
Richmond Sound Design Ltd.
Theatre Sound Design, Show Control & Virtual Sound System Software

SoundMan-Server Release Notes

Version 1.0.62
2010-07-08

- 1) The "Syncstart" information message will now show a repeat time of -1 instead of 0 when there is no repeat point set. This makes it easier to tell when looking at a log if the track will loop.
- 2) When a track loops, there is a log message to show that the track looped. This also makes analyzing logs easier. Note that the log messages are queued by the playback code, since they can't afford the time to make a log message at the time the actual loop occurs. So the 'looped' log messages may have a timestamp that is a number of milliseconds after the actual loop happened. In particular, when a group of tracks is playing in unison, the individual tracks will show 'looped' messages, and they will probably all have slightly different timestamps. This is NOT an indication that the tracks are out of sync.
- 3) In the last version, a group of tracks might loop even though it was only supposed to play once. This is fixed.
- 4) There were a couple of long-standing bugs that could cause crosspoints to have random amounts of delay in relation to each other. For a long time this wasn't noticeable because the crosspoint delays were disabled by default. However, several releases back they were changed to be enabled by default since ABShowMaker has no way to control crosspoint delay enable. Especially in installations with large ASIO buffer sizes, this made the occasional random delay quite noticeable.
- 5) Setting a delay on a timecode generator did not work properly. The signal would be delayed after the generator started, but the signal stopped instantly when the generator stopped, rather than continuing for the delay period as it should have.
- 6) Several new GET requests have been added to get information about a loaded or playing track. All of these get static information about the file that is loaded onto the channel. If no file is loaded, the attributes that return strings will return empty strings, and the attributes that return numbers will return zero.

//	TITLE	# TrackTitle	chanid="title"
//	DISPNAME	# TrackDispName	chanid="name"
//	SUBJECT	# TrackSubject	chanid="name"
//	COPYRIGHT	# TrackCopyright	chanid="name"
//	COMMENT	# TrackComment	chanid="name"
//	SAMPLERATE	# TrackSR	chanid=fpt
//	TRACKS	# TrackTracks	chanid=number

GET CHAN channel TRACK TITLE will get the title information from the file if there is any title data present. Otherwise it will return the file name without the path or "wav" suffix.

GET CHAN channel TRACK DISPNAME will get the display name attribute from the file, if it exists.

GET CHAN channel TRACK SUBJECT will get the subject attribute from the file, if it exists.

GET CHAN channel TRACK COPYRIGHT will get the copyright attribute from the file, if it exists.

GET CHAN channel TRACK COMMENT will get the comment attribute from the file, if it exists.

GET CHAN channel TRACK SAMPLERATE will get the sample rate for the file.

GET CHAN channel TRACK TRACKS will get the number of tracks or channels in the file.

For all of the GET commands above, the response will be in the form of a line starting with the keyword shown above (for instance TrackTitle) followed by one or more instances of <channel id>=<value>. A typical GET and the response might be:

```
GET CHAN P0-P1 TRACK TRACKS
TrackTracks P0=1 P1=4
```

- 7) The GET GENERATOR and SET GENERATOR commands to get and set signal generator parameters will now accept a valid signal generator channel number like IN 1000. Previously only the channel number was valid. This should eliminate some amount of confusion in using these commands. The bare number format is still valid.
- 8) GET TCGEN START TC now returns the starting time value and offset as timecode values rather than floating-point numbers as was erroneously done before. The same values can still be obtained as floating-point numbers by using GET TCGEN START TIME. Previously both commands returned the data in the same format.

```
//      START TIME                # StartTime=fpt At fpt
//      START TC                   # StartTC=tc At tc
```

The above lines show the response format for the GET TCGEN START TIME and GET TCGEN START TC commands.

- 9) The number of SMPTE timecode readers, SMPTE timecode generators, and signal generators are now controlled by the license info in the dongle. Previously there were always two of each created, regardless of the info in the dongle. Current license keys will default to two of each so that current users will not see a change from this.
- 10) The syntax of the GET VU command has been changed slightly to allow monitoring either pre or post VU on any control node.

The new syntax is:

```
GET VU [PRE|POST] <channel list>
```

If neither PRE nor POST is given (the old syntax form is used) then POST VU readings will be returned as is normal. If POST is given the same post VU readings will be returned. If PRE is given then PRE VU readings will be taken before the control point has an effect on the signal. This might be most useful for input channels.

When switching from pre to post VU or vice versa, the accumulating VU averages are cleared, so the first reading after changing will return zero for the average and peak values for all channels. It is advisable to NOT switch between pre and post VU readings frequently in an attempt to get both values from a single channel. It will not work well. It is best to only switch on a user request.

- 11) A new SET VU command has been added. Getting VU values uses quite a lot of processor time. If you got VU values for the inputs at one time but now no longer need these values, it would be best to turn off the VU accumulation code on those inputs. This is even more important when getting VU readings on crosspoints since there are so many of them.

```
SET VU [PRE|POST] channel-spec ON|OFF
```

You can specify whether you want to turn on or off pre or post VU processing for the range of channels. Only one type can be on at any time, so if you had pre VU on and now turn on post VU processing, the pre VU processing will be turned off automatically.

If you do not specify PRE or POST and use ON, the command will return an error, since only one type can be on at once. If you do not specify PRE or POST and give OFF, all VU processing will be turned off for the range of channels.

- 12) A "GET GROUP" command has been implemented to get information about groups.

```
//      GROUP group-spec          #
//      NAME                      # Name          chanid=nonename
//      CHANS|CHANNELS            # Channels      chanid=chanidnone
//      TCREADER                  # TcReader     chanid=chanidnone
```

GET GROUP channel-list NAME will get the name of the group.
This is the same as GET CHAN channel-list NAME.

GET GROUP channel-list CHANS or CHANNELS will get a list of the channels assigned to the group. Note that the response is the group channel id, an equal sign, and then a space separated list of the individual channel ids. The next group, if more than one was requested, will have the group id and an equal sign before its list of channels. If a group has no channels assigned, the first and only channel name will be "none".

GET GROUP channel-list TCREADER will get the channel ids for the timecode readers assigned to the channels in the group. If a group does not have a timecode reader assigned to it (as is often the case) the reported channel id is "none" (without the quotes). The response has the channel id of each

requested group and the matching channel id (or "none") for the reader, following the equal sign.

- 13) The SET GROUP group-id TC READER <reader-id> command will now allow either a numeric channel number for the timecode reader as before, or a full channel specification for the reader such as OUT 1000 or a channel name.
- 14) The routines that looked up channel names would fail to find named "special" channels like signal generators and timecode readers.
- 15) Added a GET FILE command to let you interrogate attributes of an audio file such as the number of tracks and the sample rate, plus some of the internal data items.

```
//      GET FILE "filename"          # FileInfo "name"
//      SAMPLERATE                    #      SampleRate=number
//      LENGTH                         #      Length=number - time in seconds
//      TRACKS                         #      Tracks=number
//      FORMAT                         #      Format=WAVE|AIFF|RAW
//      TITLE                          #      Title="string"
//      COPYRIGHT                      #      Copyright="string"
//      COMMENT                        #      Comment="string"
//      SUBJECT                        #      Subject="string"
//      DISPLAYNAME                    #      DisplayName="string"
```

The filename to be interrogated follows GET FILE, and should be quoted. Following the file name are one or more attribute names to be interrogated. If the file can be found and opened, the response will begin with FileInfo followed by the quoted name of the file, and then the attribute key words followed by equal signs and the attribute values.

- 16) Added a GET TCREADER command to let you get information about a timecode reader.

```
//      GET TCREADER channel          # TcReader chan
//      CODE|SHAPE                     #      Shape=Sine
//                                     #      SMPTE_30
//                                     #      SMPTE_30_Drop
//                                     #      SMPTE_25
//                                     #      SMPTE_24
//                                     #      IRIG_A_Pulse
//                                     #      IRIG_A_Pulse_Inverted
//                                     #      IRIG_A_Mod
//                                     #      IRIG_A_Mod_Inverted
//                                     #      IRIG_B_Pulse
//                                     #      IRIG_B_Pulse_Inverted
//                                     #      IRIG_B_Mod
//                                     #      IRIG_B_Mod_Inverted
//                                     #      Unknown
//      POSITION                         #      Position=fpt      in samples
//      TIME                           #      Time=fpt        in seconds
//      TC                             #      TC=timecode   in timecode
//      FREEWHEEL                      #      FreewheelCount=num
//                                     #      same as GET FREEWHEEL
//      DATE                           #      Date=yy/mm/dd
//      DAY                            #      Day=number
```

```
//          USERBITS          #      Userbits=8_hex
//          SPEED              #      Speed=fpt
//          RUNNINGIPLAYING    #      Running=YESINO
```

Only one timecode reader can be interrogated with a single command. You can get the current shape of the timecode being read, if it is recognized. If timecode has stopped, the code shown will be the last code recognized.

POSITION, TIME, and TC all return the same information in different formats, that is, the current timecode value being read (or the last timecode value read if timecode has stopped).

FREEWHEEL returns the current freewheel count.

DATE will return the same thing as USERBITS for a SMPTE decoder. It will return the date (usually a Julian date) in the IRIG code for an IRIG decoder.

DAY will return 0 for a SMPTE decoder and the Julian day value from the code for an IRIG decoder.

- 17) The ASIO driver input and output buffers are now being cleared when the ASIO device is opened. The drivers are supposed to clear their own output buffers, but not all drivers were doing this. This resulted in nasty noise for an instant when first starting.
- 18) Added code to keep the UI from hanging completely while polling for the dongle at startup. This should keep SM-M from wanting to take a dump because it believes (correctly) that the interface is hung.
- 19) Found and fixed several small problems with delay fades that occurred at intermediate rates of speed. There were separate problems on the output channels and the crosspoints.
- 20) Looking for the dongle on startup takes 20 seconds if there is no dongle (demo mode startup). The interface used to be hung during this time, resulting in nasty messages in the log in SoundMan-Monitor, and possibly even requests from SM-M to take a program dump to analyze the hang. The interface will no longer be hung while looking for the dongle, so SM-M will not log nasty messages and try to take dumps.

Version 1.0.61

2010-06-21

- 1) It is now possible to loop time on a timecode generator. A timecode generator looks like a playback channel. So you set the looping parameters just as you would for a track. For instance, to loop the first timecode generator from 10 seconds back to 0 seconds, you can enter:

```
SET CHAN P1000 TRACK STOP TIME 10 REPEAT TIME 0;
```

You can also set the initial start time for the timecode generator with the TRACK START TIME parameter.

- 2) You can now enter a normal timecode generator channel specification like P1000 for the SET TCGEN command. Previously you had to enter the bare number "1000" to do this. You can still enter a bare number, or no specifier at all, which will default to the first timecode generator. This was fixed in the previous release for GET TCGEN.
- 3) Changed the "timecode reader freewheel timeout" message to be an Information message rather than a Warning message.
- 4) Added a command to hide (minimize) or show the main UI window. On startup the main interface and the splash screen can be suppressed by adding the /NOSPLASH parameter to the command line starting SoundMan-Server. However, if for some reason that is inconvenient, the window can be made to hide in the tray by sending CONFIG SET WINDOW MINIMIZED. The window can be made visible again by sending CONFIG SET WINDOW VISIBLE.
- 5) Now log when the user requests display of the ASIO driver control panel.
- 6) No longer stop ASIO processing when getting an ASIO reset request from the driver. This should in most cases eliminate problems with audio disappearing after looking at the control panel for a driver and adjusting it, at least with decently written drivers. If a driver attempts to reconfigure itself on the fly before we do a complete reset (in violation of the ASIO spec) then there is a slightly better chance that we will immediately crash.

Since the driver may have internally stopped processing audio, it is still necessary to close and re-open the driver as soon as possible to allow the new driver configuration to take effect.
- 7) Now show a nasty tray popup balloon for 30 seconds when we get an ASIO reset request, describing what needs to be done.
- 8) Made a small change that may cure the problem of having random delays on playback at startup.
- 9) GET CHAN list TRACK POSITION will now return the position values as an integral number of samples. Previously this would return a floating-point number with less accuracy in most cases.

- 10) When SM-A is started minimized the tray menu default will now correctly be the show the window, not to hide it.
- 11) Fixed a crash that could occur when closing SM-S while audio was playing.
- 12) A Syncstop message will no longer be logged is a STOP command is issued to a channel that is already stopped.
- 13) Clearing a resume point on a channel that doesn't have a track loaded will no longer create a warning message in the log.
- 14) If you muted the crosspoints the output VU would hang at the current value rather than properly decaying to zero. This is now fixed.
- 15) There is a new LOG command to let the user add comments to the text output from SM-S. This can be useful for documenting problems or other reasons. The log records are timestamped and included in sequence in the history.

```
//      LOG
//      USER          "text";          // U: text
//      INFOINFORMATION "text";          // I: text
//      WARNIWARNING   "text";          // W: text
```

The LOG command is followed by a keyword that indicates the kind of log entry to make. All log entries begin with a date and time stamp, followed by a single letter and a colon that identifies the type of the log record. The character will be one of the following:

C	Command	The text of a command to SM-S
R	Response	The text of the response to a command
D	Debug	A debug output message, not common in release software
I	Information	A line of information
U	User	A LOG command marked "USER"
W	Warning	A line indicating an unusual or error condition

After the word indicating the type of log entry to make is the text of the log entry. This can be a single string in quotes, or it can be a series of individual words, ending in a semicolon or the end of the input line.

Log entries in no way change how SM-S works or what it is doing. All they do is allow the user to make their own entries in the captured log data.

Version 1.0.60
2010-05-25

- 1) The GET TCGENERATOR command with no parameters other than the timecode generator number returned a useless response. It will now be equivalent to GET TCGEN <number> TIME and return the current generated time from the generator. Note the time is returned as a floating point number of seconds and not as a timecode. To convert the fractional seconds to frames you can get the code parameter from the timecode generator, which will tell you the frames per second, and a multiply by the number of frames per second will convert the time to frames.
- 2) If a wave file had an incorrect format and showed the whole file length as the RIFF chunk length it would not be parsed. This formatting error is now handled quietly and the file will load correctly if there are no other formatting errors.
- 3) If wave file header parsing failed, the file load would succeed, but the file would show a length of 0. The load will now correctly fail with a useful error message, making it more obvious that something is wrong.
- 4) The GET TCGEN command will now accept a channel number in the normal format of "P1000" rather than requiring a bare number of "1000".
- 5) A parsing error in GET TCGEN that caused many parameters to be ignored has been fixed.
- 6) Garbage syntax on the end of a GET command was ignored rather than returning an error as it should have done.
- 7) The timecode generators are now initialized to generate SMPTE at 30fps by default. Previously they defaulted to generating 1KHz sine wave tones, which was confusing.

Version 1.0.59

2010-05-17

- 1) Track start, stop, and repeat times could be entered that were negative. For the stop and repeat times this would have disabled the stop or repeat point, but for start times, this would often result in a crash when starting to play. Negative times are now properly detected as invalid and an error message will result.
- 2) Startup was hanging with the AudioScience Cobranet driver. Changed the order of some code to work around it.
- 3) In some cases with many channels we could fail to get memory we needed but not notice the failure. Now if we can't allocate all of the channels we will return an error.
- 4) Added a SIXFOREIGHT option to CONFIG SET INTERFACE to skip the last two channels out of each group of 8 channels. This is used by SM-A in an unusual Cobranet configuration where each bundle only has six channels.

