

SPISim[®] StatEye/AMI User's Guide

Latest Version: V20180315

SPISim LLC

Vancouver, WA 98683, USA

Tel. +1-408-905-6692

<http://www.spisim.com>



This user's guide describes the SPISim's StatEye channel analysis flow and Spec. IBIS-AMI model in SPISim's products, SPIPro.

Contents:

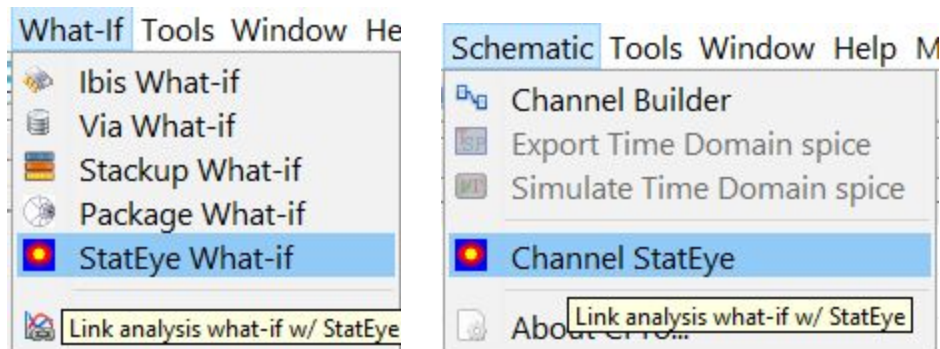
- [**1. System Requirements and Invocation**](#)
- [**2. Flow and Functional Description**](#)
 - [**2.1 Participating model**](#)
 - [**TX EQ**](#)
 - [**Channel**](#)
 - [**RX EQ**](#)
 - [**2.2 Simulation flow**](#)
 - [**Statistical mode**](#)
 - [**Bit-by-bit mode**](#)
 - [**2.3 Functional description**](#)
 - [**Description**](#)
 - [**Main**](#)
 - [**TX EQ**](#)
 - [**Channel**](#)
 - [**RX EQ**](#)
 - [**Sweep**](#)
 - [**Jitter**](#)
 - [**Output**](#)
- [**3. Spec. IBIS-AMI Model Description**](#)
- [**4. Command-line/Batch-mode Flow**](#)

2. System Requirements and Invocation:

SPIPro supports the following OS:

- Windows: 32 or 64 bit, Windows 7 or newer
- Linux: 64 bit, Released kernel 2011 or later with GLibC2.14 or newer Installed

StatEye analysis flow can be invoked via “StatEye What-if” under “What-If” main menu of “Net Analysis Suite”, or via “Channel StatEye” under “Schematic” of free “SPILite” suite.



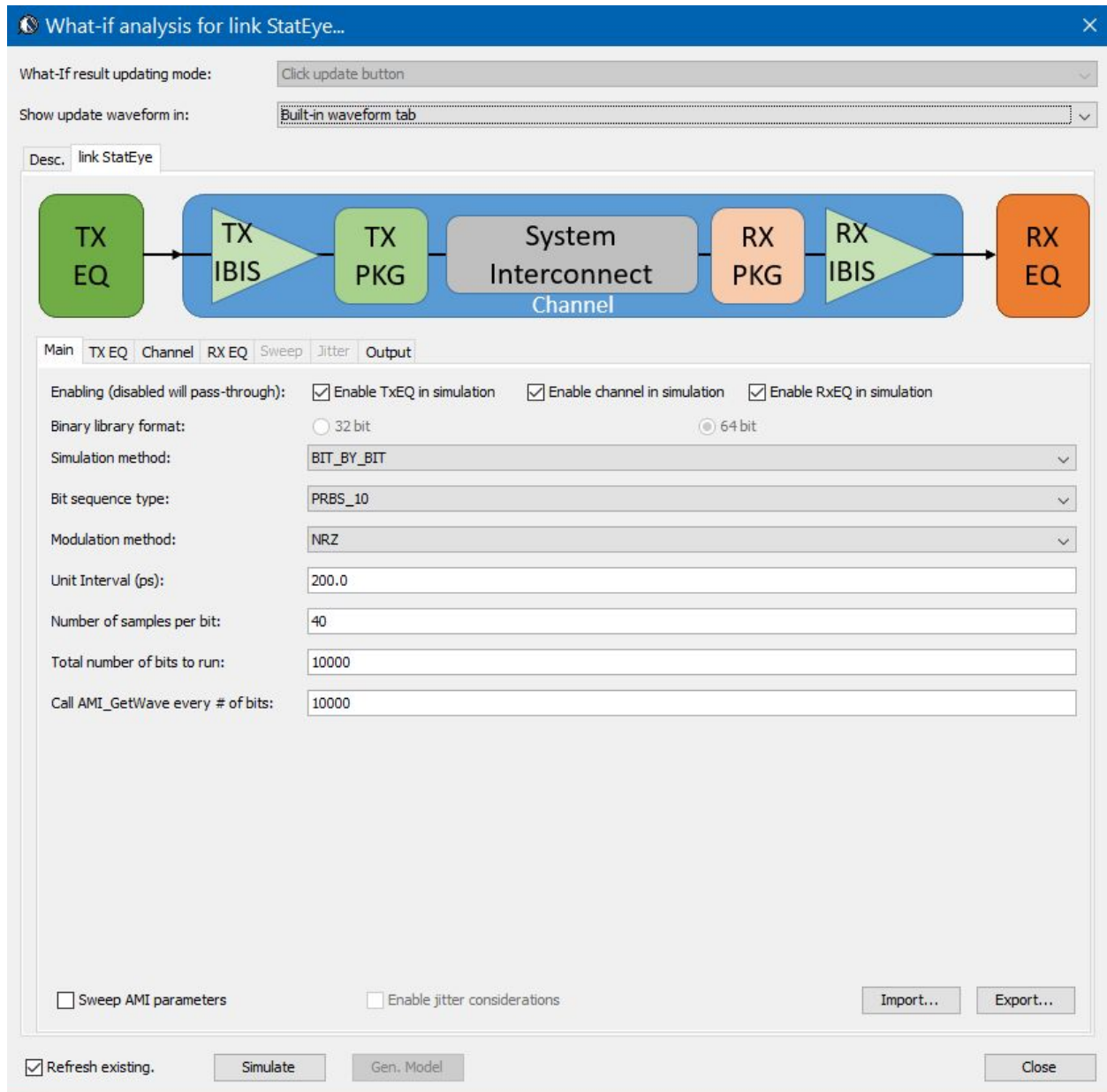
StatEye analysis in free “SPILite” suite is subject to the following limitations, which are unlocked in the full version:

- Bit-by-Bit: Can only simulate up to 5000 bits
- BER target: Can only measure eye-high equal or larger than 1E-10
- AMI sweep: Disabled
- AMI model: Generated model will expire after one week

2. Flow and Functional Description:

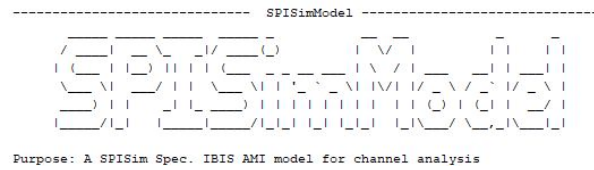
SPISim's StatEye analysis environment includes both front-end and back-end simulation parts. A GUI invoked by aforementioned procedure allows user to configure different simulation settings and participating models easily. Upon simulation starts, these settings will be sent to native executable for better performance. Generated results such as eye, bath-tub curves etc can then be viewed via the SPIPro environment after simulation is done.

