

SystemVue based Automated IBIS-AMI Modeling



Principal Engineer / SerDesDesign.com

2020.09.15

John Baprawski

**SerDes
Design.Com**



Agenda

1. Quick Review: What are IBIS-AMI models and how are they used? p.3
2. IBIS-AMI model development challenges p.6
3. Modeling a SerDes Channel in SystemVue p.8
4. Modeling a SerDes Tx IC with measured data p.12
5. Modeling a SerDes Rx IC with measured data p.22
6. Using SystemVue for SerDes system Channel Simulation p.30
7. The set of models in the SerDesDesign Library p.34
8. Additional SerDes System Examples p.35
9. Conclusion p.37

Appendix

- What are IBIS-AMI models?
- How are IBIS-AMI models used?

1. What are IBIS-AMI models; how are they used? (1/3)

- IBIS-AMI models were introduced by the IBIS Open Forum (www.ibis.org) with their IBIS 5.0 (Aug 2008) specification to address the modeling needs for SerDes systems. Today, the IBIS-AMI specification is at revision 7.0 (Mar 2019).
- A SerDes (Serializer-Deserializer) is a pair of Tx/Rx blocks used in high speed communications to compensate for lossy channel links to maintain an open eye at the receiver data slicer.
 - PCI Express, HDMI, USB and other types of links.
- IBIS stands for Input/output Buffer Information Specification.
 - IBIS-AMI models provide a standardized model interface used by the SerDes chip designers and SerDes system designers.
- Along with the introduction of IBIS-AMI models, a new class of simulator was introduced called a SerDes Channel Simulator; such as the Keysight ADS Channel Simulator.
 - A SerDes Channel Simulator combines frequency domain simulation, time domain simulation, DSP simulation and statistical simulation into one tool.

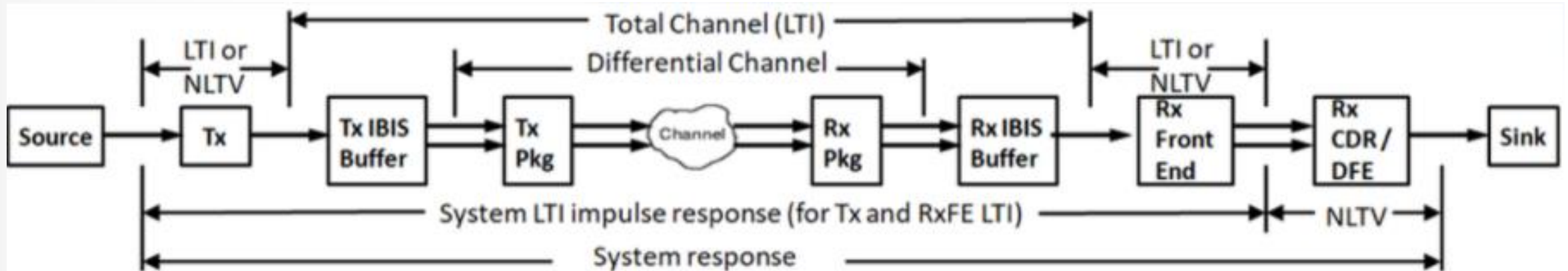
What are IBIS-AMI models; how are they used? (2/3)

- A Channel Simulator can quickly and accurately evaluate and optimize the SerDes system for performance.
 - Enables evaluation of margin analysis, robustness of the design, verification of design implementation, and design tradeoffs.
- Before IBIS-AMI, SerDes system designers used traditional SPICE-based analysis.
 - Slow and could not simulate the millions of bits.
- With IBIS-AMI models and Channel Simulators, millions of bits can be simulated.
 - Effects of inter-symbol interference (ISI),
 - jitter (deterministic, D_j ; Gaussian, R_j), and
 - cross-channel interference, and more.

What are IBIS-AMI models; how are they used? (3/3)

Here is a typical single channel layout for a SerDes system used in a Channel Simulator.

- Source: NRZ or PAM4 data source
- Tx: Transmitter equalization; typically with an FFE.
- Tx/Rx IBIS Buffers: IBIS buffer to the differential channel; defines the on-die impedance.
- Tx/Rx Pkg: Package characteristic; typically defined with an SnP file.
- Channel: SerDes channel; typically defined with an SnP file.
- Rx Front End: Receiver equalization; typically with a CTLE.
- Rx CDR/DFE: Receiver timing/equalization; typically with a CDR and DFE.



2. IBIS-AMI model development challenges

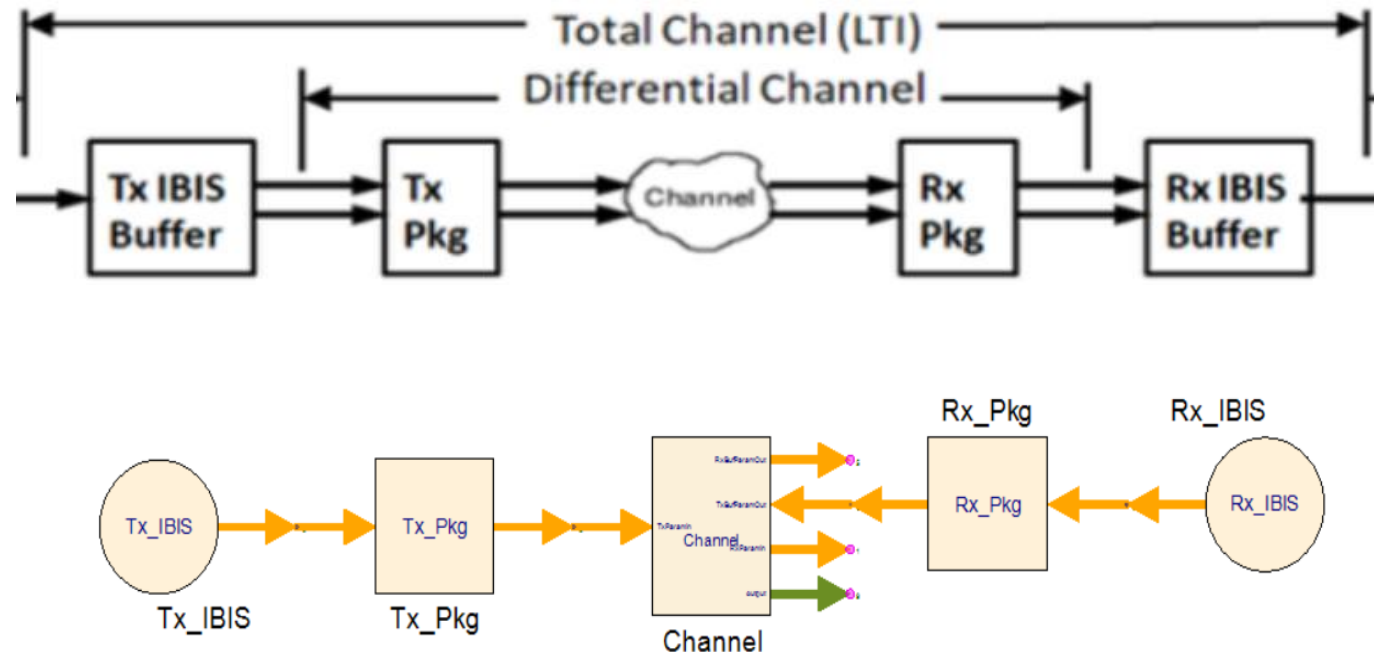
- IBIS-AMI models and Channel Simulators have become an integral part of SerDes system design.
 - SerDes vendors routinely provide IBIS-AMI models for their chips; often before the final chips.
 - End users depend on high fidelity IBIS-AMI models and an accurate channel simulator to be able to predict their system behavior in their simulations.
- IBIS-AMI models are highly configurable.
 - Model developers can design their models so that the equalization settings, adaptation loop parameters and corner cases can be chosen by the user.
- Key objective: IBIS-AMI models to accurately match the hardware performance in a real system.
- Therein lies the challenge.

Key decisions on the IBIS and AMI models

- Key decisions required to convert Tx/Rx circuits into their IBIS-AMI representations.
- One can split the decisions as follows:
 - 1) Partition the Tx/Rx design into a signal flow suitable for IBIS-AMI model representation.
 - This can be quite a challenging task and requires iterative discussion between the model developer and circuit developer to achieve a common understanding.
 - 2) Provide the IBIS-AMI models with parameters that represent the Tx/Rx circuit control flow.
 - This task is to define all the states of the circuit to be represented in the model and define a set of parameters associated with each state.
 - Each state is represented with LTI and/or NLTV functionality by the model developer.
- The above decision process is not discussed in detail here.
- Tx/Rx IBIS buffer designs are assumed to properly represent the impedance loading the channel.

3. Modeling a SerDes channel in SystemVue

- For SerDes system simulation in SystemVue, we need to first discuss modeling SerDes channels.
- The SerDesDesign Library enables modeling channels using S-parameter files and IBIS buffers.
- The equivalent SystemVue DataFlow schematic is shown.
 - Each block represents a frequency domain model.
 - The modeled characteristic includes reflection/reverse transmission effects.

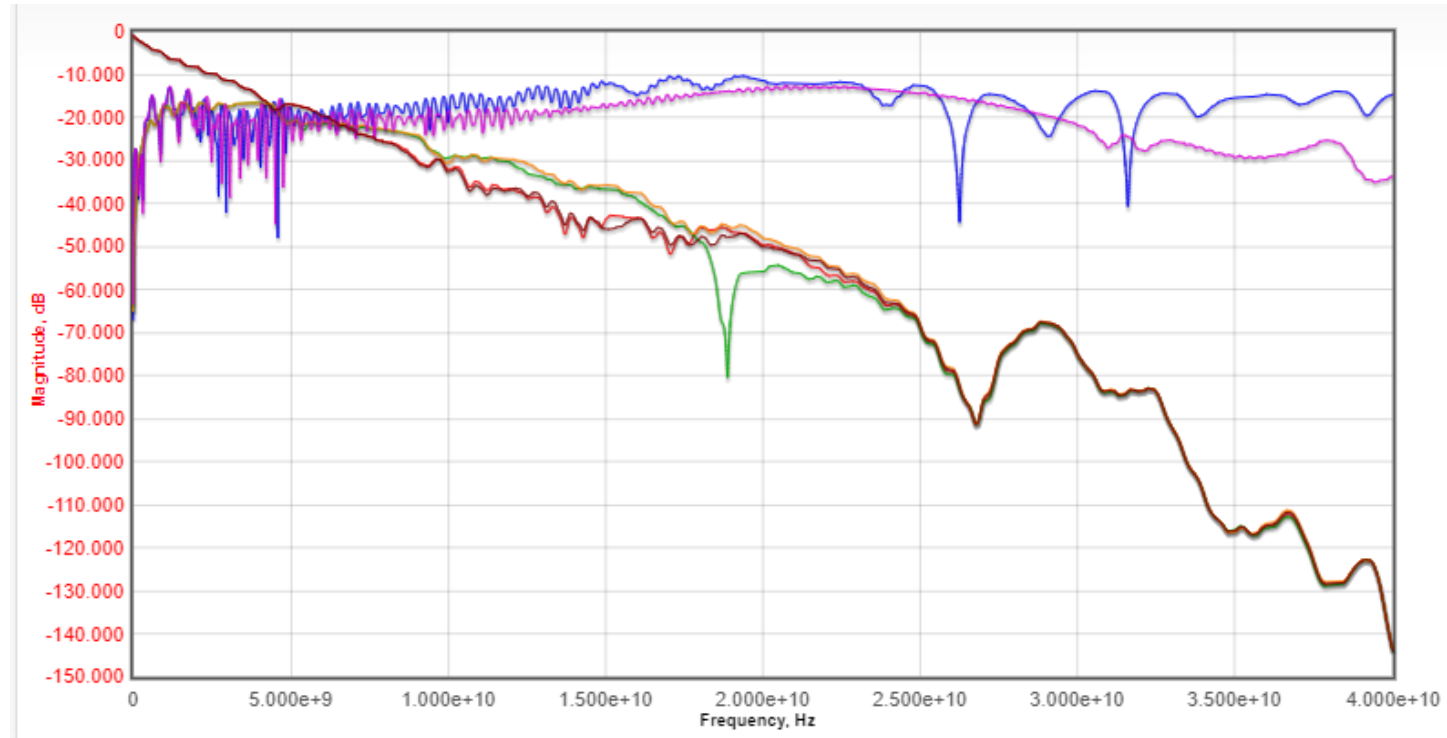


Modeling Tx/Rx IBIS buffers in SystemVue

- IBIS buffers are considered to be LTI by the Channel Simulator.
 - Can be defined for multiple corner cases (Typ, Min, Max)
 - Multiple IBIS buffers can be defined using a Model Selector.
- The Tx_IBIS and Rx_IBIS models allow specifying the IBIS buffer with one of three options:
 1. Use an IBIS file; including corner cases (Typ, Min, Max) and model name.
 - Tx: use of Ramp or Rising/Falling waveform tables.
 2. Use values instead of an IBIS file
 - Tx: Specify Rise/Fall times, VoltageRange, PullUp/PullDown resistance, C_comp capacitance, and R_pkg/L_pkg/C_pkg values.
 - Rx: Specify GND_Clamp resistance, C_comp capacitance, and R_pkg/L_pkg/C_pkg values.
 3. Use an AMI file to define the Alternate IBIS buffer based on the IBIS 7.0 specification.

