****
****

BIRD ID#:
ISSUE TITLE: IBIS-AMI clock_times Clarification
REQUESTER: Scott McMorrow, Teraspeed Consulting Group
DATE SUBMITTED:
DATE REVISED:
DATE ACCEPTED BY IBIS OPEN FORUM:

****
****

STATEMENT OF THE ISSUE:

In the Section, "NOTES ON ALGORITHMIC MODELING INTERFACE AND PROGRAMMING GUIDE", the paragraph describing clock_times has led to inconsistent and incorrect model implementation.

The suggestion is to clarify usage of the parameter consistent with the original intent.

****

Replace this text:
|
| 3.2.2.3 clock_times
| ===================
|
| Vector to return clock times. The clock times are referenced to the start
| of the simulation (the first AMI_GetWave call). The time is always
| greater or equal to zero. The last clock is indicated by putting a value
| of -1 at the end of clocks for the current wave sample. The clock_time
| vector is allocated by the EDA platform and is guaranteed to be greater
| than the number of clocks expected during the AMI_GetWave call. The clock
| times are the times at which clock signal at the output of the clock
| recovery loop crosses the logic threshold. It is to be assumed that the
| input data signal is sampled at exactly one half clock period after a
| clock time.
|

With the following text with changes noted by "|*" lines:

|
| 3.2.2.3 clock_times
| ===================
|
| Vector to return clock times. The clock times are referenced to the start
| of the simulation (the first AMI_GetWave call). The time is always
|* greater or equal to zero.   The clock_time
|* vector is allocated by the EDA platform and is guaranteed to be greater
|* than the number of clocks expected during the AMI_GetWave call. The clock
|* times are the times at which the clock signal at the output of the clock
|* recovery loop crosses the logic threshold in a full data rate CDR clocking
|* system (i.e clock period equals UI). The effective receiver sampling
|* point is equal to the clock_times plus 1/2 the nominal UI period. The last
|* valid clock of the current GetWave call is indicated
|* by placing -1 after the last valid clock in the clock_time vector.
|*
|* The clock ticks represented by clock times should be strictly monotonic,
|* both within the clock_times array returned from a single call to GetWave
|* and between successive calls to GetWave. That is, within a given clock_times

|* array each successive valid value is greater than the value that preceded it,
|* and the first valid value from a given call to GetWave must be greater than
|* the last valid value from the preceding call to GetWave.Any non-strict-monotonic
|* behavior of clock times (including two identical values) should be considered
|* by EDA platform as a DLL failure and should lead to simulation termination
|* with respective message.
|*
|* Each valid pair of values in the clock_times array shall be used to sample the output
|* waveform as previously described, regardless whether that waveform sample occurs
|* in the waveform segment being returned by the current call to GetWave, or the
|* waveform segment to be returned by the next GetWave call.
|*
|* Although clock_times will generally be related to the UI interval for the
|* primary SerDes channel being simulated, there is no requirement that there is
|* any relationship between the clock ticks generated by clock_times and the actual
|* waveform returned in the primary channel. It is possible for the CDR to go out
|* of lock, resulting in clock_ticks that have no definite relationship to the output
|* wave.
|*
|* There is no requirement that clock times should be integer multiples of the
|* sample interval (or time step used to represent the waveforms).There is also
|* no requirement that there be a relationship between clock_times in the
|* primary channel, and any additional waveform components in the wave vector, such
|* as crosstalk. Crosstalk channels shall not be constrained to any timing
|* relationship to the primary channel, or to the clock_times vector.
|

****

ANALYSIS PATH/DATA THAT LED TO SPECIFICATION

Additional notes regarding correct clock_times usage have been included as part
of this BIRD, distilled from discussions on the ibis-macro reflector.

Additional notes regarding clock_times

* Internal to a device, the sampling time tick "sees" the part of the waveform
  that immediately precedes and follows that tick, within some sampling uncertainty
window.
* That point, is the true center of the eye for that interval.
* The AMI spec requires the clock tick to be placed 1/2 UI before the actual
  sample point, essentially at the differential crossing.
* Then it requires the EDA tool to shift the tick by 1/2 UI.
    - the assumption here is that there is always a fixed relationship to the UI.
    - Thus the DLL must calculate the sampling point, then move it back by 1/2 UI
      to create a clock tick that can then be moved back by the EDA platform to
      the same sampling point that it first calculated.
* It is therefore a requirement that the DLL move the sample point back by 1/2 the
  nominal UI, and not the instantaneous UI, otherwise there will be inadvertent
  jitter in the clock_times.
* Clock_times + 1/2 nominal UI is always the center of every eye interval.

****

ANY OTHER BACKGROUND INFORMATION:

This is an editorial correction to clarify the usage of clock_times.

****