The screenshot below shows the StatEye configuration GUI:

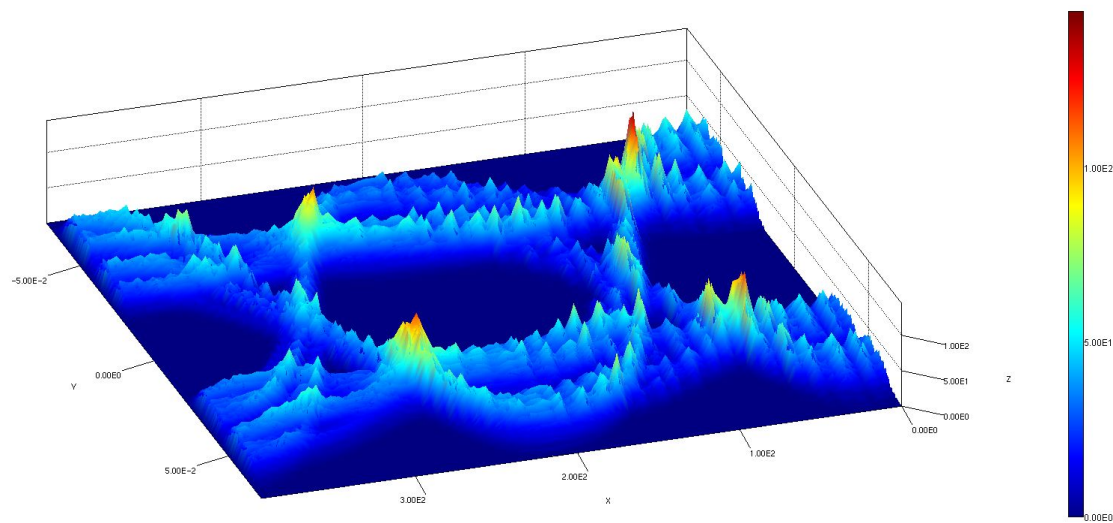
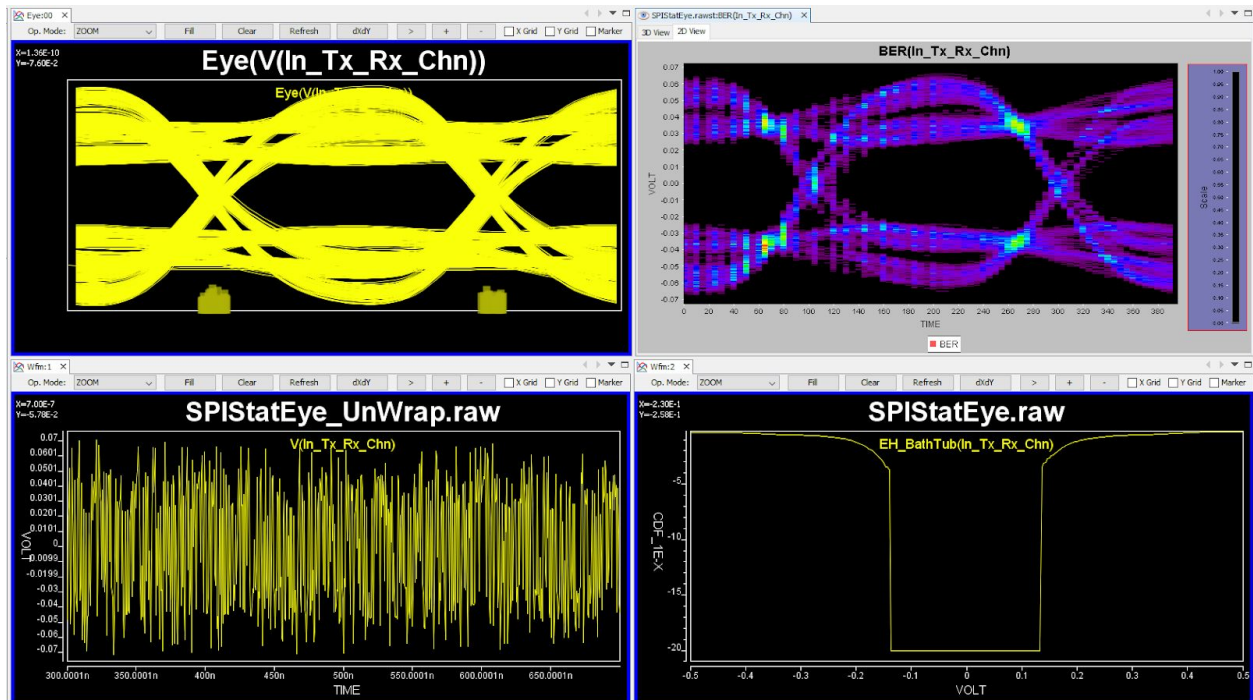


The screenshot below shows partial of the StatEye simulation process:

```
Simulate: C:/WeiProj/SPISim/Prj/SPISimPrj/build/cluster/modules/ext/SPISimAMI_WIN64.exe STATEYE D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/SPISatEye.cfg.
I[019]: Running stateye analysis using config. D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/SPISatEye.cfg ...
I[006]: Parsing AMI parameter file D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/SPISimTX.ami...
I[007]: Loading AMI library file D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/SPISimTX.dll...
I[006]: Parsing AMI parameter file D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/SPISimRX.ami...
I[007]: Loading AMI library file D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/SPISimRX.dll...
```

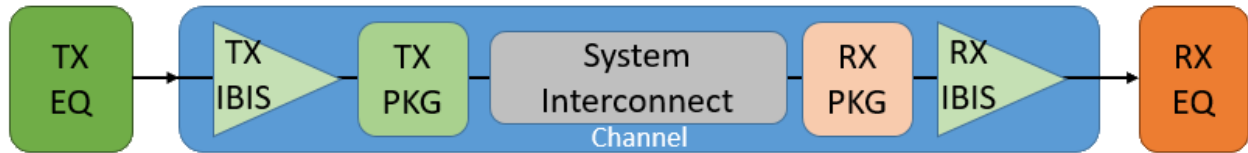


The screenshot below shows the StatEye results plotted in SPIPro's GUI:



2.1 Participating models:

The picture below, also shown at the top of the configuration GUI, depicts the schematics to be analyzed by StatEye:



There are three main participating models:

- **TX EQ:** this constitutes the equalization part at the transmitter. The main responsibility of this model is to equalize PRBS based stimulus before feeding to analog, passive channel. TX EQ model is represented by an IBIS-AMI model.
- **Channel:** this constitutes the full channel. A channel starts and ends with analog front ends which are usually represented by spice subcircuit or IBIS models, depicted as TX IBIS and RX IBIS respectively. TX IBIS must be active while RX IBIS can be either active like voltage divider or passive like an RC termination. The TX and RX package together with system interconnect are passive and linear, time-invariant. They are usually represented by S-parameters, transmission lines or converted broadband spice models extracted from pre-layout or post-layout designs using tools outside SPIPro. The system interconnect may include more stages such as vias, main routes and connectors etc. Channel's impulse or step response waveform is used by StatEye analysis.
- **RX EQ:** this constitutes the equalization part at the receiver. The main responsibility of this model is to equalize signals received at the end of analog, passive channel and recovers clocks if needed. RX EQ model is represented by an IBIS-AMI model.

SPIPro supports modeling capabilities for both IBIS and IBIS-AMI. Channel's response may be simulated by built-in or 3rd party simulator such as HSpice, LTSpice or NGSpice. Extraction from post-layout and device models within interconnect, such as via and connector, needs to be done outside SPIPro.

Any of these models may be “disabled” and be treated as a “pass-through” for analysis. In such cases, the effect of disabled model(s) will be diminished. On the other hand, SPIPro allows various data sources to be used for each of these models. Please see 2.3 for more details.