The Total Channel in SystemVue

- The total channel is derived during Initialization and output from the Channel model (green pin).
- This channel example is from NXP Semiconductor with their permission.
 - S4P data up to 40 GHz.
 - The differential s21dd characteristic:
 - $s21dd = 0.5 * (s21 + s43 - s41 - s32)$.

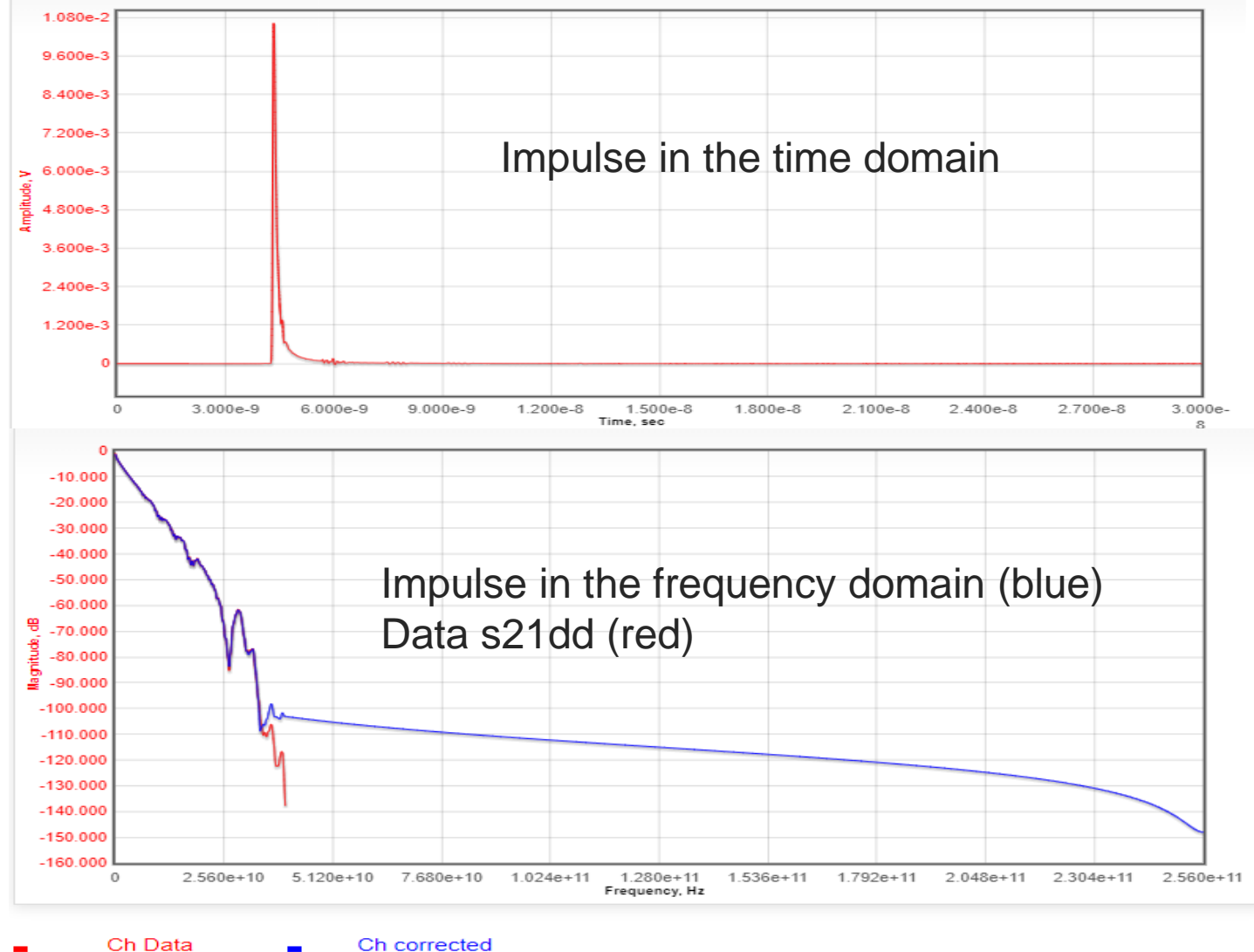


■ S21 ■ S31 ■ S32 ■ S41
■ S42 ■ S43

Input pins: 1 (p), 3 (n) Output pins: 2 (p), 4 (n)

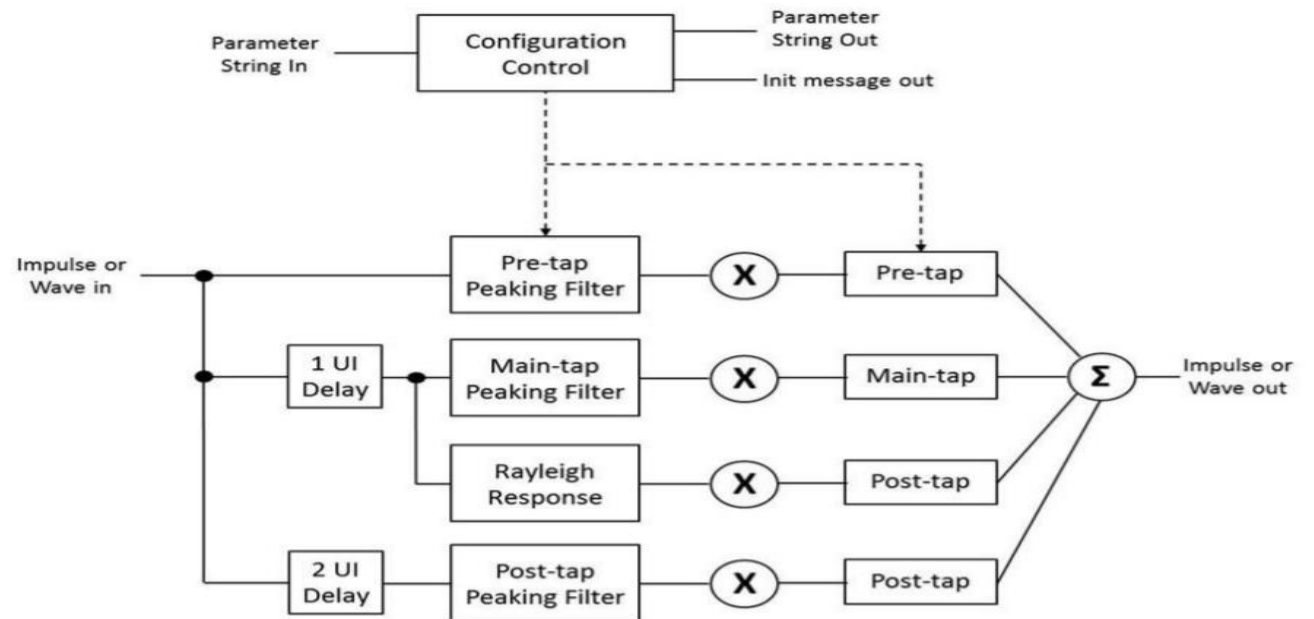
Total Channel Impulse Response

- The derived impulse will be causal and accurately represent s21dd.
 - BitRate = 16 Gbps; SamplesPerBit = 32
- The impulse has a time delay of about 4.3 nsec and is zero before that time delay.
- The impulse has no high frequency aliasing.
- The impulse accurately tracks the S-parameter s21dd data including the “suck out” at 26 GHz.



4. Modeling a SerDes Tx IC with measured data

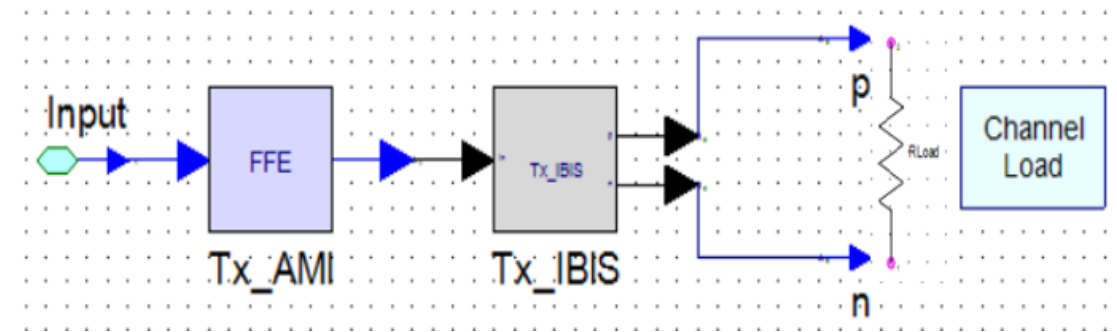
- This NXP Tx circuit is a 3 tap FFE
 - Includes a signal path and control path
 - Multiple filters, tap gains, and on-die impedance structures facing the channel
 - Inherent in the design is a distributed nonlinearity.
 - Bit rates from 1.0 Gbps to 28.05 Gbps.



Example from NXP Semiconductor
with their permission

Tx modeling approach

- Treat the Tx design as a black box.
 - Collect waveforms for all control states of interest.
- Tx control states.
 - Three corner cases: Typ, Slow, Fast
 - Swing level with 14 states
 - Pre-cursor with 25 states (12 negative, 0, 12 positive)
 - Post-cursor with 33 states (16 negative, 0, 16 positive).
 - In total, $14 \times 25 \times 33 \times 3 = 34,650$ states.
- The SerDesDesign Library can handle this many and more states.
- The SerDesDesign approach used uses less characterization waveforms but still support the full set of 34,650 states

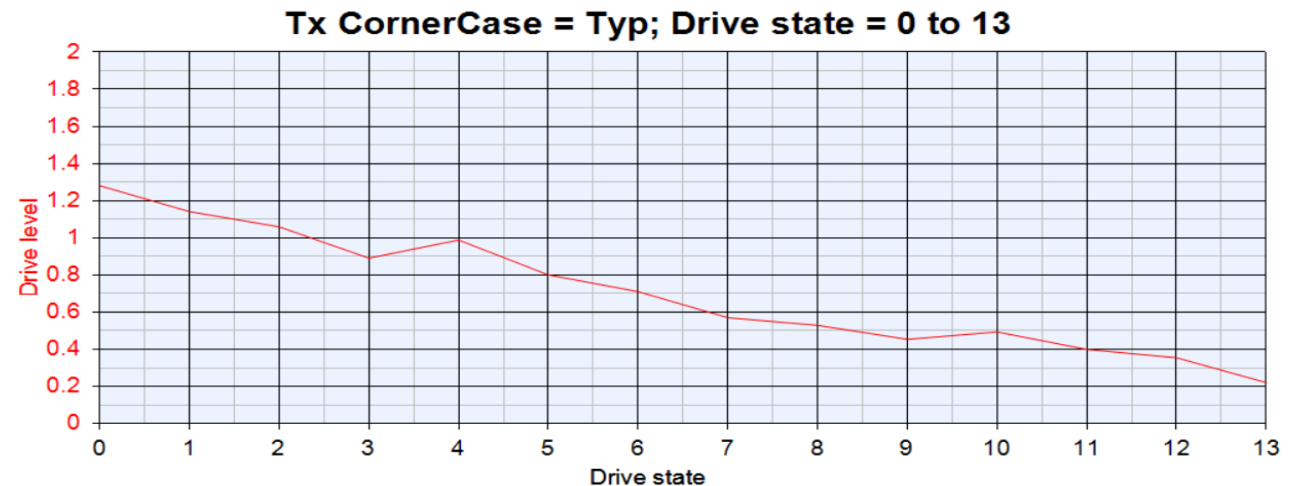
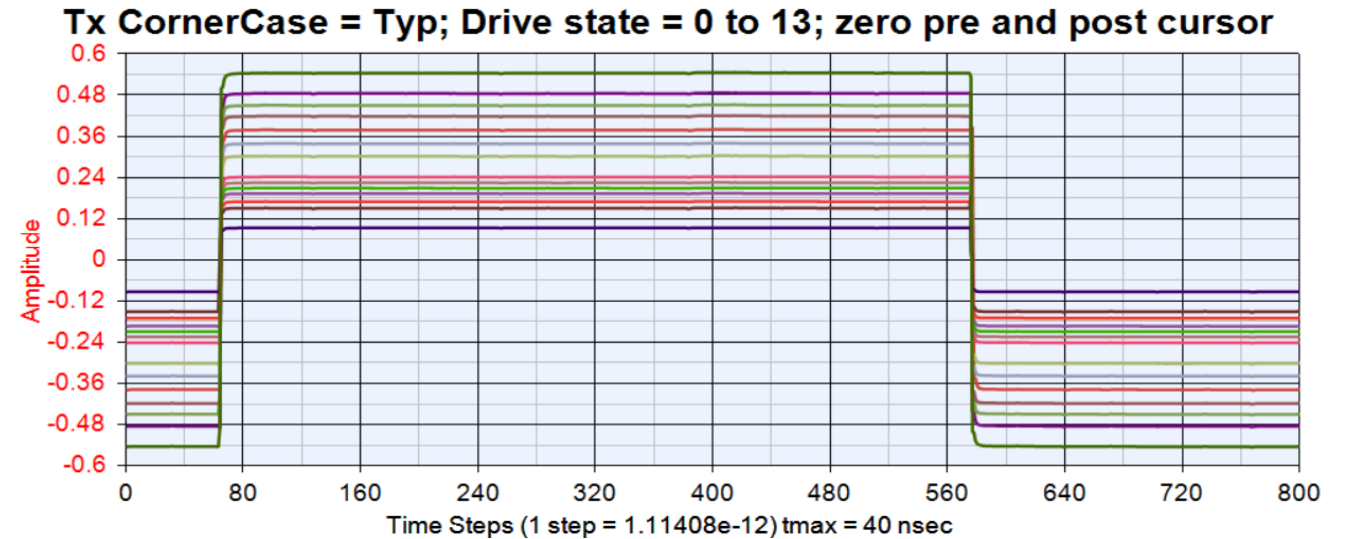


