

BUFFER ISSUE RESOLUTION DOCUMENT (BIRD)

BIRD ID#: TBD
ISSUE TITLE: *Resolving problems with Redriver Init Flow*
REQUESTER: *Walter Katz, Signal Integrity Software, Inc.*
Darshan Shah, F5Networks, Inc.

DATE SUBMITTED: TBD October 28, 2014

ANALYSIS PATH/DATA THAT LED TO SPECIFICATION:

As currently written, the reference flow for a Redriver makes the incorrect assumption that the downstream Rx equalization is determined from the downstream Tx and the downstream channel. In order for the downstream Rx to properly determine its equalization, the impulse response input to the downstream Rx must also include the impulse response output of the upstream Rx. The current flow is written because it assumes that the downstream Tx equalization is determined by the downstream channel.

The proposed revision corrects the Redriver statistical simulation flow.

A channel can have one or more Retimer and/or Redriver models between the Source Tx and the Terminal Rx. This BIRD corrects the flows in sections of the channel that contain only Redrivers between either the Source Tx or a Retimer Tx and a the Primary Rx or a Retimer Rx. For the purpose of this BIRD the Source Tx or Retimer Tx will be called the Upstream Tx, and the Primary Rx or Retimer Rx will be called the Downstream Rx.

In addition, it is assumed that the Upstream Tx is not Adaptive and the Downstream Rx is Adaptive. Adaptive means that the equalization performed by the model is a function of its input.

The Upstream Tx and the Redriver Tx commonly have `Init_Returns_Impulse = True`, and have either `GetWave_Exists True` or `False`. Cases where the Upstream Tx or Redriver Tx(s) have `Init_Returns_Impulse = False` is not considered in this BIRD. Cases where the Upstream Tx or the Redriver Tx(s) have `GetWave_Exists False` is not specifically addressed in this BIRD, other than the fact that if a Tx has `GetWave_Exists False` (with the assumption that the Tx is not adaptive) the EDA tool can do the function of the Tx `GetWave` by inputting a unit impulse response to the Tx `AMI_Init` function, the output impulse response will then contain the Tx equalization and the EDA tool can use this impulse response to emulate the missing Tx `GetWave` functionality.

The situation for the Rx models is more complex. The Downstream Rx and Redriver Rx(s) can have `Rx Init_Returns_Impulse True` and `GetWave_Exits True` (Dual Model), `Init_Returns_Impulse True` and `GetWave_Exits False` (Init Only Model), or

Init_Returns_Impulse False GetWave_Exits True (GetWave Only Model). This BIRD is limited to correcting the Statistical Flows when all of the Rx models have Init_Returns_Impulse True. It should be noted that Redrivers are often not LTI (e.g. they can amplify and saturate their input). For this case, if the downstream Rx is Init Only and any of the upstream Rx models are Dual Model the Time Domain flow is problematic and is not considered in this BIRD.

Replace the following section on page 224-225:

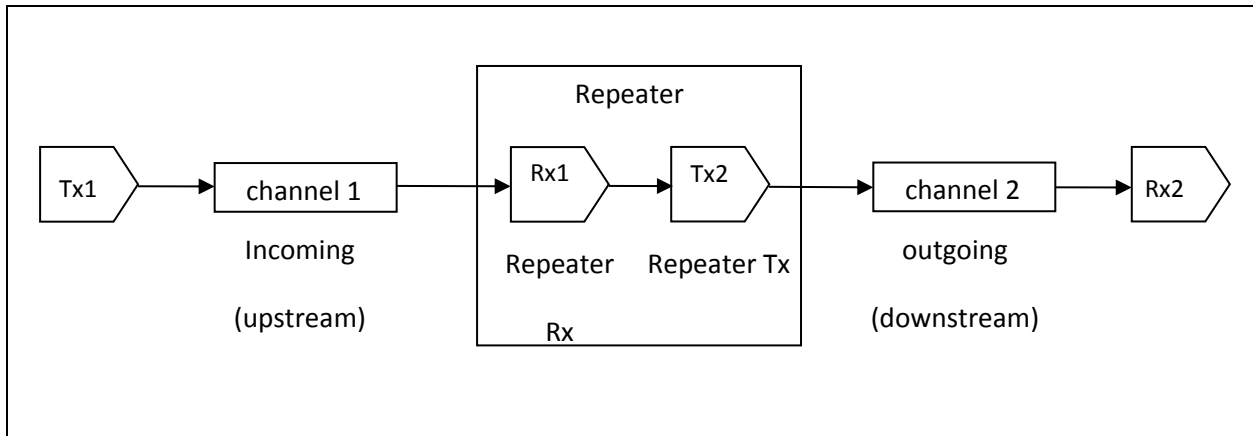


Figure 1 - Repeater link

Editorial Comment: Should be Figure 40

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).

