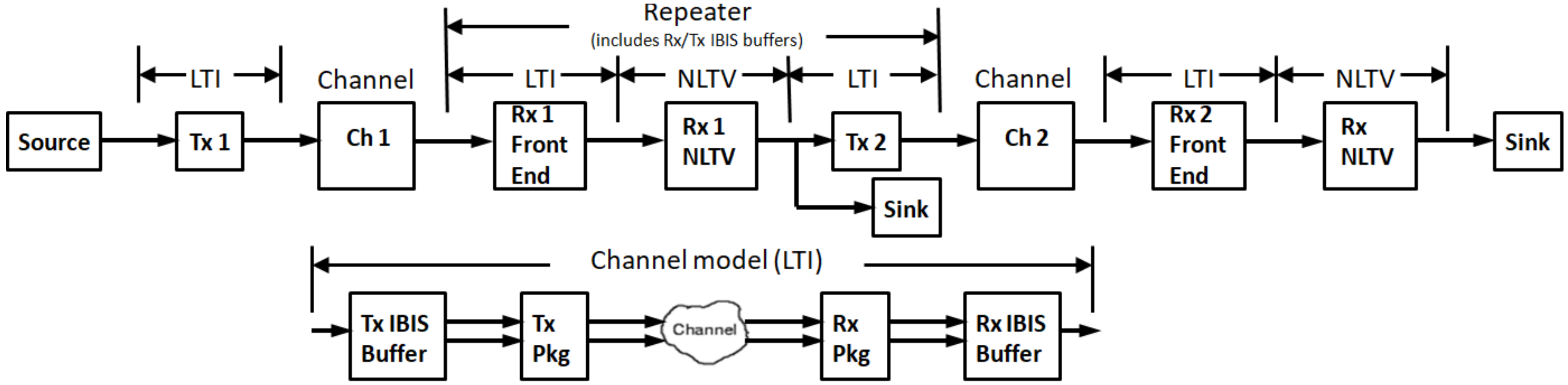
Subject: About the SerDes System Single Repeater Tool

Author: John Baprawski; John Baprawski Inc. (JB)

Date: Jan 3, 2019

This paper discusses features on the web site: **https://www.serdesdesign.com**

The analyzes a SerDes system that has the typical structure shown in this figure.



The SerDes system single repeater system has two differential channels with the repeater (Rx1+Tx2) between the two channels.

In addition to the two differential channels, the repeater system includes:

- SOURCE: a signal source that supports either an NRZ or PAM4 signal.
- TX1: a transmit equalizer (Tx) that is linear and time invariant (LTI) as used in this Channel Simulator.
- Ch1: the repeater input (upstream) channel
- Rx1 Front End: a receive front end equalizer that is LTI as used in this Channel Simulator.
- Rx1 CDR/DFE: receiver clock and data recovery (CDR) unit and decision feedback equalizer (DFE) and is a nonlinear and/or time variant (NLTV) model.

- TX2: a transmit equalizer (Tx) that is LTI as used in this Channel Simulator.
- Ch2: the repeater output (downstream) channel
- Rx2 Front End: a receive front end equalizer that is LTI as used in this Channel Simulator.
- Rx2 CDR/DFE: receiver clock and data recovery (CDR) unit and decision feedback equalizer (DFE) and is a nonlinear and/or time variant (NLTV) model.

Per the IBIS-AMI standard, a repeater is a "redriver" when the Rx1 model has no clock_times output and the input to the Tx2 model is not a regenerated bit/symbol stream.

Per the IBIS-AMI standard, a repeater is a "retimer" when the Rx1 model does have a clock_times output and the input to the Tx2 model is a regenerated bit/symbol stream.

See below for **IBIS-AMI Notes on the Repeater**.

Each differential channel often includes a Tx buffer/package and a Rx package/buffer and is LTI.

