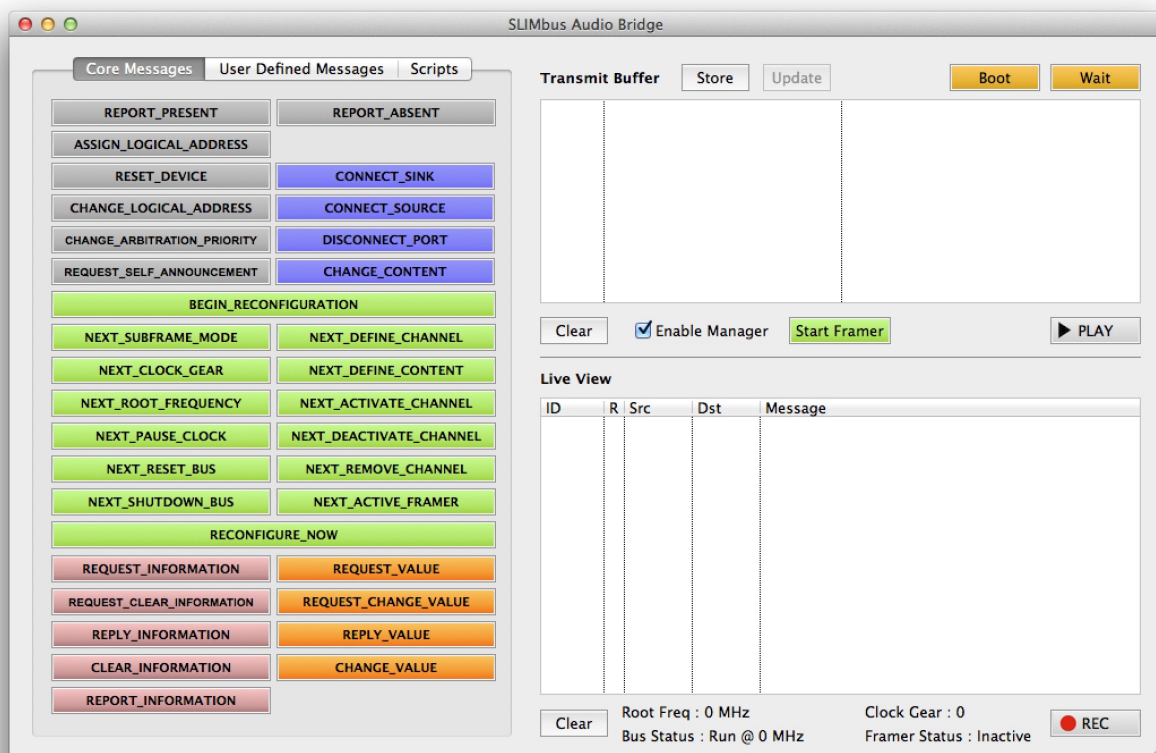




AudioBridge Software User Manual



LnK
44, rue des Combattants
B-4624 Romsée
Belgium
www.lnk-tools.com
info@lnk-tools.com

Table of Content

1. Introduction	3
2. System setup	3
3. Script Generation	5
3.1. Adding Core Messages	5
3.2. Adding User Messages	5
3.3. Special script commands	7
3.4. Storing a Script	8
3.5. Editing a Script	9
3.6. Deleting a Script	9
3.7. Script Library files	9
4. Tools	10
4.1. User Message Library Editor	10
4.2. Device Register Configuration	11
4.3. Turning the captured messages into a script	12
4.4. Script Automator	13
4.5. Component Map	15
4.6. Value Element Map	16
4.7. Information Element Decoding	17
4.8. Search Engine	18
4.9. Data Channel View (12 ports bridge model)	19
4.10. Manual Port Configuration (12 Ports Bridge Model)	20
4.11. Tone Generator setup (12 Ports Bridge Model)	21
4.12. Signal Analysis (12 Ports Bridge Model)	22
5. Hardware Operation	27
5.1. Message Sniffer Activation	27
5.2. Framer Activation	28
5.3. Manager Activation	28
5.4. Bridge Control Panel	28
5.5. Playing a Script	32

1. Introduction

AudioBridge is the control software of the hardware SLIMbus Audio Bridge.

The hardware unit is autonomous as a SLIMbus component. However, when connected to the AudioBridge software, the hardware unit can transmit any kind of messages on SLIMbus and can capture the complete message traffic going over SLIMbus.

The software also allows for remote control of the hardware through a DLL.

The graphical user interface (GUI) is structured in 3 main areas:

- The Transmit message buffer, used to build scripts and edit message parameters
- The Live View panel that shows all the messages going over SLIMbus. For convenience, the message are decoded in such a way that they appear with their parameters as described in the SLIMbus specification (see Section 11).
- The Message and Script panel. It has 3 tabs. The first one is dedicated to the SLIMbus core messages. The second one is dedicated to the user defined messages. The third one shows the content of the script library in use.

For the exact meaning of the SLIMbus Core message parameters, please refer to the SLIMbus Specification, Section 11.

Some extra tools are provided as:

- User message library editor
- Script automator
- Component map monitor
- Value Element Map monitor
- Bridge hardware control panel

2. System setup

The software runs on Windows, Mac OSX and Linux.

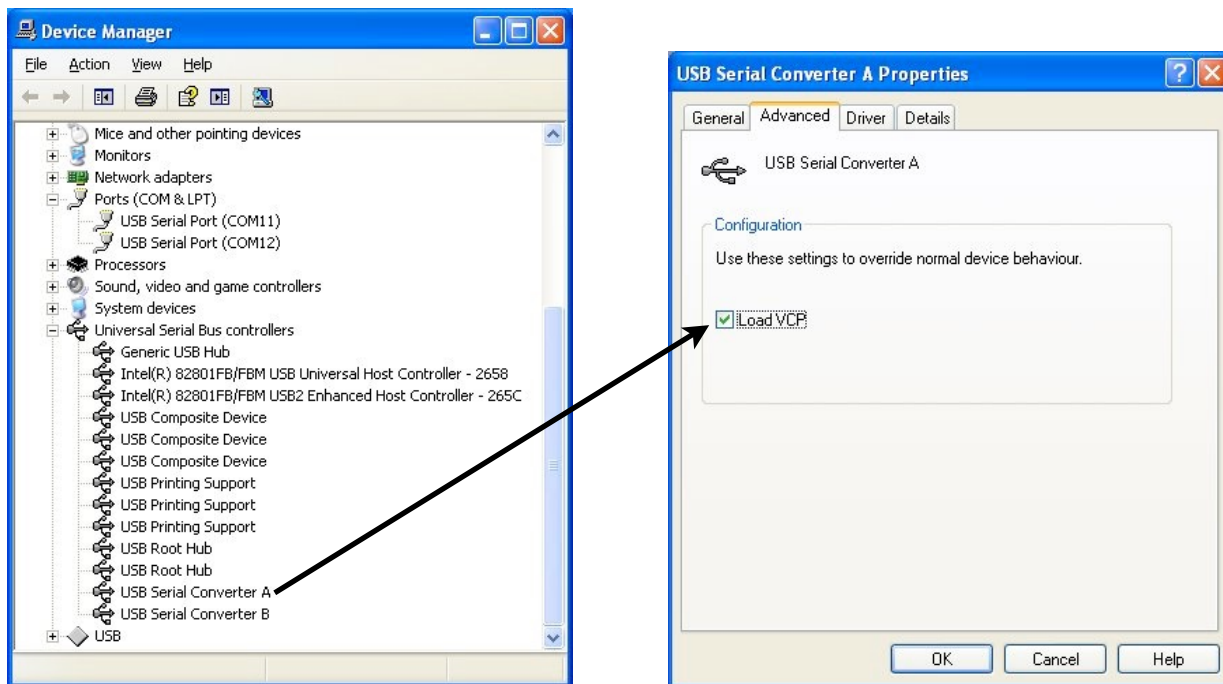
The following steps will describe how to setup the system on a Windows (XP, Vista or 7, both 32bits and 64bits OS).

Step 1: Copy the directory “AudioBridge” to the desired location on the PC HDD. There is no need to run an installation procedure.

Step 2: Install the USB driver of the bridge HW. Make sure that the bridge is not connected yet to the PC. Go to the “Bridge USB Driver” directory and execute the driver installer “**CDM20824_Setup.exe**” or similar.

Step 3: Power the bridge hardware and connect it to a free PC USB port. The proper drivers will automatically be activated.

Step 4: Driver setup. The bridge hardware uses the Virtual COM Port drivers (VCP) of Windows. They must be activated to be able to use the bridge hardware. Go to the Control Panel, open the Device Manager and look at the USB peripherals.



There should be “USB Serial Converter A” and “USB Serial Converter B”. Double click on the device. In the properties panel, go to Advanced, tick the “Load VCP” check box and click on the “**OK**” button. The Converter A is used by the 8 ports bridge. The Converter B is used by the 2 ports bridge. However, the VCP drivers can be enabled by default on both Converter A and B.

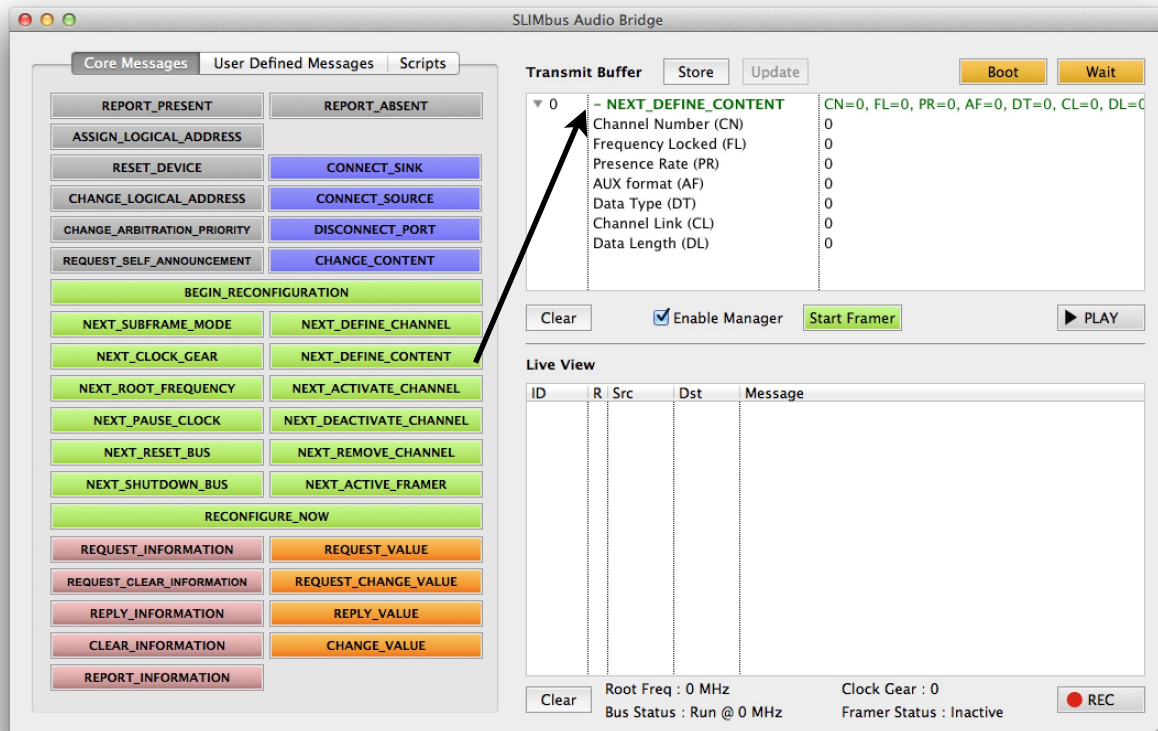
Step 5: Unplug and replug the bridge hardware. This is necessary to force the VCP drivers to be loaded by the OS. In the Device Manager, under the “Ports (COM & LPT)”, 2 serial ports shall appear. The installation process is finished.

When launched, the software will automatically search for the bridge. Once it is found, the bridge is reset and the “**PLAY**”, “**REC**” and “**Start Framer**” buttons are displayed on the main window.

3. Script Generation

The user interface is very simple. The software makes intensive use of contextual menu to drive the user choice whenever possible. Activate the contextual menu by right-clicking in an editable cell. Mac OS users will have to either right-click or press the CTRL key while clicking.

3.1. Adding Core Messages



Click on the button having the label representing the message you want to insert in the Transmit Message Buffer. It will appear in the top right panel (Transmit Buffer). Add as many messages as required. Note that the transmit buffer has a limit of maximum 10,000 messages.

The messages can be expanded to access and modify all the parameters. To edit a parameter, expand the message and click on the cell containing the parameter value. Then, type the new value. A message can be moved up and down in the list by clicking on it and dragging it up or down to the desired location. Note that the drag and drop functionality will only work if all the messages are collapsed.

