SLIMbus Audio Bridge
Validation and Experimentation

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

Table of Content

1. Purpose of the document

This document presents a list of experiments that can be done with the SLIMbus Audio Bridge and the SLIMbus Protocol Analyzer & Traffic Generator.
An audio analyzer will also be required for the tests.

With these experiments, the user will get used to the tools and will also be able to verify that the SLIMbus Audio Bridge and the SLIMbus Protocol Analyzer & Traffic Generator are operational and are meeting the product specifications.

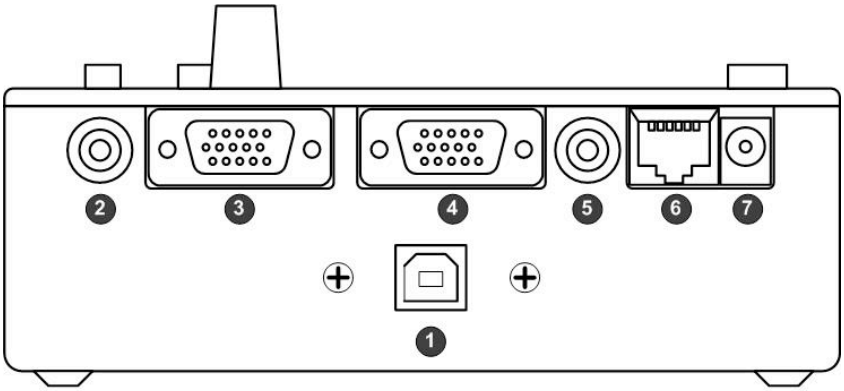
This document is **NOT** a substitute to the product user manuals. Thus, the user is kindly asked to read these manuals first.
Some very basic knowledge of the SLIMbus communication protocol and physical layer is also required.

We assume that the user is also familiar with audio testing and audio analyzer hardware and software.

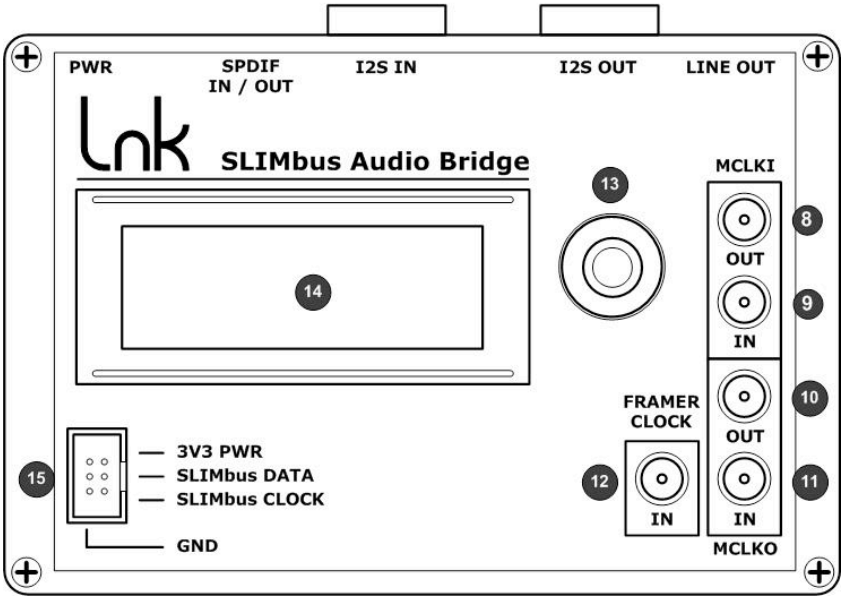
2. Presentation of the equipments

The tests described in this document will require a SLIMbus Audio Bridge, a SLIMbus Protocol Analyzer and a SLIMbus mini hub to connect all the devices together. An audio analyzer will also be required for the audio tests. Any kind of analyzer is suitable. However, the APx500 models from AudioPrecision are especially suited for these tests. For detailed information about these tools, please refer to their respective user manual.

2.1. SLIMbus Audio Bridge



1 - USB connector
2 - Analog line output
3 - I2S multichannel output
4 - I2S multichannel input
5 - SPDIF IN / OUT
6 - Factory Service
7 - Power supply

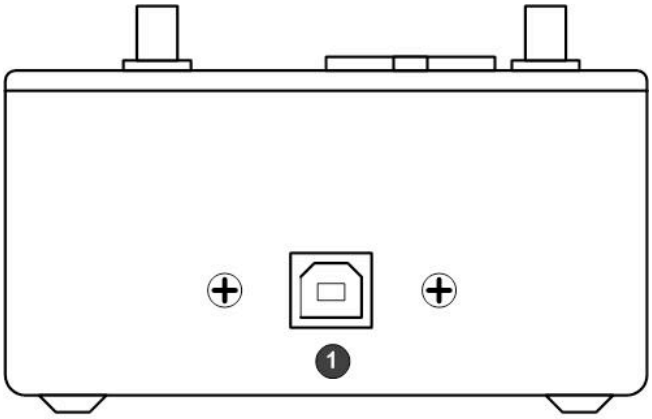


8 - Audio Master clock out
9 - Audio Master clock in
10 - Audio Master clock out
11 - Audio Master clock in
12 - Framer clock input
13 - Rotary knob + push
14 - Display
15 - SLIMbus connector

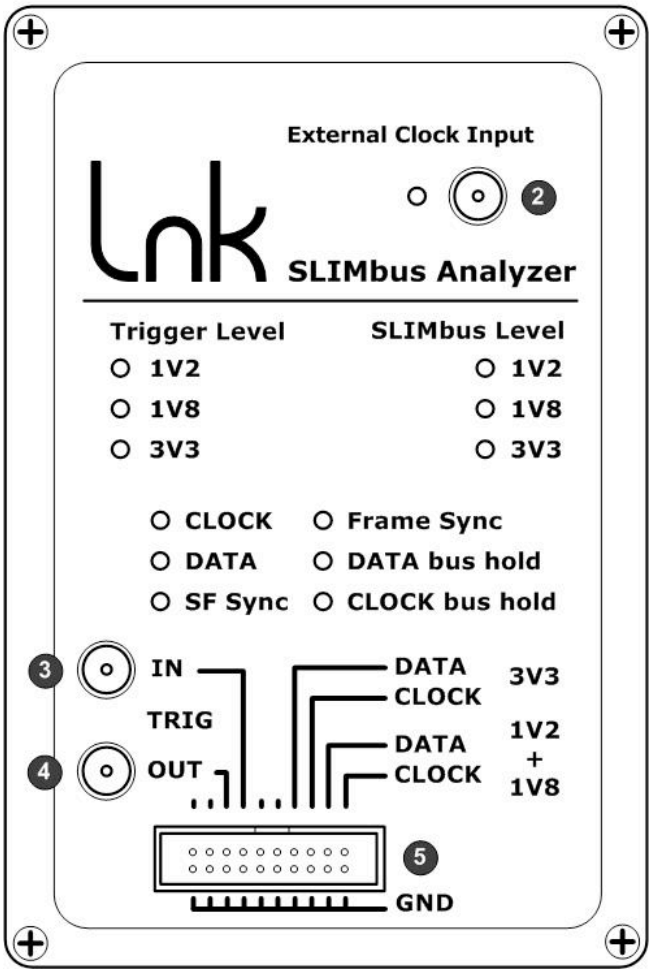
The SLIMbus Audio Bridge will transmit (or receive) digital audio streams on (from) SLIMbus. The bridge features SPDIF input and output and I2S multichannel input and output. The I2S interface use the Left Justified data format and 64 bits per frame. The I2S IN/OUT multichannel interface is pin/signal compatible with the Audio Precision DSIO interface of the APx500 machines.

The 8 port bridge needs a power supply and needs to be connected to a PC through USB to be operational. The 4 port bridge is autonomous. It just needs a power supply.

2.2. SLIMbus Protocol Analyzer



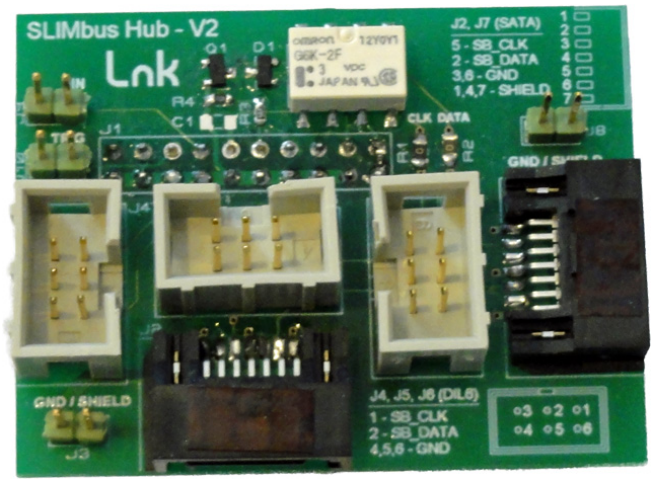
1 - USB connector



2 - Framer Clock Input
3 - Trigger input connector
4 - Trigger Output connector
5 - SLIMbus connector

The Analyzer / Generator hardware needs to be connected to a PC through USB. The SLIMbus Protocol Analyzer software must be launched to operate the analyzer hardware. The Protocol Analyzer will capture and decode all the SLIMbus traffic. The Traffic Generator will build and transmit the SLIMbus stream, especially the messages needed to configure the bus and the Audio Bridge to setup audio channels and transmission.

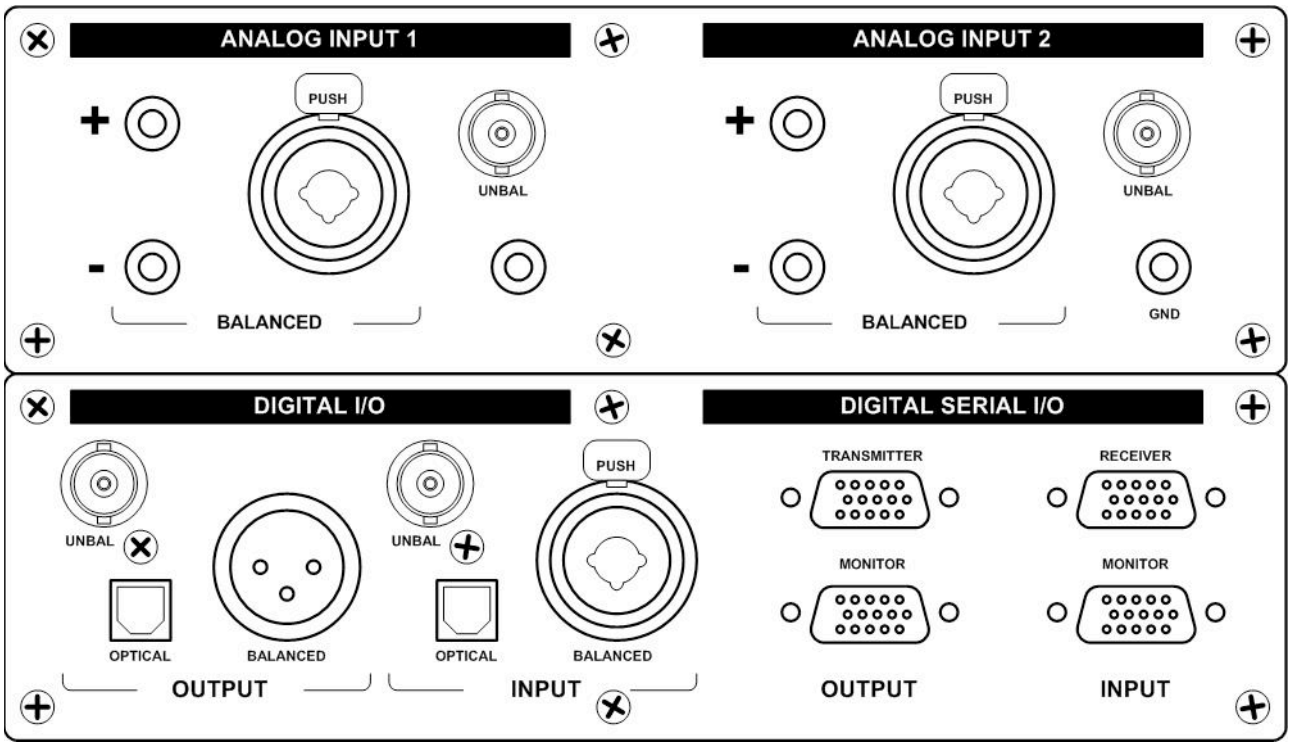
2.3. SLIMbus mini hub



This little board shall be plugged on the SLIMbus Protocol Analyzer hardware, in its SLIMbus connector. It provides SLIMbus signals on three DIL06 connectors and on 2 SATA connectors. The use of the mini hub is not mandatory but makes the connection to the SLIMbus Audio Bridge pretty easy.

2.4. Audio Analyzer

The audio Analyzer shall provide at a minimum a SPDIF input, a SPDIF output on coaxial connector (unbalanced) and 2 unbalanced analog inputs. For the multichannel tests, a DSIO (I2S multichannel) interface is needed.



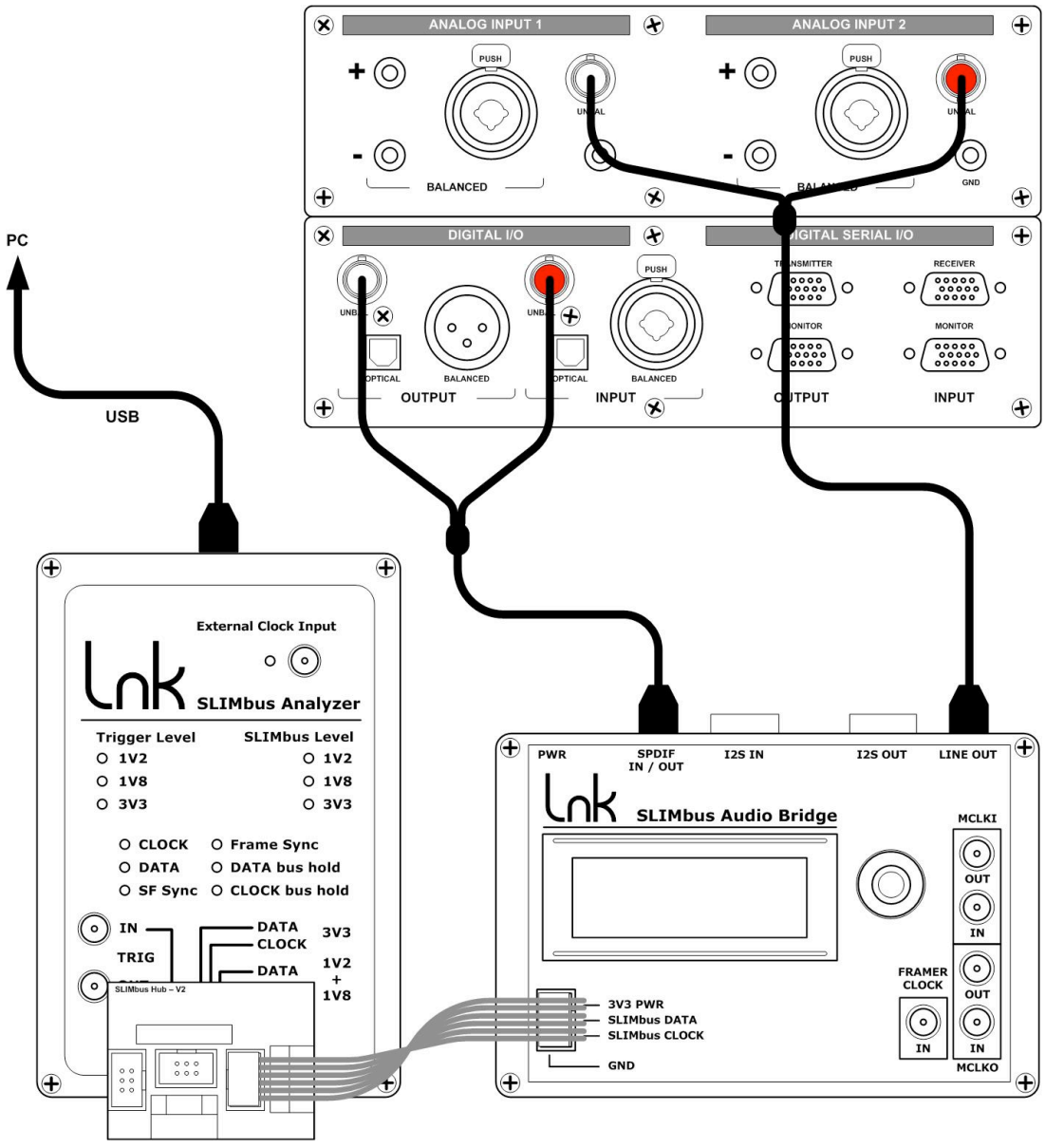
3. SPDIF IN / OUT data streaming

The purpose of this test is to verify that data streaming through SPDIF is functional.

3.1. System Setup

The following figure shows the connections between the different devices:

- The Audio Bridge and the Protocol Analyzer are connected together by the mini hub and a 6 way flat cable (delivered with the kit).
- The SPDIF IN/OUT interface is connected to the audio analyzer digital unbalanced I/Os with a stereo “jack 3.5mm to RCA” cable. The white RCA plug is connected to the SPDIF output of the audio analyzer. The red RCA plug is connected to the SPDIF input of the audio analyzer.
- The Bridge analog line out is connected to the analog inputs of the audio analyzer with a stereo “jack 3.5mm to RCA” cable.
- The Protocol Analyzer is connected to the PC with a USB cable. It is USB powered.
- The Audio Bridge needs its own power supply (DC 7V to 12V).
- If the bridge is an 8 port model, it needs to be connected to the PC with a USB cable.



3.2. Audio Bridge configuration

To perform the test, the bridge must be configured according to the following table. First power ON the bridge and use the rotary knob to display the various parameter pages of the bridge.

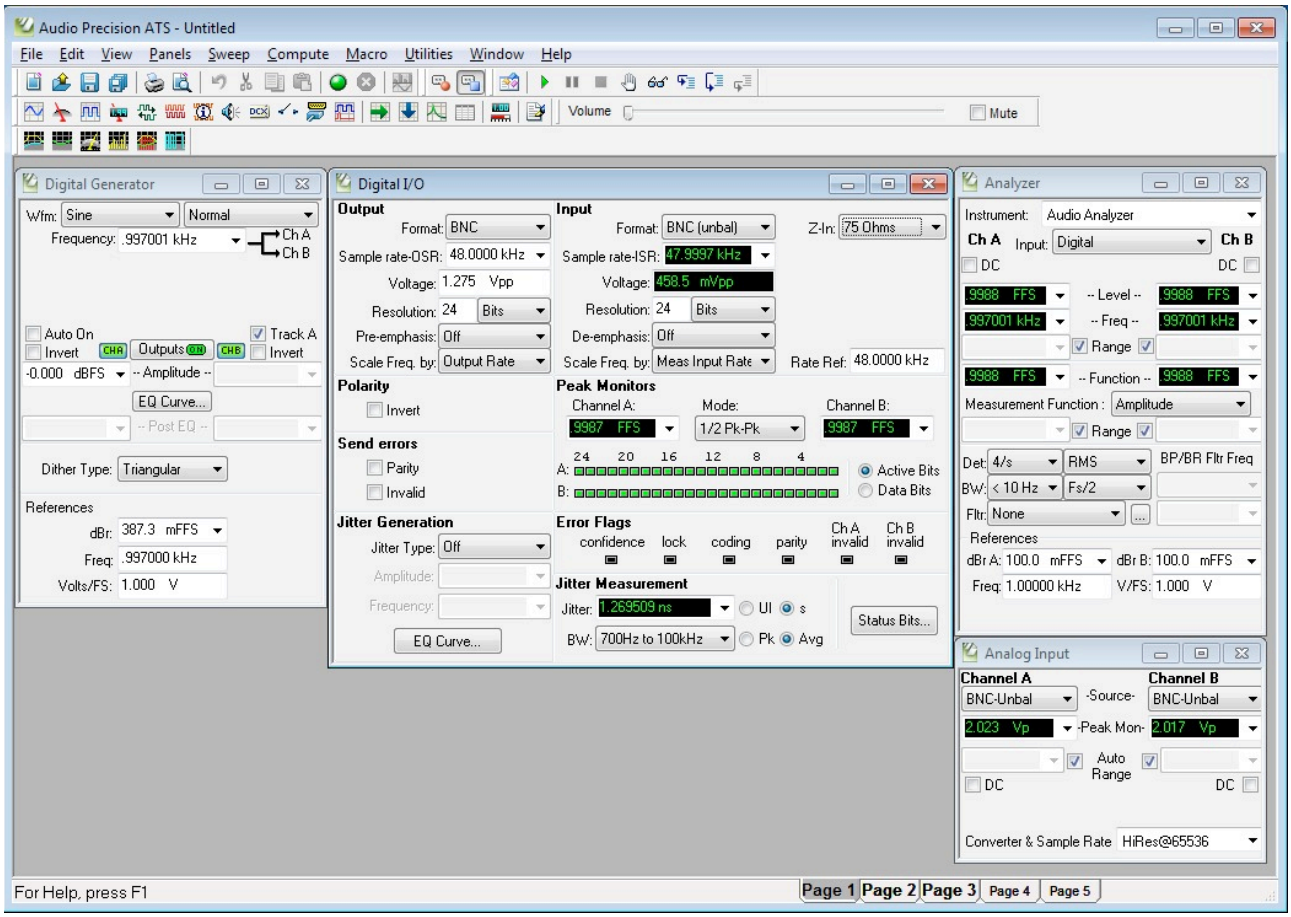
To edit a parameter, proceed as follow:
- Press the rotary knob to enter the parameter selection menu.
- Turn the knob to reach the desired parameter.
- Press the knob to enter the parameter value selection menu.
- Turn the knob to select the appropriate value for the parameter.
- Press the knob to validate the value and exit the sub menu.
- Turn the knob to reach “Exit” either at the top or the bottom of the parameter list.
- Press the knob to exit the parameter selection menu.

