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

BIRD ID#:        107.1
ISSUE TITLE:     Update to Algorithmic Modeling API (AMI) Support in IBIS
REQUESTER:       Todd Westerhoff, SiSoft and Zhen Mu, Cadence Design Systems
DATE SUBMITTED:  April 8, 2008
DATE REVISED:
DATE ACCEPTED BY IBIS OPEN FORUM:  PENDING

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

STATEMENT OF THE ISSUE:

The waveform input to a TX AMI_Getwave call and the filtering function to be performed by the AMI_Getwave call were ambiguous. This was causing simulation differences when the same TX model was run in different IBIS-AMI toolkits.

The analysis flow when an AMI model contained filtering in both the AMI_Init and AMI_Getwave calls was not clearly stated. This was causing simulation differences when the same TX model was run in different IBIS-AMI toolkits.

The existing text did not make it clear that the AMI model was responsible for deallocating memory used for AMI_parameters_out.

The existing text has three different definitions of the modified impulse response generated by AMI_Init in sections 2.1.6, 2.2.6 and 3.1.2.1. The existing text did not make clear that if Init returns a modified impulse that the modified impulse response represents the filtered response.

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

Modifications to section 6C:

The following paragraph:

```
|              Init_Returns_Impulse, GetWave_Exists, Max_Init_Aggressors and
|              Ignore_Bits
```

shall be modified as follows:

```
|              Init_Returns_Impulse, Use_Init_Output, GetWave_Exists,
|              Max_Init_Aggressors and Ignore_Bits
```

```
------------------------------------------------------------------------------
```

The following paragraph:

```
|              The following reserved parameters are optional.  If these
|              parameters are not present, the values are assumed as “0”.
```

shall be modified as follows:

```
|              Use_Init_Output:
|
|              Use_Init_Output is of usage Info and type Boolean.  When
```

Use_Init_Output
| is set to "True", the EDA tool is instructed to use the output impulse response
| from the AMI_Init function when creating the input waveform presented to
| the AMI_Getwave function.
|
| If the Reserved Parameter, Use_Init_Output, is set to "False", EDA tools will
| use the original (unfiltered) impulse response of the channel when creating the
| input waveform presented to the AMI_Getwave function.
|
| The algorithmic model is expected to modify the waveform in place.
|
| Use_Init_Output is optional. The default value for this parameter is "True".
|
| If Use_Init_Output is False, GetWave_Exists must be True.
|
| The following reserved parameters are optional.
| If the following parameters are not present, the values are assumed as “0”.

---

The following tables:

| | General | Rules | Allowed Usage | | | |
|---|---|---|---|---|---|---|
| Reserved Parameter | Required | Default | Info | In | Out | InOut |
| Init Returns Impulse | Yes | NA | X | | | |
| GetWave Exists | Yes | NA | X | | | |
| Ignore Bits | No | 0 | X | | X | |
| Max Init Aggressors | No | 0 | X | | | |
| Tx Jitter | No | No Jitter | X | | X | |
| Tx DCD | No | 0 | X | | X | |
| Rx Receiver Sensitivity | No | 0 | X | | X | |
| Rx Clock PDF | No | Clock Centered | X | | X | |

| Table 1: General Rules and Allowed Usage for Reserved Parameters

| | Data Type | | | | |
|---|---|---|---|---|---|
| Reserved Parameter | Float | UI | Integer | String | Boolean |
| Init Returns Impulse | | | | | X |
| GetWave Exists | | | | | X |
| Ignore Bits | | | X | | |
| Max Init Aggressors | | | X | | |
| Tx Jitter | X | X | | | |
| Tx DCD | X | X | | | |
| Rx Receiver Sensitivity | X | | | | |
| Rx Clock PDF | X | X | | | |

