MOBILE I/O

Mobile I/O v1.5d15

Mobile I/O v1.5d15 (3/24/03):

This release is a major update to the OS X components of the MIO Driver, Firmware and Application software. The OS 9 components provided in this release have not changed from PP#2. The Firmware included with the OS X components is compatible with the OS 9 driver, and may be used on boxes that will be used both in OS 9 and OS X. This package is appropriate for use with all Mobile I/O models (2882, 2882+DSP, ULN-2, & ULN-2+DSP). It is also compatible with OS 9, OS X Jaguar (10.2.x) and OS X Panther (10.3.x).

This version of the software provides significant enhancements to the OS X driver, as well as resolving a number of issues in the OS X version of MIOConsole. All of the enhancements delivered with PP#2 remain available in the package.

In order to get the benefit of this version, you must install the new driver and console and update your firmware.

There is no need to uninstall a previous version of this driver before installing this version. This installer is appropriate for both new installations and upgrades.

The CoreAudio driver included in this package supports the attachment of multiple Mobile I/O boxes to your computer. Each box will appear as a separate CoreAudio engine. If your audio application supports multiple CoreAudio engines simultaneously (like DP4), you will be able to use more than one box in the application. If the app does not support multiple engines, you will have to wait for a future driver that presents multiple boxes as one CoreAudio engine (or lobby your app vendor to support multiple engines).

The original Macintosh hardware with built in Firewire support only has enough isochronous resources to support audio transport to and from four Mobile I/O boxes per FireWire bus. New Macintosh hardware (units with FW800 ports) have enough isochronous resources to support audio transport between the Mac and up to eight Mobile I/O boxes per FireWire bus. In any case, more boxes may be placed upon the bus and controlled by MIO Console — the resource limitation is only for audio transport.

This implementation of the driver provides support for all sample rates and all I/O channels and all members of the Mobile I/O family, including ULN-2. This implementation provides support for OS 10.2.x and OS 10.3.x on all Macintosh models that support these OS versions.

Please check out the change list below to understand what has changed since the last release (PP#2).

Recent Firmware Changes:

This version of the firmware makes the following enhancements to previous versions:

1) The firmware now provides a mechanism to allow the driver to precisely determine when a particular sample will be sent to the physical outputs. The new driver uses this to precisely fix the output latency. See below for more information.
2) The firmware reduces the output latency by 16 samples relative to previous versions of the firmware.
3) The firmware now supports an enhanced precision S/R reporting mechanism that the driver uses to track external Sample Rate changes. See below for more information.

Recent Driver Changes:

This version of the driver makes the following enhancements to previous versions:

1) The driver now names its channels differently for each Mobile I/O Unit attached to the computer. The channel names are now prefixed with the serial number of the box. This change works around an issue in Digital Performer's built-in multibox support and enables the use of multiple boxes simultaneously with DP4.
2) The driver enables all DAW output channels, bringing it to parity with the OS 9 driver. You will be able to send 18 channels out to each 2882 @ 44.1k and 48k. At 88.2k and 96k the number of active outputs is limited to 14 (this accounts for the 4 channel reduction in ADAT output with SMUX operation).
3) The driver further reduces the overall output latency.
4) The driver, coupled with the new version of the firmware (v. 1_5_03 — included in the update package) causes the output latency to be exactly the same every time the driver starts. This was not the case in the past — the output latency would change by a small amount each time the driver was started; this shift would show up in the record offset in overdub recording. As of this version of the driver/firmware, the output latency is fixed at each sample rate.
5) The driver reports accurate input and output latency values to host applications via CoreAudio. This means that with all current versions of the major applications, the record offset during overdub recording is 0 samples. The

proper latency is reported for all sample rates and buffer sizes.