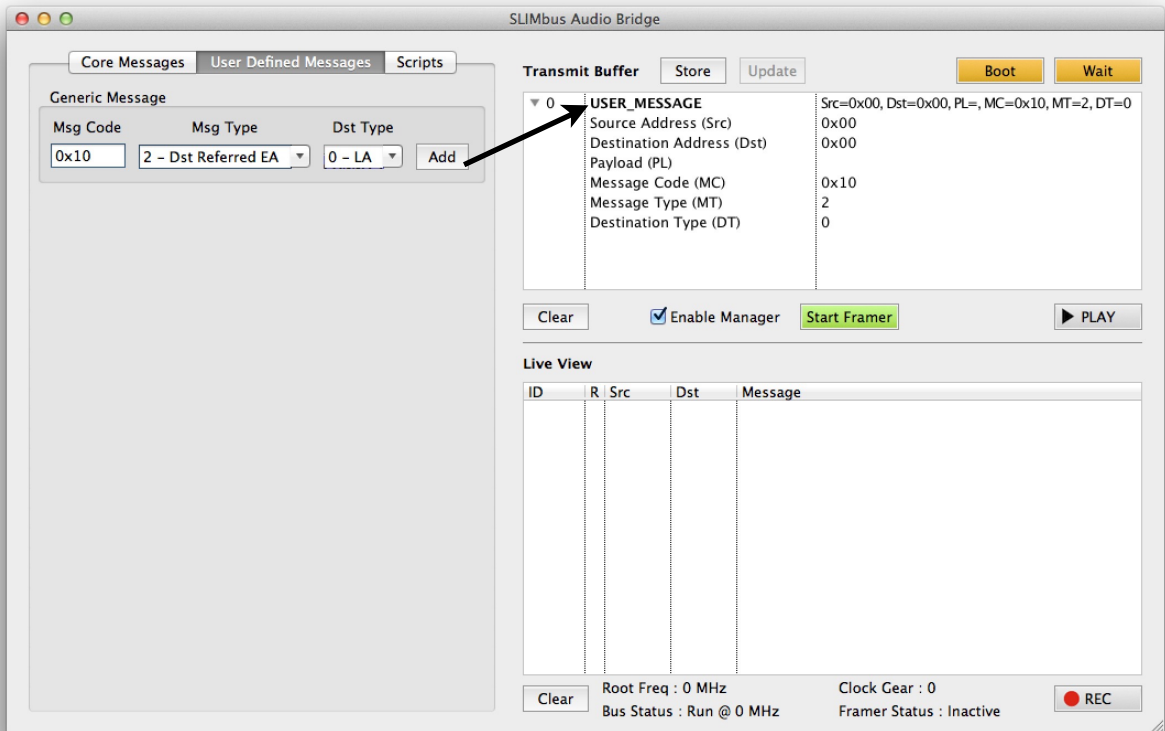
Values are entered as decimal value, hex value (0xYZ format) or obtained from the contextual menus (mouse right click or CTRL+mouse click).

3.2. Adding User Messages

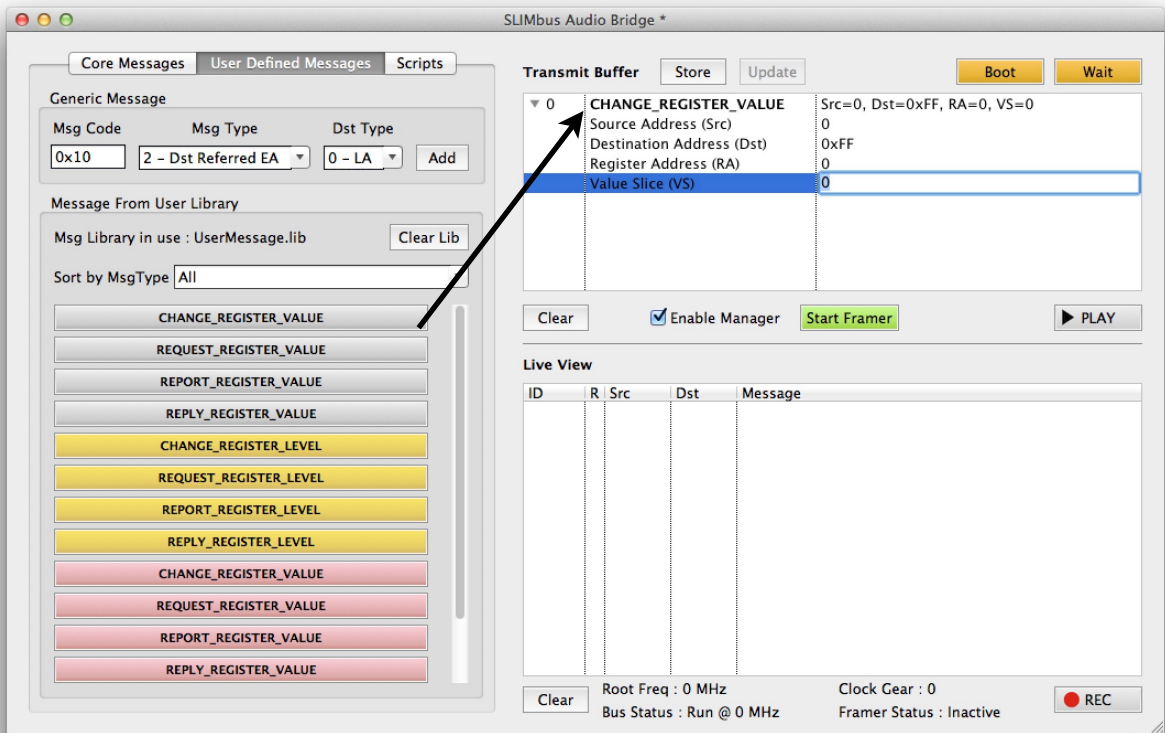
The SLIMbus specification allows the use of user defined messages. The software allows for 2 input methods:

- Specify the various fields value of a message, including the payload. This is straight forward but is probably not the most human readable way of proceeding.
- Define with the message library editor a collection of messages for which a univocal relationship exists between the raw message fields and a complete description of a

message (message name, parameter names and bit structure of the parameters). To launch the message library editor, go to the menu **“Tools/Message Library Editor”**. A new window will appear.



When load a user message library, a collection of new buttons will appear. Click on any of the new buttons to insert a message in the transmit buffer.



3.3. Special script commands

There are two specific script commands: **WAIT** and **BOOT**.

WAIT pauses the script execution for a given amount of time, expressed in ms. Note that this value is the minimum amount of time the script play engine will hold message transmission. Due to handshaking (if enabled), USB traffic and OS maintenance activities, the effective delay inserted between two transmissions can be much longer, but never shorter.

When having multiple reconfiguration sequence in a row, it is important that only of them happens in a given superframe. A good practice consists in adding a **WAIT** command after each reconfiguration sequence. The wait time shall be at least equal to the duration of a superframe. At 24.576 MHz, a superframe lasts 0.25 ms. At 6.144 MHz, it lasts 1 ms.

```
BEGIN_RECONFIGURATION
- NEXT_SUBFRAME_MODE (SM=25)
RECONFIGURE_NOW
WAIT (ms=5)
BEGIN_RECONFIGURATION
- NEXT_CLOCK_GEAR (CG=7)
RECONFIGURE_NOW
```

BOOT allows a script to start the framer. It also inserts a delay after the boot sequence to allow the device present on the bus to REPORT_PRESENT without interference of the script messages.

Depending on the system setup, **BOOT** will have different effects. When the check box “Synchronize on external framer” is ticked (in the **Scripts** tab), **BOOT** will result in the transmission of the following messages:

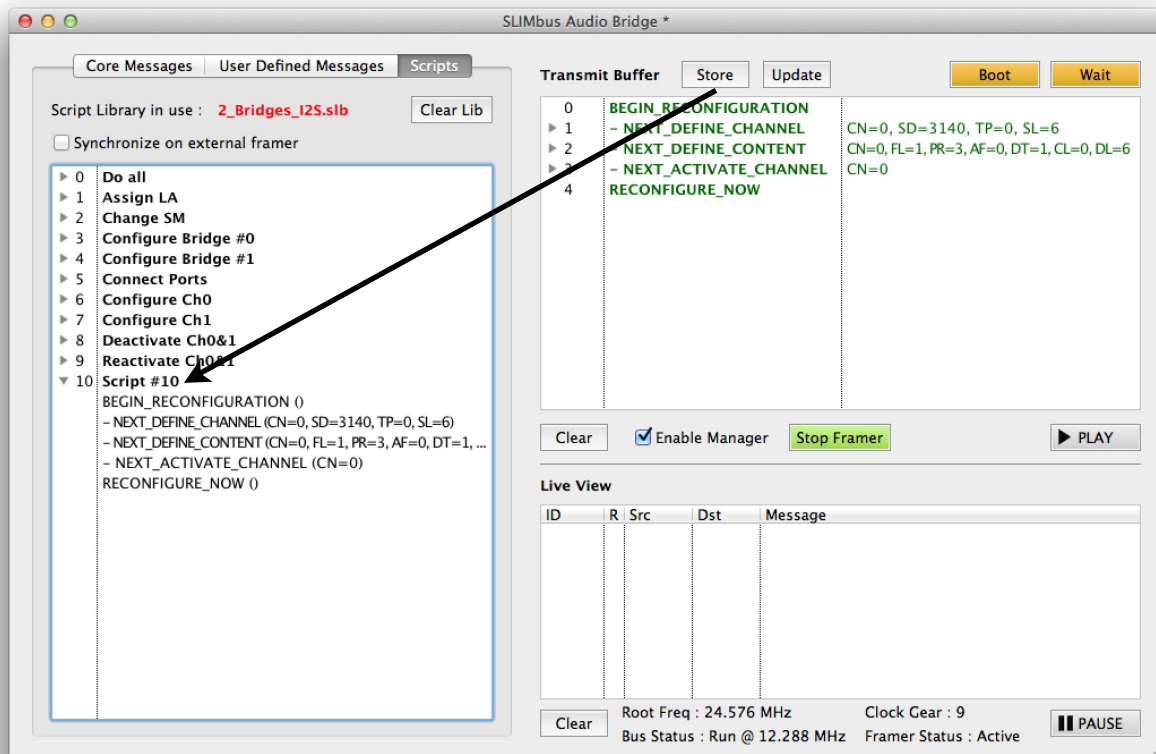
```
BEGIN_RECONFIGURATION
- NEXT_RESET_BUS
RECONFIGURE_NOW
WAIT (ms=x)
```

This message sequence will reset the external framer and trigger a bus boot.

When the check box “Synchronize on external framer” is not ticked, **BOOT** will stop and restart the bridge framer and add a delay before continuing the transmission of the messages. In this case, there is not any specific events captured by the LiveView.

3.4. Storing a Script

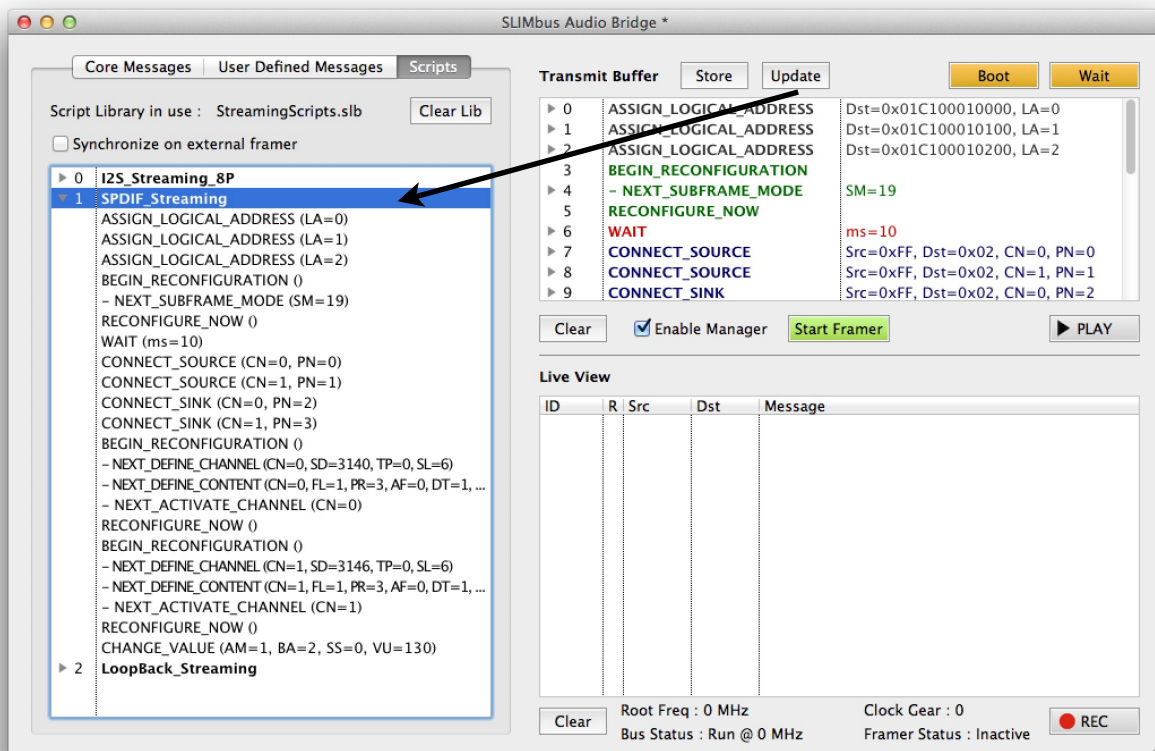
When the script is ready in the transmit buffer, it can be played by pressing the “**PLAY**” button. After being played, the content of the transmit buffer is cleared. Therefore, it might be more interesting to store the script if it has to be used more than once. To generate a new script entry in a script library, just press the “**Store**” button. It will appear in the Script panel as “Script #n”, with n the script number.



To rename a script in the library, right-click on the script name, then click on the “Rename” item of the contextual menu. The name will be editable. Just give it the desired name. The scripts can be expanded (viewable content) or collapsed. Once many scripts are in the list, it is a script library.

3.5. Editing a Script

It is always possible to edit a script stored in a library. Drag the desired script from the script panel to the transmit buffer. The script content will appear in the transmit buffer. Parameters can be modified, messages can be added, deleted or moved. Once the edition is finished, the script can be transferred back to the scrip library by clicking the “**Update**” button. That button is only available when a script is selected in the library panel. The edited script can also become a new script in the library by clicking the “**Store**” button.



3.6. Deleting a Script

Select the desired script by clicking on it. Hit the **DEL** key and the script will be removed from the library.

3.7. Script Library files

Changes in Script Library files will be permanent only if the file is saved after modifications.

Script Library files can be loaded, saved and saved as. The file extension is “**.slb**”. When in the Script Library panel, a right click will bring a contextual menu. An item of that menu has the function to append to an already loaded script library the content of another script Library. This is a convenient method to build new libraries from existing ones.

