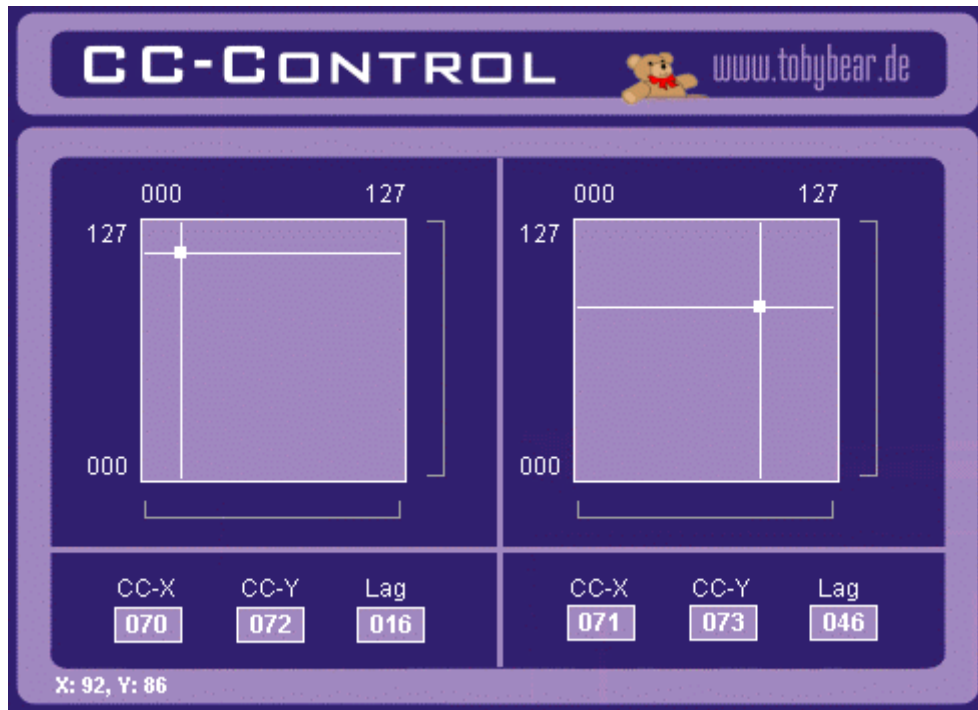
**steps:** how many steps the pattern (see below) has

**pattern:** can be of a length between 1 and 8, each position can be on or off  
If an LED at the position is on, the note will play, if it is off, a pause will be played and the note will be played at the next "on" position. This allows creation of rhythmical arpeggiator patterns.

**mode:** this sets the arpeggiator mode. Here is a detailed description of each mode:

- asplayed: the notes arpeggiate in the order they were held down
- up: notes are sorted from lowest to highest key
- down: notes are sorted from highest to lowest key
- updown1: notes go from lowest to highest, then from highest to lowest
- downup1: notes go from highest to lowest, then from lowest to highest
- updown2: notes go from lowest over highest to lowest (highest note only played once)
- downup2: notes go from highest over lowest to highest (lowest note only played once)
- random1: note order is randomized each time a new note is pressed
- random2: note order is randomized after each note that is played

## CC-Control



**CC-Control** consists of two X-Y control pads that can send user-defined MIDI CCs based on your mouse actions.

Move the mouse around on one of the X-Y pads and hold down the left mouse button to generate the appropriate values for X and Y. The MIDI CC number for each of the two axes is set below in the **CC-X** and **CC-Y** fields. The **Lag** field defines the "travelling time", i.e. how long it takes to reach the point where you just clicked from the current grid position. Set Lag to zero to deactivate it.

Note: If you have a high lag value, you can still set the pad position immediately by right-clicking on the desired position instead of left-clicking.

The numbers to the left and on top of the pads define the value range for the X- and Y-axis, which means the minimum and maximum values between which the values are generated. Standard is 0..127, meaning that the desired MIDI CC will have full range values.

You can also drag the grey bars on the side of the pads to adjust only one direction while keeping the other constant.

## Chordator



**Chordator** turns incoming notes into full chords of up to 6 notes. Now you can play bombastic power chords and complex harmonies with just one key press.

