VSTHost Slavery Suite

Slave Mode Add-Ons for VSTHost

Version 1.09

Download Information

The latest versions of VSTHost and the VSTHost Slavery Suite can be found at
http://www.hermannseib.com/english/vsthost.htm
or
http://www.hermannseib.com/vsthost.htm
for the German-speaking minority.

Contact Information

The author can be reached per email at office@hermannseib.com

Copyright Information

Table of Contents

Introduction

What is VSTHost?

Hmmm. If you really don't know this product, I'd strongly suggest that you take a look at my web site; the location for VSTHost is given on page 2 of this document. ☺

Now, you *do* know what VSTHost is? Good… so let's continue.

What is the Slavery Suite?

One fine day, I came across the following post in the KVR-VST forum (http://www.kvraudio.com/forum/viewtopic.php?t=99622): "**Is there a way to route the output of vsthost to Kristal Audio engine? And without latency?**"

Well, thought I… not a bad idea, so I implemented a *Slave Mode* for VSTHost, and coined the name "Slavery Suite" for it. The rest of this document deals with the components of the Slavery Suite and how to use them.

What does it cost?

Aaaah, this is the point where money comes into play; I was never good at that ☺… basically, it's free. The download version is not restricted in any way, doesn't even show a nag screen. Why encourage pirates to tinker with it?

In theory, it's "donationware", which means that you can download and use it; if you find it useful, it would be nice to register by sending a little bit of money to my PayPal account. There's a "Donate" button on VSTHost's web site for that. I don't insist on it, but it would be nice if you honored the countless hours I've invested into making this thing usable by donating a bit to the further development of VSTHost.

Installation

The VSTHost Slavery Suite consists of some parts; these are detailed below.

The first part, naturally, is **VSTHost**; it acts as the *Slave*. Now, what's a slave without a *Master*? Not much… so, to allow the use of VSTHost as a slave, I've written the following VST PlugIns that can be embedded in other VST hosting programs:

Legree	is a VST Effect PlugIn that talks to VSTHost via Shared Memory
Legree VSTi	is a VST Instrument PlugIn that talks to VSTHost via Shared Memory
Haley	is a VST Effect PlugIn that talks to VSTHost via TCP
Haley VSTi	is a VST Instrument PlugIn that talks to VSTHost via TCP

VSTHost

The VSTHost installation is detailed in the main VSTHost documentation; if you haven't installed it already, please refer to that.

Configuration

Shared Memory

As described in VSTHost's main documentation, you have to start VSTHost with the command line parameter **/slave** to make it come up in Slave Mode. The easiest way would probably be to create a link to VSTHost.exe either on the desktop, or somewhere in the start menu, open its property dialog, and append " /slave" to the command line given there.

Since V1.34, you can use up to 18 concurrent VSTHost slaves. This has been accomplished by extending the **/slave** command line parameter a bit; you can append the slave *number* now. So, the complete syntax of the **/slave** parameter is now:

```
vsthost /slave[:nn]
```

where *nn* is a number in range -1 (which is the default value) to 16.

Your mileage may vary, of course – depending on your computer and the complexity of the patches set up in the VSTHost Slaves, even 2 concurrent slaves might be too much to work reliably.

Each time you start VSTHost in Slave Mode, it comes up showing the following dialog:

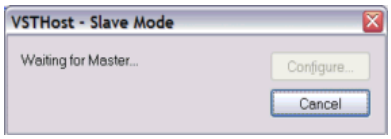

Figure 1: VSTHost waiting for Master connection

… and that's it. All configuration things are done in the Master.

Network-based

Starting in V1.52, VSTHost also contains a network slave mode. To accomplish this, the /slave parameter has been expanded yet another bit – you have to use **/slave:net** for this mode.