- The differential channel represents a hardware SerDes channel and is typically characterized by measuring its N-port S-parameters and is typically a 4-port. The 4-port differential input ports are typically port 1 (+) and port 3 (-). The associated differential output ports are typically port 2 (+) and port 4 (-). The differential characteristic (Port 1 – Port 3 vs. Port 2 – Port 4) is the channel transmission characteristic and is observed versus frequency.
- See S-parameter detail in References > **S-Parameter Channel Examples**
- The S-parameters may also be obtained from various simulators. A high speed digital SerDes channel typically has substantial high frequency attenuation at and beyond the bit/symbol rate Nyquist frequency and requires compensation using equalizers at the transmit and/or receive side of the channel.

The total channel is inclusive of the Tx IBIS Buffer and Rx IBIS Buffer. Per the IBIS-AMI standard, the IBIS Buffers are to be considered LTI for use in a Channel Simulator. Thus, the total channel is LTI.

- See IBIS Buffer detail in: **IBIS Buffers used in SerDes Simulations**
- The total channel, as used in a Channel Simulator and inclusive of the S-parameters, is converted to an equivalent single ended impulse response.
- See channel impulse response detail in References > **Channel Time-Domain Response**

- The typical approach involves zero-padding the S-parameters for the time domain SampleRate (SampleRate = Bit/Symbol Rate * SamplesPer Bit/Symbol) for a maximum frequency of SampleRate/2.0 and applying the constraints for physical realizability which include meeting the mathematical aspects of the Kramers-Kronig relations applied to linear time invariant (LTI) systems. This zero-padding approach often results in high frequency aliasing.
- SerDesDesign.com uses a proprietary algorithm to obtain the causal channel impulse response which inherently does not result in any high frequency aliasing.
- See Causal S-Parameters detail in: **About the Generate Causal S-Parameters Tool**

To use the SerDes System Single Repeater Tool, follow the steps on the web page.

1. Define the analysis name

   - An alpha-numeric character string; including underbar - case sensitive - start with alpha character.
   - The full analysis name = Serdes_<your entered name>
   - This full name is used in the Eye Analysis Tool for further processing of the SerDes system analysis output data.

2 and 6. Define the Transmitters Tx1 and Tx2.

- The transmitters (Tx1 and Tx2) are considered to be LTI and can be defined in a number of different ways.
    - As a feed forward equalizer (FFE) with optional filtering.
    - As an FFE with quantized values defining the pre and post cursors with optional filtering.
    - As an FFE with integer codes defining the pre and post cursors with optional filtering.
    - As a family of step responses
    - As the AMI model associated with an LTI Tx IBIS-AMI model.
- See detail: **Define Transmitter**

3 and 7. Define the Channels (Ch1 and Ch2).

- The channela are considered to be LTI and can be defined in a number of different ways.
    - ChannelType = 0; no channel is included
    - ChannelType = 1; channel is defined with impulse response data file with *.csv (comma separated variables) format with two columns. First column is for time (with constant time step) and second column is for impulse value. The time

    data must start with zero and have constant time step equal to the sampling interval (= 1/SampleRate = 1/BitRate/SamplesPerBit).
    - ChannelType = 2; channel is defined with an S-parameter file with at least 4 ports that represent a differential channel.
    - ChannelType = 3; channel is defined with an S-parameter file with at least 4 ports that represent a differential channel pulse options for:
        - Transmitter IBIS output buffer. See detail: **IBIS Buffers used in SerDes Simulations**
        - Transmitter differential channel packaging S-parameter file with at least 4 ports.
        - Receiver differential channel packaging S-parameter file with at least 4 ports.
        - Receiver IBIS input buffer. See detail: **IBIS Buffers used in SerDes Simulations**
- Any S-parameter file used is automatically adjusted as needed to conform to the physical realizability constraints of passivity, reciprocity, and causality, as well as reduction of noise in the S-parameters.
- See detail: **Define Channel**

4 and 8. Define the Receiver Front Ends (RxFE1, RxFE2).