Tx waveforms collected

- The waveforms collected were inclusive of the chip on-die impedance (IBIS buffer).
- Waveforms were recorded at the lowest bit rate (1.0 Gbps) with a time step compatible with the largest bit rate (28.05 Gbps using time step = $1.11408\text{e-}12$ sec).
- Waveforms were recorded while varying the Drive level control parameter over its full defined 14 states for each corner case. This was done with zero pre-cursor and zero post-cursor applied.
 - **For the 3 corner cases there are $14 * 3 = 42$ waveform simulations (files).**
- Though the pre and post cursor states can be either positive or negative, and as agreed upon with NXP, only waveforms were collected for negative pre and post cursor states with reliance on the model for data interpolation and data gain inversion as determined by the model.
- Waveforms were recorded at the maximum Drive level while varying the pre-cursor over its 13 states (zero and negative) and the post-cursor over its 17 states (zero and negative) for each corner case.
 - **For the 3 corner cases, there are $13 * 17 * 3 = 663$ waveform simulations (files).**

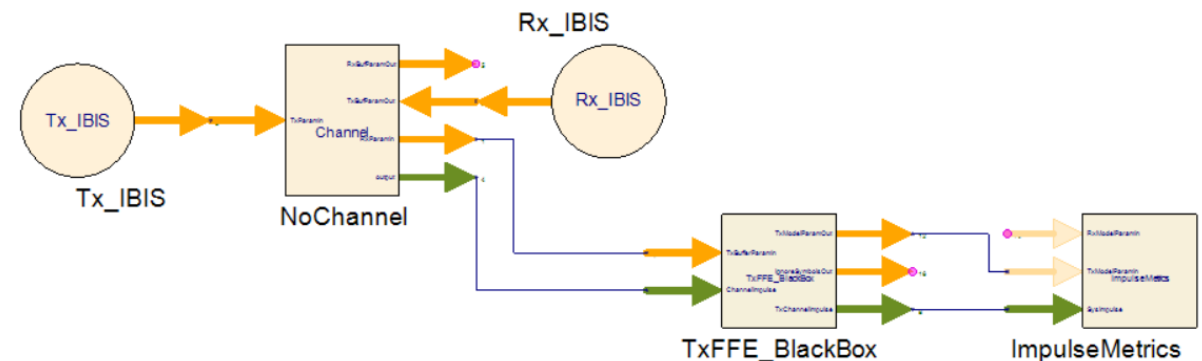
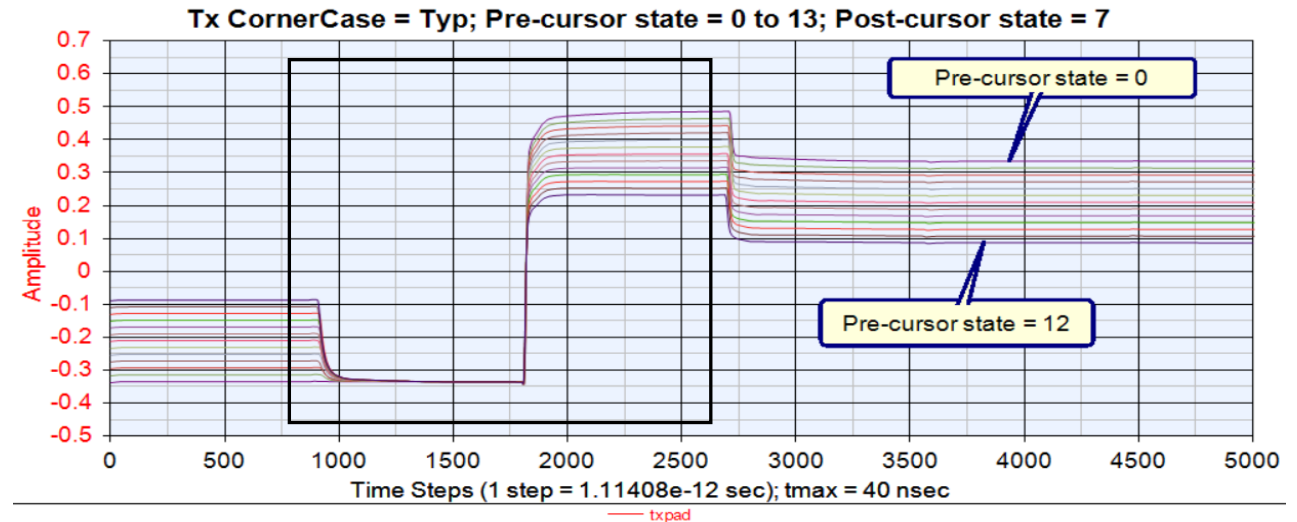
Using the Tx Drive waveform data

- The Drive waveform data is used to derive the nonlinear characteristic versus Drive level.
- The upper figure shows the waveform data collected as the Drive level was varied over its states while pre and post cursor are zero for the Typical corner case.
- The lower figure shows that the Drive level varies nonlinearly versus drive state.
- This nonlinearity is used in the Tx IBIS-AMI model to be generated.



Using the Tx Pre/Post Cursor waveform data

- The upper figure shows waveform data collected for one setting of post-cursor and a sweep of all pre-cursor states.
 - The meaningful data is bounded by the box.
- Next, the IBIS data and waveform data is setup in a SystemVue schematic.
- The lower figure shows the relevant SystemVue schematic.
- Tx_IBIS is set up with an RC circuit for all 3 corner cases.
- There is no channel and Rx_IBIS is setup with ideal 100 differential ohms.



TxFFE_BlackBox model

- The TxFFE_BlackBox model has this parameter view.

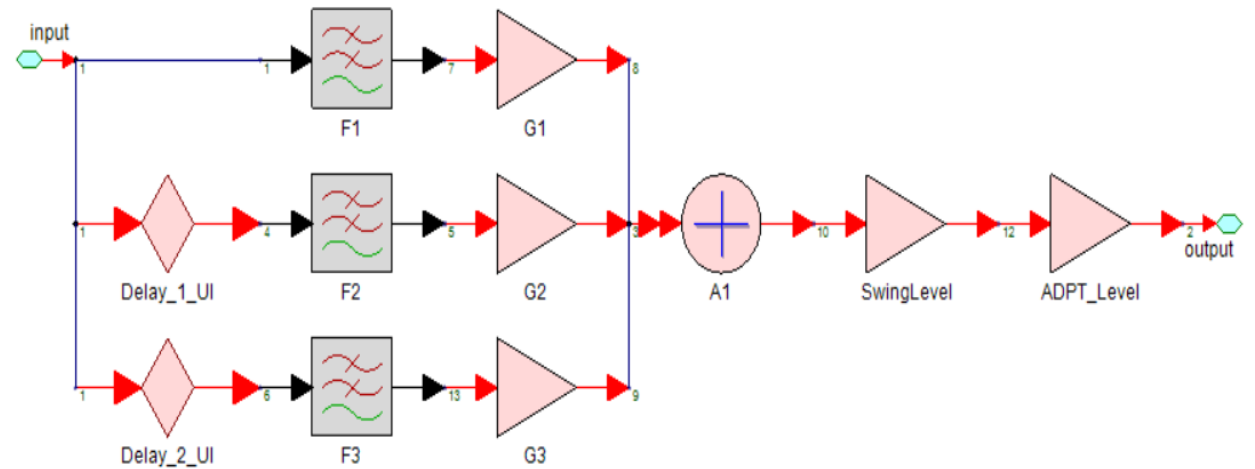
Name	Value
ChannelImpulseData	[1]
SymbolRate	BitRate
SamplesPerSymbol	SamplesPerBit
TxFFE_AdaptForChannel	1:Yes
TxFFE_EnableCorners	1:Yes
TxFFE_CornerCase	0:Typ
TxFFE_OutputGain	1
TxFFE_NumPreCursor1	13
TxFFE_NumDrive	14
TxFFE_NumPostCursor1	17
TxFFEDriveLevels	=NXP_TxFFEDriveLevels
TxFFEDriveLevels_Slow	=NXP_TxFFEDriveLevels_Slow
TxFFEDriveLevels_Fast	=NXP_TxFFEDriveLevels_Fast

Name	Value
TxFFE_PreInverted	0:No
TxFFE_PostInverted	0:No
TxFFE_DisablePrePost	2:Use both
TxFFE_DataBitRate	1.0e9
TxFFE_DataSamplesPerBit	898
UseDataSetOrFile	1:files
DataDir	=DataDir
TxDatFile	=NXP_TxDatFile
TxDatFile_Slow	=NXP_TxDatFile_Slow
TxDatFile_Fast	=NXP_TxDatFile_Fast
EnableTxJitter	0:No
EnableTxBufDeembedding	1:Yes
GenerateIBIS_AMI_Model	1:Yes

ModelBuilderDirName	=ModelBuilderDirName
AMI_SolutionDirName	=AMI_SolutionDirName
IBIS_AMI_ModelName	nxp_lynx37_tx
ExitAfterModelGeneration	0:No

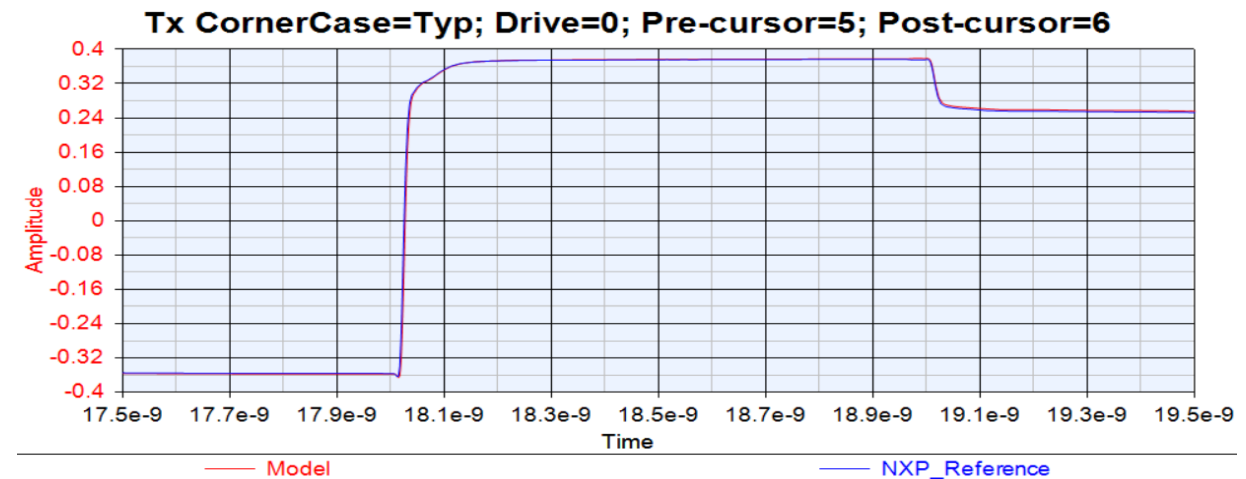
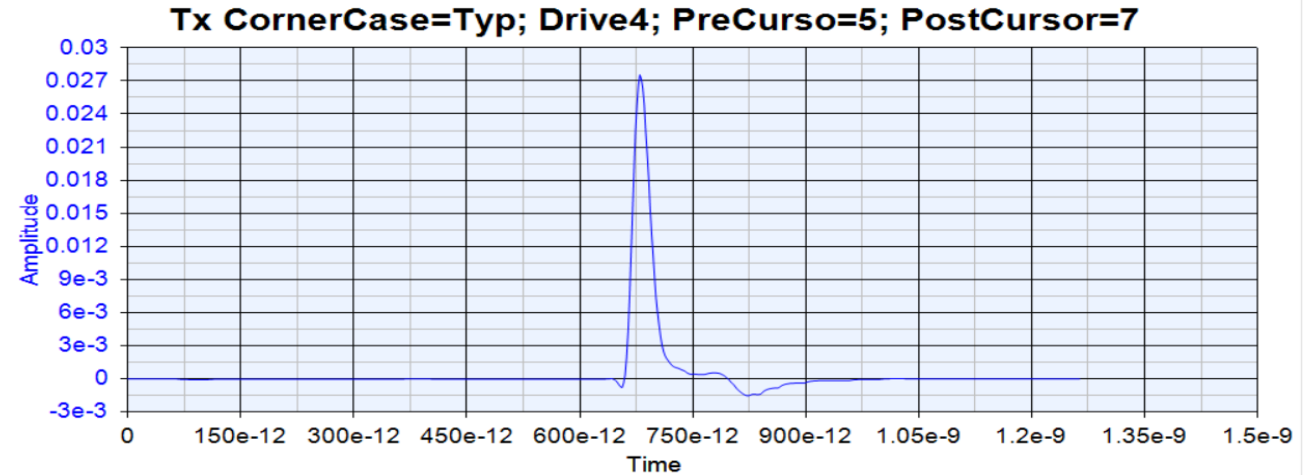
TxFFE_BlackBox model operation

- When the TxFFE_BlackBox model is run the following happens:
 - Collect the total channel impulse response.
 - Collect the Tx_IBIS compulse response.
 - De-embeds the Tx_IBIS impulse from the Tx waveform data.
 - De-construct the Tx waveform data into the FFE tap gains and FFE filters shown in the figure to the right.
 - Use the constructed FFE design in the simulation to obtain the Tx impulse response and the impulse response for the combined Tx and total channel.