MCLKI (audio master clock if the I2S input interface) source must be set to PLL. That means that the audio input sample clock will be locked to the SLIMbus clock.	1. Audio Input Clk MCLKI Src: PLL Freq: 0.000 kHz Ferr: 0 Hz.CGmin=1
MCLKO (audio master clock if the I2S ouptut inerface) source must be set to PLL. That means that the audio output sample clock will be locked to the SLIMbus clock.	2. Audio Output Clk MCLKO Src: PLL Freq: 0.000 kHz Ferr: 0 Hz.CGmin=1
The SLIMbus Audio Bridge Framer must be deactivated (read “Framer:OFF”)	——————>——————>SB>—————— P01234567 OPR=n/a ————————— IPR=n/a Framer:Off.RF=0
The SLIMbus Audio Bridge bus holder must be activated. The SLIMbus signaling level must be set to 1.8V. The SLIMbus Audio Bridge component address must be set to 0.	5. SLIMbus PHY Bus Holder : On SLIMbus Level: 1.8 V Comp. Address: 0
The SPDIF Input Asynchronous Sample Rate Converter (ASRC) must be activated	S/PDIF Input ASRC Disabled >Enabled <

3.3. Audio Analyzer configuration

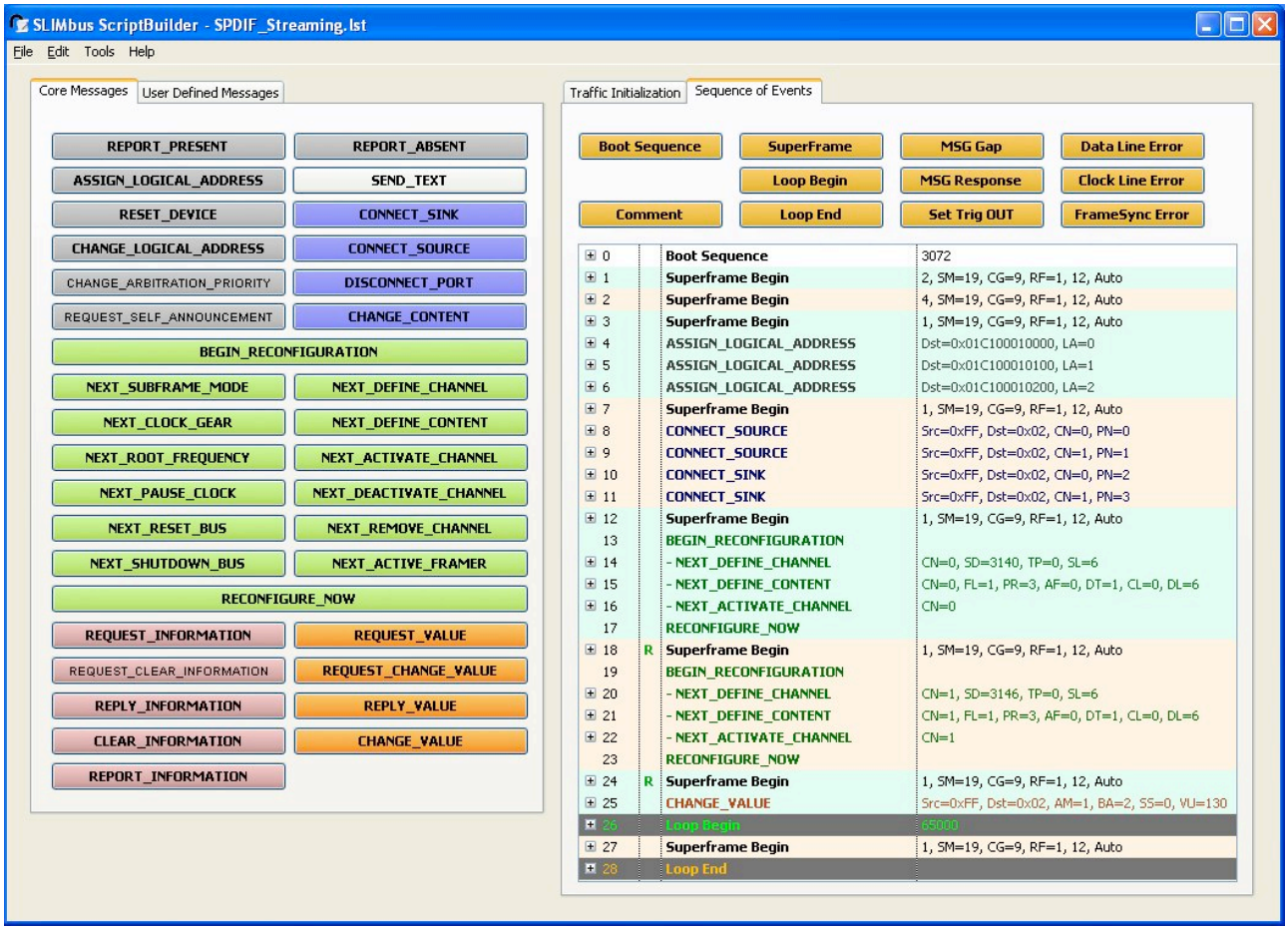
Activate the SPDIF output and generate a sine wave at 1 kHz, 0 dBFs.
Set the digital input on the unbalanced BNC connector.
Set the analog inputs on the unbalanced BNC connectors.
Connect the analyzer to the incoming digital stream.

The measurements were made with an Audio Precision ATS2. Any other audio analyzer will also work, assuming the same kind of setup can be achieved.



3.4. Traffic Generator configuration

Launch the SLIMbus Protocol Analyzer application.
Launch the ScriptBuilder application.
Load the script “SPDIF_Streaming.lst” in ScriptBuilder.



Push the script to the Traffic Generator by either hitting Ctrl+T or by going into the menu “**File / Send Script To Traffic Generator Ctrl+T**”.

In the Protocol Analyzer main window:
- Click the “MSG” view button
- Click on the “MSG only” tick box
- Click on the “LiveView” button
- Click on the “PLAY” button

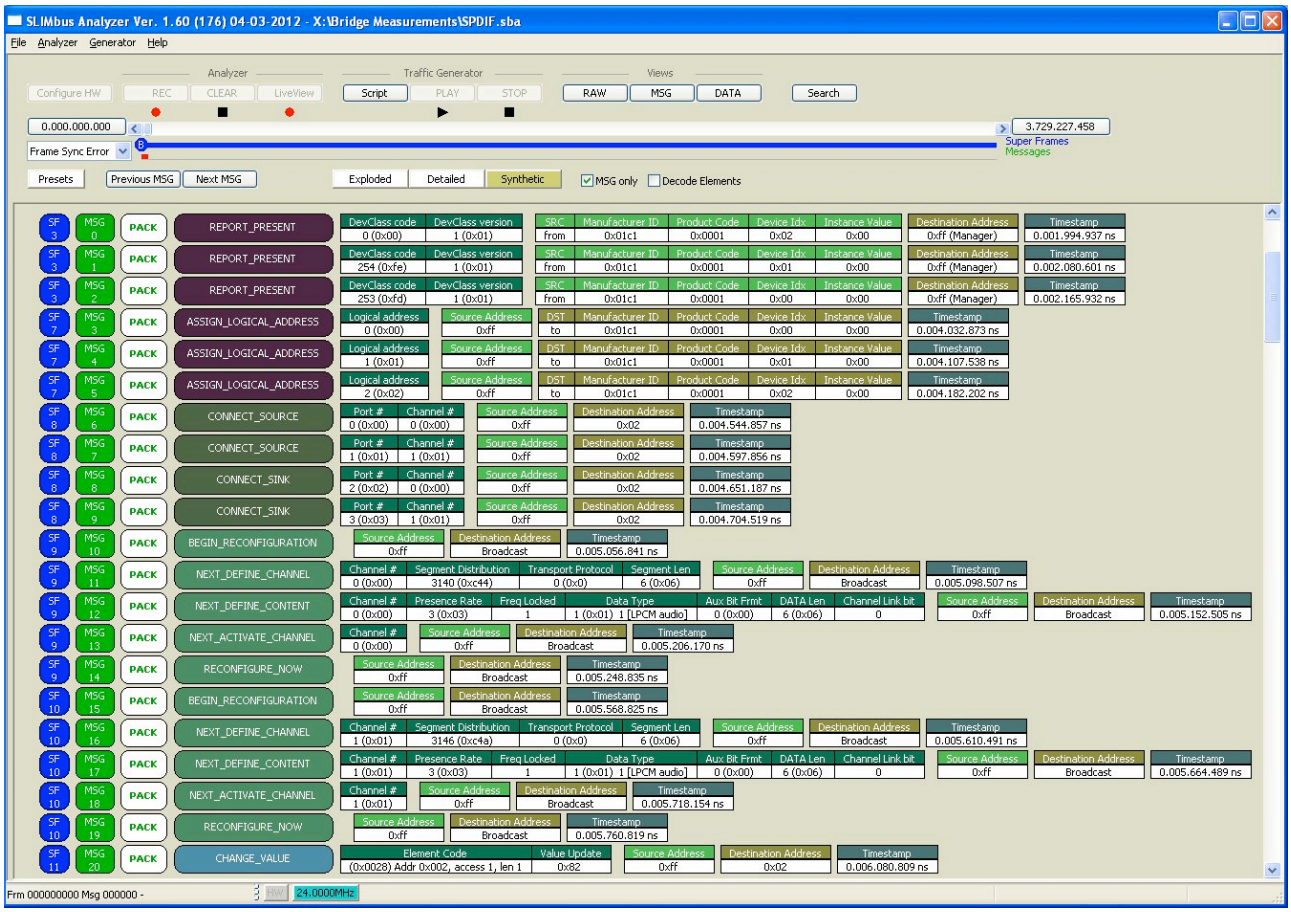
Note: When the test is finished, stop the Traffic Generator and Protocol Analyzer recording by clicking on the “STOP” buttons

---

The script is transmitted on SLIMbus and configures the Audio Bridge to establish audio streaming. 2 data channels are created. On the bridge display main page, the following shall appear:

```
SPDIF>ASRC24>SB>SD02
P0123456 OPR=48k
 OOII---- IPR=48k
Framer:Off,RF=1
```

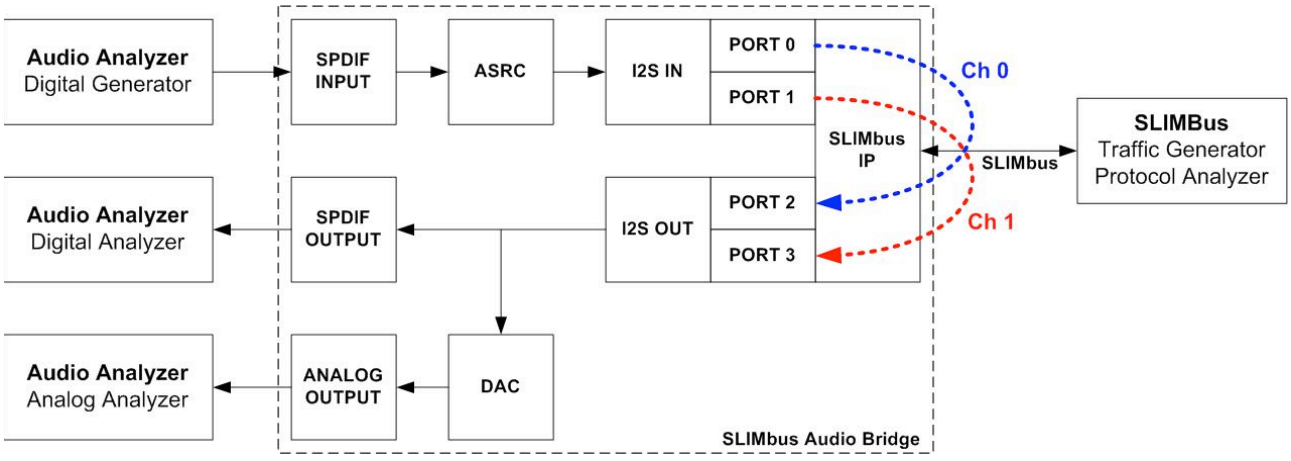
The LiveView capture of the Protocol Analyzer shall show the following list of messages.



If not stopped, the script will keep playing forever and the SLIMbus data channels are usable for any kind of tests.

3.5. Data path

The script has created the following data paths:



Channel 0 is linking Port 0 (source) to Port 2 (sink). The samples are 24bits @ 48 kHz.
Channel 1 is linking Port 1 (source) to Port 3 (sink). The samples are 24bits @ 48 kHz.
The ASRC is configured to output 24 bit samples (meaning of “ASRC24”).
The DAC and SPDIF out are configured to use the Port 2 and 3 serial data.

3.6. Script file analysis

Let's now have a look to the various parts of the script to see what has been done.

3.6.1. Device Enumeration

In the first part of the script, we have left four empty superframes to let the SLIMbus devices of the Audio Bridge to report present. Then, we assigned a logical address to each of them. The Interface device gets the address 0, the Framer device gets the address 1 and the Generic device (the one with the data ports) gets the address 2.

⊞ 2	**Superframe Begin**	4, SM=19, CG=9, RF=1, 12, Auto
⊞ 3	**Superframe Begin**	1, SM=19, CG=9, RF=1, 12, Auto
⊞ 4	**ASSIGN_LOGICAL_ADDRESS**	Dst=0x01C100010000, LA=0
⊞ 5	**ASSIGN_LOGICAL_ADDRESS**	Dst=0x01C100010100, LA=1
⊞ 6	**ASSIGN_LOGICAL_ADDRESS**	Dst=0x01C100010200, LA=2

When the script is played, we can see with the Analyzer that there has been 3 REPORT_PRESENT messages sent by the devices of the Audio Bridge.

MSG 0	PACK	REPORT_PRESENT	DevClass code	DevClass version		SRC	Manufacturer ID	Product Code	Device Idx	Instance Value		Destination Address	Timestamp						
			0 (0x00)	1 (0x01)		from	0x01c1	0x0001	0x02	0x00		0xff (Manager)	0.000.000.000 ns						
MSG 1	PACK	REPORT_PRESENT	DevClass code	DevClass version		SRC	Manufacturer ID	Product Code	Device Idx	Instance Value		Destination Address	Timestamp						
			254 (0xfe)	1 (0x01)		from	0x01c1	0x0001	0x01	0x00		0xff (Manager)	0.000.124.121 ns						
MSG 2	PACK	REPORT_PRESENT	DevClass code	DevClass version		SRC	Manufacturer ID	Product Code	Device Idx	Instance Value		Destination Address	Timestamp						
			253 (0xfd)	1 (0x01)		from	0x01c1	0x0001	0x00	0x00		0xff (Manager)	0.000.263.757 ns						
MSG 3	PACK	ASSIGN_LOGICAL_ADDRESS	Logical address		Source Address	DST	Manufacturer ID	Product Code	Device Idx	Instance Value		Timestamp							
			0 (0x00)		0xff	to	0x01c1	0x0001	0x00	0x00		0.002.000.000 ns							
MSG 4	PACK	ASSIGN_LOGICAL_ADDRESS	Logical address		Source Address	DST	Manufacturer ID	Product Code	Device Idx	Instance Value		Timestamp							
			1 (0x01)		0xff	to	0x01c1	0x0001	0x01	0x00		0.002.108.605 ns							
MSG 5	PACK	ASSIGN_LOGICAL_ADDRESS	Logical address		Source Address	DST	Manufacturer ID	Product Code	Device Idx	Instance Value		Timestamp							
			2 (0x02)		0xff	to	0x01c1	0x0001	0x02	0x00		0.002.217.210 ns							

When all ASSIGN_LOGICAL_ADDRESS messages have been sent and properly received ("PACK" = **P**ositive **ACK**nowledgment), the devices are ready for configuration.

3.6.2. Defining the data port function

The next part of the script defines how the data ports of the Generic device are going to be used.
The port 0 and the port 1 will be used as data source on SLIMbus. This choice is not arbitrary. The SPDIF input is mapped on these 2 ports. Port 0 handles the SPDIF channel B and port 1 handles the SPDIF channel A.

⊞ 8	**CONNECT_SOURCE**	Src=0xFF, Dst=0x02, CN=0, PN=0
⊞ 9	**CONNECT_SOURCE**	Src=0xFF, Dst=0x02, CN=1, PN=1
⊞ 10	**CONNECT_SINK**	Src=0xFF, Dst=0x02, CN=0, PN=2
⊞ 11	**CONNECT_SINK**	Src=0xFF, Dst=0x02, CN=1, PN=3

Ports 2 and 3 get assigned a sink role. They will take the data from SLIMbus and push them in the I2S OUT interface of the bridge, in the I2S output stream 2 ("SDO2").
Port 0 and port 2 are using the channel 0. Port 1 and port 3 are using the channel 1.

3.6.3. Channel definition

The SLIMbus data channels are built in such a way that there are 48000 segments per second, each of them carrying an audio sample. The segment size is 6 slots (24 bits). A channel is defined and activated by 5 messages (called a Reconfiguration Sequence).

```
13  BEGIN_RECONFIGURATION
14  - NEXT_DEFINE_CHANNEL      CN=0, SD=3140, TP=0, SL=6
15  - NEXT_DEFINE_CONTENT      CN=0, FL=1, PR=3, AF=0, DT=1, CL=0, DL=6
16  - NEXT_ACTIVATE_CHANNEL    CN=0
17  RECONFIGURE_NOW
```

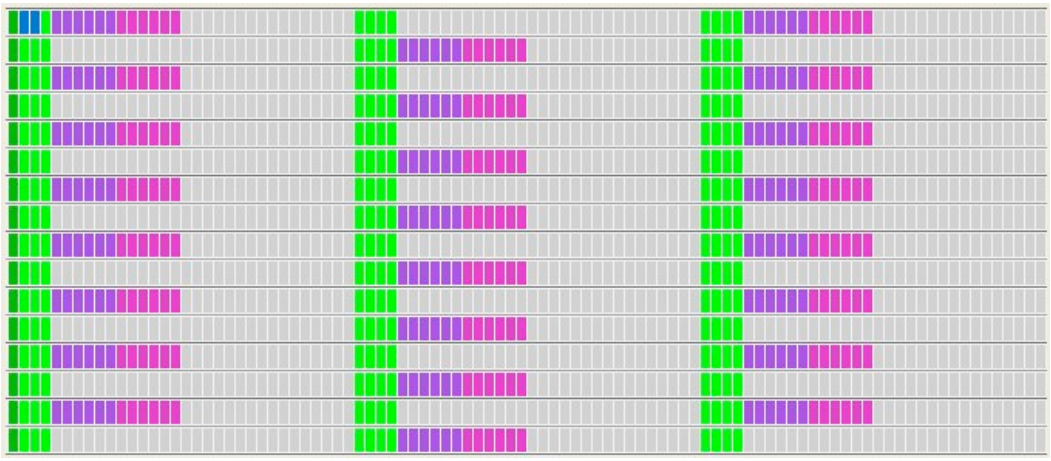
NEXT_DEFINE_CHANNEL specifies the structure of the data channel.