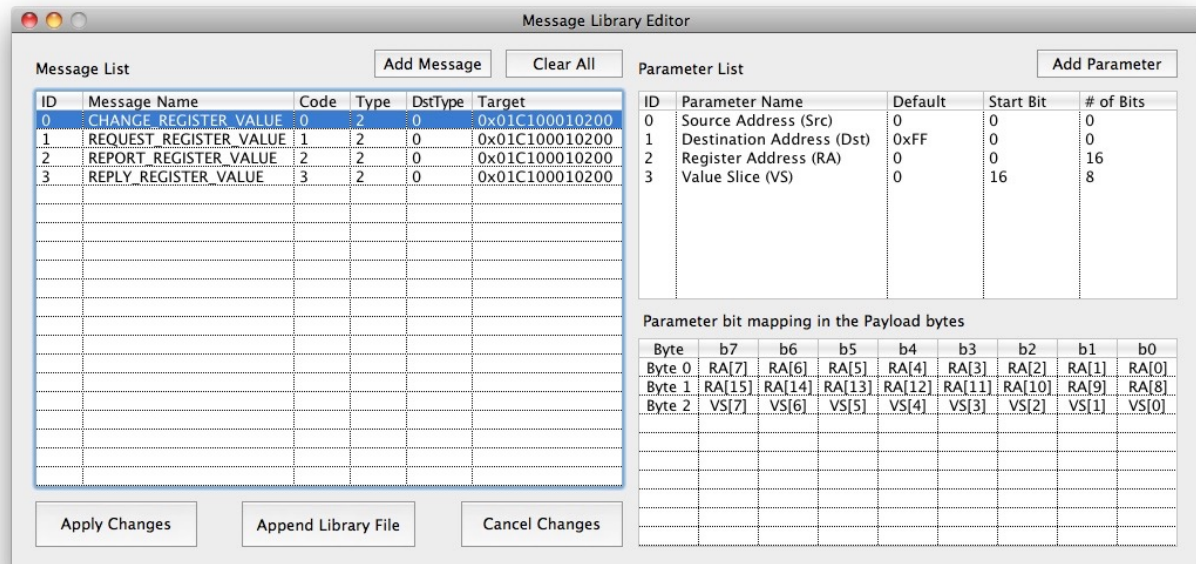
Library files can be directly dropped from the OS file browser to the Script Library panel.

4. Tools

4.1. User Message Library Editor

The SLIMbus specification allows the use of user defined messages. Obviously, as these messages are user defined, ScriptBuilder cannot have any “a priori” knowledge of them. Therefore, a message library editor has been added to the tool to allow the use of user defined messages.

Go to the menu “**Tools/Message Builder**”. A new window will appear.



The message list is on the left side. On the right, one can find the parameter list (associated to the selected message) and the bit organization of the parameters in the Message payload. The bit table has been made similar to the ones that can be found in the SLIMbus specification (see section 11).

Click on the button “**Add Message**” to add an empty message in the list. There are some message parameters that can be edited:

- **Name** is the message name. It will be used in the transmit message buffer and in the live view panel.
- **Code** is the Message Code (MC[6:0]). A value that ranges from 0 to 127.
- **Type** is the Message Type (MT[2:0]). Four possible types: Destination-Referred Class Specific, Destination-Referred Device Specific, Source-Referred Class Specific, Source-Referred Device Specific.
- **Destination Type** is the destination type (DT[1:0]). It indicates if the message destination is a Logical Address, an Enumeration Address or Broadcast.
- **Target** indicates to which device or classes the message applies.

Use the contextual menu to get appropriate values. The parameters are detailed in the SLIMbus specification, section 8.

Once the message is defined, add parameters by clicking on the “**Add Parameters**” button. A new parameter is added in the list. A maximum of 10 parameters per messages is allowed.

Edit the parameter by clicking on the column of the Parameter Name, Default Value, Start Bit and Field Length.

- **Start Bit** indicates where the least significant bit (LSb) of the parameter will be located in the payload.
- **Field length** indicates how many bits are used for the parameter. As for all the message payloads, the parameters are stored most significant bit first, least significant byte first.
- **Default Value** is the value that is used for the parameter when the message is inserted in the transmit message buffer.

To delete a message or a parameter, click on its **ID** and hit the “Del” key.

Note: For the messages that can be transmitted by any devices and that can have as a destination any devices, it is advised, for editing reasons, to add “**Source Address (Src)**” as the first parameter and “**Destination Address (Dst)**” as second parameter. Set the **Field Length** to 0 and the parameter will not be included in the Message payload.

If the Source and Destination addresses are not included in the message parameter list, the User will have to use the Message Fields Edit window to access the addresses.

In order to allow building consequent messages libraries, it is possible to merge existing libraries. Clicking on the “Append Library File” button will not delete the messages already present in the list. The existing messages and the one of the loaded library will all appear in the list. The list can then be saved as a new library (with a .mlb extension). The message Library can store a maximum of 500 messages.

4.2. Device Register Configuration

Most of the devices have a register map that allows for the configuration of the device functionalities. In case of large number of registers to configure, it may be unpractical to generate the script manually. ScriptBuilder has a special tool to turn a csv file into SLIMbus messages.

Go to menu “**File/Import Register Configuration**”. Select a CSV (ASCII comma separated value) file that has the following format.

- The comment lines are indicated by a semicolon at the start of the line.
- The parameters are sequenced “Function, Address, Data, Comment”.
- The functions are “**Reg_Write**”, “**Reg_Read**”, “**Delay_ms**” and “**Delay_us**”.

Reg_Write will generate a CHANGE_VALUE message.

Reg_Read will generate a REQUEST_VALUE message, that will trigger the transmission of a REPLY_VALUE message by the target Logical Address.

Delay_ms and **delay_us** will add some defined delays between the messages.

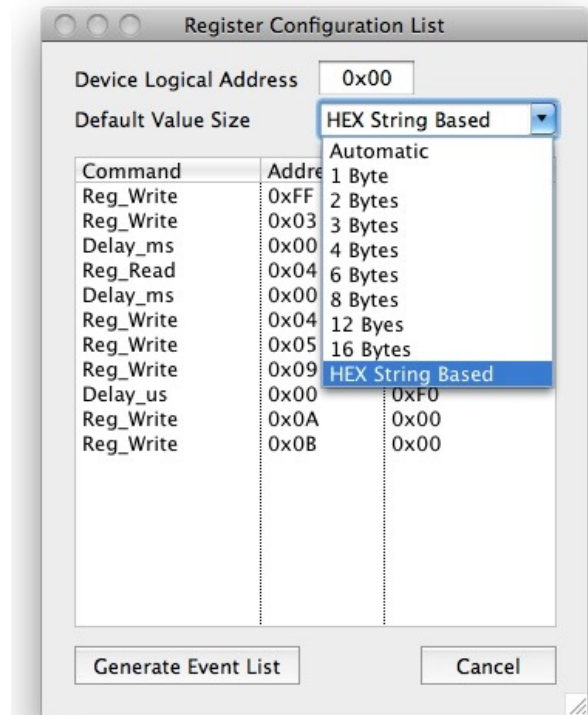
“**Address**” is the register address to be accessed.

“**Data**” is the value to be written in the register or the delay to be applied.

The delay function do not use the “Address” field (set to 0 by default). Values can be given both in HEX format (0x...) or in decimal format. The import tool is not case sensitive.

File format example:

```
; Mode = Sample,,,
; Function,Address,Data,Core
Reg_Write,0xFF,0x08,SLIMbus
Reg_Write,0x03,0x22,SLIMbus
Delay_ms,0x00,30000,SLIMbus
Reg_Read,0x04,0x00,SLIMbus
Delay_ms,0x00,0x03,SLIMbus
Reg_Write,0x04,0x8E,SLIMbus
Reg_Write,0x05,0xA0,SLIMbus
Reg_Write,0x09,0x00,SLIMbus
Delay_us,0x00,0xF0,SLIMbus
Reg_Write,0x0A,0x00,SLIMbus
Reg_Write,0x0B,0x00,SLIMbus
```



Be sure to enter the proper Logical Address of the device that needs register configuration.

The Value Size can also be specified. When set to “Automatic”, the Value Size is computed based on the actual given value. If this behavior is not desired, the Value Size can be forced to any of the allowed sizes. The “**HEX String Based**” size option will follow the size of the hexadecimal string. 0x0000 will get a Value Size of 2 bytes. 0x000000 will get a Value Size of 3 bytes. The Value Size is not based on the actual value anymore.

If the imported csv file is OK, just click on the “**Generate Event List**” button and all the new messages will get appended to the existing event list.

The delays are generated by inserting a WAIT function in between 2 messages. The minimum delay for the wait is 1 ms. Therefore, a delay_us function will always be turned into a WAIT(ms=1)

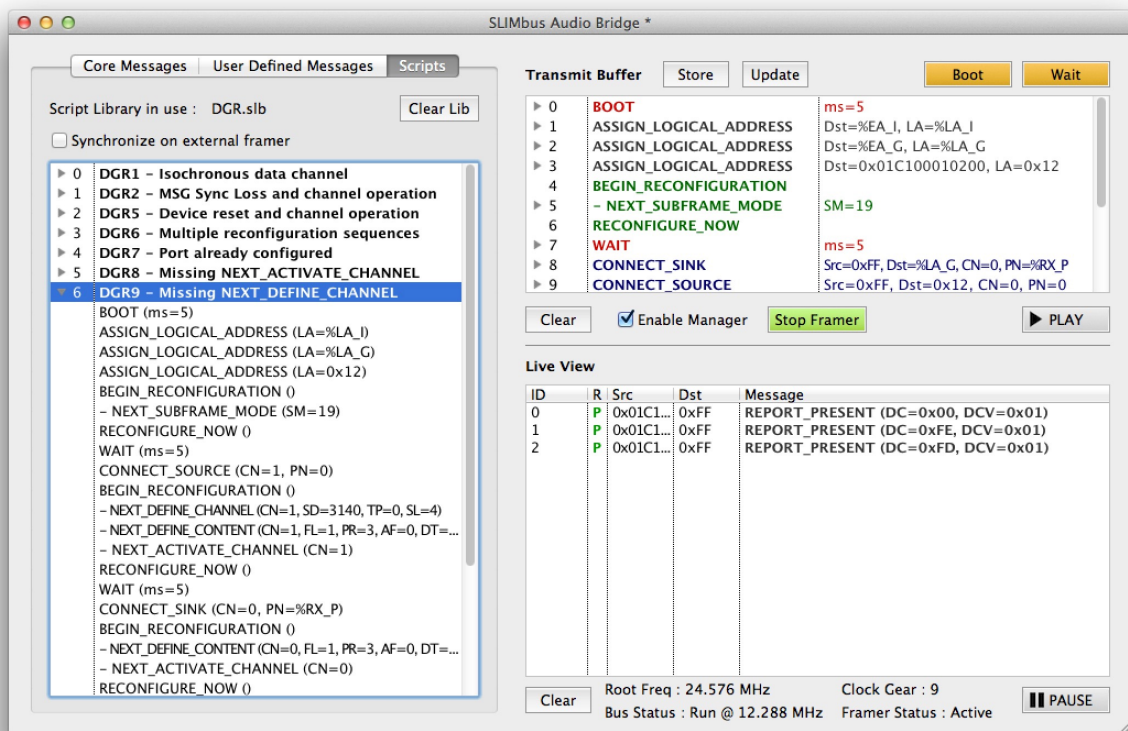
4.3. Turning the captured messages into a script

It is possible to use the capture messages to build a script. Select the messages of interest in the Live View panel and drag them over the transmit buffer panel. It is also possible to copy/paste the messages instead of dragging them.

The message parameters will be editable as if they were inserted by pressing one of the Core message buttons.

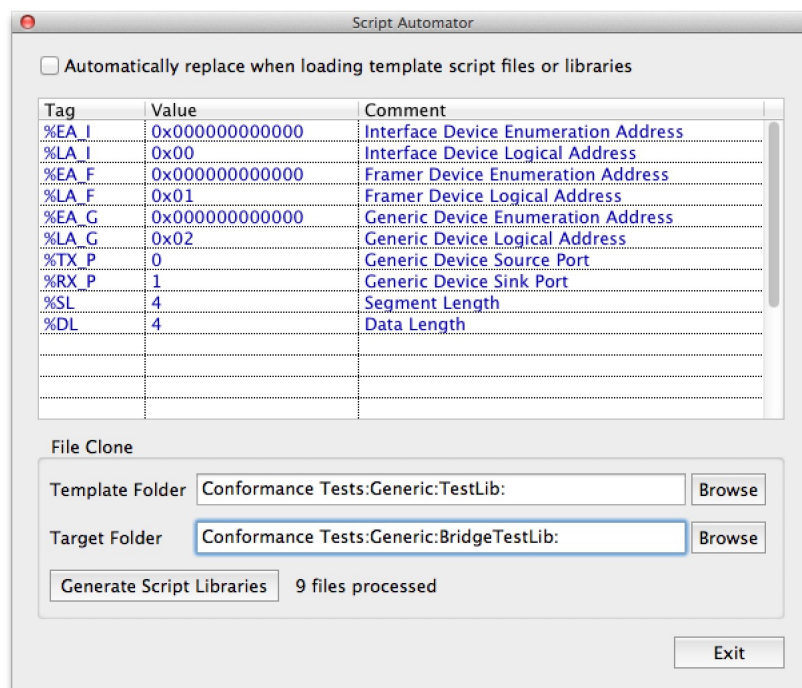
4.4. Script Automator

It's possible to write script templates with parameters having literal expressions instead of numerical values. The literal expression are starting by the character “%”.



In this example, %EA_I is used to represent the Interface Device Enumeration Address. %LA_I represents the Interface Device Logical Address.

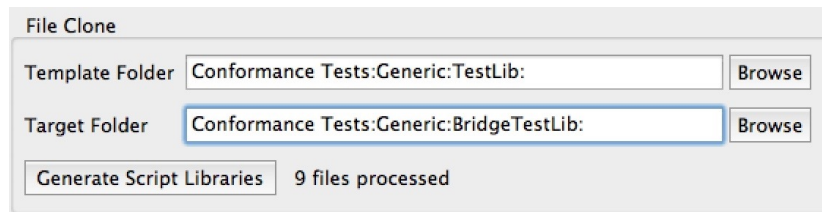
There are 10 predefined strings. As such, a script will not be transmitted properly. Go to the menu **Tools / Script Automator** to open the control window.