Example TxFFE_BlackBox results

- Tx model states: Drive=0, PreCursor=5 and PostCursor=6.
- BitRate=16 Gbps, SamplesPerBit=32
 - The upper plot shown the resultant Tx model output impulse (which includes the Tx IBIS characteristic).
- BitRate=1 Gbps, SamplesPerBit=898
 - The bottom plot compares the Tx FFE response to the original NXP reference waveform data.
 - There is an exact match.



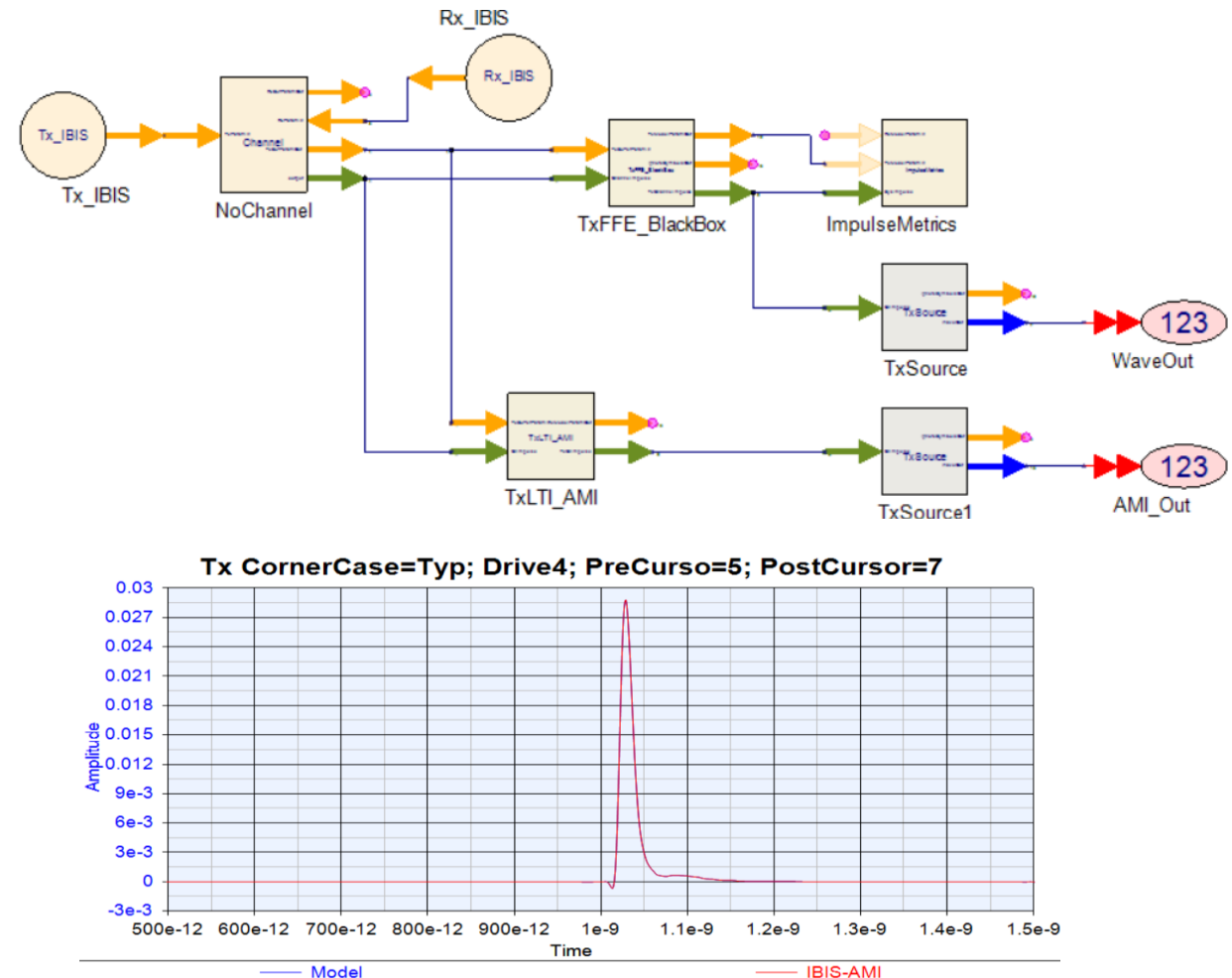
TxFFE_BlackBox IBIS-AMI model generation

- The PC must have: free Microsoft Visual Studio Community Edition C++ product.
- The following happens when run with `Generate_IBIS_AMI_Model = Yes`
 - Messages shown to the right are displayed in the SystemVue Simulation Log.
 - IBIS-AMI model files (*.ibs, *.ami, *_x64.dll) are generated.
 - The *.ibs file includes the IBIS buffer defined for all three corner cases.
 - The *.dll contains all the waveform data for all three corner cases.

Generate IBIS-AMI model started.
Create AMI_Solution directory.
Create source directory.
Create AMI-nxp_lynx37_tx directory.
Create Configure-for-Linux-Makefiles.sh file.
Create Configure-for-win64-vs2015.bat file.
Create source\CMakeLists.txt file.
Create AMI-nxp_lynx37_tx\CMakeLists.txt file.
Create nxp_lynx37_tx.ibs file.
Create nxp_lynx37_tx.ami file.
Create nxp_lynx37_tx_AMI.h file.
Create nxp_lynx37_tx_AMI.cpp file.
Success in generating custom IBIS-AMI model files.
The generated IBIS-AMI files (nxp_lynx37_tx.ibs/ ami/ _x64.dll) are in directory C:\AMI\AMI_Solution\output-vs2015\Release-AMI.

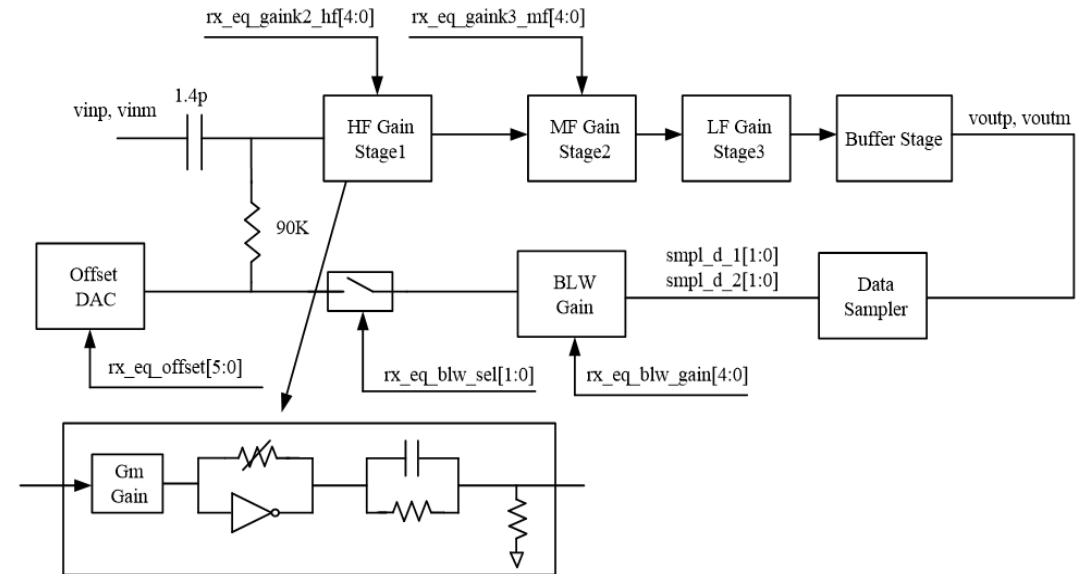
Use the generated IBIS-AMI model in SystemVue

- The generated IBIS-AMI model (or any IBIS-AMI model) can be used in SystemVue using (in this case) the SerDesDesign Library model TxAMI_LTI.
- See this revised SystemVue schematic.
- The TxAMI_LTI model receives information from the Tx_IBIS model that identifies the AMI and DLL files referenced in the *.ibs file.
- The impulse responses for the TxFFE_BlackBox and TxAMI_LTI models are identical.

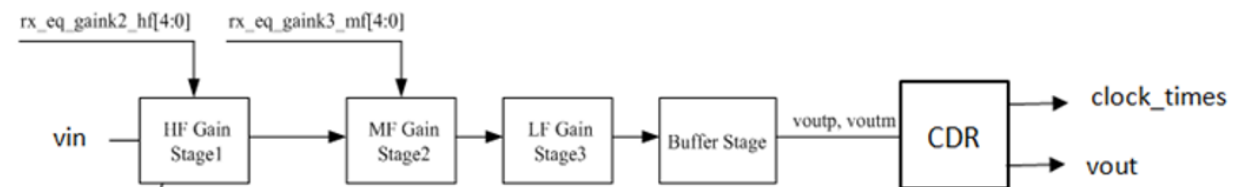


5. Modeling a SerDes Rx IC with measured data

- This NXP Rx circuit is defined for operability with NRZ data for bit rates from 1.0 Gbps to 28.05 Gbps divided into 7 bit rate ranges.
- The lower block diagram is simplified for IBIS-AMI modeling purposes.
 - The Offset DAC and BLW Gain are not used in the IBIS-AMI model.
- The four stages plus CDR are treated as black boxes with stimulus/response waveforms captured from which the circuit model can be extracted.

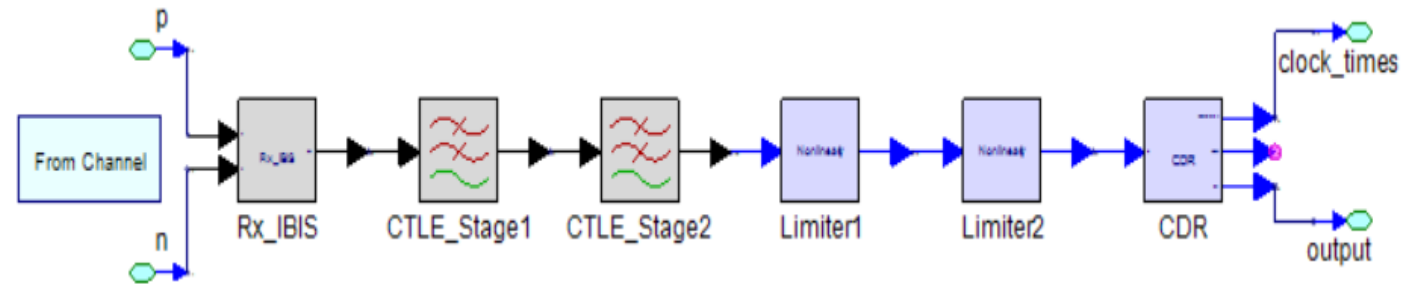


Example from NXP Semiconductor
with their permission



Rx modeling flow

- Treat the CTLE stages as black boxes.
 - Measure data from differential input to each CTLE output for all CTLE states.
- Treat the Limiter stages as black boxes.
 - Measure I/O nonlinear response and I/O filtering response for all Limiter states.
- CDR requires the JTF corner frequency for all CDR states.
- Data is used in SerDes Design Library models.



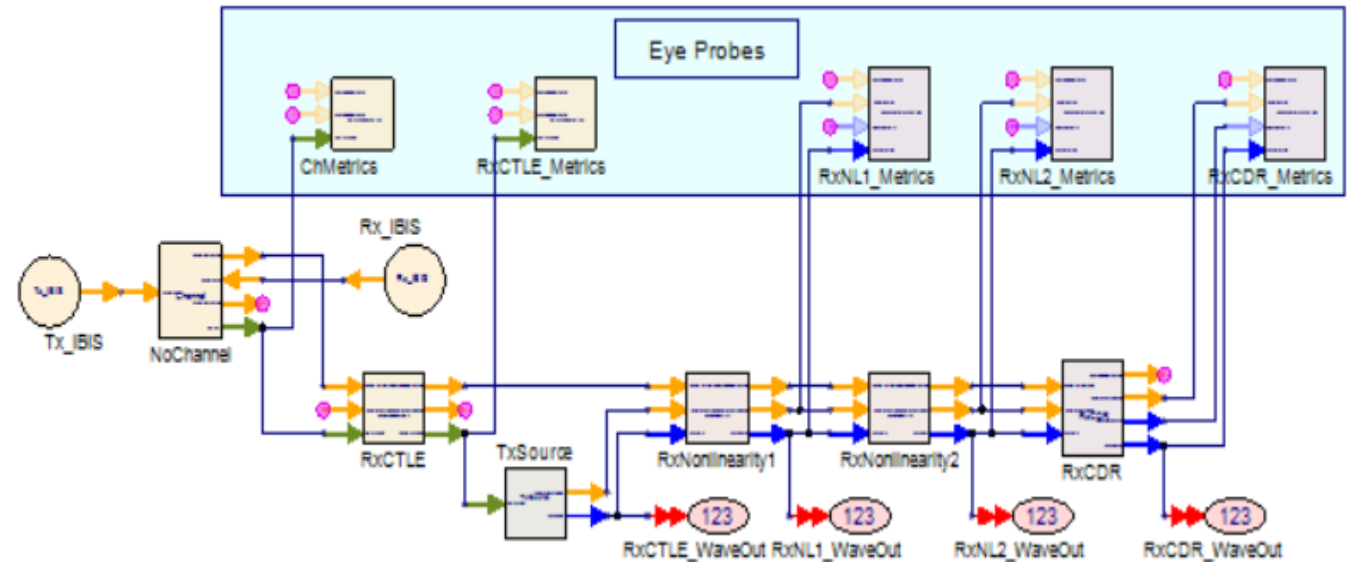
Example from NXP Semiconductor
with their permission

Rx waveforms collected

- The waveform stimulus is recorded in addition to the waveform response waveforms.
- Waveforms were recorded at the highest bit rate for each of the 7 bit rate ranges for each of the 3 corner cases using SamplesPerBit=32.
- Waveforms were recorded while varying the control parameters over their full defined states for each corner case.
- For the HF Gain Stage 1: there are $32 \times 3 \times 7 = 672$ waveform files.
 - These files are inclusive of the chip on-die impedance (Rx IBIS buffer).
- For the MF Gain Stage 2: there are $32 \times 3 \times 7 = 672$ waveform files.
- For the LF Gain Stage 3: there are $3 \times 7 = 21$ waveform files and 21 nonlinearity files.
- For the Buffer Stage 4: there are $3 \times 7 = 21$ waveform files and 21 nonlinearity files.
- The nonlinearity is defined using DC vout vs vin characteristic.
- The CDR model is based on a bang-bang type of phase detector from the SerDesDesign Library.
 - The CDR can be switch On or Off.

