
X32

Studio Mixing Applications

Windows Terminal/Command Line tools:

Windows Applications

Graphical Interface tools:

More Programs:

Other Platforms

Github repo

XAir Devices?

X32 emulator

X32PlayScript

X32GetScene / X32SetScene

X32Replay

X32_Command

X32Reaper

X32GetShow / X32SetShow

X32AutoMix

X32DeskSave / X32DeskRestore

X32CustomLayer

X32Midi2OSC

X32Commander

X32PunchControl

X32Tap

X32USB

X32GetLib, X32SetLib

X32Fade

X32CopyFX

X32FaderDisp

MidiOscIttt
Studio Mixing Applications

Windows Terminal/Command Line tools:

Windows Applications

Graphical Interface tools:

More Programs:

Other Platforms

Github repo

XAir Devices?

X32 emulator

X32PlayScript

X32GetScene / X32SetScene

X32Replay

X32_Command

X32Reaper

X32GetShow / X32SetShow

X32AutoMix

X32DeskSave / X32DeskRestore

X32CustomLayer

X32Midi2OSC

X32Commander

I made the acquisition of a **Behringer X32 standard** console in June 2014 for my music studio and gig needs.

X32PunchControl

Great unit! is all I can say. No problem with it, a pretty good sound, easy to manage and learn, with a comprehensive list of effects and a lot of capabilities, including being controllable through OSC protocol.

X32Tap

X32USB

X32GetLib, X32SetLib

IMPORTANT NOTE

All 'Windows' utilities below are designed and tested under Windows 10;
They should work with no issues on Windows XP.
Earlier versions of Windows are not supported

X32Fade

X32CopyFX

Windows 11 Users: I just moved to Windows 11 and found out the so-called "Console Window " behavior is different in Win 11 compared to previous versions of Windows. You will most likely want to set the "**Settings->Privacy&Security->for Developers->Terminal**" option to "**Windows Console Host**"

X32FaderDisp

MidiOscIttt

X32 OSC Protocol Document

I decided to spend a little time (well... initially :-)) understanding OSC programming aspects to interface with the X32, and have it do things for me. This work resulted in an updated, unofficial **X32 OSC protocol** document which I hope will help many others. [updated June 29, 2021]

X32ReaperAutoMate

While there are many very useful utilities proposed below, there is one that can really make your recording studio a lot easier to use, combining the ease of use of the X32 or M32 as a control surface and audio engine, REAPER as a DAW/recording system, and full automation support without the need to constantly use a mouse or even look at your PC! Check out **X32ReaperAutoMate**, this will change your life as a recording/sound engineer.

X32XLiveAutoMate

What has been possible with using REAPER as an audio source for X32ReaperAutoMate, has been ported with XLive as an audio source. The result is that you can record, organize, automate, mix your audio project right at the X32 desk, without the need for a DAW. You need a PC for recording and saving automation files for later use or sharing them with band members, but you'll hardly use your PC

Even better, both tools (Reaper and XLive versions) take advantage of the 500 possible cues of the X32 to organize your mix sessions into chained lists you can launch with the GO button. This is a great tool for theaters, musicals, or even for just organizing your work.

Other Utilities

Over time, I developed utilities for my own use and help other users. These utilities can run under different operating systems. Raspberry, Linux and Mac/OSX versions are available for some of them. Click on the links below to download the versions of the utilities corresponding to your platform.

With the exception of the AutoMate series of applications, all utilities below are distributed as Freeware for non-commercial users.

Commercial users should contact me before using or distributing these tools.

"Standard disclaimers apply"

Studio Mixing Applications

- **X32ReaperAutoMate**: Full mixing automation and loop capabilities for REAPER, using the X32 audio engine, the X32 as transport system, and more [2019]
- **X32XLiveAutoMate**: Full mixing automation and loop capabilities for X-Live add-on card, using the X32 audio engine, the X32 as transport system, and more [2019]
- **X32PunchControl:** Control your DAW while playing audio, recording or playing back X32 commands. [May 10, 2019]

Windows Terminal/Command Line tools:

- **X32 emulator:** The X32 you can carry in your pocket for developing apps or testing on the go. [May 28, 2022]
- **X32PlayScript**: A script file for Theater play-scripts. [May 5, 2016]
- **X32GetScene**: Get a Scene/Snippet file directly out of your X32. [Nov 17, 2019]
- **X32SetScene**: Interprets a Scene/Snippet file and set your X32 accordingly. [Nov 17, 2019]
- **X32Replay**: Record and play back a show (respecting time... of course). [Jul 17, 2020]
- **X32_Command:** A simple, yet very powerful tool for sending OSC commands and listening to X32 OSC data. [Feb 3, 2019]
- **X32FaderDisp:** Displays in real-time the X32 Fader dB value on their respective scribble [Nov 30, 2021]

Windows Applications

Graphical Interface tools:

- **X32Reaper**: 2-way communication/control X32<->Reaper - *see details below*. [Jan 11, 2020]
- **X32GetShow**: Get and save on your system the current show, cues, and all associated scenes and snippets. [Mar 8, 2020]
- **X32SetShow**: Load in X32 a given show, cues, and all associated scenes and snippets from a PC location. [Mar 8, 2020]
- **X32AutoMix:** A simple approach to Automatic Mixing tool for up to 32 channels. [Oct 28, 2017]
- **X32DeskSave/X32DeskRestore:** A utility to Save/Restore X32 State, Scene, Routing or Pattern based data sets. [Nov 17, 2019]
- **X32CustomLayer:** A utility to create and manage custom channel layers easily. [Sep 20, 2020]

- **X32Midi2OSC:** Have X32 receive OSC data from MIDI devices. [Jan 19, 2020]
- **X32Commander:** Have X32 commands control other devices through MIDI or OSC. [Jan 19,2020]
- **X32Tap:** "tap in" (manually or automatically) the time value of a delay effect of the X32. [updated Feb 20, 2023]
- **X32USBW**: A utility to play/load/execute files contained in the X32 USB stick [Feb 9, 2019]
- **X32SetLib/X32GetLib:** Manage X32 library, Channel, Effect or Routing Presets from a PC location. [updated Nov 16, 2019]
- **X32Fade:** A program to manage fade in and fade out of multiple levels / faders of the X32 - time controlled! [Feb 21, 2015]
- **X32CopyFX:** Reset or Copy sides or full parameters set from an effect to itself or another. [Nov 25, 2016]

More Programs:

There are more programs available than those described above with their respective description and link; Feel free to contact me.

- **X32Jog4Xlive**: Enabling audio Jog and Shuttle for X-Live! while using SD card audio. [Nov 25, 2017]
- **X32SetPreset:** Load in X32 a Channel, Effect or Routing Preset from a PC location. [Nov 17, 2019]
- **X32Ssaver:** An X32 LCD & LEDs "screen-saver"& "SOF" utility. [Sep 24, 2016]
- **X32SafeMutes:** A utility to prevent unwanted changes to X32/M32 Mutes. [Jul 31, 2016]
- **X32VoiceTrax**: A utility to let users manage their channel gain during recording sessions [May 11, 2018]

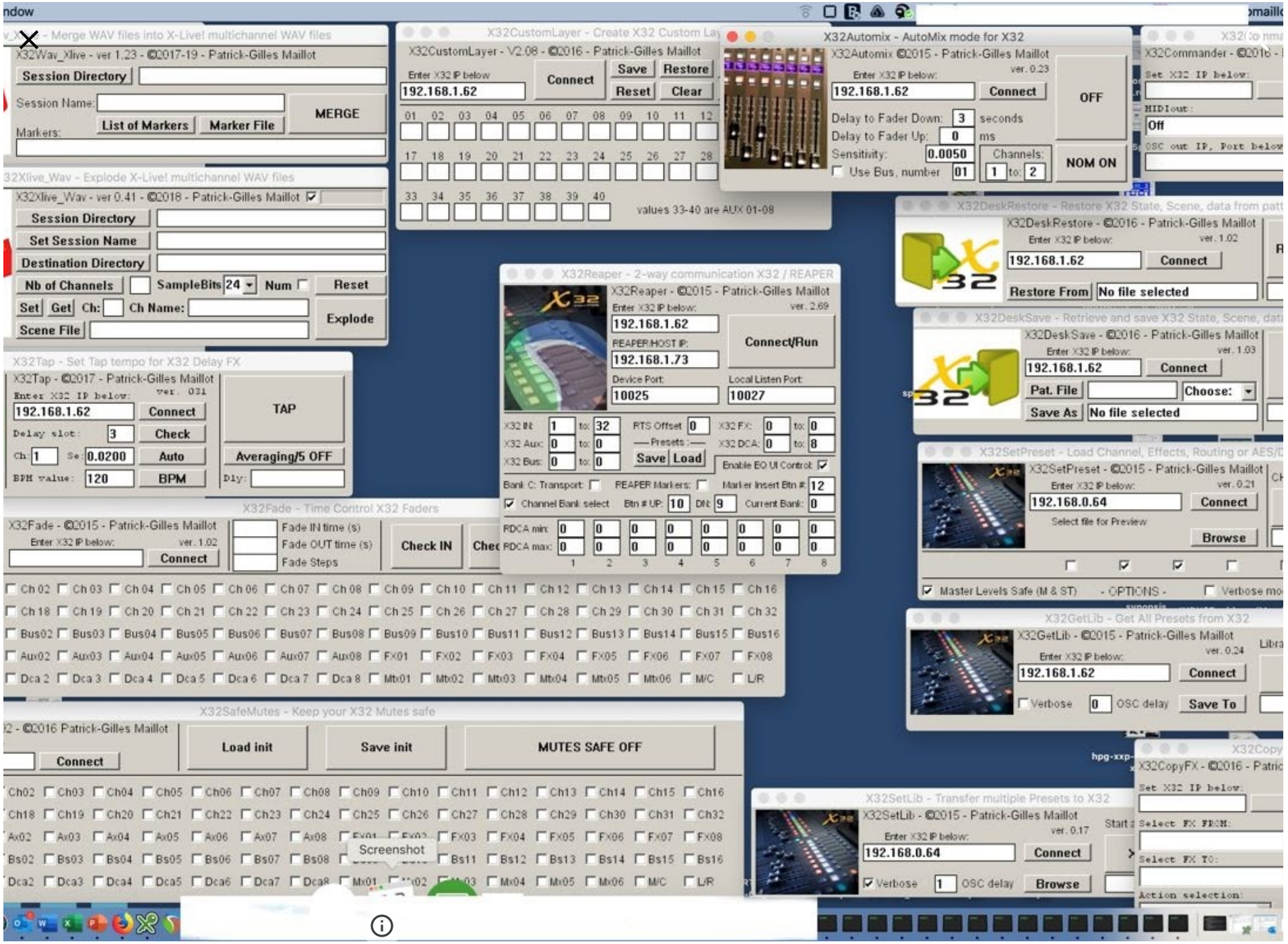
XLive, **WLive** and **MLive** related applications have their own section; use the menu on the left

Other Platforms

Apple versions of all command-line mode programs running in a terminal window, compiled on a Mac book pro OS X HighSierra: **X32mac**. [updated Dec 6, 2019].

A Mac version of the X32 Emulator, v 0.8 was compiled by a kind user and is available as a zip package.

Note that the Windows-GUI based programs run just fine under **Wine** or **CrossOver**; The easiest may be to install Wine (or its commercial equivalent, CrossOver) and run the Windows utilities you can download to you Mac; See below:


X32Wav_Xlive - ver 1.23 - ©2017-19 - Patrick-Gilles Maillot
Session Directory
Session Name:
MERGE
Markers:
List of Markers
Marker File
X32Xlive_Wav - Explode X-Live! multichannel WAV files
X32Xlive_Wav - ver 0.41 - ©2018 - Patrick-Gilles Maillot
Session Directory
Set Session Name
Destination Directory
Nb of Channels
SampleBits 24
Num
Reset
Set
Get
Ch:
Ch Name:
Scene File
Explode
X32Tap - Set Tap tempo for X32 Delay FX
X32Tap - ©2017 - Patrick-Gilles Maillot
Enter X32 IP below:
ver. 031
192.168.1.62
Connect
Delay slot:
3
Check
Ch:
1
Se:
0.0200
Auto
BPM value:
120
BPM
TAP
Averaging/5 OFF
Dly:
X32Fade - Time Control X32 Faders
X32Fade - ©2015 - Patrick-Gilles Maillot
Enter X32 IP below:
ver. 1.02
Connect
Fade IN time (s)
Fade OUT time (s)
Fade Steps
Check IN
Ch 02 Ch 03 Ch 04 Ch 05 Ch 06 Ch 07 Ch 08 Ch 09 Ch 10 Ch 11 Ch 12 Ch 13 Ch 14 Ch 15 Ch 16
Ch 18 Ch 19 Ch 20 Ch 21 Ch 22 Ch 23 Ch 24 Ch 25 Ch 26 Ch 27 Ch 28 Ch 29 Ch 30 Ch 31 Ch 32
Bus02 Bus03 Bus04 Bus05 Bus06 Bus07 Bus08 Bus09 Bus10 Bus11 Bus12 Bus13 Bus14 Bus15 Bus16
Aux02 Aux03 Aux04 Aux05 Aux06 Aux07 Aux08 FX01 FX02 FX03 FX04 FX05 FX06 FX07 FX08
Dca 2 Dca 3 Dca 4 Dca 5 Dca 6 Dca 7 Dca 8 Mtx01 Mtx02 Mtx03 Mtx04 Mtx05 Mtx06 M/C L/R
X32SafeMutes - Keep your X32 Mutes safe
Connect
Load init
Save init
MUTES SAFE OFF
Ch02 Ch03 Ch04 Ch05 Ch06 Ch07 Ch08 Ch09 Ch10 Ch11 Ch12 Ch13 Ch14 Ch15 Ch16
Ch18 Ch19 Ch20 Ch21 Ch22 Ch23 Ch24 Ch25 Ch26 Ch27 Ch28 Ch29 Ch30 Ch31 Ch32
Ax02 Ax03 Ax04 Ax05 Ax06 Ax07 Ax08
Bs02 Bs03 Bs04 Bs05 Bs06 Bs07 Bs08 Bs11 Bs12 Bs13 Bs14 Bs15 Bs16
Dca2 Dca3 Dca4 Dca5 Dca6 Dca7 Mx01 Mx04 Mx05 Mx06 M/C L/R
Screenshot
X32CustomLayer - Create X32 Custom Lay
X32CustomLayer - V2.08 - ©2016 - Patrick-Gilles Maillot
Enter X32 IP below
192.168.1.62
Connect
Save
Restore
Reset
Clear
01 02 03 04 05 06 07 08 09 10 11 12
17 18 19 20 21 22 23 24 25 26 27 28
33 34 35 36 37 38 39 40
values 33-40 are AUX 01-08
X32Reaper - 2-way communication X32 / REAPER
X32Reaper - ©2015 - Patrick-Gilles Maillot
Enter X32 IP below:
ver. 2.69
192.168.1.62
REAPER/HOST IP:
192.168.1.73
Device Port:
10025
Local Listen Port:
10027
Connect/Run
X32 In:
X32 Aux:
X32 Bus:
RTS Offset
X32 FX:
X32 DCA:
Presets :
Save
Load
Enable EQ UI Control:
Bank C: Transport:
REAPER Markers:
Marker Insert Btn #:
Channel Bank select
Btn # UP:
Dn:
Current Bank:
RDCA min:
RDCA max:
X32Automix - AutoMix mode for X32
X32Automix ©2015 - Patrick-Gilles Maillot
ver. 0.23
Enter X32 IP below:
192.168.1.62
Connect
OFF
Delay to Fader Down:
3
seconds
Delay to Fader Up:
0
ms
Sensitivity:
0.0050
Use Bus, number
01
Channels:
1
to:
2
NOM ON
X32DeskRestore - Restore X32 State, Scene, data from patt
X32DeskRestore - ©2016 - Patrick-Gilles Maillot
Enter X32 IP below:
ver. 1.02
192.168.1.62
Connect
Restore From
No file selected
X32DeskSave - Retrieve and save X32 State, Scene, dat
X32DeskSave - ©2016 - Patrick-Gilles Maillot
Enter X32 IP below:
ver. 1.03
192.168.1.62
Connect
Pat. File
Choose:
Save As
No file selected
X32SetPreset - Load Channel, Effects, Routing or AES/
X32SetPreset - ©2015 - Patrick-Gilles Maillot
Enter X32 IP below:
ver. 0.21
192.168.0.64
Connect
Select file for Preview
Browse
Master Levels Safe (M & ST)
- OPTIONS -
X32GetLib - Get All Presets from X32
X32GetLib - ©2015 - Patrick-Gilles Maillot
Enter X32 IP below:
ver. 0.24
192.168.1.62
Connect
Verbose
OSC delay
Save To
X32SetLib - Transfer multiple Presets to X32
X32SetLib - ©2015 - Patrick-Gilles Maillot
Enter X32 IP below:
ver. 0.17
192.168.0.64
Connect
Verbose
OSC delay
Browse
X32Commander - ©2016
Set X32 IP below:
MIDIout:
Off
OSC out IP, Port below
X32CopyFX - ©2016 - Patric
Set X32 IP below:
Select FX FROM:
Select FX TO:
Action selection:

Github repo

Source code of X32 utilities (C code) are available on git repositories at https://github.com/pmaillot/X32-Behringer. Feel free to clone and contribute (ideas, bug fixes / improvements).