6) We have made improvements to the driver's handling of both small and large buffer sizes, especially with DP4 (which sometimes had problems with very small or very large buffers and the previous version of the driver).
7) The driver fixes a problem with changing the sample rate while audio apps are running.
8) The driver fixes a problem that occurs when a safety booted box is attached to the system in OS X.
9) The driver now automatically detects the hardware's sample rate and reports it to CoreAudio — thus keeping everything synchronized — even if the box is on external clock and the external clock source changes...
10) The driver now has a new Metric Halo technology — adaptive Core Audio timestamping. This lets the driver report the active stream rate in a very accurate way — and it lets clients run at higher CPU loads.
11) The driver fixes a subtle issue that could lead to spikes in the Client application's CPU load.
12) The driver removes most of the debugging logging that was present in earlier versions. This reduces the amount of information written to the system.log file. The logging facility is still available in the driver, and under the appropriate circumstances may be enabled with instructions from MH technical support

Recent MIOConsole Changes:

This version of MIOConsole makes the following enhancements to previous versions:

1) This version fixes a problem that appeared in OS 10.3 that would cause a Firewire bus reset when MIOConsole was quit by the user. If audio was running in another application, this bus reset could cause a disruption in the audio, distortion, or in some cases cause the other app to hang. This has been fixed.
2) This version fixes a problem where unlinking mix busses on some hardware models would cause the console to crash the next time it was launched, or when a saved state file with unlinked mix busses was opened.
3) This version adds a readout in the Box Info pane that reports the currently running driver version, as long as the driver is v1.5d15 or newer.
4) This version automatically checks and updates the reported sample rate based upon the actual rate of the running hardware. Coupled with v1.5d15 of the driver, this allows MIOConsole to automatically track external clock changes. The complete package automatically reports the S/R changes to every CoreAudio client application, including +DSP. Since the Apple iLife apps (like iTunes and GarageBand) automatically SRC to match the reported sample rate, these apps will continue to play through Sample Rate changes with no pitch shifting.
5) This version adds four new +DSP plugins to the plug-in set: MIO Channel Summer, MIO Channel Differencer, MIO Channel Sum/Differencer, and MIO Delay. These plugs are not particularly fancy, but they are nice basic building blocks for making more complex routings for your own custom processes.
6) This version makes some small enhancements to the drawing performance of MIO Console when it is on one of the metering pages.
7) This version adds new commands to the Utilites menu. These commands allow you to reset various aspects of the Mobile I/O hardware without having to disconnect the hardware from your computer.

   The "Reset ... Output" commands reset the Digital to Analog converters in the MIO Hardware, and can be used to restore the outputs if they go silent after a bad clock condition. If you have meters running on the hardware, but no analog output, use this command to fix the problem.

   The "Reboot ... MIO" commands will physically reboot the MIO hardware, as if you had disconnected power from the boxes. If your system has gotten hopelessly confused, you can use this to "start over". This is almost the same as disconnecting the box from your system, so you should stop your audio apps before you run this command. After the box(es) have rebooted, you should quit MIOConsole and relaunch it to reset the boxes back to your desired configuration.

Known Incompatibilities and Issues:

1) This package requires Mac OS 10.2 or newer.
2) The onboard Firewire that is provided on Apple's first generation FW800 machines (the Single and Dual MDD G4 units) is currently incompatible with this driver. The symptom is that after playing for 30–60 seconds, the sound will start to "fizzle out" and will eventually drop away to silence. Starting and stopping your CoreAudio app will provide another minute or so of sound. This does not happen if you attach your Mobile I/O(s) to a PCI Firewire card (either FW400 or FW800). We are currently investigating this issue; our recommended work-around until we find a resolution is to install a PCI Firewire card in your machine and attach the Mobile I/O units to the PCI card. Any OCHI based Firewire card supported by OS X will work fine.
3) The combination of 2x sample rates (88/96k), non-MIO activity on the firewire bus (this may include the activity of other Firewire interfaces, DV, disks, and devices like the Firewire Powercore), and active meters in MIO Console can lead to distortion in the audio. Workaround: if you will be working at 88/96k, and need to interact with MIO Console at the same time, consider placing the Mobile I/O(s) on separate Firewire bus than your other active devices. Please note that multiple motherboard Firewire connectors on a machine are not, in fact, separate Firewire busses. You need a PCI or PC card to add another bus to your system.

