**BUFFER ISSUE RESOLUTION DOCUMENT (BIRD)**

**Draft 3**

**ISSUE TITLE:** *IBIS-AMI New Reserved Parameters for Jitter/Noise*

**REQUESTOR:**  Walter Katz, Mike Steinberger, Todd Westerhoff, SiSoft

**DATE SUBMITTED:** October 20, 2010 April 1, June 11, 2012

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

Model developers and EDA vendors building IBIS-AMI models using the IBIS 5.0 specification have come across a number of modeling issues that are not addressed in IBIS 5.0. In order to deliver models and EDA tools that meet end-user demands for model accuracy and functionality, EDA vendors have defined "extensions" to add new capabilities to IBIS-AMI models. Unfortunately, EDA vendors have had to use proprietary (and different) syntax to add these capabilities to models, limiting model portability between different EDA tools.

This BIRD proposes new syntax for the .ami control file that improves model functionality and accuracy. Including this syntax in the IBIS standard will allow creation of accurate, compliant IBIS-AMI models that are readily portable between commercial EDA simulators.

The parameters defined in this document are to be added in Section 6c of the IBIS 5.0 specification as new Reserved\_Parameters.

Jitter, Noise and Clock Modeling

**Tx\_Rj, Tx\_Dj, Tx\_Sj, Tx\_Sj\_Frequency, Rx\_Clock\_Recovery\_Mean, Rx\_Clock\_Recovery\_Rj,**

**Rx\_Clock\_Recovery\_Dj, Rx\_Clock\_Recovery\_Sj, Rx\_Clock\_Recovery\_DCD, Rx\_Rj, Rx\_Dj, Rx\_Sj, Rx\_DCD, and Rx\_Noise.**

There are three sources of jitter that are accounted for using the parameters introduced in this BIRD; Tx Jitter, Rx Clock Data Recovery (CDR) Jitter and Rx Reference Clock Jitter. The Rx CDR has the ability to filter low frequency Tx Jitter and Rx Reference Clock Jitter. The parameters defined in this BIRD assume that the three sources of jitter are independent. IBIS 5.0 already defines parameters **Tx\_Jitter, Tx\_DCD and Rx\_Clock\_PDF**. **Tx\_DCD** is clarified in this BIRD. The parameters **Tx\_Rj, Tx\_Dj, Tx\_Sj, and Tx\_Sj\_Frequency** have similar functionality to the existing Tx\_Jitter, but offer more granularity in defining the various components of Tx Jitter. Similarly,**Rx\_Clock\_Recovery\_Mean, Rx\_Clock\_Recovery\_Rj, Rx\_Clock\_Recovery\_Dj, Rx\_Clock\_Recovery\_Sj, and Rx\_Clock\_Recovery\_DCD** offer increased granularity in defining the CDR behavior when doing statistical analysis, and when Rx AMI\_GetWave does not return clock\_times. **Rx\_Rj, Rx\_Dj, Rx\_Sj, and Rx\_DCD** describe jitter components that do not exist in IBIS 5.0, and offer the model maker a means to inform the EDA tool about how much extra jitter it should add to sampling instants. That is, these parameters indicate how much, as well as what type of, jitter is present in the actual device, but not reflected in the model. Rx\_Noise is unique among the parameters being introduced by this BIRD, in that it describes variations in the amplitude of the received signal, as opposed to variations in the time of sampling instants.

There are other possible methods of describing jitter. These include defining Tx and Rx Jitter Spectral Density distributions and applying Rx Jitter explicitly as reference clock transition times.These advanced methods of handling jitter are left for a future BIRD.

Each parameter defined in the BIRD has included both a verbal and a mathematical description of how that parameter would affect the Tx transition times, Rx CDR and Rx jitter not accounted for in the CDR. In the AMI statistical flow, these impairments are treated as distributions which affect both the clock PDF and statistical eye. In the AMI time domain flow, the EDA tool may apply these jitter parameters directly to the Tx stimulus input and the Rx clock\_times using the given equations. Since these jitter parameters are independent, the EDA tool may use other statistical methods to account for these impairments.

Please note that even if no intrinsic jitter were present in the Tx and Rx, one would still experience an eye that has apparent jitter at the Rx data decision point. This jitter is caused by ISI, which is, typically, non-zero despite the efforts of Tx and Rx Equalization.

The model maker may assume that any and all non-zero values passed in these parameters will be used by the EDA tool in one of the following two phases:

1) Input stimulus generation, or

2) Post-processing of simulation results.