XAir Devices?

A lot of the programs published above can be adapted to XAir devices. There are differences between X32 and XAir OSC and the XAir data is available as a Cheat-sheet here. Remember the OSC communication port for XAir series is 10024, not 10023.

X32 emulator

(Mac version)

The X32 you can carry in your pocket for developing apps or testing.

"32 channels, 16 bus, 8 simultaneous effects, 8 Aux, 6 Matrix, L/R, Mono, 8 DCA, 50+ internal effects, multi-client support"

sounds familiar? If you don't have your X32 with you, this **X32 emulator** can be helpful. With no particular attempt at optimization, this tool parses and manages X32 OSC commands (as a real X32 would), keeps up to 4 xremote clients updated - Of course no sound, and even if all 32 Channels, 16 Sends, 8 FXreturns, 8 Aux, 6 Matrix, 8 DCA, Main and all FX parameters are fully implemented along with multi-client xremote update and many more, not all X32 commands are supported (and that's not the goal), but the emulator is handy for developing X32 applications.

Watch X32 Emulator in action with this video showing X32 Emulator (upper right corner),and three client applications: X32_Command (upper left corner), X32 Edit, and Mixing Station (Android phone) connected to X32 Emulator. Everyone is happily updating :)

Note [been asked this several times]: X32 emulator only manages OSC commands, no MIDI, no Audio, no USB.

X32PlayScript

A script file for Theater play-scripts

A Small utility for venues, theaters, musicals where a show is rehearsed before being played during one or more seasons with pretty much the same settings, changing during the show.

This is a bash script, optimally running X32SetScene from one show act to the next, following play-scripts settings.

During rehearsals:

The FOH engineer sets all X32 scene parameters fit for a play act/scene, and saves the X32 scene in such a way the act and scene are listed along with a common name part, possibly followed by a meaningful text, for example in the case of the play "ALEX" ©2014 by J. Johnson:
`A1S1_ALEX_The_Fond_Household, ..., A1S8_ALEX_The_Ink_Tank, ..., A2S1_ALEX_DAKs_Pad, ...`

Saved scenes can be exported to USB then copied to a PC or directly exported to a PC using the X32GetScene utility proposed here.

During plays:

The FOH Engineer groups all scene files in a single directory.

```
MyMac:misc pm$ ls -l
-rwxr--r--  1 pm  staff  63740 Apr 24 10:55 A01S00_ALEX_init state.scn
-rwxr--r--  1 pm  staff  63740 May  4 08:26 A01S01_ALEX_The_Fond_Household.scn
        [...]
-rwxr--r--  1 pm  staff  63740 May  4 08:27 A01S08_ALEX_DAKs_Pad.scn
        [...]
-rwxr-xr-x  1 pm  staff   4071 May  5 09:21 X32PlayScript
MyMac:misc pm$
```

The FOH Engineer can the run the batch file below, placed in the same directory:
`MyMac:misc pm$ ./X32PlayScript ALEX`

The utility (a shell script) will ask for the X32 IP address, list all the files in the directory and proposes to go through each of them, waiting for the FOH engineer input to set scene details by executing data contained in the scene file corresponding to the current play act and scene in an optimal way. As the script runs, you can enter the following (lowercase/uppercase):

```
y -> accept proposed scene updates to current state
k -> same as above but will send full scene
f -> skip proposed scene
```

```
b -> go back one scene
x -> early stop (exits the program)
```

any other character: do nothing

X32GetScene / X32SetScene

X32GetScene Gets a Scene/Snippet file directly out of your X32

The utility connects to the X32 (default IP is 192.168.0.64 and can be changed with option -i) reads <*stdin*> to read the elements you want to get from the X32, and saves the result to <*stdout*>.

The elements to get from the X32 are described/coded as they are in a typical .scn file. As a facility, you can provide an existing (partial or complete) scene file to get the elements from your X32. The values from the input scene file will be ignored.

Example:

```
X32GetScene -i 192.168.1.32 -s name1 -n note1 <Default.scn >myscene.scn
```

Will take all lines from Default.scn as requests to the X32, and generate a scene file with the current X32 values respective of the requests into a file called myscene.scn, with name and note values name1 and note1 respectively.

When not providing a file as input, one has to type in requests one line at a time. Typing "*exit*" will terminate the program. For example (without -s or -n, the utility will ask for a name an notes):

```
X32GetScene -i 192.168.1.32 -s name1 -n note1 >myscene.scn
```

Typing in:

```
/ch/01/config
/ch/01/delay
exit
```

creates a file myscene.scn contaning (actual values will depend on the state of your X32):

```
#2.1# "name1" "note1" %000000000 1 X32GetScene V1.3 ©2014 Patrick-Gilles Maillot
/ch/01/config "" 1 YE 1
/ch/01/delay OFF   0.3
```

X32SetScene

X32SetScene Interprets a Scene/Snippet file and set your X32 accordingly

The utility connects to the X32 (default IP is 192.168.0.64 and can be changed with option -i) reads <stdin> to read the scene file elements and transform them into OSC commands to the X32.

The elements to send to the X32 are described/coded as in typical .scn files.

Example:

```
X32SetScene -i 192.168.1.32 < myscene.scn
```

Will take all lines from myscene.scn, interpret them and set the X32 state accordingly. When not providing a file as input, one has to type in requests one line at a time. Typing "*exit*" will terminate the program. For example:

```
X32SetScene -i 192.168.1.32
```

With typing in:

```
/ch/01/mix OFF   0 ON +0 OFF -oo
/ch/01/config "MyVox" 1 GN 1
exit
```

will set Channel 1 to: muted, fader to position 0db, pan to center, mono OFF and mono level to -infinity, channel 1 scribble screen will light in green, display "MyVox", with icon 1 selected (i.e. blank), and channel 1 will have "IN 1" as source.

Watch a video of X32SetScene in action.

X32Replay

Record and play back a show

This is work in progress, but you may find it quite useful as it already is.

Start the utility... If **record** is on, all commands and modifications made on the X32 are time stamped (with a precision close to 10µs) and recorded to a file. if needed, recording can be paused, and started again.

When recording ends, the file is closed.

The file can then be **played back**, the time stamps associated with each recorded command ensure the commands will be played back at correct times, relative to the start of the play command.

The utility runs in a terminal window (Windows cmd); Example, assuming your X32 is at IP 192.168.1.32:

```
X32Replay -i 192.168.1.32
```

Additional option is-v 0/1; Attention, this option will display a lot of additional data and this can impact the responsiveness of the tool.

```
Usage: X32Replay [-i X32 console ipv4 address] -default: 192.168.0.64
                 [-v 0/1, verbose option] -default: 0
                 [-f file name] -default: X32ReplayFile.txt
  known commands:
    stop:         stops recording or playing and closes file
    record:       reports recording state
    record off:   stops recording
    record on:    starts or resumes recording
    record pause: pauses recording and keeps file opened
    pause:        pauses recording and keeps file opened
    play:         reports playing state
    play off:     stops playing
    play on:      starts playing
    # typed line: during recording, records the typed line as a user tag
```

X32_Command

Google Drive can create issues in the download, If this happens, follow this link. File guaranteed virus free.

Sends OSC commands to X32, allows listening to X32 too...

This command line Windows tool enables sending and receiving OSC data in many ways.

```
usage: X32_command [-i X32 console ipv4 address]
                   [-d 0/1, [0], debug option]
                   [-v 0/1  [1], verbose option]
                   [-k 0/1  [1], keyboard mode on]
                   [-t int  [10], delay between batch commands in ms]
                   [-s file, reads X32node (scene, snippet, presets) formatted data lines from 'file']
                   [-f file, sets batch mode on, getting input data from 'file']
                     default IP is 192.168.1.62
 If option -s file is used, the program reads data from the provided file
 until EOF has been reached, and exits after that.
 If option -f file is used, the program runs in batch mode, taking data from
 the provided file until EOF has been reached, or 'exit' or 'kill' entered.
```

Note: make sure the batch file is respecting unix standards (i.e. use notepad++ to create the file under Windows so EOL are made of only "\n" and not "\r\n").

While executing, the following commands can be used (without the quotes):

```
   '#line of text.....': will print out the input line as a comment line
   'exit' | 'quit': will quit the current mode
   'kill': [batch mode only] will exit the program
   'time <value>': [batch mode only] will change the delay between batch commands to <value>
   'verbose <on|off>': will change the verbose mode
   'verbose': will return the current verbose mode
   'xremote <on|off>': will change the xremote mode
   'xremote': will return the current xremote mode
   '' (empty line) [standard mode only]: will repeat the last entered command
```

All other commands are parsed and sent to X32.

Typical X32 OSC command structure:

```
   <command> <format> [<data> [<data> [...]]], where for example:
      command: /info, /status, /ch/01/mix/fader, ...
      format: ',' ',i' ',f' ',s' or a combination: ',siss' ',ffiss' ...
      data: a list of int, float or string types separated by a space char...
```

```
      data: a list of int, float or string types separated by a space char!!!
```

Note: Per OSC spec and in the format above *OSC Type Tag String* is "mandatory", therefore one should always provide at least a ',' to complete the *OSC Address Pattern*. This is nevertheless not necessary for X32 as the system accepts "older" notations where empty OSC Type Tag Strings are not present (see examples below)

Examples:

```
/info                              - "old" notation sending an info request to X32
/info ,                            - "correct" (i.e. OSC compliant) notation sending an info request to X32
/ch/01/config/name ,s "My Guitar" - set name of channel 01 to My Guitar (note: space in the name)
/ch/01/config/name ,s MyGuitar     - set name of channel 01 to MyGuitar (note: no space)
/ ,s 'ch/01/config "" 1 YE 1'      - erase channel 01 name, set icon to index 1,
                                     color to Yellow and source to IN01
/ch/01/mix/fader ,f 0.5            - set mixing fader of channel 01 to mid-position
/node ,s fx/01/par                 - retrieve the 64 parameters of effect at FX slot 1
```

Please refer to the unofficial X32 OSC protocol document for a list of commands.

X32Reaper

X32 <-> REAPER controls.

The main idea is to use the X32 as a playback unit and a control surface to REAPER. Simultaneously. Digital audio is stored and managed on a computer via REAPER. REAPER controls translate to the X32 and vice-verse, changes made at the X32 deck are reflected in REAPER.

An OSC config file and a daemon program enable Reaper actions to be reflected on the X32. Simultaneously, the daemon program listens to the X32 and translates X32 actions and changes to REAPER changes on the computer. The videos below are showing different situations

- Reaper controlling X32: http://youtu.be/1oKBHDrCgKg
- X32 controlling Reaper: http://youtu.be/uOwawSH0Nlw
- 2-Way communication: http://youtu.be/5e148oruC70
- Watch it in action: http://youtu.be/3l7TfPX140Y
- X32Reaper v1.3 on a Raspberry Pi while Reaper is running on another system, 2-way comm: http://youtu.be/WBhFSEG8DRQ
- X32Reaper v1.4, with lots of channels and nearly all X32 Faders used: http://youtu.be/9L1ap2nW-Ts
- X32Reaper v2.0, with all X32 Faders used and a lot more REAPER track being handled with DCAs. https://www.youtube.com/watch?v=QCGXndXEqo8
- X32Reaper v2.20, shows scribble icons and color capability. https://youtu.be/slFBPb13zMM
- X32Reaper v2.21, shows REAPER virtual banks management from X32: https://youtu.be/U-brAQ-BmGk
- X32Reaper v2.62...v2.65, showing bank size 8: https://youtu.be/eco3AnOgkJw
- X32Reaper v2.66, showing REAPER controlling X32 EQ both ways control: https://youtu.be/1Er6UXVcqUs

X32Reaper - 2-way communication X32 ...

X32Reaper - ©2015 - Patrick-Gilles Maillot

ver. 2.71

Enter X32 IP below: 192.168.1.40

REAPER/HOST IP: 192.168.1.40

Connect/Run

Device Port: 10025

Local Listen Port: 10027

X32 IN: 1 to: 32

X32 Aux: 33 to: 40

X32 Bus: 49 to: 64

RTS Offset 0

—— Presets :——

Save Load

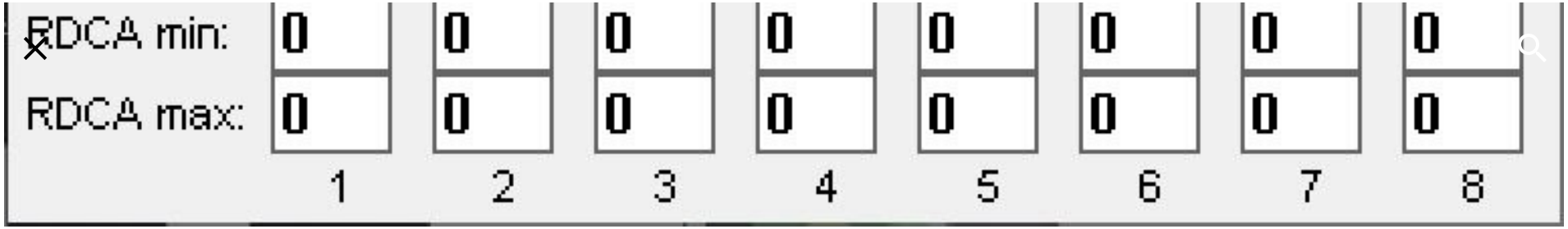
X32 FX: 41 to: 48

X32 DCA: 0 to: 0

Enable FX UI Control: ☑

Bank C: Transport: ☐ REAPER Markers: ☐ Marker Insert Btn #: 12

☑ Channel Bank select Btn # UP: 10 DN: 9 Current Bank: 0



Windows Graphical User Interface

I tested this on all 32 channels and no problem!, just be careful in selecting the X32 inputs from the card, and setting up the REAPER routing matrix correctly. It's simple but a little tedious at first: in 1->track 1->out 1, etc... until in 32 -> track 32 -> out 32. Nice diagonals.