- The receiver front ends (RxFE1 and RxFE2) are considered to be LTI and can be defined in a number of different ways.
    - As a 1 section continuous time linear equalizer (CTLE) defined with a family of step responses and with optional automatic gain control (AGC).
    - As a 2 section CTLE each defined with a family of step responses and with optional AGC.
    - As a 3 section CTLE each defined with a family of step responses and with optional AGC.
    - As a 4 section CTLE each defined with a family of step responses and with optional AGC.
    - As the AMI model associated with an LTI Rx IBIS-AMI model.
- See detail: **Define ReceiverFrontEnd**

5 and 9. Define the Receiver CDR/DFE models (RxCDRDFE1, RxCDFDFE2).

- The receiver CDR/DFE models (RxCDRDFE1 and RxCDRDFE2) are considered to be nonlinear and/or time variant (NLTV) and can be defined in a number of different ways.
    - As a pass through model for use with Bit-by-bit analysis.
    - As a model with only a clock and data recovery (CDR) model.

  - As a model with a CDR and decision feedback equalizer (DFE).
  - As a model with a CDR and DFE with DFE taps that are quantized.
  - As a model with a CDR and DFE with DFE taps that are integer codes.
  - As the AMI model associated with an NLTV Rx IBIS-AMI model.
- See detail: **Define ReceiverCDRDFE**

10. Setup the analysis.

- Define the bit (symbol) rate in bits (symbols) per second.
- Define the number of samples per bit (symbol).
- Set up analysis options: **Setup Options**
- Set up bit-by-bit mode options: **Setup Bit-by-Bit Mode**
- Specify whether the Repeater is of type "Redriver" or "Retimer".
- Specify whether to generate IBIS-AMI model for the Repeater: **Portable IBIS-AMI Models**

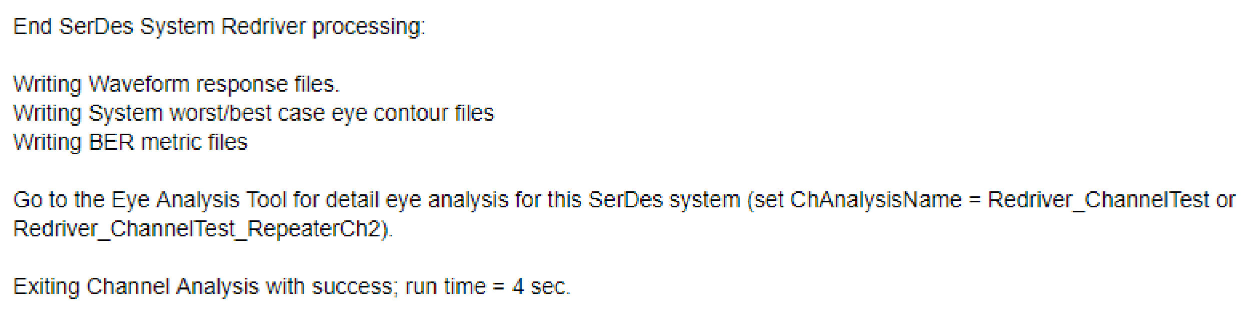
11. Run the analysis.

12 and 13. Display results for each channel and each Rx output.

- Observe the channel frequency domain characteristic, its equivalent impulse response, its eye diagram, and its BER bathtub curve.
- See detail: **Typical SerDes System Characteristics and Displays**

After the Analysis is Run, the Analysis Log file is displayed.

Look at the bottom of the file to see that the analysis was successful:



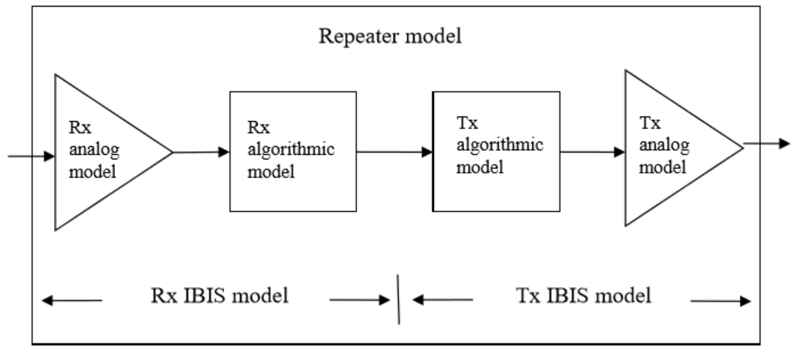
```
End SerDes System Redriver processing:

Writing Waveform response files.
Writing System worst/best case eye contour files
Writing BER metric files

Go to the Eye Analysis Tool for detail eye analysis for this SerDes system (set ChAnalysisName = Redriver_ChannelTest or
Redriver_ChannelTest_RepeaterCh2).

Exiting Channel Analysis with success; run time = 4 sec.
```