The definition of Tx\_DCD is clarified, and the allowed Usage is changed to Info.

**ANY OTHER BACKGROUND INFORMATION:**

{*These documents will be archived, so use this section to add any detail that is not part of the section above or the changed text itself , but should not be lost.}*

**Change the following text on page 188**

*Descriptors*:

Usage: Info, Out

Type: Float, UI

Format: Value, Range, Corner, List, Increment, Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* Tx\_DCD (Transmit Duty Cycle Distortion) tells the EDA tool the maximum deviation of the duration of a transmitted pulse as a fraction of the nominal pulse width. Entries are assumed to be in units of seconds when declared as Type Float.

To:

*Descriptors*:

Usage: Info

Type: Float, UI

Format: Value, Range, Corner, List, Increment, Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* Tx\_DCD (Transmit Duty Cycle Distortion) defines half the peak to peak clock duty cycle distortion to be added to the behavior implemented by the EDA tool by modifying the stimulus input or by post processing the simulation

Time(n)=n\*bit\_time+Tx\_DCD\*(-1.0)^n

n\*bit\_time is the ideal time of the nth clock.

 Time(n) is the time of the nth clock modified when creating input waveforms for the Tx.

Entries are assumed to be in units of seconds when declared as Type Float. Note that all equations using jitter parameters that can be defined as UI shall be assumed to seconds in these formulae.

The following text is added immediately before Table ? on page ???: Jitter, Noise and Clock Parameters

The following optional Reserved Parameters are used to specify impairments for the transmitter output. These budgets specify the impairment as measured at the TX output (i.e. the transmitter output is expected to be directly modulated by these amounts). This data is used by the simulator to either modify the input stimulus presented to the algorithmic model or when post-processing the results from the model; the budget values specified by these parameters are not passed directly to the model itself.

*Parameter:* **Tx\_Rj**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* The standard deviation of a white Gaussian phase noise process at the transmitter which is to be added to the behavior implemented by the EDA tool by modifying the stimulus input or by post processing the simulation results. Entries are assumed to be in units of seconds when declared as Type Float.

*Usage Rules:*

*Other Notes:* Time(n)=n\*bit\_time+Tx\_Rj\*gaussian\_rand()

gaussian\_rand() is a function that returns floating point numbers between -inf and +inf. The distribution of these numbers shall be a white Gaussian distribution centered at 0.0 with a standard deviation of 1.0. The EDA tool can protect against abs(Tx\_Rj\*gaussian\_rand())>0.5UI.

*Example:*

(Tx\_Rj (Usage Info)(Corner 0.005 0.006 0.004)(Type UI)

 (Description "Tx Random Jitter in UI."))

*Parameter:* **Tx\_Dj**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* The worst case half the peak to peak variation at the transmitter implemented by the EDA tool by modifying the stimulus input or by post processing the simulation results. **Tx\_Dj** shall include all deterministic and uncorrelated bounded jitter that is not accounted for by **Tx\_DCD**, and **Tx\_Sj**. Entries are assumed to be in units of seconds when declared as Type Float.

*Usage Rules:*

*Other Notes:* Time(n)= n\*bit\_time+2.0\*Tx\_Dj\*rand()

rand() is a function that returns floating point numbers between -0.5 and

+0.5 with white uniform distribution.

*Example:*

(Tx\_Dj (Usage Info)(Value 0.1)(Type UI)

 (Description "Tx Bounded Jitter in UI."))

*Parameter:* **Tx\_Sj**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* Half the peak to peak amplitude of a sinusoidal jitter which is to be added to the behavior implemented directly by the transmitter model.

*Usage Rules:* If **Tx\_Sj\_Frequency** is not assigned (either in the model or by the user), **Tx\_Sj** should be ignored. Entries are assumed to be in units of seconds when declared as Type Float.

*Other Notes:* Time(n)=n\*bit\_time+**Tx\_Sj**\*sin((n\*bit\_time\*2.0\*Pi)\***Tx\_Sj\_Frequency**)

*Example:*

(Tx\_Sj (Usage Info)(Corner 0.005 0.006 0.004)(Type UI)

 (Description "Tx Sinosoidal Jitter in UI."))

*Parameter:* **Tx\_Sj\_Frequency**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* The frequency, in Hertz, of the sinusoidal jitter at the transmitter.

*Usage Rules:* If **Tx\_Sj\_Frequency** is not assigned (either in the model or by the user), **Tx\_Sj** should be ignored.