Usage is very simple, just set the desired note transpose, then click on the little boxes below it to switch this transpose on or off.

An example for a major chord would therefore be 0, 4, 7, for an octave doubling 0, -12. There is also a sample fxb-bank included with some presets by Matias.

## Harmonisator



**Harmonisator** is a plugin that "harmonizes" an audio stream according to MIDI notes. You feed audio into the plugin, hold down a chord on your keyboard and the audio data is duplicated and pitched according to the chord.

It is definitely not a high-class harmonizing algorithm, but useful for creating choirs quickly or for manipulating drum loops in strange ways.

**Attack** and **Release** control the fade in and outs of the generated streams once a key is pressed.

**Gain** sets the gain factor that is applied to the whole audio output of the plugin.

**Root** sets the root note – when you send this note to the plugin, it plays in original pitch.

**Mode** defines if each channel should be processed separately (stereo mode) or if the stereo channels should be mixed down to one mono channel and then be processed (mono mode).

**Amp** mode changes the way the amplitude of the output is calculated depending on the number of notes pressed (and therefore the number of streams generated): "one" means the amplitude of the channel merely adds up, "max" means that the amplitude is divided by the number of currently active streams and finally "all" means that the amplitude is divided by the maximum number of streams (currently 5). Just try them out and choose the mode that works best for you.

## Humanisator



**Humanisator** is a plugin that works as some kind of "anti-quantizer", bringing more human feel into sequenced notes: You can define how much and in what range Humanisator should alter the timing, velocity, pitch and modulation of the MIDI stream, allowing very natural sounding and interesting phrases to be created easily!

You can use your left and right mouse buttons on the sliders in this plugin to set the minimum and maximum values. If the maximum (set with the right mouse button) is below the minimum (set with left button) then the two areas outside the designated range will be used (sort of like a band-reject filter), else the normal range from minimum to maximum is used. The currently selected range(s) will be shown in white.

## MetroGnome



**MetroGnome** is a little metronome plugin that plays 2 WAV samples, the "hi" sound on each first beat, the "lo" sound on all the other beats of the time signature.

You can of course exchange the WAV samples, define your own tempo and time signature and can even send the timing information back to the host to sync other plugins (although only Plogue Bidule supports this at the moment)

**Installation:**

Please copy the two WAV files into the same directory as the DLL so that the plugin can find them. You can of course replace them with your own samples! You can even create different metronomes by copying and renaming the DLL for example to "blabla.dll", then the two wave files it will be looking for would be named "blabla\_lo.wav" and "blabla\_hi.wav".

You will need a fully VST2 compliant host to use this in "external sync" mode (see below).

**Controls:**

**volume:** controls the volume of the metronome WAV files, 0 is silence, 100 is original volume, 200 is twice the original volume.

**bpm:** sets the desired tempo, only works in "internal sync" mode (see below)

**sync mode:** if set to "external", the plugin will retrieve tempo and timing information from the host and play back the metronome at that speed, the bpm setting will be ineffective then.

If set to "internal", the plugin will generate its own tempo and play the metronome at the speed set in the bpm box.

**measure:** set the time signature in numerator in denominator. This is used to trigger the appropriate "hi" and "lo" samples.

**send:** if set to yes, timing code will also be sent out as a VST timeinfo event. This is yet unsupported by the VST SDK and the only host who seems to support this is Plogue Bidule (which is a really great program by the way). You can use this to sync other devices/rewire slaves to your tempo.

Use the "**play**" and "**stop**" controls to start/stop playing.

As an experimental feature, there is a rough tempo-tap included by clicking on the gnome face repeatedly – the program tries to identify the bpm and sets the tempo accordingly. This is however not working too well and should only be regarded as an additional gimmick at the moment.

## **MicroTuner**



**MicroTuner** is yet another revolutionary plugin: It adds microtuning capabilities to every VSTi and external synthesizer module, as long as the target synth supports receiving pitchbend on all MIDI channels. I am not talking about changing a note some cents up or down in pitch, no, you can completely redefine the whole tuning scale, allowing you to set an arbitrary frequency separately for each MIDI note! And it gets even better, it already has a loader built in (with drag and drop GUI) that can load the widely used Scala, VAZ and Anamark tuning files directly!