Version 1.0.58
2010-04-24

- 1) A code bug caused the timecode reader to freewheel forever if timecode was lost. This is now fixed.
- 2) Warning and information log messages have been added to show when the TC reader loses timecode, sees a constant repeated (stopped) timecode value, and when timecode again begins moving. All of these messages include the timecode value and reader number.
- 3) A new command has been added to set the freewheel count for a timecode reader. The freewheel count defaults to 5 frames, and can be set anywhere from 0 to 500 frames, or to INFINITE.

The new command syntax is:

SET CHAN chan FREEWHEEL n | INFINITE

- 4) A new command has been added to get the freewheel count value for a timecode reader:

GET CHAN chan FREEWHEEL

The returned value is either

	Freewheel	On=count
or	Freewheel	On=INFINITE

Where "On" is "O" for OUTPUT followed by the reader channel number, and the count is the number of freewheel frames, 0 to 500.

- 5) The SMPTE timecode reader was accidentally leaving a debug log file on the C drive. This has been removed.

Version 1.0.57

2010-04-14

- 1) If the tray notification icon did not create correctly we would exit the program. This was unnecessary and has been removed.
- 2) It seems that creating the icon could fail on Windows Embedded if SMS was started immediately on system power up. SM-S will now retry for up to 20 seconds to create the tray icon. This seems to get around the timeout response being returned by Windows.
- 3) SM-S was not correctly parsing wave file headers that contained invalid or unexpected items in a LIST chunk. This resulted in the file being seen as zero size, so it wouldn't play.

Version 1.0.56
2010-04-05

- 1) Updated the dongle code to match changes in SM-A.
- 2) Now check for up to 20 seconds for the dongle being present when starting. Previously if SM-S was auto-started from the startup folder on boot-up and auto-logon, there was a timing window that could occasionally result in not finding the dongle even though it was present.
- 3) A really annoying VU meter interaction with SM-A that caused the VU meters to flash has been fixed.
- 4) Fixed the copyright date to 2010 in the splash screen.
- 5) There could be a delay on some of the playback channels, even though checking the delay time for the channel would return zero. Setting the delay to zero would clear this phantom delay. This problem should now be fixed, and no phantom delays should be possible.
- 6) Found and fixed a small bug in the code that sets a cross-mute on other input channels when one input channel is soloed.

Version 1.0.55
2010-03-18

- 1) There was a timing window in the dongle code that could result in a deadlock on a multiprocessor machine.
- 2) Updated the dongle code to match changes in SM-A.

Version 1.0.54
2010-03-02

- 1) Disabled the "TCP received XXX bytes" debug message that was putting unneeded noise into the log file.
- 2) There is no longer a "beep" sound when the tray icons for various status messages and warnings are displayed. This can prevent bad sounds coming out of speakers if Windows is sharing the show audio hardware.
- 3) Fixed a timing window that could cause crashes on some machines.
- 4) Track start/stop/repeat/loop points were incorrectly being set using the interface sample rate rather than the file sample rate. When playing 44100 wave files using a 48K interface this could result in the times being considerably off from what was actually requested.
- 5) Setting a repeat point that was before the start point on the track would not loop correctly back to the desired point, the track just stopped at the end. This is fixed.
- 6) The fix for early starts broke normal late starts. That is now fixed.
- 7) If you tried to resume a track that had finished playing you would get hung waiting for a track to go ready that was never going to go ready.

Version 1.0.53
2010-02-04

- 1) Updated the copyright to 2010.
- 2) Changed things so that all 64 output channels are now available from SoundMan-Assistant.
- 3) Fixed a problem in the timecode generator that was preventing it from generating an offset time value when locked to an incoming timecode stream.

When setting up the command sequence to lock an offset generator to timecode the order of the commands is important. The following sequence will generate a timecode offset by 1 hour from the incoming timecode:

```
set tcgen 1001 code smpte30
set group g0 chans p1001 tc reader 1000 lock time 0;
set tcgen 1001 start tc 1:0:0:0 at time 0;
```

The first command sets the code of the timecode to be generated. This needs to be done before the generator needs to generate anything, so is best done before the generator channel is added to the timecode reader group.

The second command adds the generator (channel P1001) to a timecode reader group, selects reader 1000 (channel OUT 1000), and sets up to lock to a timecode chase beginning at time 0 (that is, no hour offset).

The third command **MUST** be after the SET GROUP command or it will be ignored. It tells the timecode generator to generate time starting at 1 hour when it has an incoming timecode of zero hours. Thus the generated time will be one hour ahead of the time read.

If **ONLY** the timecode generator is locked to the timecode reader (just regenerating timecode) then this set of commands can be simplified:

```
set tcgen 1001 code smpte30
set group g0 chans p1001 tc reader 1000 lock tc 1:0:0:0 at time 0;
```

This will have the same effect as the previous set of commands **IF** only the tc generator is locked to the reader. However, if audio files are also locked to the reader, this will have the undesirable effect of starting the audio files 1 hour into playback, which will probably be past the end of the files.

To keep from having audio offset problems the initial command sequence is strongly recommended when offset timecode is needed.

Version 1.0.52

2009-11-11

- 1) Externally requested dumps were not being taken. They now are.
- 2) A popup message after an externally requested dump or a UI timeout would cause SM-S to hang until the user OKed the popup. This was a bad idea, and the popup will now only occur if a serious fault occurs in SM-S, not from a timeout or other recoverable problem.
- 3) The "SoundMan Dumps" directory will be compressed if it isn't already.
- 4) Updated copyright on splash screen to 2009, which should have been done about a year ago.
- 5) Modified the TCP receive logic to hopefully be more robust in the face of possible errors. Added logging of error conditions that would otherwise have gone unreported.
- 6) The debug interface is now the same width as the SM-A debug interface. This makes things look better if both windows are visible.
- 7) Fixed a problem in the code that checked at startup if the dump file directory was compressed.
- 8) Added code to check periodically if there is any data waiting on the TCP port and process the data. This may help if we are getting lost interrupts on input processing.
- 9) Corrected a misformatted TCP error message.
- 10) Added code to try to keep from blocking the user interface when waiting to send a response to the TCP command requestor.

Version 1.0.51
2009-10-18

- 1) Fixed a severe meter layout problem when starting in iconic mode. Meters laid out correctly if SM-S was started with the window displayed.

Version 1.0.50
2009-10-13

- 1) Removed some old unused dialog templates to simplify code.
- 2) Fixed a problem with VU meter layout that could cause a crash on exit.
- 3) Control input and output may now be entered by either TCP port 20000 or UDP port 20000. When using UDP multiple commands separated by a carriage return and linefeed ("\r\n", or in hex 0D0A) can be given in a single packet. The first command in the packet starts at the front of the packet. A command can not be split across packets. The maximum packet size is 1500 bytes.
- 4) The command line entry window in the SM-S window no longer requires a semicolon to recognize the input. A semi-colon still ends an input line but it is not necessary. A carriage return will also work. Previously carriage returns were ignored.

Version 1.0.49
2009-10-01

- 1) The routines to calculate the machine serial number were not the same in ShowMan, SM-A, SM-S, and the E-Show tools. This release has consistent routines that will produce the same serial number.

Version 1.0.48
2009-09-23

- 1) If there were more than 32 output channels, every time you opened the ASIO device the window would get taller by one or more rows of VU meters. Now the window is correctly sized to the VU meters.
- 2) The SM-S window now originally comes up without the blank space for a row of VU meters. The correct space will be added when an interface is opened. When the interface is closed all of the extra space will go away again.
- 3) Added debugging code to check for smashed memory around the areas of opening and closing the ASIO drivers. Several ASIO drivers have been found that seem to have memory corruption problems.
- 4) Some drivers could produce errors that would result in the watchdog thread killing itself when it was trying to kill an overtime callback thread from the driver. This would hang SM-S. This has been fixed.
- 5) Fixed a usually benign memory corruption problem that could occur when using an ASIO device that uses 24 bit sample format. One byte past the end of the output buffer would be corrupted.
- 6) Now only display the last two digits of the channel number over the VU meters. Previously with channels of 100 and over the number was centered and difficult to read.
- 7) Previously it was difficult to get more than 64 channels of VU meters to be seen, and almost impossible to get more than 96 visible, even on a very large desktop screen layout. The VU meters will now shrink vertically as needed to insure that they fit in a reasonable amount of screen space.
- 8) Fixed a memory leak that occurred when exiting the program and using multiple ASIO drivers in an ASIOWGROUP statement.
- 9) Fixed some problems with allocating and deallocating asiogroups. While the old code worked in most cases, some drivers would fail to close correctly and would not reopen after being closed.
- 10) Fixed several problems with asiogroups possibly getting locked up while processing audio. In extreme cases this could require a PC reboot to clear the device and let it make audio again.
- 11) The processor and disk usage monitors could blank when they reached a value of 100%, rather than simply showing a full bar. This has been fixed.
- 12) Holding the shift key down while clicking the X (close) gadget in the upper right of the title bar will cause the program to exit rather than to minimize. Clicking the X without holding the shift key will minimize in the normal way.
- 13) There was an error in the calculation of the machine serial number that

could keep temp dongle files from working.

Version 1.0.47
2009-08-31

- 1) Changed some code in the temp dongle file reading logic to correct some minor errors that were preventing temp dongle files from working correctly.
- 2) Added code to work around ASIO drivers that provide a bad preferred buffer size value, but do give reasonable minimum and maximum buffer sizes.
- 3) Additional info on the DROPSPEED timecode reader parameter added in the previous release:

Setting DROPSPEED results in the time values from the timecode reader being multiplied by 1.001. This corrects the speed error, but it also results in shifting the time value from the reader. For instance, when the reader reads 01:00:00:00 (one hour) it will actually report 01:00:03:18 at 30fps. If you have audio tracks that start at specific time offsets, you may need to slightly shift the start time value to compensate for this.

This chart shows the desired time in hours and the shifted time value for various times and framerates.

Desired	24 FPS	25 FPS	30 FPS
00:00:00:00	00:00:00:00	00:00:00:00	00:00:00:00
01:00:00:00	01:00:03:14	01:00:03:15	01:00:03:18
02:00:00:00	02:00:07:05	02:00:07:05	02:00:07:06
03:00:00:00	03:00:10:19	03:00:10:20	03:00:10:24
04:00:00:00	04:00:14:10	04:00:14:10	04:00:14:12
05:00:00:00	05:00:18:00	05:00:18:00	05:00:18:00
06:00:00:00	06:00:21:14	06:00:21:15	06:00:21:18
07:00:00:00	07:00:25:05	07:00:25:05	07:00:25:06
08:00:00:00	08:00:28:19	08:00:28:20	08:00:28:24
09:00:00:00	09:00:32:10	09:00:32:10	09:00:32:12
10:00:00:00	10:00:36:00	10:00:36:00	10:00:36:00
11:00:00:00	11:00:39:14	11:00:39:15	11:00:39:18
12:00:00:00	12:00:43:05	12:00:43:05	12:00:43:06
13:00:00:00	13:00:46:19	13:00:46:20	13:00:46:24
14:00:00:00	14:00:50:10	14:00:50:10	14:00:50:12
15:00:00:00	15:00:54:00	15:00:54:00	15:00:54:00
16:00:00:00	16:00:57:14	16:00:57:15	16:00:57:18
17:00:00:00	17:01:01:05	17:01:01:05	17:01:01:06
18:00:00:00	18:01:04:19	18:01:04:20	18:01:04:24
19:00:00:00	19:01:08:10	19:01:08:10	19:01:08:12
20:00:00:00	20:01:12:00	20:01:12:00	20:01:12:00
21:00:00:00	21:01:15:14	21:01:15:15	21:01:15:18
22:00:00:00	22:01:19:05	22:01:19:05	22:01:19:06
23:00:00:00	23:01:22:19	23:01:22:20	23:01:22:24

Version 1.0.46
2009-07-31

- 1) If the number of output channels is a multiple of 16, the VU meters will be laid out in rows of 32 columns. This will make 64 channels fit into two rows, rather than the three rows currently.

If the number of channels is more than 32 and not a multiple of 16, the VU meters will be laid out in rows of 30 columns, as they were in previous versions. Less than 32 output channels will have the VU meters laid out in a single row, as previously.

- 2) Previous versions had a bug that prevented output VU meters in rows past the first row from working. This is now fixed.
- 3) The timecode reader checked the wrong bit in the SMPTE frame to determine the difference between 30fps and dropframe. The result was that dropframe timecode was decoded as "slow 30fps" timecode. This has been fixed.
- 4) Sometimes people record timecode in strange ways. A classic case is recording plain 30 fps timecode at an actual rate of 29.97 fps.

The timecode reader can only accurately detect 4 framerates by itself:

- 30 fps
- 29.97 dropframe
- 25 fps
- 24 fps

It is important to note that the difference between 30fps and 29.97 dropframe is that the SMPTE frame has the "drop frame" bit set in the frame. If the dropframe bit is set the timecode is expected to appear at 0.999 times the normal rate, which is the difference between 30fps and 29.97 fps.

But if someone records 30fps NON-DROPFRAME timecode at 29.97 fps, SM-S cannot detect that the nominal framerate is 29.97 rather than 30. This is only 0.1% slow, and a movie projector generating SMPTE can easily be off by +- 5%. So it merely looks like normal code that is a little slow, and as a result the timecode reader generates time that is a little slow. This isn't very good if you are trying to sync audio to that 29.97 fps source, as the audio will play slow and drop behind the image.

To get around this problem, new syntax has been added to SET GROUP to allow you to specify that you know the timecode is slow, and then SM-S can compensate for the slow rate.

```
SET
  GROUP group-spec
    NAME <name>
    CHANS | CHANNELS channel-spec
    CLEAR
    TC | TIMECODE
      READER {channel number}
        DROPSPEED
        NORMALSPEED
      LOCK
        SPEED
        {TIME|TC} <timecode> [AT {TIME|TC} value]
    UNLOCK
```

The SET GROUP command to set up a number of playback channels on a timecode reader now looks like the above. The **DROPSPEED** and **NORMALSPEED** parameters have been added. Note that these MUST come right after READER and the reader channel number!

DROPSPEED tells the timecode reader that the timecode is running at "dropframe rate", or 0.1% slower than the nominal rate. Use this when you have plain 30fps timecode recorded at 29.97.

NORMALSPEED is the default and tells the reader that the timecode is running at the nominal rate. Use this when 30fps timecode is running at a real 30fps, or when you have dropframe-coded timecode running at 29.97 fps. If you do not put either **DROPSPEED** or **NORMALSPEED**, then **NORMALSPEED** is assumed.

An example of normal 24/25/30fps or 29.97 drop timecode setup:

```
SET GROUP G0 CHANS P0-3 TIMECODE READER 1000 LOCK TC 1:0:0:0;
```

An example of locking to 30fps timecode running at 29.97 fps:

```
SET GROUP G0 CHANS P0-3 TIMECODE READER 1000 DROPSPEED LOCK TC 1:0:0:0;
```

Version 1.0.45
2009-07-14

- 1) Fixed a playback lockup that could occur under certain conditions when many commands were issued at once.
- 2) Setting a resume point that was past the end of a track would cause SM-S to loop. This is now fixed. A resume point at or past the end of a track will be ignored. A warning message is generated in the log.
- 3) There is now an info message in the log when a track actually stops as well as when it starts.