Table 2: Allowed Data Types for Reserved Parameters

| | Data Format | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Reserved Parameter | Value | Range | Corner | List | Incr | Steps | Gaus | DualDirac | DjRj | Table |
| Init Returns Impulse | X | | | | | | | | | |
| GetWave Exists | X | | | | | | | | | |
| Ignore Bits | X | | | | | | | | | |
| Max Init Aggressors | X | | | | | | | | | |
| Tx Jitter | | | | | | | X | X | X | X |
| Tx DCD | X | X | X | | | | | | | |
| Rx Receiver Sensitivity | X | X | X | | | | | | | |
| Rx Clock PDF | | | | | | | X | X | X | X |

shall be modified as follows:

| | General Rules | | Allowed Usage | | | |
|---|---|---|---|---|---|---|
| Reserved Parameter | Required | Default | Info | In | Out | InOut |
| Init Returns Impulse | Yes | NA | X | | | |
| GetWave Exists | Yes | NA | X | | | |
| Use Init Output | No | True | X | | | |
| Ignore Bits | No | 0 | X | | X | |
| Max Init Aggressors | No | 0 | X | | | |
| Tx Jitter | No | No Jitter | X | | X | |
| Tx DCD | No | 0 | X | | X | |
| Rx Receiver Sensitivity | No | 0 | X | | X | |
| Rx Clock PDF | No | Clock Centered | X | | X | |

Table 1: General Rules and Allowed Usage for Reserved Parameters

| | Data Type | | | | |
|---|---|---|---|---|---|
| Reserved Parameter | Float | UI | Integer | String | Boolean |
| Init Returns Impulse | | | | | X |
| GetWave Exists | | | | | X |
| Use Init Output | | | | | X |
| Ignore Bits | | | X | | |
| Max Init Aggressors | | | X | | |

| | | |
|---|---|---|
| Tx Jitter | X | X |
| Tx DCD | X | X |
| Rx Receiver Sensitivity | X | |
| Rx Clock PDF | X | X |

Table 2: Allowed Data Types for Reserved Parameters

| | Data Format | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Reserved Parameter | Value | Range | Corner | List | Incr | Steps | Gaus | DualDirac | DjRj | Table |
| Init Returns Impulse | X | | | | | | | | | |
| GetWave Exists | X | | | | | | | | | |
| Use Init Output | X | | | | | | | | | |
| Ignore Bits | X | | | | | | | | | |
| Max Init Aggressors | X | | | | | | | | | |
| Tx Jitter | | | | | | | X | X | X | X |
| Tx DCD | X | X | X | | | | | | | |
| Rx Receiver Sensitivity | X | X | X | | | | | | | |
| Rx Clock PDF | | | | | | | X | X | X | X |

*****************************************************************************

Modifications to section 10:

The following index:

```
| 2 APPLICATION SCENARIOS
|   2.1 Linear, Time-invariant equalization Model
|   2.2 Nonlinear, and / or Time-variant equalization Model
```

shall be modified as follows:

```
| 2 APPLICATION SCENARIOS
|   2.1 Linear, Time-invariant equalization Model
|   2.2 Nonlinear, and / or Time-variant equalization Model
|   2.3 Reference system analysis flow
```

-----------------------------------------------------------------------------

The following paragraph 2.1.6

```
|   6. AMI_Init parses the configuration parameters, allocates dynamic memory,
|      places the address of the start of the dynamic memory in the memory
|      handle, computes the impulse response for the [Model] and passes it
|      back to the EDA platform.
```

shall be modified as follows:

```
|   6. AMI_Init parses the configuration parameters, allocates dynamic memory,
|      places the address of the start of the dynamic memory in the memory
|      handle, computes the impulse response of the block and passes the modified
|      impulse response to the EDA tool. The new impulse response is expected to
|      represent the filtered response.
```

------------------------------------------------------------------------------

The following paragraph 2.2.6

```
|  6. AMI_Init parses the configuration parameters, allocates dynamic memory
|     and places the address of the start of the dynamic memory in the memory
|     handle.  AMI_Init may also compute the impulse response of the block
|     and pass the modified impulse response to the EDA tool.
```

shall be modified as follows:

```
|  6. AMI_Init parses the configuration parameters, allocates dynamic memory
|     and places the address of the start of the dynamic memory in the memory
|     handle.  AMI_Init may also compute the impulse response of the block
|     and pass the modified impulse response to the EDA tool. The new impulse
|     response is expected to represent the filtered response.
```