*Other Notes:* Time(n)=n\*bit\_time+**Tx\_Sj**\*sin((n\*bit\_time\*2.0\*Pi)\***Tx\_Sj\_Frequency**)

*Examples:*

(Tx\_Sj\_Frequency (Usage Info)(Corner 6.5E7 6.5E7 6.5E7)(Type Float)

 (Description "Tx Sinusoidal Jitter Frequency in Hz."))

The following optional Reserved Parameters are used to specify characteristics of the receiver's recovered clock. This data is used by the simulator when post-processing the results from the model when the model does not return clock\_times, or when Rx AMI\_GetWave is not used; the budget values specified by these parameters are not passed directly to the model itself. For Rx models that do return clock\_times by AMI\_GetWave, these parameters represent the amount of jitter THAT HAD ALREADY BEEN IMPLEMENTED BY RX AMI\_GETWAVE AND ALREADY INCLUDED IN THE RETURNED clock\_times. For this reason, the EDA platform should NOT apply these jitter parameters again to the Rx clock\_times. These parameters are provided by the model creator to the EDA platform and end users for the sole purpose that these jitters can be properly accounted for when Rx AMI\_GetWave is NOT used or Rx clock\_times was not returned, in which cases the EDA platform is responsible to apply these jitters to the Rx output."

"Rx\_Clock\_Recovery\_Mean" is an AMI parameter of Type either Float or UI, Format

either Value, List, Range, Corner, Increment, or Steps, and Usage Info which defines

a static offset, in seconds or UI, between the recovered clock and the point half

way between the PDF medians of consecutive eye zero crossings.

*Parameter:* **Rx\_Clock\_Recovery\_Mean**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* A static offset between the recovered clock and the point half way between the PDF medians of consecutive eye zero crossings. Entries are assumed to be in units of seconds when declared as Type Float.

*Usage Rules:*

*Other Notes:* actual\_time=ideal\_time+Rx\_Clock\_Recovery\_Mean

ideal\_time half way between the median of the eye crossing 0.0 on both sides of the eye.

*Examples:*

(Rx\_Clock\_Recovery\_Mean (Usage Info)(Value 0.05)

 (Type UI)(Description "Recovered Clock offset in UI."))

*Parameter:* **Rx\_Clock\_Recovery\_Rj**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* The standard deviation of a Gaussian phase noise exhibited by the recovered clock and included in the clock\_times vector returned by the AMI\_GetWave function. Entries are assumed to be in units of seconds when declared as Type Float.

*Usage Rules:*

*Other Notes:* actual\_time=ideal\_time+Rx\_Clock\_Recovery\_Rj\*gaussian\_rand()

*Examples:*

 (Rx\_Clock\_Recovery\_Rj (Usage Info)(Corner 0.005 0.006 0.004)

 (Type UI)(Description "RX Random Clock Jitter in UI."))

*Parameter:* **Rx\_Clock\_Recovery\_Dj**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* The worst case half the peak to peak variation of the recovered clock. **Rx\_Clock\_Recovery\_Dj** shall include all deterministic and uncorrelated bounded jitter that is included in the clock\_times vector returned by the AMI\_GetWave function and not accounted for by **Rx\_Clock\_Recovery\_DCD** and **Rx\_Clock\_Recovery\_Sj**. Entries are assumed to be in units of seconds when declared as Type Float. The **Rx\_Clock\_Recovery\_Dj** parameter can be used by those model makers not wishing to break down their deterministic and uncorrelated bounded jitter components into their Sj and DCD parts.

*Usage Rules:*

*Other Notes:* actual\_time = ideal\_time + 2.0\*Rx\_Clock\_Recovery\_Dj\*rand()

*Example:*

(Rx\_Clock\_Recovery\_Dj (Usage Info)(Value 0.1)(Type UI)

 (Description "Tx Bounded Jitter in UI."))

*Parameter:* **Rx\_Clock\_Recovery\_Sj**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* Half the peak to peak variation of a sinusoidal phase noise exhibited by the recovered clock and included in the clock\_times vector returned by the AMI\_GetWave function. Entries are assumed to be in units of seconds when declared as Type Float.

*Usage Rules:*

*Other Notes:* actual\_time = ideal\_time + Rx\_Clock\_Recovery\_Sj\*sin(Pi\*rand())

*Example:*

 (Rx\_Clock\_Recovery\_Sj (Usage Info)(Corner 0.05 0.07 0.4)(Type UI)

 (Description "RX Sinusoidal Jitter in UI."))