Version 1.0.44
2009-05-31

- 1) A broken ASIO driver installation with a missing code file could cause SM-S to crash on startup. This is now handled. (This was a day-one bug in Steinberg-supplied ASIO driver code!)
A warning message will appear in the log indicating the ASIO name of the bad driver installation.
- 2) Removed extraneous newline from "Mapped channel ordering" log message.
- 3) Creating a dump file could fail, so it is now protected in a try block.
Hopefully this will catch the failures.
- 4) Added code to retry a dongle read if it fails, since that seems to happen now and then.
- 5) Added some protection logic to the audio file read routines as they could sometimes fault under unusual conditions.
- 6) Crosspoint delay setting was badly broken in a number of ways. This didn't show up with the small buffer sizes we usually use for testing the audio path, but was blatantly obvious on large buffer sizes.
- 7) Some problems also existed with delay fades on input and output channels when using large buffers.
- 8) Added pitch shifting ability to output channels. Previously it was only available on input channels, although the commands appeared to work for all types of channels.

Version 1.0.43
2009-05-12

- 1) Setting phase reverse on a crosspoint would incorrectly mute the channel. This now works correctly.
- 2) If you attempted to PLAY a group before defining the channels for the group things would mess up and refuse to let the group be used ever again. This is now fixed.
- 3) It is now possible to use multiple ASIO interfaces at the same time, with some important restrictions:
 - 1 All ASIO interfaces MUST have the same number of samples in the buffer size. For example, all must be 64 samples or all 512 samples. One driver at 64 samples and one at 512 samples will not work.
 - 2 All ASIO interfaces MUST have the same sample rate, or be possible to set them all to the same rate. This means they must all do 48000 or 44100, unless you specify some other specific sample rate, in which case all of the drivers must run at that rate.
 - 3 All of the physical interfaces MUST be sample-locked using Wordclock or something similar. The interfaces will run if they are not sample locked. But as with all digital audio, if the source and destination do not have the same clock you will get clicks and pops. In the case of SM-S you can also get phase cancellation if you play the same source through multiple interfaces and they are not sample-locked.
 - 4 The interfaces do not have to use the same sample data format. It is possible to mix drivers that use 24 bit samples and 32 bit samples, or even 16 bit samples. If you mix ASIO devices with 16 bit samples with devices using 32 bit samples the difference in audio quality will be immediately obvious, so this is not recommended.
 - 5 Grouping interfaces adds one buffer time of delay to the output path. Therefore latency will be longer by the time it takes to process one buffer of data. You can compute this time by:
$$\text{added delay} = \text{buffersize} / \text{samplerate}$$
For instance, a buffer size of 64 samples will add 1.3ms, which is probably not enough to worry about. But a buffer size of 2048 samples will add 43ms, which is a significant delay and would make it impossible to use the outputs to drive stage monitors. If the output audio is synchronized to video this is nearly 4 frames times at 60 frames/second, and would be very obvious. You would have to adjust the timecode following offset to compensate for this.
 - 6 If several ASIO devices are grouped for use together, they CANNOT be used separately while the group exists, even if the group is not currently in use. you must destroy the grouping before the devices can be used individually. Of course, the group can be recreated later if

needed.

- 7 A group CANNOT be created if one of the interfaces to be used in the group is currently in use. You can only create a group from interfaces that are not currently in use.
- 8 A group of interfaces has a name. This name CANNOT be the same as the name of an existing interface or interface group.
- 9 You can only create a group from actual ASIO interfaces. You CANNOT create a group by combining another group.

To group multiple ASIO devices you use the command

```
CONFIG SET ASIOWGROUP "name" INTERFACE {"name"|number}...
```

For instance with two existing interfaces they will be numbered 0 and 1.
You could create a group with:

```
CONFIG SET ASIOWGROUP "MYGROUP" INTERFACE 0 INTERFACE 1
```

This will group both interfaces together as a single interface group, letting you use all of the channels on both interfaces at the same time (if your license limits will allow that many channels).

While interface numbers are shown above, you could have used the names of the interfaces as shown by CONFIG GET INTERFACES. Remember to use quote marks around the names!

You may want to remove an ASIOWGROUP so that you can use the interfaces individually. You can do this with:

```
CONFIG CLEAR ASIOWGROUP "name"
```

Of course this will only work if the group is not currently in use. You may have to do a CONFIG CLOSE INTERFACE before you can clear the group.

Once you have an interface group defined you use it exactly like you would use a single interface, for instance:

```
CONFIG SET INTERFACE "MYGROUP".
```

- 4) Timecode generation was broken in the last version or two. This has now been fixed and the timecode generators again work correctly.
- 5) Timecode following was broken in the last release. If the file position had to seek to follow a skip in the timecode, the track would get marked as not ready to play at the start of the seek and stop playing. This is now fixed.
- 6) Setting the start time of the timecode generator while it was running would only randomly affect the generator. It was originally supposed to have no effect until the generator was stopped and restarted, but it didn't quite work that way. It has now been changed to immediately set a new time in the generator.

Version 1.0.42
2009-04-02

- 1) Fixed yet another way we could get a dump during shutdown.
- 2) A nonsensical crash back on version 37 indicates that sometimes ASIO drivers don't work the way the ASIO spec says they should. Put protection code around the attempts to release the ASIO drivers when we no longer need one.
- 3) Did some minor cleanup on file type determination code.
- 4) Added log messages to show how many worker threads are being used for the audio processing. Multithreading was added in version 38.
- 5) Added a checkbox to the interface to disable multithreading on the audio processing. There are reports that some machines click and pop like crazy when multithreading, but work fine normally.
- 6) Added a log message that shows the version of OS in use.
- 7) Some optimization code put into version 38 caused ticks when playing back if sample rate conversion was not needed.
- 8) If a new track was loaded on a playback channel without stopping the previous track, there were cases where the new track could continue to play for various lengths of time. This could also result in various clicks and buzzes, and arcane results like one channel of two playing at twice or three times normal speed. Hopefully all this has now been fixed.

Version 1.0.41
2009-03-22

- 1) We could get a tick on the end of a sound file if there were extra hunks on the end of a wave or AIFF file.
- 2) Fixed a crash that could happen late in exiting the program.
- 3) SET GROUP G0 TIMECODE UNLOCK would still produce a syntax error due to an incomplete fix in version 39.

Version 1.0.40
2009-03-22

- 1) Minor formatting cleanup in an error message that occurs if SM-S is unable to send to the TCP socket connected to the controlling application.
- 2) If SM-S got an EWOULDBLOCK on an attempted send to a TCP socket, it would loop retrying the send with no delay. On a single processor system this could have used all of the time needed for a sender to get around to reading from its socket. There is now a 10ms delay between retries.
- 3) Some locking has been added where selection pointers are being set and cleared in input channels. This should prevent a very rare crash where a channel pointer is deleted from a selection on two threads at once.
- 4) Fixed some problems with setting up the operating mode and VU meters correctly if CONFIG SET INTERFACE was entered from the command line. This problem would occasionally result in missing VU meters the first time the window was displayed.
- 5) Changed the memory dump code to make larger dumps. The previous dumps often lacked information necessary to find the problems.
- 6) CCriticalSection doesn't seem to work as it is defined. Replaced all uses with my own critical section wrapper that does work.
- 7) Added some interlock logic to prevent attempting two motion commands at the same time on a single channel. The new logic doesn't handle all possible cases, but it handles the cases that can occur commonly with SM-A. This prevents some possible very rare crashes in SM-S, and also insures that the commands will have the intended result.
- 8) Found a problem in the thread manager that could cause a crash on shutdown. This is now fixed and SM-S shuts down cleanly again.

Version 1.0.39

2009-03-21

- 1) A crash that could happen when opening the ASIO interface a second time has been fixed.
- 2) Unlocking a group of channels from a timecode reader was not working correctly and would give a syntax error. This has been fixed.
- 3) The previous version broke being able to have the same audio file opened on multiple playback channels at once. This is now fixed.
- 4) Setting a delay on a playback channel and then stopping it could cause a buzz due to looping the last buffer that was playing. This is now fixed. Note that if you have a delay on a playback channel, the stopping and starting of the channel will seem to be delayed by the delay amount. What is actually happening is that the track is stopping and starting when requested, but the delay is causing the audio to take time to get to the outputs.

Version 1.0.38
2009-03-10

- 1) GET GAINMIDI returned incorrect values for input and output channels.
- 2) Added a CONTROL button to the interface to display the ASIO control panel for the current ASIO driver, if the driver has one. Some ASIO drivers have important configuration settings that can only be accessed through the ASIO control panel interface.
- 3) Validated operation with AudioScience ASI6464 Cobranet cards and their latest development ASIO driver. When using the latest driver it is necessary the first time the driver is selected to perform the following actions:
 - a) Select the AudioScience ASIO driver in the SM-S control panel. (If you see more than one AudioScience ASIO driver to choose from, you have an old version of the ASI firmware which will not work correctly. Get a newer download from AudioScience.)
 - b) Click the Connect button in the SM-S control panel and observe that the ASI driver opens correctly.
 - c) Click the Control button in the SM-S control panel. The ASI ASIO driver control panel will open.
 - d) On the left of the "Adapter and Sample Type" pane of the ASI driver will be a checkbox list of all of the ASI6464 cards in the system. Only one of them will be checked. Check all of them.
 - e) On the right of the "Adapter and Sample Type" pane of the ASI driver is a radio button list allowing you to select the sample format that the ASIO driver will use. The 16 bit signed integer button will be selected. As this is a lower-quality sample format, you will want to select one of the other formats. If you are only going to be using the ASI cards with SM-S, select the 32 bit floating point option. If you will be using the cards with other applications, select the 32 bit signed integer option.
 - f) In the "Buffer Settings" tab you can change the default buffer size for the ASIO driver. This will be set to 2304 samples. This setting has a quite long latency. You can set this to the minimum of 1152 samples, but on a slower machine you may run some chance of having dropped samples. Unless low latency is critical I recommend leaving this option set to the default value.

Do not change any of the other options in the Buffer Settings tab.
 - g) Click OK to save the settings changes. The control panel will close.
 - h) **VERY IMPORTANT! CLICK THE DISCONNECT BUTTON IN THE SM-S CONTROL PANEL AT THIS POINT.** The driver settings have changed drastically, but SM-S is still using the old settings.

If you do not disconnect from the driver and then reconnect, SM-S or the driver will almost certainly crash.

- i) Once you have disconnected from the driver you may reconnect and begin using the driver normally. You will not need to visit the ASI control panel again in the future.
- 4) The history log that appears in a dump file now correctly has timestamps on the log lines.
- 5) If an ASIO driver callback thread appears to be open after closing the driver, but it terminates normally before we can kill it, we no longer claim that we killed the driver thread.
- 6) If a driver has Float32 sample format but the buffers are not aligned on 16-byte addresses, SM-S would crash. This is now fixed. Sixteen byte alignment is MUCH faster than other possible sample alignments, but having things run at all is faster than crashing and not running.
- 7) If an attempt was made to open an ASIO driver and the driver failed to load (perhaps because the interface hardware was unplugged) it was possible that SM-S would crash. This has been fixed.
- 8) SET MATRIX no longer includes the signal generators and timecode generators and timecode readers in the matrix setup. Only the live inputs and the playback inputs are included in the setup.
- 9) The SET MATRIX command has been enhanced. It is now possible to set up only the live inputs or only the playback part of the matrix without also setting the other part. This allows you to leave existing settings for one section while setting the other, or to set different matrix parameters on each section. By default the command will still set both sections.

The syntax now is:

```
SET MATRIX
  IN | INPUT | PB | PLAYBACK | ALL
  FULL | DIAGONAL
  ON | OFF | GAIN <gain> | GAINDB <gain>
```

The SET MATRIX ROW command remains unchanged:

```
SET MATRIX ROW n GAIN|GAINDB channel [-|TO channel] @value...
```

In the SET MATRIX command the IN/INPUT/PB/PLAYBACK/ALL parameters have been added. Specifying IN or INPUT will limit the matrix settings to the live inputs and the outputs. Specifying PB or PLAYBACK will limit the settings to the playback inputs and the outputs. Specifying ALL, or not specifying any of the new options, will result in setting both the live and playback inputs.

If FULL is specified all crosspoints in the requested sections will be set to the requested gain level. ON and OFF are the same as full gain and zero gain. The input and output channel gains will be set to 1 for any gain other than OFF or 0.

If DIAGONAL is specified all crosspoints in the requested sections will initially be set to 0 gain, and then only the "diagonal" crosspoints will be set to the requested gain. Diagonal crosspoints are those crosspoints with the same input and output channel numbers. The input and output channel gains will be set to 1 for any gain other than OFF or 0.

In all cases the signal generator gains will be set to 0 so that they don't interfere with the overall matrix setup.

- 10) The GET CHAN <chan> PITCH command now works. Previously it was accidentally not enabled.
- 11) If you set up a track to loop and also set a non-zero start point for playback, the track playback would seriously mess up when it tried to loop. This has been fixed and loops now work correctly in all known cases.
- 12) SoundMan-Server now recognizes WAVEFORMATEXTENSIBLE files and most of the possible sample formats. This allows wave files with up to 18 channels of data (or possibly more, but Microsoft only defines 18 channels) and wave files with sample formats larger than 16 bits. The most common sample format is 24 bit PCM samples, but some programs also create 32 bit IEEE floating point samples. There are many other sample formats that are possible in WAVEFORMATEXTENSIBLE, and the more likely ones are handled.
- 13) SoundMan-Server now recognizes Broadcast Wave Format wave files that can have more than 2GB of audio data. These files use a type of RF64 rather than RIFF. Chained BWF files are now yet handled automatically.
- 14) SoundMan-Server now plays AIFF files in most non-compressed formats. They are handled exactly like wave files.
- 15) Much of the internal list and linkage logic has been reworked to reduce the program overhead while playing audio. The reduction probably isn't huge, but it should help on slower machines.
- 16) When running on a multiprocessor system, SM-S will now use all processors to process audio. Previously only a single processor would be used. This should allow considerably more high-overhead processing such as eq and pitch shifting without running out of processor time.
- 17) Problems with shutdown sequencing when SoundMan-Designer requests that we shut down have been fixed. We should no longer be getting a dump every time the program is closed.

Version 1.0.37
2008-12-18

- 1) Updated to new dongle files with some bug fixes.

Version 1.0.36
2008-11-30

- 1) A change in the last version broke parsing of input and channel numbers that were entered in lower case.
- 2) Mixing groups and playback channels in the same command such as PLAY GO,P0 was not working correctly. Only the first item in the list was used.
- 3) Timecode following was broken in recent versions. This is now fixed and timecode chase again works.
- 4) A timecode generator chasing another timecode source was very slow in responding to a major jump in the source timecode. This is now fixed.
- 5) If a parametric filter section is specified as highpass or lowpass and given a gain of exactly 0db, it will now reduce the gain in the filter band relative to the passband. Previously a 0db gain had been accidentally treated as a 'boost' gain, resulting in a gain increase in the filter band.

Version 1.0.35
2008-11-05

1) The recent changes to the timestamp format on log lines broke the trick of selecting a previous command from the log and having it copied back into the command line. This is now fixed.

2) SET GEQ FREQ 400 GAINDB 2 was setting the wrong band gain.