In this mode, VSTHost tries to establish a *network service* at a configurable port. In recent Windows version, this automatically and inevitably gives you an instant yelp from Windows which tells you that VSTHost does so, and whether you really want to allow it. Weeeellll... if you want to use this feature, you had better tell Windows it's OK :-)

The default port by VSTHost is 18515 – an unprivileged port which should not cause any trouble. You can change this, of course; the dialog mentioned above contains a usable “Configure” button in network slave mode, which opens a little dialog:

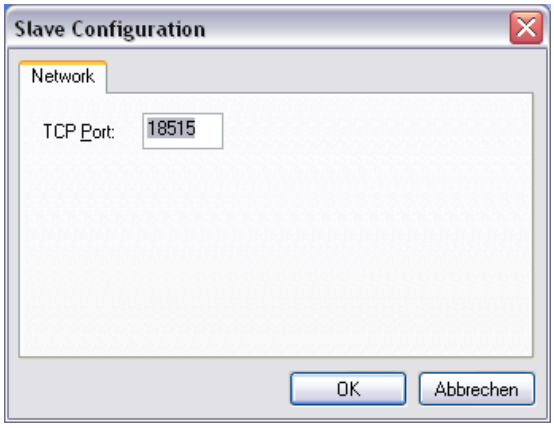

Figure 2: Network-based configuration

Here, you can configure the port VSTHost should use for the service.

Legree Effect, Legree VSTi

Both PlugIns come in a .zip file. Simply unpack all the .dll files in this .zip file into your VST PlugIn folder, and that’s it. Legree is too simple (or too intelligently written? You decide ☺) to need an elaborate installation procedure.

Since V1.07, Legree comes in both 32-bit and 64-bit format; if you're using a 32-bit host, take the DLLs from the x86\ subdirectory, and for a 64-bit host, take the DLLs from the x64\ subdirectory.

Note: this has nothing to do with the “bitness” of the slaved VSTHost – for Legree, it doesn't matter whether the slave is a 64-bit or 32-bit program. This way, VSTHost can be used as a cheap bridging program – you can use it to load 64-bit PlugIns when the real host can't do that, or 32-bit PlugIns if the host can only load 64-bit PlugIns. All on a 64-bit operating system only, of course – a 32-bit Windows won't let you run 64-bit programs.

Depending on the VST Host of your choice that should act as the Master, you might have to let it re-scan for modified PlugIns after you installed Legree.dll and LegreeI.dll; most hosts, however, will do that automatically when they are started the next time.

Also depending on the VST host used, each of the PlugIns can have advantages (or drawbacks, depending on your point of view); not all hosts, for example, send MIDI data to VST Effects, even if they tell the host that they expect them… and not all hosts send audio input data to VSTis, even if the VST Instrument tells the host that it expects them. You got to find out which of the two works best with your host.

Configuration

Both PlugIns come up showing a little dialog:

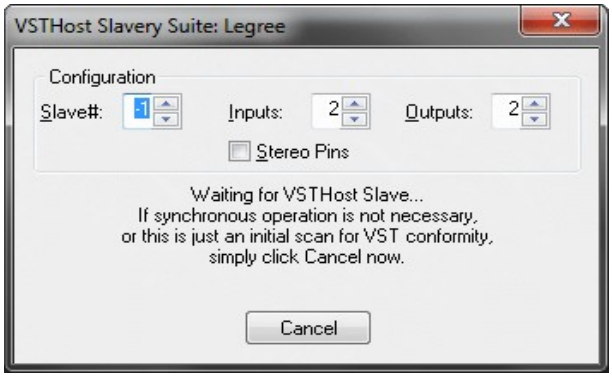

Figure 3: Legree Initialization Dialog

This is done while the PlugIn is initialized; at this point, it has not been fully loaded. This is the only time where you can reliably define the PlugIns' I/O configuration (while some hosts allow on-the-fly reconfiguration, I haven't added this rarely used capability to Legree(I)).

If you see this dialog while the host is just scanning for new PlugIns, simply configure the inputs and outputs, and then press Cancel; in all cases I tried, this is enough to make the host happy.

This dialog only appears *as long as no VSTHost Slave with the given ID is open*; as soon as Legree(I) finds out that there's a slave available, it terminates the dialog, initializes the slave and busily goes to work. This can have an unwanted side-effect during the host's PlugIn scan phase – it has to open all unknown .dll files in its VST PlugIn directory to examine them for VST conformity. Most hosts *load* the PlugIn completely to find out its name, type, and other details… and then throw it away again. Since Legree(I) closes the slave host when it exits, a loaded VSTHost Slave is closed as soon as the host examines the new PlugIns. Since V1.02, this effect can be avoided by setting the "Terminate Slave" parameter to **Off** (see below).