This plugin does currently NOT have any on-screen controls for modifying anything, just two buttons: one for loading a tuning file, the other to reset to the standard tuning.

You can generate special tunings for example with the Scala freeware program or just write down the frequency for every note in an Anamark/VAZ tuning file (see included examples).

Developers might also take a look at my homepage ([www.tobybear.de](http://www.tobybear.de)) under "developers – snippets" for some code how to implement micro-tuning in their own projects.

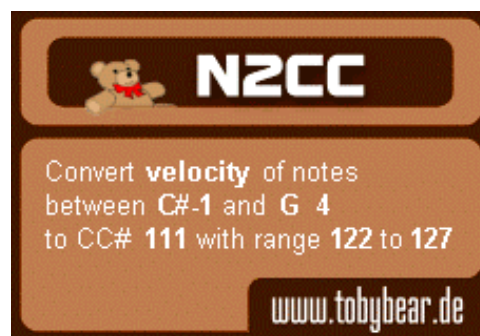


## MIDI-Mirror



**MIDI-Mirror** does what it says: it mirrors all incoming MIDI notes relative to one (users-definable) "mirror note" on the MIDI scale. Just play your favorite melody through this plugin and you will be surprised how it sounds now :-)

## N2CC



**N2CC** converts the velocity or pitch of MIDI notes in a certain range to MIDI CC controller values within a predefined range. This allows you to create some unconventional MIDI effects very easily.

## PeakFreak



**PeakFreak** is an effect/MIDI plugin that generates MIDI notes from the peaks of an incoming audio signal. More specific, if the audio input falls into a certain frequency and amplitude range, a MIDI note (or controller data) is generated. So you can use this for triggering MIDI devices with the audio input. But is NOT a plugin that converts audio frequencies to MIDI notes of appropriate pitches, sort of like melody detection! It works best with hosts that can assign where the MIDI output of one plugin should be routed, like Cubase SX or Plogue Bidule.

Most important thing is the square in the middle of the screen. The purple sliders to the left and right set the amplitude range, the left one being the lowest amplitude and the right one being the highest. So you could for example say, only audio input in the range of 20% and 70% should be relevant for triggering.

The purple sliders above and below the square set the lowest and highest frequency to be used in triggering. Of course the frequency values should not be regarded as the ultimate cutoff frequency since the filters used do of course have a certain steepness, but they should work for this purpose.

*Note:* you can make the lower amplitude higher than the high amplitude setting and also set the lower filter border higher as the high filter border: then the settings are inversed, for example:

normal: xxxxxxL-----Hxxxxxx

inverted: -----HxxxxxxL-----

('x' means don't pass, '-' means pass through, 'L' the lower setting, 'H' the higher setting)

Which could give you even four fields, for example 'Do only trigger if the amplitude of the incoming signal is higher than 80% **or** below 10% and the frequency lies below 100 Hz **or** above 10000 Hz'. The dark yellow area of the square gives you visual feedback of the currently selected range(s). If the audio input falls into this range, that area of the square turns from dark yellow to bright yellow.

The first yellow slider on the left of the interface (with the little 'G' below it) is used for controlling the input gain. Normally it is set to 100% which means no change. You can set it to anything from 0% to 200%, meaning attenuation and amplification of the original signal is possible.

The VU meter next to this slider show the amplified/attenuated input visually. The highest peak is shown by a thin purple line and always rests a little bit longer on the display so you can check the current range of your audio input.

The second yellow slider (with a little 'R' below it) is for controlling the release rate. This does basically mean how long the generated note should last. Everytime the audio input is in the specified range(s), a note of that length is generated.

The CH (channel) setting is for telling the plugin which MIDI channel it should use for the MIDI output. Choose a channel from 1 to 16.

The NOTE (note/CC) field sets the note to be generated, or, if you press the little '<>' field next to the note box, what MIDI CC Controller number is to be generated.

The VEL (velocity) field sets the velocity of the generated note (0-127). By pressing the '<>' field next to the velocity box, you can switch velocity mapping on or off.