- **CN** is the channel number to be affected by the definition.
- **SD** (Segment Distribution) sets the number of slots between two consecutive segments of a give channel. It also defines where is happening the first segment of a channel in a superframe (Segment Offset). The SD value is computed by ScriptBuilder, based on the channel rate. The user can also refer to the SLIMbus specification about the way SD is coded. In our script, the bus clock is equal to 12,288 MHz. SD=3140 means a repeat interval of 64 slots and an offset of 4 slots, leading to 48000 segments per second.
- **TP** is the transport protocol. In our script, the number of segment equals the number of samples. We use TP=0 (Isochronous protocol).
- **SL** is the segment size. As we want to carry 24 bits samples, we use SL=6 slots (of 4 bits).

NEXT_DEFINE_CONTENT specifies the nature of the information carried in the channel.

- **CN** is the channel number to be affected by the definition.
- **FL** indicates that the frequency of the channel and the sample rate frequency are phase locked.
- **PR** is the Presence Rate. It is equivalent to the sample rate. PR=3 means that the sample rate in use in the channel is equal to 48 kHz.
- **DT** is the Data Type. DT=1 means that we carry L-PCM audio samples.
- **CL** is not used.
- **DL** is the Data Length. DL=6 means that the usable part of the segment for data is 6 slots or 24 bits (the actual size of the segment). **DL will affect the ASRC setting**. If DL was set to 5, the ASRC would have been programmed to output 20 bits samples (5 slots long).

NEXT_ACTIVATE_CHANNEL will tell all the device using the channel **CN** to start operation with the channel.



Segment repartition in 1 superframe

3.6.4. Selecting the right SPDIF OUT stream

By addressing a specific register of the Generic device through SLIMbus, we can actually select which of the output streams shall go to the DAC and the SPDIF output. The bridge can have up to 4 stereo streams.

Port 2 and 3 are mapped on the stream 2 so we need to route the stream 2 to the DAC and SPDIF output.

⊞ 25	CHANGE_VALUE	Src=0xFF, Dst=0x02, AM=1, BA=2, SS=0, VU=130

Loading the value 130 (VU=130) in the register address 2 (BA=2) of the Generic device is forcing the stream 2 to go to the DAC and SPDIF output. The Audio Bridge user manual gives all the details about the bit meaning of the Generic device registers.

3.6.5. Never-ending streaming

Now that the data channels are defined and usable, the Traffic Generator shall keep clocking the bus for ever to allow various kind of audio tests.
This is achieved by looping some parts of the script in the hardware.

⊞ 26	Loop Begin	65000
⊞ 27	Superframe Begin	1, SM=19, CG=9, RF=1, 12, Auto
28	Loop End	

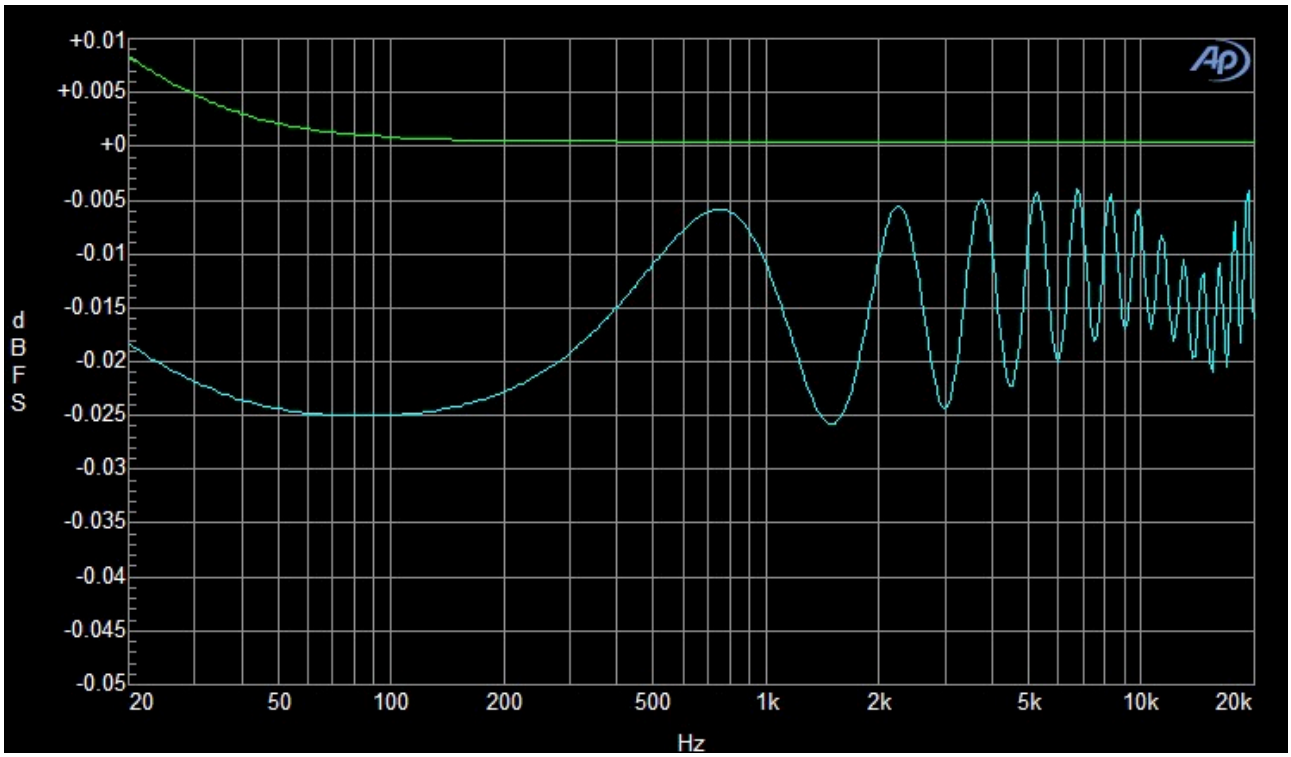
This is looping 1 complete superframe forever. Any number bigger than 32768 (and lower than 65535) in the Repeat parameter of the Loop event will trigger a forever loop.

3.7. Characterization of the audio performances of the Audio Bridge

SLIMbus is running, the Audio Bridge is configured with 2 channels. We are ready to run some audio measurements.
Obviously, in our case, we will only use the digital generator of the audio analyzer. The Audio Bridge does not have analog inputs.
The Audio Bridge as such does not have any influence on the data that are getting transmitted over SLIMbus. However, the input asynchronous sample rate converter (ASRC), when it is used, has some very minor but measurable effects.

3.7.1. In-band frequency response.

Even if the ASRC is of very high quality, it is not 100% transparent. So it is possible to measure its contribution to the data loop.

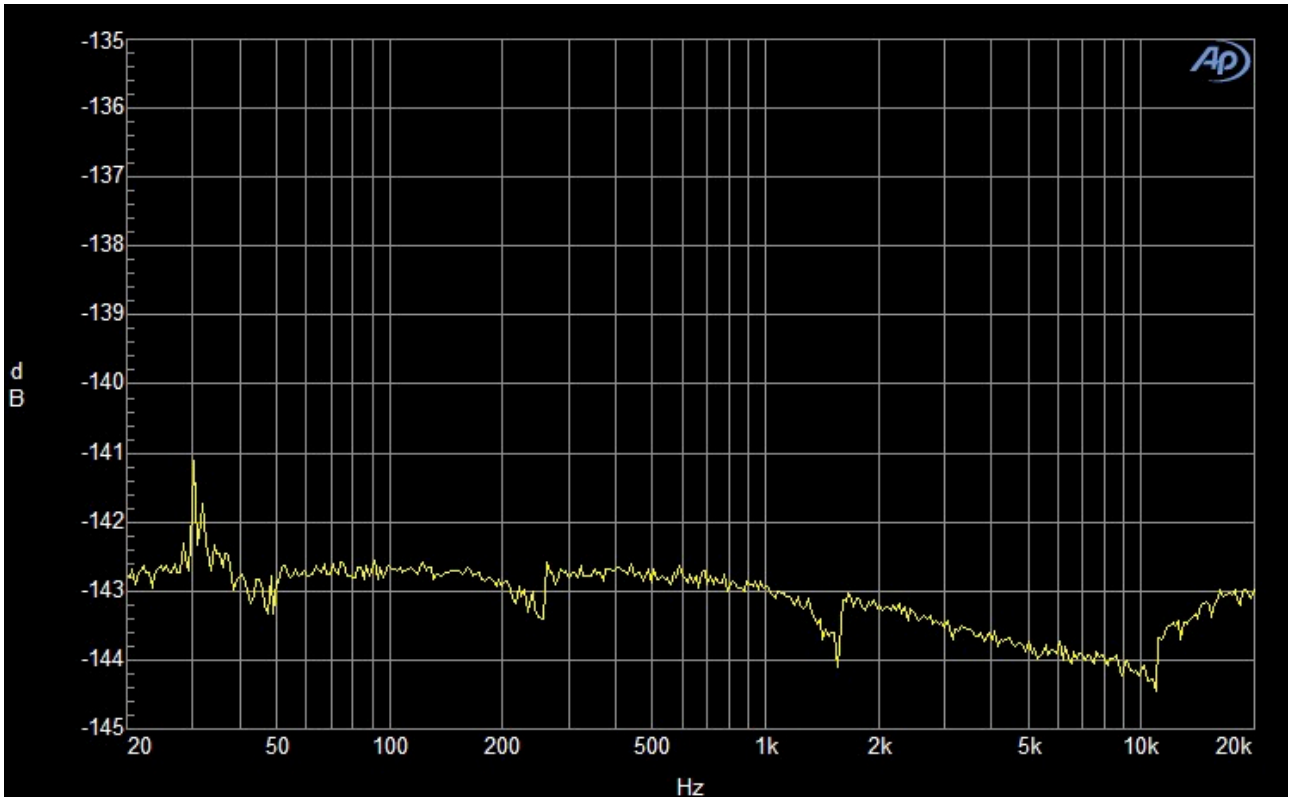


The blue curve is the contribution of the ASRC. For reference, the Audio Precision ATS2 contribution has been measured (in green). The in-band ripple introduced by the ASRC is +/- 0.01 dB. This measurement is valid for a conversion ratio equal to 1 (48k:48k).

3.7.2. THD+N vs. Frequency measurements

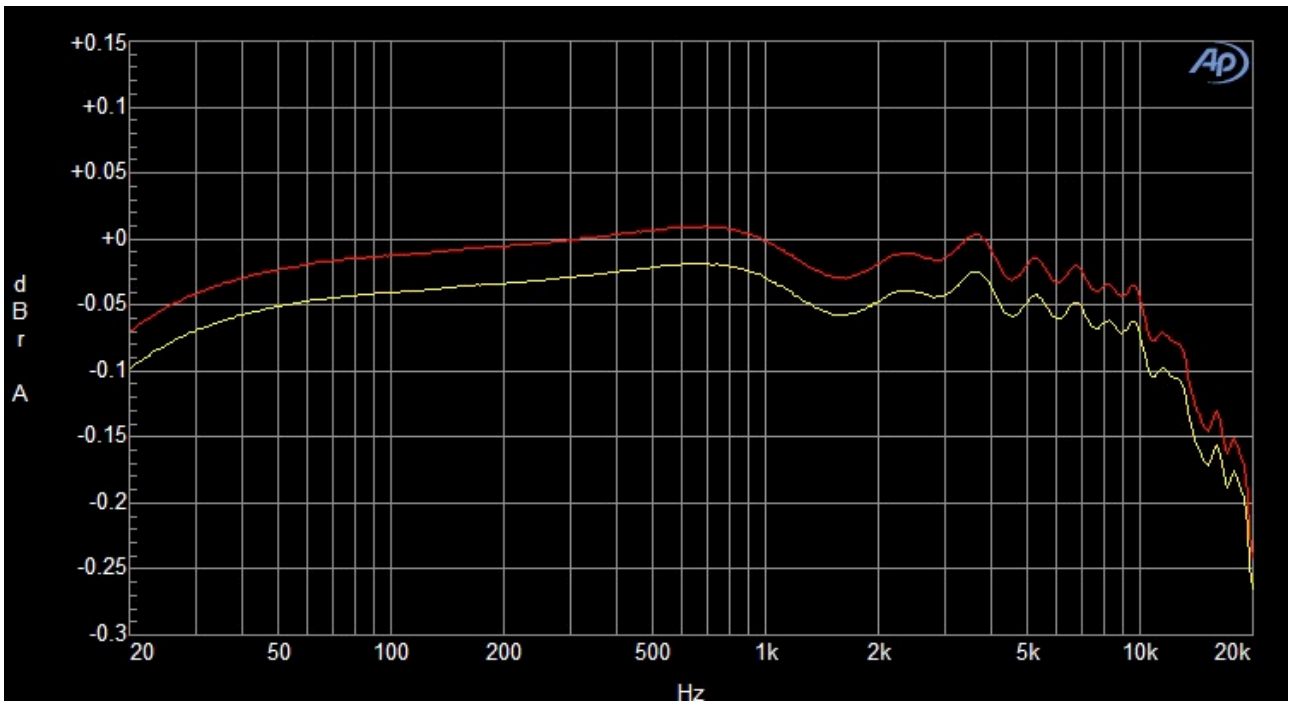
Similarly, it is possible to characterize the THD+N contribution of the ASRC. The following measurement is valid for a conversion ratio of 1 (48k:48k).
The measured value is below -142 dB. This shall be enough not to disturb any THD+N measurements of D-to-A converters.

Note: These values are measured with the full signal bandwidth and with A-weighting filter. The Audio Precision test signal dithering has been disabled. When not using A-weighting and enabling the triangular dithering, the measured value is below **-137 dB**.



3.7.3. DAC Frequency response

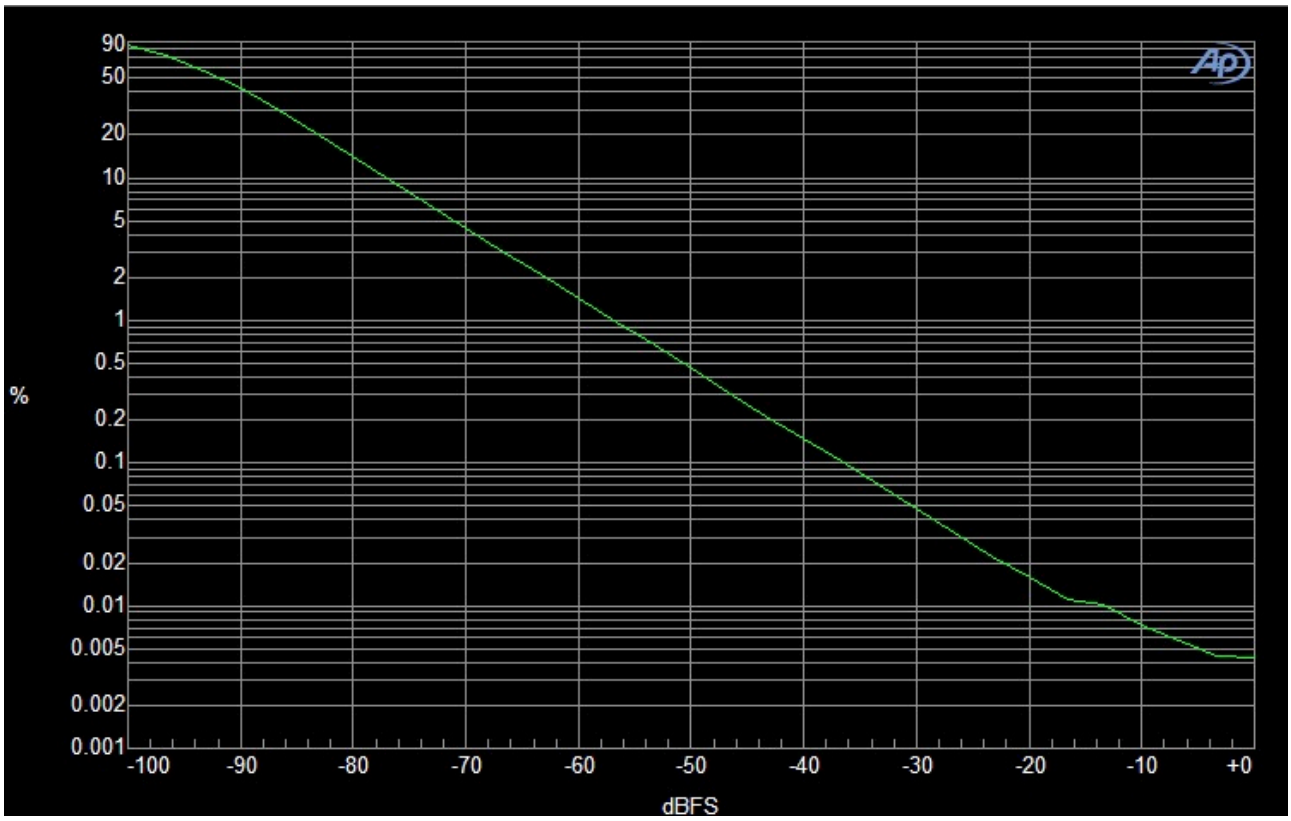
Some interesting measurements can be also done on the analog monitoring output of the Audio Bridge. This output can be seen as a SLIMbus D-to A component. All the measurements that can be done of the Audio Bridge DAC can also be done on any external SLIMbus audio codec. As such, it is very instructive to learn how to best use the tools.



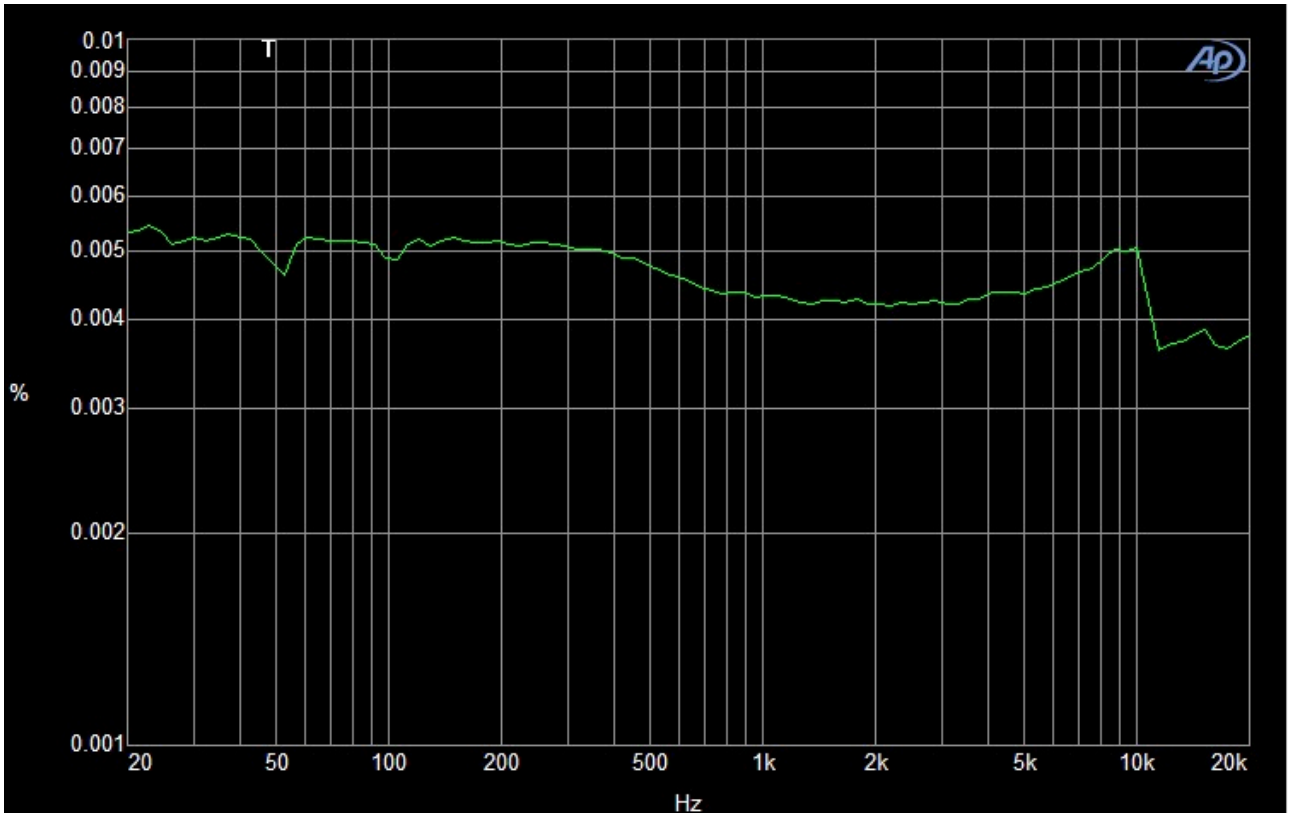
The 2 channels were measured simultaneously. We can see that there is an offset of 0.02 dB between the two analog outputs. The ripple we measure is due to the decimation filter

of the DAC, the 1st order HPF and the 4th order LPF of the analog filter, plus the contribution of the ASRC.