The string can get assigned a given value. The user can create up to 10 new strings on top of the existing ones (colored in dark blue).

When the check box is ticked, AudioBridge will automatically replace the strings by the define values when opening the script file or the script library.

It is also possible to use the “File Clone” function and create Component specific script files or libraries out of a template script folder.

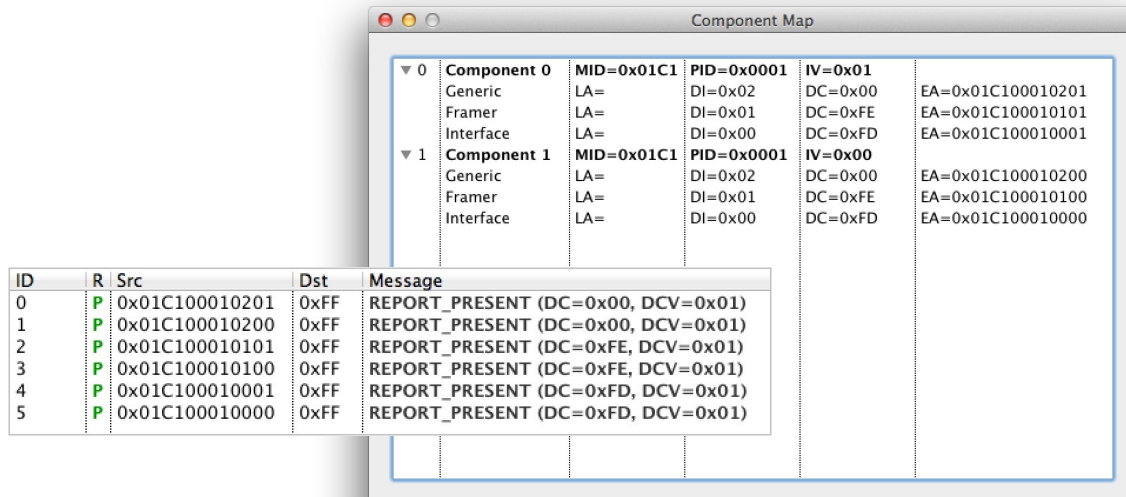


Once the template folder and the target folder are identified, click on the “Generate Script Libraries” button and wait for the file copy to be over. The target directory will contain the same files than the template directory, but with the literal strings replaced by the values given in the table.

Note that the file clone tool will use any text files present in the template folder: .lst (native ScriptBuilder format); .slb (Audio Bridge script library), ...

4.5. Component Map

At the end of the boot sequence, all the devices present on the bus will send a REPORT_PRESENT message with their device class and device class version. AudioBridge records the source address and the device class information to build a component map.



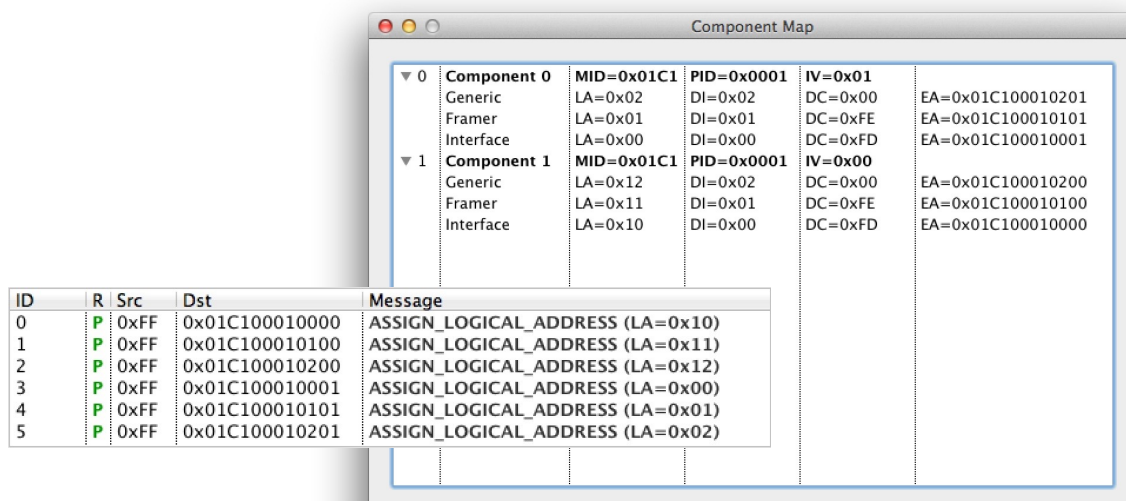
The screenshot shows the 'Component Map' window with two components listed. Component 0 has MID=0x01C1, PID=0x0001, and IV=0x01. Component 1 has MID=0x01C1, PID=0x0001, and IV=0x00. The message log shows six REPORT_PRESENT messages from various source addresses to 0xFF.

ID	R	Src	Dst	Message
0	P	0x01C100010201	0xFF	REPORT_PRESENT (DC=0x00, DCV=0x01)
1	P	0x01C100010200	0xFF	REPORT_PRESENT (DC=0x00, DCV=0x01)
2	P	0x01C100010101	0xFF	REPORT_PRESENT (DC=0xFE, DCV=0x01)
3	P	0x01C100010100	0xFF	REPORT_PRESENT (DC=0xFE, DCV=0x01)
4	P	0x01C100010001	0xFF	REPORT_PRESENT (DC=0xFD, DCV=0x01)
5	P	0x01C100010000	0xFF	REPORT_PRESENT (DC=0xFD, DCV=0x01)

Component	MID	PID	IV	Generic	Framer	Interface	EA
0	0x01C1	0x0001	0x01	LA=	LA=	LA=	EA=0x01C100010201
1	0x01C1	0x0001	0x00	LA=	LA=	LA=	EA=0x01C100010200

The above pictures are showing 2 Audio Bridges reporting present and being mapped. At this moment, none of the device are enumerated. This is shown by “LA=”, no Logical Address (LA) assigned yet.

When the ASSIGN_LOGICAL_ADDRESS messages are transmitted, the Logical Address in use for a device is displayed in the Component map.



The screenshot shows the 'Component Map' window after ASSIGN_LOGICAL_ADDRESS messages. The Logical Address (LA) is now assigned for each component's Generic, Framer, and Interface. The message log shows five ASSIGN_LOGICAL_ADDRESS messages.

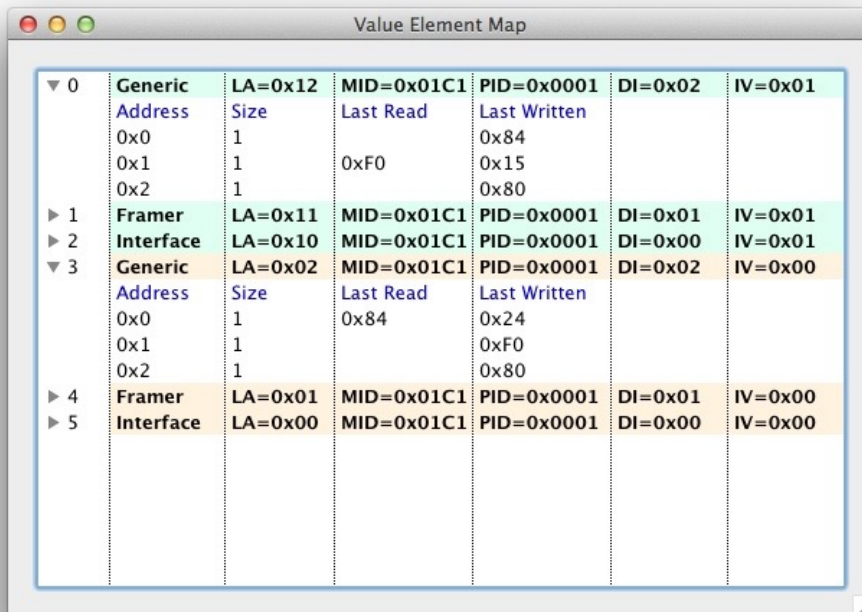
ID	R	Src	Dst	Message
0	P	0xFF	0x01C100010000	ASSIGN_LOGICAL_ADDRESS (LA=0x10)
1	P	0xFF	0x01C100010100	ASSIGN_LOGICAL_ADDRESS (LA=0x11)
2	P	0xFF	0x01C100010200	ASSIGN_LOGICAL_ADDRESS (LA=0x12)
3	P	0xFF	0x01C100010001	ASSIGN_LOGICAL_ADDRESS (LA=0x00)
4	P	0xFF	0x01C100010101	ASSIGN_LOGICAL_ADDRESS (LA=0x01)
5	P	0xFF	0x01C100010201	ASSIGN_LOGICAL_ADDRESS (LA=0x02)

Component	MID	PID	IV	Generic	Framer	Interface	EA
0	0x01C1	0x0001	0x01	LA=0x02	LA=0x01	LA=0x00	EA=0x01C100010201
1	0x01C1	0x0001	0x00	LA=0x12	LA=0x11	LA=0x10	EA=0x01C100010200

AudioBridge will keep tracking the Logical Address management messages to make sure that the component map is always up to date.

4.6. Value Element Map

AudioBridge tracks all the Value Element management messages to build a map of registers with their last read and last written values. The map is not exhaustive as it is a representation of the bus traffic. Value Elements that are never accessed will not appear in the list. A list of Value Elements is maintained for each logical device that is present on the bus.



	Generic	LA=0x12	MID=0x01C1	PID=0x0001	DI=0x02	IV=0x01
▼ 0	Address	Size	Last Read	Last Written		
	0x0	1		0x84		
	0x1	1	0xF0	0x15		
	0x2	1		0x80		
▶ 1	Framer	LA=0x11	MID=0x01C1	PID=0x0001	DI=0x01	IV=0x01
▶ 2	Interface	LA=0x10	MID=0x01C1	PID=0x0001	DI=0x00	IV=0x01
▼ 3	Generic	LA=0x02	MID=0x01C1	PID=0x0001	DI=0x02	IV=0x00
	Address	Size	Last Read	Last Written		
	0x0	1	0x84	0x24		
	0x1	1		0xF0		
	0x2	1		0x80		
▶ 4	Framer	LA=0x01	MID=0x01C1	PID=0x0001	DI=0x01	IV=0x00
▶ 5	Interface	LA=0x00	MID=0x01C1	PID=0x0001	DI=0x00	IV=0x00

The content of the Value Element Map can be saved in a CSV file format.

Go to **File/Export Register Configuration**.

4.7. Information Element Decoding

AudioBridge decodes the content of the Information Element management messages that are appearing in the LiveView and adds the information in the help tag of the message cell. The Core Information Elements and the Class Specific Information Elements are supported for all the device classes described in the SLIMbus specification.

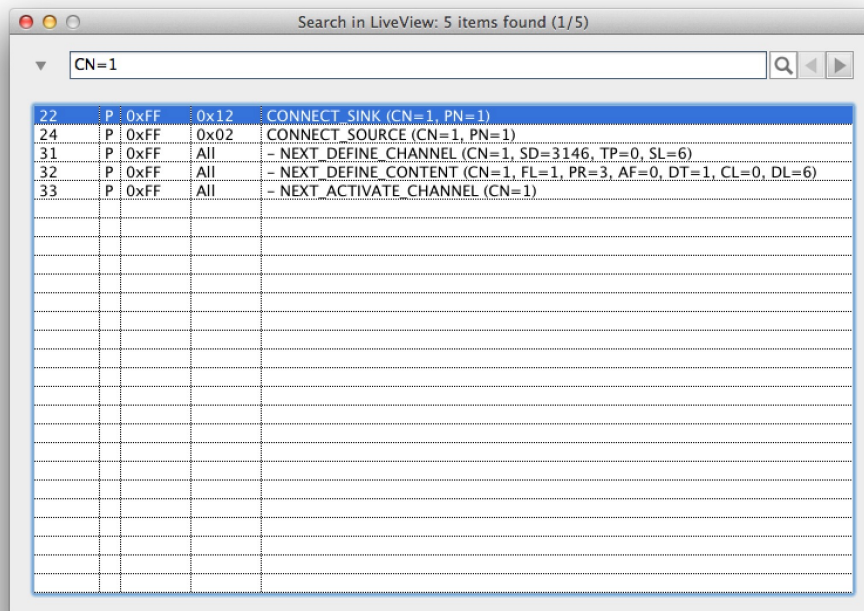
12	P	0xFF	0x00	CONNECT_SINK (CN=0, PN=0)
13	P	0x00	0xFF	Interface/ Unsupported Msg N (AM=1, BA=0x000, SS=0, IS=0x01)

Just place the mouse above the message and a help tag will appear, if relevant. The help tag will be generated for the REPORT_INFORMATION message and for the REPLY_INFORMATION message if the corresponding request has also been captured by the LiveView.

4.8. Search Engine

When capturing many hundreds or thousands of messages, it might be helpful to get an easy navigation through them. To perform a search in the Live View, go to the menu **Edit/Search in LiveView**. A new window will appear.

Type the request in the search field and hit enter or click on the search button.



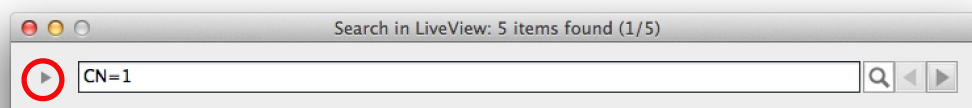
The search applies to:

- Parameter name and parameter value
- Source and Destination addresses
- Message string with abbreviated parameter names and value
- Response field (use "NACK", "PACK", "NORE", "UDEF")
- The cell help tags

For instance, to find all the messages in relation with channel 1, just type CN=1 in the search field. To find messages reporting loss of syncs, just type "loss" in the search field.