Do read the documentation that comes with program, read REAPER documentation on setting up a control surface, this can / will be useful. Cut&paste the OSC config file provided in the X32Reaper documentation to install your config file. Be careful in selecting and matching ports and IP addresses, and you'll be just fine.

Versions **2.80** and beyond accept 1-key keyboard shortcuts for all GUI buttons and check-marks actions.

Notes:
If you are a user of a previous version, you most likely need to update the REAPER OSC configuration and .X32Reaper.ini resource files to benefit from the new features and avoid errors. Please follow instructions in the documentation for managing this.
For the Windows-GUI version, two additional files need to be with the utility. There's a graphics .bmp file of 115x140 pixels you can replace with your preferred design, and a resource file to keep and reuse the panel values from the last run.
Starting with version 1.2.1 X32Reaper will automatically select user bank C when the Connect/Run button is clicked.
Starting with version 1.2.2 X32Reaper enables a jog-style navigation (forward or backwards by beat, measure, marker or item) from encoders when in "play" mode.
Starting with version 1.4, X32Reaper offers a lot more features in its Windows GUI version. Please favor this over older versions.
Version 1.5 corrects a mis-ordered set of DCA min and max vs. Bus min and max in the config file.
Version 1.8 adds a setting for managing REAPER Track Send Offset; indeed REAPER Track sends are logical values; Physical bus send tracks numbers are offset by the number of HW outputs the track is assigned to.
Version 1.9 fixes a weird behavior when selecting several REAPER tracks at once and moving one of the faders (REAPER or X32).
Version 2.0 introduces the notion of DCA for REAPER tracks. Of course this is already possible within REAPER, but the idea is to easily control multiple REAPER tracks from an X32 DCA (which can still control its REAPER associated track and X32 internal channels. Setting all associated REAPER tracks volumes to the value set by the respective X32 DCA. Also applies to tracks ON/OFF. REAPER track numbers are set in groups of min-max for each DCA. This is typically useful when controlling REAPER tracks beyond X32 capabilities, routed (withing REAPER) to stereo buses and having the bus pairs controlled by a single X32 DCA fader controlling the overall mix of all associated REAPER tracks, without cannibalizing X32 channels.
Version 2.10 provides (virtual) bank selection for REAPER tracks so they can be controlled from your X32, acting as a pure control surface. This is exclusive from the Transport Loop control as two buttons are reserved for bank up/down management. It also covers X32 → REAPER direction, for REAPER tracks, banks are in 32 tracks blocks. Useful for managing large numbers of REAPER tracks using X32 as a surface control, the actual mix being performed by REAPER.
Version 2.22 (2.20 & 2.21 had a small bugs) enables switching back and forth between virtual banks, keeping the settings that have been set with the X32 or with REAPER. Also, X32 scribbles icons and colors can be set from REAPER using a specific syntax for REAPER tracks.
Version 2.4 enables bank selection capability without having to set transport. When only bank selection is on, the user can choose the two buttons assigned to the bank up and down actions (still user set C only, buttons 5 to 12). When used in conjunctions with transport, the two buttons are 9 and 10 and cannot be changed. No change to the user interface, although I may enable changing buttons assignments from the tool GUI in a near future.
Version 2.41 corrects small things in the way channel banks up and down buttons are using internal features.
Versions 2.5 & 2.51 enable the user to set the bank up/down buttons from the Windows graphical user interface.
Version 2.52 fixes a small rounding function and enables filtering what is actually sent to X32 from REAPER.
Version 2.61 offers the possibility to set REAPER markers on the fly directly from the X32 during recording or playback; Bank C color can also be chosen in the resource file.
Version 2.63 offers the possibility to set bank size to a value less than 32, and therefore give more flexibility in what part of the X32 is used as a surface control for REAPER while the rest of the desk can manage actual audio.
Version 2.66 enables controlling EQ from X32 and from REAPER for Input, Auxin and FxRtn channels or their corresponding REAPER tracks.
Version 2.67 introduces a bit mask to enable preventing some X32 data changes to affect REAPER.
Version 2.68 fixes an issue where setting a REAPER track name,color, or icon when it belongs to a channel bank other than the lowest one are not immediately reflected onto the X32.
Version 2.69: REAPER EQ FX get modified and modifiable only if the "EQ UI Control" flag is set; Requires X32Reaper restart. Also correctly manages the internal bank size default value when UI "Channel Bank select" flag is set to OFF.
Version 2.71: Manages REAPER FX EQ and COMP. Effects can also be 'placed' in the effect chain list.
Version 2.72: Added track record arm toggle functionality (action 40294) with combination of keys: Beg Loop/Repeat (i.e. Beg Loop acts as a shift key - works too if bank is on, then it's UP that acts as a shift key)
Version 2.73: Xdelayb and Xdelayg were mixed up; plus removed some of the Xdelayb delays to speedup bank changes, otherwise too slow, even with a 1ms delay.
Version 2.80: Added keyboard shortcuts for all button-press GUI functions: connect, save, load, transport, bank, marker, EQ/FX
Window must have keyboard focus; shortcuts are C, S, L, T, B, M, F, respectively
Version 2.81: Display connect, save, load, transport, bank, marker, EQ states in the title bar to provide NVDA support
Version 2.82: Fixed crash when ch bank select unselected and user sets EQ on (was the same for other effects such as dyn)

Ready?

... download **X32Reaper**, unzip, and give it a run!

You can also watch Michael's video, setting up his X32 with X32Reaper; Another video from Michael shows/explains using transport section.

REAPER Template

You'll find here a template I often use to start a project using my X32 and REAPER. This results in the project screen shown here and the routing shown here. It includes all currently available controls, and maps to the default X32Reaper resource file provided with the program. All 16 bus mixes are setup to enable a classical routing of input channels, mapped to the X32 capabilities. Setting up your X-USB card to 32in/32out will provide you with the capability to control REAPER Tracks and use the X32 capabilities to mix your work, listening to the result directly with the X32. A 32 channels sound "card". Make sure you toggle your X32 channel input routing between In and Out in order to mix or record with REAPER, respectively.

I have set all Channel, Aux, Bus[1-12], and DCA to their minimum as a precaution; all SENDs are also set to -oo. Better keep your audio levels safe when you start a new project. REAPER master track is not used; mixing/mastering is done only on the X32 as this is what I use for mixing audio.

The latest X32Reaper for Mac is available as a command line (terminal) application with the X32mac package mentioned at the top of this page, but you can also use Wine. It includes all controls as mentioned in X32Reaper for Windows, reading all needed parameters from a configuration file (.X32Reaper.ini); below is an example of config file. The two first integers are ignored.

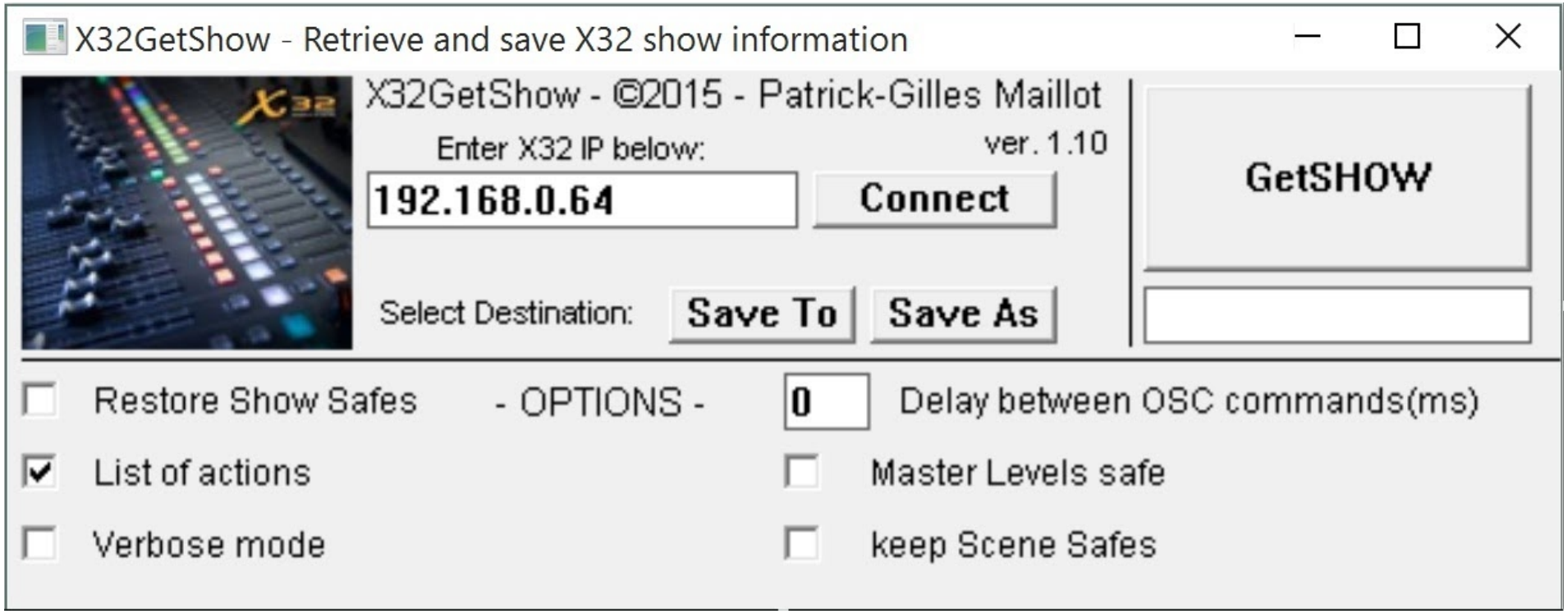
A typical resource file for the program (should be placed in the directory where the program is)

```
400 260 0 0 1 1
1 32 33 40 41 48 49 64 65 72 1
192.168.1.62
192.168.1.65
10025
10027
```

Use ^C to kill the program.

X32GetShow / X32SetShow

Get/Save on your PC the X32 current show, cues, and all associated scenes and snippets



X32GetShow connects to the X32 (using the IP provided in the dedicated label). Once connected to X32, it reads the current **show** from X32, and saves the show as a *.shw file on your PC (in a directory you select using the "save To" button); all cues, scenes, and snippets are listed and saved with the show.

There are two options for saving the show: "Save To" and "Save As". Both are quite similar and will use the directory you point at in the opened dialog as destination directory for the saved data (show files). In the first option (*Save To*), the X32 show and saved files are keeping the name as set in the X32. The second option (*Save As*) enables renaming the X32 show files to a new show name as data is saved.

For all **scenes** and **snippets** listed in the show file, the utility creates the respective files (*.scn or *.snp) and saves all data associated with each scene or snippet. The saved files can then be kept for later or offline use, manual edit, or reload in a X32 from a USB drive.

The different scenes and snippets names will appear under the "**GetSHOW**" button as the program progresses in saving files. When finished, "Complete" will be displayed. In the event something does not complete as expected; "Error" will be displayed and the log file will have more information.

Example: on a X32 system with show name "show1", containing 2 scenes "scene1" & "scene2" at positions 00 and 01 respectively, a snippet "snippet1" with Parameter Filter "Preamp (HA)", channel "Ch 1", return "Aux 1" and main/matrix/group "Matrix 1" selected,

X32GetShow will connect to the X32 and read data from it. The following files (actual data values may vary) will be created in the specified directory:

c:/test/show1.shw:

```
#2.1#
show "show1" 0 0 0 0 0 0 0 0 0 0 "X32-Edit 2.30"
cue/000 100 "cue1" 0 0 0 0 1 0 0
cue/001 101 "cue101" 0 1 0 0 1 0 0
scene/000 "scene1" "" %000000000 1
scene/001 "scene2" "" %000000000 1
snippet/000 "snippet1" 1 1 1 1 1
```

c:/test/show1.000.scn

```
... a 2000+ lines file
```

c:/test/show1.001.scn

```
... a 2000+ lines file
```

c:/test/show1.000.snp

```
#2.1# "snippet1" 1 1 1 1 1
/ch/01/preamp -0.5 OFF OFF 24  61
/ch/01/delay OFF   0.3
```

```
/auxin/01/preamp +0.0 OFF
/mtx/01/preamp OFF
```

A **log file** will also be created in a dedicated directory. It enables listing the different actions taken by the program (if the "***list of actions***" option is checked) and the full set of OSC commands sent and received (if the "***Verbose***" option is checked).

Additional options enable restoring or not ***Show Safes*** after running the program, Use of not the ***Master Level Safe*** X32 feature or not when saving scenes, and keeping or not ***Scene Safes*** when saving scenes.

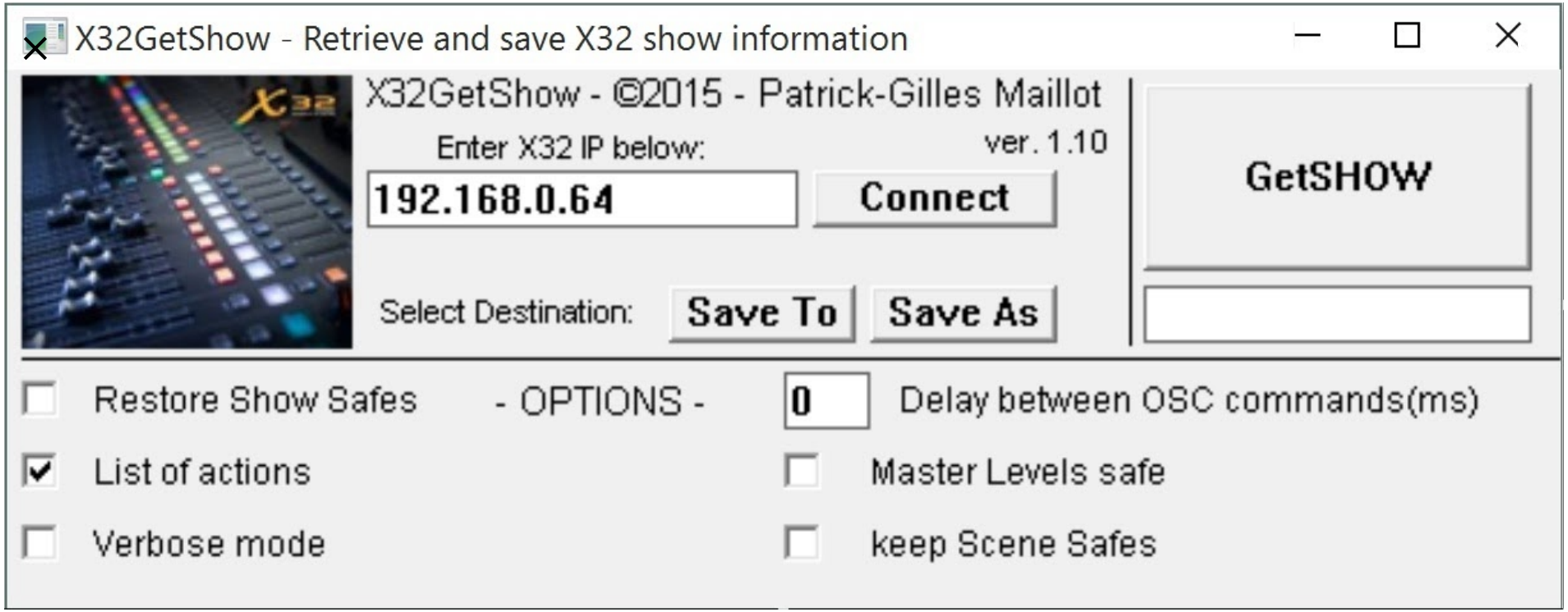
Finally, a ***delay*** value (in milliseconds) can be adjusted to allow for scenes and snippets to load safely before additional commands are sent to X32 (tested with 0ms delay with a wired 100Mbit/s Ethernet connection is OK)

All options are saved in a file: .X32GetShow.ini which also contains the *Width*, *Height* and Window loop *Refresh* factor as the first 3 parameters to be adjusted to fit your Windows settings, as shown below:

```
540 200 300 0 1 0 0 0 0     <W>  <H> <R> < options >
192.168.1.77                <    IP    >
.                           <log directory>
```

X32SetShow

X32SetShow connects to the X32 (using the IP provided in the dedicated label). Once connected to X32, enables you to browse (opens a directory browsing window) your PC files for a location where a show file and associated scenes&snippets are located. It reads the **show** file, and saves all data to your X32 as the current show; all cues, scenes, and snippets are loaded and installed as well.