3.7.4. DAC THD+N vs. Amplitude



3.7.5. DAC THD+N vs. Frequency



All in all, it is a pretty decent DAC implementation.

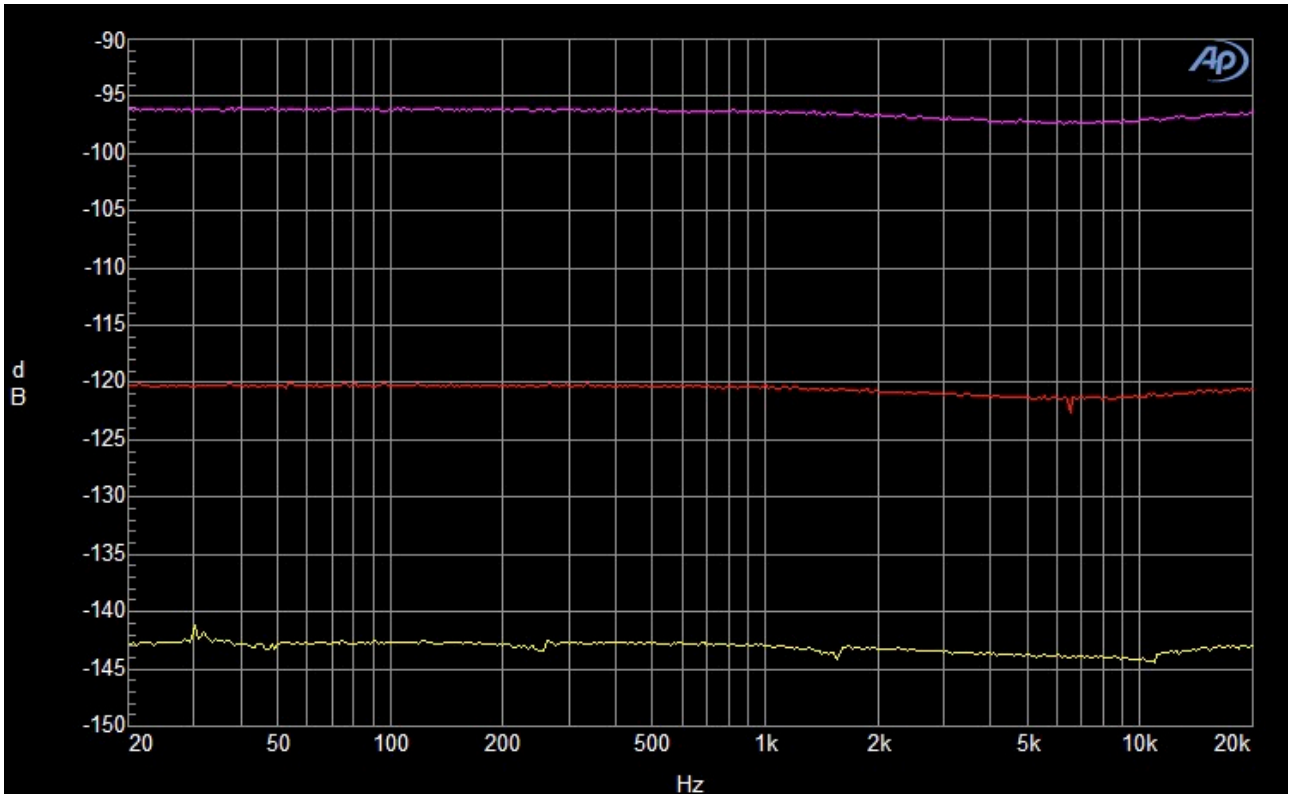
3.8. Modifying the script parameters

Now, we can experiment a bit on the script parameters.

3.8.1. Changing the sample size

The most obvious candidate for change is the sample size. We can make it 20 bits (5 slots) or 16 bits (4 slots) by changing the **DL** value in the NEXT_DEFINE_CONTENT messages. We have to do it for the 2 channels if we want a stereo measurement.

The ASRC THD+N vs. Frequency performances were measured for 16, 20 and 24 bits sample/segment size. The curves are in accordance with what we can expect (about 24 dB difference between each curves).



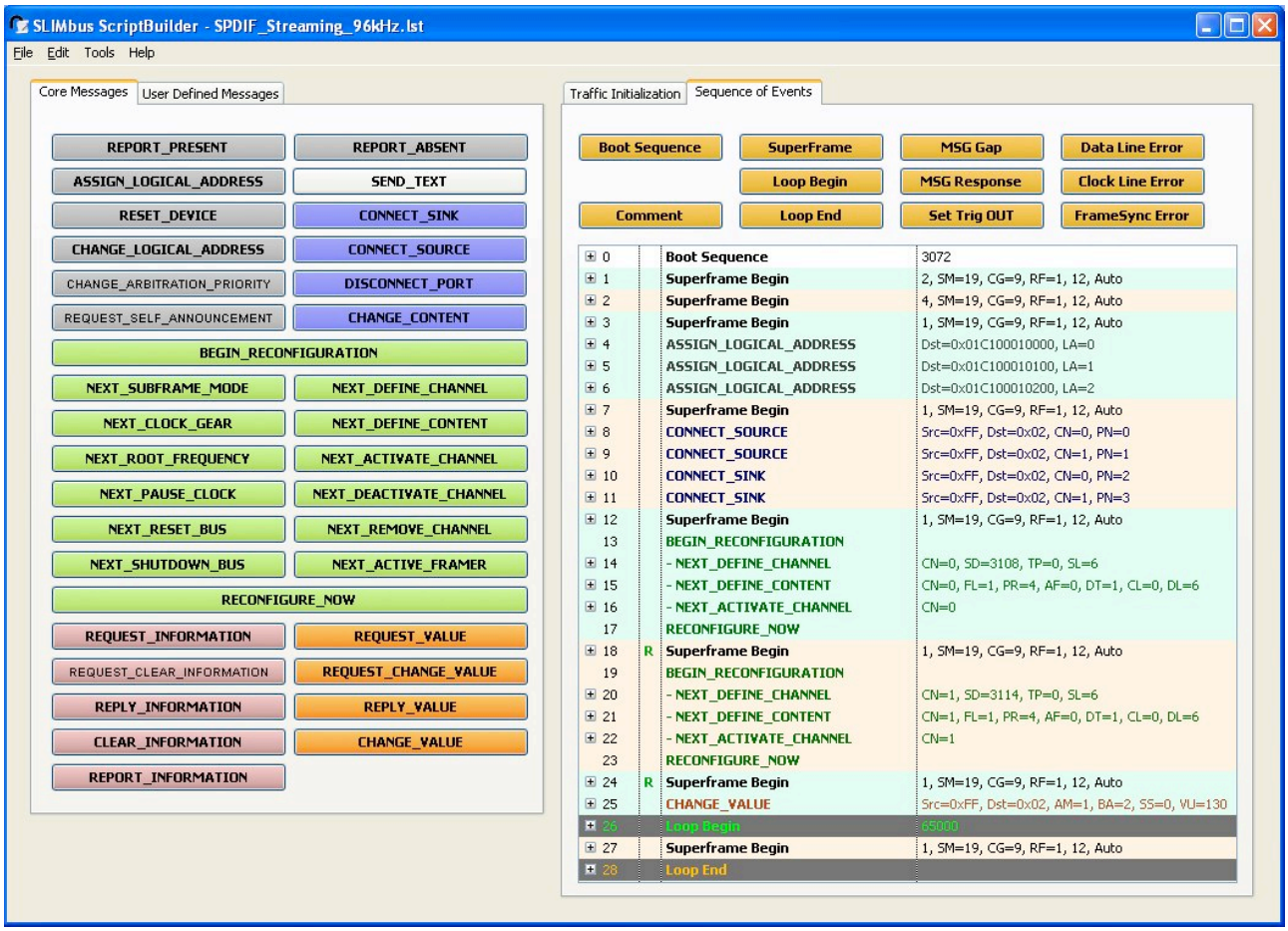
The purple curve corresponds to a sample size of 16 bits.
The red curve corresponds to a sample size of 20 bits.
The yellow curve corresponds to a sample size of 24 bits.
The measurements conditions are the same as before.

3.8.2. Changing the channel rate (and sample rate)

We can modify the channel rate to have 96 kHz instead of 48 kHz.
However, this has little visible effects. It can be seen on the ASRC in-band ripple that is getting stretched.

The parameter **SD** of the NEXT_DEFINE_CHANNEL must be changed for both channels. Instead of 3140, channel 1 will use 3108. Channel 2 will use 3114. This is setting the channel rate to 96kHz. Rely on ScriptBuilder to compute these new values.

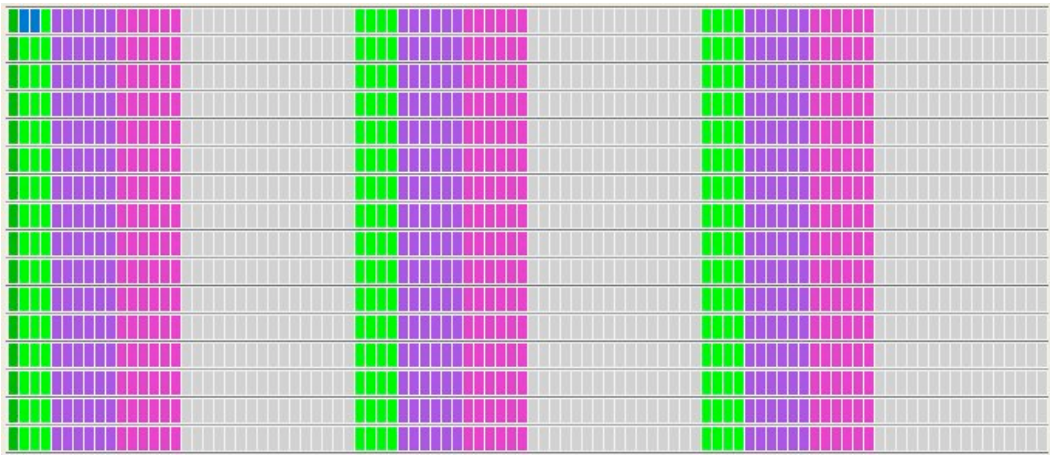
However, this is not sufficient to have a valid 96 kHz streaming. We also need to change the **PR** parameter in the NEXT_DEFINE_CONTENT messages. It will take the value 4 that corresponds to 96 kHz (PR=4).



The new script is named “SPDIF_Streaming_96kHz.lst”.
When playing this modified script, the Bridge display will reflect the new channel rate.

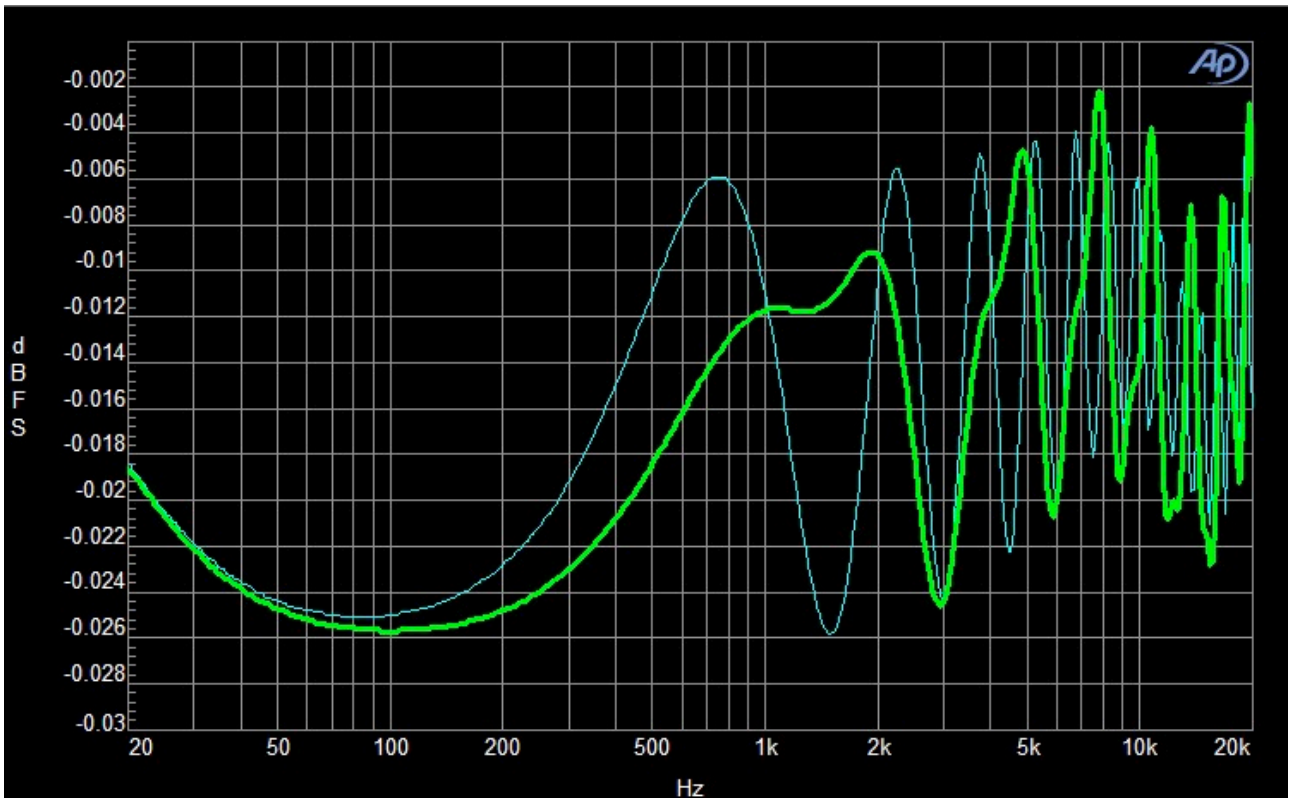
```
SPDIF>ASRC24>SB>SD02
P01234567 OPR=96k
  OOII---- IPR=96k
Framer:Off,RF=1
```

We also see that we have now twice the number of segment in a superframe.



Segment repartition in 1 superframe

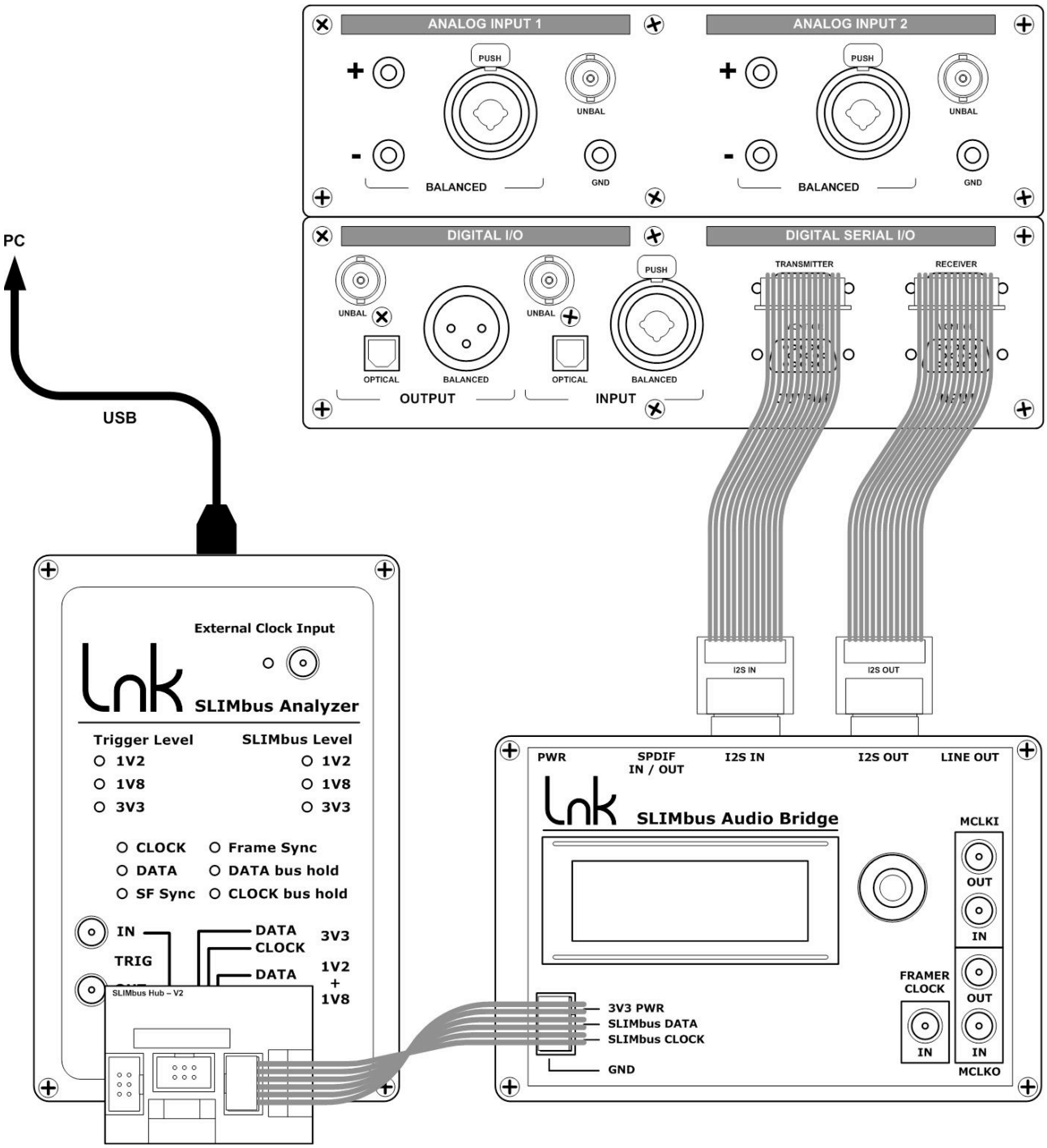
The green curve shows the ASRC ripple with channels running at 96 kHz and and SPDIF audio stream at 48 kHz. The blue curve is for a channel rate at 48 kHz with a SPDIF stream at 48 kHz.



4. I2S IN / OUT data streaming

The tests we have been running with the SPDIF input / output can also be performed with the I2S multichannel input and output. We will mainly focus on the digital part of the tests.

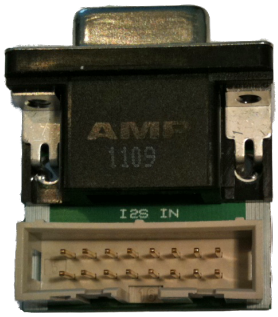
4.1. System Setup



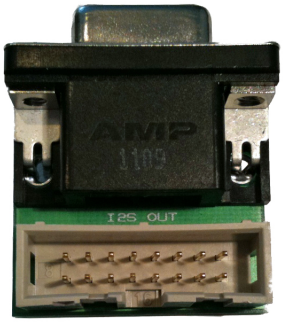
A pair of I2S cables and adapters are delivered with the bridge to ease connection to a variety of audio analyzers. However, when it is about testing I2S multichannel, the best option is to use an Audio Precision APx585 or APx525 model with the DSIO option.

There is a small difference between the I2S IN and I2S OUT cables. Do not exchange them. The streaming won't work anymore (swap of the bit clock and word clock).