*Parameter:* **Rx\_Clock\_Recovery\_DCD**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* Half the peak to peak variation of a clock duty cycle distortion exhibited by the recovered clock and included in the clock\_times vector returned by the AMI\_GetWave function. Entries are assumed to be in units of seconds when declared as Type Float.

*Usage Rules:*

*Other Notes:* actual\_time=ideal\_time+Rx\_Clock\_Recovery\_DCD\*(-1.0)^n

*Example:*

 (Rx\_Clock\_Recovery\_DCD (Usage Info)(Corner 0.008 0.016 0.005)

 (Type UI)(Description "RX Duty Cycle Distortion in UI."))

The following optional Reserved Parameters are used to modify the statistics

associated with receiver's recovered clock. These parameters are used to account for

jitter that is not included in either the clock\_times returned by Rx AMI\_GetWave or

the Rx\_Clock\_Recovery parameters. This data is used by the simulator when post-

processing the results from the model; the budget values specified by these

parameters are not passed directly to the model itself.

*Parameter:* **Rx\_Rj**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* The standard deviation of a Gaussian phase noise driven by impairments external to the receiver that are input to the RX CDR, but are not included in the CDR clock\_times output. This phase noise is to be accounted for by the EDA tool, in both Statistical and Time-Domain simulations. Entries are assumed to be in units of seconds when declared as Type Float.

*Usage Rules:*

*Other Notes:* clock\_times(n)= time+Rx\_Rj \*gaussian\_rand()

time = ideal\_time in Statistical, and Time-Domain when clock\_times(n) is not available

 = clock\_times(n) in Time-Domain when clock\_times(n) is returned by Rx AMI\_Getwave

*Example:*

 (Rx\_Rj (Usage Info)(Corner 0.005 0.006 0.004)(Type UI)

 (Description "Rx Random Jitter in UI."))

*Parameter:* **Rx\_Dj**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* The worst case half peak to peak variation of the recovered clock.**Rx\_Dj** shall include all deterministic and uncorrelated bounded jitter that is not accounted for by either Rx clock\_times, **Rx\_Rj**, or Rx\_Clock\_Recovery parameters. This phase noise is to be accounted for by the EDA tool in both Statistical and Time-Domain simulations. Entries are assumed to be in units of seconds when declared as Type Float. The **Rx\_ Dj** parameter can be used by those model makers not wishing to break down their deterministic and uncorrelated bounded jitter components into their Sj and DCD parts.

*Usage Rules:*

*Other Notes:* actual\_time = time + 2.\*Rx\_Dj\*rand()

time = ideal\_time in Statistical, and Time-Domain when clock\_times(n) is not available

 = clock\_times(n) in Time-Domain when clock\_times(n) is returned by Rx AMI\_Getwave

*Example:*

 (Rx\_Dj (Usage Info)(Value 0.1)(Type UI)

 (Description "Tx Bounded Jitter in UI."))

*Parameter:* **Rx\_Sj**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* Half the peak to peak variation of a sinusoidal phase noise, but are not included in the CDR clock\_times output. This phase noise is to be accounted for by the EDA tool in both Statistical and Time-Domain simulations. Entries are assumed to be in units of seconds when declared as Type Float.

*Usage Rules:*

*Other Notes:* actual\_time = time+ Rx\_Sj\*sin(Pi\*rand())

time = ideal\_time in Statistical, and Time-Domain when clock\_times(n) is not available

 = clock\_times(n) in Time-Domain when clock\_times(n) is returned by Rx AMI\_Getwave

*Example:*

 (Rx\_Sj (Usage Info)(Corner 0.05 0.07 0.04)(Type UI)

 (Description "RX Sinusoidal Jitter in UI."))

*Parameter:* **Rx\_DCD**

*Required:* No

*Descriptors*:

Usage: Info

Type: Float or UI

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* Half the peak to peak variation of a clock duty cycle distortion. This phase noise is to be accounted for by the EDA tool in both Statistical and Time-Domain simulations. Entries are assumed to be in units of seconds when declared as Type Float.

*Usage Rules:*

*Other Notes:* actual\_time = time + Rx\_DCD\*(-1.0)^n

 n is the nth clock

time = ideal\_time in Statistical, and Time-Domain when clock\_times(n) is not available

 = clock\_times(n) in Time-Domain when clock\_times(n) is returned by Rx AMI\_Getwave

*Example:*

 (Rx\_DCD (Usage Info)(Corner 0.008 0.016 0.005)(Type UI)

 (Description "RX Duty Cycle Distortion in UI."))