3) The command to get the graphic eq gain for all bands has been simplified. Previously it was either

```
GET CHANNEL <chans> GRAPHICEQ BANDS GAIN
GET CHANNEL <chans> GRAPHICEQ BANDS GAINDB
```

Now it is

```
GET CHANNEL <chans> GRAPHICEQ GAIN
GET CHANNEL <chans> GRAPHICEQ GAINDB
```

Note that the word BANDS is no longer needed. The response format has not been changed, only the command syntax itself.

Also, if you just enter the following with no other parameters,

```
GET CHANNEL <chans> GRAPHICEQ
```

You will now get the band gains in dB. Previously this returned the enable state of the graphic eq for the channel. Note that gains will be shown for the bands even if the eq is disabled for that channel!

The complete list of GET commands for the graphic eq now is:

QUALITY	# GrEqQuality	chanid=number
DELAY	# GrEqDelay	chanid=fpt
ENABLE	# GrEq	chanid=ON/OFF
GAIN	# GrEqGain	chanid=fpt,fpt... 31 times
GAINDB	# GrEqGainDB	chanid=fpt,fpt,.. 31 times
BAND number	#	
FREQ FREQUENCY	# GrEqBandFreq	chanid=band=fpt
GAIN	# GrEqBandGain	chanid=band=fpt
GAINDB	# GrEqBandGainDB	chanid=band=fpt

4) The timecode generators were not being freed when SM-S shut down, resulting in a minor memory leak. This is now fixed.

5) The GET CHAN <chan> GRAPHICSEQ BAND <number> GAIN/GAINDB commands were not returning the band number in the response as they should have.

6) Commands have been added to get the response for both individual parametric eq bands, and for all eq enabled on a channel, including both parametric and graphic eq. The response values are given in dB, where 0dB indicates a "flat" response for the channel. The eq response values do NOT include overall channel gains, only the gain contributions from the eq for the channel.

EQ		# Eq	chanid=ONIOFF
	RESPONSE	# EqResp	chanid=fpt,fpt,fpt... 31 times
	AT <freq>	# EqFreqResp	chanid=freq=fpt
	BAND number		
	RESPONSE	# EqBandResp	chanid=band=fpt,fpt,fpt... 31 times
	AT <freq>	# EqBandFreqResp	chanid=band=freq=fpt

The command

GET CHANNEL <chans> EQ RESPONSE

Will get the response of all enabled parametric eq bands for the channel, plus the response of the graphic eq if it is enabled. If no eq is enabled, the response will show as flat at all frequencies.

The response is returned as 31 numbers representing dB values for specific frequencies. The frequencies are the same as the 31 band centers for the graphic eq bands. These are logarithmically spaced across the audio frequency spectrum, so will give a good overall impression of the frequency response, but it will not show fine detail from very narrow filter bands.

If it is known that there are very narrow parametric bands in use, the user can get a more accurate overall response by issuing a few commands of the form:

GET CHANNEL <chans> EQ RESPONSE AT <freq>

Where the frequencies are picked to cover the area of the narrow eq band. For instance, this could be used to request the overall response at the center frequency of a bandpass filter to get an accurate value for the bottom of a notch or the top of a bump. Since the 31 band responses are in 1/3rd octave increments they will provide good overall frequency response coverage, and requesting individual responses should only be needed for the center frequency of bandpass filters, or at most a few frequencies around the corner frequencies of the various enabled filters.

Because it can be desirable to show a graphic response of the individual eq bands as well as the overall eq curve, you can get the response for each band individually. In this case the response will show the eq band response even if the band is disabled. It is up to the caller to realize that disabled eq bands do not contribute to the overall channel eq value. These individual band responses do not include any contribution from the graphic eq subsystem. However, you can get the graphic eq response in dB with

GET CHAN <chans> GEQ GAINDB

To get the overall response for an individual band, use a command in the form:

GET CHAN <chans> EQ BAND <band> RESPONSE

This will return the response as 31 numbers, which are the gains in dB at the standard 31 band centers, as described above. The response format (as shown above to the right of the command) is slightly different than described previously, since the band number is included at the front of the response.

Since the eq band might be a narrow notch or bump, you can get the exact response at a specific frequency with

```
GET CHAN <chans> EQ BAND <band> RESPONSE AT <freq>
```

This is again very similar to the command described above, except that it returns the response for only the single band requested, and it will show the band response whether or not the band is enabled.

- 7) Several problems were found and fixed in the parametric eq interrogation commands.
- 8) If the window is minimized to the taskbar (rather than hidden in the tray) double-clicking the tray icon will now make the window visible. Previously double clicking in this case had no effect and was somewhat frustrating.
- 9) The error responses returned for bad values of MUTE and SOLO had an incorrect description of the error.
- 10) On Vista you can't write files to the root directory on the C drive from a program. Therefore the dump files are now written to a directory created in My Documents.
- 11) If the program is started with the window minimized (as would be the case from SM-D) the window will quietly be moved to the tray to clean up the unnecessary icons on the task bar.
- 12) Commands are now logged in their original case rather than being forced to uppercase before being logged. Previously strings in quotes would have been uppercased in the log records, and the command could then fail if the command was reused from the command history in the control panel. By logging in the original case the case of the strings is preserved.
- 13) The VU meters were not always being laid out correctly, and would come up as blank a lot of the time.
- 14) The disk usage meter was unrealistically sensitive and would tend to run very high, even on almost unused disks. It is still deliberately a bit "generous" in its indications, but it is much more realistic now.
- 15) Found and fixed a very unlikely crash that could occur if you maximized the SM-S window at just the wrong time.
- 16) The ANALOGFIRST parameter was ignored the first time an ASIO port was opened. This is now fixed.
- 17) The first time you clicked the minimize button on the main window the window would minimize to the tray rather than correctly minimizing to the taskbar. This would only happen once, and then it would work correctly. This has been fixed.

Version 1.0.34
2008-10-14

- 1) If a socket error occurs on a send to the socket, the log message is now more explicit about both the text sent and the error returned.
- 2) The logging code that computed the milliseconds for the current message could have been off by a few milliseconds. This has been fixed.
- 3) Improved the accuracy of the time function that produces the log timestamps.
- 4) Eliminated a blank line in the log records for the device channel ordering.
- 5) If a socket send gets an EWOULDBLOCK response it will now be retried for up to 100ms before it is declared to be an error. This should almost never happen, but can happen for a few milliseconds while SM-A is loading a show.
- 6) When the Close item is selected in the system menu, or the big X in the upper right corner is clicked, the program will now minimize to the tray. The normal Minimize bar (near the big X) will minimize to the normal taskbar location.

To exit the program, use either the Exit item in the menu on the system tray icon, or use the new Exit menu item in the system menu.

- 7) Due to an oversight the code to set up the graphic eq had not been enabled. This is fixed.
- 8) The GET GRAPHICEQ BANDS GAINDB command was not returning a correct result.

Version 1.0.33
2008-10-10

- 1) The GET EQ ENABLED command was erroneously returning solo state, not parametric eq enable state as it should have.
- 2) If attempting to set the ASIO interface using the Connect button and SoundMan-Assistant is running, a dialog box will warn that this is only a temporary setting, and the right place to set the ASIO interface is in SoundMan-Assistant. If the user clicks on Yes, the SM-A configuration dialog will be displayed so that the user can pick the interface there.
- 3) Parametric eq bands are numbered starting from 1, but the GET EQ responses were showing band numbers starting at 0. This has been fixed. Also, the GET EQ command expected requested band numbers to start at 0 rather than 1. This has also been fixed.
- 4) Entering a band number of 0 on SET EQ command erroneously complained that the band number was too large, when it was actually too small. It now just says that the band number is invalid.
- 5) If you attempt to set frequency or bandwidth for a FLAT parametric filter section you will get an error response, as a flat filter does not have a specific frequency or bandwidth. Previously the parameters were accepted but ignored, leading to confusion about how the eq was working.
- 6) If you attempt to set the bandwidth for any filter shape except bandpass you will now get an error response. Previously the parameters were accepted but ignored, leading to confusion about how the eq was working.

Version 1.0.32
2008-09-12

- 1) Corrected some problems in the code that creates the dump files if something goes wrong.
- 2) Log messages now have timestamps accurate to about a millisecond, rather than 10-15ms as previously.
- 3) There is now a date stamp on messages as well as a time stamp. This helps when a log covers several days.

Version 1.0.31
2008-08-25

- 1) Log messages now have the time in milliseconds rather than just seconds.
- 2) SM-S log messages can now be captured by the SoundMan-Monitor logging monitor application.
- 3) A great deal of extra logging of internal events has been added.
- 4) A new button on the interface will save the log to a file with one click rather than having to copy and paste into Notepad or a mail message.
- 5) Added code to automatically clear the debug message log if it gets more than about 65,000 messages in it.
- 6) Fixed up the tab order in the main window to follow a logical progression.
- 7) Added a top-level unhandled exception filter that will try to take a dump if a crash occurs. If the dumps are sent to me this should result in better support of unusual problems.
- 8) Fixed a potential problem that could result in a crash when setting up a new audio interface.
- 9) Fixed a memory leak where we weren't cleaning up the space used by the graphic EQ component.
- 10) The code in the ASIO callback routine now checks to see if it is using all of the processor time. If so, it begins slowing down so that the UI won't be locked up and require a reboot to regain control of the computer.

Slowing down will probably result in glitches in the sound, but this is better than locking up the computer. When slowing down like this the computer will be very slow and unresponsive, but with patience it is possible to kill SM-S or take some other action to reduce the workload on the computer.

The most likely reason at the moment for running out of processor time would be either using a matrix size that is too big for the amount of processing power on an older and slower computer, or using too much EQ. When VST plugins are added, they will also be a reason to run out of processing time, as not all VST plugins are efficiently coded.

- 11) Added a 31 band graphic equalizer to the channels and crosspoints. This can be used as an alternative to the 7-band parametric eq that has always been present.

The graphic equalizer is implemented using an FFT algorithm. While this allows good control over the overall waveshape and is somewhat less processor overhead than 31 individual parametric filters would be, it has some potential drawbacks, and should be used only when absolutely necessary. The graphic EQ is quite processor hungry and memory hungry, and a small

number of them can make quite a dent in the amount of processor time needed to process the audio.

The graphic EQ has a QUALITY setting of 1 to 8, where 1 is the lowest audio quality and 8 is the highest. The default is 3, which should be sufficient for most uses. The quality setting affects three things:

1. The amount of processor time required for the EQ operation
2. The amount of latency in the audio processing
3. The overall amount of audio distortion that can be caused.

A lower quality uses less processor power. It can also result in aliasing or chorusing distortion, especially at lower frequencies. This distortion is a natural result of how an FFT works: it divides the frequency spectrum into a number of "bins", where each bin contains the same number of frequencies. For instance, a 4-bin FFT would divide the range of 0-48KHz into 0-12000, 12000-24000, 24000-36000, 36000-48000. This FFT could produce severe aliasing or distortion, because all simultaneous tones in the range of 0-12000Hz would be merged into a single output tone. The same would happen in the range of 12000-24000Hz.

This FFT size would be quite low overhead, but the chances of it being very usable are slim. You might think it would be unusable in all cases, but that might not be the case. Consider a single person speaking or singing, with no background sound in the same channel. The human voice generally makes only a single frequency at any instant. So there would be no two frequencies to merge in either of the two important frequency bins, and the resulting output would presumably be the same as the input: no distortion would result.

SoundMan-Server does not use a 4-point FFT as the quality would only rarely be usable. The usable sizes are in the range of 128 to 4096 points. The QUALITY figure is used to select the number of points in the FFT, and also changes some other internal parameters that affect the overall sound quality. The higher the number the more FFT buckets, so the less chance of aliasing distortion.

Before you arbitrarily decide to just set the quality setting to 8 any time you use a graphic EQ, there are two other important things to know:

1. A setting of 8 uses about 64 times as much processor as a setting of 1.
2. The more points in the FFT the longer the latency!

The FFT size can be related directly to the amount of latency in milliseconds it will add to the channel:

FFT size	quality	sample rate	Latency
512	1 2	48000	10.67 ms
1024	3 4	48000	21.33 ms
2048	5 6	48000	42.67 ms
4096	7 8	48000	85.33 ms
512	1 2	44100	11.60 ms
1024	3 4	44100	23.22 ms
2048	5 6	44100	46.44 ms
4096	7 8	44100	92.88 ms

As you can easily see, even a small "QUALITY 1" FFT will add more than 10ms to the audio path. The highest quality FFT processing will add almost a tenth of a second to the audio path. This would not be a good choice to EQ the foldback for a singer or musician. In fact, it would be unusable for any form of live audio, but it could be used on recorded audio. (But then, why didn't you EQ the recording before using it?)

The graphic EQ can be used along with the parametric EQ on the same channel. This should be avoided unless absolutely required, as both the graphic and parametric EQs use a fair amount of processor time. Use the minimum number of parametric bands, as each band added adds more processing time.

There are new channel command parameters to set up the graphic EQ:

```
SET CHANCHANNEL channel-spec
  GEQ | GRAPHICEQ
    QUALITY 1..8
    ONIOFF
    BAND int // by band number 1..31
      GAINDB fpt
      = fpt // gaindb
    FREQ | FREQUENCY // by frequency 20..20000
      GAINDB fpt
      = fpt // gaindb
```

"SET CHANNEL n EQ" and "SET CHANNEL n GEQ" are different. The first sets the parametric EQ sections. The later sets up the graphic EQ.

Each type has its own enable. If the enable is off, the EQ processing is bypassed. As EQ processing adds latency to the channel, you can expect a glitch if you turn the EQ processing on or off. You can keep the same latency by setting the various parametric EQ sections to FLAT or by setting the band gains for the graphic EQ to 0, but this also keeps the same amount of processor overhead for not much purpose.

The QUALITY setting is discussed above. The default QUALITY is 3. You should rarely find a need for a setting above 6, and there are many cases where a setting of 1 or 2 is perfectly adequate.

The graphic EQ works just like the physical graphic EQs that you are used to. There are 31 "gain sliders" numbered 1 to 31 left to right. These are each set at the EIA standard band center frequencies that are used by almost all graphic EQs these days.

Each gain value can be set to anywhere between +24dB and -24dB. The default is 0dB, which does not change the signal level passing through. Note that you can either say "BAND 5 GAINDB 3.6" or "BAND 5 = 3.6" and get the same results. The equal sign will save a little typing.

If you don't remember band numbers but do know the band frequency, you can set the band gains by frequency rather than band number. The frequency you give will be converted to the nearest band number, and then the gain for that band will be set.

Here are the band numbers and center frequencies:

Audio Band#	Nominal Center Frequency Hz	Exact Center Frequency Hz	Passband Hz
1	20	19.95	17.8 - 22.4
2	25	25.12	22.4 - 28.2
3	31.5	31.62	28.2 - 35.5
4	40	39.81	35.5 - 44.7
5	50	50.12	44.7 - 56.2
6	63	63.1	56.2 - 70.8
7	80	79.43	70.8 - 89.1
8	100	100	89.1 - 112
9	125	125.89	112 - 141
10	160	158.49	141 - 178
11	200	199.53	178 - 224
12	250	251.19	224 - 282
13	315	316.23	282 - 355
14	400	398.11	355 - 447
15	500	501.19	447 - 562
16	630	630.96	562 - 708
17	800	794.33	708 - 891
18	1000	1000	891 - 1120
19	1250	1258.9	1120 - 1410
20	1600	1584.9	1410 - 1780
21	2000	1995.3	1780 - 2240
22	2500	2511.9	2240 - 2820
23	3150	3162.3	2820 - 3550
24	4000	3981.1	3550 - 4470
25	5000	5011.9	4470 - 5620
26	6300	6309.6	5620 - 7080
27	8000	7943.3	7080 - 8910
28	10000	10000	8910 - 11200
29	12500	12589.3	11200 - 14100
30	16000	15848.9	14100 - 17800
31	20000	19952.6	17800 - 22400

- 12) Many new responses for GET CHANNEL info have been added, and several bugs in the existing code have been fixed. The following chart shows the GET CHANNEL options that are now implemented and shows the keyword and result format that will be returned.