X32GetShow - Retrieve and save X32 show information
X32GetShow - ©2015 - Patrick-Gilles Maillot
ver. 1.10
Enter X32 IP below:
192.168.0.64
Connect
GetSHOW
Select Destination:
Save To
Save As
Restore Show Safes
- OPTIONS -
0
Delay between OSC commands(ms)
List of actions
Master Levels safe
Verbose mode
keep Scene Safes

For all **scenes** and **snippets** associated to the show, the utility reads the respective files (*.scn or *.snp) and loads all data from each scene or snippet. The show is then fully available from your X32.

The different scenes and snippets names will appear under the "**SetSHOW**" button as the program progresses in loading data from files. When finished, "Complete" will be displayed. In the event something does not complete as expected; "Error" will be displayed and the log file will have more information. *Note: the scene and snippet files are expected to be in the same directory as the show file they report to.*

Example: on a PC with show name "show1", referencing 2 scenes "scene1" & "scene2" files, a snippet "snippet1" file,

X32SetShow will connect to the X32 and read data from it. The following files (actual data values may vary) will be accessed and their respective data loaded to X32:

c:/test/show1.shw:

```
#2.1#
show "show1" 0 0 0 0 0 0 0 0 0 0 "X32-Edit 2.30"
cue/000 100 "cue1" 0 0 0 0 1 0 0
cue/001 101 "cue101" 0 1 0 0 1 0 0
scene/000 "scene1" "" %000000000 1
scene/001 "scene2" "" %000000000 1
snippet/000 "snippet1" 1 1 1 1 1
```

c:/test/show1.000.scn

```
... a 2000+ lines file
```

c:/test/show1.001.scn

```
... a 2000+ lines file
```

c:/test/show1.000.snp

```
#2.1# "snippet1" 1 1 1 1 1
/ch/01/preamp -0.5 OFF OFF 24  61
/ch/01/delay OFF   0.3
/auxin/01/preamp +0.0 OFF
/mtx/01/preamp OFF
```

A **log file** will also be created in a dedicated directory. It enables listing the different actions taken by the program (if the "***list of actions***" option is checked) and the OSC commands sent and received (if the "***Verbose***" option is checked).

Additional options enable updating or not ***Show Safes*** from the show file data after running the program, Set of not the ***Master Level Safe*** X32 feature or not when saving scenes, and Update or not ***Scene Safes*** when loading scenes. The tool also enables a full erase of the show in the console prior to load a new one or replace some of the scenes/snippets only with the ***Reset Show Data*** flag.

Finally, a ***delay*** value (in milliseconds) can be adjusted to allow for scenes and snippets to load safely before additional commands are sent to X32 [a value between 1 (verbose on) and 2 (verbose off) a wired 100Mbit/s Ethernet connection is OK].

All options are saved in a file: .X32SetShow.ini which also contains the *Width*, *Height* as the first 2 parameters to be adjusted to fit your Windows settings, as shown below:

```
540 200 1 1 0 0 1 1 4                    <W>  <H> < 6x options ><delay>
192.168.0.64 <X32 IP address>
MyShow                                   <Name of last Show>
.                                        <Path to log file>
C:\test1 <path to last show file
```

X32AutoMix

Automatic Mixing Tool for X32 - Up to 32 Channels. **X32AutoMix** enables automatic action of level Faders based on the actual signal present at X32 Inputs.
Note: this tool/utility does not claim providing professional AutoMixing functions such as found in Dugan Automixing products.

X32AutoMix drives input levels based on the presence of a signal at a given channel. Optionally, the global mix level will also be adjusted automatically based on the number of active inputs, a feature called **NOM** [Number of Open Mics]: each time the number of active inputs doubles, the global mix drops 3dB, readjusting up 3dB when appropriate.

Global mixing typically takes place at the channel faders, the L/R bus being used (by default) as global mix. This can be inconvenient when other sources (out of the AutoMix feature) are to be used as well, without going through X32AutoMix. This version enables to use a mix bus rather than L/R. In that case you can select a bus number (01...16), and use the bus send rotary button to adjust the minimum/nominal level values. The global mix will take place on the bus you selected, using the bus fader as global mix level.

It is a good idea to unselect L/R from the auto-mixed channels when auto-mixing through a bus, Auto-mixed input signals would otherwise go through both L/R and the bus. Another (important!) benefit of using a mix bus rather than the L/R bus is to avoid unnecessary wear on channel motorized faders.

For all active channels:

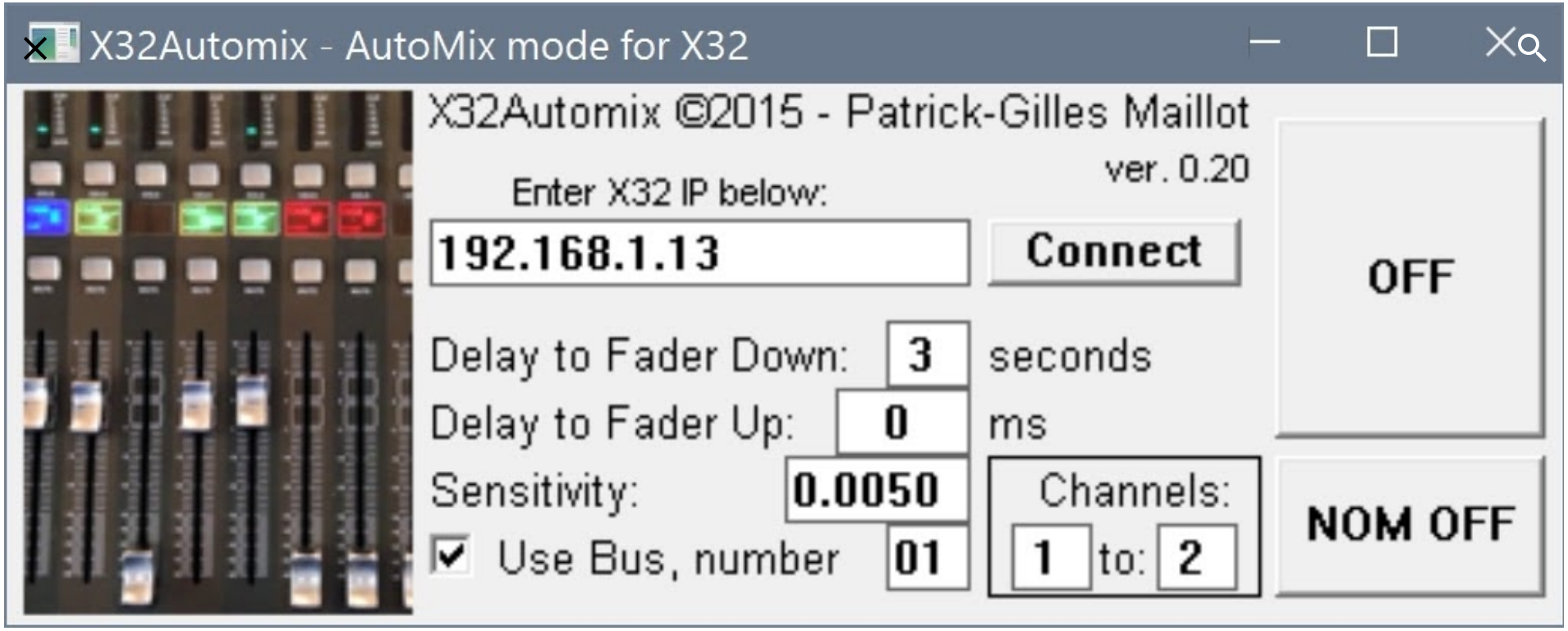
No signal:The channel level goes to its minimum position after a predefined delay (Delay to Fader Down).

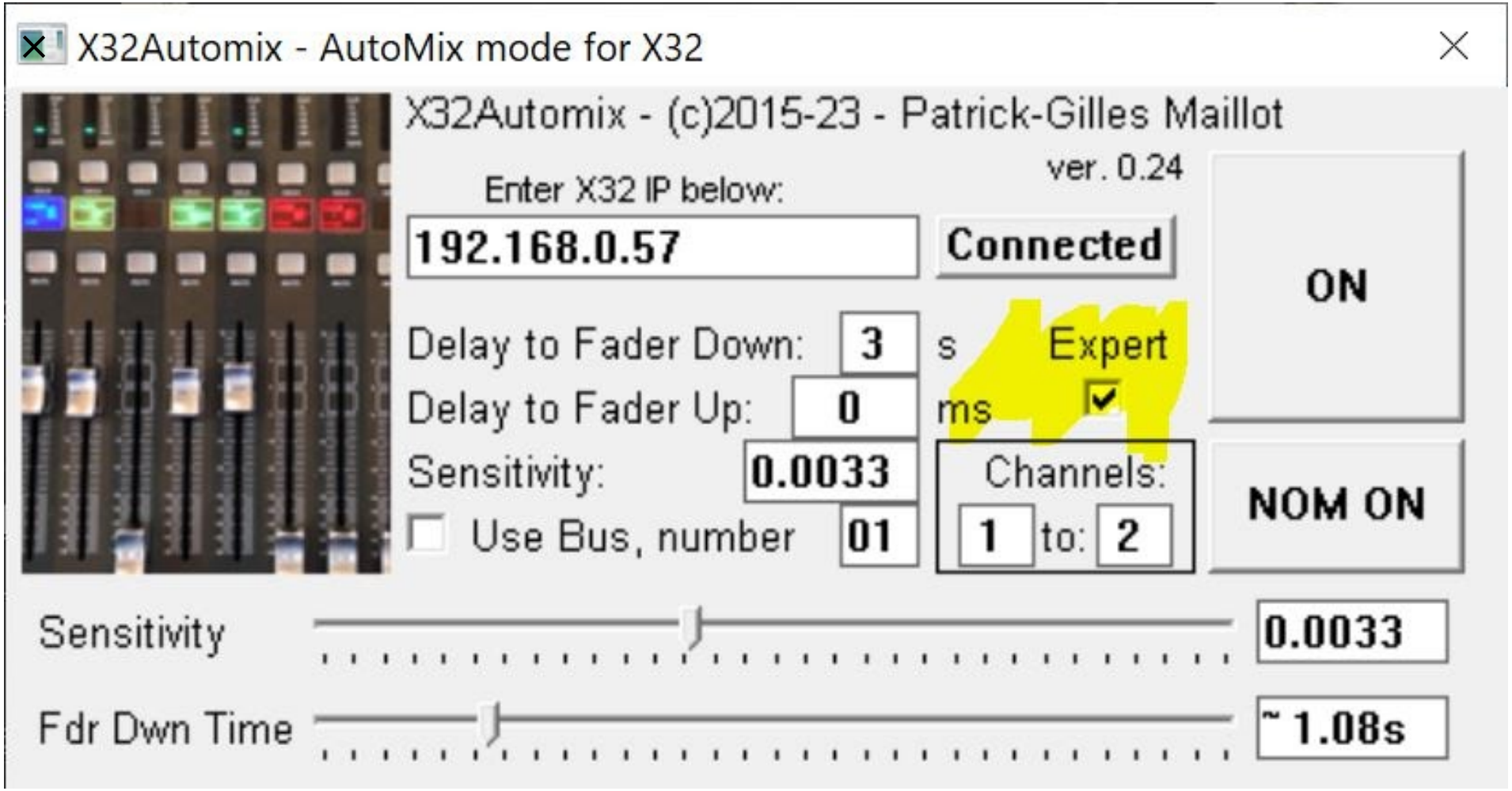
A signal above a predefined level: The channel level moves to its nominal position, after an adjustable delay (Delay to Fader Up).

For all active channels, the **minimum** (Default: -oo) and **nominal** (Default: 0db) channel level positions can be changed at anytime, and will be remembered for the current session.

The **Channels** controlled by the tools are selected within a range (Channel numbering is from 01 to 32). A single channel can be set by using the same channel number for both values on each side of the "to:" , as in the picture above.

Delay to Fader Up: is set by the frequency at which data is read from X32. Typical values are from 0ms to 1000ms. The value granularity is around 50ms. This is a global value (applies to all channels).


X32Automix - AutoMix mode for X32
X32Automix ©2015 - Patrick-Gilles Maillot
ver. 0.20
Enter X32 IP below:
192.168.1.13
Connect
OFF
Delay to Fader Down:
3
seconds
Delay to Fader Up:
0
ms
Sensitivity:
0.0050
Channels:
1
to:
2
Use Bus, number
01
NOM OFF



0.24 A newer version offers an "**Expert**" flag that will show two additional sliders

Sensitivity can be adjusted more easily and can be modified during an AutoMix session if needed

Fader Down Time can also be adjusted between 0 and approx 6 seconds. This is the time it will take for faders to travel to their low position, once the Delay To Fader Down time value has expired.

Also, pressing the unlit **Select** button of a strip when its fader is traveling down will immediately stop the travel of that fader; a quick way to cancel the 'switching off' of a microphone.

Delay to Fader Down: is controlled by a timer within the utility, and is set in seconds. A channel fader will stay Up if a signal is present, and will go Down as soon as no signal has been sensed on that channel for x seconds.

Sensitivity: is a threshold value above which a signal is considered present. It is based on the X32 preamp input. practical values are above 0.001 and 1.000 is the max value (fully saturated/clipping preamp). This is a global value (applies to all channels).

You can decide to use a **mix Bus** (check the button to that effect) and set the mix Bus number accordingly. Automixing will go through that bus and make all corrections on that bus rather than the L/R bus (the default).

The **NOM** feature is optional and can be enabled by pressing the NOM OFF/NOM ON button.

The utility panel values are reloaded from the utility each time the tool is set into a running state by pressing the OFF button (it will then display "ON")

X32AutoMix tool saves the Fader Up and Fader Down values from one session to the next, so you don't have to reset these when starting a new Automix session.

Watch a small video of the tool (then ver 0.10) in action (Sorry for the poor quality).