Clicking on a row of the search list highlight the corresponding row in the LiveView. The **UP** and **DOWN** keyboard keys can also be used to navigate through the list. A right click (contextual menu) on the search list allows to select all found items in the LiveView list.

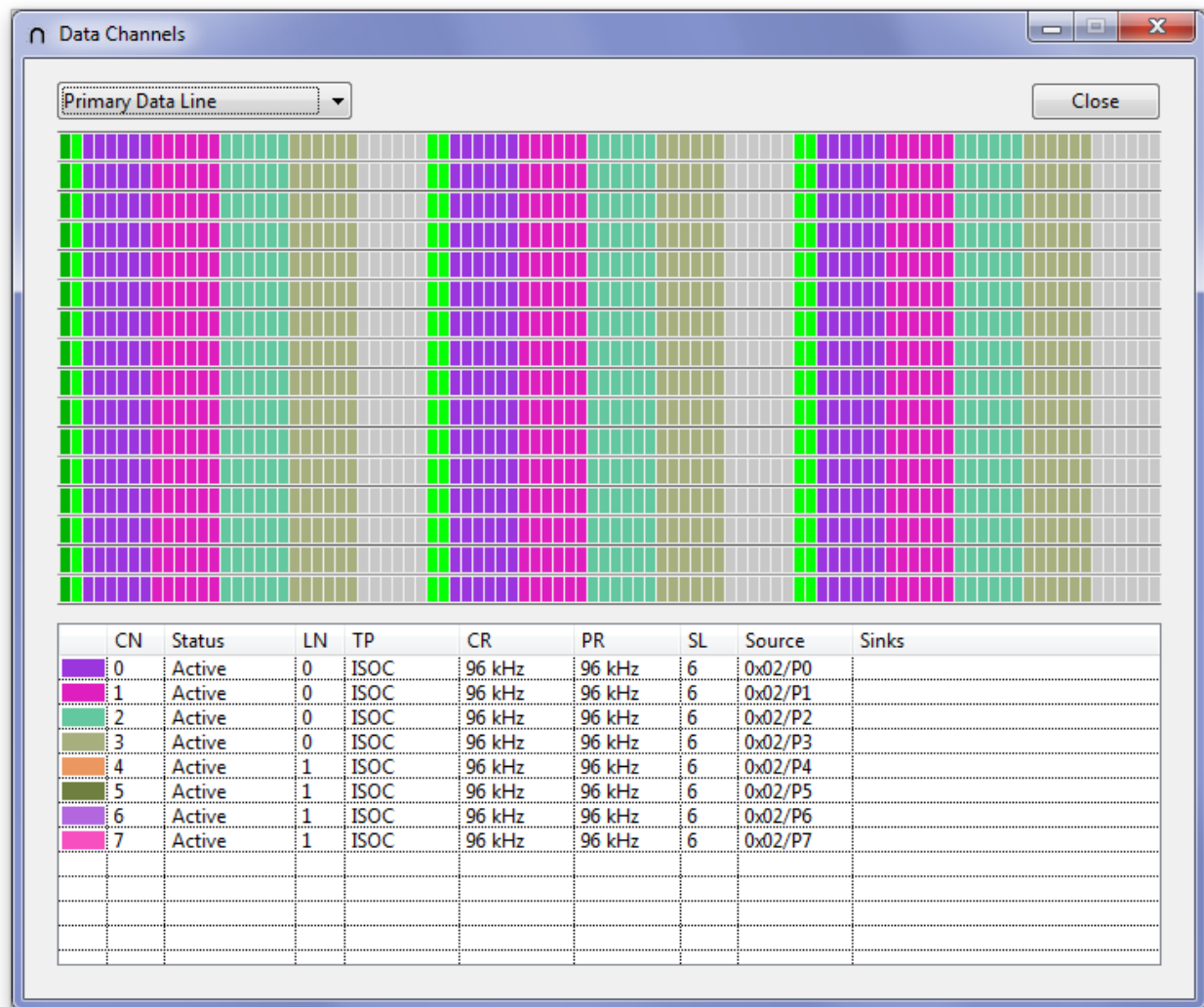
For easy search, it is possible to shrink the search window by clicking on the arrow at the left side of the search text field.



Use the "<" and ">" buttons on the right side to navigate through the list.

4.9. Data Channel View (12 ports bridge model)

This window shows the position of the data channel in the superframes and for each data lines (from 0 to 7).



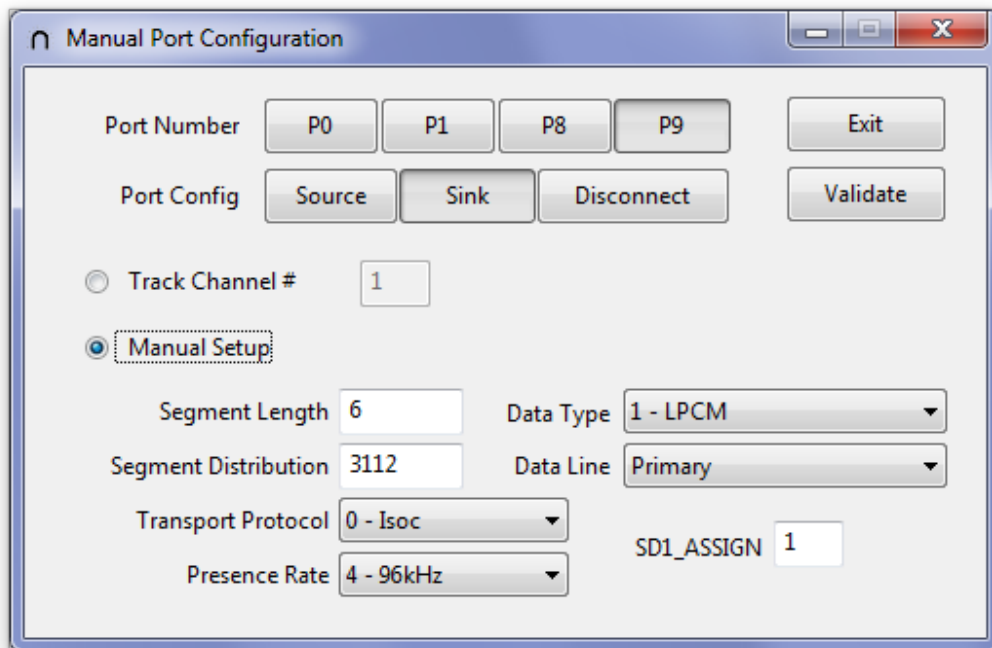
The window is refreshed in real time (or as close as real time can mean with a graphical user interface). Any change in the subframe mode and the channel definition will be shown. The table below the graphical area also displays some important channel information.

The ports 0, 1, 8 and 9 of the 12 ports audio bridge can be manually configured to connect to any existing channels without the need of SLIMbus configuration messages. If any of these special ports are available, right click on any channels of the list and a popup menu will offer the possibility to connect one of these ports to the desired channel. This feature is especially useful to sniff the content of a running channel without disturbing it.

The user also has the possibility to fully manually configure the port parameters by calling the **Manual Port Configuration** panel in the **Tools** menu.

4.10. Manual Port Configuration (12 Ports Bridge Model)

This panel offer the user to manually define the parameters of the port 0, 1, 8 and 9. There are two modes of operation: Semi-automatic and fully manual.



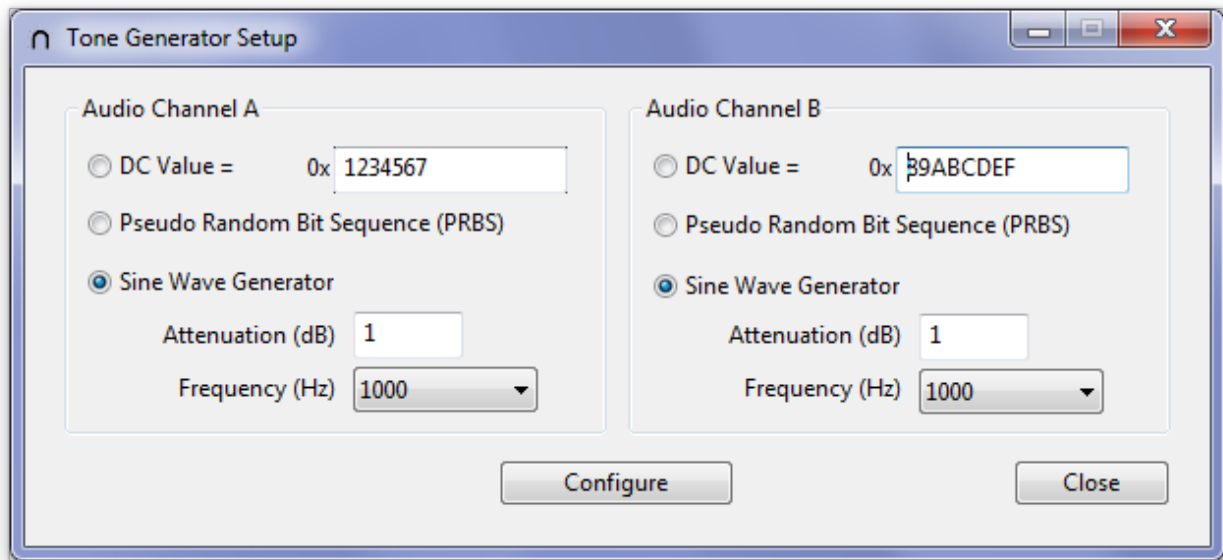
Semi-automatic mode: Define the channel number of interest for the port and the port will track all the messages related to this channel. This is useful if the channel parameters are not known by the user.

Manual setup: All the parameters of the port can be defined manually, allowing the bridge to capture any portion of the data space without having being involved in any message transaction.

When a port is configured, press the **Validate** button to activate the changes.

4.11. Tone Generator setup (12 Ports Bridge Model)

The audio bridge has an internal stereo signal generator that can feed ports 0 to 7. The two channels are independent.



Three types of signals are proposed:

- A constant (DC) values coded on 32 bits.
- A pseudo random bit sequence (PRBS) based on a 16 bit linear feedback shift register
- A sine wave with programmable amplitude (by step of 1 dB) and selectable frequencies

It is necessary to press the **Configure** button to activate the changes.

4.12. Signal Analysis (12 Ports Bridge Model)

The audio bridge hardware is able to capture the samples generated by all of its sink configured ports (up to 12 ports). A maximum of 8192 segments can be captured at a time.

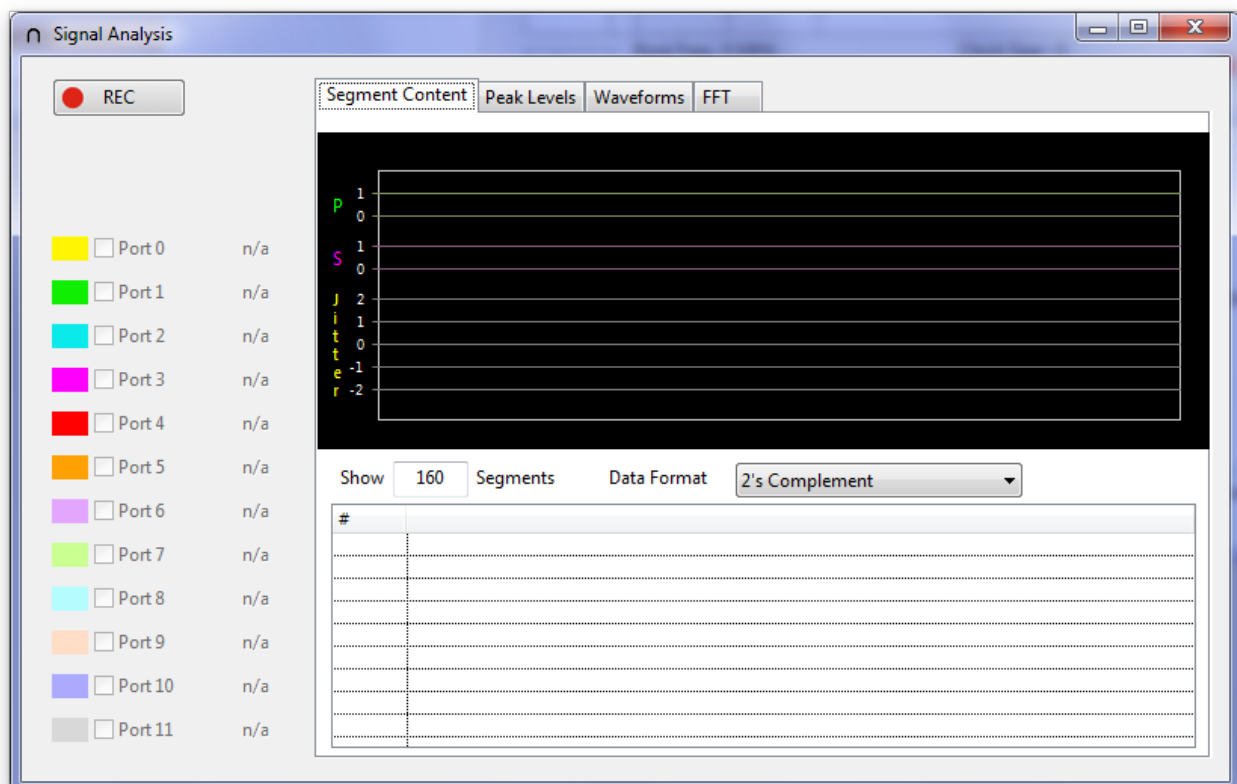
These segments are processed to offer variety of information:

- Segment content (TAG bits and DATA slots) and a TAG bit analysis as well as the Presence bit jitter calculation
- Peak level analysis of the various signals
- Waveform display of the various signals
- Fourier Transform of the various signals

4.12.1. Data Recording

The Signal Analysis panel is split in two areas. The signal analysis tabs, graphs and tables are located on the right side.

The record controls are located on the right side. Pressing the **REC** buttons launch the capture of the segments (8192 in total). Before launching a capture, the target ports must be selected. When a port is active and configured as a sink, it will appear bold and will be selectable. The colored rectangle besides the port check box indicates the color in which most of the port data will be displayed. When clicking in the rectangle, the port graphs will be highlighted (bold trace).