I2S cable assembly

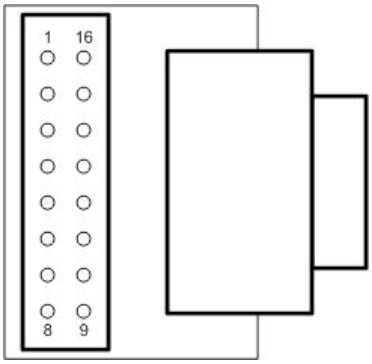


I2S IN adapter



I2S OUT adapter

I2S DSIO adapter



The I2S IN & OUT adapter pin assignment is as follow:

1 - BCLKI (Bit clock at 64Fs)
2 - MCLKI (Master clock at 128Fs or 256Fs)
3 - GND
4 - SDATA1 (Serial data stream 1)
5 - SDATA2 (Serial data stream 2)
6 - SDATA3 (Serial data stream 3)
7 - SDATA4 (Serial data stream 4)
8 - LRCLKI (Word or Sync clock at 1Fs)
9..16 - GND

To test the I2S data paths, we will use the fact that one I2S stream carries two audio streams. Therefore, it is sufficient to feed only one audio channel in the I2S stream to verify that the signal goes through.

4.2. Audio Bridge configuration

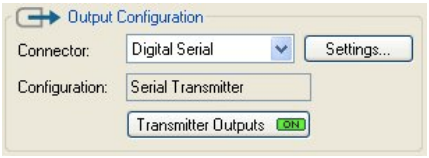
The bridge configuration is the same as for the previous test. Just make sure that the I2S input ASRC is enabled.

The I2S IN Asynchronous Sample Rate Converter must be activated	I2S Input ASRC Disabled >Enabled <

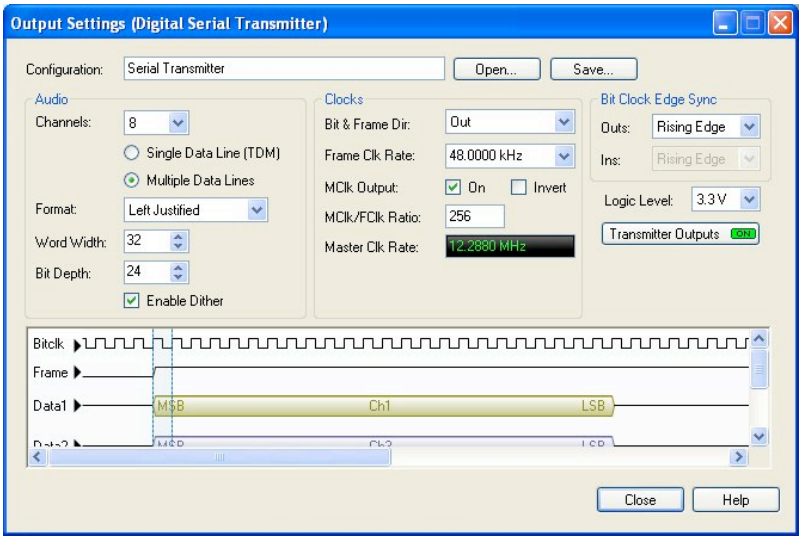
4.3. Audio Analyzer configuration

The configuration shown is valid for an Audio Precision APx585 analyzer. However, the settings are applicable for all kind of tools.

In the main software panel, the output configuration shall be set to “**Digital Serial**”. This is selecting the I2S multichannel interface.

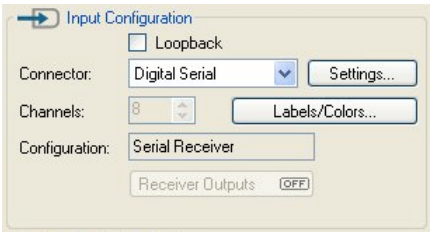


Access the settings to define the properties of the digital serial bit stream. The data format is “**Left Justified**”, the word width is 32 bits and the bit depth is 24 bits. Multiple data line output must be selected (not TDM).

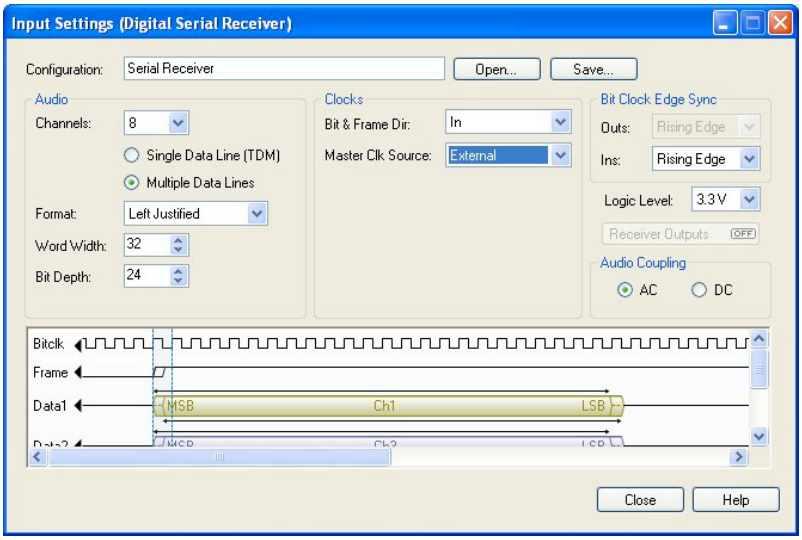


The Bit and Frame clock direction are set to Out.
The Bit Clock Edge Sync is set on the Rising Edge.
The signaling logic level is set on 3.3 Volts.

In the main software panel, the input configuration shall be set to “**Digital Serial**”. This is selecting the I2S multichannel interface.

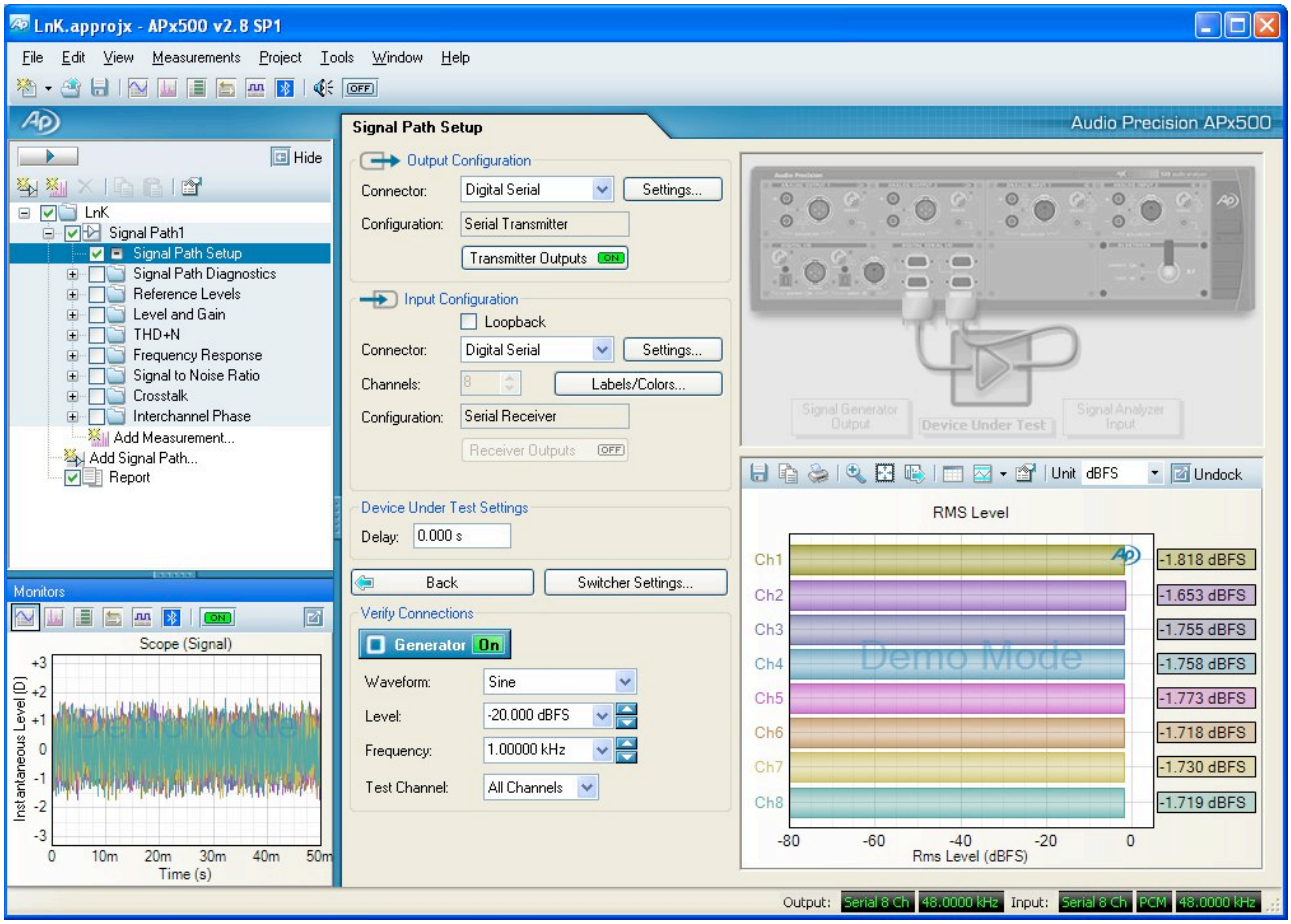


Access the settings to define the properties of the digital serial bit stream. The data format is “**Left Justified**”, the word width is 32 bits and the bit depth is 24 bits. Multiple data line output must be selected (not TDM).
The Bit and Frame clock direction are set to Out.
The Bit Clock Edge Sync is set on the Rising Edge.



The signaling logic level is set on 3.3 Volts.

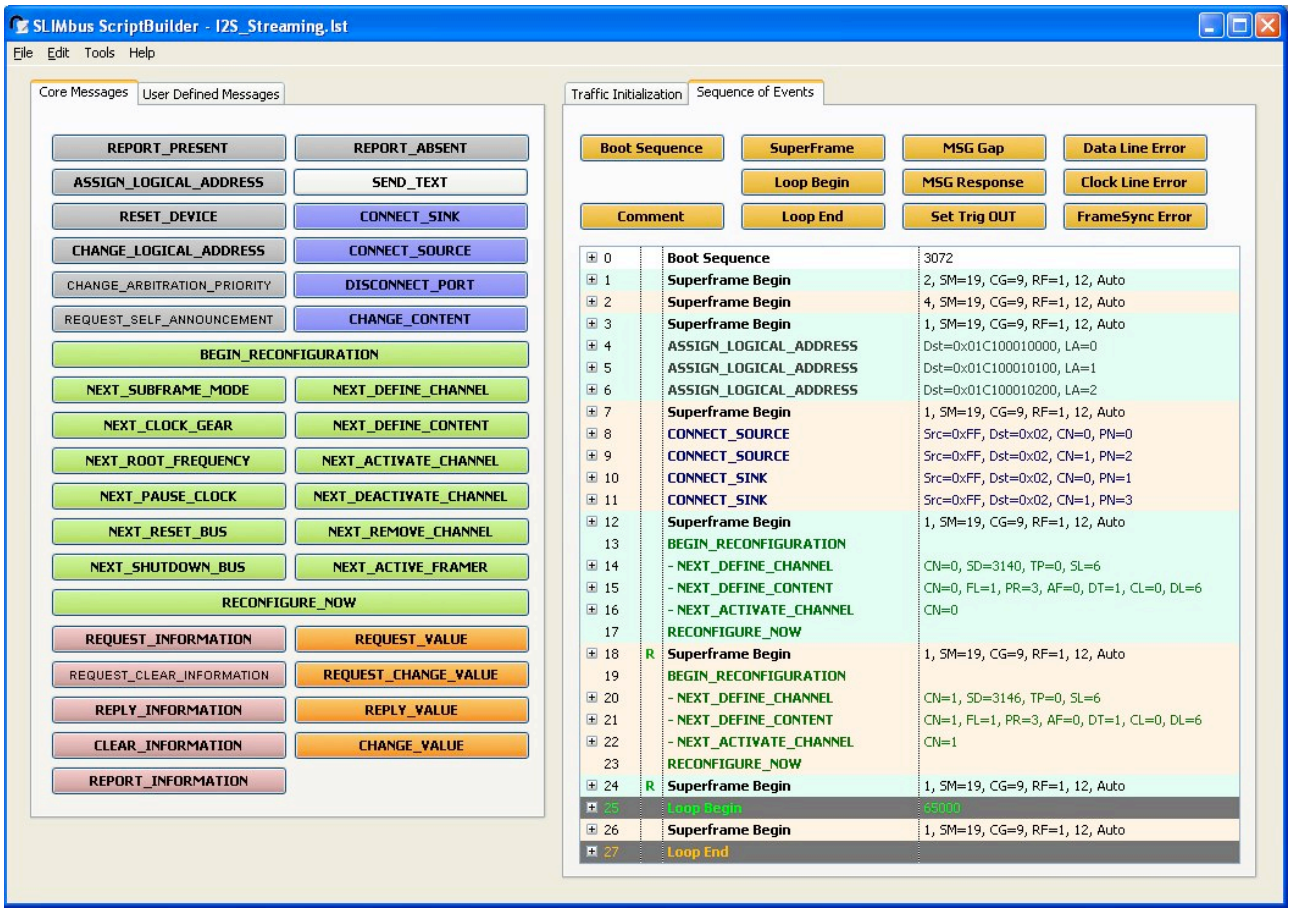
The Transmitter outputs and the Signal Generator must be “**On**”.



At that stage, the audio analyzer is outputting sine waves on all 8 channels of the Digital Serial interface.

4.4. Traffic Generator configuration

Launch the SLIMbus Protocol Analyzer application.
Launch the ScriptBuilder application.
Load the script “I2S_Streaming.lst” in ScriptBuilder.



Push the script to the Traffic Generator by either hitting Ctrl+T or by going into the menu “**File / Send Script To Traffic Generator Ctrl+T**”.

In the Protocol Analyzer main window:
- Click the “MSG” view button
- Click on the “MSG only” tick box
- Click on the “LiveView” button
- Click on the “PLAY” button

Note: When the test is finished, stop the Traffic Generator and Protocol Analyzer recording by clicking on the “STOP” buttons

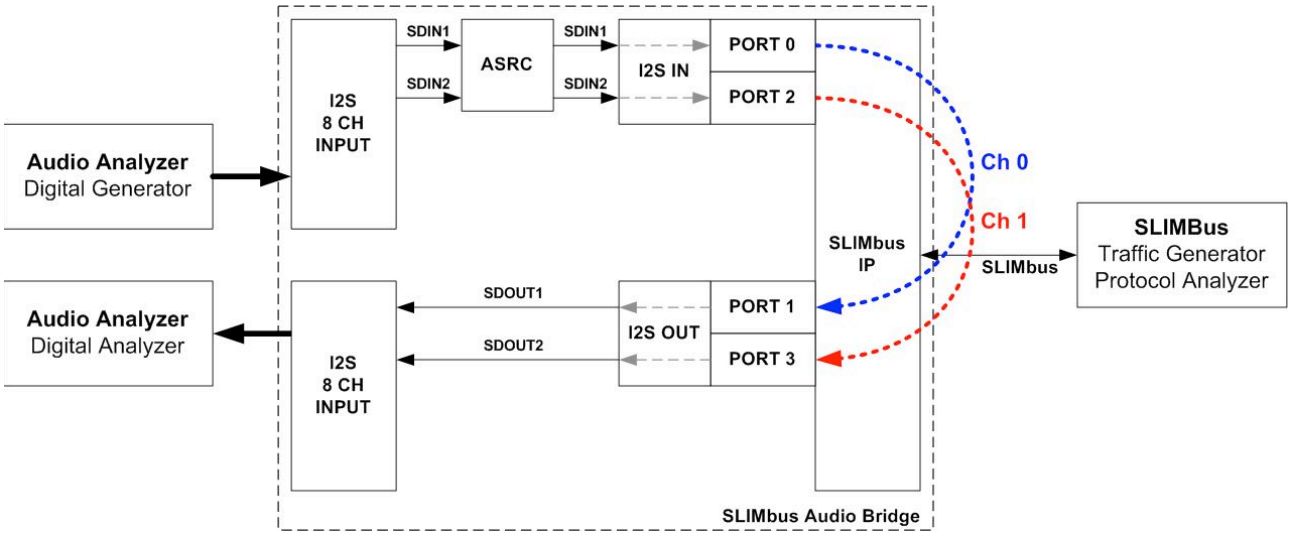
---

The script is transmitted on SLIMbus and configures the Audio Bridge to establish audio streaming. 2 data channels are created. On the bridge display main page, the following shall appear

```
  I2S  >ASRC24>SB>SD02
P01234567 OPR=48k
  IOIO---- IPR=48k
Framer:Off,RF=1
```

4.5. Data path

The script has created the following data paths:



Channel 0 is linking Port 0 (source) to Port 1 (sink). The Samples are 24bits @ 48 kHz.
Channel 1 is linking Port 2 (source) to Port 3 (sink). The Samples are 24bits @ 48 kHz.
The ASRC is configured to output 24 bit samples (meaning of “ASRC24”).

Port 0 will take the data of the SDIN1 stream, channel B.
Port 2 will take the data of the SDIN2 stream, channel B.
Port 1 will put the data in the SDOUT1 stream, channel A.
Port 3 will put the data in the SDOUT2 stream, channel A.

Note: This script is valid for both 4 ports and 8 ports bridges.

4.6. Script file analysis

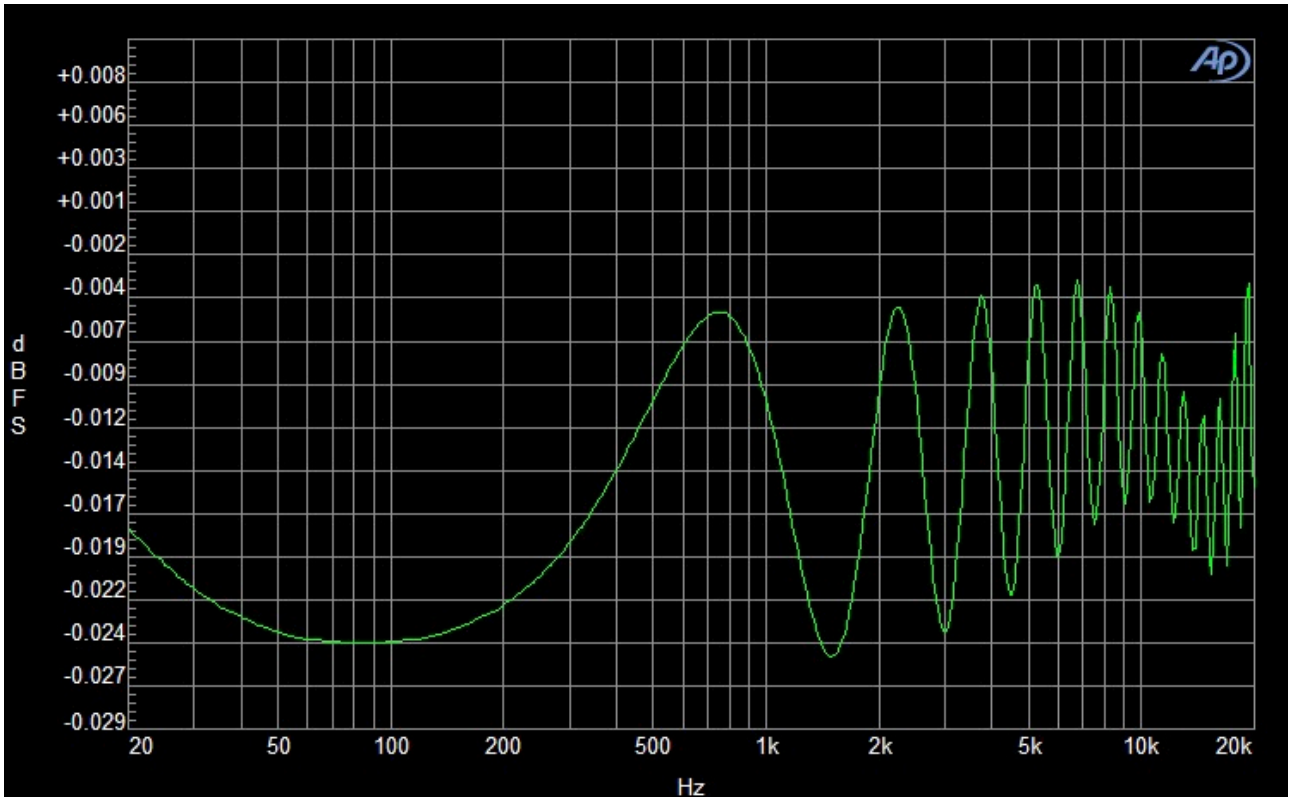
This script is very similar to the SPDIF script in its structure. The only difference is in the port to channel assignment. Compare the CONNECT_SINK and CONNECT_SOURCE message parameters to understand well the differences.