If you encounter an error in this software, please send a detailed bug report to Metric Halo. In order to ensure that the bug report is routed properly, please include the text **MIOBUGX** in the subject line of the email message.

Installation
Installation is straightforward. This package is currently composed of three files:

1) **Firmware** – The firmware update procedure is the same as before in OS X. You apply the firmware update from the MIOConsole "Utilites" menu. In order to get rock-solid latency compensation, your box must have firmware rev. 1.5.03 or newer. After completing the firmware update, we recommend that you reboot your computer and power-cycle the box.

2) **MIOConsole** — You can drag install this application to put it on your hard-drive. This version (1.5d15) of the Console should be compatible with previous Firmware revisions.

3) **MobileIODriver_1.5d15.pkg**
This is an Apple Installer package that installs the Mobile I/O CoreAudio driver. You need to have an administrator password to install this package on a machine. After the installation is complete, you will be requested to reboot your computer. After the reboot, your Mobile I/O is OS X capable. Have fun!

Documentation from the PP#2 release:

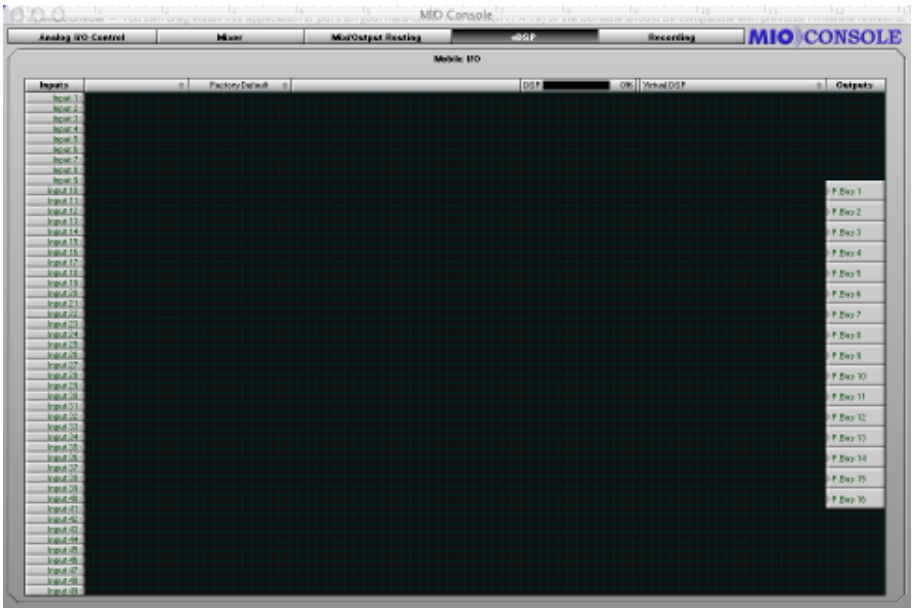
PRIMARY LIMITATION: This version does not support +DSP at 88.2k and 96k. Sorry for the inconvenience.

Known Issues:
1) This package requires Mac OS 9, Mac OS 10.2.x or Mac OS 10.3

This version of the MIOConsole adds a new Panel to the Console window. The main Panels are now:
1) Analog I/O Control
2) Mixer
3) Mix/Output Routing
4) +DSP

The functions of 1, and 2 have not changed. The "Mix/Output Routing" Panel (3) now has Process busses available in the Patchbay router, and we have added (4) the +DSP Panel.

The +DSP Panel is the heart of the plug-in system in Mobile I/O:



The +DSP Panel includes a number of new elements. The large dark area with the grid is the "Graph area". To the left of the Graph area are the input ports from the primary DSP (the primary DSP is the DSP that has been providing the mixing, metering and routing services on Mobile I/O since the beginning). On the right are the output process bus ports that return the processed signal to the primary DSP. Along the top of the Graph area are a number of controls that you use to configure the DSP signal processing. Right next to the "Inputs" label, is the Plug-in pop-up menu. Next to that is the Patch

library popup menu. The header then has a spacer, followed by a DSP load meter. Next to the DSP load meter is a DSP popup menu. These are all described below.