In the following table, the part to the left of the # shows the command.

Parts of the commands are indented, and an indented line must be preceded by the command part on the next outer line. For example, CHAN or CHANNEL must be preceded by GET (forming GET CHAN or GET CHANNEL). DELAY must be preceded by CHAN or CHANNEL, which itself must be preceded by GET, forming GET CHANNEL DELAY.

The part to the right of the # shows the response that will be returned. All responses start with a unique single keyword that is a mixture of upper and lower case. This makes it possible to distinguish easily from an error response or some other line. Since all response starters are unique, if responses are parsed asynchronously it is possible to determine

what each response is about without knowing the command that generated the response.

A GET command can ask about multiple channels at once. Each channel response is in the form of <channel identifier>= <response> <space>. The final channel response is followed by a semicolon, which terminates the response line. The entire response, no matter how long, will be on a single "line" with no carriage returns or the like in the response text. There is a carriage return and linefeed after the semicolon at the end of the response.

The channel identifiers are the same as the most compact form used for SET command input. For instance, input channel 0 is I0 and playback channel 10 is P10. Remember that crosspoints have two numbers, so will be of the form X2.3 or PX12.14. There is always exactly one channel identifier immediately followed by an equal sign with no spaces. The response to the query for that channel then immediately follows the equal sign with no intervening spaces. Another channel identifier, equal sign, and response value can then follow the response value for the first channel listed.

Channels are not necessarily listed in the same order in responses that they were given in the original GET command. Do not depend on the order being the same.

Query responses are usually numeric. They can be integer numbers, floating point numbers that contain a decimal point, or a timecode value. None of these items contain spaces. So you can determine the end of the response value for each channel by looking for the first space after the equal sign.

Sometimes the response value is a string value of some form. There are three general cases for this. The first is querying something like the shape of an EQ section. The response is a fixed string that is more descriptive than an arbitrary number would be. This string has a fixed spelling and does not contain spaces, so it is given immediately after the equal sign, and again the end of the response word can be determined by looking for the first space.

The second case of a string response is asking for a channel, group, or other item name. Names are always uppercase and never contain spaces. They may be a mixture of numbers and letters and underscore characters. The first character will always be a letter. Again, names never have spaces in them, so scanning from the equal sign for the first space will find the end of the name field.

The final case of a string response is asking for a file name. Here the file name can contain any characters except a double-quote mark, and can be upper and lower case mixed. The file name can also contain one or more spaces, and very often does. In this case there will be a " mark immediately after the equal sign. The file name will follow this, and will be terminated with another " mark. In this case the file name string must be parsed by looking for the trailing quote mark, and NOT by looking for the next space! There will be a space after the trailing quote mark and then the next channel identifier or the trailing semicolon will show up, in the usual manner.

In the following lines, there are small keywords that indicate the form of the response you should expect. The more common ones are:

chanid the channel identifier, such as P1 or O12 or PX1.1

number	an integer with no decimal point
fpt	a floating-point number containing a decimal point
ON/OFF	either the word ON or the word OFF, in uppercase
none	The word "none" in lowercase. This is used when querying a value and there is no value set.
fpt,fpt,fpt	three floating point number (see above) separated by commas
fpt,...	one or more floating point numbers, all but the last one followed by a comma to separate it from the next one. The last number is NOT followed by a comma.
band	In EQ responses, an integer band number. Note there are TWO equal signs in this form of response for each channel!
timecode	a timecode value of the form 01:02:03:04.05, giving the timecode value to a hundredth of a frame. Timecodes are 30FPS.

GET

```
CHAN|CHANNEL channel-spec
NAME # Name chanid=namelnone
PORT # Port chanid=portname
ROW # Row chanid=numberlnone
GAIN|GAINDB # Gain|GaindB chanid=fpt
GAIN|MIDI # GainMidi chanid=number
PHASEREVERSE # PhaseReverse chanid=ON|OFF
MUTE # Mute chanid=ON|OFF
SOLO # Solo chanid=ON|OFF

DELAY # Delay chanid=ON|OFF
  ENABLE # Delay chanid=ON|OFF
  TIME # DelayTime chanid=fpt
  TC # DelayTC chanid=timecode
  FADETIME # DelayFade chanid=fpt
  FADETC # DelayFadeTC chanid=timecode

EQ # Eq chanid=ON|OFF
  ENABLE # Eq chanid=ON|OFF
  BANDS # EqBands chanid=number
  BAND number
    SHAPE # EqBandShape chanid=band=shape
      # shape =FLAT|LP6|LP12|HP6|HP12
      # BP |LS6|LS12|HS6|HS12
    ENABLE # EqBand chanid=band=ON|OFF
    PARAMS # EqBandParams chanid=band=freq,gain,bw
    GAIN # EqBandGain chanid=band=fpt
      FADETIME # EqBandGainFade chanid=band=fpt
      FADETC # EqBandGainFadeTC chanid=band=timecode
    GAINDB # EqBandGaindB chanid=band=fpt
      FADETIME # EqBandGaindBFade chanid=band=fpt
      FADETC # EqBandGaindBFadeTC chanid=band=timecode
    BANDWIDTH|BW # EqBandBW chanid=band=fpt
      FADETIME # EqBandBWFade chanid=band=fpt
      FADETC # EqBandBWFadeTC chanid=band=timecode
    FREQ|FREQUENCY # EqBandFreq chanid=band=fpt
      FADETIME # EqBandFreqFade chanid=band=fpt
      FADETC # EqBandFreqFadeTC chanid=band=timecode

GEQ|GRAPHICEQ
  ENABLE # GrEq chanid=ON|OFF
  BAND number #
    GAIN # GrEqBandGain chanid=band=fpt
    GAINDB # GrEqBandGaindB chanid=band=fpt
  BANDS
    GAIN # GrEqGain chanid=fpt,fpt... 31 times
    GAINDB # GrEqGaindB chanid=fpt,fpt... 31 times

PITCH # Pitch chanid=ON|OFF
  QUALITY # PitchQuality chanid=number
  DELAY # PitchShiftDelay chanid=fpt
  ENABLE # Pitch chanid=ON|OFF
  FFTSIZE # PitchFftSize chanid=number
  OVERLAP # PitchOverlap chanid=number
  SHIFT # PitchShift chanid=fpt
```

FADETIME		# PitchShiftFadeTime	chanid=fpt
FADETC		# PitchShiftFadeTC	chanid=timecode
TRACK			
FILE		# TrackFile	chanid="name"
STATUS		# TrackStatus	chanid=PLAY STOP
POSITION		# Position	chanid=fpt
TIME		# TrackTime	chanid=fpt
LENGTH		# TrackLength	chanid=fpt
SPEED		# TrackSpeed	chanid=fpt
PITCH		# TrackPitch	chanid=fpt
START	#		
TIME		# TrackStart	chanid=fpt
TC		# TrackStartTC	chanid=timecode
STOP	#		
TIME		# TrackStop	chanid=fpt
TC		# TrackStopTC	chanid=timecode
REPEAT	#		
TIME		# TrackRepeat	chanid=fpt
TC		# TrackRepeatTC	chanid=timecode

- 13) Changed the first log message of "SoundMan-Server starting" to include the current version number.
- 14) If a sound file fails to open, the Windows error message will be included in the log message.
- 15) The Configuration Limits log line now shows the maximum licensed sample rate as well as the chosen sample rate.
- 16) A bug in input converter setup could have caused some wave file channels to not play when they should have,
- 17) Fixed a number of minor inconsistencies in the menus between SM-A and SM-S. Also insured that the version number will show up in the title bar of the main dialog at all times.

Version 1.0.30
2008-06-27

- 1) A small problem in delay time calculations that could make small delay errors is fixed.
- 2) A recent change was causing input delay values to be doubled incorrectly.
- 3) Changed a compile option to put more debug data in the code file. This should help if it is necessary to look at crash dumps.
- 4) Added syntax to allow setting the channel delay in samples as well as fractional seconds or a timecode value. Also, if the time value is entered as a number without "TIME" in front of it, you can now also enter any of the other delay parameters. Previously all that was allowed in this format was "DELAY <time>".

The new syntax now is:

```
SET CHAN|CHANNEL channel-spec
    DELAY
        <number> [SAMPLES]
        TIME fpt [SAMPLES]
        TC          timecode
        FADETIME fpt
        FADETC fadetime
        ONIOFF
```

- 5) SM-S will no longer completely lock up the system if the audio processing takes more time than there is available. The processing routines will check if the user interface is able to run at all, and if not, will give a few milliseconds per second to the UI processing. This may very well cause glitches in the audio; however, glitches are very likely anyway when all of the processor time is in use.

A balloon popup will be displayed by the tray icon indicating that the processing is running overtime. On a single processor system the mouse and keyboard will be very sluggish, but enough time will be left that it will be possible to stop SM-S (or change some parameters) without having to reboot the system.

- 6) It is now possible to do a pitch shift on any input or playback channel. This is a processing pitch shift on the audio stream, NOT changing the playback speed to achieve a pitch shift. On playback channels the channel pitch shift can be used with an inverse track speed or pitch shift to achieve a change in playback speed without changing the output pitch.

The pitch shifting routines use a fair amount of processor, so the number of channels they can be used on is limited. Also, they do not guarantee identical phase tracking over short durations on multiple channels, so it is possible to get some stereo image shifting when pitch shifting a multi-channel audio stream.

Several processing quality settings are possible. The standard setting should be good for many uses, but will fail miserably with dense rock music that is highly compressed. It can work quite well on vocal and folk music.

Higher quality settings have two drawbacks: they can use MUCH more processor power, and they can have very long processing delays. The standard level of processing results in about a 20ms delay. Delays up to 170ms are possible at the high quality settings. It can be possible to get acceptable results with some audio sources using a low quality setting that will have a 10ms or even a 5ms input to output delay. Whether this can be done will depend entirely on the nature of the source audio material, and of course how critical you are of the results.

Pitch can be shifted plus or minus one octave, and the shift value is entered in semitones, possibly with fractional semitone values.

SET CHAN <channels> PITCH [ON | OFF] <semitones> [FADETIME <time>]

- 7) A bug in the command parser that could cause it to lock up on some kinds of invalid input syntax has been fixed.
- 8) The SET GENERATOR syntax has been enhanced to allow sweep parameters of ON and OFF as well as PAUSE, STOP, and RESUME. ON and RESUME are the same and will resume the sweep if it is paused. STOP, PAUSE and OFF will stop the sweep if it is currently running, holding the current frequency.

The syntax for the generator is now:

```
SET GEN | GENERATOR      [channel]
    SHAPE SINE|SQUARE|TRIANGLE|WHITE|PINK
    FREQ|FREQUENCY      fpt 5..24000
    SWEEP PAUSE|STOP|RESUME|ON|OFF| minf maxf time
```

- 9) Setting the frequency of the signal generator will pause any current sweep and set the new frequency as a constant value.
- 10) Get track position now will correctly return an integer value in samples rather than a floating point value.
- 11) Chasing a timecode that jumped around or had a large offset into the files tracking the timecode could behave poorly. This has been greatly improved.

Version 1.0.29
2008-06-07

- 1) Moved to a new version of the ASIO SDK that makes it easier to handle multiple ASIO interfaces at once. Still need to do more to actually open multiple interfaces at once.
- 2) Fixed up the code that closes ASIO drivers to handle stupid drivers like the Maya driver that do not terminate their callback thread during ASIOExit. Previously this could result in random crashes due to the driver data disappearing out from under the still-running callback thread,
- 3) Found a problem with the callback routines that with some drivers could sometimes cause a crash. Fixed it.
- 4) Fixed some static variables in the pink noise generator so that multiple pink noise generators can be used at the same time without interference.
- 5) SET GROUP parsing was incorrect and would not allow more parameters after the NAME parameter. This is fixed.
- 6) An uninitialized variable could make a valid SET GROUP command fail.
- 7) Channel groups were not being constructed properly unless there was a timecode reader attached to the group. Now they are.
- 8) A new GET command has been added: CONFIG GET CHANNELINFO. This command will return a multiline response that will display the ASIO description string for all of the input channels, followed by all of the output channels. This can be useful in figuring out which channel number corresponds to which physical input or output on devices with multiple different interfaces.
- 9) Some ASIO interfaces mix up the channel order. For instance, if you have two M-Audio Delta 1010 interfaces, they will work together. However, the analog channels on the second interface will be separated from the analog channels on the first interface by the sp/dif channels and the mixer return channels. If you have a 16 channel license, you will not be able to use all 16 channels.

To get around this there is a new modifier on CONFIG SET INTERFACE. If you add ANALOGFIRST to the parameters, the descriptions of all of the channels will be examined, and any channel that has the word "analog" in its description will be moved before any other channels. Other than moving the analog channels before other channels, the channel order is preserved. In the case of the Delta 1010s, the channels will be ordered with the 16 analog channel first, followed by the sp/dif channels for the first interface and then the sp/dif channels for the second interface. This can also be useful with a MOTU 324 or 424 card and multiple 2408 interface modules to get the analog channels for all interfaces on the low channels.

Note that this trick will only work if the description string for the channel contains "analog" in any case, including "Analog" and "ANALOG".

If there are no channels that declare themselves to be analog the channel order will not be changed.