The following optional Reserved Parameter is used to modify the statistics associated with the data input to the receiver's sampling latch (a.k.a. 'slicer'). This data is used by the simulator when post-processing the results from the model; the budget values specified by this parameter are not passed directly to the model itself.

*Parameter:* **Rx\_Noise**

*Required:* No

*Descriptors*:

Usage: Info or Out

Type: Float

Format: Value, List, Range, Corner, Increment, or Steps

Default: <numeric\_literal*>*

Description:<string>

*Definition:* The standard deviation, in Volts, of a white Gaussian random process, which is to be added by the EDA tool to the signal measured at the sampling latch of a receiver.

*Usage Rules:* If Rx\_Noise is Usage Out, then the EDA tool shall use the value returned by Rx AMI\_Init if Rx AMI\_GetWave is not used. If Rx AMI\_GetWave is used, then the EDA tool may apply the value returned by each AMI\_GetWave call to the waveform returned by that call to AMI\_GetWave, or use the average value of **Rx\_Noise** returned by all calls to AMI\_GetWave (after **Ignore\_Bits**), or the value of **Rx\_Noise** returned by the last call to AMI\_GetWave.

*Other Notes:* wave(t)=wave(t)+Rx\_Noise\*gaussian\_rand()

wave(t) is the waveform returned by Rx AMI\_GetWave

*Example:*

 (Rx\_Noise (Usage Info)(Value .010) (Type Float)

 (Description "Rx amplitude noise at sampling latch in Volts."))

Note:

The "Rx\_Clock\_Recovery Parameters" (**Rx\_Clock\_PDF**, **Rx\_Clock\_Recovery\_Mean**, **Rx\_Clock\_Recovery\_Rj**, **Rx\_Clock\_Recovery\_Dj**, **Rx\_Clock\_Recovery\_Sj** and **Rx\_Clock\_Recovery\_DCD**, should be used by the simulator when analyzing the output of Rx AMI\_Init (for statistical analysis) or Rx AMI\_GetWave (time domain) when Rx AMI\_GetWave does not return clock\_times. When Rx AMI\_GetWave returns clock\_times, the simulator should not use the "Rx\_Clock\_Recovery Parameters".

Note:

The "Rx Jitter Parameters" (**Rx\_Rj, Rx\_Dj, Rx\_Sj** and **Rx\_DCD**, should be used by the simulator when analyzing the output of either Rx AMI\_Init (for statistical analysis) or Rx AMI\_GetWave (for time domain analysis).

Tables summarizing the rules for the jitter, noise and sensitivity parameters for information only.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 +------------------------+-------------------+

 | General Rules | Allowed Usage |

 ============================================================================

 | Reserved Parameter | Required Default | Info In Out InOut |

 +-----------------------------+------------------------+-------------------+

 | Tx\_Jitter | No No Jitter | X |

 | Tx\_Dj | No 0 | X |

 | Tx\_Rj | No 0 | X |

 | Tx\_Sj | No 0 | X |

 | Tx\_DCD | No 0 | X |

 | Tx\_Sj\_Frequency | No Undefined | X |

 | Rx\_Receiver\_Sensitivity | No 0 | X X |

 | Rx\_Clock\_PDF | No Clock Centered | X |

 | Rx\_Clock\_Recovery\_Mean | No 0 | X |

 | Rx\_Clock\_Recovery\_Dj | No 0 | X |

 | Rx\_Clock\_Recovery\_Rj | No 0 | X |

 | Rx\_Clock\_Recovery\_Sj | No 0 | X |

 | Rx\_Clock\_Recovery\_DCD | No 0 | X |

 | Rx\_Dj | No 0 | X |

 | Rx\_Rj | No 0 | X |

 | Rx\_Sj | No 0 | X |

 | Rx\_DCD | No 0 | X |

 | Rx\_Noise | No 0 | X X |

 +-----------------------------+------------------------+-------------------+

 Table 1: General Rules and Allowed Usage for Reserved Parameters

?