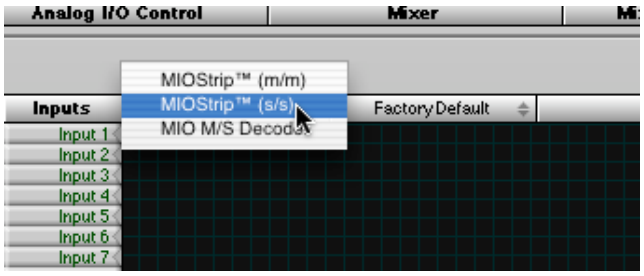
DSP pop-up menu

We'll start with the **DSP pop-up menu.** This is important! When the console starts up, this menu will be set to "Virtual DSP" which is sort of a Playground demo area that can be used to create patches, but will never actually run them. In order to actually use +DSP, you will need to select the DSP that you want to work with. When you click on this pop-up, you will be given a list of available DSP's in the system. There should be one for each +DSP box you have attached to your computer. They will be listed by box type, serial number, and DSP number. All the +DSP units have one available DSP (DSP #1). The software is capable of addressing any number of boxes, so if you have more than one 2882+DSP, you should definitely try working with multiple DSP graphs.

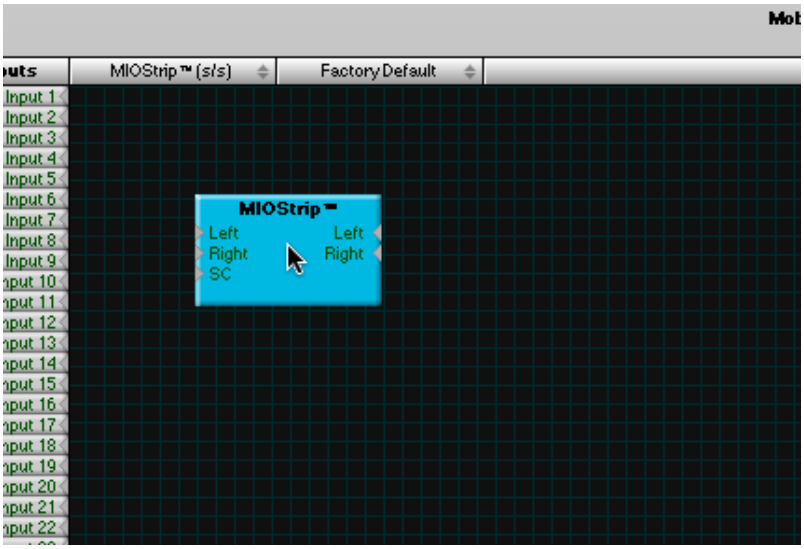
Once you have selected the DSP that you want to use:

Plug-in pop-up menu

The **Plug-in pop-up menu** contains all of the available instantiable plug-ins. When you select a plug-in from this menu, a new instance is created on the selected DSP, and you may drag the instance to a convenient location in the Graph area.

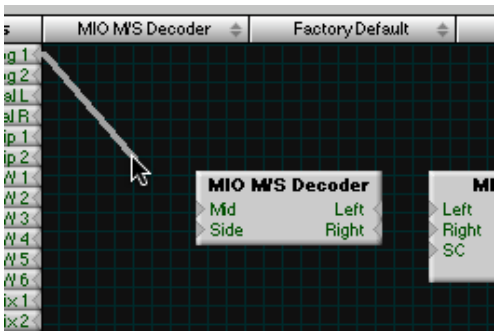


Selecting a new instance from the Plug-in Menu.