Step 1. The simulation platform 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 simulation platform 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 simulation platform 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 simulation platform 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 simulation platform samples the output waveform of Retimer Rx AMI_GetWave at $\frac{1}{2}$ 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 \geq Rx1's Rx_Receiver_Sensitivity and 0 if sampled value \leq $-$ Rx1's Rx_Receiver_Sensitivity. If $-$ Rx1's Rx_Receiver_Sensitivity $<$ sampled value $<$ Rx1's Rx_Receiver_Sensitivity, the logic level is unchanged from the previous bit. The digital stimulus have values of $-\frac{1}{2}$ volt for logic 0 and $+\frac{1}{2}$ volt for logic 1.

Step 9. The simulation platform 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 simulation platform. 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 a Repeater link shown in Fig. 2 is defined below.

Step 1. The simulation platform 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 simulation platform 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 simulation platform convolves impulse responses returned by Rx1's AMI_Init in step 3 and by Rx2's AMI_Init in step 6 to obtain the full channel impulse response and uses it to perform statistical simulation.

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

With the following

Retimer Flow

Editorial Comment: Change Repeater to Retimer in the following graphic.

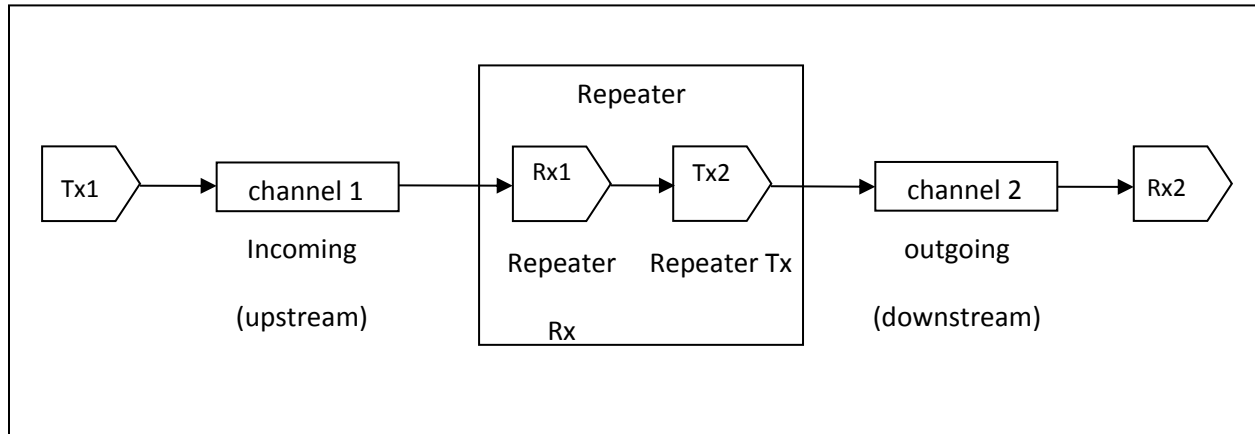


Figure 1 - Repeater link

Editorial Comment: Should be Figure 40

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

Step 1. The simulation platform 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 simulation platform 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 simulation platform 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 8. The simulation platform samples the output waveform of Retimer Rx AMI_GetWave at $\frac{1}{2}$ 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 \geq Rx1's Rx_Receiver_Sensitivity and 0 if sampled value \leq $-$ Rx1's Rx_Receiver_Sensitivity. If $-$ Rx1's Rx_Receiver_Sensitivity < sampled value < Rx1's Rx_Receiver_Sensitivity, the logic level is unchanged from the previous bit. The digital stimulus have values of $-\frac{1}{2}$ volt for logic 0 and $+\frac{1}{2}$ volt for logic 1.

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

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 a **Retimer** link shown in Fig. 2 is defined below.

