**BUFFER ISSUE RESOLUTION DOCUMENT (BIRD)**

**BIRD ID#:**

**ISSUE TITLE:** *IBIS-AMI Extension for Mid-channel Redriver*

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

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

**DATE SUBMITTED:**

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

Redriver is one type of device that is placed in the middle of the channel to compensate channel loss. It 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. Since the redriver can be nonlinear and noisy, its presence breaks the linear channel assumption in AMI simulation today. The proposed revision allows the accommodation of redrivers.

No new keywords are added. A redriver is modeled by two back-to-back input-output IBIS-AMI models as shown below.

|---------------------------------------------------------------------|

| Redriver model |

| |

| \* \* \* \* |

| \* \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \* \* |

| \* \* \* \* \* \* \* \* |

| \* input \* \* input \* \* output \* \* output \* |

-->|---\* analog \*-->\* algorithmic \*-->\* algorithmic \*-->\* analog \*---|-->

| \* model \* \* model \* \* model \* \* model \* |

| \* \* \* \* \* \* \* \* |

| \* \* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* \* \* |

| \* \* \* \* |

| |

|<------ input ibis model ------->| |<------ output ibis model ------>|

| |

|---------------------------------------------------------------------|

The analog part of the input model represents the input termination at the device input. The analog part of the output model represents the output impedance at the device output. The two algorithmic models represent equalizations and pre-emphasis inside the device. Both models can optionally implement the AMI\_GetWave function. The order of signal flow in a redriver model is from input analog to input algorithmic to output algorithmic to output analog. It should be pointed out that the output analog model is driven continuously by the analog waveform returned by the output algorithmic model instead of digital trigger events as in a regular IBIS output model. Therefore, the output analog model is expected to describe an analog circuit as oppose to the conventional digital-to-analog converter. Looking from the input analog portion, the input algorithmic block is assumed to have infinite input impedance. Looking from the output analog portion, the output algorithmic block is assumed to have an output of an ideal voltage source. A redriver model is specified in a single .ibs file that includes both input and output models. An example of 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\_input\_1p redriver\_input

1n redriver\_input\_1n redriver\_input

2p redriver\_output\_2p redriver\_output

2n redriver\_output\_2n redriver\_output

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

1p 1n NA NA NA NA

2p 2n NA NA NA NA

[Model] redriver\_input

Model\_type Input

…

[Algorithmic Model]

Executable Windows\_VisualStudio10.0.30319\_32 redriver\_input\_32.dll redriver\_input.ami

Executable Windows\_VisualStudio10.0.30319\_64 redriver\_input\_64.dll redriver\_input.ami

Executable Linux\_gcc4.6.1\_32 redriver\_input\_32.so redriver\_input.ami

Executable Linux\_gcc4.6.1\_64 redriver\_input\_64.so redriver\_input.ami

[End Algorithmic Model]

[Model] redriver\_output

Model\_type Output

…

[Algorithmic Model]

Executable Windows\_VisualStudio10.0.30319\_32 redriver\_output\_32.dll redriver\_output.ami

Executable Windows\_VisualStudio10.0.30319\_64 redriver\_output\_64.dll redriver\_output.ami

Executable Linux\_gcc4.6.1\_32 redriver\_output\_32.so redriver\_output.ami

Executable Linux\_gcc4.6.1\_64 redriver\_output\_64.so redriver\_output.ami

[End Algorithmic Model]

[End]

In AMI simulations both redriver analog models are treated as if they are linear and time-invariant. The upstream channel, including the redriver input analog model, is represented by an impulse response, hAC1. It is the input impulse to the redriver input algorithmic model’s AMI\_Init function. The downstream channel, including the redriver output analog model, is represented by another impulse response, hAC2. It is the input impulse to the redriver output algorithmic model’s AMI\_Init function. During time-domain simulations, the output signal from the upstream SerDes Tx algorithmic model is convolved with hAC1. The resulting waveform is the input waveform to the redriver input algorithmic model, whose output is the input waveform of the redriver output algorithmic model. Its output signal is convolved with hAC2 and the resulting waveform is the input waveform to the downstream SerDes Rx algorithmic model. Redrivers can be cascaded in a channel.

Since the redriver output signal is driven continuously by the input signal, clock times are ignored if they are returned by a redriver model. For the same reason, jitter parameters specified in redriver .ami files are also ignored. Device noise is expected to be modeled in AMI\_GetWave functions.