 +-------------------------------------------+

 | Data Type |

 ===========================================================================

 | Reserved Parameter | Float | UI | Integer | String | Boolean |

 +-----------------------------+-------+------+---------+--------+---------+

 | Tx\_Jitter | X X |

 | Tx\_Dj | X X |

 | Tx\_Rj | X X |

 | Tx\_Sj | X X |

 | Tx\_DCD | X X |

 | Tx\_Sj\_Frequency | X |

 | Rx\_Receiver\_Sensitivity | X |

 | Rx\_Clock\_PDF | X X |

 | Rx\_Recovery\_Mean | X X |

 | Rx\_Clock\_Recovery\_Dj | X X |

 | Rx\_Clock\_Recovery\_Rj | X X |

 | Rx\_Clock\_Recovery\_Sj | X X |

 | Rx\_Clock\_Recovery\_DCD | X X |

 | Rx\_Dj | X X |

 | Rx\_Rj | X X |

 | Rx\_Sj | X X |

 | Rx\_DCD | X X |

 | Rx\_Noise | X |

 +-----------------------------+-------------------------------------------+

 Table 2: Allowed Data Types for Reserved Parameters

?

 +---------------------------------------+

 | Data Format |

 =======================================================================

 | Reserved Parameter | V | R | C | L | I | S | G | D | D | T |

 | | a | a | o | i | n | t | a | u | j | a |

 | | l | n | r | s | c | e | u | a | R | b |

 | | u | g | n | t | r | p | s | l | j | l |

\*| | e | e | e | | e | s | s | - | | e |

\*| | | | r | | m | | i | D | | |

\*| | | | | | e | | a | i | | |

\*| | | | | | n | | n | r | | |

\*| | | | | | t | | | a | | |

\*| | | | | | | | | c | | |

 +-----------------------------+---+---+---+---+---+---+---+---+---+---+

 | Tx\_Jitter | X X X X |

 | Tx\_Dj | X X X X X X |

 | Tx\_Rj | X X X X X X |

 | Tx\_Sj | X X X X X X |

 | Tx\_DCD | X X X X X X |

 | Tx\_Sj\_Frequency | X X X X X X |

 | Rx\_Receiver\_Sensitivity | X X X X X X |

 | Rx\_Clock\_PDF | X X X X |

 | Rx\_Recovery\_Mean | X X X X X X |

 | Rx\_Clock\_Recovery\_Dj | X X X X X X |

 | Rx\_Clock\_Recovery\_Rj | X X X X X X |

 | Rx\_Clock\_Recovery\_Sj | X X X X X X |

 | Rx\_Clock\_Recovery\_DCD | X X X X X X |

 | Rx\_Dj | X X X X X X |

 | Rx\_Rj | X X X X X X |

 | Rx\_Sj | X X X X X X |

 | Rx\_DCD | X X X X X X |

 | Rx\_Noise | X X X X X X |

 +-----------------------------+---------------------------------------+

 Table 3: Allowed Data Format for Reserved Parameters

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

With the exception of the "Table" format, the Tx\_Jitter parameter has been essentially superseded by the Reserved\_Parameters Tx\_Rj, Tx\_Dj, Tx\_Sj, Tx\_Sj\_Frequency, and Tx\_DCD, which enable SerDes transmitter jitter to be specified in greater detail. It is recommended for AMI model developers to use these preferred jitter parameters when possible instead of Tx\_Jitter.With the exception of the "Table" format, the Rx\_Clock\_PDF parameter has been essentially superseded by the Reserved\_Parameters Rx\_Clock\_Recovery\_Rj, Rx\_Clock\_Recovery\_Dj, Rx\_Clock\_Recovery\_Sj, and Rx\_Clock\_Recovery\_DCD, which enable SerDes receiver jitter to be specified in greater detail. It is recommended for AMI model developers to use these preferred jitter parameters when possible instead of

Rx\_Clock\_PDF.?

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ANALYSIS PATH/DATA THAT LED TO SPECIFICATION

The parameters defined in this BIRD came from commercial IBIS-AMI model development efforts where new functionality was needed to meet customer expectations for model functionality, accuracy and performance. The parameters in this BIRD were defined by SiSoft and its semiconductor partners. These parameters are being contributed to IBIS to ensure IBIS-AMI model accuracy and portability.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ANY OTHER BACKGROUND INFORMATION:

This BIRD is being requested by the following IBIS users and model developers, in conjunction with the authors:

Cisco Systems: Upen Reddy, Doug White

Ericsson: Anders Ekholm

Broadcom: Yunong Gan

IBM: Adge Hawes

TI: Alfred Chong, Srikanth Sundaram

Markup copies of this document, in Adobe PDF\* and Microsoft Word\* format, are

available at:

http://www.eda.org/ibis/birds/bird123.3/bird123.3\_markup.pdf

http://www.eda.org/ibis/birds/bird123.3/bird123.3\_markup.docx

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*