The form for CONFIG SET INTERFACE is now:

```
//      CONFIGCONFIGURATION
//          SET
//          INTERFACE    n!"name"
//                      SAMPLERATE n
//                      INPUTS      n
//                      OUTPUTS     n
//                      PLAYBACKS   n
//                      ANALOGFIRST
```

- 10) To see the description strings the hardware manufacturer has provided for the audio channels (and thus determine if ANALOGFIRST will work) you can now do CONFIG GET CHANNELINFO once the interface is open. This will produce a multi-line response, listing the description strings for all of the input and output channels, in order. Note that all of the channels on the interface will be shown, even if the license limits you to less than all of the available channels.
- 11) To see the description strings for the channels actually in use on the interface (and to verify that ANALOGFIRST did what you wanted it to do), you can use CONFIG GET CHANNELMAP. This will produce a multi-line response that will show the internal channel number, the interface channel number, and the description string for that channel. The output is limited to the number of channels actually in use, which may be less than the physical number of channels available on the interface.
- 12) If the dongle is set to allow any operating mode, a CONFIG SET INTERFACE with no mode parameter will now open the interface in NORMAL mode. Previously it defaulted to AUDIOBOX mode, and you would have had to override that to get normal mode.
- 13) The CONFIG SET INTERFACE command with no channel count modifiers will now attempt to open the maximum licensed number of channels by default. Previously it defaulted to 16 channels.
- 14) Replaced popup message about "initialization failed" with a balloon message on the tray icon. This will no longer require clicking on OK to continue working. Also eliminated a possible popup message when setting interface parameters if the set failed. This has been replaced with a log message and a balloon message.
- 15) Multiline responses from various commands will no longer have an extraneous blank line following the end of the response.
- 16) Multiline responses are now formatted correctly in the history log.
- 17) If there is no dongle present there was an extraneous blank line in the history log on CONFIG SET INTERFACE. This is now fixed.
- 18) The DELAY parameter for channels now accepts the following syntax:
DELAY <time> FADETIME <fadetime>;
Previously "DELAY <time>" was only permitted if none of the other delay

parameters were present; to fade a delay you had to say
DELAY TIME <time> FADETIME <fadetime>;

- 19) When doing multiple slow delay fades with large delay values on an output channel, cumulative roundoff error could eventually end up with an incorrect delay value, sometimes producing audible spatter. This has been fixed.
- 20) All input, output, and crosspoint channels now do delay fades as temporary pitch shifts while the fade is running. Previously only output channels did this. This replaces the delay fade form that skipped and jumped to try to not do a pitch shift on the fades. That form of fade would still sound like a pitch shift under many conditions, and additionally had severe noise problems under many conditions.
- 21) Playback pitch change was broken in recent versions and was not setting the correct semitone value requested.
- 22) Playback pitch and speed changes had gotten all screwed up in recent releases, and are now fixed. Playback pitch and speed are separate controls for the same thing: the speed the track will play at. Both playback pitch and playback speed will change how fast the track will play. The only difference between them is "speed" is a multiplier of track speed, where 1 is normal track speed, 2 is double speed, and 0.5 is half speed. Pitch on the other hand is speed expressed as musical pitch in semitones. So a pitch of 0 is normal speed, a pitch of 12 is double speed, and a pitch of -12 is half speed.

Speed and pitch for playback are multiplicative. If you double the speed and take the pitch down an octave, the track will play at normal speed, because $2 \text{ times } 0.5 = 1$.

Speed and pitch changes to a playing track are ephemeral. They will be remembered as long as the track is playing. But when the track stops playing and is restarted, any speed or pitch changes made during the last playback are forgotten.

Sometimes it is desirable to set a speed or pitch variation to be used on the entire playback of the track, and you want to keep that modification even if the track is stopped and restarted. There is a special way to do this. If the speed or pitch parameter is entered on the same SET command with the file name, then the speed or pitch change will be remembered and reused every time the track is played. This speed variation can be overridden with other speed and pitch commands while the track is playing. But as mentioned above, these extra changes to the playback speed will be lost if the track is stopped and restarted. Only the speed or pitch change entered on the initial SET command that loaded the track file will be remembered.

- 23) CONFIG GET INPUTS, CONFIG GET OUTPUTS, and CONFIG GET PLAYBACKS
are now valid before the ASIO interface is opened. When requested before the interface is opened, these commands will return the licensed number of channels possible. After the interface is opened these commands will return the actual number of each type of channel available.
- 24) Included new dongle files with more error recovery abilities.

- 25) Increased text fields for dongle info from 50 characters to 100, since the dongle will allow at least 64 characters.

Version 1.0.28

- 1) SM-S will now show a tray icon rather than a normal program taskbar icon when it is minimized. You can use the tray icon to display the window, show the About Box, or exit the program.
- 2) If we fail to connect to an interface, and the driver name starts with the string "MOTU", we will not do a messagebox about the driver failure, since the MOTU drivers already do this themselves. We don't need two annoying message boxes when one will do.
- 3) In earlier versions if we failed to connect to the driver we could insome cases crash shortly after. This has been fixed.
- 4) If the driver fails to open, a balloon tooltip will show over the icon in the lower right corner of the desktop. This will give the operator a hint as to why sound might not be coming out even though SM-D seems to be running cues.
- 5) Fixed a small memory leak that could sometimes occur under very unusual conditions.
- 6) The positive gain limit for a submaster was 20dB,consistent with all other gain controls. However SM-A can send positive submaster gains up to 47.25 db. Changed the limit for positive submaster gain to 50db. The range is now -160 to +50. In AB mode or CC mode the overall gain of the control point will still be limited to +0db maximum.

Version 1.0.27

This is a major new version of SoundMan-Server which reads and generates SMPTE time code and has a large number of timecode manipulation, chase and lock capabilities.

Here is the summary of this release, provided by Loren:

This version of SM-S can both generate and read SMPTE timecode, and can lock any number of playback channels to the input timecode. You can also lock the timecode generators to incoming timecode from a reader, so you can either regenerate timecode or you can generate timecode of a different form locked to the input timecode (for instance, generate 24 fps timecode from 30 fps timecode).

Note that in this version the reader and generator deal with SMPTE, *** NOT *** WITH MTC!

SMPTE readers are "fake output channels". There are two of them, on channels O1000 and O1001. You can route any input channel or for that matter playback channel to either of the readers. You should never route more than ONE source to a given reader if you want it to decode correctly

SMPTE generators are "fake playback channels". This lets you start and stop them in synchronism to other playback channels, and to set speed and pitch and starting time using "track" parameters, just as you might do if you were playing back a timecode track recording.

There are two SMPTE generators, on channels P1000 and P1001.

Playing back multiple tracks in sync to timecode requires the use of a GROUP. There are a number of group channels with names starting from GROUP 0 or G0. The group is the ringmaster that receives incoming timecode from a timecode reader and makes sure that all of the playback channels in the group stay in sync.

To make this work you first have to declare a group that has a number of playback channels in the group, and which also has a SMPTE reader. You also specify the timecode lock mode and starting timecode value for the tracks. All tracks in a group need to start at the same timecode value, and tracks locked to timecode position cannot be set to loop. If the track looped there would be multiple possible timecodes for each position in the track. This is not currently allowed.

There are two ways you can lock tracks to timecode. You can lock them by pitch/speed, or you can lock them by time.

Sometimes all you need is to make sure that your playback channels will play at the same speed as some external device, like a video projector, but you can start and stop the sound manually. If this is what you want, you want to lock to timecode speed. This lets you start and stop tracks manually, and the tracks can loop. However, they will faithfully follow the incoming SMPTE playback rate.

More commonly you want to lock the playback tracks to a constant time position. When you do this you can set all of the tracks to be locked (in the group declaration) and start the tracks. If timecode is not present, or it is before the track starting time, the tracks will not play. Once the timecode appears and is stable and within the track range the tracks will play to timecode. You can manually stop the tracks even if timecode is still playing, and they will stop. When you start them again there will be a short delay while they seek to the current position for the current timecode value.

Audio will only follow timecode that is running forward. If the timecode stops, or starts running backward, or runs at a constant frame number, the audio will not play. Once the timecode is running forward and is within the track time range the tracks will play IF they have previously received a PLAY command.

When playing tracks to timecode, you do not send the Play command to the individual tracks as you would normally do. Instead, you tell the group to play, for instance, PLAY G0.

The commands necessary to play to timecode and all specific new commands are described below.

1) There are now two signal generators. Previously there was a single generator. Both signal generators are identical, and can be used to generate two different tones or other wave shapes routed to different outputs. The SET GENERATOR and GET GENERATOR commands have been enhanced to allow you to specify the generator of interest. The syntax now is:

```
SET {GEN | GENERATOR} [channel number]
```

```
    SHAPE SINE|SQUARE|TRIANGLE|WHITE|PINK
    FREQ|FREQUENCY fpt 5..24000
    SWEEP PAUSE|STOP|RESUME|min max time
```

```
GET {GEN | GENERATOR} [channel]
```

```
    SHAPE
    FREQ|FREQUENCY
    SWEEP [MIN|MAX|TIME]
```

This is identical to the previous syntax, with the addition of the optional generator channel number following GEN or GENERATOR. If the channel number is not given it will default to 1000, which is the first generator. If the channel number is given, it must be either I1000 or I1001, as those are the two signal generator channels.

2) There are now two timecode generators on channels P1000 and P1001. The timecode generators can be used independently to generate two different timecode streams. The timecode streams can have different time bases, or can have different timecode types. For instance, you could generate SMPTE at 30 frames/second from one generator and 24 frames/second from the second generator if you needed to lock a projector and a lighting console to the same time, but they required different frame rates.

The timecode generators appear to be playback channels. Since they are playback channels they can be started and stopped with PLAY and STOP commands, just like real playback channels. This lets you generate timecode in synchronism with audio playback. The timecode generators can start, stop, pause, loop, and change speed just like playback channels, so can track all movement actions identically to playback channels.

The timecode generators can generate many different forms of timecode. They can also generate pure sine waves, so can be used as an additional two signal generators if this is desired. By default the generators generate full output level. The command syntax for the timecode generators is similar to the setup for normal signal generators:

```
SET {TCGEN | TCGENERATOR} [channel number]

    FREQ|FREQUENCY fpt 5..24000
    CODE
        SINE
        SMPTE30
        SMPTE30D | SMPTE30DROP
        SMPTE25
        SMPTE24
        IRIG A [INV|INVERTED]
        IRIG AMOD [INV|INVERTED]
        IRIG B [INV|INVERTED]
        IRIG BMOD [INV|INVERTED]
    START [TIME|TC] <start time> [AT [TIME|TC] <offset>]
    DATE yy/mm/dd
    DAY nnn
    USERBITS nnnnnnnn
    SPEED fpt number
    RUNNING YES | NO | ON | OFF
```

```
GET {TCGEN | TCGENERATOR} [channel]
    FREQ|FREQUENCY
    START {TIME | TC}
    CODE
    TIME
    DATE
    DAY
    USERBITS
    SPEED
    RUNNING
```

The frequency should only be set if you are generating a sine wave. When you set any other code type the frequency is set automatically to the correct rate for that timecode type. It is possible though to manually set the frequency after setting the timecode type. This will probably cause the timecode to run off of the correct frequency, but there may be times when that is useful. The timecode type generated are:

```
SINE      A plain sine wave
SMPTE30   30fps non-drop SMPTE
SMPTE30D  29.97fps drop frame timecode
SMPTE30DROP 29.97fps drop frame timecode
SMPTE25   25fps SMPTE timecode
SMPTE24   24fps SMPTE timecode
IRIG A    1000Hz IRIG-A pulsed timecode
IRIG AMOD 10KHz IRIG-A sinewave modulated timecode
IRIG B    100Hz IRIG-B pulsed timecode
IRIG BMOD 1000Hz IRIG-B sinewave modulated timecode
```

SMPTE timecodes are most commonly used for synchronization between show equipment of various forms. IRIG is an older timecode form that is very good for long-range synchronization, between hundreds of feet and for some codes, tens of thousands of miles. It tends to be more resistant to noise and distortion

than SMPTE timecode. However, it is polarity sensitive to decode correctly. Since the output polarity of an audio channel is not necessarily predictable, you can generate the IRIG codes either positive or inverted.

The TIME field is used to set or get the starting (or current) generation time. The time will start from zero by default. If the playback channel for the generator has a TRACK START TIME specified, the TIME field will be the timecode value corresponding to that relative position. If no track start time is given, the TIME timecode is the time generated at the front of the track.

The DATE field is used to set or get a date field. This will be placed into the USERBITS field for SMPTE. The field will be validated to insure that it is a valid date. If the DATE and USERBITS parameter are both used and DATE occurs second, it will override the USERBITS value. The date value is initialized to zero by default, and is not automatically rolled over.

The DAY field takes a Julian day number between 1 and 366. This will be inserted into the Julian day field for IRIG timecode. This value is automatically incremented when the time rolls over from 23:59:59 to 00:00:00. It will also automatically roll over from 366 to 1, but not from 365 to 1.

The USERBITS field will place an arbitrary hexadecimal value into the SMPTE timecode user bits field. This will override any value set with the DATE parameter if it occurs after the DATE parameter. For IRIG timecode, the low 27 bits of the userbits value will be placed in the "control bits" field in the timecode value.

The SPEED value can be used to tune timecode generation to match some external source that is running at the wrong speed. Normally pitch = 1.0, which is exact speed. A smaller value will slow down timecode generation and a larger value will speed it up. It is generally more practical to use the TRACK SPEED parameter on the generator channel than to use the SPEED parameter on the generator command.

The RUNNING value will start or stop the generator. The generator is stopped initially, and can be started from the generator command or from a PLAY or RESUME command, possibly along with other playback channels that it will then track.

- 3) The annoying flutter of the VU meters with a constant signal level has been corrected.
- 4) The PLAY command will now start all tracks at the defined start point for each track. Previously it always erroneously started from the first sample of the wave file, regardless of the defined starting position.
- 5) Fixed a very rare crash that could happen if you were positioning into a wave file and closed the file at the same time.
- 6) There are now two timecode readers on channels OUT 1000 and OUT 1001. You can route signal from any input (or playback) channel to either of the timecode readers. Typically you would have an incoming SMPTE (or IRIG) timecode on an input channel, for instance from an external SMPTE generator or other source. However, you might have a click track or other canned show where the SMPTE source was a recorded track, so you can also route from a playback channel to the timecode readers. You should never route more than one source channel to any single reader. It is not illegal to route multiple channels, but doing so will degrade the timecode decoding if there is noise on the other channels.

Obviously if more than one timecode source is fed to a single timecode reader at the same time, chaos will result, and in all probability the correct timecode values will not be read. The current timecode read by the first reader (channel O1000) is displayed on the debug monitor display for SoundMan-Server. The timecode from the second reader is not displayed. There are currently no options to set in the timecode readers. They decode the incoming timecode data, determine the general format, and then produce an output that can be used to synchronize playback channels and the timecode generators. NEVER lock a playback channel or timecode generator to a timecode reader if that channel is supplying the timecode to the reader! This will result in positive feedback, and the timecode will very quickly run off the tracks.

7) You can now lock a group of playback channels to a timecode reader. Note that the timecode generators are also considered to be playback channels, so you can lock the timecode generators to the timecode readers. You might do this to regenerate timecode, or to generate timecode of a different format slaved to incoming timecode. For instance, if you have 30 fps SMPTE coming in and you need to slave a projector to the timecode, you could generate 24fps SMPTE in one of the generators and lock that generator to the reader following the 30fps timecode. You lock the playback channels to a reader indirectly through one of the group channels. The group channel insures that all of the playback channels in the group will remain in sample-accurate sync while following timecode.

If you locked individual channels to the timecode reader, the channels would follow the timecode within a fraction of one frame, but sample accuracy between the channels could not be guaranteed. The SET GROUP syntax has been extended to deal with timecode locking. The syntax now is:

```
SET { G | GROUP } < nn >
    nn = group number 0 to 127 or group name
    NAME < alphanumeric group name >
    { CHANS | CHANNELS } < channel_list >
CLEAR
    remove all channels
{ TC | TIMECODE }
    READER { channel number }
    LOCK
    SPEED
    { TIME | TC } < timecode > [ AT { TIME | TC } value ]
    UNLOCK
```

Setting a group sets group properties. Group properties include the list of channels in the group, an arbitrary name for the group, and various timecode-following parameters.

By grouping channels you can send many commands to the group that you would normally send to a list of channels. In that sense they are somewhat like a VCA on an analog mixer.

You can also group playback channels that will play simultaneously to timecode. Here the group is more important, since it insures that all of the channels will remain exactly in sync as they follow the timecode variations.

Examples

```
SET GROUP G0 NAME BACKING_TRACKS CHANS P0-P7
SET G BACKING_TRACKS TC READER 1000 LOCK TC 01:00:00:00
PLAY GROUP BACKING_TRACKS
```

```
SET GROUP G0 CHANS P0-P7
SET G0 TC READER 1000 LOCK TC 01:00:00:00
PLAY G0
```

Either of the two above sets do exactly the same thing.