4.7. Characterization of the audio performances of the Audio Bridge

SLIMbus is running, the Audio Bridge is configured with 2 channels. We are ready to run some audio measurements.
The Audio Bridge as such does not have any influence on the data that are getting transmitted over SLIMbus. However, the asynchronous sample rate converter (ASRC) of the I2S input interface, when it is used, has some very minor but measurable effects.

4.7.1. In-band frequency response.

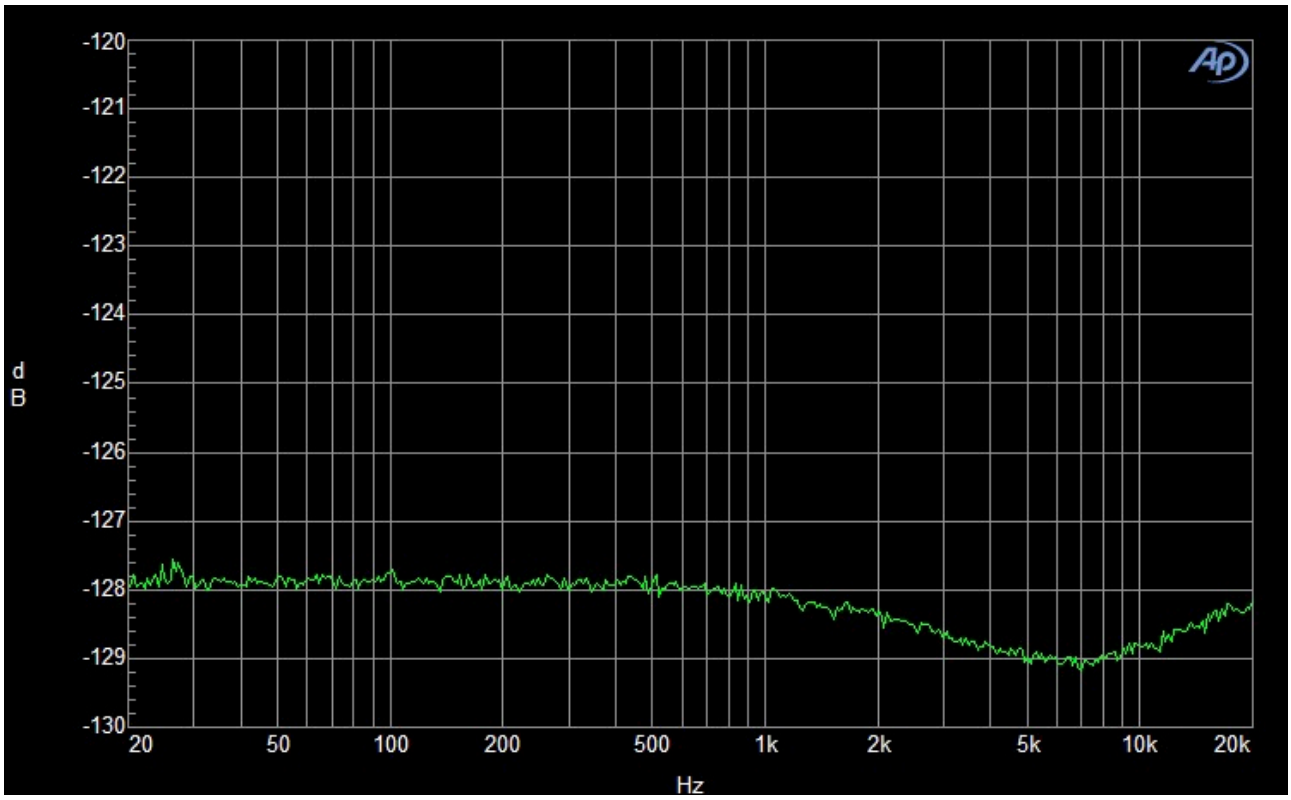
Even if the ASRC is of very high quality, it is not 100% transparent. So it is possible to measure its contribution to the data loop.



The in-band ripple introduced by the ASRC is +/- 0.01 dB. This measurement is valid for a conversion ratio equal to 1 (48k:48k).

4.7.2. THD+N vs. Frequency measurements

Similarly, it is possible to characterize the THD+N contribution of the ASRC. The following measurement is valid for a conversion ratio of 1 (48k:48k).
The measured value is below -128 dB. This shall be enough not to disturb any THD+N measurements of D-to-A converters.

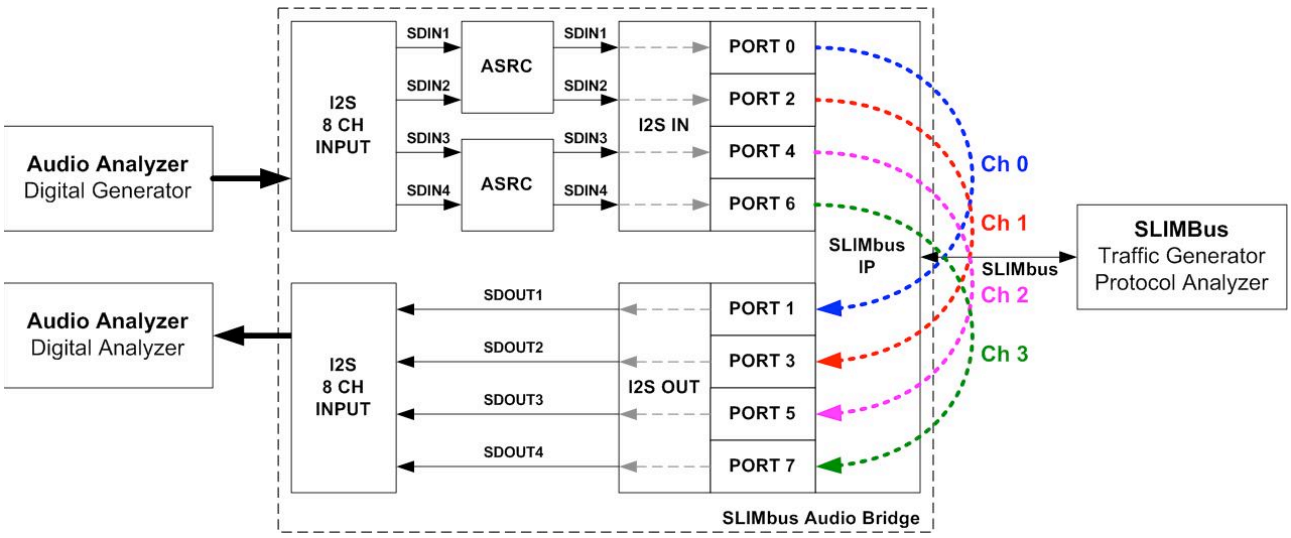


The ASRC used on the I2S input is a bit less performant (about 12 dB) in THD+N and dynamic range than the one used on the SPDIF input.

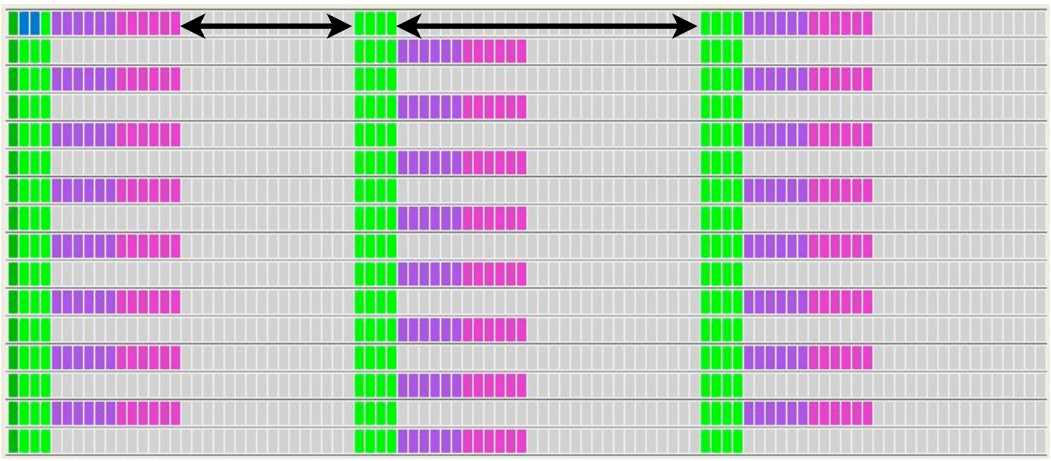
4.8. Modifying the script to add 2 more channels

To validate an 8 ports Audio Bridge, we need to test the streams SDIN3, SDIN4, SDOUT3 and SDOUT4. This can be done by adding two new channels and duplicating the data port connection scheme.

The data paths will be as follow:

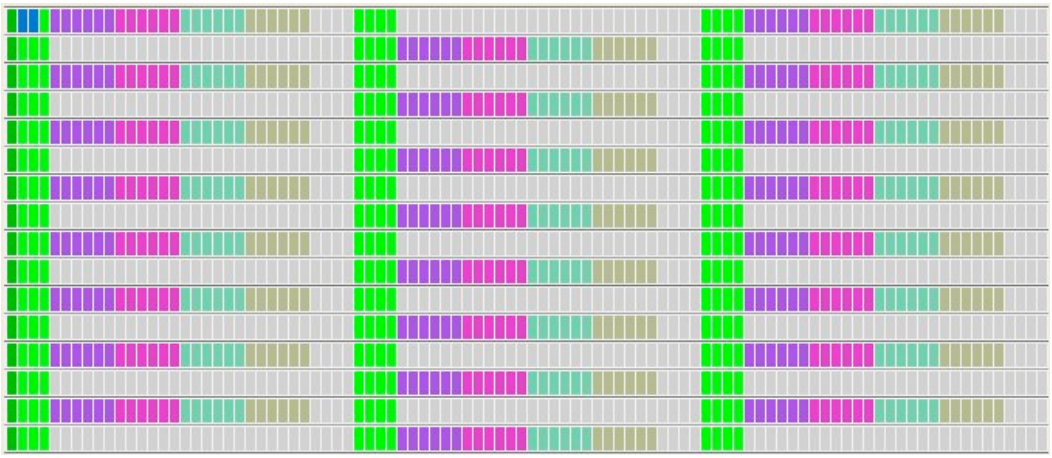


To create 2 new channels, we must make sure that they will fit in the available bandwidth and that they will not conflict with the existing channels. The current bandwidth allocation is as follow:
Channel 0 segments are drawn in purple. Channel 1 segments are drawn in pink.
As we want 2 more channels at 48 kHz (like the two existing ones), the available bandwidth is shown by the grey slots in between 2 consecutive segments of channel 0. We need 2 x 6 slots and we have 8 + 20 = 28 slots free. No problem of bandwidth, then.



We will place channel 2 and channel 3 besides channel 1. Channel 2 will have an offset equal to 16 and channel 3 will have an offset equal to 22. All other parameters of the messages required to create a data channel will remain the same.

After execution of the modified script, the channel bandwidth allocation will be:

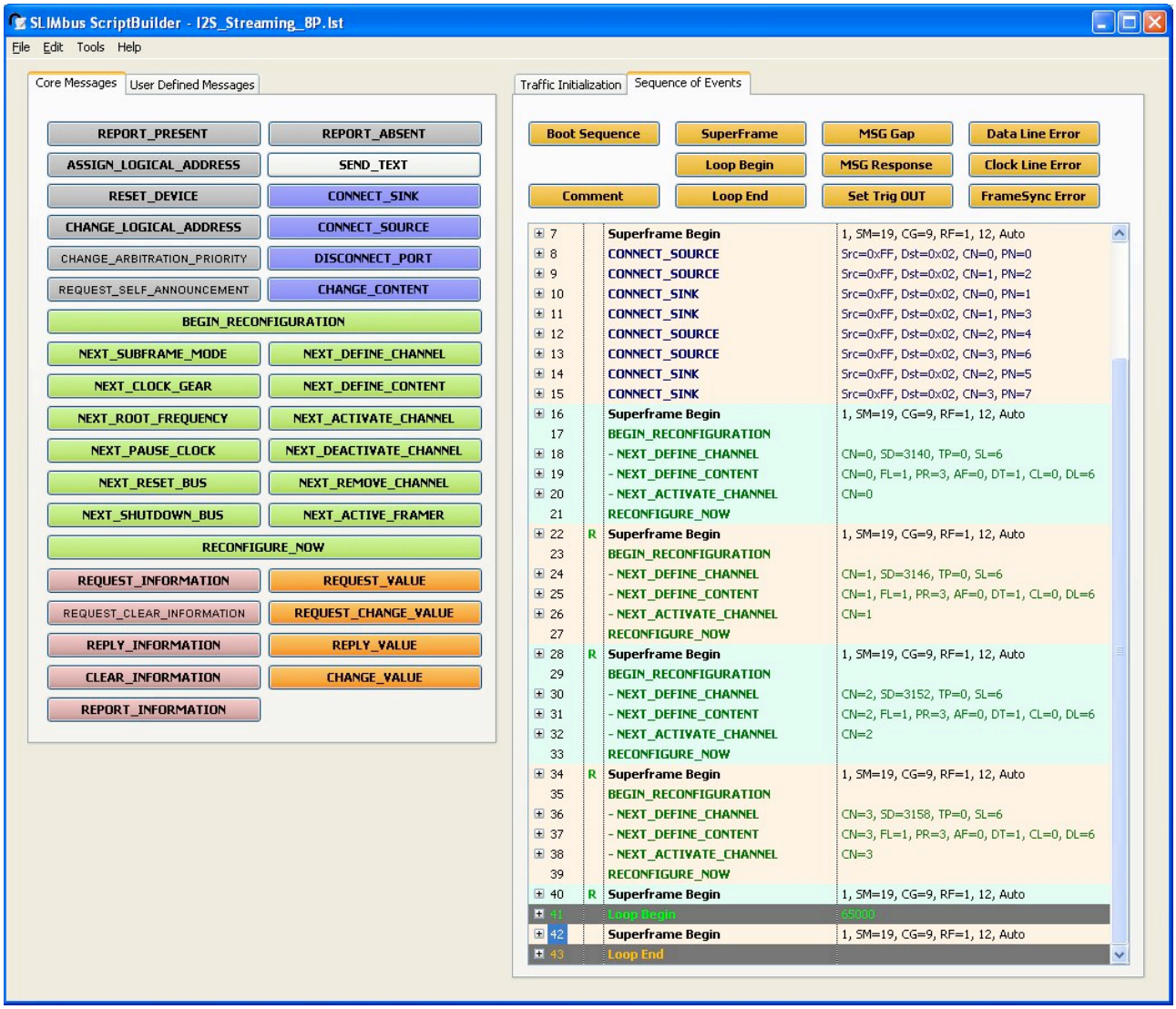


Channel 2 segments are drawn in blue and channel 3 segments are drawn in kaki.

The bridge display will show the following port assignments:

```
  I2S >ASRC24>SB>SDO2
P0123456 7 OPR=48k
  IOIOIOIO IPR=48k
Framer:Off,RF=1
```

The script windows shows clearly the 2 new channels and the new port assignment.
The new script name is “I2S_Streaming_8P.lst”.



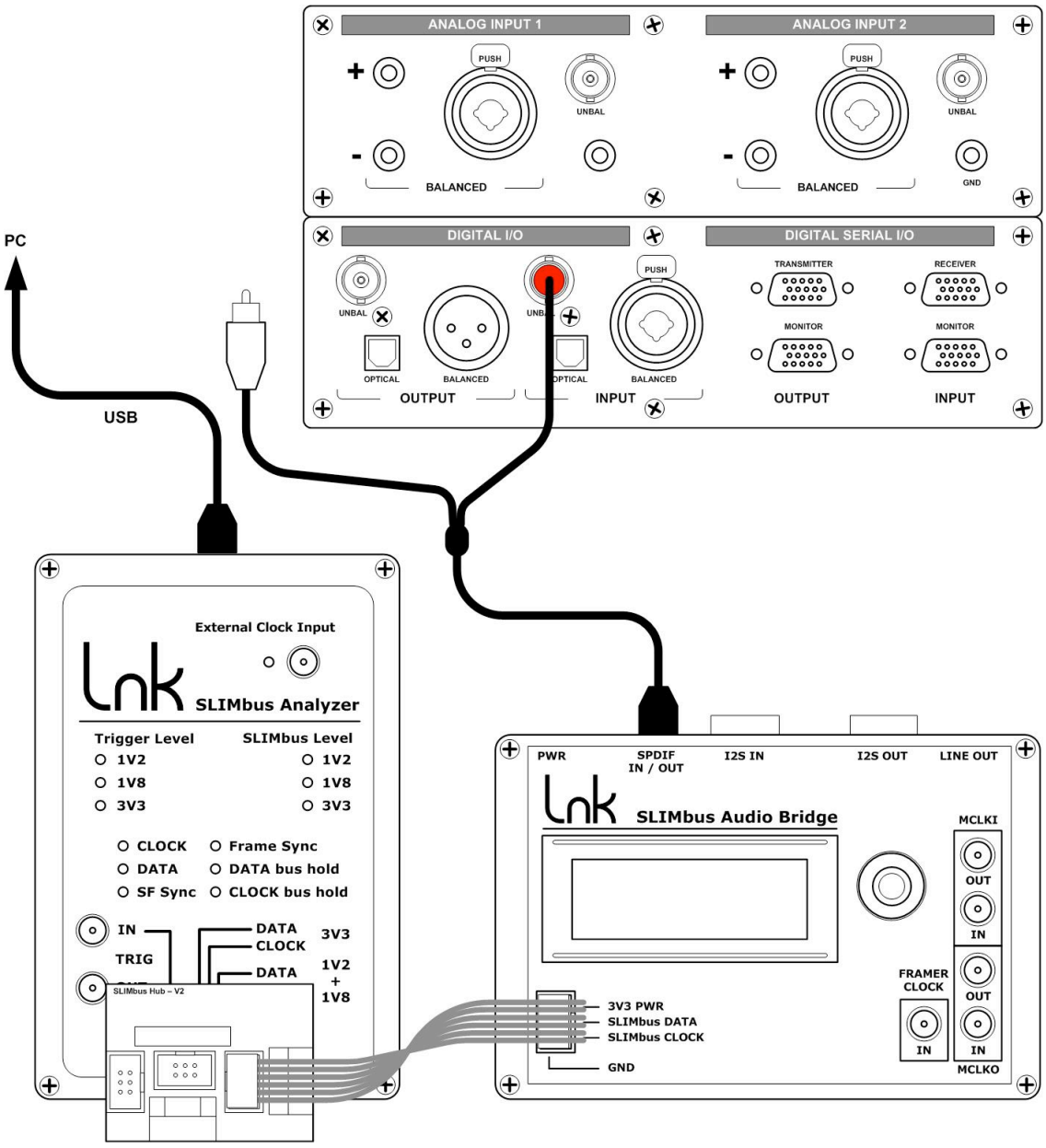
5. Audio Bridge Loopback

The Audio Bridge has an internal loopback functionality. The data that are received from SLIMbus , on the sink ports, are retransmitted without any modification on the source ports, and then on SLIMbus if the source ports are attached to data channels.

This feature is especially useful to test Application Engines (processors).

To exercise this functionality, we do not really need an audio analyzer, as all the data traffic will happen on SLIMbus. However, the data present on the sink ports are outputted on the I2S outputs and on the SPDIF output. We can use the Bridge SPDIF output for data monitoring.

5.1. System Setup



We will not connect any signal source to the bridge for this test.