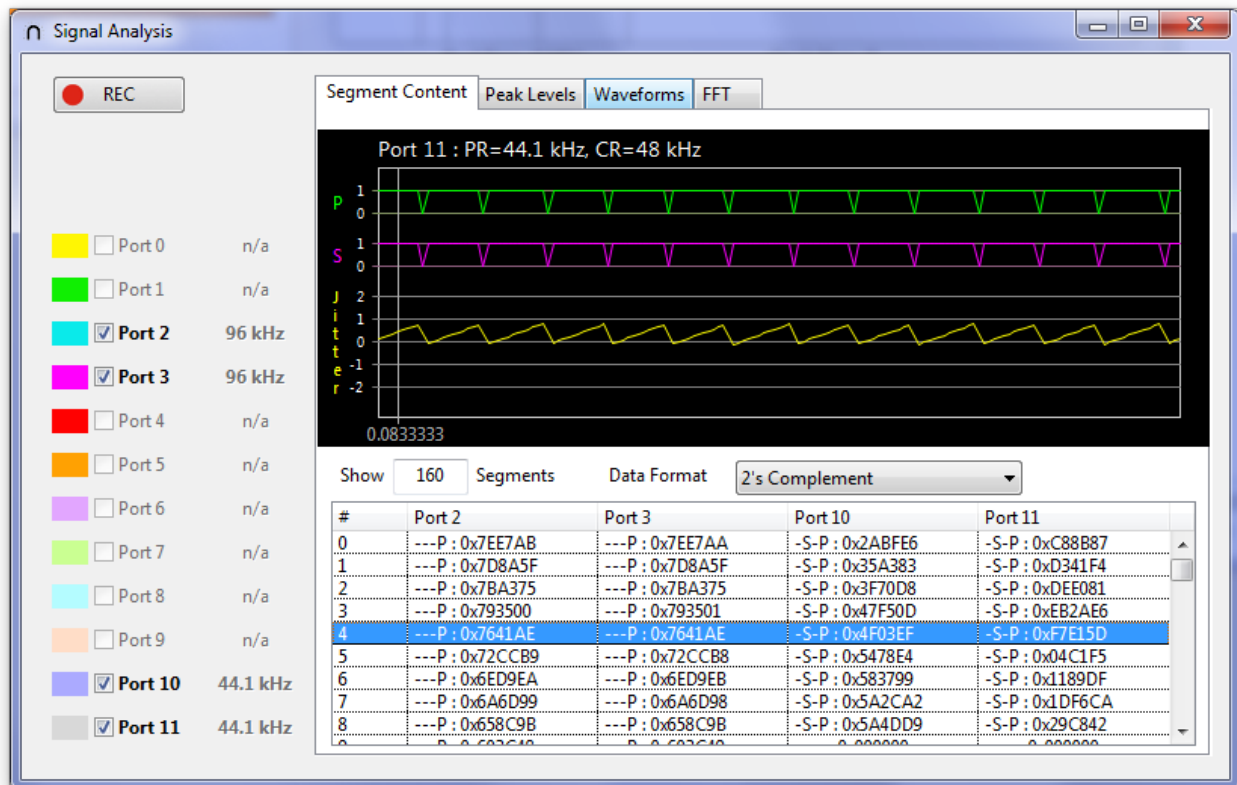


At the end of the capture, all the tabs are automatically updated.

A capture always starts at the beginning of a superframe.

4.12.2. Segment Content

In this tab, the segment content for each selected ports is shown. The TAG bits are also explicitly shown. P is the Presence bit, S is Strobe bit for the Pushed protocol and the SRQ (Sample Request) bit for the Pulled protocol. Both P and S bits are displayed in the graph. The P bit jitter is also computed and displayed.



To display the TAG bit graphs of a given port, just click on the desired segment in the table. The vertical marker and a time stamp will be shown at the position of the selected segment.

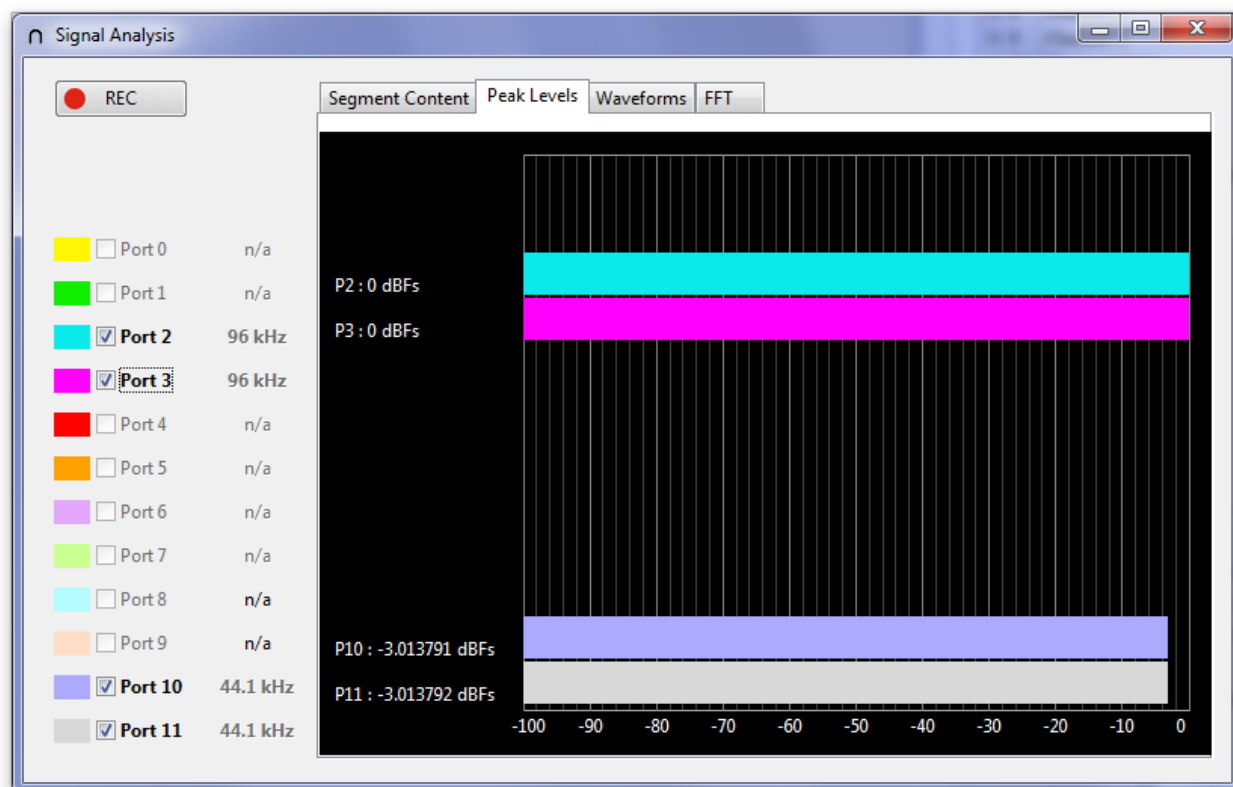
The user can choose the number of segment to be shown on the graph. When this number is below or equal to 20, the segment position is figured out by a point.

The user can also change the data format of the segment. By default, the data is shown in the 2's complement format. However, if the segment is using the offset sign & magnitude coding, it is possible to display the data as they are transported over SLIMbus.

4.12.3. Peak Levels

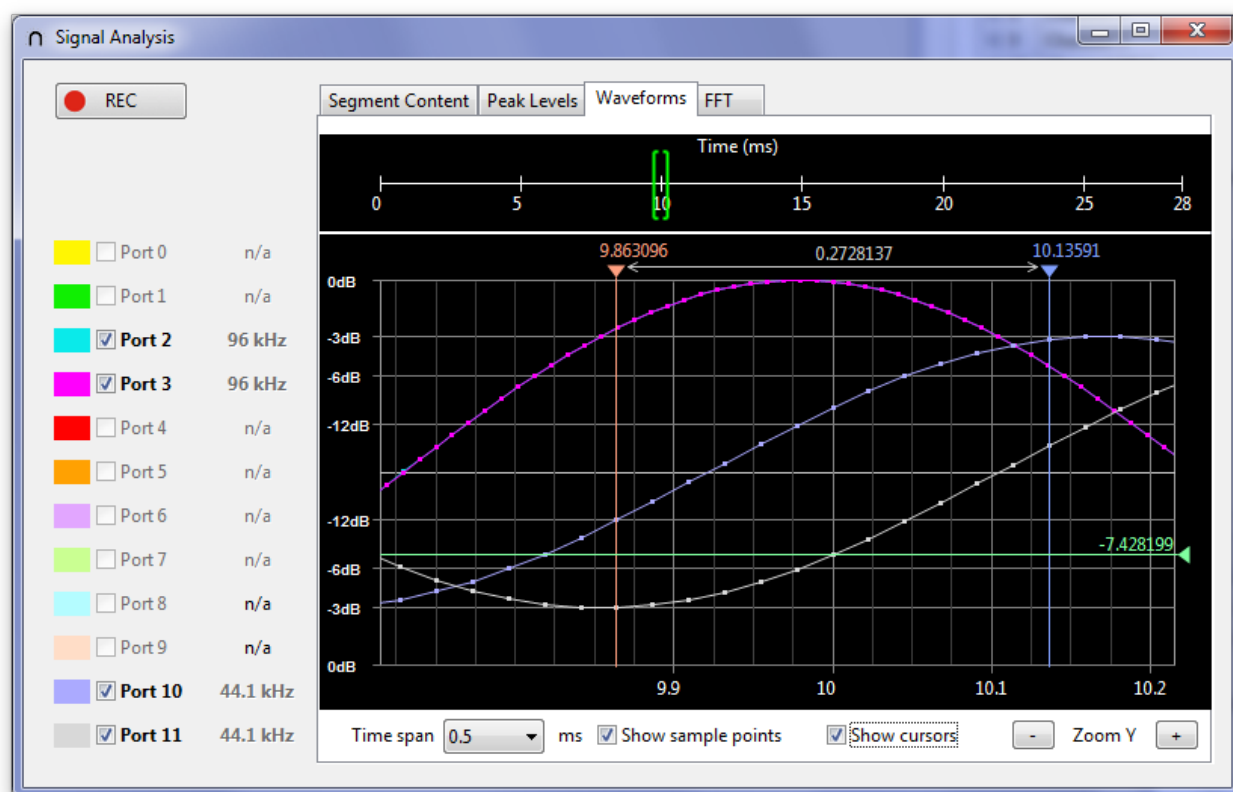
This tab displays graphically the peak level of the capture signals. A quadratic interpolation is used to best estimate the real level. Relying on the sample values usually underestimate the peak level measurement. Note that in case of random noise, the peak level estimator may displays values greater than 0 dBFs, which is theoretically impossible. This is due to the interpolation method.

The peak level measurement does not have any particular controls or options.



4.12.4. Wave forms

The graph is split in two zones. The top one is a time marker indicator. It shows the user of the displayed wave forms span over the capture period and also allow him to scroll the data over time. Just click (and eventually drag) at the desired to display the data.



The time span can be changed from 0.25 ms up to 32 ms. It is also possible to force the marking of the samples by dots. Note that when Pushed or Pulled protocol is used, the wave form is reconstructed by filtering out the empty segment and using the broadcasted Presence Rate as a timing reference.

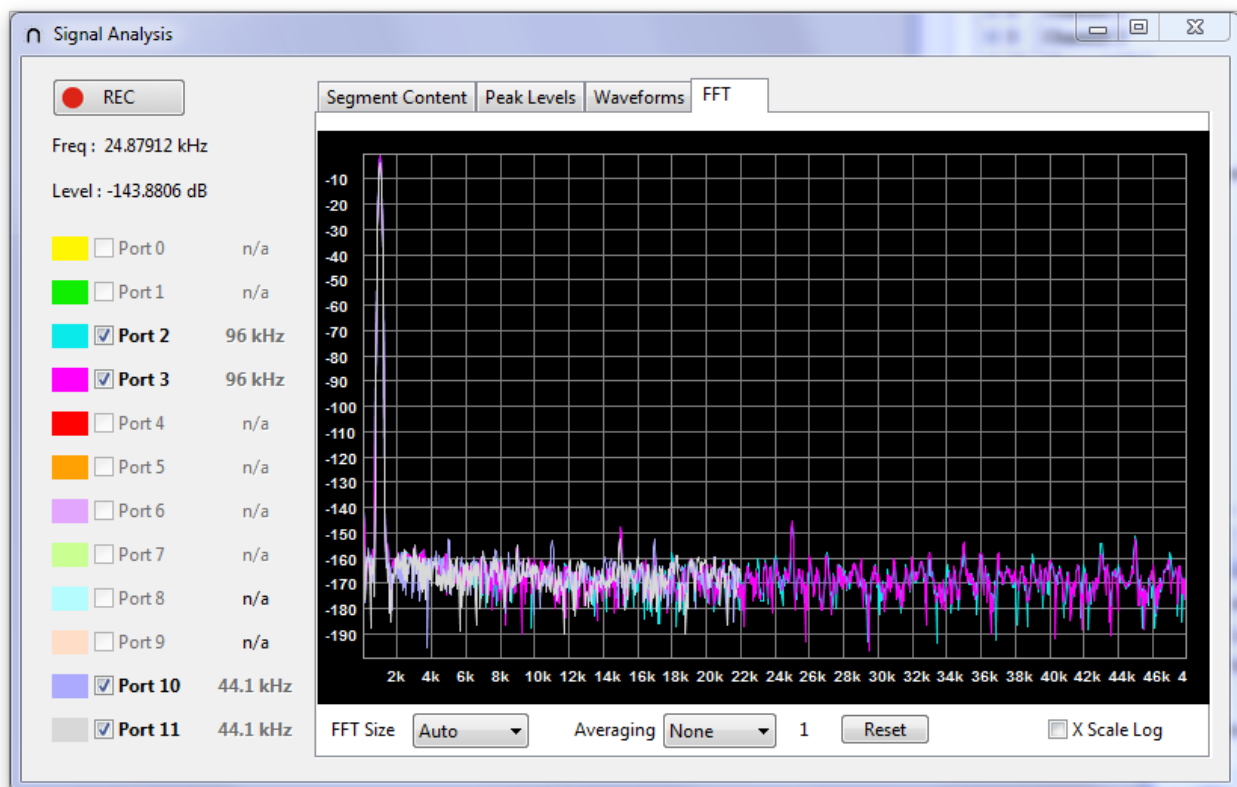
It is also possible to display time and amplitude markers on the graph. To control them, just click on the top or right arrows and drag the cursor to the desired location.