Another typical example might be:

```
SET GROUP G0 CHANS P0-3 TC READER O1000 LOCK TC 1:0:0:0 AT 0
```

There are two ways you can lock to timecode: you can just follow the speed or rate at which the timecode is running, or you can lock to the actual timecode position. Sometimes it is sufficient to insure that a

playback started at some random time will remain in sync with some other device, for instance a video playback. If the audio runs at the same speed as the video source the lock will be maintained over extended periods of time. If this is all that is required, you can do TC LOCK SPEED to insure that the playback and the timecode will run at the same rate. Note that the audio will NOT follow the timecode if the timecode value jumps, since only the speed of the timecode is being followed. If the timecode slows down the audio will slow down to match.

More commonly it is necessary to lock the audio playback to a particular time relationship in the incoming timecode stream. If you do this and the audio channel is in playback mode, the audio will start and stop if the incoming timecode stops. If the timecode jumps, the audio will also jump to follow the timecode. If the timecode is arriving continuously but is showing the same frame value over and over, the audio will be frozen at the current time position. It will begin playing the instant the timecode begins moving. Audio will not follow timecode that is running backwards. For that matter the timecode reader will not follow backwards timecode. The timecode must be running in the forward direction to be recognized. (Stationary timecode must be presented with the bits in the forward direction to be recognized.)

Audio will speed up or slow down as necessary to remain as close to the current timecode value as possible. However, timecode, especially when generated by a mechanical source like a projector, can have a lot of wow and flutter. The timecode reader aggressively filters the timecode values to eliminate flutter and to reduce wow to the lowest possible levels consistent with remaining in sync with the timecode. Note that the audio pitch will change as the timecode speed changes. Thus if the timecode is shuttled forward, you would expect the audio playback to increase in frequency during the shuttle movement. (The audio will not currently follow a backwards shuttle.)

If the timecode jumps, the audio will jump to follow it. Depending on the number of tracks being played and various other conditions, there may be a short silence while the audio files seek to the new position and re-lock to the timecode. Timecode jumps either forward or backward will be followed, as long as the timecode bits continue to run forward. When locking to timecode, you need to specify a timecode value for some position in the audio track.

Typically you would specify the timecode value for the front of the track. However, it might be that the important timecode position is 12.71 seconds into the track. So that you don't have to convert 12.71 to frames at the current timecode rate and subtract that from the desired starting timecode, the group syntax lets you specify both the locking timecode value and where it appears in the track. In the example given above, the first sample of the track file occurs at one hour of timecode. However you could have specified LOCK TC 1:0:0:0 AT TIME 12.71; This would have made the one hour mark occur exactly 12.71 seconds into the track file.

8) The various bar graph displays would freeze when the interface was closed. This is now fixed and they update continuously.

9) The IO usage bar graph no longer hangs non-zero when nothing is playing.

10) There are now two timecode display lines, one for each timecode reader.

11) Commands have been implemented to get the current speed multiplier and pitch multiplier from a playback channel. These commands have the format

```
GET {CHAN|CHANNEL} <channel-list> TRACK SPEED  
GET {CHAN|CHANNEL} <channel-list> TRACK PITCH
```

The full implemented syntax for GET TRACK is

```
GET {CHAN|CHANNEL} <channel-list>
```

```
TRACK FILE          # TRACK FILE "name"
```

START|STOP|REPEAT # TRACK WORD ON|OFF NOTIFY ON|OFF
TIME|TC # TRACK WORD TIME fpt | TC tc
LOOP # TRACK WORD LOOP ON|OFF NOTIFY ON|OFF
START|STOP|REPEAT TIME|TC # TRACK START|STOP TIME fpt | TC tc
SPEED # current speed multiplier
PITCH # current pitch multiplier
POSITION # current position in samples
TIME # current position in seconds
LENGTH # track length in seconds
STATUS # PLAY or STOP

Version 1.0.26

1) Changed the code that reads temporary dongle files to be more reliable.

Version 1.0.25

- 1) You can now copy/paste from the Response line in the control panel.
- 2) The window position is now saved in the registry rather than an ini file.
- 3) Added the dongle serial number to the About Box contents.
- 4) Changed copyright date to include 2008.
- 5) You can now copy and paste from the info panel in the About Box.
- 6) The operating mode indicator on the control panel is now reliable.
- 7) Changed the warning message when we can't set the desired sample rate from a popup to a log message. This keeps from screwing up programs that open SM-S remotely.
- 8) All single-line responses except OK or ERROR should now be terminated by a semicolon to make parsing easier. The caller receiving a response can tell if it is single-line or multi-line by making some simple checks:
 - a) If the first word is OK or ERROR it is a single-line response.
 - b) If the response line ends with a semicolon it is a single-line response.
 - c) In all other cases it is the first line of a multi-line response and the complete response will end with a line containing a single ".\r\n" in the first character position. Asynchronous responses (ones that are not the direct result of a preceding command being issued) are not yet implemented. However when they are, the first character of the first line of an async response will be an "@" character to flag the line as async. If it is a multi-line response only the first line will be flagged with an @ sign. It is expected that all async responses will be single line responses, but the program receiving one should check the rules given above to determine if the response is a single-line async response or the first line of a multi-line async response. All multi-line responses are guaranteed to be transmitted as a group. No other single or multi-line response will ever be mixed with another multi- line response.
- 9) If you try to open an audio file and request a track number that exceeds the number of tracks in the file, you now get an error message that clearly states the problem. Previously you got a generic "could not open file" error message, which was confusing since the file was in fact opened.
- 10) When closing the ASIO interface, we now lower the priority on thecallback thread first. This should prevent broken drivers locking up the entire system if they loop during the close attempt. However, SM-S itself will probably still get locked up in this case and have to be killed and restarted. Better SM-S than the entire system.
- 11) Added an extra half second delay in the process of closing an ASIO driver. The Maya driver takes a very long time to shut down the callback thread (which should have been complete when ASIOStop returned) and the callback thread would then die after the buffers had been freed. The extra delay gives the callback thread time to stop, and then close completes cleanly.
- 12) You can now give start, stop, loop, and repeat positions in samples as well as time. This can be handy for controlling programs that have accurate sample position info in the files being played. The sample positions are in terms of file sample rate, which may be different than the interface sample rate. The syntax for SET TRACK has been changed to add this new feature. The changed areas are:

SET CHAN <channels> TRACK

START

SAMPLES <samplenum>

TIME <time in seconds, floating point>
TC <time in timecode value>
<time in seconds, floating point>

STOP

SAMPLES <samplenum>
TIME <time in seconds, floating point>
TC <time in timecode value>
<time in seconds, floating point>

REPEAT

SAMPLES <samplenum>
TIME <time in seconds, floating point>
TC <time in timecode value>
<time in seconds, floating point>

LOOP FROM

SAMPLES <samplenum>
TIME <time in seconds, floating point>
TC <time in timecode value>
<time in seconds, floating point>

LOOP TO

SAMPLES <samplenum>
TIME <time in seconds, floating point>
TC <time in timecode value>
<time in seconds, floating point>

Note that either LOOP or REPEAT should be used for looping, but not both.

13) A "REPEAT <channels> NOW" command will now correctly loop back to the start position for the tracks rather than stopping the tracks.

14) Get channel gain for a crosspoint channel returned an incorrect channel number.

15) Get channel gaindb failed to indicate in the response that the value was in dB. The lead word is now correctly "GaindB" rather than "Gain" (which indicates a linear gain value).

16) Getting the channel gain in dB for a channel with zero gain returned an invalid gain number. It now arbitrarily returns a value of -144 for zero gain.

17) You can now GET the attributes of the signal generator. GET {GEN|GENERATOR} FREQ|FREQUENCY SHAPE SWEEP [MIN|MAX|TIME] FREQ or FREQUENCY returns the generator frequency. SHAPE returns the generator waveshape. This can be one of Sine Square Triangle WhiteNoise PinkNoise PinkSweep Sine, Square, and Triangle are pure tone waveshapes. If sweep is disabled, the generator makes a constant frequency. If normal sweep is enabled, the tone is swept from the minimum frequency to the maximum frequency in the given time, and then swept back to the minimum frequency in the same amount of time. The amplitude of the wave is constant across all swept frequencies. WhiteNoise and PinkNoise are random noise generator functions. WhiteNoise has equal amplitude at all frequencies. PinkNoise decreases in amplitude with frequency. PinkSweep is a swept sine wave that has an amplitude envelope matching the pink noise envelope. This can be used with a traditional spectrum analyzer to make various passband measurements. The PinkNoise function is more appropriate when using an RTA. SWEEP returns one of four values, depending on what follows the word SWEEP. SWEEP MIN returns the minimum sweep frequency SWEEP MAX returns the maximum sweep frequency SWEEP TIME returns the time in seconds for the sweep. SWEEP with nothing following it returns On or Off.

18) If a "config set interface" command is issued that has exactly the same parameters as the current open interface, the interface will no longer be closed and reopened, as there is no need to do that.

Version 1.0.24

- 1) Changed the dongle stuff to use the static library, so it is no longer necessary to have Rockey2.dll available.
- 2) Fixed config stuff to correctly allow more than 16 playback channels in modes other than AB mode.
- 3) Implemented a commandline lockout feature. `CONFIG SET COMMANDLINE LOCK key` `CONFIG SET COMMANDLINE UNLOCK matching-key` The commandline lock command will lock the commandline in the dialog and set the unlock key to match the 'key' parameter. This command is only valid when the command line is NOT locked. This command can be sent from any command source, not just from the command line. The commandline unlock command can also be sent from any source and not just the commandline itself. If the commandline is currently locked and the key on this command matches the key set with the previous commandline lock command, the commandline will be unlocked. When the commandline is locked, only `GET`, `CONFIG GET`, and `CONFIG SET COMMANDLINE UNLOCK` commands will be accepted from the command line. All other commands will return an error response of "Commandline locked". Also, the various buttons in the dialog, including the Close button are locked out until the interface is unlocked.
- 4) Made sure that the droplist with the ASIO device name is set in all cases when setting the interface from the command line.
- 5) Added a `CLOSE <channel-list>` command to stop playback on the list of channels and then close the files that were in use. Normally files are kept around in case they might be used again shortly. However that makes it difficult to delete one of the files if you want to replace it with a new version of the file.
- 6) Fixed some bugs in buffer LRU list handling that could result in crashes in rare cases.
- 7) Fixed some bugs in keeping track of files when the same file is open on multiple channels.
- 8) It was possible to play a channel with no file loaded on it and not get an error. When you later loaded a file onto the channel it started playing immediately, which was generally unexpected. Play now checks that there is a file present on the channel before attempting to play and will send an error response if the channel isn't able to play.
- 9) Sped up the code that computes VU values.
- 10) Reduced signal generator overhead when not generating output (which is the usual case).
- 11) SM-S will now accept temporary license files in place of a dongle. These files only live for a few days, but it is sufficient time to receive your actual dongle in the mail. A temporary license file is tied to one system and cannot be moved between systems as a real dongle can.
- 12) When running on a temporary dongle file, this is a **BIG ANNOYING MESSAGE** that comes up to remind you that you are on borrowed time and it is running out. It will show the number of days you have remaining until the system reverts back into demo mode.
- 13) To get a temporary license file you must supply RSD with the machine id for the system you wish to license. You can get this number from the About Box

when the system is running in demo mode. The number is not available when you are running with a dongle or a temporary license file.

14) Fixed a potential deadlock that could happen during startup.

15) Fixed an annoying complaint that could occur during ASIO open if the dongle had an incorrect sample rate configured.

16) Fixed some problems with gain setting that made recent versions not work with the Waterworld show.

Version 1.0.23

- 1) If an output channel wasn't accumulating VU samples, we would erroneously keep repeating the last value VU sample rather than correctly returning 0.
- 2) Reimplemented the TCP message receive path to drastically reduce processor usage to something reasonable. Submaster changes in SM-A no longer swamp the entire system with overhead.

Version 1.0.22

- 1) The license info now shows up in the About box, the splash screen, and is logged when the interface is opened.
- 2) If the interface is locked into a single operating mode by the licenseinfo the interface cannot be switched to some other mode.
- 3) Changed the security dongle stuff to be able to decrypt the dongle contents on hopefully all MS systems, and not just most of them.
- 4) The Close button again works as God, Microsoft, Apple, and Linux expect a Close button to work -- it will Close the program. People that just want to MINIMIZE the window are directed to use the MINIMIZE BUTTON in the title bar, which has been designed specifically for this purpose, as stated in all of the system UI design manuals.
- 5) Fixed a problem where crosspoint gain was not set during a fade, but only at the end of the fade time (yuk).
- 6) Fixed another problem where fading a crosspoint up from 0 over time didn't work at all. Now it does.

Version 1.0.21

1) Implemented dongle code.

Version 1.0.20

1) Added an ugly hack to allow invalid channel numbers < 16 when talking to SM-A. As long as at least one valid channel is given we will act on the command, ignoring the invalid channels.

Version 1.0.19

1) Implemented an assortment of channel status request commands. This is not an exhaustive set of possible status requests, but it covers many of the most common needs. A channel status request can request a common status type (such as gain or delay or mute status) for multiple channels of any type. The response for all channels will be returned on a single line.

All status response lines have a common format:

<status_type> <channel_id> = <status_value> ... ; <newline>

The status_type identifies the type of channel status on this line. All channels on the line are returning the same kind of status. The status type is always a single word with an initial capital letter and usually the remainder of the word in lower-case. Each channel, even if there is only one, is identified by the short form of the channel identifier. This consists of one or two letters and the appropriate channel numbers. The status value will be appropriate for the status being returned.

For numeric values this can be an integer or a floating-point number, as appropriate. For an on/off status such as the mute status of a channel it will be ON or OFF. For the channel name it will be the name string in all uppercase, or "None" (without the quotes) if the channel does not have a name assigned.

The following table shows the currently valid channel status commands, the response status_type value, and indicates the type of value to expect. All requests are of the form

GET CHANNEL <channel_list> <word>

Request	Response	Type	What
NAME	Name	string	channel name or 'None'
GAIN	Gain	floating	the channel gain between 0..N
GAINDB	GaindB	floating	the channel gain in dB
GAINMIDI	GainMidi	integer	gain between 0..127
PHASEREVERSE	PhaseReverse	ON OFF	channel polarity status
MUTE	Mute	ON OFF	mute status
SOLO	Solo	ON OFF	solo status
DELAY	Delay	floating	the delay in seconds
TRACK FILE	TrackFile	"string"	the full path to the file
TRACK STATUS	TrackStatus	PLAY STOP	playback status
TRACK POSITION	Position	integer	playback position in samples
TRACK TIME	TrackTime	floating	playback position in seconds
TRACK LENGTH	TrackLength	floating	track length in seconds
TRACK START TIME	TrackStart	floating	track start offset in seconds
TRACK STOP TIME	TrackStop	floating	track stop offset in seconds
TRACK REPEAT TIME	TrackRepeat	floating	loop-to point in seconds

2) The GET VU command is implemented. It returns average and peak VU readings for input and output channels, but not for crosspoints. The VU values are expressed as integers in the range of 0 to 255. These are suitable to be fed

into a VU meter with this range from minimum to maximum. The VU meter should be linear over this range; the values have been pre-warped to display reasonably.