Let's assume that you can see the dialog, i.e., that no VSTHost Slave is loaded. In this case, you can enter the number of Input and Output Channels that Legree(I), and the VSTHost Slave with it, should provide. You can configure 0..16 input channels and 1..16 output channels. If you're providing Stereo inputs and outputs, the "Stereo Pins" check box should be checked; if you're providing Mono inputs and outputs, leave it unchecked. Changes are immediately saved into the registry and used as default values when Legree(I) comes up the next time. Legree and LegreeI maintain different value sets, since it's quite possible that you want to set different values for the both cases. After having (re-)configured the channels, simply load the VSTHost Slave to let Legree(I) continue.

Note: while the PlugIn is loaded, you can turn off the VSTHost Slave at any time you want; in this case, Legree(I) will simply do nothing. When a new VSTHost Slave comes up, Legree initializes it and continues.

Haley Effect, Haley VSTi

Both PlugIns come in a .zip file. Simply unpack all the .dll files in this .zip file into your VST PlugIn folder, and that's it. Haley is too simple (or too intelligently written? You decide ☺) to need an elaborate installation procedure.

Haley comes in both 32-bit and 64-bit format; if you're using a 32-bit host, take the DLLs from the x86\ subdirectory, and for a 64-bit host, take the DLLs from the x64\ subdirectory.

Note: this has nothing to do with the "bitness" of the slaved VSTHost – for Haley, it doesn't matter whether the slave is a 64-bit or 32-bit program. This way, VSTHost can be used as a cheap bridging

program – you can use it to load 64-bit PlugIns when the real host can't do that, or 32-bit PlugIns if the host can only load 64-bit PlugIns. In contrast to Legree above, Haley even allows you to run 64-bit PlugIns in a 32-bit host – if the slaved VSTHost runs on a 64-bit Windows on another machine in your network.

Depending on the VST Host of your choice that should act as the Master, you might have to let it re-scan for modified PlugIns after you installed Haley.dll and HaleyI.dll; most hosts, however, will do that automatically when they are started the next time.

Also depending on the VST host used, each of the PlugIns can have advantages (or drawbacks, depending on your point of view); not all hosts, for example, send MIDI data to VST Effects, even if they tell the host that they expect them… and not all hosts send audio input data to VSTis, even if the VST Instrument tells the host that it expects them. You got to find out which of the two works best with your host.

Configuration

Both PlugIns come up showing a little dialog:

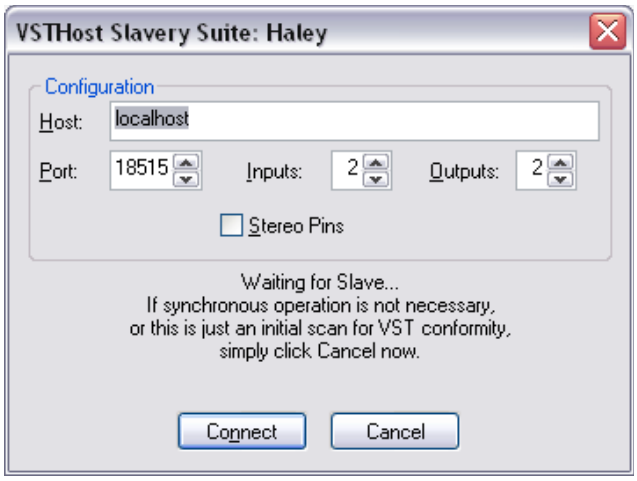

Figure 4: Haley Initialization Dialog

This is done while the PlugIn is initialized; at this point, it has not been fully loaded. This is the only time where you can reliably define the PlugIns' I/O configuration (while some hosts allow on-the-fly reconfiguration, I haven't added this rarely used capability to Haley(I)).

If you see this dialog while the host is just scanning for new PlugIns, simply configure the inputs and outputs, and then press Cancel; in all cases I tried, this is enough to make the host happy.