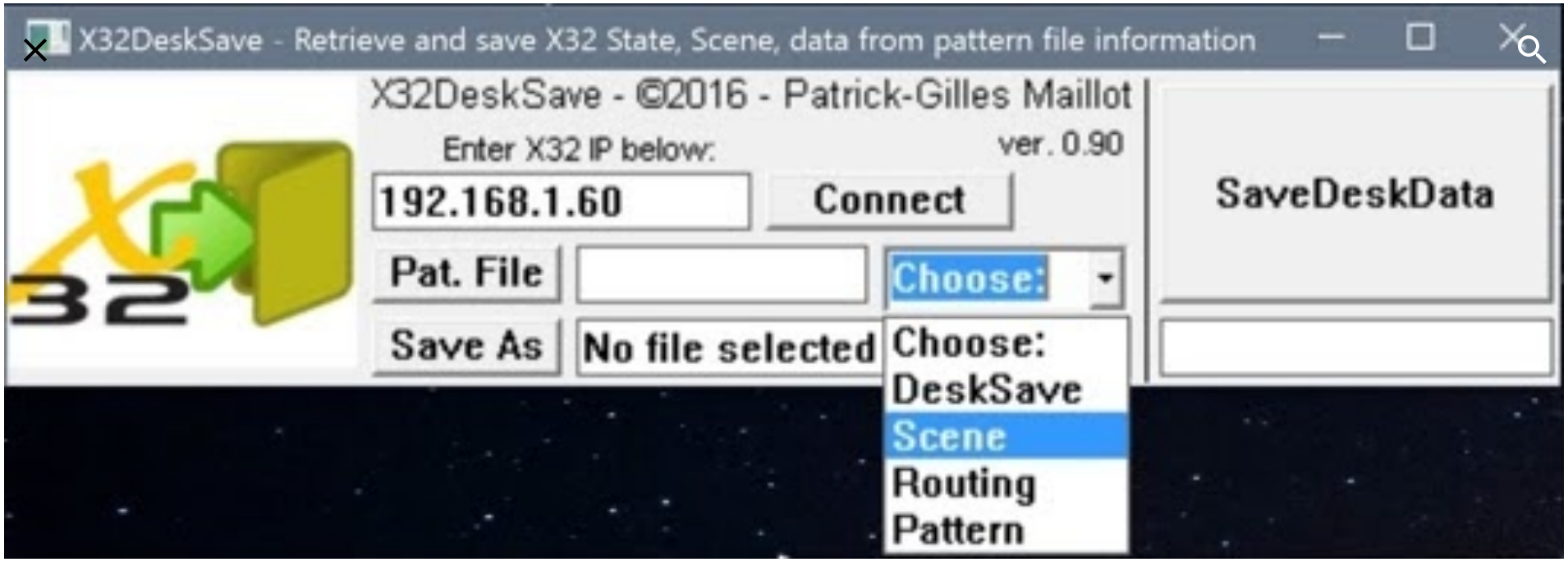
X32DeskSave / X32DeskRestore

Save and Restore your X32 Preferences and State, Scene, Routing, Commands

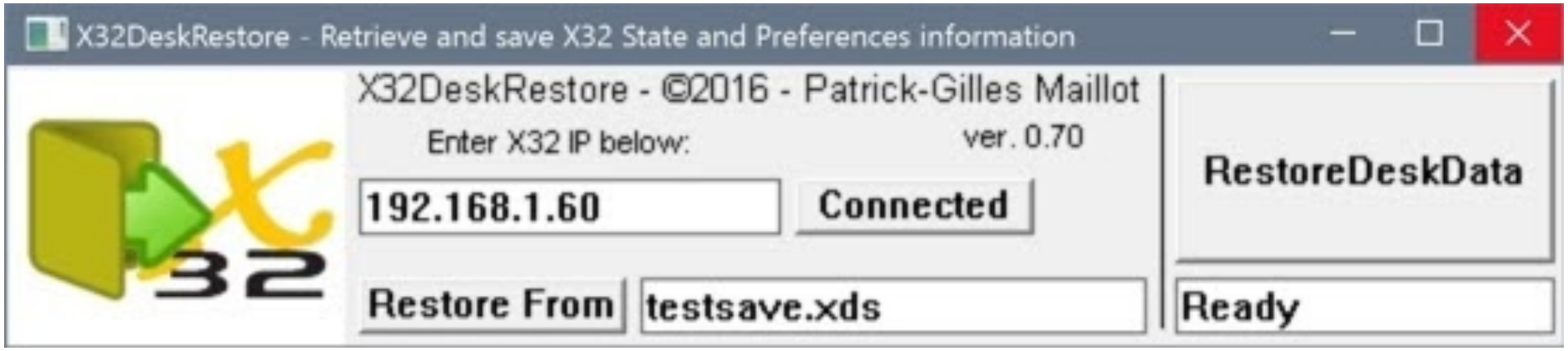
Say you want to be able to save those parameters that are not handled by shows, scenes, snippets, cues, presets, routing, effects or AES/DP48 files... So when you connect to your desk, it is exactly as you want it: screen, light brightness, view, etc.

That's what **X32DeskSave** supports... but wait! there's more: You can also save scenes, routing, AES/DP48 presets or any type of file or settings based on a pattern file (a scene, any set of commands, etc.); what the "Pattern" option does is read a file you selected by pressing on the "Pat.File" button, extract the X32 commands from this file and send the extracted commands to X32 to retrieve the values from your X32. These are saved in the file you select using the "Save As" button. Of course you don't need to select a pattern file if you choose "Scene", "Routing" or "AES/DP48" options.

Saving a full X32 Scene takes about 3 seconds (over WiFi), saving Routing or State/Preferences takes about 1.5 second.



X32DeskSave and its parameters



X32DeskRestore

X32DeskRestore looks rather similar and enables you to select a file as a source of commands and values to be restored to your X32. It can be any of the files you saved using X32DeskSave, but it can also be an actual X32 Scene or an X32 Snippet file you saved directly on your system and imported you your PC.

For both utilities, you need to connect to your X32, select the files or options you want to use. The status line under the "SaveDeskData" or "RestoreDeskData" buttons will indicate "Ready" meaning the utility has the information it needs to perform an operation. "Complete" will indicate the completion of the save or restore action. Other status lines will indicate actions you should take prior to using the utility.

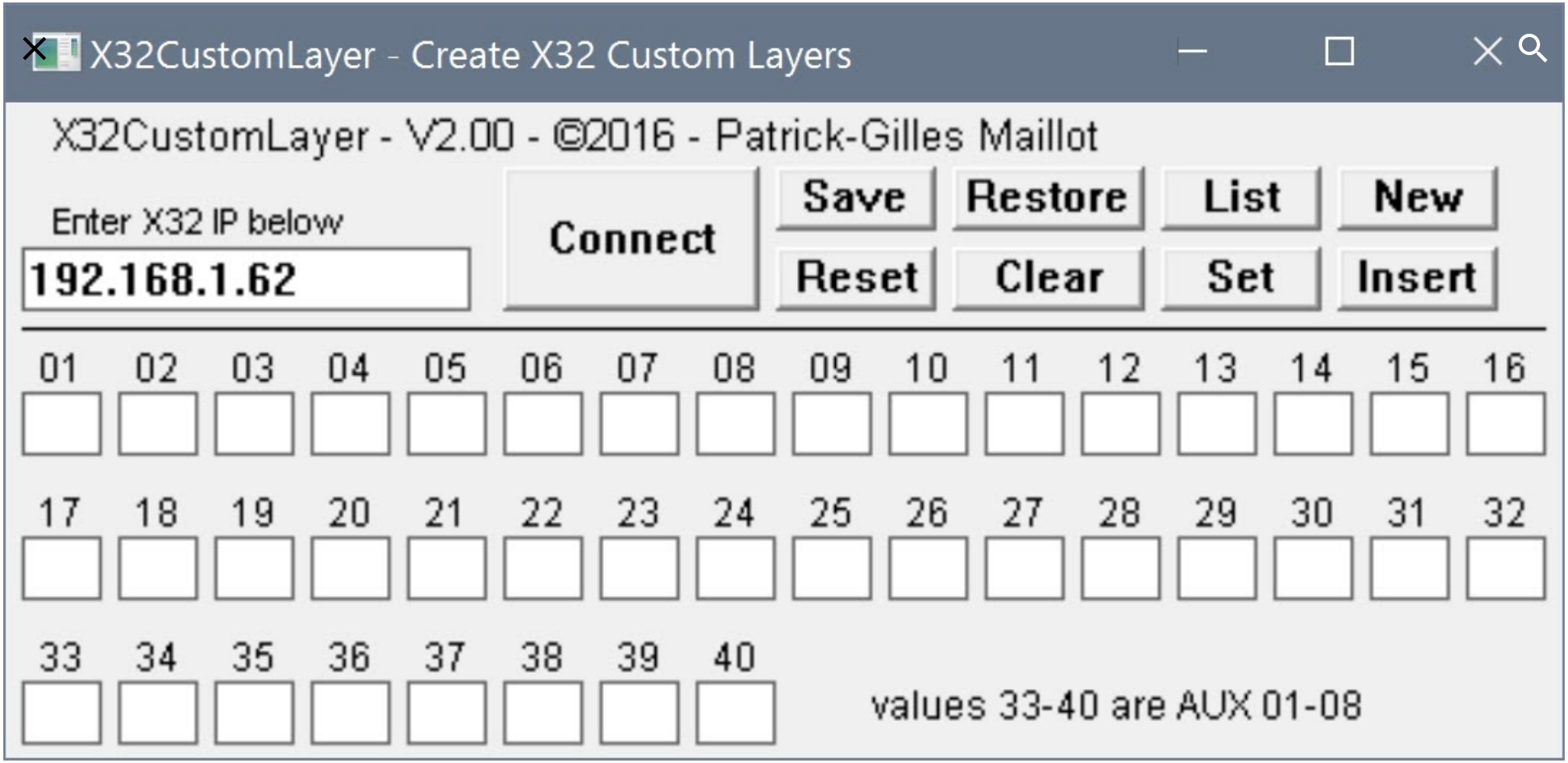
Versions 1.00 and above of X32DeskSave and X32DeskRestore include a command-line version for Mac users in the X32mac.zip file as mentioned at the top of this page.

Attention! This will not save/restore your current X32 show; for this you will need to use X32GetShow/X32SetShow. This will not save/restore all X32 current libraries types; for this you will need to use X32GetLib/X32SetLib

---

X32CustomLayer

Manage and re-organize X32 Channel strips!



X32CustomLayer is a simple utility to create and manage custom channel layers easily. Starting from a standard or current X32 setup (that is Input channels 01 to 32 and Aux inputs 01 to 08), X32CustomLayer enables you to reorganize the way “channels” are set (or layered).

Basically this enables you to virtually move a “channel” to a different position, keeping the actual input assigned to that “channel”; For example, say you have your singer microphone set at “channel” 01 and using XLR input 01. For ease of use, you need or would prefer to have this microphone handled by fader 25, and no time to unplug everything, and change all “channel” 25 settings to values that were used for “channel” 01.

You enter 01 in the box under “channel” 25, and voila! the program does the rest in copying config, eq, dyn,..., sends sections and in reassigning the input source, in a snap! X32CustomLayer will manage X32 Input channels 01 to 32, and Aux inputs 01 to 08 as numbers 33 to 40.

X32Midi2OSC

Have X32 receive OSC data from MIDI devices using standard MIDI commands.

MIDI commands are typically 3 bytes (some are 2 and SYSEX data can be a lot more) . X32 natively accepts OSC data over SYSEX messages so this is not covered here. X32Midi2OSC is focusing on 3 bytes MIDI messages, based on a translation file as shown below.

```
#
#
# Translation file for Midi2OSC
#
# ©2018 Patrick-Gilles Maillot
#
#
# Expected format uses space or tab character as separator and separates
# MIDI and respective OSC command by a '|' character:
#
#    {A0..F0} ch d1 d2 | <OSC message>
#
#   - B0 means MIDI control message, where:
#        ch is the channel number [1..16]
#        d1 is the control event number [0..127]
#        d2 is the value associated to the command [0..127]
#
# OSC messages consist of an OSC Address Pattern followed by an
# OSC Type Tag String followed by zero or more OSC Arguments.
#
# Possible OSC Type Tags are i or f
#
# OSC Arguments are listed between brakets '[]' and can be a fixed value,
# such as 0, 1 or an expressionsuch as $n to make reference to data issued
# from the incoming MIDI command, where
#        $0 represents the channel number
#        $1 represents the control event number
#        $2 represents the value data (the value 0 or 127 here will
#                be replaced during execution with the actual data
#                from the MIDI command
#
# Accepted MIDI messages:
#    Message                   Type        Data 1                 Data 2
#    Note Off                   8n        Note Number            Velocity
#    Note On                    9n        Note Number            Velocity
#    Polyphonic Aftertouch      An        Note Number            Pressure
#    Control Change             Bn        Controller Number      Data
#    Program Change             Cn        Program Number         from file
#    Channel Aftertouch         Dn        Pressure               from file
#    Pitch Wheel                En        LSB                    MSB
#    System Message             Fn        depends on message     depends on message
#
#    Key
#    n is the MIDI Channel Number (0-F)
#    LSB is the Least Significant Byte
#    MSB is the Least Significant Byte
#
#
```

```
#    The reverse polish notation calculator supports the following operators, on numbers,
#    (possibly preceded with a $ to represent a MIDI parameter), or hexadecimal data:
#    (+) (-) (*) (/)  Boolean operators (~ >> << & ^ |), modulo (%) on ints, test operator (?),
#    equal comparison (=), different comparison (!), exp (e), log_n conversion (l),
#    log_10 conversion (L), and truncate to int (i).
#
# Lines starting with a # are comment lines
#
# some examples below (uncomment if you want to test)
#B0  1 10 127 | /ch/10/mix/fader ,f [3 1 & 2 < / 3]     # testing simple operations
#B0  1 11 127 | /ch/11/mix/fader ,f [$1]                # testing simple access to parameter
#B0  1 12 127 | /ch/12/mix/fader ,f [127 $2 -]          # equivalent to inversing effect
#B0  1 13 127 | /ch/12/mix/fader ,fi [127 $2 -] [$2]    # not a valid command, but for test purpose
#90  1 56 127 | /ch/26/mix/fader ,f [$2 2 * i 60 /]     # () force an int value
#
B0 3 12 127 | /fxrtn/01/mix/on ,i [$2 0 =]              # if $2 is 0 then result is 1 (true), 0 (false) otherwise
B0 4 12 127 | /ch/01/mix/fader ,f [$2 3 ! 0.7 0.5 ?]    # if $2 equals 3 then result is 0.5, 0.7 otherwise
#
```

The program can apply operations to any of the MIDI command attributes to modulate the OSC message parameters that will be sent. To this end, an RPN calculator is used to apply operations to the incoming MIDI data before applying it to OSC message parameters. Midi thru is untouched.

The "Dbg" button enables easy debugging of MIDI commands that are received by the program, Each recognized MIDI message will be displayed in hexa form. Use only when debugging specific patterns! as this blocks the program until the displayed message is acknowledged.

The following convention sand operators apply: $0, $1, $2 represent the incoming MIDI ch, d1, and d2, respectively.

At this time, only short MIDI messages are considered. Long SYSEX messages are ignored. The list of accepted MIDI commands is listed above in the conversion file example.

The reverse polish notation calculator supports the following operators, on numbers, (possibly preceded with a $ to represent a MIDI parameter), or hexadecimal data: (+) (-) (*) (/) Boolean operators (~ >> << & ^ |), modulo (%) on ints, test operator (?), equal comparison (=), different comparison (!), exp (e), log_n conversion (l), log_10 conversion (L), and truncate to int (i).

Formulas should be saved in a file with a ".m2o" extension, as shown above.

X32Commander

Have X32 commands control other devices through MIDI or OSC.

X32Commander scans for a user selected subset of X32 Commands. If a matching command is found, the tool can fire a MIDI or OSC command to another device, optionally using parameters sent with the matching command.

When using the X32 Command parameters, a formula can be applied using a reverse polish notation calculator.

The program can be launched directly, in that case it will attempt to open ./X32Commander.txt, a file that contains the list of commands to match and replace with user selected commands. X32Commander can also be launched from a terminal window, with a -f option, enabling you to create and keep a set of files to fit different situations.

