```
******************************************************************************
******************************************************************************

BIRD ID#:         123.~~2~~3
ISSUE TITLE:      IBIS-AMI New Reserved Parameters for Jitter/Noise
AUTHOR:           Walter Katz, Mike Steinberger, Todd Westerhoff, SiSoft
DATE SUBMITTED:  October 20, 2010
DATE REVISED:    April 1, 2011; January 3, 2012; May 8, 2012
DATE ACCEPTED BY IBIS OPEN FORUM:

******************************************************************************
******************************************************************************
```

STATEMENT OF THE ISSUE:

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.

****

The following parameter exists in the IBIS 5.0 specification but its definition is replaced using the text in this BIRD:

Tx_DCD

On page 146 replace:

```
|                Tx_DCD:
|
|                Tx_DCD (Transmit Duty Cycle Distortion) can be of Usage Info
|                and Out.  It can be of Type Float and UI and can have Data
|                Format of Value, Range and Corner.  It tells the EDA platform
|                the maximum percentage deviation of the duration of a
|                transmitted pulse from the nominal pulse width.  Example of
|                TX_DCD declaration is:
|
|                (Tx_DCD (Usage Info)(Type Float)
|                        (Format Range <typ> <min> <max>))
```

with:

```
|                Tx_DCD:
|
|                Tx_DCD (Transmit Duty Cycle Distortion) must be of Usage Info.
|                It can be of Type Float or UI and can have
|                Format either Value, List, Range, Corner, Increment, or
|                Steps.  It defines half the peak
|                to peak clock duty cycle distortion, in seconds or UI, to be
|                added to the behavior implemented by the EDA tool by modifying
|                the stimulus input or by post processing the simulation
|                results.
|
|                Example of TX_DCD declaration is:
|
|                (Tx_DCD (Usage Info)(Corner 0.008 0.016 0.005)(Type UI)
|                        (Description "TX Duty Cycle Distortion in UI.")
|       )
```

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.
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 1 on page 148: 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.

"Tx_Rj" is an AMI parameter of Type either Float or UI, Format either Value, List,
Range, Corner, Increment, or Steps, and Usage Info which defines the standard
deviation, in seconds or UI, of an 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.

Example:

```
      (Tx_Rj (Usage Info)(Corner 0.005 0.006 0.004)(Type UI)
         (Description "Tx Random Jitter in UI.")
)
```

"Tx_Sj
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 an
white Gaussian distribution centered at ~~zero~~ 0.0 with a standard deviation
of 1.0. The EDA tool can protect against abs(Tx_Rj*gaussian_rand())>0.5UI.

"Tx_Dj" is an AMI parameter of Type either Float or UI, Format either Value, List,
Range, Corner, Increment, or Steps, and Usage Info which defines the worst case half
the peak
to peak variation, in seconds or UI, 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.

Example:

```
      (Tx_Dj (Usage Info)(Value 0.1)(Type UI)
         (Description "Tx Bounded Jitter in UI.")
)
```

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.

"Tx_Sj" is an AMI parameter of Type either Float or UI, Format either Value, List, Range, Corner, Increment, or Steps, and Usage Info which defines half the peak to peak amplitude, in seconds or UI, of a sinusoidal jitter which is to be added to the behavior implemented directly by the transmitter model.

Example:

```
      (Tx_Sj (Usage Info)(Corner 0.05 0.07 0.4)(Type UI)
         (Description "Tx Sinusoidal Jitter in UI.")
)
```

Note: If Tx_Sj_Frequency is not assigned (either in the model or by the user), Tx_Sj should be ignored.

"Tx_Sj_Frequency" is an AMI parameter of Type Float, Format Value, and Usage Info which defines the frequency, in Hertz, of the sinusoidal jitter at the transmitter.

Example:

```
      (Tx_Sj_Frequency (Usage Info)(Corner 6.5E7 6.5E7 6.5E7)(Type Float)
         (Description "Tx Sinusoidal Jitter Frequency in Hz.")
)
```

Time(n)=n*bit_time+Tx_Sj*sin((n*bit_time*2.0*Pi)*Tx_Sj_Frequency)

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.

Example:

```
      (Rx_Clock_Recovery_Mean (Usage Info)(Value 0.05)
         (Type UI)(Description "Recovered Clock offset in UI.")
)
```

```
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.

"Rx_Clock_Recovery_Rj" is an AMI parameter of Type either Float or UI, Format either Value, List, Range, Corner, Increment, or Steps, and Usage Info which defines the standard deviation, in seconds or UI, of a Gaussian phase noise exhibited by the recovered clock and included in the clock_times vector returned by the AMI_GetWave function.

Example:

```
      (Rx_Clock_Recovery_Rj (Usage Info)(Corner 0.005 0.006 0.004)
         (Type UI)(Description "RX Random Clock Jitter in UI.")
)
```