In contrast to Legree(I), this dialog is not automatically terminated as soon as Haley(I) finds the configured VSTHost slave – searching for the slave can take quite some time, so it's only done if you press the "Connect" button. So, to really make the connection at this point, you have to:

- set up the slaved VSTHost on the target machine
- configure the used host's name or IP address and port in this dialog (the default value "localhost" assumes that it's running on the same machine – actually a bad idea, Legree(I) is better for that)
- press "Connect" here

You can enter the number of Input and Output Channels that Haley(I), and the VSTHost Slave with it, should provide. You can configure 0..16 input channels and 1..16 output channels. If you're providing Stereo inputs and outputs, the "Stereo Pins" check box should be checked; if you're providing Mono inputs and outputs, leave it unchecked. Changes are immediately saved into the registry and used as default values when Haley(I) comes up the next time. Haley and HaleyI maintain different value sets,

since it’s quite possible that you want to set different values for the both cases. After having (re-)configured the channels, simply load the VSTHost Slave to let Legree(I) continue.

Note: while the PlugIn is loaded, you can turn off the VSTHost Slave at any time you want; in this case, Haley(I) will try a reconnection every 2 seconds. In the current version, this is the weakest point, since the connection attempt can take quite a while – and this can cause audio st-t-u-t-tt-e-ring. I'll work on this. When a new VSTHost Slave comes up at the configured address, Haley(I) initializes it and continues.

Operation

VSTHost

VSTHost maintains a completely separate set of parameters for Slave Mode; if you keep your VSTHost Programs in the Internal Bank, i.e., in the registry, you will have to either re-create them, or save them to a VSTHost Bank file in normal mode, and then use this bank file in Slave Mode. Please look at the main VSTHost documentation if some of the terms used in this paragraph sound like gibberish to you. ☺

Since most of the operational parameters of the VSTHost Slave are controlled by the Master program, quite a lot of VSTHost's menu and toolbar entries are grayed out. This is intentional.

MIDI Devices

Since not all hosts provide MIDI data to the loaded VST(i)s, the VSTHost Slave allows to open as many additional MIDI devices as possible. On the various configuration dialog pages, you might encounter strange MIDI device names: "Slave In 1" and "Slave Out 1". These are simulated MIDI devices. "Slave In 1" transports all MIDI data that are sent from the master to Legree(I) / Haley(I) into the VSTHost Slave, whereas "Slave Out 1" transports all MIDI data that are sent back from the VSTHost Slave into Legree(I) / Haley(I), which passes them on to the Master VST host.

Audio Channels

Wherever these can be used / configured in the VSTHost Slave, you'll meet channel names like "Master In 1", or "Master Out 1". Mental note to self: MIDI uses "Slave", audio uses "Master" notation… make same in both cases! These are simulated audio channels. "Master In *x*" transports all audio data that are sent from the master to Legree(I) / Haley(I) into the VSTHost Slave, whereas "Master Out *x*" transports all audio data that are sent generated in the VSTHost Slave into Legree(I) / Haley(I), which passes them on to the Master VST host. The number of In and Out channels is defined in Legree(I) / Haley(I) when you start it up (see above).

Note: the Master/Slave communication between Legree(I) / Haley(I) and VSTHost is rather simple; only *one* master can be active at a time, and only *one* slave can be active at a time for each of the possible slave addresses.

Legree Effect, Legree VSTi

Once loaded and configured, these two PlugIns should simply work under the control of the Master VST host. In case of the Kristal Audio Engine, make sure that you do *not* configure more than **2** input or output channels for Legree – Kristal (V1) doesn't load it in this case.

Parameters

Legree(I) is so simple that it has only very few parameters. This doesn't justify a complete editor, so it relies on the host to display/modify the parameters. Here, for example, is a Cubase SL V1 rendition:

Figure 5: Legree Parameter Display

If the host supports asynchronous operation, Legree(I) allows to toggle the **Synchronous** parameter between **On** and **Off**. When it is on, and if the host doesn't support anything else, Legree doesn't report anything back to the Master host until it has received the processed audio data back from the VSTHost Slave. When it is off, it immediately returns to the Master host as soon as it has sent the data

to the VSTHost slave; as soon as the results come back, the Master host is informed that it can fetch them.