Open the **Eye Analysis Tool** for detail eye and BER analysis of the SerDes system results by entering as the ChAnalysisName for the Ch1 results or the Ch2 results as noted in the log file message.

IBIS-AMI Notes on the Repeater

A Repeater (per the IBIS-AMI 6.0 standard) is a type of device that is placed in the middle of the channel to compensate channel loss. Repeaters consist of two categories, Redrivers and Retimers. A Redriver equalizes the upstream channel signal and retransmits it to the downstream channel. The output signal is continuously driven by the input signal. **A Redriver has no retiming performed when the Redriver retransmits the signal**. **A Retimer equalizes the upstream channel signal, recovers the clock using a CDR and generates a digital stimulus that is transmitted to the downstream channel**.

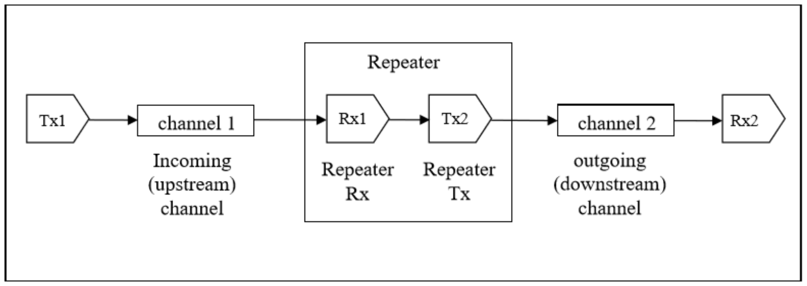
A Repeater is modeled by two back-to-back input-output IBIS-AMI models as shown here.



The analog part of the Rx model represents the input termination at the device input. The analog part of the Tx model represents the output impedance at the device output. The two algorithmic models represent equalizers, clock data recovery or CDR circuits (if they exist) and/or preemphasis inside the devices. In a Redriver, both algorithmic models can optionally implement the AMI_GetWave function. In a Retimer, the Rx algorithmic model must implement AMI_GetWave and the function must return clock times. The Retimer Tx algorithmic model can optionally implement AMI_GetWave. The order of signal flow in a Repeater model is from Rx analog to Rx algorithmic to Tx algorithmic to Tx analog. Looking from the Rx analog portion, the Rx algorithmic block is assumed to have infinite input impedance. Looking from the Tx analog portion, the Tx algorithmic block is assumed to have an output of an ideal voltage source. A Repeater model is specified in a single .ibs file that includes both input and output models.

In Repeater AMI simulations, both Repeater analog models are treated as if they are linear and time-invariant. The incoming (upstream) analog channel of the Redriver, including the upstream Tx analog model, the physical channel and the Repeater Rx analog model, is represented by an impulse response. The outgoing (downstream) analog channel of the Repeater, including the Repeater Tx analog model, the physical channel and the downstream Rx analog model, is represented by another impulse response.

The time domain simulation flow for a Repeater link is shown here.



Here Tx1 denotes the Repeater upstream channel (channel 1) Tx AMI model (including analog and algorithmic models), Rx1 the Repeater Rx AMI model (including analog and algorithmic models), Tx2 the Repeater Tx AMI model (including analog and algorithmic models) and Rx2 the Repeater downstream channel (channel 2) Rx AMI model (including analog and algorithmic models).

The **bit-by-bit time domain simulation flow** for the Repeater link shown in the above figure is defined here.

Step 1. The EDA tool obtains the impulse response of the upstream analog channel, which represents the combined impulse response of Tx1's analog model, physical channel 1, and Rx1's analog model.