```
actual_time=ideal_time+Rx_Clock_Recovery_Rj*gaussian_rand()
```

"Rx_Clock_Recovery_Dj" is an AMI parameter of Type either Float or UI, Format either Value, List, Range, Corner, Increment, or Steps, and Usage Info which defines the worst case half the peak to peak variation, in seconds or UI, 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.

Example:

```
      (Rx_Clock_Recovery_Dj (Usage Info)(Value 0.1)(Type UI)
         (Description "Tx Bounded Jitter in UI.")
)
```

```
actual_time = ideal_time + 2.0*Rx_Clock_Recovery_Dj*rand()
```

"Rx_Clock_Recovery_Sj" is an AMI parameter of Type either Float or UI, Format either Value, List, Range, Corner, Increment, or Steps, and Usage Info which defines half the peak to peak variation, in seconds or UI, of a sinusoidal phase noise exhibited by the recovered clock and included in the clock_times vector returned by the AMI_GetWave function.

Example:

```
      (Rx_Clock_Recovery_Sj (Usage Info)(Corner 0.05 0.07 0.4)(Type UI)
         (Description "RX Sinusoidal Jitter in UI."))
```

actual_time = ideal_time + Rx_Clock_Recovery_Sj*sin(Pi*rand())

~~clock_times(n)=clock_times(n)+Rx_Clock_Recovery_Sj*sin(Pi*rand())~~
~~rand()is a function that returns floating point numbers between -.5 and +.5. The distribution of these numbers shall be an uncorrelated uniform distribution between -.5 and +.5.~~

"Rx_Clock_Recovery_DCD" is an AMI parameter of Type either Float or UI, Format either Value, List, Range, Corner, Increment, or Steps, and Usage Info which defines half the peak to peak variation, in seconds or UI, of a clock duty cycle distortion exhibited by the recovered clock and included in the clock_times vector returned by the AMI_GetWave function.

Example:

```
      (Rx_Clock_Recovery_DCD (Usage Info)(Corner 0.008 0.016 0.005)
   (Type UI)(Description "RX Duty Cycle Distortion in UI.")
)
```

actual_time=ideal_time+Rx_Clock_Recovery_DCD*(-1.0)^n

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.

"Rx_Rj" is an AMI parameter of Type either Float or UI, Format either Value, List, Range, Corner, Increment, or Steps, and Usage Info which defines the standard deviation, in seconds or UI, 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.

Example:

```
      (Rx_Rj (Usage Info)(Corner 0.005 0.006 0.004)(Type UI)
         (Description "Rx Random Jitter in UI.")
)
```

clock_times(n)= ~~clock_times(n)~~ time+Rx_Rj *gaussian_rand()
~~clock_times(n) is the times returned by Rx AMI_Getwave~~
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

"Rx_Dj" is an AMI parameter of Type either Float or UI, Format either Value, List, Range, Corner, Increment, or Steps, and Usage Info which defines the worst case half peak to peak variation, in seconds or UI, of the recovered clock, not including the random jitter specified by Rx_Rj, Rx_Sj, or Rx_DCD . 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.

Example:

```
      (Rx_Dj (Usage Info)(Value 0.1)(Type UI)
         (Description "Tx Bounded Jitter in UI.")
)
```

actual_time = ~~ideal_~~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

"Rx_Sj" is an AMI parameter of Type either Float or UI, Format either Value, List, Range, Corner, Increment, or Steps, and Usage Info which defines half the peak to peak variation, in seconds or UI, 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.

```
      (Rx_Sj (Usage Info)(Corner 0.05 0.07 0.04)(Type UI)
         (Description "RX Sinusoidal Jitter in UI.")
      )
```

actual_time = ~~ideal_time~~ time+ Rx_Sj*sin(Pi*rand())
~~rand() Returns random numbers between -.5 and +.5~~ 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

"Rx_DCD" is an AMI parameter of Type either Float or UI, Format either Value, List, Range, Corner, Increment, or Steps, and Usage Info which defines half the peak to peak variation, in seconds or UI, 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.

Example:

```
      (Rx_DCD (Usage Info)(Corner 0.008 0.016 0.005)(Type UI)
         (Description "RX Duty Cycle Distortion in UI.")
)
```

actual_time = ~~ideal_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

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.

"Rx_Noise" is an AMI parameter of Type Float, Format either Value, List, Range, Corner, Increment, or Steps, and Usage either Info or Out which defines 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.

Example:

```
   (Rx_Noise (Usage Info)(Value .010) (Type Float)
         (Description "Rx amplitude noise at sampling latch in Volts.")
)
```

```
wave(t)=wave(t)+Rx_Noise*gaussian_rand()
wave(t) is the waveform returned by Rx AMI_GetWave
```

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.

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	Value	Range	Corner	List	Increment	Steps	Gaussian	Dual-Dirac	DjRj	Table
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/bird123.2/bird123.2_markup.pdf
http://www.eda.org/ibis/birds/bird123.2/bird123.2_markup.docx

****