The **Slave Host** parameter can be used to change the slave host number, just like on the configuration dialog, just with one little difference: if there is an active connection to the previously configured VSTHost slave, this slave is *terminated*, since VSTHost in Slave mode doesn't allow reconnections yet. The **Slave Host** parameter has been added to make it easier to define and save complex setups with more than one Legree(I) master.

The **Config Dialog** parameter determines whether Legree(I) should display the initial configuration dialog. While this dialog can be important to configure the number of inputs and outputs of the PlugIn, it can become a real stumbling block when it comes to quick setup changes in the host, so I decided to make its appearance configurable.

The **Terminate Slave** parameter determines whether Legree(I) closes the slave host when it is unloaded; in versions before V1.02, this was always the case. Sometimes, it is more convenient to let the slave host open in the background, so you can turn this setting off. This requires VSTHost V1.35 or later; earlier versions will close in any case as soon as Legree(I) is unloaded.

Note: the asynchronous mode doesn't make much sense on a single-processor system; on a hyperthreading and/or multiprocessor machine, however, it can give a little performance boost.

Haley Effect, Haley VSTi

Once loaded and configured, these two PlugIns should simply work under the control of the Master VST host. In case of the Kristal Audio Engine, make sure that you do *not* configure more than **2** input or output channels for Legree – Kristal (V1) doesn't load it in this case.

Parameters

Haley(I) is so simple that it has only very few parameters. This doesn't justify a complete editor, so it relies on the host to display/modify the parameters. Here, for example, is a VSTHost rendition:

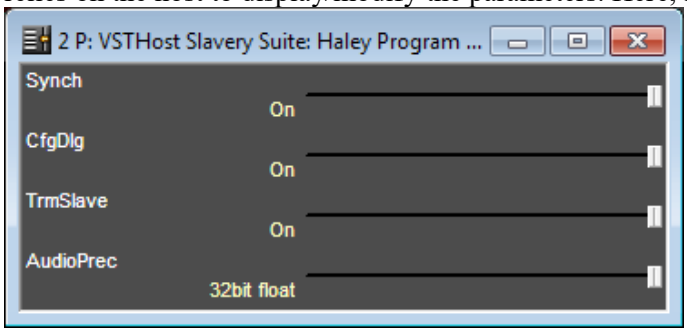

Figure 6: Haley Parameter Display

If the host supports asynchronous operation, Haley(I) allows to toggle the **Synch** parameter between **On** and **Off**. When it is on, and if the host doesn't support anything else, Haley doesn't report anything back to the Master host until it has received the processed audio data back from the VSTHost Slave. When it is off, it immediately returns to the Master host as soon as it has sent the data to the VSTHost slave; as soon as the results come back, the Master host is informed that it can fetch them.

Note: the asynchronous mode doesn't make much sense on a single-processor system; on a hyperthreading and/or multiprocessor machine, however, it can give a little performance boost.

Note 2: in the current version, this doesn't work correctly – better don't set it to Off :-)

The **CfgDlg** parameter determines whether Haley(I) should display the initial configuration dialog. While this dialog can be important to configure the number of inputs and outputs of the PlugIn, it can

become a real stumbling block when it comes to quick setup changes in the host, so I decided to make its appearance configurable.

The **Terminate Slave** parameter determines whether Haley(I) closes the slave host when it is unloaded; in versions before V1.02, this was always the case. Sometimes, it is more convenient to let the slave host open in the background, so you can turn this setting off.

The **Audio Precision** parameter allows to tweak Haley(I)'s network performance a bit. On a gigabit local network, you should experience no problems with the standard 32-bit floating point values sent to and from the slave; on slower networks, you might. In this case, you can reduce the audio quality in 3 steps that reduce the generated traffic by roughly 33% , 50%, and 75% (although the lowest one, which transmits samples in 8bit format, isn't really usable – for short demos, it might be sufficient, however).

I hope I didn’t forget anything… if I did, please let me know.

Have fun using the VSTHost Slavery Suite!

Hermann Seib
Vienna, July 9th, 2018