2.2 Simulation flow:

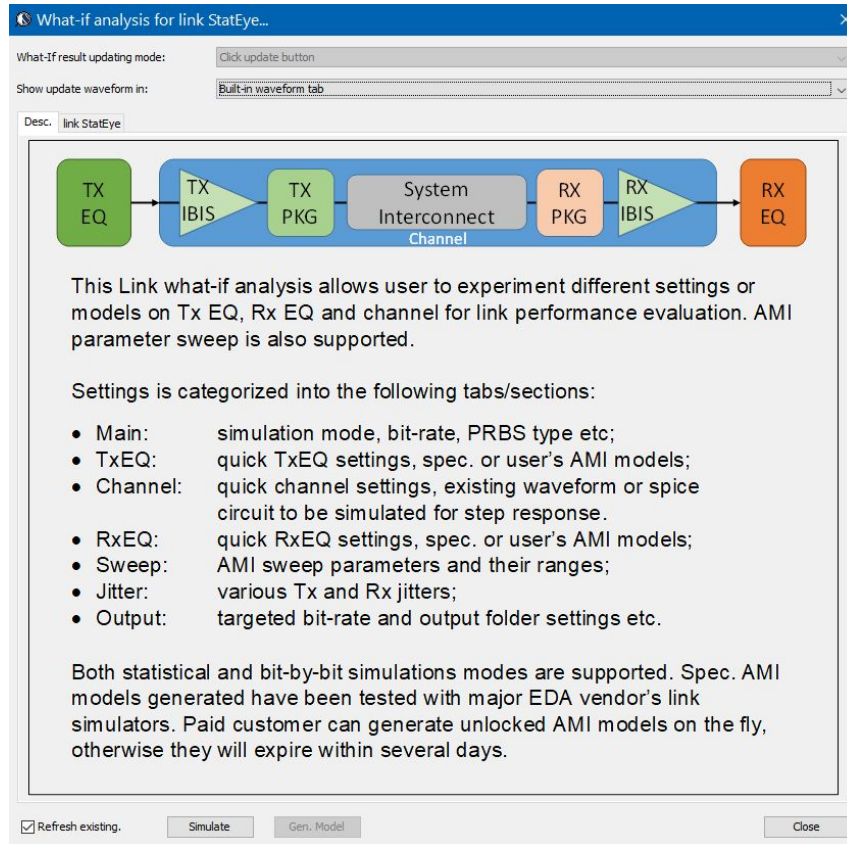
SPISim’s StatEye flow follows the common methodology of same name (i.e. StatEye) document in various DesignCon paper and IBIS spec. Both “statistical” modes and “Bit-By-Bit” modes are supported. With the assumption that channel model is already passive and linear time-invariant (LTI):

- Statistical mode: applicable when both TX EQ and RX EQ are also LTI.
Channel’s impulse response will be used directly to convolve both EQs and a lone bit. Resulting pulse will be used to synthesize a PDF/BER eye. This process is very quick, usually done within several seconds, and the resulting bathtub curve may be used to extrapolate for low BER measurement.
- Bit-by-bit mode: applicable when either of TX EQ and RX EQ is non-linear and/or time-variant (NLTV).
Channel’s impulse response will be convolved with PRBS stimulus for full length of specified number of bits. These bit sequences may be broken into chunk of different size. Then time-domain waveform of each chunk will be fed into TX EQ followed by RX EQ. Resulting waveform is then assembled to form different performance metrics such as eye and bath-tub curves. Unwrapped time-domain waveform may be exported. It usually takes several minutes to simulate one million bits. Further processing is needed to remove deterministic noise before extrapolation for low BER can be performed.

2.3 Functional description:

This section describe StatEye's GUI usage:

- Description: this tab gives simple descriptions of the StatEye process.



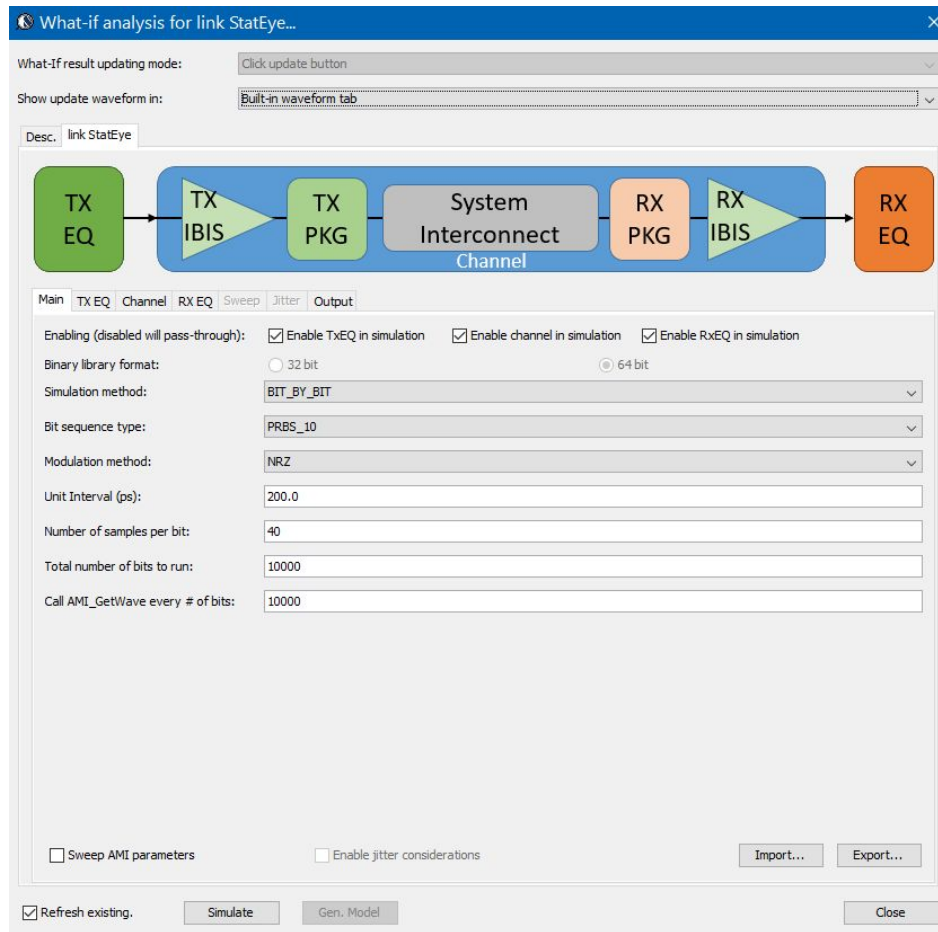
"Show update waveform": the selection at the top specifies how SPIPro should display the results, either within the same application or launching a separate child process/window.

"Refresh existing": enables waveform auto-refresh upon results becomes available. To compare results from different interactive/single run, uncheck this so that waveform will be available individually for plotting or overlapping in GUI.

"Simulate": launch simulation. GUI will be disabled during simulation and re-enabled upon completion. User should configure the rest of the models and settings before clicking "Simulate" button.

"Close": close this StatEye dialog.

- Main: this tab specifies the main simulation settings:



“Enabling”: specify which channel model will participate in the simulation. Disable model will become “pass-through” and their associated tabs will be disabled.

“Simulation method”: either Statistical or Bit-by-bit. Some of the settings below such as bit sequences etc are only enabled in Bit-by-bit simulation mode.