Using the Rx data in a SystemVue schematic

- To the right is the schematic.
- Rx_IBIS contains the NXP specified buffer configuration for all 3 corners.
- Tx_IBIS is defined with 100 differential ohms.
- For now, there is NoChannel; a channel will be used later for SerDes system modeling.
- RxFE_CTLE is the Rx front end CTLE and contains the Stage 1 and 2 data.
- Nonlinearity 1/2 contain the Stage 3/4 data.



RxFE_CTLE model

- The RxFE_CTLE model has this parameter view.

Name	Value	Name	Value		
ChannelImpulseData	[1]	RxFE_Data1Index	31		
SymbolRate	BitRate	RxFE_Data1File	HFGainStg1_Typ_16Gbps.txt		
SamplesPerSymbol	SamplesPerBit	RxFE_Data1File_Slow	HFGainStg1_Slow_16Gbps.txt		
RxFE_InputGain	1	RxFE_Data1File_Fast	HFGainStg1_Fast_16Gbps.txt		
RxFE_Type	2:2 sections	RxFE_ColumnArrangement2	1:t1, v1, v2, ...		
RxFE_AutoGainControl	0:No	RxFE_NumData2	32		
RxFE_AdaptForChannel	0:No	RxFE_Data2Index	21		
EnableCorners	1:Yes	RxFE_Data2File	MFGainStg2_Typ_16Gbps.txt		
CornerCase	0:Typ	RxFE_Data2File_Slow	MFGainStg2_Slow_16Gbps.txt		
UseDataSetOrFile	1:files	RxFE_Data2File_Fast	MFGainStg2_Fast_16Gbps.txt		
DataDir	=ModelDirName	RxFE_SectionTopology	0:Series sections		
RxFE_ColumnArrangement1	1:t1, v1, v2, ...	EnableRxJitter	0:No		
RxFE_NumData1	32	EnableRxBufDeembedding	1:Yes	GenerateIBIS_AMI_Model	0:No

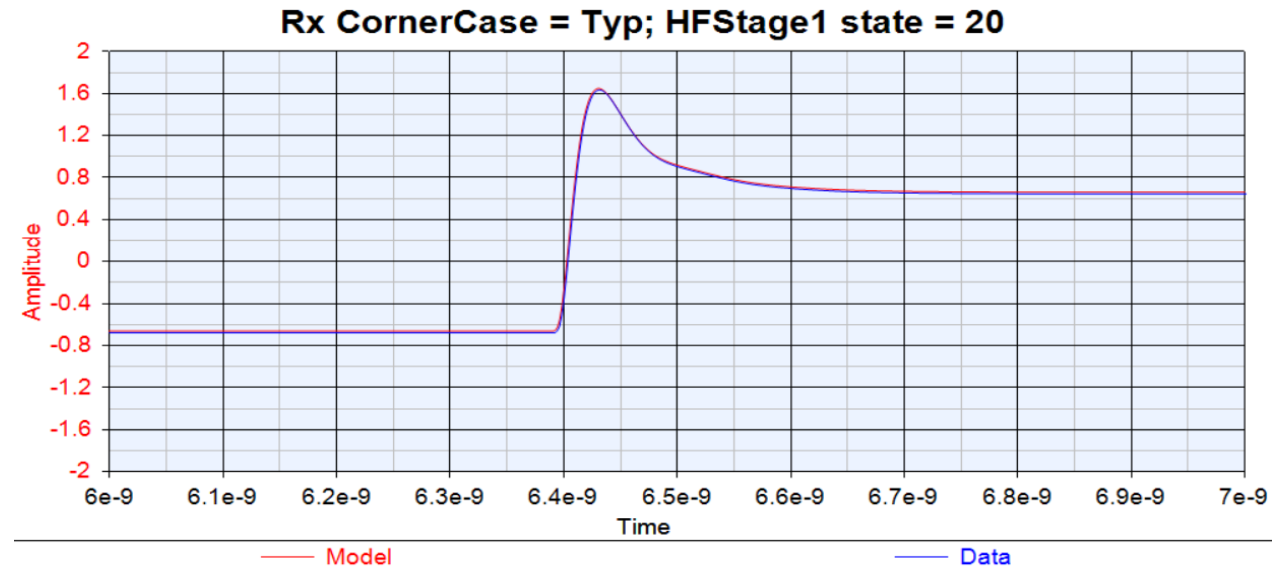
RxNL model

- The RxNL model has this parameter view.

Name	Value		
SymbolRate	BitRate		
SamplesPerSymbol	SamplesPerBit		
ChannelImpulseData	[1]		
DataDir	=ModelDirName		
EnableCorners	1:Yes		
CornerCase	0:Typ	NL1_LPFTType	2:Step response data
NL1_InputGain	1	StepRespDataFileNoDir	LFGainStg3_Typ_16Gbps.txt
NL1_OutputGain	1	StepRespDataFileNoDir_Slow	LFGainStg3_Slow_16Gbps.txt
NL1_NonlinearityType	3:Table data	StepRespDataFileNoDir_Fast	LFGainStg3_Fast_16Gbps.txt
NonlinearityDataFileNoDir	LFGainStg3_Typ_NL_16Gbps.txt	NL1_LPFNorm	3:Remove avg(LPF+NL) gains
NonlinearityDataFileNoDir_...	LFGainStg3_Slow_NL_16Gbps.txt	NL1_FilterTimeScale	0.7
NonlinearityDataFileNoDir_...	LFGainStg3_Fast_NL_16Gbps.txt	BlockSize	BlockSize
NL1_GainSplit	1	GenerateIBIS AMI Model	0:No

Comparing the Model and Data response

- The NRZ waveform from each of the 4 stages can be compared against the reference NXP waveform and nonlinearity data.
- For example, here is the waveform from the HFStage1 for the typical corner for the state = 20 compared against its reference NXP waveform.
 - They are identical



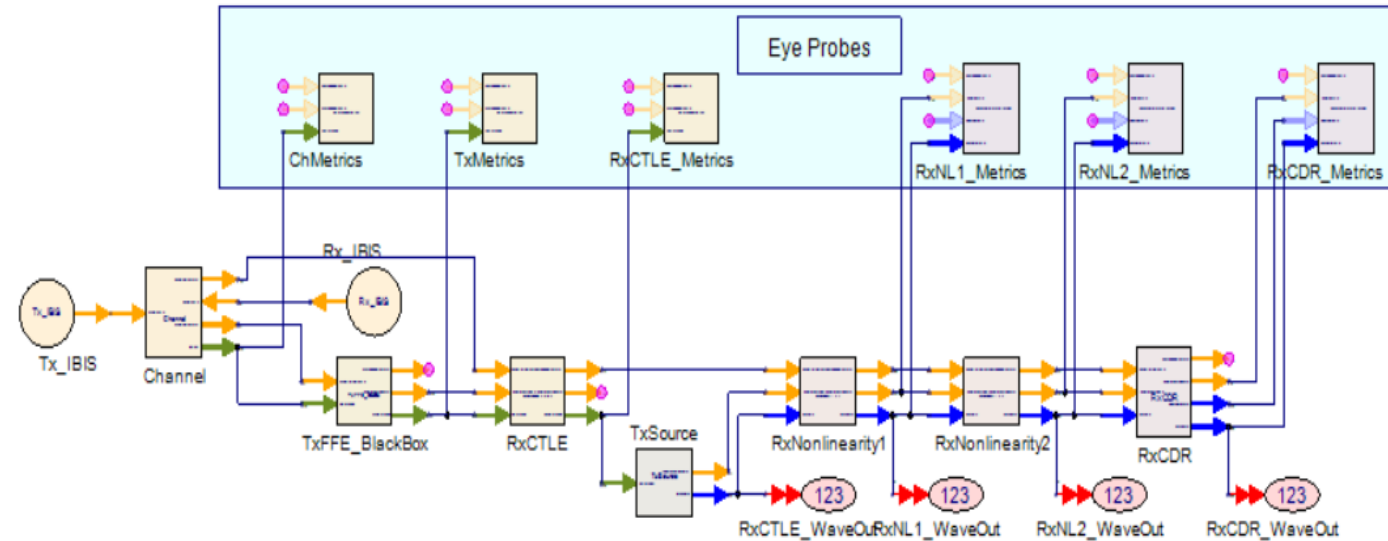
Rx IBIS-AMI model generation

- Run the simulation when the CDR
Generate_IBIS_AMI_Model = Yes
 - Messages shown to the right are displayed in the SystemVue Simulation Log.
 - IBIS-AMI model files (*.ibs, *.ami, *_x64.dll) are generated.
 - The PC must have the free Microsoft Visual Studio Community Edition C++ product.
 - The *.ibs file includes the IBIS buffer defined for all three corner cases.
 - The *.dll contains all the waveform and nonlinearity data for all three corner cases.

Generate IBIS-AMI model started.
Create AMI_Solution directory.
Create source directory.
Create AMI-NXP_Rx_16Gbps directory.
Create Configure-for-Linux-Makefiles.sh file.
Create Configure-for-win64-vs2015.bat file.
Create source\CMakeLists.txt file.
Create AMI-NXP_Rx_16Gbps\CMakeLists.txt file.
Create NXP_Rx_16Gbps.ibs file.
Create NXP_Rx_16Gbps.ami file.
Create NXP_Rx_16Gbps_AMI.h file.
Create NXP_Rx_16Gbps_AMI.cpp file.
Success in generating custom IBIS-AMI model files.
The generated IBIS-AMI files (NXP_Rx_16Gbps.ibs/.ami/_x64.dll) are in directory C:\AMI\AMI_Solution\output-vs2015\Release-AMI.

6. Using SystemVue as a SerDes Channel Simulator

- This SerDes system schematic includes the Tx, Channel and Rx models discussed earlier.
- BitRate=16 Gbps, SamplesPerBit=32
- The total channel includes the channel S4P file, the Tx IBIS buffer and the Rx IBIS buffer. At Nyquist (8 GHz), the total channel has an insertion loss of 18.6 dB.