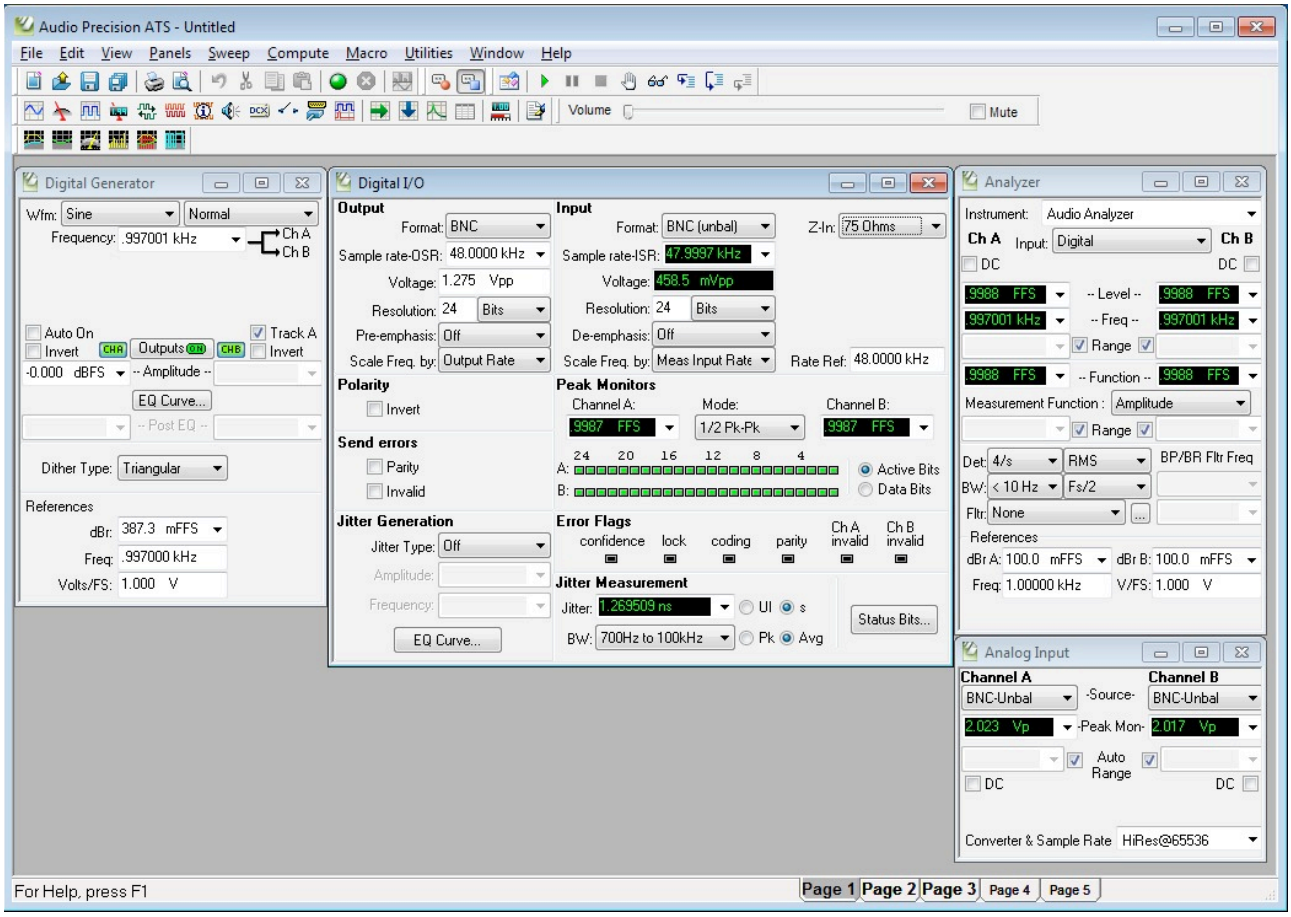
5.2. Audio Bridge configuration

The Bridge does not need any specific configuration. All necessary settings will be done through the messages sent by the test script.

5.3. Audio Analyzer configuration

Set the digital input on the unbalanced BNC connector and select the digital input in the analyzer.

The measurements were made with an Audio Precision ATS2. Any other audio analyzer will also work, assuming the same kind of setup can be achieved.

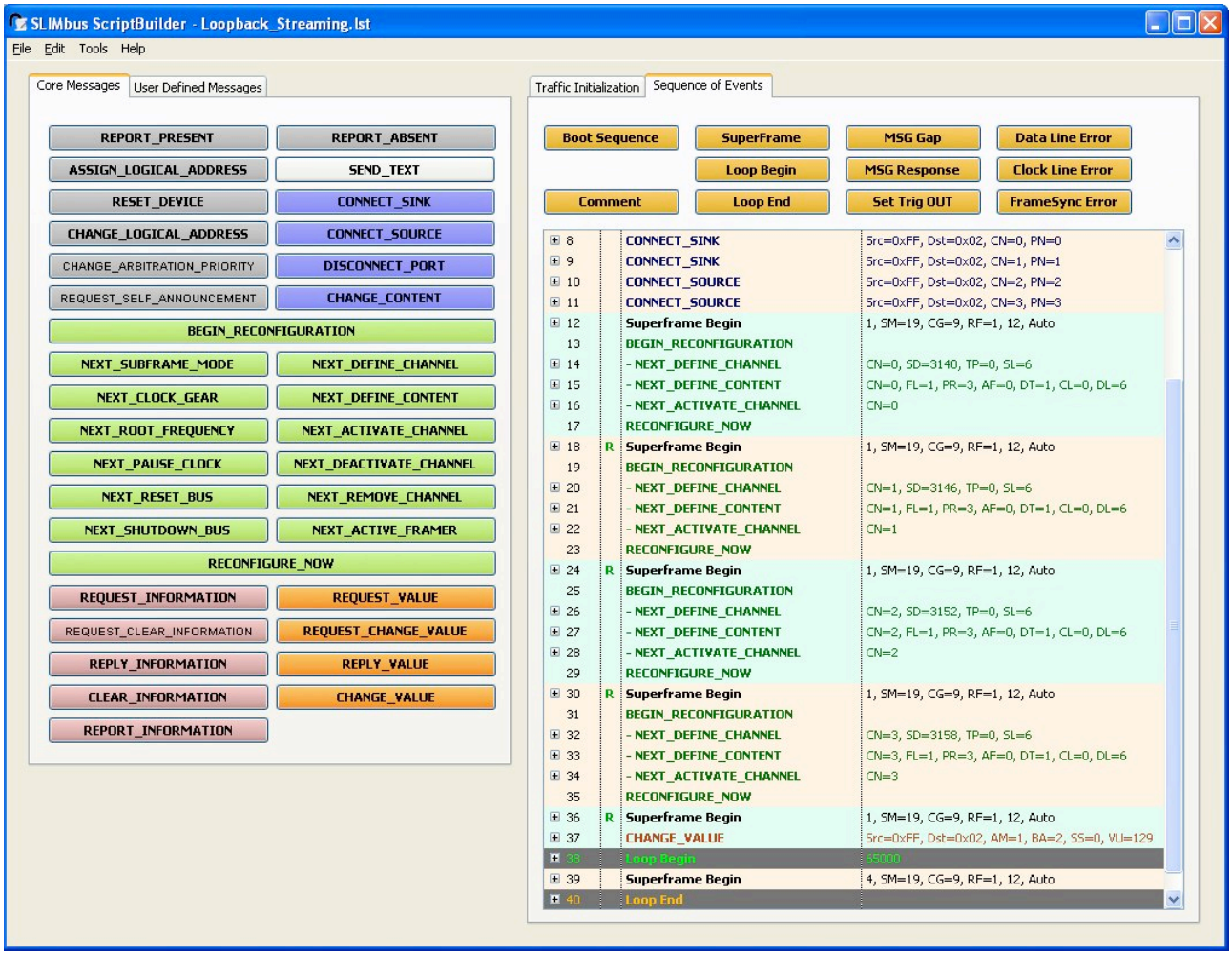


5.4. Traffic Generator configuration

Launch the SLIMbus Protocol Analyzer application.
Launch the ScriptBuilder application.
Load the script “I2S_Streaming.lst” in ScriptBuilder.

Push the script to the Traffic Generator by either hitting Ctrl+T or by going into the menu “**File / Send Script To Traffic Generator Ctrl+T**”.

In the Protocol Analyzer main window:
- Click the “MSG” view button
- Click on the “MSG only” tick box
- Click on the “LiveView” button
- Click on the “Play” button

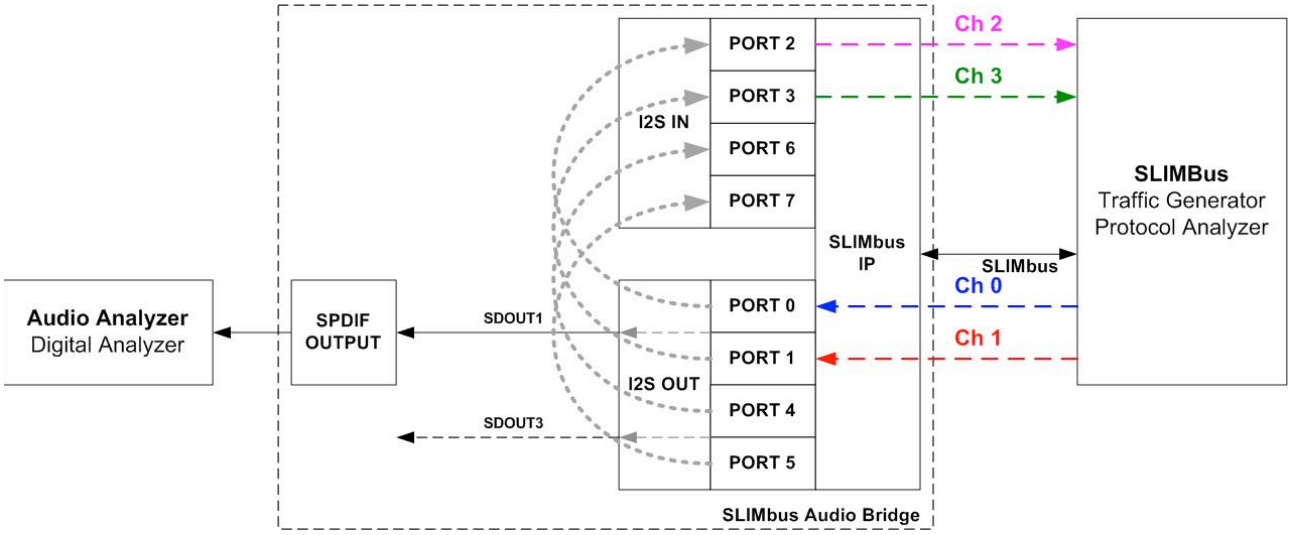


The bridge is put in internal loopback mode and four data channels are created.
On the bridge display main page, the following shall appear:

```
  SB  >  Loopback  >  SB
P0123456 7  OPR=48k
  IIOO----  IPR=48k
Framer:Off,RF=1
```

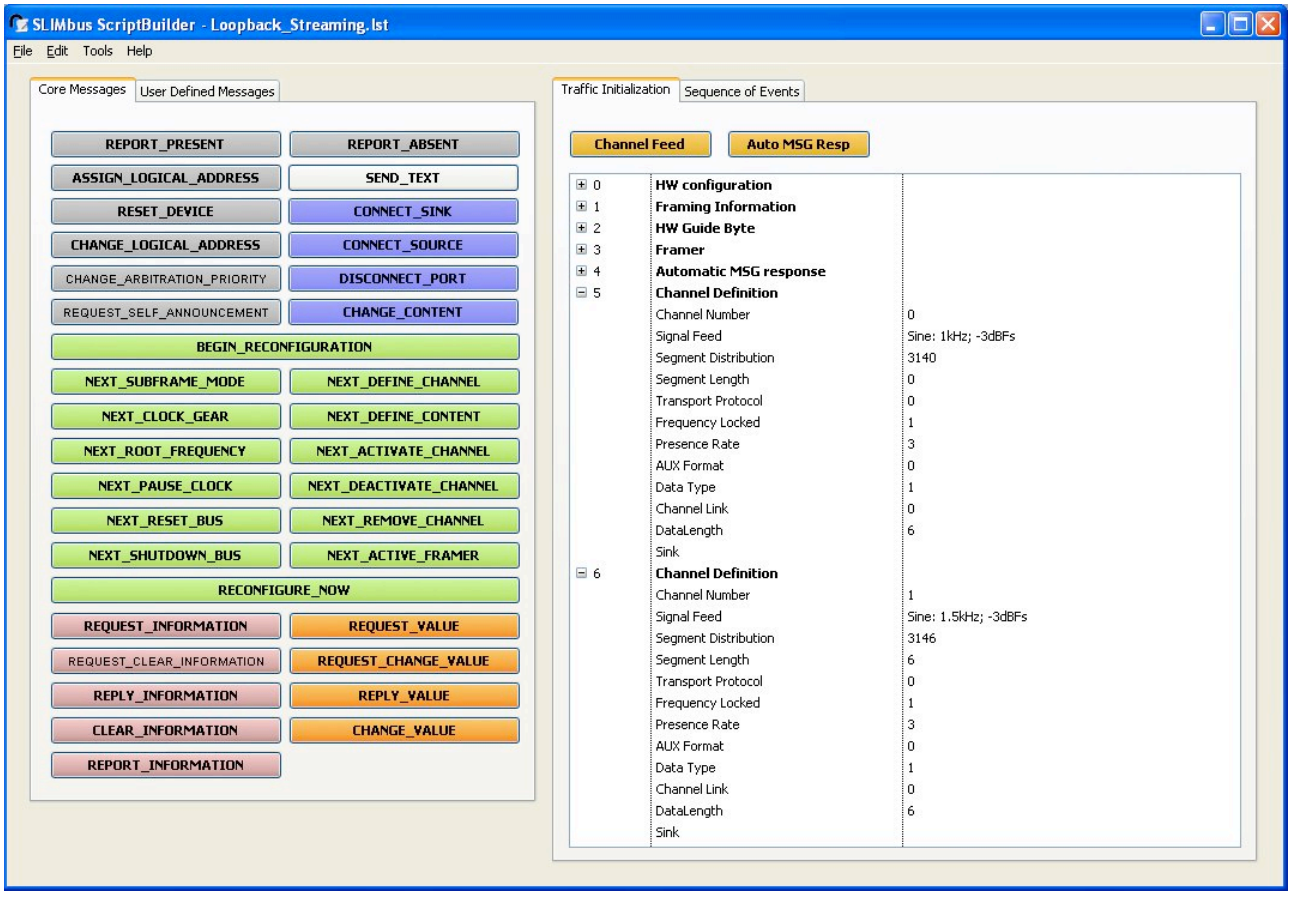
5.5. Data path

The script has created the following data paths:
- Channel 0 from Traffic Generator to Bridge port 0
- Channel 1 from Traffic Generator to Bridge port 1
- Channel 2 from Bridge port 2 to Protocol Analyzer
- Channel 3 from Bridge port 3 to Protocol Analyzer
- Port 0 output loop back to port 2 input
- Port 1 output loop back to port 3 input
- Port 1 and 2 outputs are routed to the SPDIF output



5.6. Script file analysis

Let's now have a look to the script to see what has been done. The data channel creation is similar to the previous scripts. The big news is about the Traffic Generator transmitting sine waves in channel 0 and channel 1.



In the Traffic Initialization tab of ScripBuilder, we see two Channel Definition entries. We only need to specify the sine wave frequency and amplitude in the Signal Feed parameter. ScriptBuilder will automatically fill the other parameters when it detects data channels with the same number.
When the channels 0 and 1 are created, the Traffic Generator will automatically feed them with the sine waves we just defined.

These sine waves will be received by port 0 and port 1 and they will be re-injected in port 2 and port 3 to finally be transmitted in channel 2 and channel 3.

The bridge is actually looping back the two audio streams.

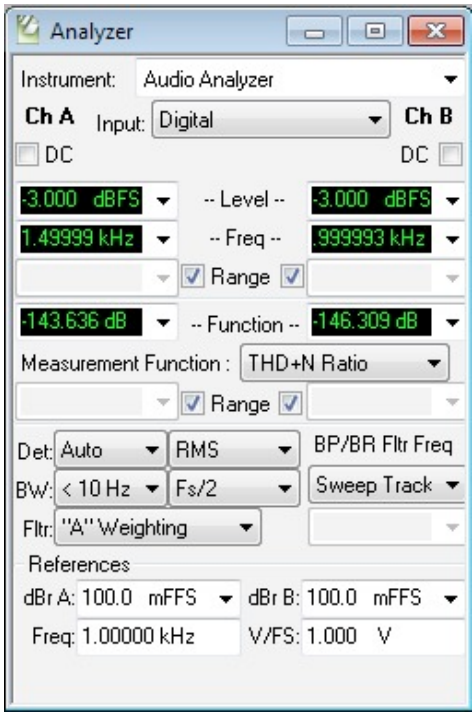
Note: In the two previous tests, we did data loop back over SLIMbus, by connecting ports together with SLIMbus data channels. This test does data loop back directly through the I2S interfaces by creating an artificial link between them.

The bridge loopback mode is activated by setting bit b7 and bit b0 of the register address 2 of the Generic device (which has get assigned Logical Address LA=2). This is achieved by the CHANGE_VALUE message in the script:

⊟ 37	CHANGE_VALUE	Src=0xFF, Dst=0x02, AM=1, BA=2, SS=...
	Source Address (Src)	0xFF
	Destination Address (Dst)	0x02
	Access Mode (AM)	1 - Byte Access
	Byte Address (BA)	2
	SS or BN (SS)	0
	Value Update (VU)	129

Incidentally, this message also forces the SPDIF output stream on SDOUT1 (the content of channel 0 and channel 1) by setting bit b1 to 0 and bit b2 to 0 (see the SLIMbus Audio Bridge user manual).

When we measure the signal quality with the audio analyzer, we find the following results:



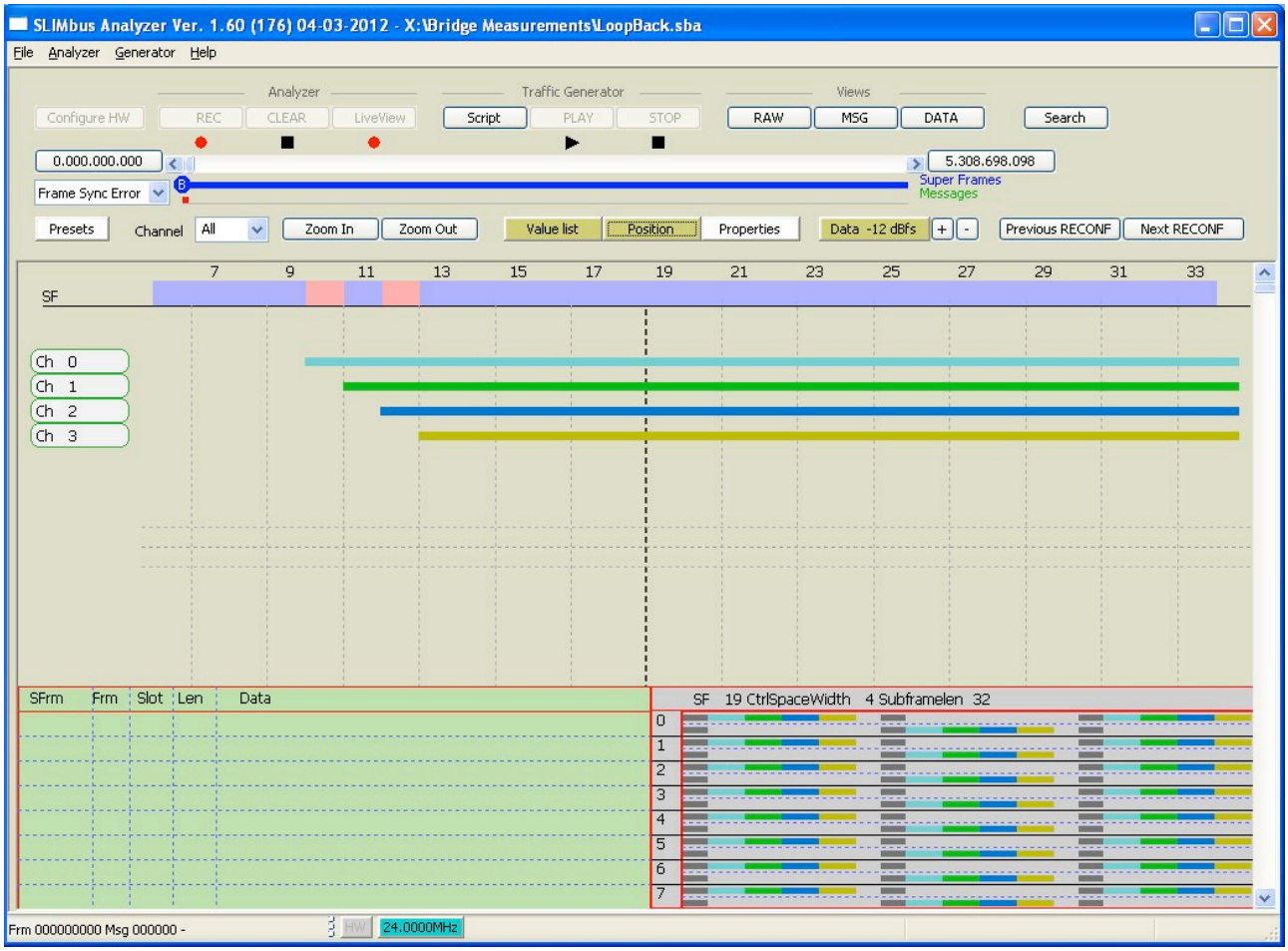
The amplitude is right, the frequency is also correct and the THD+N is as low as it can be.