“Bit sequence”: PRBS7, 10, 15, 23 or 31

“Modulation”: NRZ or PAM4, may only affect DFE/CDR EQ models

“Unit Interval”: one bit time

“Number of samples per bit”: how many samples in one UI

“Total number of bits to run”: run how many bits

“Call AMI_GetWave...”: break total number of bits to run into chunk(s) at most this many bits. AMI_GetWave will be called one chunk at a time.

“Sweep AMI parameters”: enables the “Sweep” tab

“Enable jitters”: enables the “Jitter” tab, to be supported in future release

“Import/Export”: import/export the full StatEye settings to a plain-text config file. Note that this config. file only for StatEye GUI only, not for command line run.

TX EQ: when enabled, this tab specifies the TX EQ settings.

Both TX EQ and RX EQ present similar GUI components. The tool supports these EQ models from one of three data sources:

- Quick Settings: user can specify parameters for pre-defined EQ function. An AMI model will be generated dynamically before simulation based on these settings.
- Spec. AMI Model: user can configure and specify customized EQ sub-stages, such as FFE followed by a level shifter or CTLE followed by an FFE. Please see section 3 for different spec. models pre-defined in SPIPro.
- User's AMI Model: user can specify their existing IBIS-AMI models, either generated by SPIPro or other model providers.

In TX EQ quick settings, user can specify FFE parameters such as number of taps and their weight (numerical value, not dB)

Main TX EQ Channel RX EQ Sweep Jitter Output

Tx model type: QUICK_SETTINGS

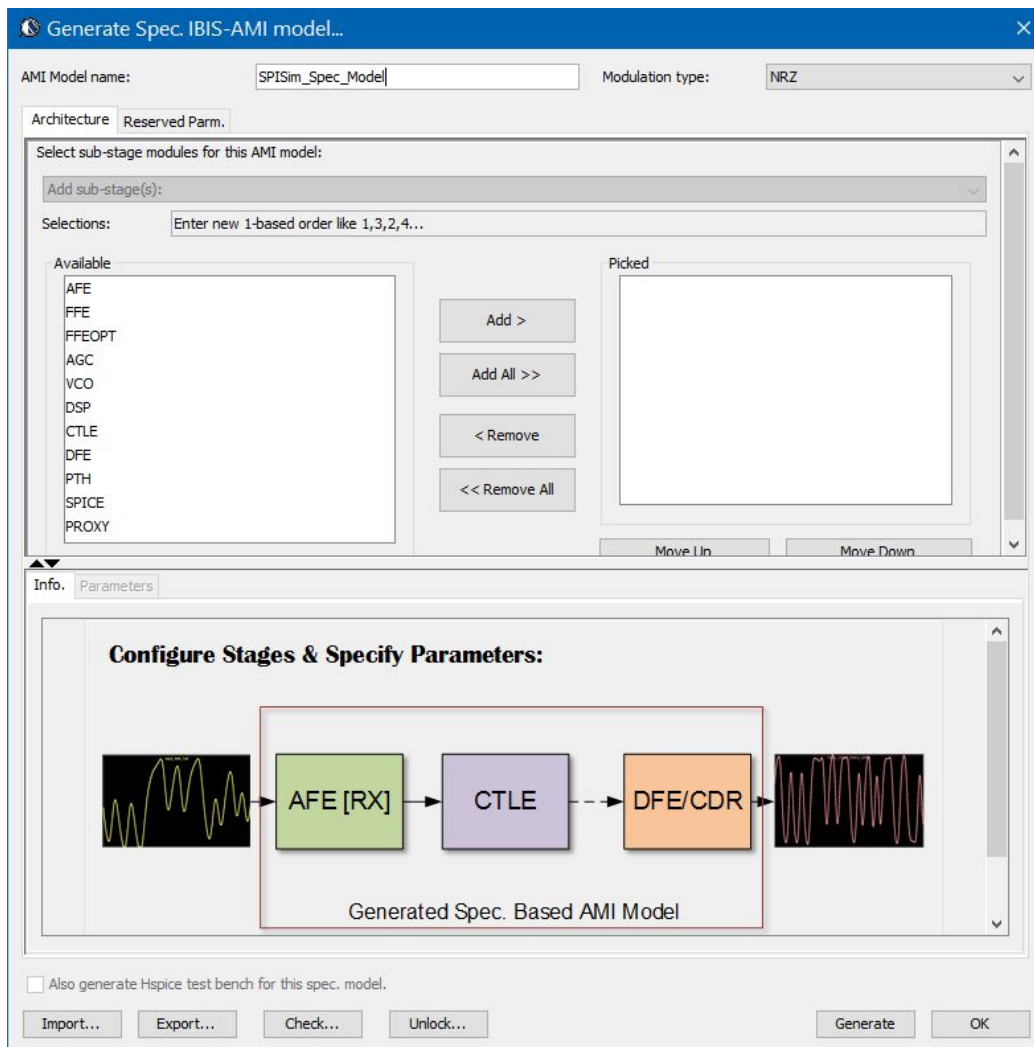
Quick Spec. AMI User's AMI

Enabling pre and post cursors: One_PreCursor_Two_PostCursors

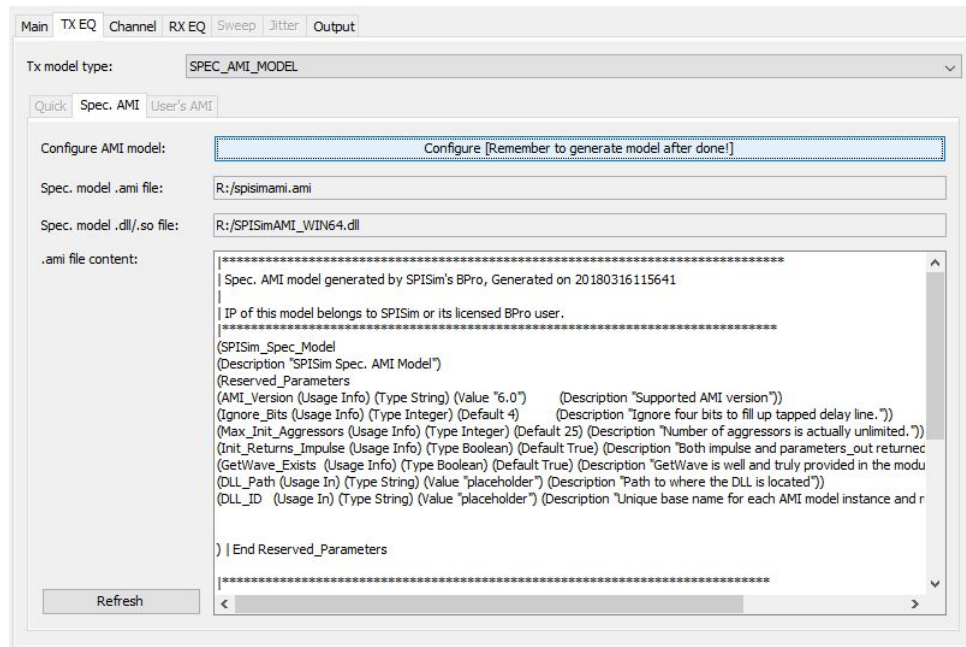
Cursor	Min	Step	Val	Max
Pre Cursor	-1	0.01	0	1
Post Cursor 1	-1	0.01	0	1
Post Cursor 2	-1	0.01	0	1
Post Cursor 3	-1	0.01	0	1

In “Spec AMI Model” mode, user should click “Configure” button to launch a separate GUI for spec. AMI model configuration. After model is configured and generated, associated .ami and .dll/.so files will be entered into settings automatically. The plain text .ami file will also be shown in the “.ami file content” text area as shown below. The text editor in SPIPro’s main GUI (outside this dialog) will also open the .ami file at the same time. User may perform required customization such as defining variable for sweeping. The modified .ami file need to be saved to have effect in the StatEye simulation.