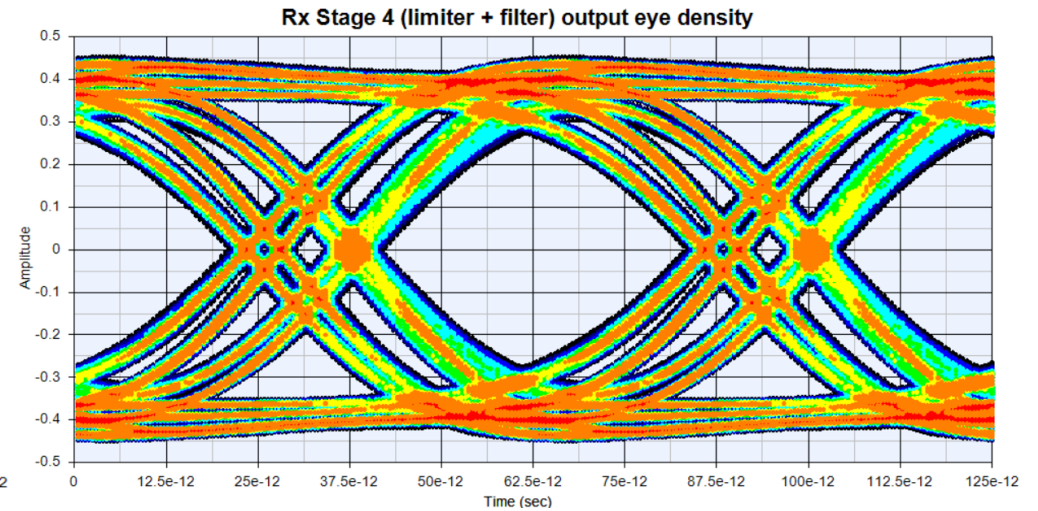
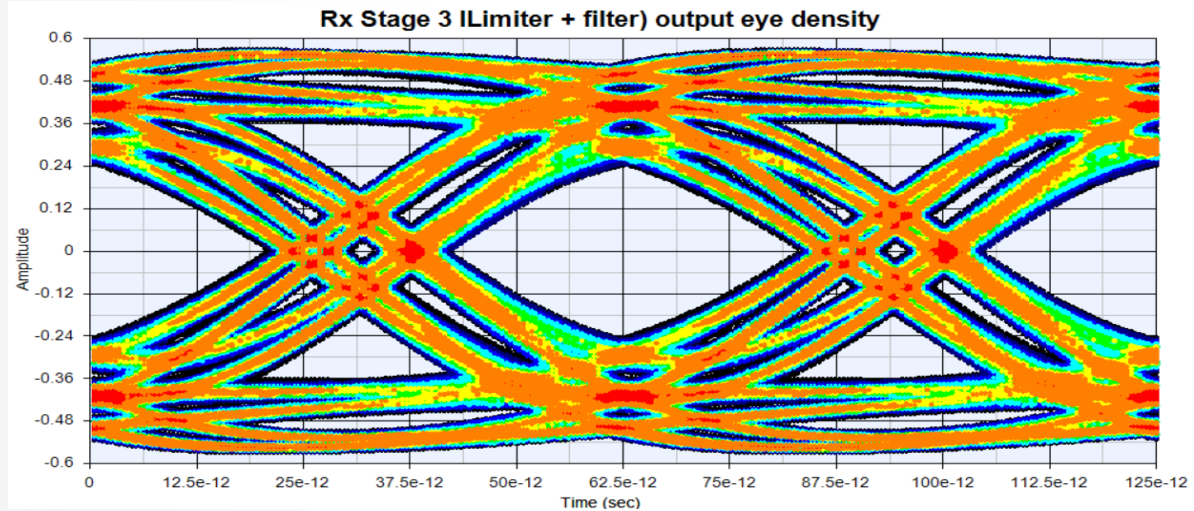
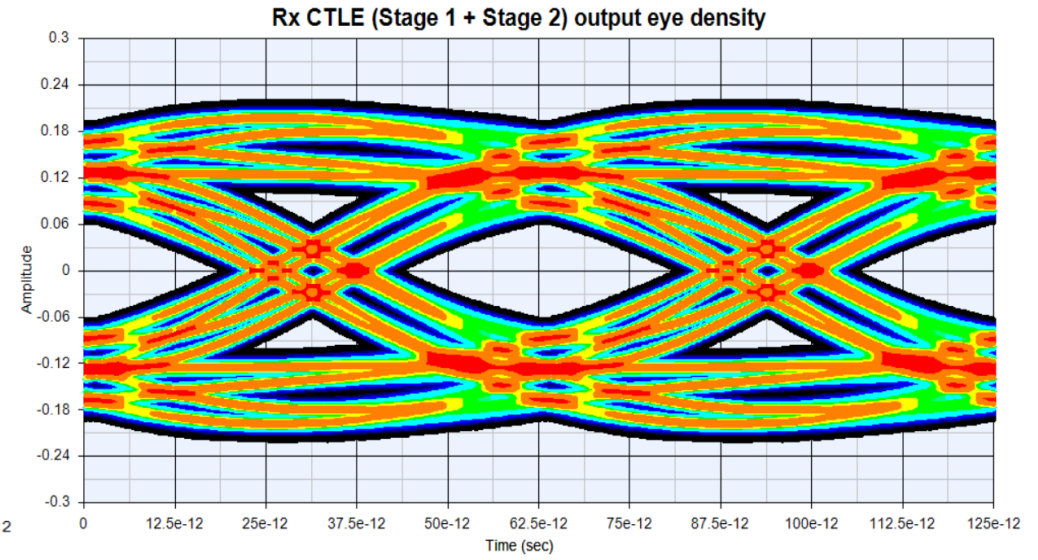
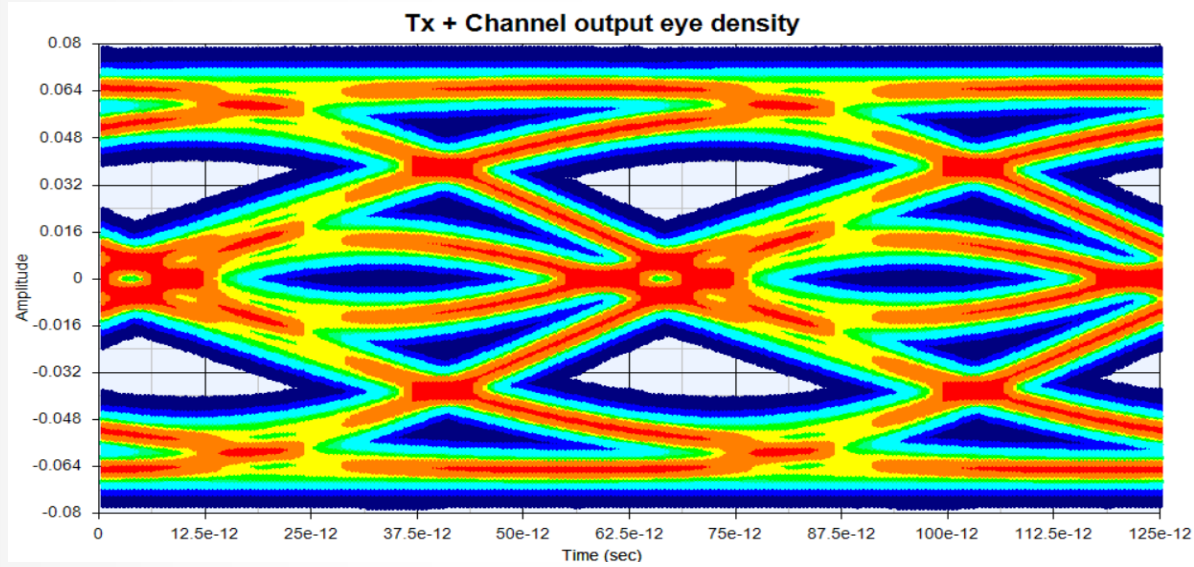
The Tx and Rx models optimize their equalization states

- When the SystemVue simulation is run:
 - The Tx/Rx models de-embed their IBIS impulse response.
 - The Tx/Rx models AdaptForChannel parameters are set to 'Yes' and result in optimized Tx and Rx CTLE settings for the total channel.

```
Generating FIR based on step response data.  
FFE PreCursor = 8, Drive = 4, PostCursor = 9  
FFE gain values: G1 = -0.159968, G2 = -0.005847, G3 = 0.422187  
PreInvert = 0, PostInvert = 0, SwingLevel gain = 0.698  
Adaptation of the channel+equalizer for best eye opening results in closed eye.  
IgnoreSymbolsOut set to 18
```

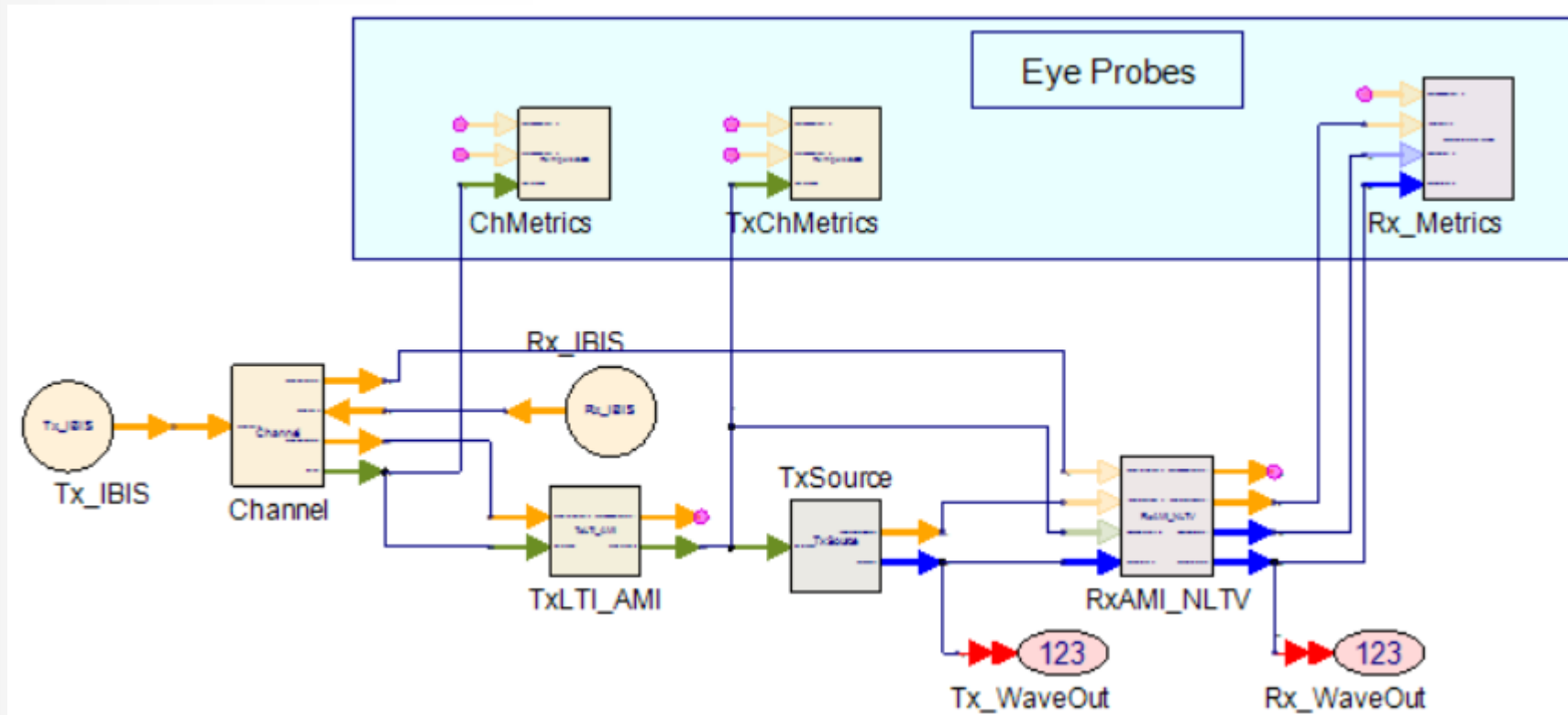
```
CornerCase = Typical (0)  
Front end (FE) model based on step response data.  
Optimization of EQ over count = 1024 states resulting in max open eye ratio = 0.31512 with optimized states:  
Data1Index = 23  
Data2Index = 19  
IgnoreSymbolsOut set to 410
```

Tx and Rx Eye Density Plots






Using the Tx and Rx IBIS-AMI models

- The TxAMI_LTI and RxAMI_NLTV models receive information from their IBIS models that identify the AMI and DLL files referenced in their *.ibs files.






7. The set of models in the SerDesDesign Library

Category: Channels

Name	Description
 Channel_wTxRx_IBIS_Pkg	SerDes channel defined with Tx and Rx IBIS buffers and SnP packages
 ChannelImpulse	SerDes channel defined with impulse
 ChannelSNP	SerDes channel defined with SNP file













Category: Receivers, LTI

Name	Description
 RxAMI_LTI	Receiver AMI LTI model
 RxFE_CTLE	Receiver front end CTLE with 1 to 4 stages
 RxFFE	Receiver FFE

Category: Receivers, NLTV

Name	Description
 RxAGC1	Receiver waveform AGC 1
 RxAMI_NLTV	Receiver AMI NLTV model
 RxCDR	Receiver CDR
 RxCDRDFE	Receiver CDR and DFE
 RxCTLE1	Receiver CTLE 1
 RxDFE	Receiver DFE
 RxNL1	Receiver nonlinearity 1
 RxNL2	Receiver nonlinearity 2

Category: Signal Processing

Name	Description
 ConverterFFE_Data	Converter for FFE data
 ConverterNonlinearityData	Converter for nonlinearity data
 ConverterStepRespData	Converter for step response data
 EyeMetricsNRZ	Eye metrics for NRZ histogram data
 EyeMetricsPAM4	Eye metrics for PAM4 histogram data
 ImpulseMetrics	Metrics for impulse
 ReadFile	Read text data file
 SerDes_FIR	SerDes FIR filter
 SymDetWJitter	SerDes NRZ/PAM4 symbol detector with jitter
 TxSource	SerDes NRZ/PAM4 source
 WaveformMetricsNRZ	Metrics for NRZ waveform
 WaveformMetricsPAM4	Metrics for PAM4 waveform

Category: Transmitters, LTI

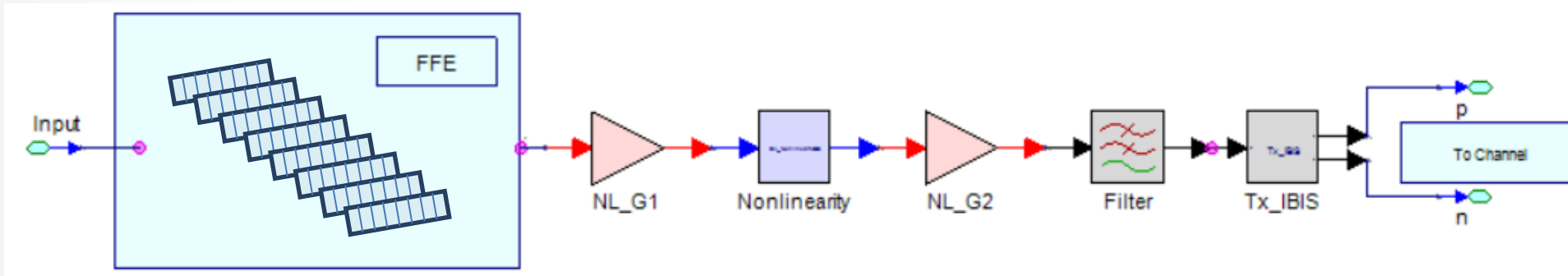
Name	Description
 TxAMI_LTI	Transmitter AMI LTI model
 TxBE_CTLE	Transmitter back end CTLE with 1 to 4 stages
 TxFFE	Transmitter FFE
 TxFFE_BlackBox	Transmitter FFE black box model
 TxFFEwRegisters	Transmitter FFE with registers