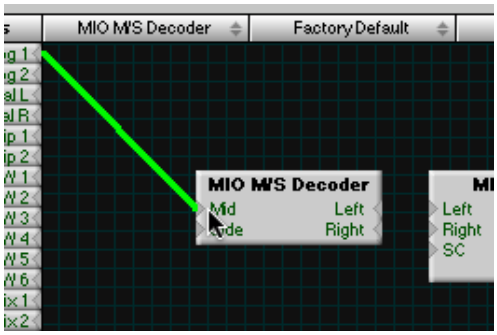


Positioning the new instance in the Graph.

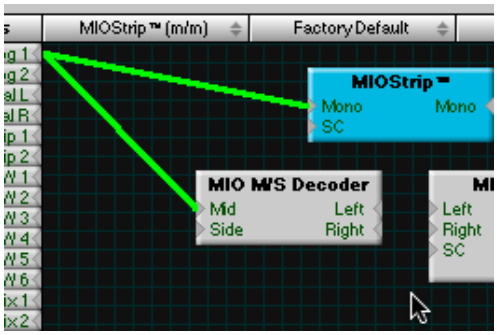
Once you have added the Plug-ins you want to use, you can wire them up (actually, you can wire and add plug-ins in any order, as you like). To make connections, click on a port (one of the small gray triangles next to the port name), and then drag the connection to the target (you can go from input to output, or from output to input). When you have made a valid connection, the connection line will switch from Gray to Green. You can make as many mults as you like of a signal source but only one connection can be made to a processor input or process bus port. If you make a new connection to an input that already has a connection, the old connection will be automatically disconnected. To remove a connection without establishing a new one, <control>-click on the **input** port to which the connection is made.



Starting a connection.

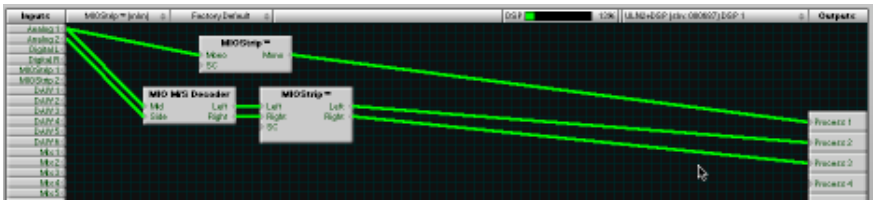


Completing the connection.



Making a mult.

After everything is placed and wired up, you will need to ensure that you have routed the output of your signal processors to the appropriate process bus on the output side of the graph. When you are done, you will have a complete graph. For example:



A complete +DSP Graph.

This graph routes Analog 1+2 into a M/S Decoder, and mults Analog 1 to a Mono MIOStrip. The output of the M/S Decoder is routed into the inputs of a Stereo MIOStrip. The outputs of the MIOStrips are sent to Process Busses 1, 2, and 3. It is important to connect both the input and output of a plug-in if you want to use it. The +DSP system is very intelligent about scheduling resources, and will only run plug-ins when they have at least one input and one output connected.

The graph is continuously modifiable. You can drag the plug-ins around as you like, and you can add new plug-ins, even while you are processing audio with the existing graph. You can make and break connections as you please.

There is no routing delay within the plug-in graph. So if you make mults with different plug-in paths on each side of the

mult, the two paths will remain time-aligned. This allows you to configure parallel processing paths without the virtually impossible task of time-aligning the parallel paths. The graph itself has a 16 sample delay from input to output, so signals routed through the graph get back to the primary DSP 16 samples after they leave it.

At this point, you will want to be able to control each plug-in's parameters. We will describe the Plug-in UI's later, but, in order to open a plug-in's UI, all you have to do is double click the plug-in in the graph.