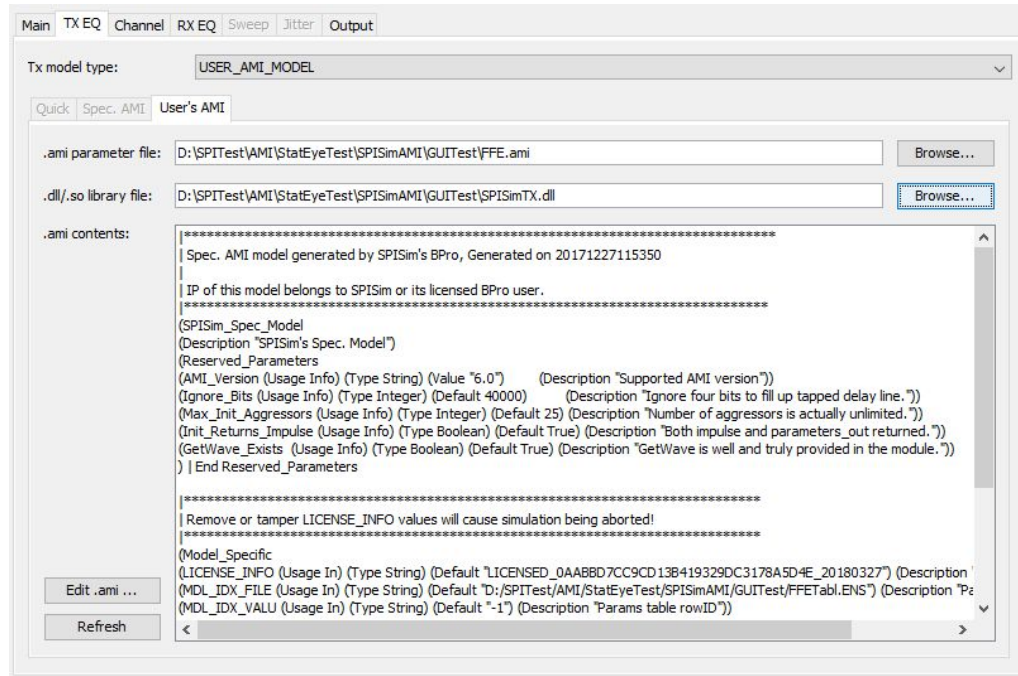
The screenshot below shows the configuration GUI brought up by the “Configure” button:



“Refresh”: will reload the original or modified .ami content into the text area.



In “User’s AMI Model” mode, user click “Browse” button to select existing .ami and .dll/.so files. These model files can be from either SPISim or other model vendors. The selected .ami file content will be displayed in the “.ami contents” text area as shown below:



“Edit .ami”: will open a SPIPro text editor window (outside this StatEye GUI) allowing user to further edit the selected .ami file. Modified .ami content should be saved first to see the effect in the StatEye simulation.

“Refresh”: reload the .ami file content into the text area.

Channel: when enabled, this tab specifies the channel settings.

Similar to either TX EQ or RX EQ, three modes have be provided for channel model. User select either “Quick Settings”, “Waveform” or “Spice Subckt” from the drop-down list at the top and only the selected sub-tab will be enabled for further configuration.

In “Quick Settings” mode, user can specify the Tx and Rx analog front end parameters such as impedance and C-Comp. Settings for interconnect will be used to construct a transmission line model. These IBIS parameters and interconnect settings will then be used to create an impulse response which represent the channel model

The screenshot shows the 'Channel' tab in the SPISim configuration window. The 'Channel model type' is set to 'QUICK_SETTINGS'. Below this, there are three sub-tabs: 'Quick', 'Waveform', and 'Spice Subckt'. The 'Quick' sub-tab is active, showing the following parameters:

Tx IBIS:	
Output impedance:	100.0
C_Comp:	0.5p

Interconnect:	
TLine length in meter:	1
Prop. speed in Meter/Sec:	2.0E8
Characterisitc impedance:	100.0
Loss tangent:	0.02

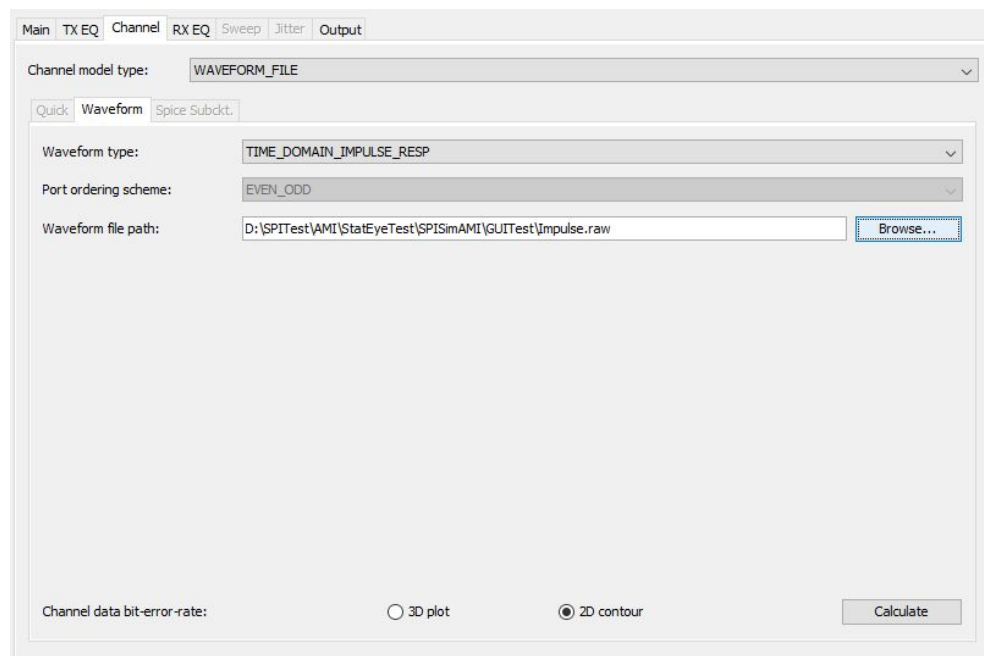
Rx IBIS:	
R of RC termination:	100.0
C of RC termination:	1u
C_Comp	0.5p

In “Waveform File” mode, user provide an existing channel response in the form of waveform file. There are three different format supported:

- Thru channel .s4p: with this s-parameter format being selected, user can further specify the port ordering scheme: “Sequential” means port 1 and 2 are inputs while port 3 and 4 are outputs. “Even Odd” means port 1 and 3 are inputs and port 3 and 4 are outputs. Please note that while tool will convert single-ended s-parameter file into differential-mode with corresponding ordering scheme and make use of only Sdd, user needs to condition the given s-parameter to be in good quality (e.g. causal, passive, symmetric etc) using either SPIPro’s SPro function or other alternatives.
- Impulse response: user can provide impulse response in either .csv, .raw (Berkeley spice or LTSpice compressed format) or HSpice’s tr0 format. For .csv format, the first row should be header and data from only the first two columns are used. The first column should be time points and second column is the response. They do not need to be uniform spaced as tool will perform interpolation internally. For HSpice, please make sure the following simulation options are added so that output waveform format can be processed by the tool.

```
7 .option PROBE POST
8 .option POST_VERSION=9601
```

- Step response: user may also specify the step response of the channel. The assumption is that the time it takes for stimulus to drive from low to high is one bit time divided by number of samples per bit. Tool will perform differentiation internally to obtain the associated channel impulse response.