Step 1. The simulation platform 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 simulation platform 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 simulation platform uses the impulse responses returned by Rx1's AMI_Init in step 3 to perform a statistical simulation of channel 1. The simulation platform uses the impulse responses returned by Rx2's AMI_Init in step 6 to perform a statistical simulation of channel 2.

Redriver Flow

Editorial Comment: Change Repeater to Redriver in the following graphic.

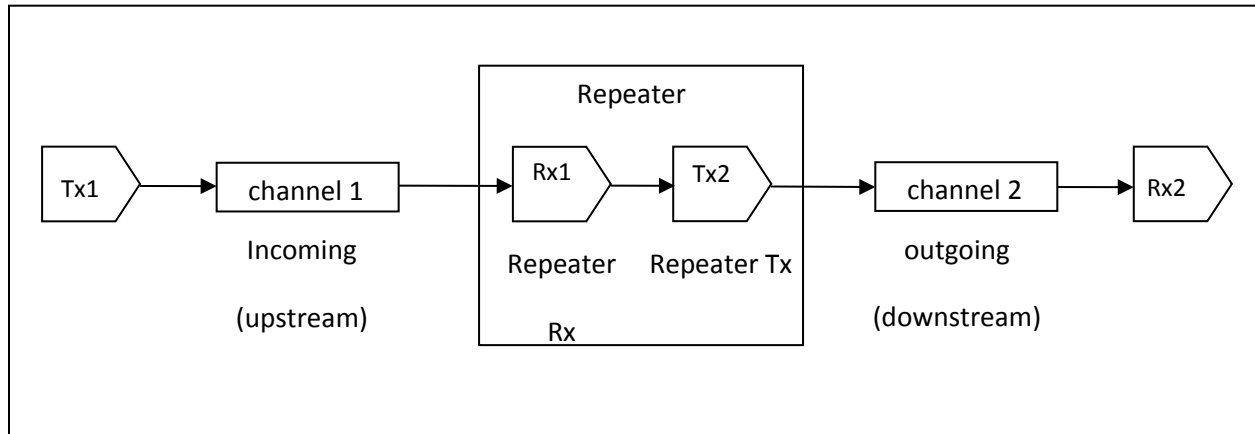


Figure 1 - Repeater link

Editorial Comment: Should be Figure 40a

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

The time domain simulation flow for a Redriver link shown in Fig. 40a is defined below.

Step 1. The simulation platform 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. A Unit Impulse Response is presented to the Tx2's AMI_Init function and Tx2's AMI_Init function is executed. A Unit Impulse Responses' first point is 1/sample_interval, all other values are 0.

Step 5. The simulation platform 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 6. The outputs of Step 3, Step 4 and Step 5 are convolved together.

Step 7. The output of step 6 is presented to Rx2's AMI_Init function and Rx2's AMI_Init function is executed.

Step 8a. If the Rx2 has an AMI_GetWave function: The simulation platform 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 Redrivers. The simulation platform 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. If the Tx2 algorithmic model has an AMI_GetWave the signal waveform at the output end of Rx1's algorithmic is input to the Tx2 AMI_GetWave function and the output of the Tx2 AMI_GetWave function is convolved with the output of Step 5. If the Tx2 algorithmic model does not have an AMI_GetWave the signal waveform at the output end of Rx1's algorithmic model is convolved with the output of Step 4 and Step 5.

Step 8b. If the Rx2 does not have an AMI_GetWave function: The simulation platform performs simulation by convolving the stimulus input to Tx1 with the output of Step 7.

Step 9. The simulation platform 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 simulation platform.

The statistical simulation flow for a Redriver link shown in Fig. 40a is defined below. This flow is only valid if all of the Tx and Rx models have `Init_Returns_Impulse` True.

Step 1. The simulation platform 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. A Unit Impulse Response is presented to the Tx2's `AMI_Init` function and Tx2's `AMI_Init` function is executed. A Unit Impulse Responses' first point is $1/\text{sample_interval}$, all other values are 0.

Step 5. The simulation platform 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 6. The outputs of Step 3, Step 4 and Step 5 are convolved together.

Step 7. The output of step 6 is presented to Rx2's `AMI_Init` function and Rx2's `AMI_Init` function is executed.

Step 8. The output of Step 7 is used to perform statistical simulation.