Patch Library Pop-up Menu

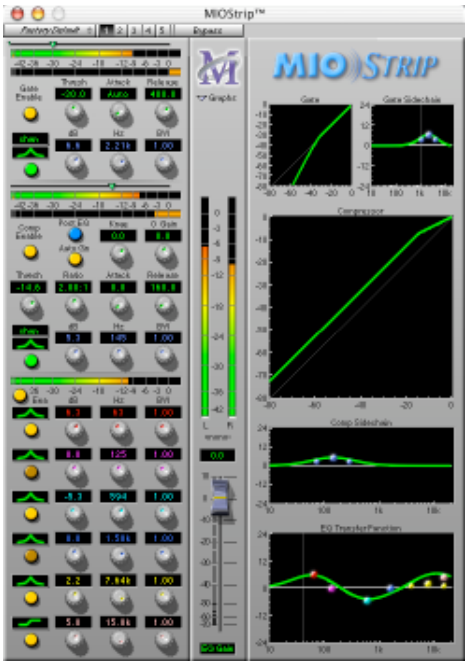
The **Patch Library pop-up menu** allows you to save the complete state of the plug-in graph. MIOConsole does not currently save the +DSP state with the console state file, so if you wish to preserve the state of the DSP graph, you need to use this pop-up menu to save the settings. This menu works like all the other Library popup menus in MIOConsole. Please note that since we are not providing any presets with this beta, there are no categories created for the library yet. If you try to save a setup without a category the program will crash. We are aware of this and are working to correct it. In the meantime, make sure you create at least one category before you try to save any set ups. The factory default setup is an empty graph, and will clear your current configuration. When you recall a setup, the audio running through the graph will mute for a very short period during the change over — generally < 100ms.

DSP Load Meter

The **DSP Load Meter** shows you the current active, measured load on the DSP that is selected in the **DSP pop-up menu.** Since the plug-ins in Mobile I/O are dynamic, this is the best way to determine if you are close to overloading the DSP. The Plug-ins in Mobile I/O only use the resources they need to accomplish the job that you have selected with the plug-in settings. So, if the compressor in MIOStrip is not enabled, it will not use up DSP cycles. This means that you can instantiate more plug-ins if you do not use all the features of each instance. But it is possible to overload the DSP by enabling features that utilize more DSP than is available. We are examining how to allow the plug-ins to operate dynamically, and still protect you from DSP overload. This will appear later. In any case, the Load Meter shows you the actual dynamic load, and will vary with the settings of the plug-ins and may vary with the audio program material (depending on the DSP algorithm; e.g. the compressor uses more DSP when the signal is above threshold than when it is below threshold).

Plug-in UI's

To open the UI for a plug-in, double click the plug-in in the graph. If you double-click the same plug-in more than once, multiple UI's will open for the same plug-in. This will change soon, but we will note that the UI windows will remain synchronized as you manipulate them.



The M/S decoder uses a generic interface — one that is automatically created from the parameters in the plug-in. The MIOStrip uses a custom interface — one generated by us with a specific layout and special UI elements. All of the plug-in UI's share the plug-in bar at the top of the window. This bar provides generic services for managing the state of any plug-in.

Every plug-in window has a parameter library popup, 5 setup registers and a master bypass button. The parameter library popup is like all the other parameter library popups in the console. The parameter library is automatically shared amongst all instances of a particular plug-in type. Actually, it is automatically shared amongst all instances of compatible plug-in types, so MIOStrip Mono and MIOStrip Stereo automatically share preset libraries. As with the **Patch Library pop-up menu** described above, there are no predefined categories by default. Please be careful to create at least one category before saving any presets — otherwise you will crash when you go to save the preset.

The 5 setup registers are a unique feature of +DSP. Each button corresponds to a set of parameters. If the button has not been activated, the register will be clear, and nothing will happen when you click on the button. When you click onto another register button, the current plug-in parameters will be saved into the register button. This allows you to make multiple alternate setups, and instantly switch between the setups. Sort of an A/B/C/D/E switch. The state of the register buttons is not saved automatically, so, if you come up with a setting you like, be sure to save it to a named preset in the parameter library.

Finally, the bypass button in the header bar is a master bypass for all processing in the plug-in.