In “Spice Subckt” mode, user define a spice .sub-circuit with four terminals: two inputs and two outputs. Tool will create a driving circuit, i.e. top level netlist, to drive this user specified subckt. Step response will be applied to two input terminals and response at the outputs with high-Z assumption will be measured. User can also specify the simulator to be used and the provided .subckt format must match the syntax supported by that spice simulator. User may need to provide grounding within the subckt as tool does not assume two inputs are either single ended or differential.

“Spice simulator”: select simulator to be used. All but the SPIPro built-in “SSolver” will enable the selection of path to this particular simulator. Please note that user is responsible to have appropriate license (e.g. for HSpice) and environment set-up. For LTSpice, please select the “XVIIx64.exe” or “SCad*.exe” of older version.

“Path to the simulator”: click “browse” to choose path to the simulator. Path value will be entered automatically.

“Spice file ...”: browse to choose the user defined, four terminal .subckt file. All .subckt name will be parsed by the tool upon completion.

“.subckt name”: select the name of the subcircuit to be instantiated by the tool in dynamically created top-level spice netlist.

- RX EQ: when enabled, this tab specifies the RX EQ settings.

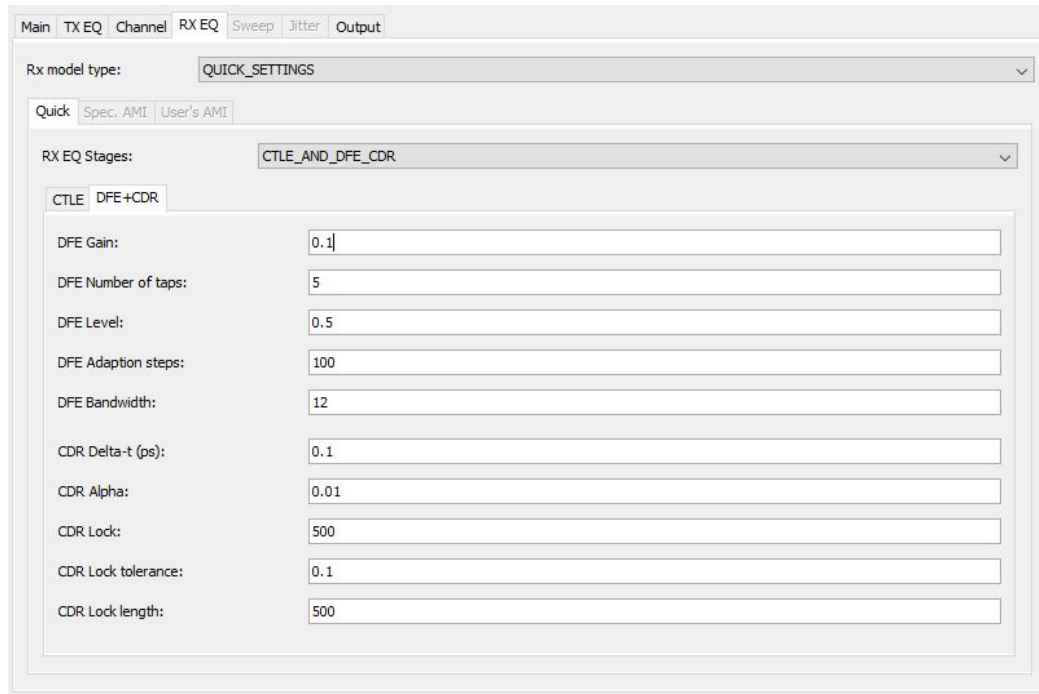
Similar to TX EQ tab, three modes have been provided for user's easy configuration. The operations of both "Spec AMI Model" and "User's AMI Model" are exactly the same as those in TX EQ, thus only the "Quick settings" model is described below.

In "Quick Settings" mode, user can specify RX EQ's composing stages: both CTLE and DFE/CDR or only one of them. Please note that DFE/CDR is NLTV and thus with it being part of the channel, "Statistical" simulation mode will not be supported.

For CTLE, user can specify its behaviors in one of the three ways: give a frequency response file, using PCIe like parameters or locations of poles and zeros.

The screenshot shows the RX EQ configuration window. The 'RX EQ' tab is selected, and the 'QUICK_SETTINGS' model is chosen. Under the 'Quick' sub-tab, the 'RX EQ Stages' are set to 'CTLE_AND_DFE_CDR'. The 'CTLE' sub-tab is active, showing various parameters for CTLE configuration, including operation mode (PASSIVE), data type (PARAMS), and several numerical fields for frequency response and bandwidth.

For DFE/CDR, user specify the parameters for adaptation/slicing of DFE and phase-detectors of CDRs. For more information about these settings, please refer to section 3, Spec. AMI Model descriptions.



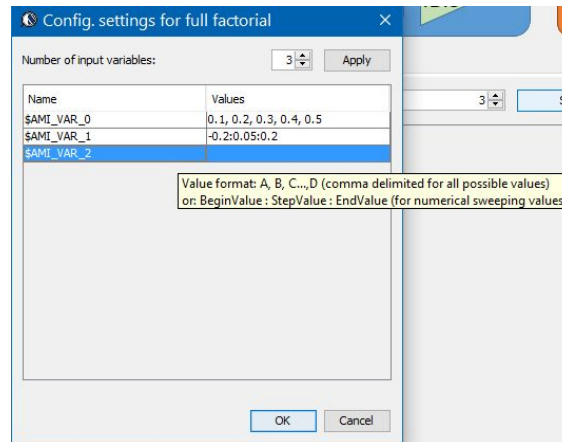
The screenshot displays the 'RX EQ' configuration window in the SPISim StatEye/AMI software. The window features a tabbed interface with 'Main', 'TX EQ', 'Channel', 'RX EQ', 'Sweep', 'Jitter', and 'Output'. The 'RX EQ' tab is selected, revealing a 'Rx model type' dropdown menu set to 'QUICK_SETTINGS'. Below this, there are three sub-tabs: 'Quick', 'Spec. AMI', and 'User's AMI'. The 'RX EQ Stages' dropdown is set to 'CTLE_AND_DFE_CDR'. Under this stage, there are two sub-tabs: 'CTLE' and 'DFE+CDR'. The 'DFE+CDR' sub-tab is active, showing a list of parameters with corresponding input fields:

Parameter	Value
DFE Gain:	0.1
DFE Number of taps:	5
DFE Level:	0.5
DFE Adaption steps:	100
DFE Bandwidth:	12
CDR Delta-t (ps):	0.1
CDR Alpha:	0.01
CDR Lock:	500
CDR Lock tolerance:	0.1
CDR Lock length:	500

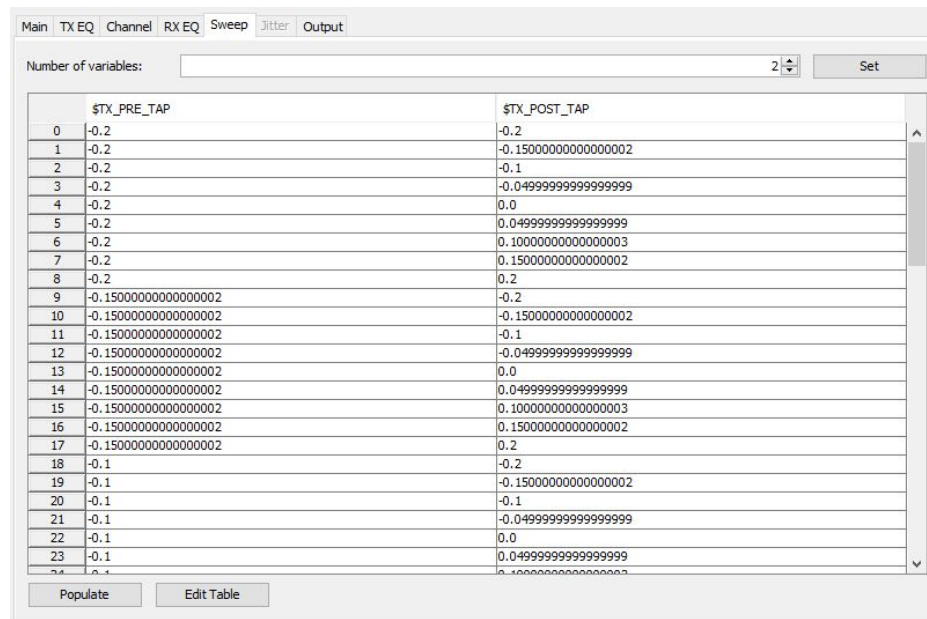
- Sweep: when enabled, this tab allows user to configure .ami parameter sweeps.