Below is an example of user command file, used at program startup used to have a second X32 mimic what the first one does; Make sure to use with version 1.10 or later.

```
# X32Commander translation file ©Patrick-Gilles Maillot
# These two first lines must be kept part of the this file
#
# Describes and lists MIDI or OSC command to send, corresponding to the X32
# OSC command the program scans for, listed below.
# A line starting with M means the expected command to send will be a MIDI command
#                      O means the expected command to send will be an OSC command
# In the command to send, a '$0' string element will be replaced by the parameter value
# of the respective OSC command 0: first parameter, 1: 2nd parameter, and so on
# '$n' parameters or their calculated data must be enclosed within '[' and ']' characters
#
#    The reverse polish notation calculator supports the following operators, on numbers,
#    (possibly preceded with a $ to represent a MIDI parameter), or hexadecimal data:
#    (+) (-) (*) (/)  Boolean operators (~ >> << & ^ |), modulo (%) on ints, test operator (?),
#    equal comparison (=), different comparison (!), exp (e), log_n conversion (l),
#    log_10 conversion (L), and truncate to int (i).
#
# Once all M (for Midi) or O (for OSC) commands are detailed, a wildcard line as below can be
# set to mention that any other OSC command should just be copied to OSC output
# The line should be (without quotes) "O   *   "
#
# comment line below if only one instance of a line can match
scan all
O   *   # Just copy everything to output --- WARNING! Do not use if OSC in and
#         out IP addresses are the same or you'll end-up with infinite loops
#
# end of file
```

Below is an example of user selection, saved in a file used at program startup:

```
# X32Commander translation file ©Patrick-Gilles Maillot
# These two first lines must be kept part of the this file
#
# Describes and lists MIDI or OSC command to send, corresponding to the X32
# OSC command the program scans for, listed below.
# A line starting with M means the expected command to send will be a MIDI command
#                      O means the expected command to send will be an OSC command
# In the command to send, a '$0' string element will be replaced by the parameter value
# of the respective OSC command 0: first parameter, 1: 2nd parameter, and so on
# '$n' parameters or their calculated data must be enclosed within '[' and ']' characters
#
#    The reverse polish notation calculator supports the following operators, on numbers,
#    (possibly preceded with a $ to represent a MIDI parameter), or hexadecimal data:
#    (+) (-) (*) (/)  Boolean operators (~ >> << & ^ |), modulo (%) on ints, test operator (?),
#    equal comparison (=), different comparison (!), exp (e), log_n conversion (l),
#    log_10 conversion (L), and truncate to int (i).
#
# Once all M (for Midi) or O (for OSC) commands are detailed, a wildcard line as below can be
# set to mention that any other OSC command should just be copied to OSC output
# The line should be (without quotes) "O   *   "
#
# comment line below if only one instance of a line can match
scan all
#
M   /-stat/selidx ,i 0        | F0 7F 00 [$0] 02 F7    # first command select channel 1
#                                                ... sends F0 7F 02 00 02 F7 for chan 1
#                                                ... sends F0 7F 02 01 02 F7 for chan 2
# Channel Faders [0., 1.]-> [0..127] etc.
O   /ch/01/mix/fader ,f 0     | /ch/05/mix/fader ,f [1 $0 -]    # for fader moves on channel 1
O   /ch/02/mix/fader ,f 0     | /ch/06/mix/fader ,f [2 $0 *]    # for fader moves on channel 2
O   /ch/03/mix/fader ,f 0     | /ch/07/mix/fader ,f [1 $0 -]    # for fader moves on channel 3
O   /ch/03/mix/fader ,f 0     | /ch/08/mix/fader ,f [2 $0 *]    # for fader moves on channel 3
M   /ch/03/mix/fader ,f 0     | F0 7F 04 [$0 127 *] 02 F7       # for fader moves on channel 3
M   /ch/04/mix/fader ,f 0     | F0 7F 04 [1 $0 - 127 *] 02 F7   # for fader moves on channel 4
#
# end of file
```

Below is an example of user selection, saved in a file used at program startup:

```
# X32Commander translation file ©Patrick-Gilles Maillot
# These two first lines must be kept part of the this file
#
# Describes and lists MIDI or OSC command to send, corresponding to the X32
# OSC command the program scans for, listed below.
# A line starting with M means the expected command to send will be a MIDI command
#                      O means the expected command to send will be an OSC command
# In the command to send, a '$0' string element will be replaced by the parameter value
# of the respective OSC command 0: first parameter, 1: 2nd parameter, and so on
# '$n' parameters or their calculated data must be enclosed within '[' and ']' characters
#
#    The reverse polish notation calculator supports the following operators, on numbers,
#    (possibly preceded with a $ to represent a MIDI parameter), or hexadecimal data:
#    (+) (-) (*) (/)  Boolean operators (~ >> << & ^ |), modulo (%) on ints, test operator (?),
#    equal comparison (=), different comparison (!), exp (e), log_n conversion (l),
#    log_10 conversion (L), and truncate to int (i).
#
# Once all M (for Midi) or O (for OSC) commands are detailed, a wildcard line as below can be
# set to mention that any other OSC command should just be copied to OSC output
# The line should be (without quotes) "O   *   "
#
# comment line below if only one instance of a line can match
scan all
#
# This example file sets ch10 as the stereo pair of ch01
# Commands may be added or removed in order to adjust or optimize what parts of the audio path
# are actually used. Note that pan functions are always set as [1 $0 -] to move in opposite direction
#
# General settings
O   /ch/01/config/color ,i 0            | /ch/10/config/color ,i [$0]
O   /ch/01/delay/on ,i 0                | /ch/10/delay/on ,i [$0]
O   /ch/01/delay/time ,f 0              | /ch/10/delay/time ,i [$0]
# Preamp section - may adjust to adapt to uneven levels
O   /ch/01/preamp/trim ,f 0             | /ch/10/preamp/trim ,f [$0]
O   /ch/01/preamp/invert ,i 0           | /ch/10/preamp/invert ,i [$0]
O   /ch/01/preamp/hpon ,i 0             | /ch/10/preamp/hpon ,i [$0]
O   /ch/01/preamp/hpslope ,i 0          | /ch/10/preamp/hpslope ,i [$0]
O   /ch/01/preamp/hpf ,f 0              | /ch/10/preamp/hpf ,f [$0]
# Gate
O   /ch/01/gate/on ,i 0                 | /ch/10/gate/on ,i [$0]
O   /ch/01/gate/mode ,i 0               | /ch/10/gate/mode ,i [$0]
O   /ch/01/gate/thr ,f 0                | /ch/10/gate/thr ,f [$0]
O   /ch/01/gate/range ,f 0              | /ch/10/gate/range ,f [$0]
O   /ch/01/gate/attack ,f 0             | /ch/10/gate/attack ,f [$0]
O   /ch/01/gate/hold ,f 0               | /ch/10/gate/hold ,f [$0]
O   /ch/01/gate/release ,f 0            | /ch/10/gate/release ,f [$0]
O   /ch/01/gate/keysrc ,i 0             | /ch/10/gate/keysrc ,i [$0]
O   /ch/01/gate/filter/on ,i 0          | /ch/10/gate/filter/on ,i [$0]
```

```
O   /ch/01/gate/filter/type ,i 0          | /ch/10/gate/filter/type ,i [$0]
O   /ch/01/gate/filter/f ,f 0             | /ch/10/gate/filter/f ,f [$0]
# Comp
O   /ch/01/dyn/on ,i 0                    | /ch/10/dyn/on ,i [$0]
O   /ch/01/dyn/mode ,i 0                  | /ch/10/dyn/mode ,i [$0]
O   /ch/01/dyn/det ,i 0                   | /ch/10/dyn/det ,i [$0]
O   /ch/01/dyn/env ,i 0                   | /ch/10/dyn/env ,i [$0]
O   /ch/01/dyn/thr ,f 0                   | /ch/10/dyn/thr ,f [$0]
O   /ch/01/dyn/ratio ,i 0                 | /ch/10/dyn/ratio ,i [$0]
O   /ch/01/dyn/knee ,f 0                  | /ch/10/dyn/knee ,f [$0]
O   /ch/01/dyn/mgain ,f 0                 | /ch/10/dyn/mgain ,f [$0]
O   /ch/01/dyn/attack ,f 0                | /ch/10/dyn/attack ,f [$0]
O   /ch/01/dyn/hold ,f 0                  | /ch/10/dyn/hold ,f [$0]
O   /ch/01/dyn/release ,f 0               | /ch/10/dyn/release ,f [$0]
O   /ch/01/dyn/pos ,i 0                   | /ch/10/dyn/pos ,i [$0]
O   /ch/01/dyn/keysrc ,i 0                | /ch/10/dyn/keysrc ,i [$0]
O   /ch/01/dyn/mix ,f 0                   | /ch/10/dyn/mix ,f [$0]
O   /ch/01/dyn/auto ,i 0                  | /ch/10/dyn/auto ,i [$0]
O   /ch/01/dyn/filter/on ,i 0             | /ch/10/dyn/filter/on ,i [$0]
O   /ch/01/dyn/filter/type ,i 0           | /ch/10/dyn/filter/type ,i [$0]
O   /ch/01/dyn/filter/f ,f 0              | /ch/10/dyn/filter/f ,f [$0]
# Insert
O   /ch/01/insert/on ,i 0                 | /ch/10/insert/on ,i [$0]
O   /ch/01/insert/pos ,i 0                | /ch/10/insert/pos ,i [$0]
O   /ch/01/insert/sel ,i 0                | /ch/10/insert/sel ,i [$0]
# EQ
O   /ch/01/eq/on ,i 0                     | /ch/10/eq/on ,i [$0]
O   /ch/01/eq/1/type ,i 0                 | /ch/10/eq/1/type ,i [$0]
O   /ch/01/eq/1/f ,f 0                    | /ch/10/eq/1/f ,f [$0]
O   /ch/01/eq/1/g ,f 0                    | /ch/10/eq/1/f ,g [$0]
O   /ch/01/eq/1/q ,f 0                    | /ch/10/eq/1/f ,q [$0]
O   /ch/01/eq/2/type ,i 0                 | /ch/10/eq/2/type ,i [$0]
O   /ch/01/eq/2/f ,f 0                    | /ch/10/eq/2/f ,f [$0]
O   /ch/01/eq/2/g ,f 0                    | /ch/10/eq/2/f ,g [$0]
O   /ch/01/eq/2/q ,f 0                    | /ch/10/eq/2/f ,q [$0]
O   /ch/01/eq/3/type ,i 0                 | /ch/10/eq/3/type ,i [$0]
O   /ch/01/eq/3/f ,f 0                    | /ch/10/eq/3/f ,f [$0]
O   /ch/01/eq/3/g ,f 0                    | /ch/10/eq/3/f ,g [$0]
O   /ch/01/eq/3/q ,f 0                    | /ch/10/eq/3/f ,q [$0]
O   /ch/01/eq/4/type ,i 0                 | /ch/10/eq/4/type ,i [$0]
O   /ch/01/eq/4/f ,f 0                    | /ch/10/eq/4/f ,f [$0]
O   /ch/01/eq/4/g ,f 0                    | /ch/10/eq/4/f ,g [$0]
O   /ch/01/eq/4/q ,f 0                    | /ch/10/eq/4/f ,q [$0]
# SENDS - remove unnecessary/unwanted ones if needed
O   /ch/01/mix/on ,i 0                    | /ch/10/mix/on ,i [$0]
O   /ch/01/mix/fader ,f 0                 | /ch/10/mix/fader ,f [$0]
O   /ch/01/mix/st ,i 0                    | /ch/10/mix/st ,i [$0]
O   /ch/01/mix/pan ,f 0                   | /ch/10/mix/pan ,f [1 $0 -]
O   /ch/01/mix/mono ,i 0                  | /ch/10/mix/mono ,i [$0]
```

O	/ch/01/mix/mlevel ,f 0	/ch/10/mix/mlevel ,f [$0]
O	/ch/01/mix/01/on ,i 0	/ch/10/mix/01/on ,i [$0]
O	/ch/01/mix/01/level ,f 0	/ch/10/mix/01/level ,f [$0]
O	/ch/01/mix/01/pan ,f 0	/ch/10/mix/01/pan ,f [1 $0 -]
O	/ch/01/mix/01/type ,i 0	/ch/10/mix/01/type ,i [$0]
O	/ch/01/mix/02/on ,i 0	/ch/10/mix/02/on ,i [$0]
O	/ch/01/mix/02/level ,f 0	/ch/10/mix/02/level ,f [$0]
O	/ch/01/mix/03/on ,i 0	/ch/10/mix/03/on ,i [$0]
O	/ch/01/mix/03/level ,f 0	/ch/10/mix/03/level ,f [$0]
O	/ch/01/mix/03/pan ,f 0	/ch/10/mix/03/pan ,f [1 $0 -]
O	/ch/01/mix/03/type ,i 0	/ch/10/mix/03/type ,i [$0]
O	/ch/01/mix/04/on ,i 0	/ch/10/mix/04/on ,i [$0]
O	/ch/01/mix/04/level ,f 0	/ch/10/mix/04/level ,f [$0]
O	/ch/01/mix/05/on ,i 0	/ch/10/mix/05/on ,i [$0]
O	/ch/01/mix/05/level ,f 0	/ch/10/mix/05/level ,f [$0]
O	/ch/01/mix/05/pan ,f 0	/ch/10/mix/05/pan ,f [1 $0 -]
O	/ch/01/mix/05/type ,i 0	/ch/10/mix/05/type ,i [$0]
O	/ch/01/mix/06/on ,i 0	/ch/10/mix/06/on ,i [$0]
O	/ch/01/mix/06/level ,f 0	/ch/10/mix/06/level ,f [$0]
O	/ch/01/mix/07/on ,i 0	/ch/10/mix/07/on ,i [$0]
O	/ch/01/mix/07/level ,f 0	/ch/10/mix/07/level ,f [$0]
O	/ch/01/mix/07/pan ,f 0	/ch/10/mix/07/pan ,f [1 $0 -]
O	/ch/01/mix/07/type ,i 0	/ch/10/mix/07/type ,i [$0]
O	/ch/01/mix/08/on ,i 0	/ch/10/mix/08/on ,i [$0]
O	/ch/01/mix/08/level ,f 0	/ch/10/mix/08/level ,f [$0]
O	/ch/01/mix/09/on ,i 0	/ch/10/mix/09/on ,i [$0]
O	/ch/01/mix/09/level ,f 0	/ch/10/mix/09/level ,f [$0]
O	/ch/01/mix/09/pan ,f 0	/ch/10/mix/09/pan ,f [1 $0 -]
O	/ch/01/mix/09/type ,i 0	/ch/10/mix/09/type ,i [$0]
O	/ch/01/mix/10/on ,i 0	/ch/10/mix/10/on ,i [$0]
O	/ch/01/mix/10/level ,f 0	/ch/10/mix/10/level ,f [$0]
O	/ch/01/mix/11/on ,i 0	/ch/10/mix/11/on ,i [$0]
O	/ch/01/mix/11/level ,f 0	/ch/10/mix/11/level ,f [$0]
O	/ch/01/mix/11/pan ,f 0	/ch/10/mix/11/pan ,f [1 $0 -]
O	/ch/01/mix/11/type ,i 0	/ch/10/mix/11/type ,i [$0]
O	/ch/01/mix/12/on ,i 0	/ch/10/mix/12/on ,i [$0]
O	/ch/01/mix/12/level ,f 0	/ch/10/mix/12/level ,f [$0]
O	/ch/01/mix/13/on ,i 0	/ch/10/mix/13/on ,i [$0]
O	/ch/01/mix/13/level ,f 0	/ch/10/mix/13/level ,f [$0]
O	/ch/01/mix/13/pan ,f 0	/ch/10/mix/13/pan ,f [1 $0 -]
O	/ch/01/mix/13/type ,i 0	/ch/10/mix/13/type ,i [$0]
O	/ch/01/mix/14/on ,i 0	/ch/10/mix/14/on ,i [$0]
O	/ch/01/mix/14/level ,f 0	/ch/10/mix/14/level ,f [$0]
O	/ch/01/mix/15/on ,i 0	/ch/10/mix/15/on ,i [$0]
O	/ch/01/mix/15/level ,f 0	/ch/10/mix/15/level ,f [$0]
O	/ch/01/mix/15/pan ,f 0	/ch/10/mix/15/pan ,f [1 $0 -]
O	/ch/01/mix/15/type ,i 0	/ch/10/mix/15/type ,i [$0]
O	/ch/01/mix/16/on ,i 0	/ch/10/mix/16/on ,i [$0]
O	/ch/01/mix/16/level ,f 0	/ch/10/mix/16/level ,f [$0]

```
# DCA/Mute group
O   /ch/01/grp/dca ,i 0                 | /ch/10/grp/dca ,i [$0]
O   /ch/01/grp/mute ,i 0                | /ch/10/grp/mute ,i [$0]
# Automix
O   /ch/01/automix/group ,i 0           | /ch/10/automix/group ,i [$0]
O   /ch/01/automix/weight ,i 0          | /ch/10/automix/weight ,i [$0]
#
# end of file
```