Category: Transmitters, NLTV

Name	Description
 TxAMI_NLTV	Transmitter AMI NLTV model
 TxFFEwRegistersNLTV	Transmitter FFE with registers NLTV

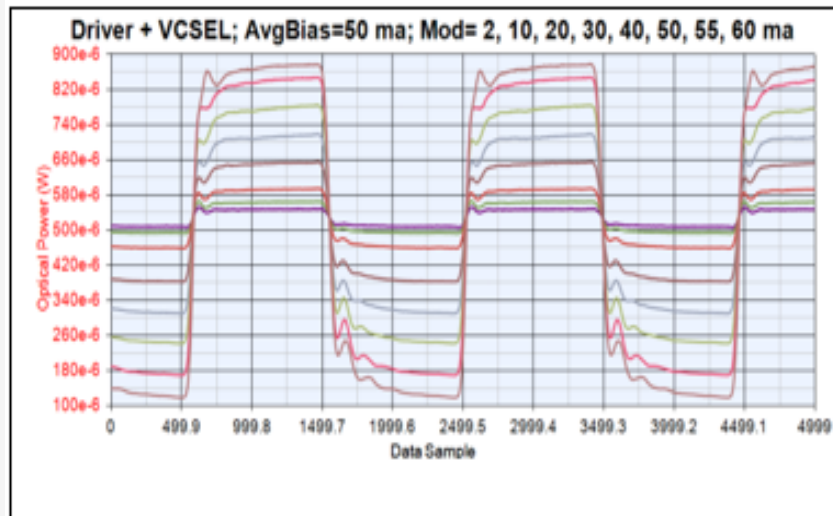
8. Additional SerDes system examples (1/2)

- A Tx using 8 8-bit registers defines an FFE and is modeled using “black box” measurements.
 - 8 eight bit registers are used to define the FFE precursor, main cursor and post cursor.
 - The register bit settings are automatically optimized for the total channel.
 - The IBIS-AMI model was an accurate representation of the Tx circuit.

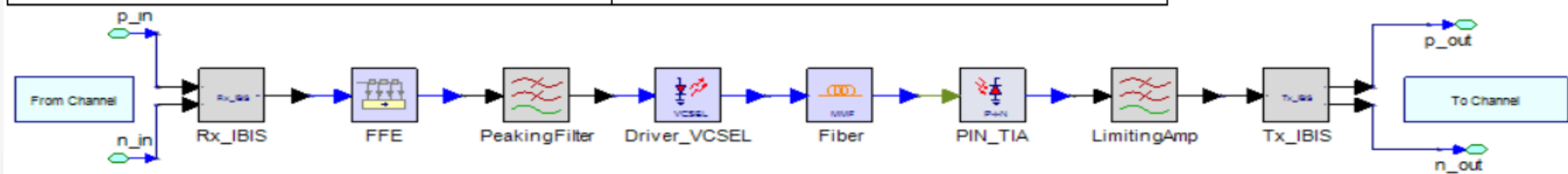


Additional SerDes system examples (2/2)

- A combined Rx/Tx design that defines an optical (E-O-E) repeater.
 - The VCSEL Driver + VCSEL (Driver_VCSEL) is modeled using measurements of states that vary as a function of VCSEL drive level and modulation level. BitRate = 10.3125 Gbps for NRZ data.



Repeating bit pattern with 10 zeros, 10 ones. Data collected with a Keysight Infiniium scope with averaging set to 256. Notice the differences in the rising and falling waveforms as the modulation current is varied from min to max. Data was collected for various settings of the AvgBias.



9. Conclusion

- This discussion was on IBIS-AMI modeling for High-Speed Digital ICs with first-pass integration success.
- Such success is critical for SerDes system design and validation.
- An overview of IBIS-AMI models was presented and how they are used in SerDes system simulations.
- IBIS-AMI model development challenges were discussed.
- An example from NXP was discussed for using SystemVue to model a SerDes channel, model a Tx SerDes IC, model a SerDes Rx IC and model a SerDes system in SystemVue.
- Additional details on the SerDesDesign Library and SerDes system examples were discussed.
- The IBIS-AMI modeling using the SystemVue SerDesDesign Library allows the HSD IC designer to focus on data extraction from their HSD IC for circuit characterization and rely on an automated IBIS-AMI modeling process.

“

The IBIS-AMI models have worked well from first delivery in all EDA tools and for all bit-rate ranges.

They were generated in a timely and efficient manner.

New features were easily added to the models.

Jon Burnett

Engineer / NXP Semiconductor

“

The feature-rich IBIS-AMI models have been critical to our first-pass success.

They easily supported customer integration of our SERDES IP.

Causality corrections fixed issues found in other channel simulations.

Blake Gray Ph.D.

**Director of Engineering /
Silicon Creations LLC**



Appendix

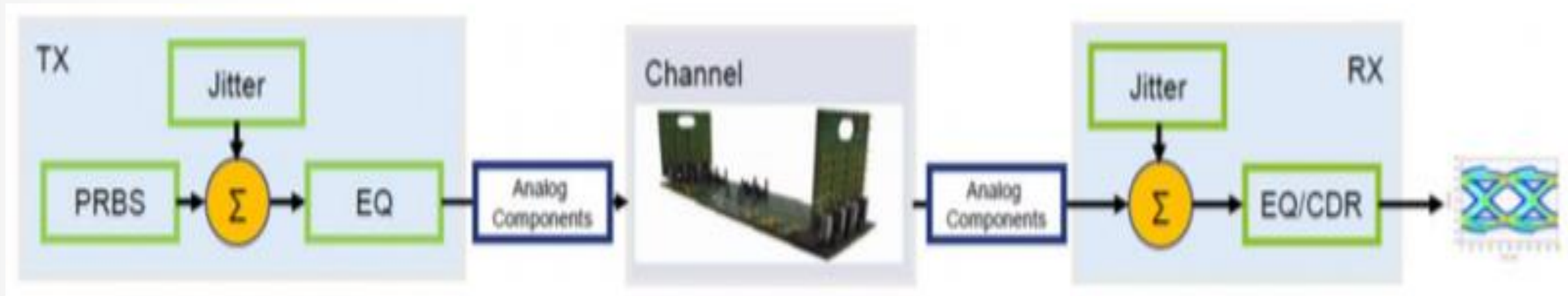
- What are IBIS-AMI models?
- How are IBIS-AMI models used?

1. What are IBIS-AMI models?

- IBIS-AMI models were introduced by the IBIS Open Forum (www.ibis.org) with their IBIS 5.0 specification to address the modeling needs for SerDes systems. Today, the IBIS-AMI specification is at revision 7.0.
- IBISver5.0; IBIS version 5.0 ratified August 2008
- IBISver5.1; IBIS version 5.1 ratified 24 August 2012
- IBISver6.0; IBIS version 6.0 ratified 20 September 2013
- IBISver6.1; IBIS version 6.1 ratified 11 September 2015
- IBISver7.0; IBIS version 7.0 ratified 15 March 2019

IBIS-AMI models are used in modeling SerDes systems

- A SerDes (Serializer-Deserializer) is a pair of Tx/Rx blocks used in high speed communications to compensate for lossy channel links to maintain an open eye at the receiver data slicer.
 - PCI Express, HDMI, USB and other types of links.
- Provides serial data transmission typically over a differential pair
 - Multiple SerDes can be used in parallel for higher throughput.
- At higher data rates (>5 Gbps), equalization was introduced with digital signal processing (DSP).
- Analog and DSP functionality is combined into IBIS-AMI models.



Example SerDes

- A Cadence® 112Gbps Multi-Rate PAM-4 SerDes IP
- 7nm semiconductor process technology
- Evaluated by a Keysight Infiniium DCA-X Oscilloscope.

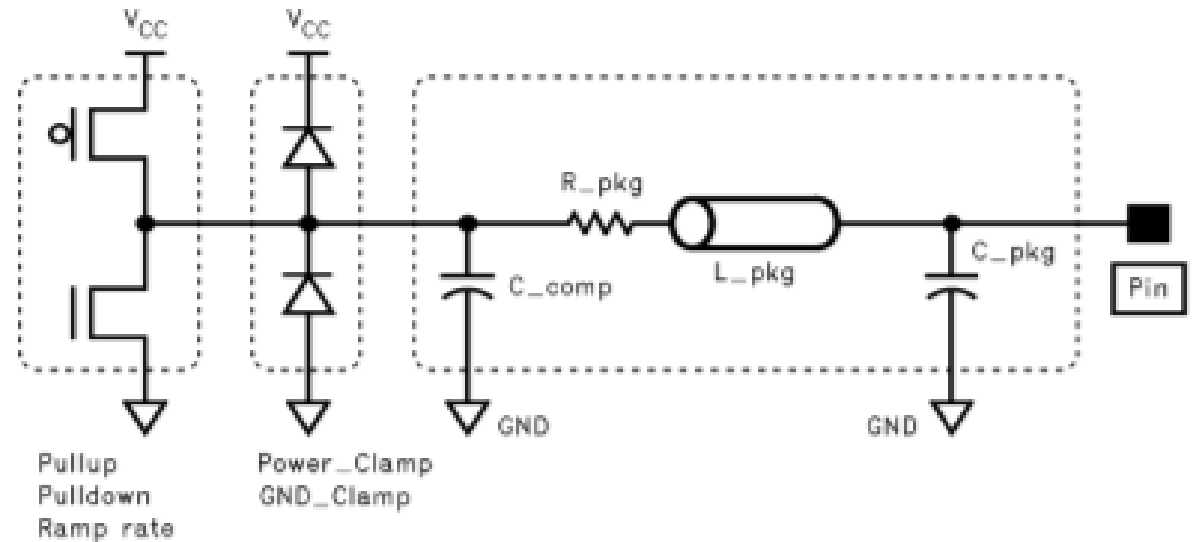


IBIS-AMI is a standard

- IBIS stands for Input/output Buffer Information Specification.
- IBIS-AMI models provide a standardized model interface used by the SerDes chip designers and SerDes system designers.
- The IBIS portion are behavioral models of analog buffers.
- Analog buffers define the SerDes chip impedance interfacing the transmission media.
- Interoperable: different vendor models work together
- Portable: one model runs in multiple simulators
- Flexible: support statistical and time-domain simulation
- High Performance: simulate a million bits per CPU minute
- Accurate: high correlation to simulations / measurement
- Secure: represent IP behavior without exposing internal details

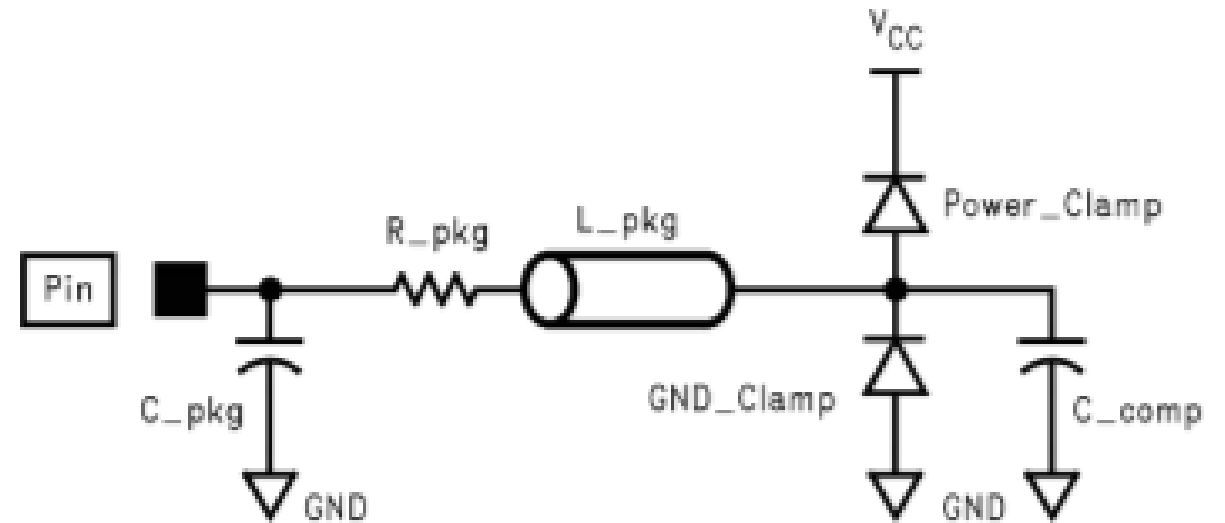
Tx IBIS models

- A Tx IBIS analog output buffer circuit has this simplified representation per pin for a differential output.
- For IBIS-AMI applications only a subset of the IBIS buffer models defined in the IBIS standard are allowed.
- Per the 7.0 standard, can be defined with an S4P file.