The Traffic Generator is effectively transmitting two very clean sine waves on the SLIMbus. Now, to verify that the bridge is effectively looping back the data on channel 2 and 3, we need to use another feature of the protocol analyzer: the full record mode.

Up to now, we only used the LiveView record mode (where only the messages are shown). We are now going to use the full record mode to be able to record and visualize the content of the data channels.

In the Protocol Analyzer main window:
- Click on the “REC” button
- Click on the “PLAY” button
- Let the PC record the SLIMbus stream for about 10 seconds then click on the two “STOP” buttons to stop the recording and streaming and to start the data decoding.
- Click on the “DATA” button to access the data channel analyzer

Select “All” in the Channel drop box. The activity cart of the data channels is displayed. Time goes from left to right. The top bar with the numbers indicate the superframe numbers.

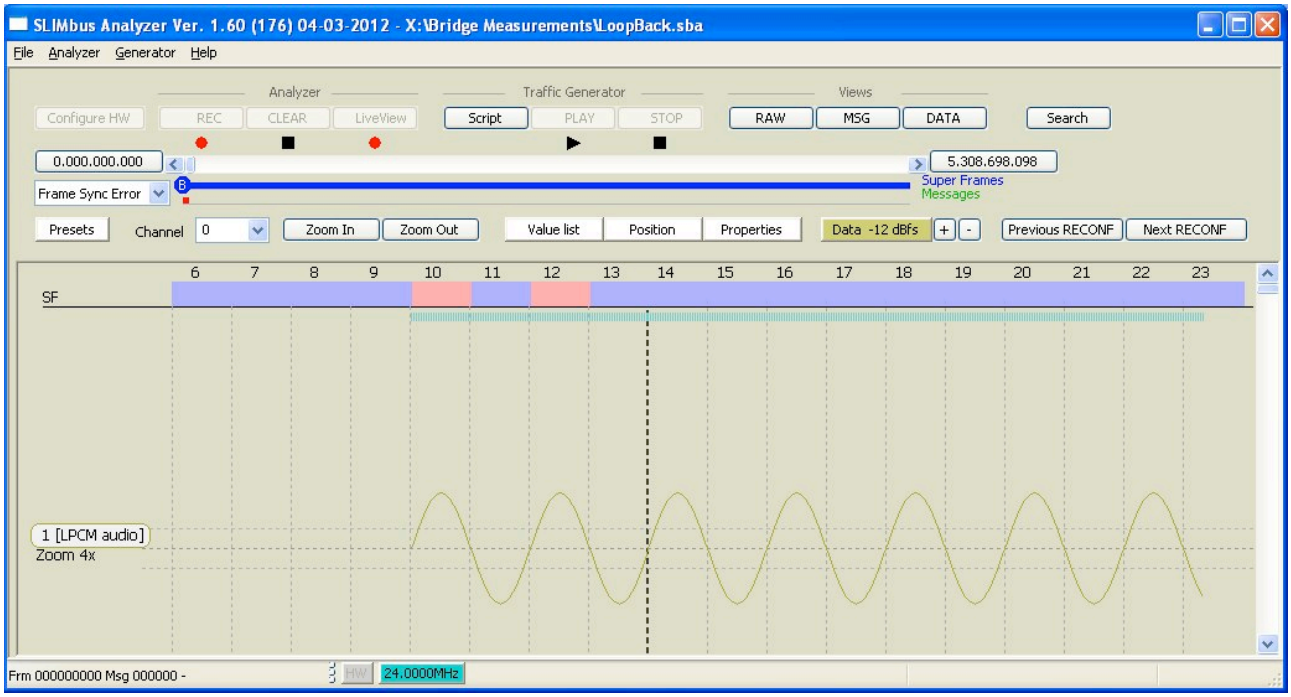


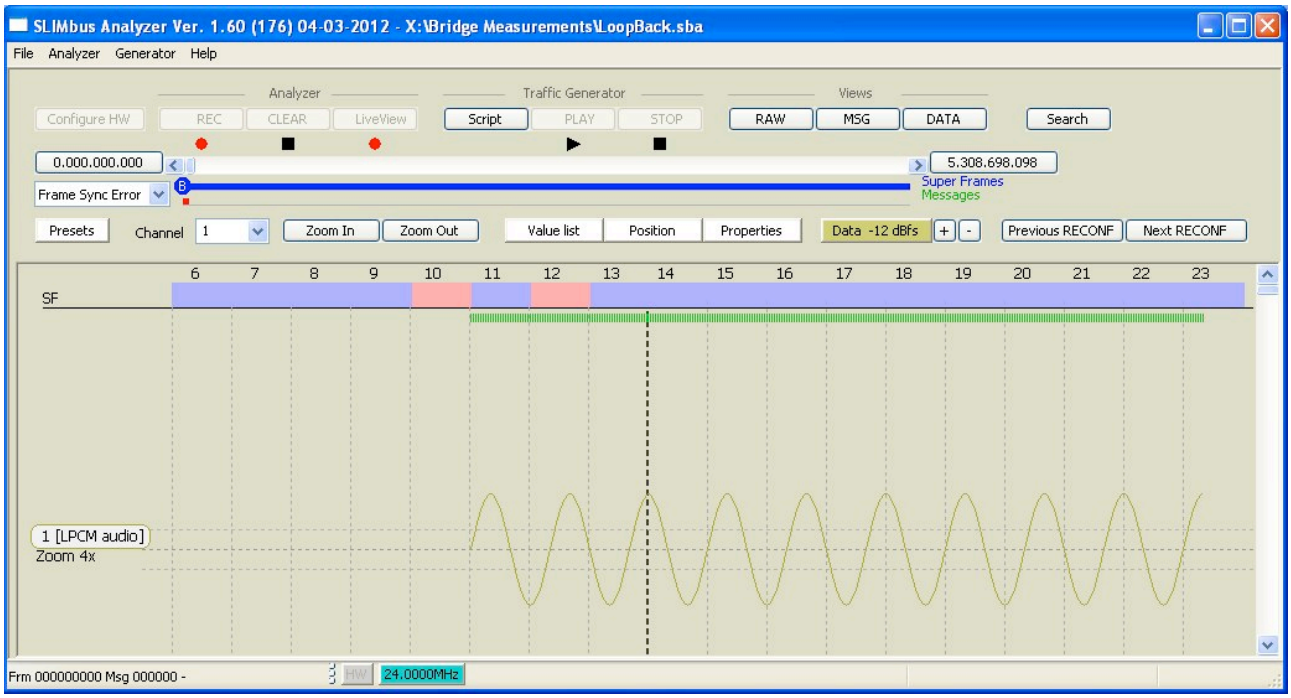
Note: We can see that the four channels have been activated sequentially. By modifying the script to have all four NEXT_ACTIVATE_CHANNEL messages in the same reconfiguration sequence, they would have started at the exact same time.

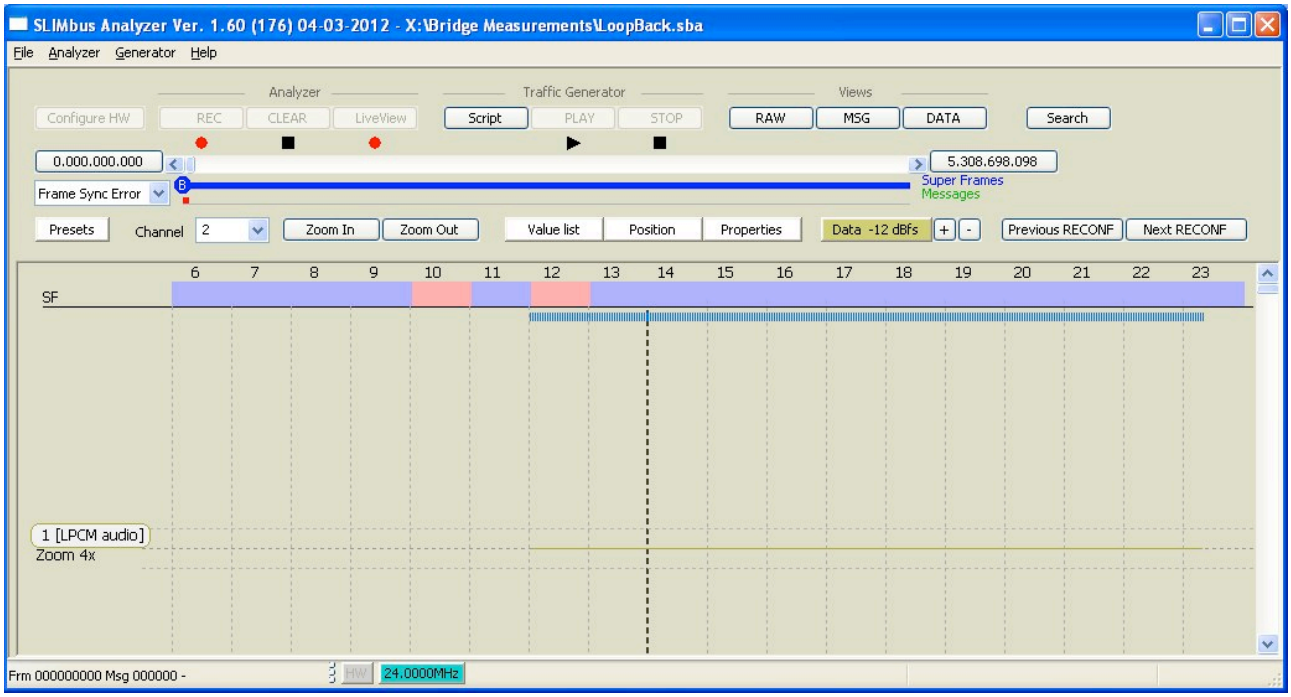
Now, select channel 0 in the drop box and click on the “Data 0dBFs” button. The sine wave will be displayed. Zoom IN or OUT in amplitude by clicking on the “+” or “-” buttons. Zoom IN or OUT in time by clicking on the “Zoom In” or “Zoom Out” buttons.

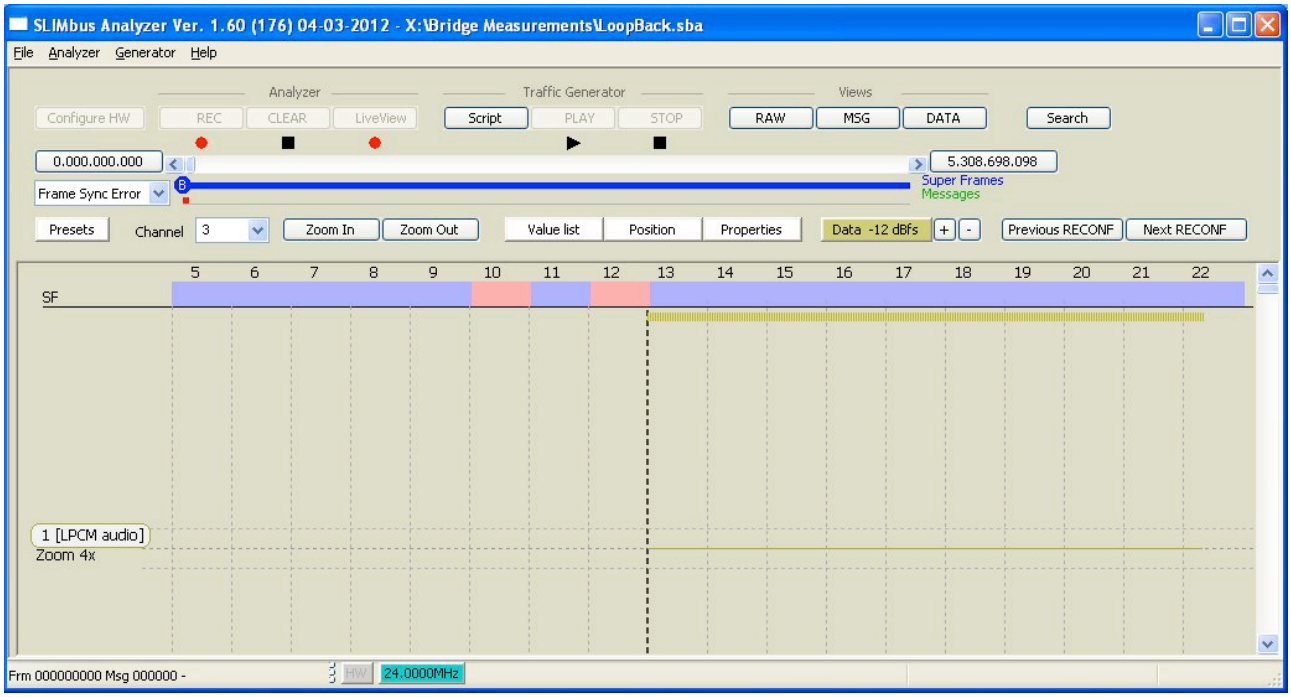
Only one channel at a time can be displayed on the screen. So we will have to check them one by one.

Channel 0 and channel 1 carry the sine wave generated by the traffic generator. We used 1000 Hz in channel 0 and 1500 Hz in channel 1. The period difference shall be visible.



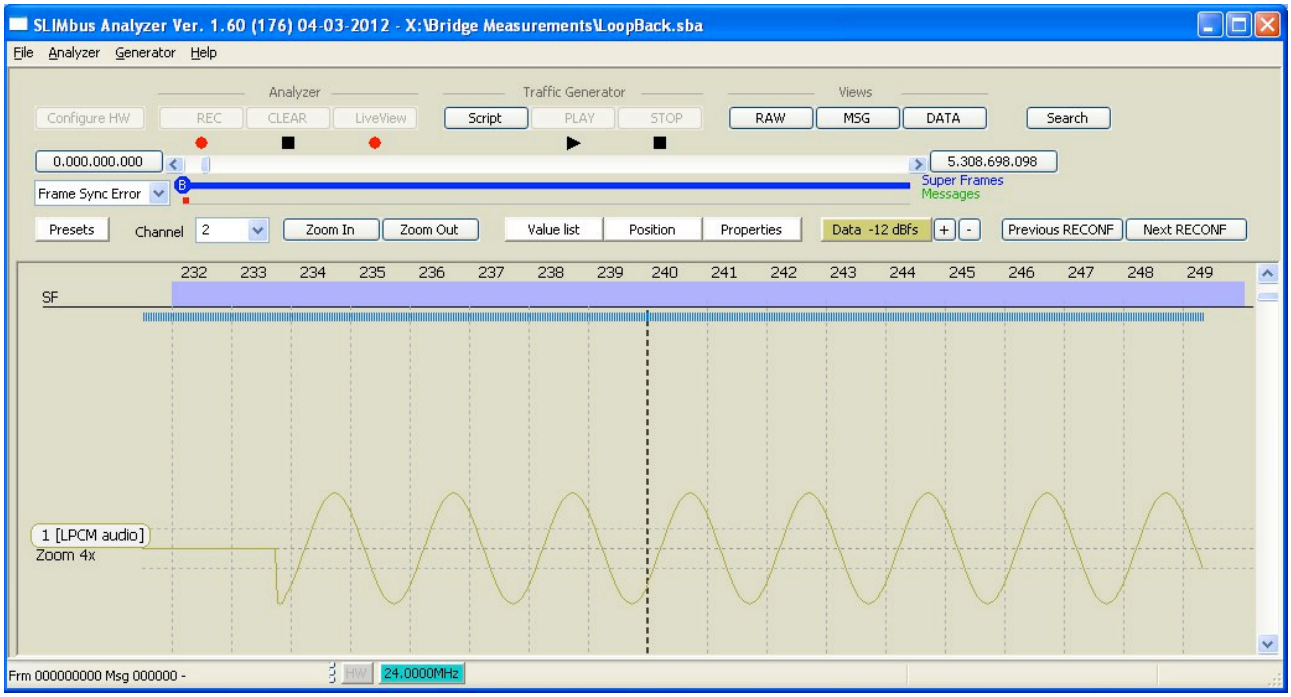


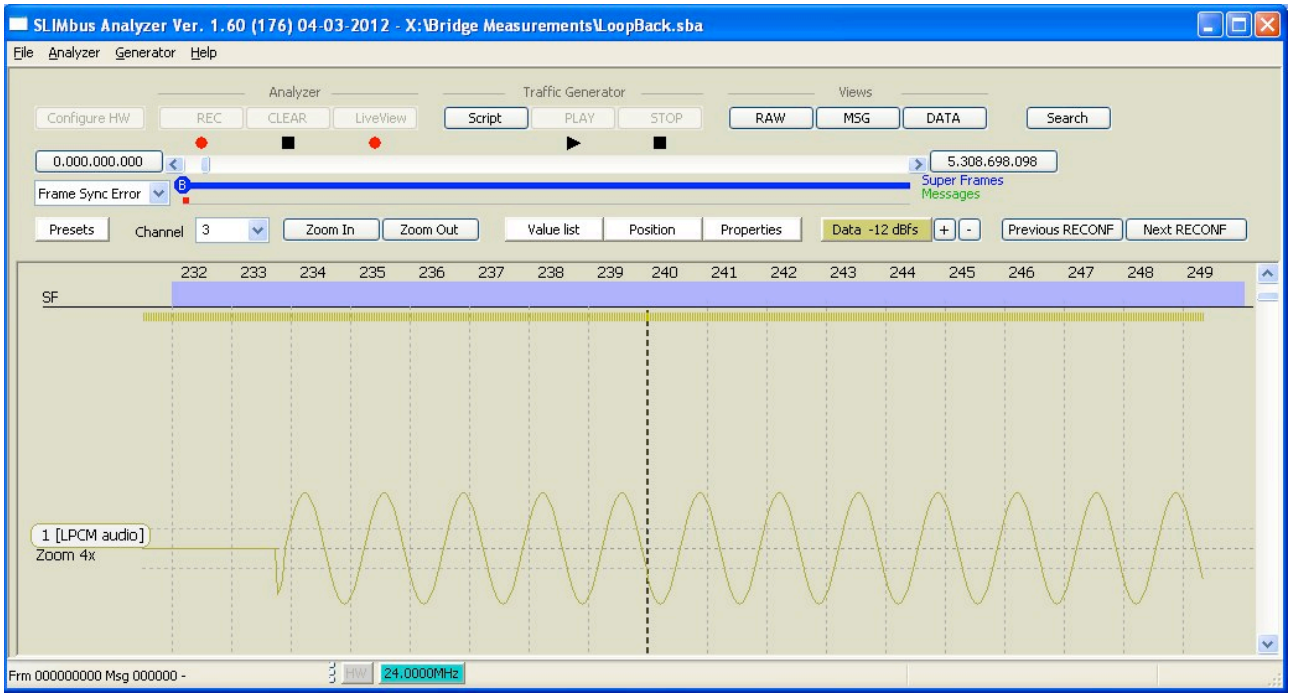




We can see that at the moment the streaming is starting, there is no data yet in channel 2 and 3.

Indeed, the bridge needs some time (about 100ms) to activate the internal loop back. Scroll in time to reach superframe 240 and then look again the content of channel 2 and channel 3.





Channel 2 and 3 are effectively carrying the expected data. The internal loop back is operational.