To perform sweep, first user need to define variable “pattern” in the participating TX and/or RX .ami file. These pattern will be replaced on the fly for each of the sweep settings right before simulation.

After selecting total number of sweeping variables in TX and RX .ami files combined, click “Set” button to configure their pattern and sweeping range. The range can be either a list of values or the MIN:STEP:MAX format, as shown below:



Upon clicking “OK”, full combinatorial sweep table will be created. This table will be saved or restored together with the other settings when doing “export” and “import”.



“Populate”: Will transfer table content from SPIPro’s top-level table component to this sweeping table. This is useful when user already created a csv file for parameter sweepings or use tools like SPIMPro, JMP for parameter combination other than full factorial, such as space filling or design-of-experiment. Using this button allowing fully customization of sweeping plan.

“Edit Table”: Will transfer sweeping table content to SPro’s top-level table component for further editing or exporting.

The screenshot below shows an example SPIPro's top-level table components.

AMI Sweeping Table

Wfm:0

Column tasks...

Set default header

Select all columns

Unselect all columns

Invert selection

Add one column

Add multi columns

Delete selected

Sort by value

Rename column.

Goto column...

Row tasks...

Columns:

\$AMI_VAR0

\$AMI_VAR1

\$AMI_VAR2

☒ Reordering allowed
 ☒ Row selection
 ☒ Column selection
 ☐ Recycle Window

Resize mode:

Off

	\$AMI_VAR0	\$AMI_VAR1	\$AMI_VAR2
1	1	1	1
2	1	1	2
3	1	1	3
4	1	1	4
5	1	1	5
6	1	2	1
7	1	2	2
8	1	2	3
9	1	2	4
10	1	2	5
11	1	3	1
12	1	3	2
13	1	3	3
14	1	3	4
15	1	3	5
16	1	4	1
17	1	4	2
18	1	4	3
19	1	4	4
20	1	4	5
21	1	5	1
22	1	5	2
23	1	5	3
24	1	5	4
25	1	5	5
26	2	1	1
27	2	1	2
28	2	1	3
29	2	1	4
30	2	1	5
31	2	2	1
32	2	2	2
33	2	2	3
34	2	2	4
35	2	2	5
36	2	2	1
37	2	2	2
38	2	2	3
39	2	2	4
40	2	2	5
41	2	2	1
42	2	2	2
43	2	2	3
44	2	2	4
45	2	2	5
46	2	2	1
47	2	2	2
48	2	2	3
49	2	2	4
50	2	2	5
51	2	2	1
52	2	2	2
53	2	2	3
54	2	2	4
55	2	2	5
56	2	2	1
57	2	2	2
58	2	2	3
59	2	2	4
60	2	2	5
61	2	2	1
62	2	2	2
63	2	2	3
64	2	2	4
65	2	2	5
66	2	2	1
67	2	2	2
68	2	2	3
69	2	2	4
70	2	2	5
71	2	2	1
72	2	2	2
73	2	2	3
74	2	2	4
75	2	2	5
76	2	2	1
77	2	2	2
78	2	2	3
79	2	2	4
80	2	2	5
81	2	2	1
82	2	2	2
83	2	2	3
84	2	2	4
85	2	2	5
86	2	2	1
87	2	2	2
88	2	2	3
89	2	2	4
90	2	2	5
91	2	2	1
92	2	2	2
93	2	2	3
94	2	2	4
95	2	2	5
96	2	2	1
97	2	2	2
98	2	2	3
99	2	2	4
100	2	2	5

The screenshot below show three sweeping AMI variables defined in an example .ami file.

```
usageInfo (Type Boolean) (Default True) (Description "Getwave is well")
1) (Type String) (Value "placeholder") (Description "Path to where the")
1) (Type String) (Value "placeholder") (Description "Unique base name f")
```

neters

```
*****  
ZENSE_INFO values will cause simulation being aborted!  
*****
```

```

je In (Type String) (Default "LICENSED_SPISIM_D43D7EBA61E0_C86BF02EA8C
-- MAIN Settings -----
je In (Type String) (Default "MOD_TYPE_NRZ") (Description "Modulation
je In (Type String) (Default "FFE") (Description "Cascaded stages"))
-- FFE Settings -----
age In (Type String) (Default "One_PreCursor_Two_PostCursors") (Descri
age In (Type String) (Default "-1.00000,1.00000,0.0100000,$AMI_VAR0")
age In (Type String) (Default "-1.00000,1.00000,0.0100000,$AMI_VAR1")
age In (Type String) (Default "-1.00000,1.00000,0.0100000,$AMI_VAR2")

```

- Jitter: Jitter function will be enabled in future release.

- **Output:** this tab let user specify simulation outputs.

The screenshot shows the 'Output' tab of the SPISim StatEye/AMI configuration window. It includes the following settings:

- Target bit error rate for Eye-Height:** 1E-12
- Ignore first number of bits:** 1000
- Save result waveform to folder:** %SysTemp% (with a 'Browse...' button)
- Generate un-wrapped waveform:** (unchecked checkbox)
- from time (sec.):** 100n
- to time (sec.):** 500n

“**Target BER**”: this value will be used as threshold to measure eye-height at the center of the CDF/bathtub output.

“**Ignore first number of bits**”: the first specified number of bits here will be ignored for EYE and CDF processing.

“**Save result waveform to**”: select the path to store the output waveforms. %SysTemp% will be converted to system’s temporary folder automatically upon simulation.