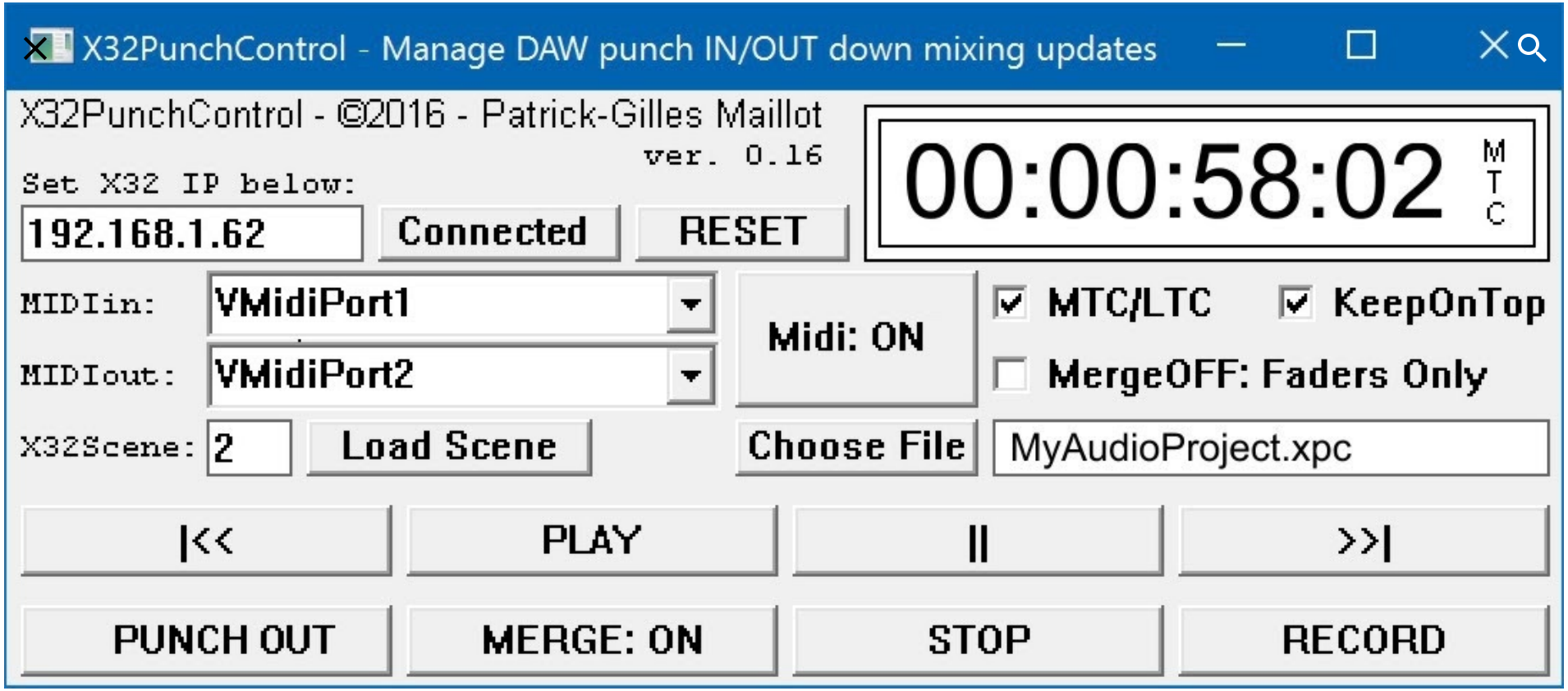
Such tool can enable you to control for example a lighting system, certain features of a DAW connected to X32, or even another X32 family member. See it in action: https://youtu.be/W_UQt0YdnhU

X32PunchControl

Control your DAW while playing audio, recording or playing back X32 commands.

Many DAW software offer “automation”; a way to record and play DAW changes (volume, pan, effect settings, etc.) as audio is recorded to or played from the DAW. Nevertheless, this is limited to the DAW capabilities, and does not offer the physical comfort of a control surface for managing mix changes.

Many users with an X32 will want to use the large palette of audio capabilities the X32 offers, from basic multi-track mixing, FX assignments and changes, along with their DAW software.



X32PunchControl enables the missing link between your DAW software and X32, under the DAW timing control

X32PunchControl will record all X32 OSC commands respective of mixing, channels and FX controls, routing, etc. into a file of time-stamped records. These records can later be played again, in sync with the audio from DAW to X32, thanks to the recorded time stamp and the MIDI time code from the DAW to X32PunchControl. These records can also be modified in real time when playing DAW audio and the already recorded data, via standard punch in/out operations. This way you can incrementally enhance your mix, by applying successive modifications. When re-positioning the DAW song cursor prior or further to the current position, X32PunchControl will manage keeping up with the DAW software, replaying (catching up) OSC commands to ensure the X32 state is what it should be at the cursor position, so your mix, controls, effects are all correctly set.

In all cases the data is always recorded in real time to a file. X32PunchControl typically uses 2 files, one for reading only (if a previous recording has taken place) and one for writing only. When done, the "writing" file becomes the "reading" file for the next update or play X32PunchControl session. In order to start from a known state of settings, routing, etc., X32PunchControl enables loading an X32 scene file from the X32 scene library section.

Most transport options and commands needed during normal workflow are under the control of X32PunchControl. This can be done from the program GUI, clicking on transport buttons (REW, PLAY, PAUSE, PUNCH, RECORD, STOP, FF), or using a user set of buttons, directly from the X32. X32PunchControl enables assigning transport buttons to a set of X32 Buttons of the User Assign Section.

***Note**: If you are interested in using your X32 to the fullest in your music studio, I would suggest you consider X32ReaperAutoMate or X32XLiveAutomate; These two programs offer a flexible setup, a lot more functionalities, a much faster recording engine and better file management system, and exclusive functionality that can save a lot of time and hassle in the studio for recording or mixing audio from a band, and also for Productions, Musicals, and Theater plays.*

X32Tap

Sets a Tempo delay from a button, a value or automatically from audio.

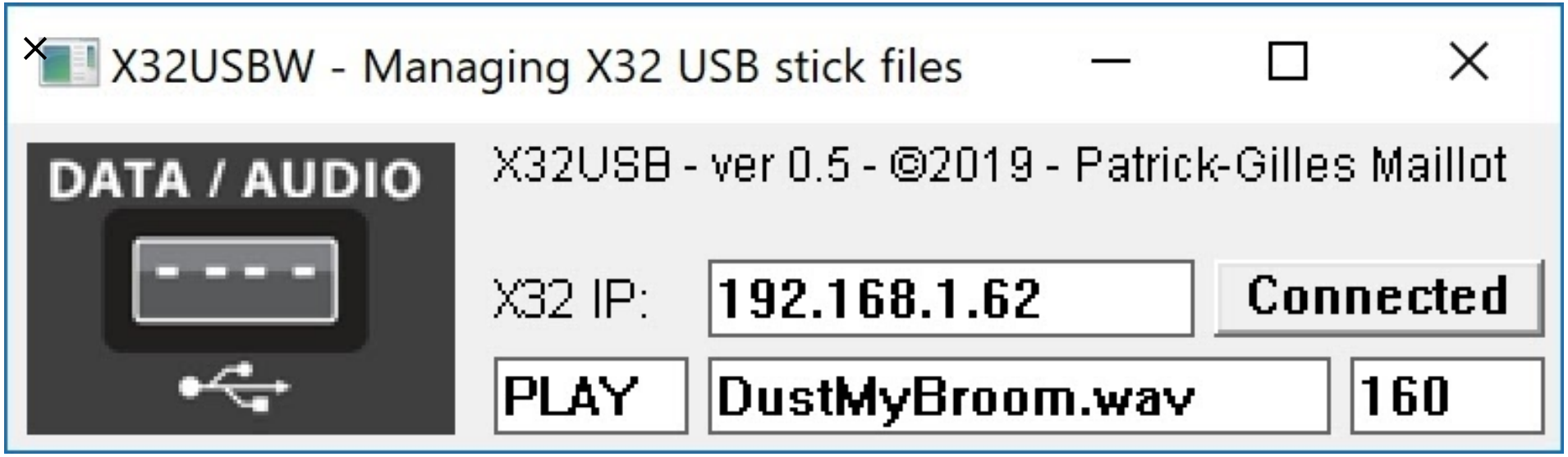
A Windows utility capable of setting Tap Tempo for any delay capable X32 effect, either from tapping tempo on the GUI button, keyboard ctrl-t key, or getting the tap tempo from audio signals on a chosen X32 Channel.

It checks for a delay capable effect to be on FX slots 1 to 4. If an effect is found, tempo can be set by clicking on the TAP button, or (can be simultaneous) from an audio channel; In all cases, tempo tap can be set between 0 and 3000ms. Please read the documentation (included with the package) on how to best use the "auto" mode enabling audio tempo setting.

Ver .35 also accepts 'tap' from keyboard 'ctrl-t' key when the application has Windows' focus (i.e. mouse pointer in the application windows).

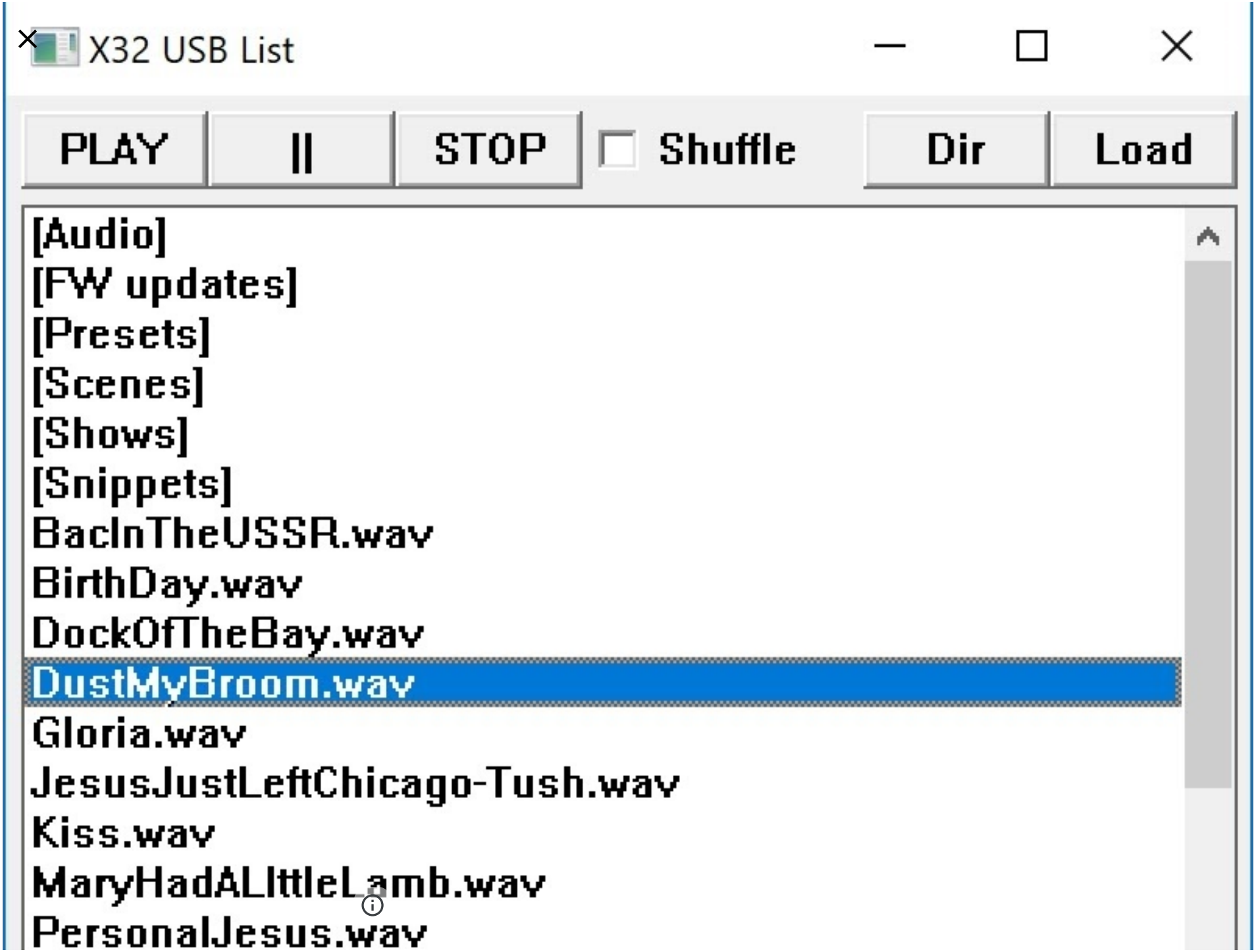
X32USB

A simple Windows tool to list, play, load, execute files contained in the USB drive attached to X32.



This program is a Windows application version of a command line utility (not listed here). It starts with connecting to the X32 by specifying the IP address of the X32 and clicking on **Connect**. When this happens, a second window opens with the list of directories and files contained in the current directory of the USB stick.

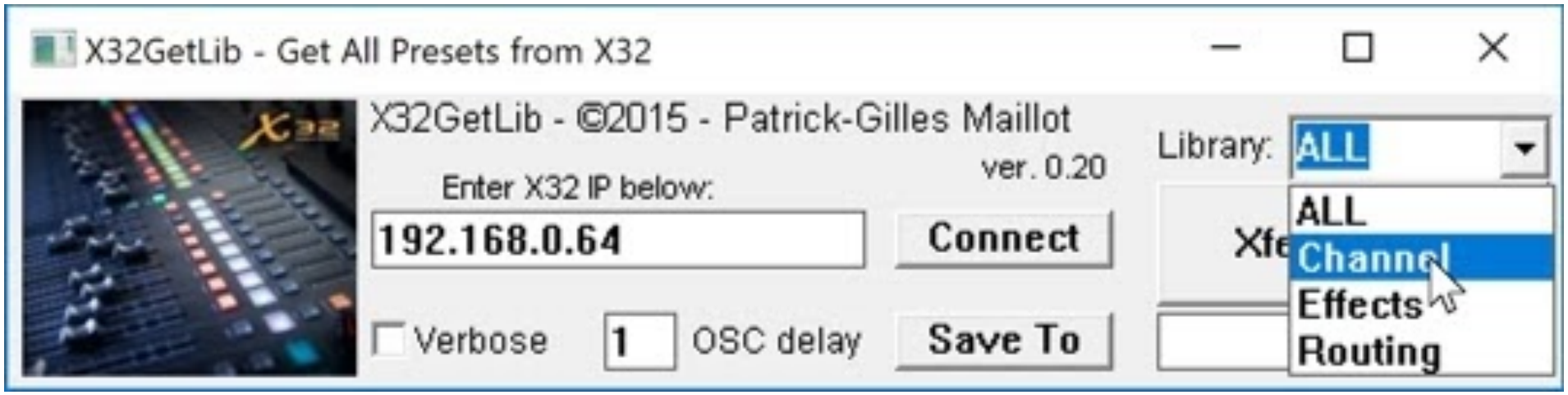
From this window, one can select a file or directory and click on an action (**PLAY**, **PAUSE** or **STOP** for wav files, **Dir** for entering and listing a selected directory, and **Load** for loading an X32 scene, snippet, or preset.