The processing in MIOStrip is a completely new set of algorithms. While we worked to stay within the original spirit of ChannelStrip, the algorithms are radically different. We believe that you are really going to enjoy the sounds you can get out of these processors. The Compressor, especially, is a dramatically different beast. While you can still get the kinds of sounds that you could from ChannelStrip (for those of you who loved that aspect of ChannelStrip), you can get many new sounds that ChannelStrip could not create, and, in particular, you will find that it behaves like a standard compressor in its controls and behaviors, unlike ChannelStrip (for those of you who hated that aspect of ChannelStrip).

NOTE: You may experience clicks in the audio if you change parameters rapidly, this is something that we are aware of and currently tweaking.

Additional Notes:

1) Multibox firmware update. The Update Firmware command now acts on the box represented by the selected tab in the MIOConsole window.

2) Firmware progress feedback. During Firmware update, a progress window now appears which indicates the box being updated, the firmware version that it is being updated to, and ongoing progress during the update procedure.

3) Multibox persistent state management. The "Save Boot State..." and "Save Snapshot x State..." commands now act on the box represented by the selected tab in the MIOConsole window.

4) UI now always matches the config for the active box.

5) Multibox state documents are now supported (and are the default). These state documents are not backward compatible. These state documents include all the state data for every box in the system, including +DSP configuration. State recall is now tied to S/N — state data is automatically mapped to the proper box. If the box is not available, the state is recalled to an OFFLINE box representation which can be manipulated, and will be automatically asserted if the box is reattached.

6) Cross-box state may be saved and recalled via box import and export. Old console state files maybe imported to the selected box. New export state files contain box-type info, and will not import into the wrong box-type. Old console state files do not contain box-type info and will import into the wrong box type, so user beware. New export state files also include +DSP state data for the box.

7) A bunch of hardware <-> console UI bugs have been fixed (e.g. Mute/Dim button synchronization, lock state indication).

This version of the Console and plug-in set contains a significant number of changes. The primary thing you have to be aware of (and is not obvious just from looking at the software), is that the UI interaction model for plug-ins has changed. Now, in addition to being able to double click on a plug-in tile to bring up a plug-in window, we have added a shared master plug-in window that will show the selected plug-in UI. In order to turn on this browsing, you must click the Enable PI Window button in the header bar of the graph. When this button is engaged, clicking on a plug-in tile in the graph will put the plug-in's UI into the floating PI window. Clicking the same tile again will remove the UI. The when a Plug-in's UI is in the floating window, the Plug-in's tile will have a green bar under the name of the plug-in. If you double click a plug-in, a separate window will open for the Plug-in's UI. Before, each time you double-clicked, a new window would open. Now, there is only one per-instance window, and it will be brought to the front of the Plug-in windows if it is already open. This window will properly maintain the snapshots until you delete the Plug-in instance. The snapshots are still not auto-saved with the graph state (we are still thinking about this). In addition, all plug-in windows are now floating.

You can use Command-B to toggle the visibility of all floating windows.

You can use the Command-P to toggle the visibility of the shared master plug-in window. If you hide the shared master plug-in window, it will not be made visible automatically — you will need to unhide it with the Command-P again. This key command is modifiable.

You can use Command-I to toggle the "Enable PI Window" state. This key command is modifiable.

In the graph, you can use:
1) the A key to select all the plug-ins.
2) the D key to deselect all the plug-ins.
3) the 'T' key to toggle the bypass state of the selected plug-ins.
4) the 'B' key to toggle the bypass state of the selected plug-ins uniformly (e.g. after 'B' the selected plugs will all be bypassed or enabled, based upon the original state of the first instantiated plug that is selected).

The new Comp, EQ and Limiter plugs have different parameter ranges and capabilities compared to the MIOStrip modules.

In the graph, you can now option click a plug-in tile to toggle the plug-in's bypass state.

The arrow keys will now move the selected plug-in nodes in the graph. Hold down the option key for fine nudging.

Contacting Metric Halo

If you need to contact us:
Metric Halo
5 Donovan Dr.
Hopewell Junction, NY 12533

support@mhlabs.com

Please send bug reports to:
support@mhlabs.com
bj@mhlabs.com
Subject: MIOBUGX+DSP