**BUFFER ISSUE RESOLUTION DOCUMENT (BIRD)**

**BIRD ID#:** 156.3

**ISSUE TITLE:** *IBIS-AMI Extension for Mid-channel Redrivers and Retimers*

**REQUESTER:** *Mahbubul Bari, Ron Olisar, and Hassan Rafat, Maxim Integrated;*

 *Fangyi Rao and Colin Warwick, Agilent Technologies, Inc.;*

 *Walter Katz and Todd Westerhoff, Signal Integrity Software, Inc.*

**DATE SUBMITTED:** January 11, 2013

**DATE REVISED:** January 12, 2013; May 24, 2013, June 7, 2013

**DATE ACCEPTED BY IBIS OPEN FORUM:** June 7, 2013

**ANALYSIS PATH/DATA THAT LED TO SPECIFICATION:**

A Repeater is a type of device that is placed in the middle of the channel to compensate channel loss. A Repeater model consisted of an Rx and Tx model. The [Repeater Pin] keyword is used to pair the Rx and Tx pins to define a Repeater. A Repeater can either be a Redriver or Retimer.

A Redriver equalizes the upstream channel signal and retransmits it to the downstream channel. The output signal is continuously driven by the input signal and no retiming is performed when the Redriver retransmits the signal. As a Redriver Rx does not have a CDR (clock data recovery) circuit, it passes the equalized output waveform of the Rx half directly to the Tx half of the Repeater. Since the Redriver can be nonlinear and noisy, its presence breaks the linear channel assumption in AMI simulation today. A Retimer contains a CDR, samples the Rx equalized waveform and generates a digital stimulus for the Tx half of the Repeater. The conversion from the Rx output waveform to the Tx digital input stimulus also breaks the linear channel assumption.

The proposed revision allows the accommodation of redrivers.

This BIRD introduces a new IBIS keyword [Repeater Pin] which associates the pins of an Rx model with the pins of a Tx model, thus associating an Rx model with a Tx model to form a complete Repeater. It also introduces a new AMI reserved parameter for Repeater Rx model to specify whether the Repeater is a Redriver or a Retimer.

**ANY OTHER BACKGROUND INFORMATION:**

Approved with changes shown below with highlighting in color, during the IBIS Open Forum teleconference on June 7, 2013.

Step 8b. Retimer: The simulation platform 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.

**STATEMENT OF THE RESOLVED SPECIFICATIONS:**

A Repeater is a type of device that is placed in the middle of the channel to compensate channel loss. Repeater has two categories, Redriver and Retimer. 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 does not have a CDR, and no retiming is 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 in Fig. 1.

Rx

analog

model

Rx

algorithmic

model

Tx

algorithmic

model

Tx

analog

model

Repeater model

Rx IBIS model

Tx IBIS model

Figure 1: Repeater model

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 equalizations, clock data recovery or CDR (if it exists) and pre-emphasis inside the device. 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 ticks. 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. An example redriver.ibs file is:

[IBIS Ver] 5.2

[File Name] Redriver.ibs

[Component] Redriver

…

[Pin] signal\_name model\_name R\_pin L\_pin C\_pin

1p Redriver\_Rx\_1p Redriver\_Rx

1n Redriver\_Rx\_1n Redriver\_Rx

2p Redriver\_Tx\_2p Redriver\_Tx

2n Redriver\_Tx\_2n Redriver\_Tx

[Diff\_Pin] inv\_pin vdiff tdelay\_typ tdelay\_min tdelay\_max

1p 1n NA NA NA NA

2p 2n NA NA NA NA

[Repeater Pin]

1p 2p

[Model] Redriver\_Rx

Model\_type Input

…

[Algorithmic Model]

Executable Windows\_VisualStudio10.0.30319\_32 Redriver\_Rx\_32.dll Redriver\_Rx.ami

Executable Windows\_VisualStudio10.0.30319\_64 Redriver\_Rx\_64.dll Redriver\_Rx.ami

Executable Linux\_gcc4.6.1\_32 Redriver\_Rx\_32.so Redriver\_Rx.ami

Executable Linux\_gcc4.6.1\_64 Redriver\_Rx\_64.so Redriver\_Rx.ami

[End Algorithmic Model]

[Model] Redriver\_Tx

Model\_type Output

…

[Algorithmic Model]

Executable Windows\_VisualStudio10.0.30319\_32 Redriver\_Tx\_32.dll Redriver\_Tx.ami

Executable Windows\_VisualStudio10.0.30319\_64 Redriver\_Tx\_64.dll Redriver\_Tx.ami

Executable Linux\_gcc4.6.1\_32 Redriver\_output\_32.so Redriver\_Tx.ami

Executable Linux\_gcc4.6.1\_64 Redriver\_output\_64.so Redriver\_Tx.ami

[End Algorithmic Model]

[End]

New AMI Reserved Parameters:

The Rx AMI model of a Repeater must have the Reserved Parameter “Repeater\_Type”. This is a String parameter with either the value “Retimer” or “Redriver”.

As mentioned above, a Retimer Rx must has AMI\_GetWave (GetWave\_Exist = True) and the AMI\_GetWave function must return clock\_ticks. The simulation platform shall generate a digital input to the Retimer Tx by sampling the Rx AMI\_GetWave output waveform ½ UI after each clock tick, The digital stimulus shall have values of -½ and +½.

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 shown in Figure 2 is defined below.

Repeater

Rx

Tx1

Rx1

Tx2

Rx2

channel 1

channel 2

Repeater

Repeater Tx

Incoming

(upstream)

channel

outgoing

(downstream)

channel

Figure 2: Repeater link

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 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 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 obtained 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.

A mixture of Redrivers and Retimers can be cascaded in a channel.

## Parameter DEFINITIONs

*Parameter:* **Repeater\_Type**

*Required:* No

*Descriptors*:

Usage: Info

Type: String

Format: Value

Default: None

Description:<String>

*Definition:* This parameter is a reserved parameter of Repeater Rx model and shall have the value “Redriver” or “Retimer”

*Usage Rules:* This parameter is required if the Rx model is part of a Repeater Rx/Tx pair. A Retimer Rx model must has AMI\_GetWave (GetWave\_Exists = True) and the function must return clock\_ticks.

*Examples:*

(Repeater (Usage Info)(Type String)(Value “Redriver”))

Add to Section 3A "KEYWORD HIERARCHY" after

 │ ├── **[Diff Pin]** inv\_pin, vdiff, tdelay\_typ,

 │ │ tdelay\_min, tdelay\_max

the following

 │ ├── **[Repeater Pin]** tx\_non\_inv\_pin

Add section 5 "COMPONENT DESCRIPTION" before

*Keyword:* [Series Pin Mapping]

the following

*Keyword:* [Repeater Pin]

*Required:* No

*Description:* Associates a differential Rx non-inv pin with a Tx non-inv pin to form a Repeater.

*Sub-Params:* tx\_non\_inv\_pin

*Usage Rules:* Enter only Repeater pin pairs. The first column, [Repeater Pin] contains a non-inv pin name of an entry in the [Diff Pin] section that represents an Input or Input\_diff model corresponding to the Rx part of the Repeater model. The second column, tx\_non\_inv\_pin contains a non-inv pin name of an entry in the [Diff Pin] section that represents an Output or Output\_diff model corresponding to the Tx part of the Repeater model.

*Other Notes:* Each line must contain two columns. A pin name may appear in only one [Repeater Pin] record.

The column length limits are:

[Repeater Pin] 5 characters max

tx\_non\_inv\_pin 5 characters max

*Example:*

[Repeater Pin] tx\_non\_inv\_pin

3 11