Step 2. The output of step 1 is presented to Tx1's AMI_Init function and Tx1's AMI_Init function is executed.

Step 3. The output of step 2 is presented to Rx1's AMI_Init function and Rx1's AMI_Init function is executed.

Step 4. The EDA tool obtains the impulse response of the downstream analog channel, which represents the combined impulse response of Tx2's analog model, physical channel 2, and Rx2's analog model.

Step 5. The output of step 4 is presented to Tx2's AMI_Init function and Tx2's AMI_Init function is executed.

Step 6. The output of step 5 is presented to Rx2's AMI_Init function and Rx2's AMI_Init function is executed.

Step 7. The EDA tool performs simulation on the upstream channel, which consists of Tx1, physical channel 1, and Rx1, according to the AMI flow defined in the specification for channels without Repeaters.

Step 8a. **Redriver**: The EDA tool uses the signal waveform at the output end of Rx1's algorithmic model in step 7, regardless whether Rx1's AMI_GetWave exists or not, as the stimulus of Tx2's algorithmic model, regardless whether Tx2's AMI_GetWave exists or not, and performs simulation on the downstream channel, which consists of Tx2, physical channel 2 and Rx2, according to the AMI flow defined in the spec for channels without Redrivers.

Step 8b. **Retimer**: The EDA tool samples the output waveform of Retimer Rx AMI_GetWave at ½ UI after each clock tick returned by the function, generates a digital stimulus as the input to Tx2's algorithmic model, regardless whether Tx2's AMI_GetWave exists or not, and performs simulation on the downstream channel, which consists of Tx2, physical channel 2 and Rx2, according to the AMI flow defined in the spec for channels without Redriver. The logic level of the digital stimulus is 1 if sampled value >= Rx1's Rx_Receiver_Sensitivity and 0 if sampled value <= −Rx1's Rx_Receiver_Sensitivity. If –Rx1's Rx_Receiver_Sensitivity < sampled value < Rx1's Rx_Reciver_Sensitivity, the logic level is unchanged from the previous bit. The digital stimulus have values of -½ volt for logic 0 and +½ volt for logic 1.

Step 9. The EDA tool calls the AMI_Close function of each algorithmic model in Tx1, Rx1, Tx2 and Rx2.

Since the Redriver output signal is driven continuously by the input analog signal and does not have a sampling latch, clock times, if returned by a Redriver model, jitter parameters and the Rx_Noise parameter specified in Redriver .ami files are ignored by the EDA tool. Since the Retimer output signal is driven by a digital stimulus as described above in step 8b, jitter and noise parameters specified in Retimer .ami files are applied according to the specification for channels without Repeaters.

The **statistical simulation** flow for the Repeater link shown in the above figure is defined here.

Step 1. The EDA tool obtains the impulse response of the upstream analog channel, which represents the combined impulse response of Tx1's analog model, physical channel 1, and Rx1's analog model.

Step 2. The output of step 1 is presented to the Tx1's AMI_Init function and Tx1's AMI_Init function is executed.

Step 3. The output of step 2 is presented to the Rx1's AMI_Init function and the Rx1's AMI_Init function is executed.

Step 4. The EDA tool obtains the impulse response of the downstream analog channel, which represents the combined impulse response of Tx2's analog model, physical channel 2, and Rx2's analog model.

Step 5. The output of step 4 is presented to Tx2's AMI_Init function and Tx2's AMI_Init function is executed.

Step 6. The output of step 5 is presented to Rx2's AMI_Init function and Rx2's AMI_Init function is executed.

Step 7a. **Redriver**: The EDA tool convolves impulse responses returned by Rx1's AMI_Init in step 3 and by Rx2's AMI_Init in step 6 to obtained the full channel impulse response and uses it to perform statistical simulation.

Step 7b. **Retimer**: The EDA tool uses the impulse responses returned by Rx1's AMI_Init in step 3 to perform a statistical simulation of channel 1. The EDA tool uses the impulse responses returned by Rx2's AMI_Init in step 6 to perform a statistical simulation of channel 2.