The command has the format

```
GET VU <channel_list>
```

The response has the format

```
VU <channel_id>=<average>:<peak> ... ;
```

Avoid doing VU requests more often than every 100ms or so. VU processing is expensive, and the time is generally better used for other things.

3) Converted the Server to run in demo mode if there is no dongle. In demo mode the server runs in AudioBox-emulation mode with 2 input channels, 2 output channels, and 4 playback channels, at no more than 48K sample rate.

4) CONFIG SET INTERFACE now allows the sample rate to be specified. Previously a separate setup command was required.

```
CONFIG SET INTERFACE number | "name" INPUTS count OUTPUTS count PLAYBACKS count  
SAMPLERATE speed MODE NORMAL | AUDIOBOX | ABEMULATION | COMMANDCUE
```

5) Various CONFIG GET commands exist. This documents the commands and what they return. These commands do not require an open interface:

CONFIG GET VERSION Returns the current SM-S version number string: Version 1.0.19.0 for example

CONFIG GET INTERFACES Returns a list of the available ASIO interfaces terminated by ".":

```
Interfaces 2 Interface 0 "ASIO 2.0 - Maya 7.1" Interface 1 "MOTU FireWire Audio"
```

CONFIG GET INTERFACE [n] Get information on the specified interface number:

```
InterfaceInfo 1 "MOTU FireWire Audio" Inputs 18 Outputs 18 DefaultSampleRate 48000
```

These commands require an open interface

```
CONFIG GET INTERFACE
```

When issued without an interface number this returns info on the current interface, including any limitations on the number of channels:

```
InterfaceInfo 1 "MOTU FireWire Audio" Inputs 2 Outputs 2 DefaultSampleRate 44100
```

```
CONFIG GET INPUTS
```

Returns the number of inputs available:

```
Inputs = 2
```

```
CONFIG GET OUTPUTS
```

Returns the number of outputs available:

Outputs = 2

CONFIG GET PLAYBACKS

Returns the number of playback channels available:

Playbacks = 4

CONFIG GET DEMO

Returns 1 if in demo mode, 0 otherwise:

Demo = 1

CONFIG GET DONGLE

Returns 1 if the dongle is detected, 0 otherwise:

Dongle = 0

CONFIG GET MODE

Returns the matrix operating mode:

Mode = AUDIOBOX

Possible values are NORMAL, AUDIOBOX, COMMANDCUE.

CONFIG GET SAMPLERATE

Gets the current interface sample rate:

SampleRate = 44100

6) Output channels now zero the VU reading when muted, as they should.

7) Input VU readings with SM-S are post-fade rather than pre-fade as they are in a real AB. This has always been the case, this just documents that this is what is intended to happen.

8) SM-S will now show demo mode and missing dongle information in the titlebar.

9) The log window will now expand wider when dropped down, making it easier to read long log lines. If SM-S is too near the right edge of the window the dropdown will be trimmed at the right edge of the screen so that the scroll bar doesn't disappear.

Version 1.0.18

1) Changed a lot of strings to SoundMan-Server from SoundMan Server.

Version 1.0.17

- 1) Added code to insure that the ASIO callback thread is running at highest priority after lowering the GUI thread to normal priority. Some insane ASIO drivers seem to somehow use the main thread as the ASIO driver thread!
- 2) When the close button is clicked and the interface is open, just minimize the window instead of asking if you want to end the app.

Version 1.0.16

- 1) Extended SET MATRIX ROW to allow GAINMIDI and MIDI gain values in the range of 0..127. This simplifies some code in SoundMan-Assistant.
- 2) Extended SET MATRIX ROW to allow a FADETIMEIFADETC parameter on the end to set the gain for all specified crosspoints at a non-zero rate.
- 3) Cleaned up the startup sequence so that we don't try to lay out the VU meters three different times.
- 4) Found that the VU meters weren't laid out correctly if the window was iconic when the meters are laid out. Added code to redo the layout when the window is finally displayed.
- 5) Slightly enlarged the size of the display for the connected ASIO device so that more of the text should be visible.
- 6) Added some experimental timecode decoding logic. This is not yet usable.
- 7) The ECHO driver for the GINA 24 sets the GUI thread to realtime priority. This results in audio breakup any time the interface is touched. Added code to change it back to normal priority like it should be.

Version 1.0.15

- 1) Playback channel soloing didn't work. It now does, and properly cross-mutes with the input channels.
- 2) SET CHAN P0-3 TRACK "NAME" will now correctly load the same file onto all of the selected playback channels. Previously only the first channel would have been loaded.
- 3) Fixed a timing window that in very rare conditions could cause a crash.
- 4) Under some conditions when starting many channels at once, some would not always start.
- 5) SET MATRIX ROW will now accept a row number, an input channel name, or a playback channel name to designate the crosspoint row to be set.
- 6) A crosspoint can be named, and referred to by name in commands that take a crosspoint identifier. The crosspoint can also be referred to by input_name.output_name, or by the usual number.number.

Version 1.0.14

- 1) "set chan i1 port none" did not work correctly.
- 2) Audio playback logic has been moved to detachable input nodes from the main playback channels. This allows any input channel to attach to a playback stream in Command Cue mode. It also makes it easier to properly implement multiple sample formats for wave files.
- 3) Channels can now be named. The syntax is SET CHANNEL <chan> NAME <name>. The name string cannot contain spaces or special characters other than an underscore and must start with an alphabetic character.
- 4) Port selection in Command Cue mode can now be done by name.
- 5) Channels in commands may now be specified by name as well as number. For an input or output channel the name can be given as
 <type> <channel name> or just <channel name>
For a crosspoint or playback crosspoint channel the syntax can be
 <type> <input name>.<output name> or <type> <crosspoint name>
Note that you can leave the channel type off for an input or output channel but you have to give it for a crosspoint or playback crosspoint channel.

Channel names do not have to be unique. However, if there is more than one channel of a given type with the same name, only the lower-numbered channel will ever be found by name. It is perfectly all right though to give both an input and an output channel the same name. The input channel will be found if no <type> is given, and either can be found if the correct <type> is given.

Example:

```
set chan in 1 name fred
set chan in 2 name wilma
set chan in 3 name barney
set chan out 1 name dsl
set chan out 2 name cluster
set chan out 3 name dsr
set chan wilma gaindb -2
set chan fred gaindb -5
set chan barney gaindb 0
set chan x wilma.dsl gaindb 0
set chan x fred.dsr gaindb 0
set chan x barney.cluster gaindb -2
set chan dsl-dsr gaindb 0
set chan cluster mute on
```

- 6) Added 'set <channel> track path "pathname"' command. This lets you set a path to all of the files that will be played on this particular channel so that you don't have to enter a full path on every file. You can still override the path by giving a full path as a file name. You can clear the path by setting it to an empty string.

- 7) The window position will now be remembered on application close and the position will be restored the next time the program runs.
- 8) Added a "CONFIG CLEAR LOG" command to clear the history buffer. This can be useful for applications that leave the server up for extended periods of time. For instance, a command could be sent to clear the log at midnight.
- 9) Added 128 group masters, named G0 to G127.
- 10) Added SET GROUP Gnn CHANNELS <channel list> to allow groups to master channels and other groups.
- 11) Added G | GROUP nn syntax to the channel list syntax so that you can mix normal channels and groups of channels in the same command.

Version 1.0.13

- 1) Again recognize "mode abemulation" to make the old SM-AB setup happy.
- 2) Fixed crosspoint GAINMIDI to set the live gain rather than the submaster gain if we aren't in CommandCue mode (and didn't ask for a log fade.)
- 3) Fixed a problem where we re-laid out the vu meters on every command from SM-AB!
- 4) The flag to do vu processing was not being managed correctly, so vu meter processing was random.
- 5) There was a race condition in shutdown that could cause crashes when deallocating the vu meters.

Version 1.0.12

- 1) Argh! The check for valid processor hardware got optimized out!

Version 1.0.11

- 1) Made the 30-day timeout actually work correctly.
- 2) Added VU meters on the output channels.
- 3) Various unrecorded simple bug fixes during runup to LDI 2006 in Las Vegas.
- 4) Changed the copyright to 2007.

Version 1.0.10

- 1) Fixed a largish handful of bugs all over the place. Most notably made sure that the floating point rounding mode is set correctly in all cases.
- 2) Now parse simple wave file headers correctly and decode the playback speed and number of tracks. Works for mono or stereo files with the standard sample rates.
- 3) Made 'config set samplerate' work correctly.
- 4) Added syntax to the 'set chan xxx track file "xxx" track n' command to let you pick which channel in the wave file will play back.
- 5) Added some basic Command Cue matrix support abilities.

For hard input channels you can reconnect the input control cluster to another inputs in a mix and match form. Any control cluster can be assigned to any input and you can have more than one cluster on a single physical input. This corresponds to the typical Selector usage in Command Cue. Note this only works for hard inputs, not playback inputs or the signal generator!

The crosspoint row has been split off of the input, and you can now assign an input cluster's output to any crosspoint row, or to no row at all. This corresponds to the typical Assignor usage in Command Cue. Again, you can have multiple input clusters assigned to a single crosspoint row, and crosspoint rows with no inputs.

The default initialization is to map all physical inputs to an input control cluster and map that to the associated crosspoint row. If no changes are made in mapping the matrix will work normally.

- 6) Changed the 'config set mode' syntax to allow more input modes. The old syntax was 'config set mode abemulation [on/off]'. The new syntax is 'config set mode [normallaudiobox|commandcue]'
- 7) Added 'set chan IN xx PORT n' to assign a particular hard input to this particular input cluster. The 'port xx' syntax is only valid on input channels, and only in CommandCue mode. Note that you cannot assign the input cluster to NO physical input as you can in Command Cue. You can get the same effect by muting the channel.
- 8) Added 'set chan [INIPLAYBACK] xx ROW [n|NONE]' to select the crosspoint row to be used for this particular input or playback cluster. This is only valid for input and playback channels and only in Command Cue mode. By default each input cluster is mapped 1:1 to the same crosspoint row number.
- 9) Adjusted the log fade volume curve timing to match the old Command Cue hardware. Command Cue on a fade down from full by definition drops at 36db over the specified fade time. This means a complete fade to -144db will take 4 times the specified fade time. The time values for a CommandCue AutoPan fader should now match our time values and produce the same results.

- 10) Added a fade type of "Log" as a synonym for Exp or Exponential in the gain fade command format.
- 11) Set Matrix has been modified to set some or all of the crosspoints in a single matrix row to individual values for each point. The general syntax is: SET MATRIX ROW n [GAIN|GAINDB] channel [TO channel]@level...
If neither GAIN nor GAINDB is specified the gains are in dB by default.
- 12) In CommandCue mode, a new command has been added to control the AutoPan fader separately from the main channel gain:
SET CHAN IN n AUTOPAN ON|OFF|UP|DOWN
LIN|LINEAR|LOG|EXPI|EXPONENTIAL
FADETIME time
FADETC timecode

Version 1.0.9

- 1) Increased max playback channels to 1024.
- 2) Some assorted minor performance improvements.

Version 1.0.8

- 1) The server could crash in the socket handling stuff under conditions that I don't understand. Fixed the writes to be synchronous, which may help. Also added debug code to spit a message to DbgView if the socket is called from the wrong thread.
- 2) Changed eq band numbers to be 1-relative since I could never remember that they were zero relative before.
- 3) Changed the eq to allow positive gains up to +100db.

Version 1.0.7

- 1) Added commands to the parser to allow the signal generator to be configured from the interface.
- 2) Added the ability to sweep the signal generator over a frequency range in sine/square/triangle modes. This is a linear sweep, which is the most useful kind with an FFT analysis program as all signals will show the same level on a flat response.
- 3) Added the ability to do a 'pink sweep'. This will sweep a sine wave across a frequency range, but decreases the level at -3db/octave to match normal pink noise response.
- 4) Improved the pink noise generator. The ripple is now around .05db from a straight line rather than the 1.5db ripple previously.

Version 1.0.5

- 1) Fixed a bug in pitch and speed parsing.
- 2) Fixed a hang problem where a failed IO request would lock things up.
- 3) Fixed problems with playing off the end of a track, and with a track not looping when only the resume point has been set. Seemingly contrary to the command set documentation, the AB will loop from the end of the track to the resume point with only a resume point set.

Version 1.0.4

- 1) If the stop point was disabled the track wouldn't play. Fixed.
- 2) Changing the ASIO driver from SoundMan-AB might not load the driver when it should. Fixed.
- 3) Improved parsing of track start and stop times.
- 4) Now allow a submaster gain in dB for any channel or crosspoint. This was needed to make submasters and controllers work correctly in SoundMan-AB.
- 5) Channel and crosspoint gain is not allowed to go above unity in AudioBox emulation mode.

Version 1.0.3

- 1) Slightly reduced the processing overhead for several of the live channel input routines.
- 2) Added yet more debugging log messages to the process of opening an ASIO device.
- 3) Changed the parameters to the ASIOInit call to only put a window handle in the sysRef field if the driver name starts with "M-Audio", otherwise leave the sysRef field zero. This is documented as zero on Mac platforms, and a valid window handle on Windows platforms. NULL is a valid window pseudo-handle (meaning 'the desktop'), but the next-to-latest version of the M-Audio drivers will crash on exit if the handle isn't non-zero.

On the other hand, it turns out Digigram ASIO drivers seem to think this field is some sort of an undocumented device index, and the driver will crash if it isn't zero!

- 4) Previously if we couldn't set the interface sample rate to 48000 we would give up. Now we check the interface sample rate after attempting to set it. If the rate is something other than 48000 we pop up a warning message but allow the use of the interface anyway.

A possible reason the interface sample rate might not be settable is if the interface is working off of an external clock source. Depending on what the interface reports to us as a sample rate this might work out OK, or might produce totally bogus sound. The user can decide for himself.

- 5) After just an amazing amount of work, fixed a problem where loading a new track that was the same as the current track would not start playing from the front of the file as it should, but would just continue to play without stopping.
- 6) Previously playing a short track and then a long track on the same channel would end up stopping at the time the first track ended. This is fixed, and the longer track will now correctly play to the end or stop point.

Version 1.0.2

- 1) No longer crash with a divide by zero calculating processor utilization.
- 2) SoundMan-AB no longer opens ASIO devices, possibly contending with the server.
- 3) Removed some probably unneeded code that was triggering a bug in the ECHO Layla/Gina24 drivers. The drivers now seem to come up and work fine, at least on my system.
- 4) Added logging of AsioResyncRequest, which probably indicates data loss occurred. Not all drivers will make this message, but the Echo drivers do.
- 5) Added logging code to log all possible bailouts from sound device initialization. This will make it easier to see why some interfaces don't work.
- 6) Fixed a problem with crosspoint linking and delinking that could end up getting the links broken and locking up the system.
- 7) Added an option to the interface to set the server to "below normal" priority instead of the usual Realtime priority. At low priority the audio can be expected to click and burp as you do other things. However, if the server gets itself into an internal loop (as happened sometimes in version 1.0.1 when changing crosspoint gains) the system won't get locked up and you will be able to kill the server.
- 8) Added a button to copy the command history log to the clipboard. From there you can copy it into a Notepad or mail message window. The command history could help me solve problems when they occur.
- 9) Selecting "None" for the ASIO interface won't spuriously try to open a driver with that name.

Version 1.0.1

- 1) Initial release of SoundMan Server.