X32 USB List
PLAY
||
STOP
Shuffle
Dir
Load
[Audio]
[FW updates]
[Presets]
[Scenes]
[Shows]
[Snippets]
BacInTheUSSR.wav
BirthDay.wav
DockOfTheBay.wav
DustMyBroom.wav
Gloria.wav
JesusJustLeftChicago-Tush.wav
Kiss.wav
MaryHadALittleLamb.wav
PersonalJesus.wav

The main window reflects the file currently playing, if any, as well as the PLAY, PAUSE or STOP status. At the right of the displayed filename is a decreasing counter showing the remaining time (in seconds) to the end of the file, before automatically jumping to the next file in the list.
Automatic playing of the file next to the current one is limited to the current directory.
If the **Shuffle** button is checked, the play list is played in a random order.
If the PLAY request is made on a non playable file (a folder for example), the application will search the next available WAV file to play and start from there.
The user can navigate to any other directory while a file is playing.

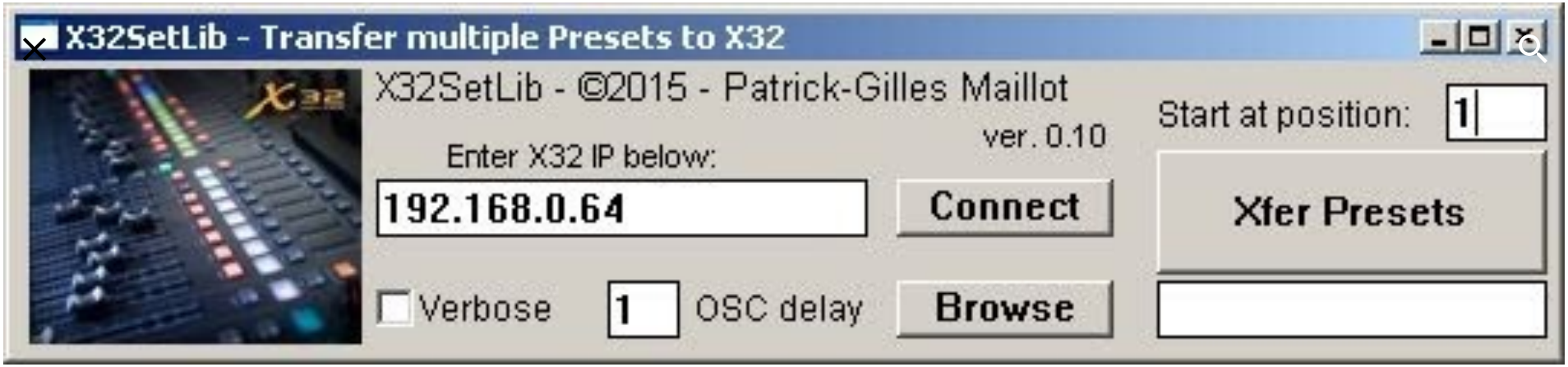
X32GetLib, X32SetLib

Manage your Presets (Channel, Effect, Routing, AES/DP48) on your PC from your X32.
No more limits and a lot of flexibility to change settings and values for your preset files.



Connect to X32, select a destination directory on your PC for Preset files. Select the preset library you want to retrieve (Channel, Effects or Routing). Hit the "Xfer Presets" button to start the transfer. A status will display as the transfer progresses and when transfer is complete.

Starting with ver. 0.20, X32GetLib enables to save ALL presets (Channel, Effects, Routing, and AES/DP48) in a single click.



Connect to X32, browse your PC for Preset files (Channel, Effect or Routing) and select one or multiple presets to transfer to X32 Presets library.

Set the position for the first preset in the library, positions will increment automatically, as presets are saved in the X32 library.

Hit the "Xfer Presets" button to start the transfer. A status will display as the transfer progresses and when transfer is complete.

Important note: All your preset files should be version 4.0; You can easily check this with a text editor (file begins with a #4.0# header) if not, you can actually set the header manually, as this does the trick in most situations.

X32Fade

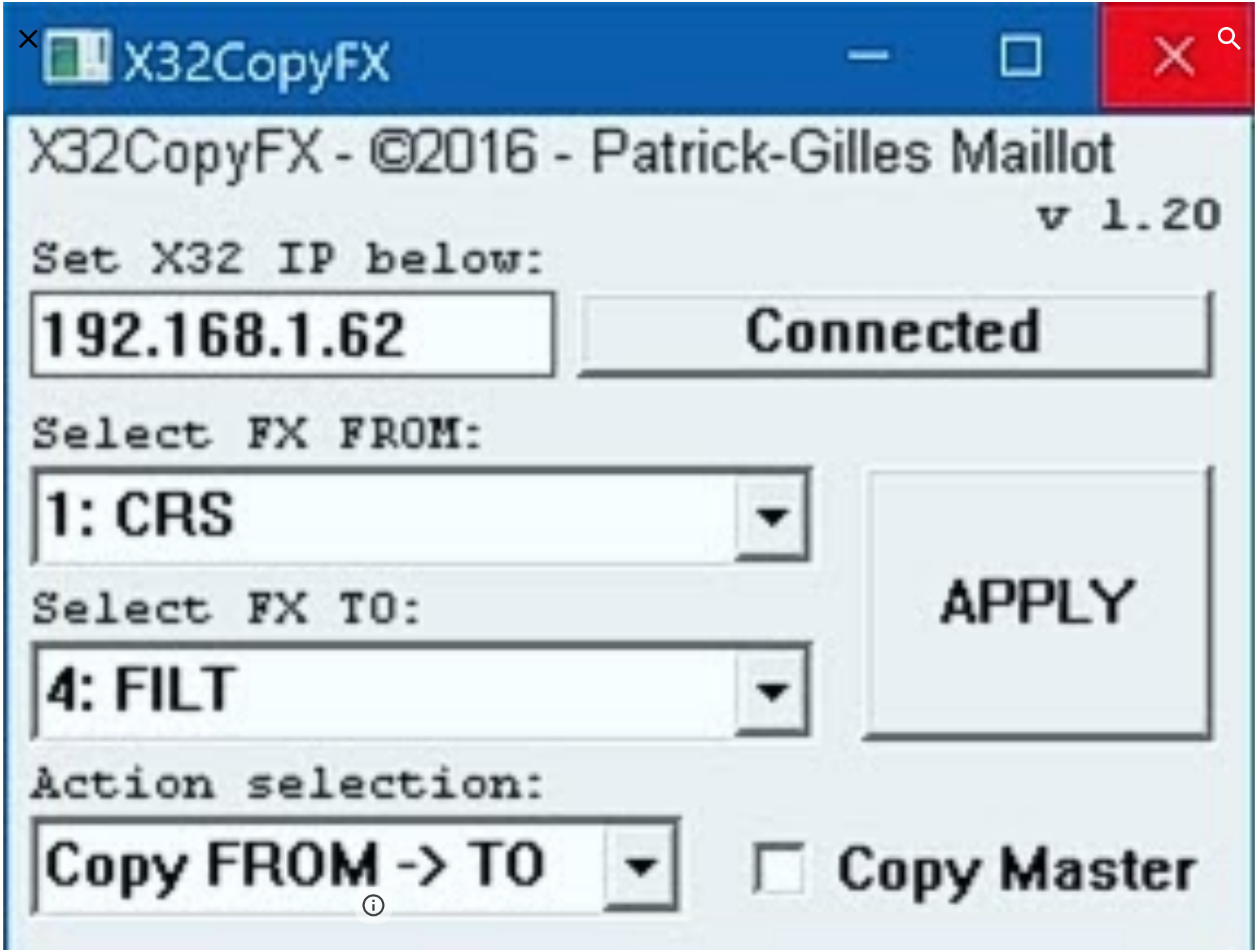
An automated Fade-IN / Fade-OUT tool for X32

The program reads user commands from a GUI to select which X32 Faders (or levels) should be controlled. It opens a communication stream and generates the appropriate commands to change selected level values down (Fade-out) or up (Fade-in) over a given amount of time.

Watch a video of X32Fade in action. Please read the documentation to decide if this is for you. Then download and use the program!

v 2.0 includes USER ASSIGN buttons and MIDI controls

X32CopyFX


X32CopyFX
X32CopyFX - ©2016 - Patrick-Gilles Maillot
v 1.20
Set X32 IP below:
192.168.1.62
Connected
Select FX FROM:
1: CRS
Select FX TO:
4: FILT
APPLY
Action selection:
Copy FROM -> TO
Copy Master

X32CopyFX Reset or Copy sides or full parameters set from an effect to itself or another.

This small utility is the Windows GUI equivalent to the command line X32GEQ2cpy tool which runs as a Unix or Windows command line utility, with a few additions.An automated Fade-IN / Fade-OUT tool for X32

The program reads user commands from a GUI to select which X32 Faders (or levels) should be controlled. It opens a communication stream and generates the appropriate commands to change selected level values down (Fade-out) or up (Fade-in) over a given amount of time.As one launches X32CopyFX, the tool displays 5 fields the user has to fill;

The first one to fill is for the X32 IP address. As soon a the IP address is entered and the user clicks on "Connect", the tool request from the X32 its set of currently loaded in the X32. The list of available effects is loaded into the two combo-boxes left of the APPLY button. The user then selects a source (FROM) and destination (TO) effect slot. A third combo-box lists the possible operations: reset, copy side A to side B, or the opposite (applies to the FROM effect), or copy the FROM effect data to the TO effect. The "Copy Master" check box enables the user to select if the selected operation also applies to the Master parameter (#32 and #64 for EQ types).

Starting with version 1.20, The tool offers two RESET types of effects settings to their respective default values. Choices are to reset only the effect in the FROM: box, or all effects from the FROM: to the TO: boxes; each effect will receive their respective default values.

Starting with ver 1.30, X32CopyFX will accept an optional "-f " argument to the main program. If such option is passed on, the provided by the user (a text file) is expected to hold pairs of lines in the form:

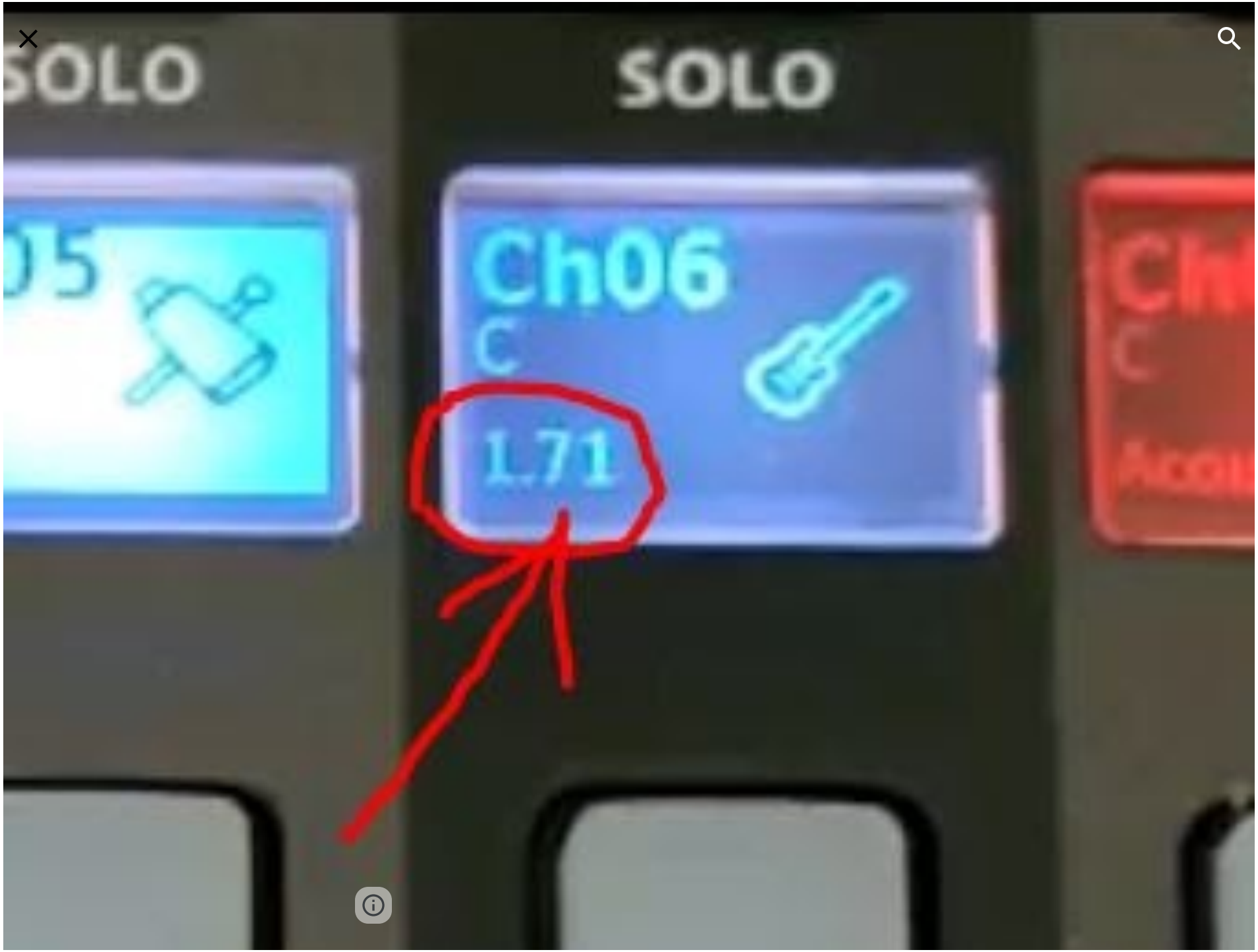
```
{
 effect name
 effect parameters values
}
```

This enables to set default values for effects to user preferences rather than default X32 values, for example the file below will overwrite HALL and PIT default values, all other effects will keep the X32 standard default values.

```
<myfx.txt>:
HALL
40 1.57 60 5k74 25 0.0 83 7k2 0.95 25 50 40
PIT
-12. 0 5.0 52 15k8 80
```

The APPLY button starts the selected operation. The button momentarily (1 second) displays "DONE" as the operation progresses.

NOTE: Microsoft Defender seems to inappropriately report this program as potential malware: X32CopyFX uses (arrays of) pointers and can overwrite some of these pointers (managing memory correctly) to replace X32 effects parameters default values by user sets. While a very effective way of doing this (from a memory management point of view), it can be something considered “suspicious”.


SOLO
SOLO
Ch06
C
L71

X32FaderDisp

MidiOscIttt

OSC based command line program program to display X32 fader values (in dB) in real-time on the respective X32 channel strip scribble:
An OSC & MIDI based "If This Then That" program

MidiOscIttt is a simple “If This Then That” program with triggers based on MIDI or OSC data input.

The Output is also MIDI or OSC data. While there are many services and applets capable of handling IFTTT scenarios for MIDI, and some for OSC, very few handle both simultaneously and as a standalone application.

MidiOscIttt does not rely on external http servers, doesn’t need an internet connection (but needs ethernet for handling UDP/OSC datagrams) and can be tailored to your needs in order to perform scenarios you decide, based on a text-only definition file (no JSON involved).

MidiOscIttt can also launch applications based on the received MIDI or OSC data.