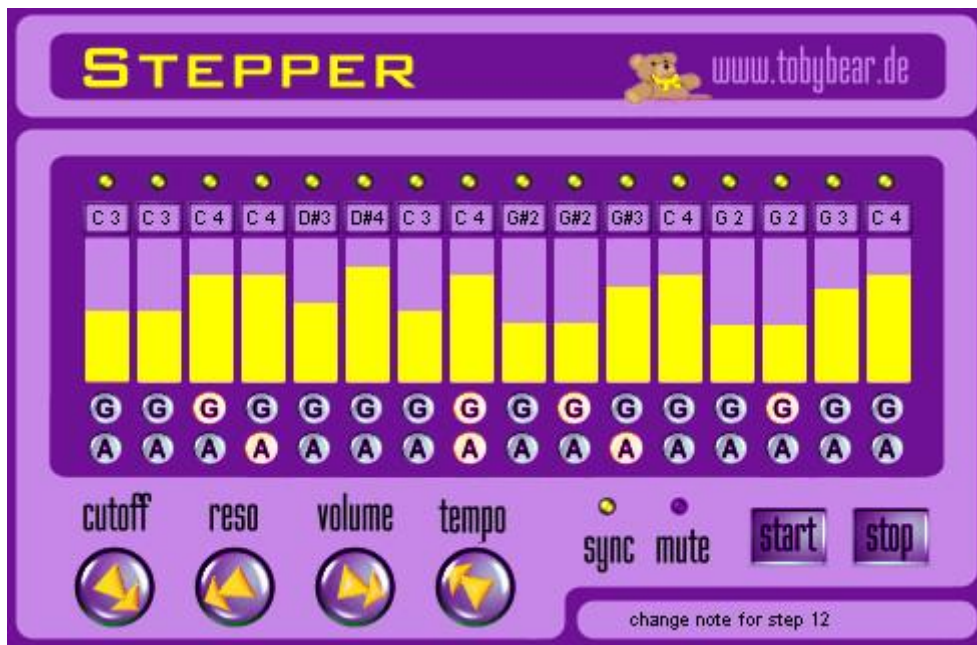
When velocity mapping is on, the velocity field changes to VELMIN, so that you can now set the minimum velocity. The peak between the lowest and highest amplitude range is then mapped to values from VELMIN to 127.

An example:

The velocity setting is VEL=120 and current amplitude settings are: Low=40%, High=90%, which means if the amplitude of the incoming signal is between 40% and 90%, a MIDI note of velocity 120 is generated. But if velocity mapping is on and VELMIN is still 120, then 40% is regarded as velocity 120, 90% is regarded as 127 and every peak inbetween is calculated (mapped) to the appropriate range between 120 and 127. Sounds confusing? :-) Either try it out for yourself or ask me for a better explanation by mail.

And finally the OUT setting determines whether the original (unchanged) signal should be send to the plugin output ('Orig.'), only the peak/trigger signal ('Proc.') or complete silence ('Mute'). The first option is good if you want to do further processing with your audio, for example put another PeakFreak plugin with different settings in the next plugin slot. The processed signal is good for checking what is actually 'left' after filtering the ranges and the muted signal is perfect if you just want to trigger MIDI and do not need the actual audio signal any more.

## Stepper



**Stepper** is a little step sequencer that outputs MIDI notes periodically in 16 steps. You can of course synchronize playback to the host tempo and output 2 additional CCs to control for example cutoff and resonance. External transposing by MIDI notes is also possible.

## QuicKeys



**QuicKeys** is a plugin that does not produce any sound by itself. It is rather intended to be used as a piano GUI to control other VSTis within your sequencer: You load it up as a regular VSTi or as an effect, then click on any track in the arrange window that is routed to another synth/VSTi and now you can play this synth by clicking on QuicKeys' virtual keyboard (or with your own computer keyboard). MIDI processing/converting options are included as well as sending CCs, a MIDI LFO and an auto-chord feature

Most important: Clicking on the left or right wooden panel activates **PANIC**, resetting the device which might sometimes be helpful :-)

QuicKeys allows you to generate MIDI note and CC data with your mouse or your computer keyboard. You have a virtual **4-octave keyboard** on screen and you can generate MIDI notes easily by clicking the appropriate keys. The MIDI data is sent out via the VST interface to the VST host.

You can change the base octave by clicking on the number inside the '**OCT:**' field below the keyboard (left-click increases, right-click decreases).