To move the wave forms in time, the user can click anywhere on the traces and drag them (left or right). This allows for a fine grain time shift.

The Y axis (amplitude) of the trace has a zoom capability but the trace stays centred on 0.

4.12.5. FFT - Spectral Analysis

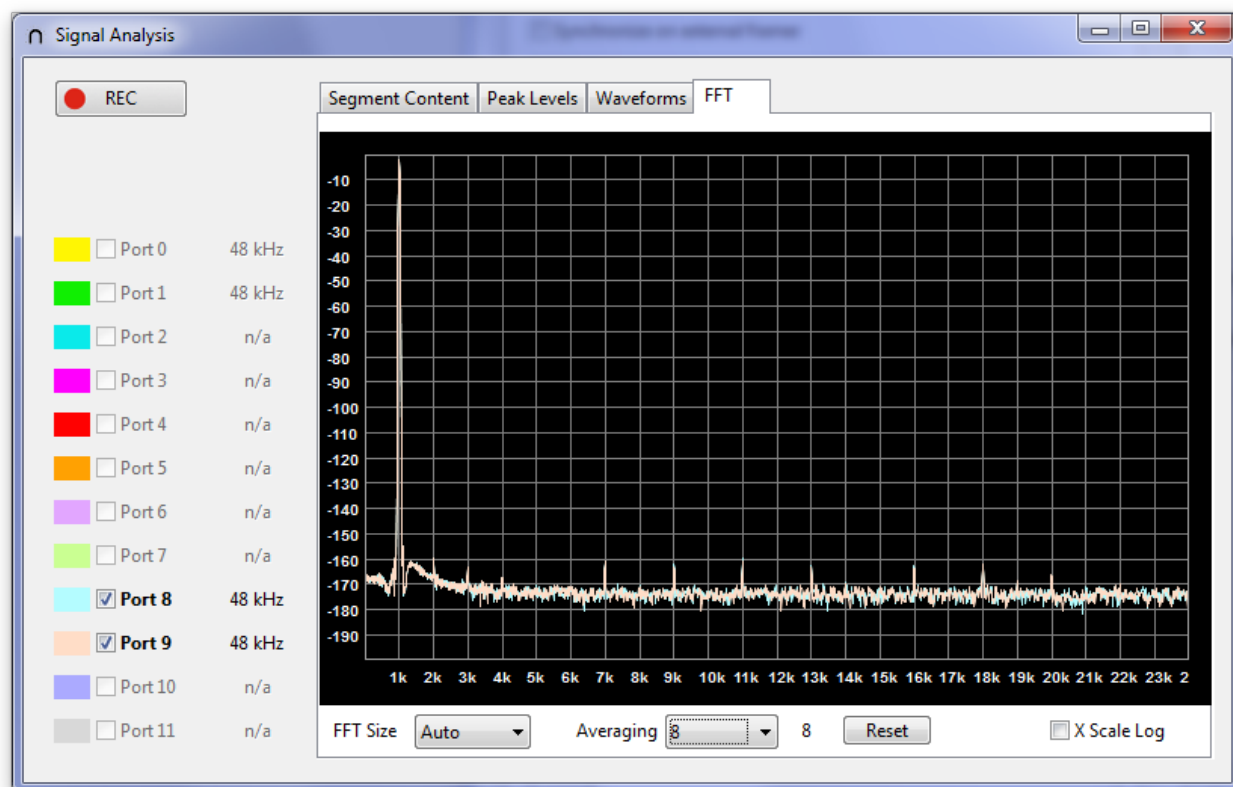
The graph can mix FFT traces having each a different sampling rate. The highest sampling rate dictates the maximum displayed frequency.



When the FFT size is set to auto, the software uses as many points for each port as possible. However, the user can manually select any desired value between 256 and 8192 sample points. However, if there are not enough captured points, the data array will be padded with 0s and the graph will not be representative of the real spectral content of the signal. It is strongly advised to keep the “**Auto**” size selection.

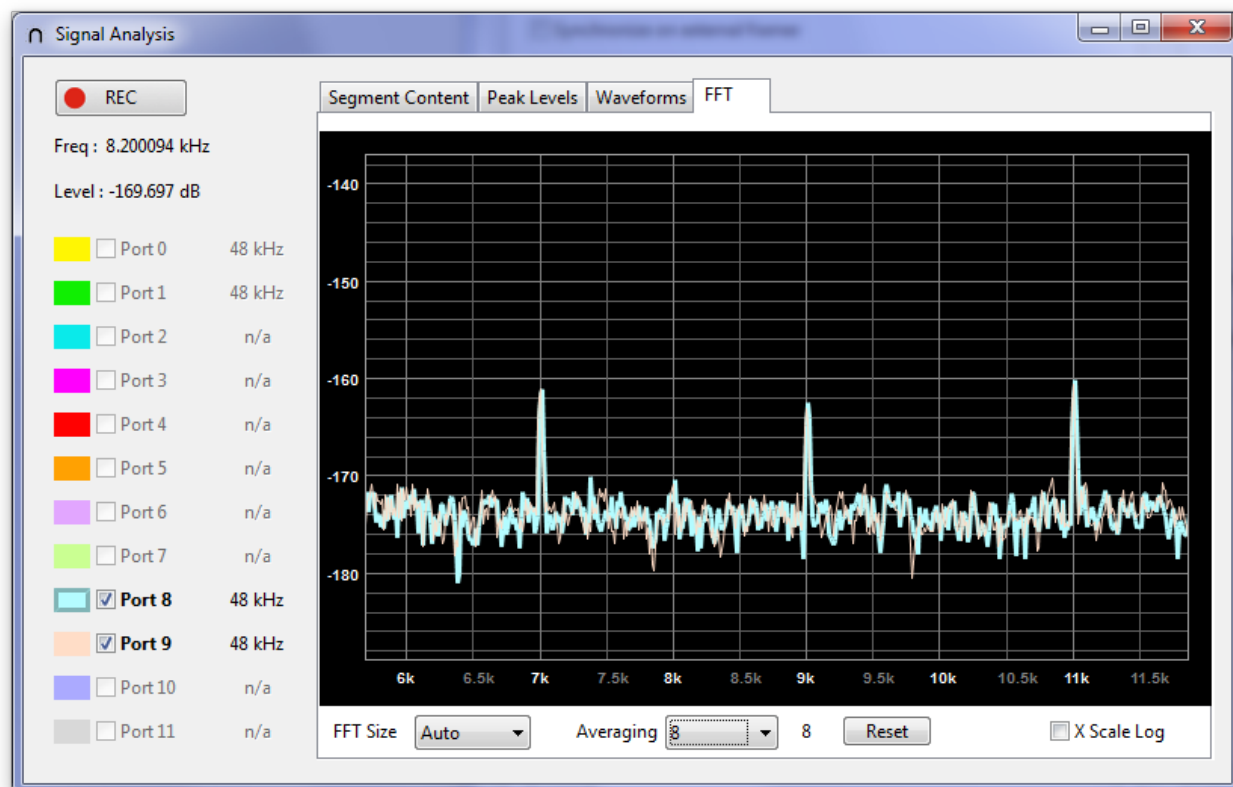
It is possible to average multiple captured traces to limit the noise effect. The number of traces to be averaged must be selected (from 2 to 16). Every time the user will record samples, the averaging engine will process the data. For instance, to get an averaged display over 8 captures, the user will have to manually press 8 times the REC button to effectively capture all these 8 traces. The **RESET** buttons delete all the previously captured traces. Another multiple acquisitions will then be required to show an average trace.

The FFT window is a custom LnK implementation to have very low side lobes.



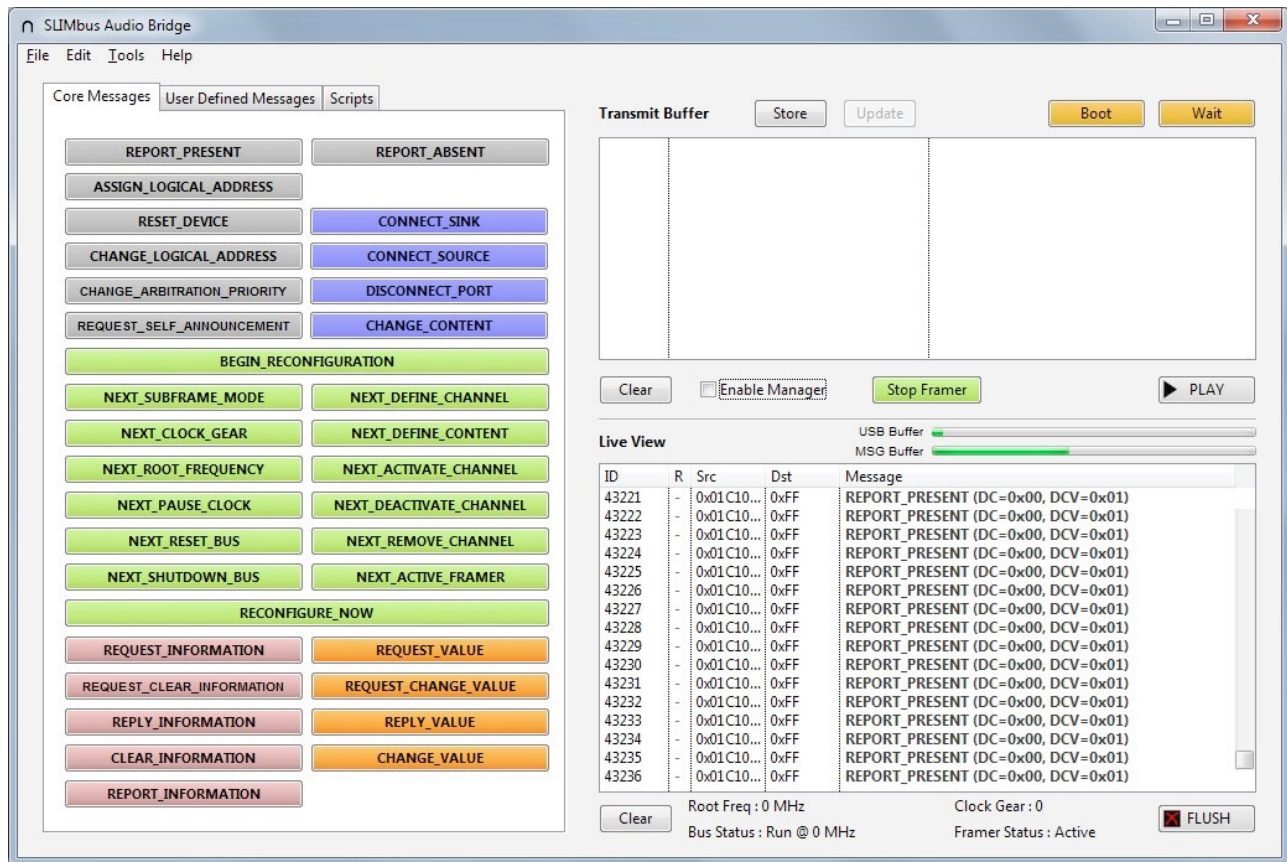
Use the mouse wheel to zoom in and out on the traces. The zoom will happen around the pointer of the mouse. Clicking on the graph and moving the mouse will drag the picture to show other portion of the traces.

It is also possible to zoom on a desired part of the graph by holding the **ALT** key will selecting a portion of the graph (click and move to draw a rectangle).



5. Hardware Operation

Once the bridge HW has been detected by the application, the buttons “PLAY”, “Start Framer” and “REC” are displayed, allowing the control of the bridge hardware.



5.1. Message Sniffer Activation

Click on the “REC” button (located in the bottom right corner of the window) to activate the message recording in the bridge hardware. Once it is done, the button will show “STOP”, indicating that clicking again on that same button will stop the message capture.

There are limitations about the message rate the sniffer can support. The Audio Bridge hardware is capable of sniffing and storing a continuous message flow at 27 MHz and 100% control mode. However, this corresponds to about 26.7 Mbps or in average 333750 messages a second. The software application can decode about 1000 messages a second (depending on CPU speed and the number of core). There is a big difference between what can be streamed on the SLIMbus and what can be shown real time by the bridge application. Another big limitation of the system is the VCP (Virtual Comm Port) drivers of Windows that are limiting the actual throughput over USB to about 900 kbps. The internal buffer of the Audio Bridge (512 messages) will get full pretty fast. A special mechanism has been implemented to let the user know about the number of missed messages: “Missed Msg (NB=124)” for instance.

To summarize, the bridge Live View display will cope with average message bit rate below 900 kbps without losing messages. However, as the decode capability is about 1000 messages a second, the internal buffers of the application will fill up quite rapidly. The software buffer size will allow for about 30 seconds recording. After that, the buffers will overflow and data will be lost. If messages are transmitted in burst mode (groups of less than 512 messages), even at full speed, there will not be any limitations as long as the average sustained bit rate is lower than 900 kbps.

The buffer fill-up status is shown by 2 progress bars. This allows the user to eventually stop the recording before a buffer overflow happens. Once the sniffer stopped, the buffers will still feed the message decoder till they get empty. In the case, the sniffer control button will show “**FLUSH**” instead of “**REC**”. Click “**FLUSH**” to empty the buffers and stop any decoding activities, if desired. Once the buffers are empty, the button will show “**REC**”.