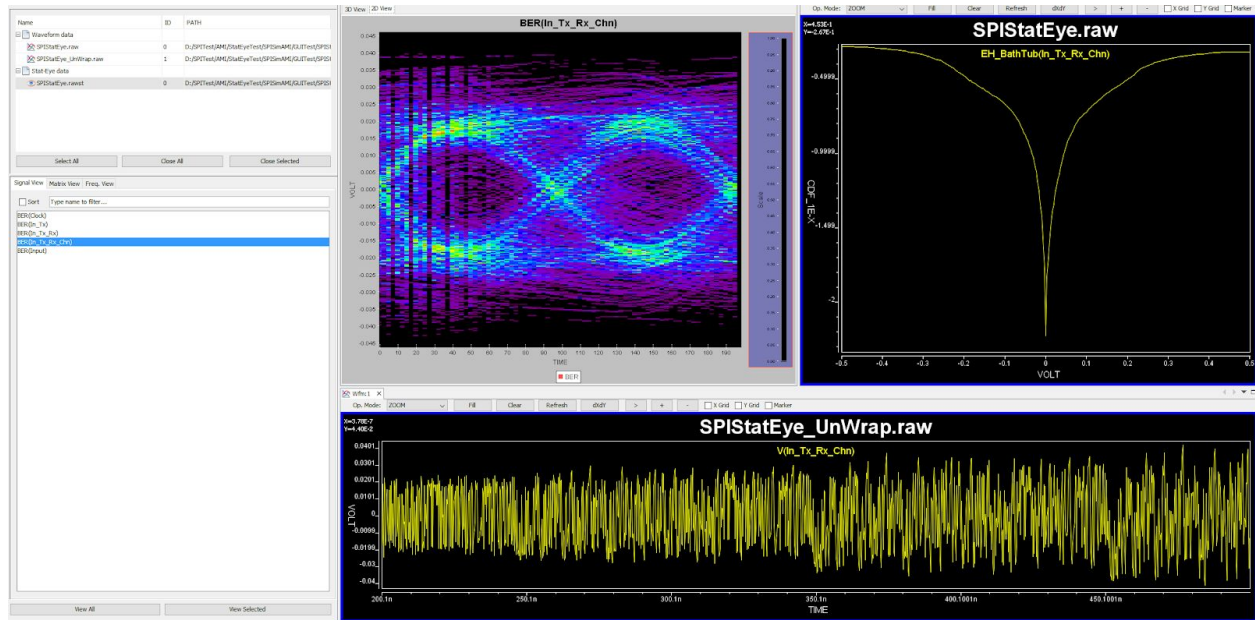
“**Generate unwrapped waveform**”: when checked, specify the range of time window tool should output unwrapped waveform. The output format is Berkeley spice compatible .raw and may be further processed by SPIVPro or other waveform viewer.

Depending on simulation mode, several file types will be generated after simulation. All of them will be prefixed with SPIStatEye with possible sweeping index as a suffix, ends with extension described below:

- SPIStatEye.raw: this is the bath-tub curve
- SPIStatEye_UnWrap.raw: this is unwrapped time-domain waveform
- SPIStatEye.rawst: this is 3d or 2d with color coded eye plot
- SPIStatEye.csv: this is the measurement of eye-height at specified BER

Among which, .rawst is SPISim’s proprietary format while other .raw and .csv are open format. These data may be viewed or analyzed further within SPIPro or other tools.

The screenshot below show example results.



4. Command-line/Batch-mode Flow

SPISim's StatEye flow supports command line or batch mode on either Windows 64 or Linux 64 bit environment. The GUI elements mentioned in section two facilitates the parameter settings of models of different stages. Upon simulation starts, these settings will be exported into a config. file to invoke native coded simulator in the backend. This section describes this process.

The StatEye simulator is a single executable called "SPISimAMI_XXXX.exe" within installed folder. XXXX is the operating system plus bit type. So for windows 64 bit, the executable is called SPISimAMI_Win64.exe

The command to invoke StatEye flow is:

SPISimAMI_XXXX.exe StatEye StatEye.cfg

The screenshot below shows command line StatEye run using both SPISim AMI models at the TX EQ and RX EQ.

```
D:\SPITest\AMI\StatEyeTest\SPISimAMI\GUITest>SPISimAMI.exe StatEye SPISatEye.cfg
I[019]: Running stateye analysis using config. SPISatEye.cfg ...
I[001]: SPISimAMI is licensed to:
I[006]: Parsing AMI parameter file D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/FFE.ami...
I[007]: Loading AMI library file dummyrx.dll...
I[006]: Parsing AMI parameter file D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/dummyrx.ami...
I[007]: Loading AMI library file R:/VSUserDir/x64/debug/SPISimAMI.dll...
I[004]: AMI model is licensed to LICENSED_0AABBD7CC9CD13B419329DC3178A5D4E_20180327
I[009]: <<<<[ SPISim_Spec_Model ]>>>>
I[013]: Loading data from D:/SPITest/AMI/STATEYETEST/SPISIMAMI/GUITEST/FFETABL.ENS...
I[016]: Using parameters from preset/row 15 of table R:\WinTemp\SysTemp\s3ww.0...
I[017]: FFE: set FFE_PARM_PRE1 to -0.1171875 ...
I[017]: FFE: set FFE_PARM_POS1 to 0 ...

----- SPISimModel -----

SPISimModel

Purpose: A SPISim Spec. IBIS AMI model for channel analysis
Support: support@spisim.com or http://www.spisim.com
Version: V1.21 (Built Feb 9 2018 15:43:18)
OS Type: Win64
-----
SPISim Built-in AMI library
I[003]: Status after Setup: OK
I[003]: Status after CalTD: OK
I[003]: Status after Reset: OK

----- SPISimModel -----

SPISimModel

Purpose: A SPISim Spec. IBIS AMI model for channel analysis
Support: support@spisim.com or http://www.spisim.com
Version: V1.25 (Built Mar 14 2018 10:19:33), Win64
License: LICENSED_0AABBD7CC9CD13B419329DC3178A5D4E_20180327
ModelID: SPISim_Spec_Model
-----
I[003]: Status after Setup: OK
I[003]: Status after CalTD: OK
I[003]: Status after Reset: OK
I[009]: Start simulating 1 block(s) for 10000 bits...
I[010]: Done simulating bit block 001 in 4 sec.
I[011]: Generating eye data...
```

StatEye's config file is in plain text format with Key-Value pair settings as shown below. Lines starting with “#” are considered comment lines. It requires proper license info., entered after “STE_LIC_INFO” keyword, to be able to invoke successfully. License is available in either node-locked or non-node-locked time-limited format.

```
#####
# MODULE: SPISim StatEye
# GENERATED ON 20180315104009
#####

# STE_LIC_INFO: Licensing info
STE_LIC_INFO =
# STE_SPC_INFO: Path info etc to spice executable (for channel response)
STE_SPC_INFO = C:/WeiProj/SPISim/Prj/SPISimPrj/build/cluster/modules/ext/ssolver.exe
# STE_SIM_MODE: Simulation mode
STE_SIM_MODE = BIT_BY_BIT
# STE_PRBS_SRC: Type of PRBS
STE_PRBS_SRC = PRBS_10
# STE_NUM_BITS: Number of bits to run
STE_NUM_BITS = 10000
# STE_BIT_TIME: Bit-time
STE_BIT_TIME = 2.0E-10
# STE_SAMP_INT: Sampling interval
STE_SAMP_INT = 5.0000000000000005E-12
# STE_BLK_SIZE: One block = how many bits
STE_BLK_SIZE = 10000
# STE_THRU_WFM: Through waveform file
STE_THRU_WFM = D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/SPI2577364797372209830.raw
# STE_NEXT_WFM: NEXT waveform file
STE_NEXT_WFM =
# STE_FEXT_WFM: FEXT waveform file
STE_FEXT_WFM =
# STE_PORT_ODR: Port ordering
STE_PORT_ODR = EVEN_ODD
# STE_IGN_BITS: Number of bits to ignore at the beginning
STE_IGN_BITS = 1000
# STE_BER_TARG: Target BER for EH measurement
STE_BER_TARG = 1E-12
# STE_OUT_BASE: Output base (including file name but without extension)
STE_OUT_BASE = D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/SPISatEye
# STE_TAMIFILE: Path to the tx ami file
STE_TAMIFILE = D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/FFE.ami
# STE_TDLLFILE: Path to the tx dll file
STE_TDLLFILE = D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/SPISimTX.dll
# STE_RAMIFILE: Path to the rx ami file
STE_RAMIFILE = D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/dummyrx.ami
# STE_RDLLFILE: Path to the rx dll file
STE_RDLLFILE = D:/SPITest/AMI/StatEyeTest/SPISimAMI/GUITest/dummyrx.dll
# STE_UNWRAPTD: Generate unwrapped TD waveform
STE_UNWRAPTD = 1.000E-07,5.000E-07
```