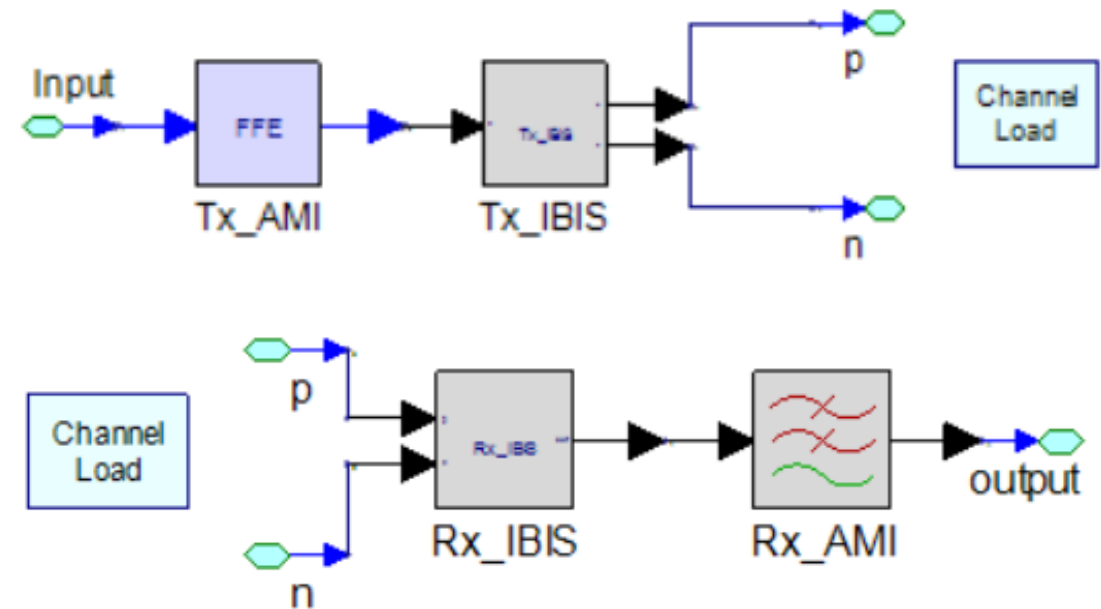
Rx IBIS models

- An Rx IBIS analog input buffer circuit has this simplified representation per pin for a differential input.
- For IBIS-AMI applications only a subset of the IBIS buffer models defined in the IBIS standard are allowed.
- Per the 7.0 standard, can be defined with an S4P file.



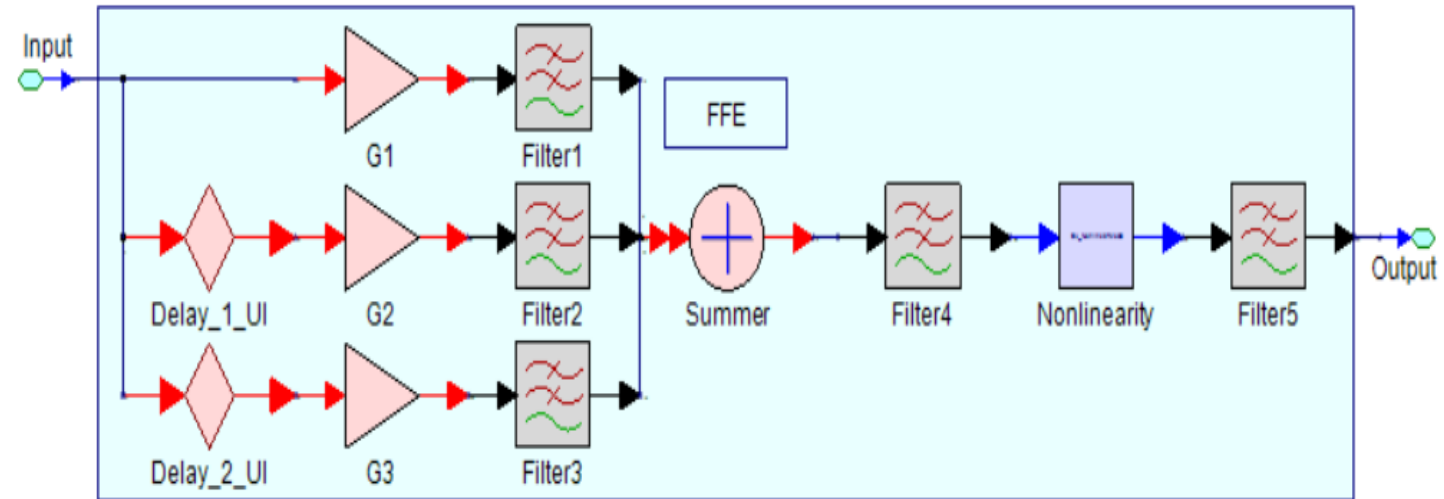
AMI models

- AMI stands for Algorithmic Modeling Interface
- Designed to handle modeling of the signal processing (analog and/or digital) of the SerDes chip.
- The AMI portion can be defined either as linear and time invariant (LTI) or nonlinear and/or time variant (NLTV).



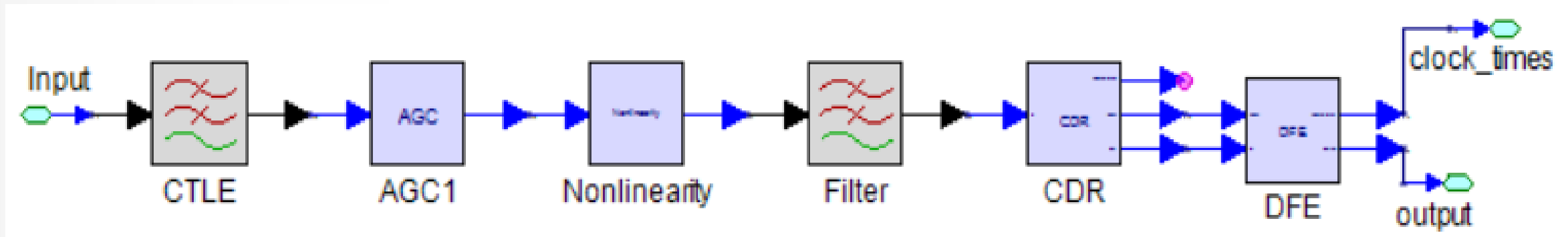
Tx AMI models

- The Tx AMI portion often is a Feed Forward Equalizer (FFE).
- The signal flow graph shown here for a 3 tap FFE.



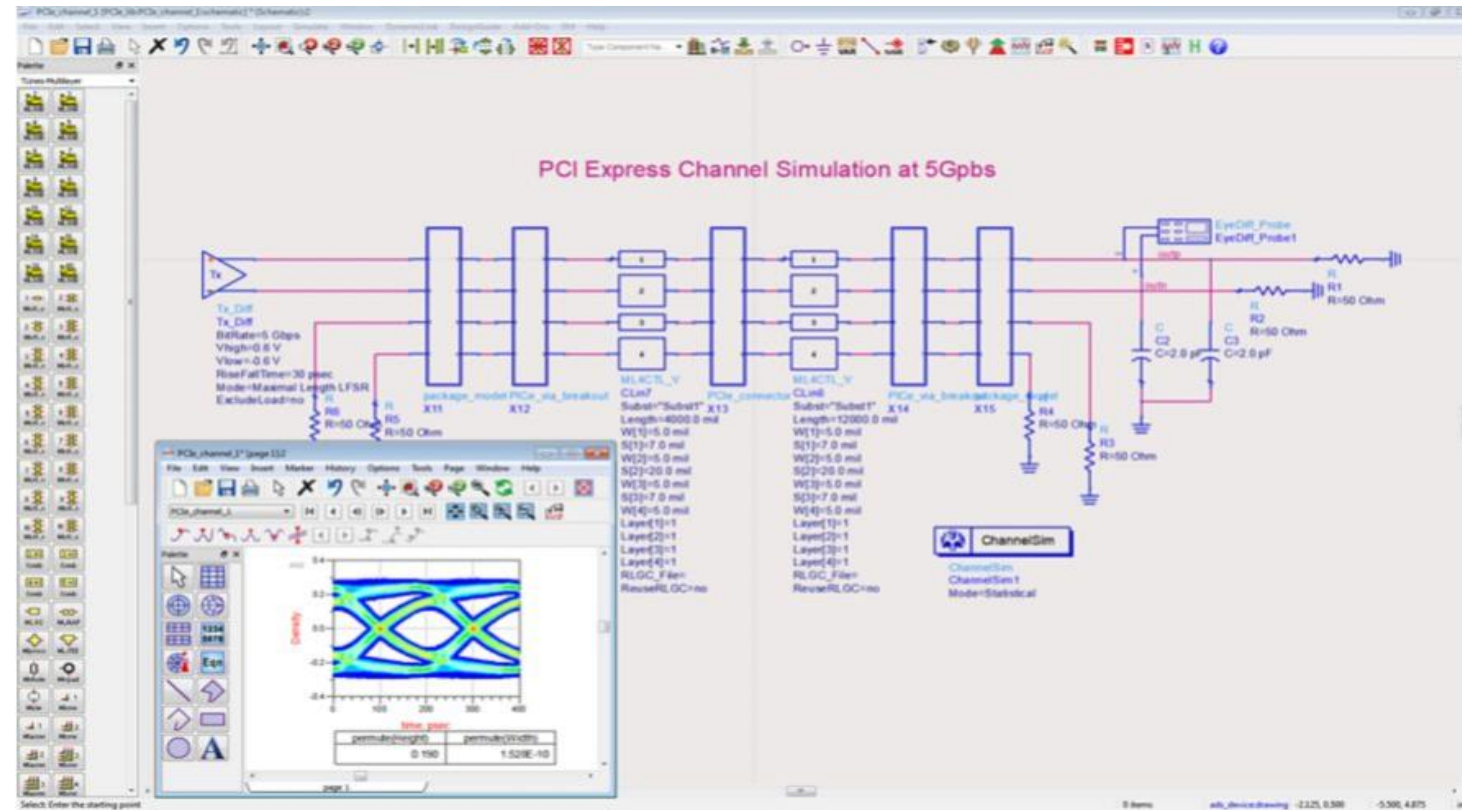
Rx AMI models

- The Rx AMI portion often contains
 - Continuous Time Linear Equalizer (CTLE),
 - Clock and Data Recovery (CDR) circuit and a
 - Decision Feedback Equalizer (DFE)
- The signal flow as shown here is typical.



2. How are IBIS-AMI models used in SerDes systems?

- Along with the introduction of IBIS-AMI models, a new class of simulator was introduced called a SerDes Channel Simulator.
- Here is a screen capture from the Keysight ADS Channel Simulator.

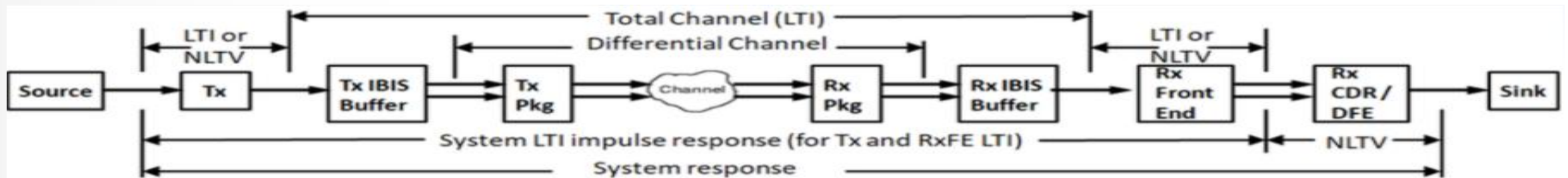


What is a SerDes Channel Simulator?

- A SerDes Channel Simulator combines frequency domain simulation, time domain simulation, DSP simulation and statistical simulation into one tool.
- Can quickly and accurately evaluate and optimize the SerDes system for performance.
- Enables evaluation of margin analysis, robustness of the design, verification of design implementation, and design tradeoffs.
- Before IBIS-AMI, SerDes system designers used traditional SPICE-based analysis.
 - Slow and could not simulate the millions of bits.
- With IBIS-AMI models and Channel Simulators, millions of bits can be simulated.
 - Effects of inter-symbol interference (ISI),
 - jitter (deterministic, D_j ; Gaussian, R_j), and
 - cross-channel interference, and more.

Typical SerDes system line up in a Channel Simulator

- Source: NRZ or PAM4 data source
- Tx: Transmitter equalization; typically with an FFE.
- Tx/Rx IBIS Buffers: IBIS buffer to the differential channel; defines the on-die impedance.
- Tx/Rx Pkg: Package characteristic; typically defined with an SnP file.
- Channel: SerDes channel; typically defined with an SnP file.
- Rx Front End: Receiver equalization; typically with a CTLE.
- Rx CDR/DFE: Receiver timing/equalization; typically with a CDR and DFE.



Channel Simulator Statistical and Bit-by-bit Modes

- Statistical mode: Used when Tx/Rx AMI models are LTI.
 - The entire system is represented by its impulse response.
 - All system metrics (eye density, BER, etc) obtained by statistical post processing of the system impulse.
 - Provides a convenient for a quick analysis, but most practical systems have an Rx model that is NLTV.
- Bit-by-bit mode: Time domain/DSP simulation; post processing is still statistical.
 - BER extrapolation typically used below 10^{-6} BER.