After capturing 100k messages, the sniffer will automatically be stopped and the buffer flushed.

If the capture of messages transmitted at full speed has to happen without any loss of data, the SLIMbus Protocol Analyzer is a much more efficient tool than the sniffer of the Audio Bridge application.

5.2. Framer Activation

When the bridge needs to clock the bus, its Framer must be activated. Click on the “**Start Framer**” button to turn on the Framer and boot the SLIMbus. The button will then show “**Stop Framer**”, indicating that clicking again on that same button will stop the framer and consequently stop the bus.

During a clock pause event, the button caption will become “**Resume**”. Clicking on the button will resume clock operation.

5.3. Manager Activation

The Manager functionality of the bridge simply consists in acknowledging all messages having **0xFF** as destination address. It is enabled by default. If the bridge Manager is disabled and there are no other Manager on the bus, none of the REPORT_PRESENT messages will be positively acknowledged and the devices will keep transmitting their REPORT_PRESENT messages for ever. The sniffer will not be able to follow such high message traffic.

5.4. Bridge Control Panel

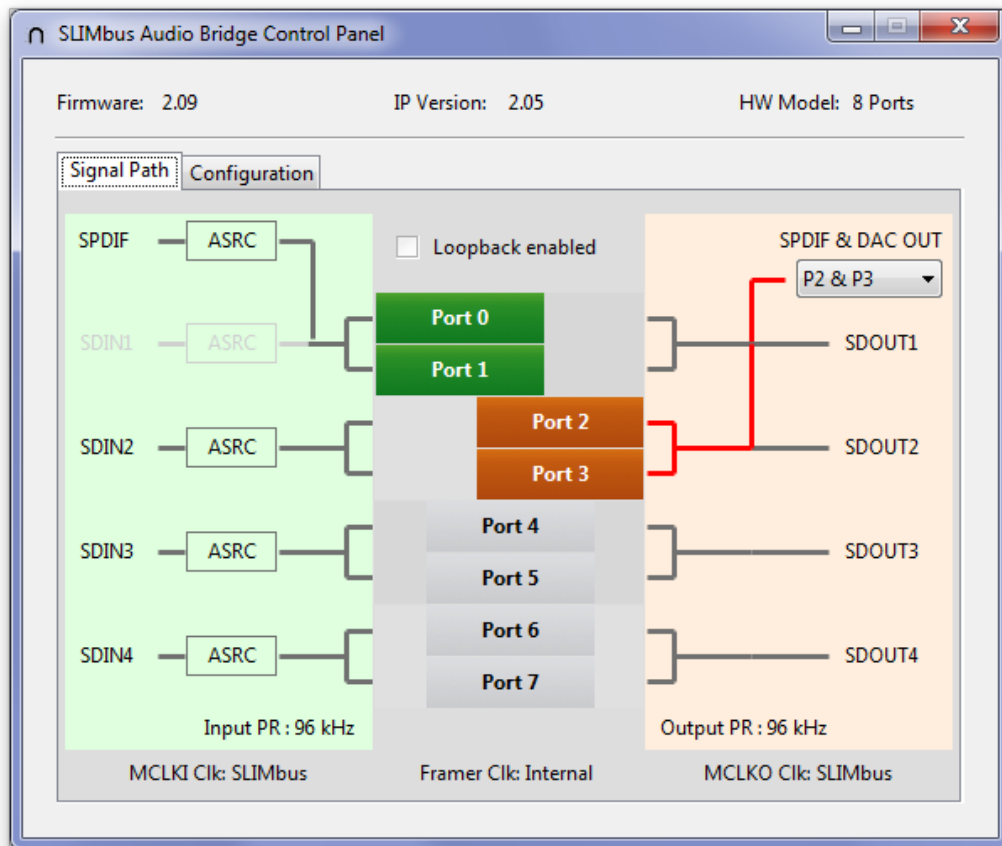
Most of the hardware function of the bridge can be controlled by software. These functions are described in detail in the bridge hardware user manual. The control panel depends on the bridge model. The software can control the 2/8 ports bridge (first generation) or the 12 ports bridge (2nd generation). The control panel reflects the real time status of the bridge. Any changes initiated by some SLIMbus transactions will be automatically reflected on the panel. That is especially true for the data port settings.

5.4.1. 2/8 Ports Bridge Model

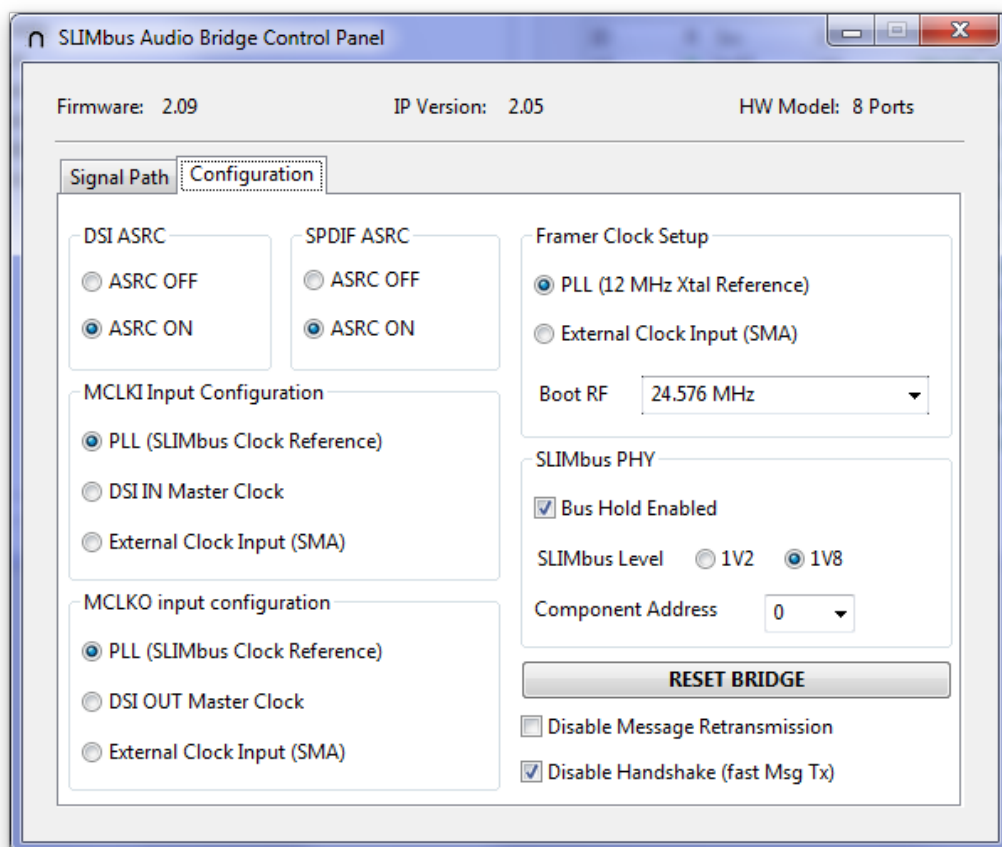
The control panel consist in 2 sections: the signal path and the parameter definition.

In the **Signal Path** pane, the port configuration is shown. A port that is not configured appears grey and is located in the middle of the graph. When a port is configured as a source, it appears green and is located on the left side. When a port is configured as a sink, it appears red and is located on the right side.

The input signals are located on the left side. The output signals are located on the right side.



The **Configuration** pane gives access to all the programmable features of the audio bridge hardware.

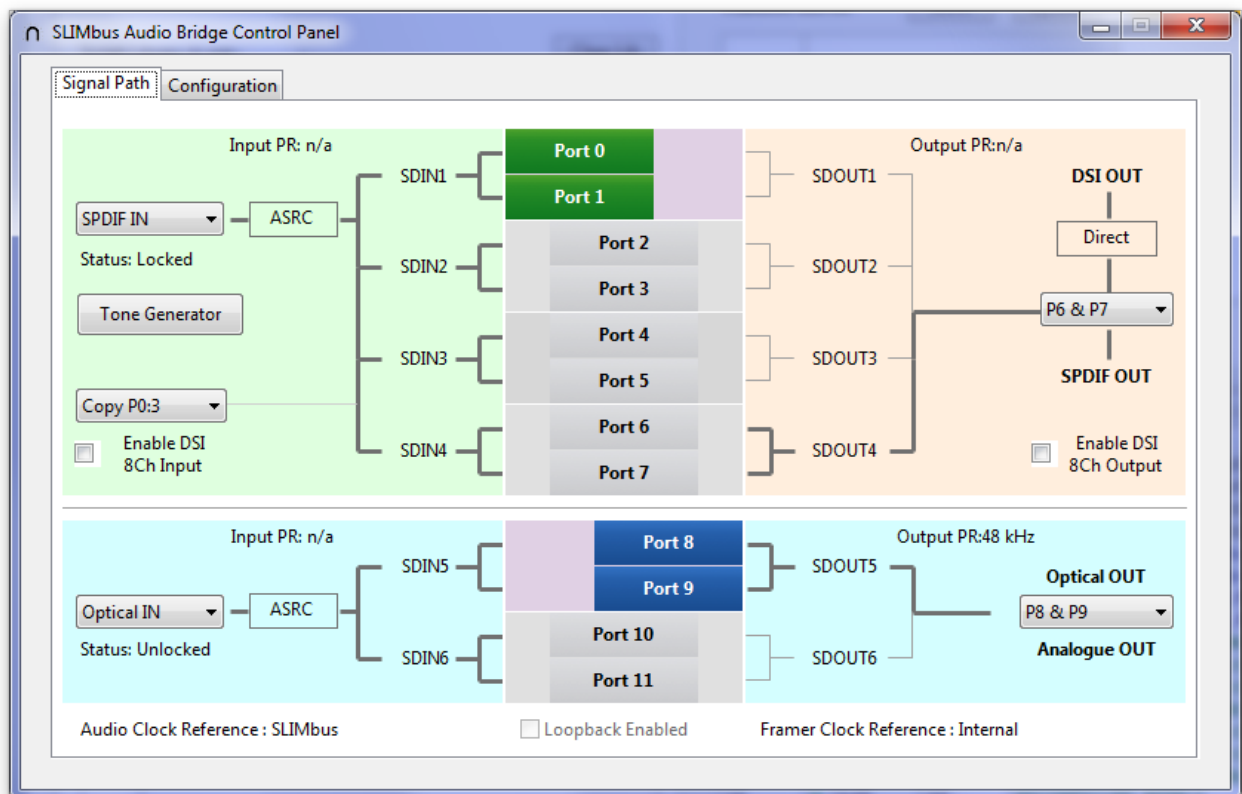


Note 1: The message retransmission (in case of NORE or NACK) is only controllable through the bridge control panel. This function enables or disables the retransmission of a message when this one does not receive a positive acknowledgment.

Note 2: By default there is a transmission handshake between the bridge and the software. On slow computers, it may be necessary to keep it active to prevent loss of data. On fast computers, a trial can be given to disable the handshake. Without handshake, transmission of messages is much faster.

5.4.2. 12 Ports Bridge Model

The control panel consist in 2 sections: the signal path and the parameter definition.



The Signal Path pane shows the port configuration and the input and output signals. It mimics as closely as possible the GUI of the bridge hardware. Port boxes that are moved to the left are source ports. Port boxes that are moved to the right are sink ports. The input signals are located on the left side. The output signals are located on the right side.

The **Configuration** Pane is split into 2 sections: SLIMbus and Audio parameters and the Clock tree parameters. Refer the the bridge hardware user manual for the description of each of these parameters.

SLIMbus Audio Bridge Control Panel

Signal Path Configuration

Firmware: 3.00.000 IP Version: 3.00 HW Model: 12 Ports

INPUT 1 ASRC

☐ Disabled ☒ Enabled

INPUT 2 ASRC

☐ Disabled ☒ Enabled

DSI OUT ASRC

☒ Disabled ☐ Enabled

Master Clock: MCLKO1

☐ Disable Handshake (fast Msg Tx)

☐ Disable Message Retransmission

Reset SLIMbus Core

DSI Common Setup

DSI Signal Level: 3V3

DSI Input Setup

Format: Left Justified

Sample Size: 16 bits

Clock Mode: Slave

DSI Output Setup

Format: Left Justified

Sample Size: 16 bits

Clock Mode: Slave

SLIMbus PHY

PHY Signaling level: 1.8 V

☒ Data Bus Hold Enabled

Framer Boot information

Component Address: 0

Framer Mode: Inactive

SM: 19 - 4/32

CG: 9 - 6.4 to 14.4 MHz

RF: 1 - 24.576 MHz

Touch Screen Calibration

Factory Reset

SLIMbus & Audio Clocks

SLIMbus and Audio parameters

SLIMbus Audio Bridge Control Panel

Signal Path Configuration

Firmware: 3.00.000 IP Version: 3.00 HW Model: 12 Ports

External clock setup

Direction: Input Output Level: 3V3

Input Impedance: 1 MOhms

Frequency: 0.0000 MHz

Output Clock Source: Framer Clock

Output Clock Divider: None /2 /4 /8 Custom 3

Frequency: 24.5760 MHz

PLL Reference Clock Setup

Framer PLL Reference		Audio PLL Reference
Internal		SLIMbus
SPDIF IN	11.2896 MHz	SPDIF IN
OPTICAL IN	0.0000 MHz	OPTICAL IN
DSI MCLK	0.0000 MHz	DSI MCLK
SMA IN	0.0000 MHz	SMA IN

SLIMbus & Audio Clocks

Clock tree parameters

5.5. Playing a Script

There are 2 ways to play a script:

- Press the “**PLAY**” button. It will start the transmission of the message transmit buffer content on SLIMbus. While the script is playing, the button caption will turn to “**STOP**”. Pressing the button when “**STOP**” is displayed will cause the running script to abort. When a script is completely transmitted, the buffer is flushed. When a script is aborted, the buffer remains unchanged.
- Drag and drop a script from the Script Library panel right on the LiveView panel. The script will be automatically played till it is finished.