Next to the 'OCT:' field is the '**CH:**' field which simply sets the MIDI output channel for the generated data.

'**VEL:**' is also almost self-explaining, it is the note velocity of the notes you generate by clicking on the keys (NOTE: you can switch to a different behaviour with the 'velosens' option that will be mentioned later).

And finally, the fields next to the **up/down-arrows** and **left/right-arrows** allow you to select a CC (MIDI Controller) number that is to be generated when you left-click on a key of the virtual keyboard and then move your mouse (while still pressing down the left mouse button) horizontally or vertically. A setting of '000' means that no CC value in this direction is generated. You can right-click on the numbers to enter the CC value manually on your keyboard.

If you use the **right mouse button** instead of the left one for playing notes, CC generation is switched off and instead the note under the mouse cursor is played immediately without having to release the mouse button first. Just try this out to see what I mean.

Below the keyboard in the lower left corner is a little button that switched **chord- and note detection** on and off. If it is on, QuickKeys tries to guess the chord being played (which comes from the VST MIDI input or from the internal chord maker) and displays it right next to this button. There are usually one or two possibilities presented, but of course for more complex chords, there is no definite chord display (which is also highly dependent upon the key the song is in). Single notes are also displayed in the field.

Clicking on the '**Options**' button above the keyboard opens up the 'Options' page – Aha!

Here you find the **chordmaker**, which adds additional transposed notes relative to the original MIDI note on the input. You can have up to five additional notes. Click (left to increase, right to decrease) on the numbers to set how many semitones above the entered note the additional note should be, then click on the button above that field to make it active. The range goes from one octave below to one octave above the base note (-12...+12).

The **transpose** option (click on the brown button next to the label to switch it on) allows you to transpose the incoming MIDI data stream in semitones, the range is here from -24 to +24 which are 4 octaves.

The **mirror** option allows you to mirror the incoming MIDI data stream at a certain note, creating different melodic lines from the original MIDI input. The number corresponds to the MIDI note number that should be used as the mirror point. Again, press the brown box to activate this option.

The **keyboard** switch activates the connection between your computer keyboard and the virtual keyboard. NOTE: Not all hosts support this as they might intercept the keystrokes and use it themselves!

The assignment between notes and keys:

Note:	C	C#	D	D#	E	F	F#	G	G#	A	A#	B	C	C#	D
Key:	y	s	x	d	c	v	g	b	h	n	j	m	,	l	.

The **velosens** switch activates velocity sensitive note generation. If this is active and you click with your mouse on a key of the virtual keyboard, the velocity will be mapped to a range between 0 and the given velocity within the 'VEL:' field, dependent upon the position of the mouse cursor (top of the key=0 velocity, bottom area of the key=maximal velocity).

And finally you have a **MIDI CC LFO** on board :-)

If it is switched on, it generates sinusoidal values in the range of '**Depth**' and the speed of '**Rate**' and outputs this value as MIDI controller data for the controller number set in the '**CC**' field.

## **Thanks & Credits**

Thanks to all my beta testers on [www.kvr-vst.com](http://www.kvr-vst.com) for extensive testing, suggestions and bug reports! Without you guys, many bugs would still be in here!

Very special thanks go to **Frederic Vanmol** (In-Line Software/FruityLoops), who created/translated the VST SDK to Delphi and helped out with some bugs. If you want to program VST plugins or ASIO applications in Delphi, check out his site: [www.axiworld.be](http://www.axiworld.be)

Thanks for trying out my little toy! I hope you have fun working/playing with these plugins!

*Tobias Fleischer alias Tobybear*

### ***TOBYBEAR PRODUCTIONS is:***



**Well, actually it is just \*me\* :-)**

Main Coding, Project Coordination, Manual, Interface Design, Presets, Testing

#### *Tobybear Productions:*

Tobias Fleischer  
Prisnitzweg 7  
D-82538 Geretsried  
Germany

e-mail: [tobybear@web.de](mailto:tobybear@web.de)

web: <http://www.tobybear.de>

#### *Copyright information:*

© 2002-2004 Tobybear Productions. All Rights Reserved.

**VST is a trademark of Steinberg Hard- und Software GmbH.**

All other product names and any trademarks mentioned are used for identification purposes only and are copyrights of their respective holders.