------------------------------------------------------------------------------

The following paragraph 2.2.10:

```
| 10. The EDA platform uses the output of the receiver AMI_GetWave function
| to complete the simulation/analysis.  For transmitter, it simply passes
| the output to the receiver AMI_GetWave.
```

shall be modified as follows:

```
| 10. The EDA platform uses the output of the receiver AMI_GetWave function
| to complete the simulation/analysis.
```

------------------------------------------------------------------------------

Insert the following text for item 2.3 immediately before item 3, "FUNCTION SIGNATURES"

```
|  2.3 Reference system analysis flow
|
|  System simulations will commonly involve both TX and RX algorithmic models,
|  each of which may perform filtering in the AMI_Init call, the AMI_Getwave call,
|  or both.  Since both LTI and non-LTI behavior can be modeled with algorithmic
models,
|  the manner in which models are evaluated can affect simulation results.  The
|  following steps are defined as the reference simulation flow.  Other methods
|  of calling models and processing results may be employed, but the final
simulation
|  waveforms are expected to match the waveforms produced by the reference
simulation
```

```
|  flow.
|
|  The steps in this flow are chained, with the input to each step being the output
|  of the step that preceded it.
|
|  Step 1. The simulation platform obtains the impulse response for the analog
channel. This
|          represents the combined impulse response of the transmitter's analog
output,
|          the channel and the receiver's analog front end.  This impulse response
represents
|          the transmitter's output characteristics without filtering, for example,
|          equalization.
|
|  Step 2. The output of Step 1 is presented to the TX model's AMI_Init call.  If
|          Use_Init_Output for the TX model is set to True, the impulse response
returned by
|          the TX AMI_Init call is passed onto Step 3.  If Use_Init_Output for the
TX model
|          is set to False, the same impulse response passed into Step 2 is passed
on to
|          step 3.
|
|  Step 3. The output of Step 2 is presented to the RX model's AMI_Init call.  If
|          Use_Init_Output for the RX model is set to True, the impulse response
returned by
|          the RX AMI_Init call is passed onto Step 4.  If Use_Init_Output for the
RX model
|          is set to False, the same impulse response passed into Step 3 is passed
on to
|          step 4.
|
|  Step 4. The simulation platform takes the output of step 3 and combines
|          (for example by convolution) the input bitstream and a unit pulse
|          to produce an analog waveform.
|
|  Step 5. The output of step 4 is presented to the TX model's AMI_Getwave call.
If
|          the TX model does not include an AMI_Getwave call, this step is a pass-
through
|          step, and the input to step 5 is passed directly to step 6.
|
|  Step 6. The output of step 5 is presented to the RX model's AMI_Getwave call. If
|          the RX model does not include an AMI_Getwave call, this step is a pass-
through
|          step, and the input to step 6 is passed directly to step 7.
|
|  Step 7. The output of step 6 becomes the simulation waveform output at the RX
|          decision point, which may be post-processed by the simulation tool.
|
|  Steps 4 though 7 can be called once or can be called multiple times to process
the full
|  analog waveform.  Splitting up the full analog waveform into mulitple calls
minimized the
|  memory requirement when doing long simulations, and allows AMI_Getwave to return
model
|  status every so many bits. Once all blocks of the input waveform have been
processed,
|  TX AMI_Close and RX AMI_close are called to perform any final processing and
```

```
release
|  allocated memory.
|

------------------------------------------------------------------------------

The following paragraph in 3.1.2.1

| The AMI_Init function may return a modified impulse response by modifying
| the first column of impulse_matrix. If the impulse response is modified,
| the new impulse response is expected to represent the concatenation of the
| model block with the blocks represented by the input impulse response.

shall be modified as follows:

| The AMI_Init function may return a modified impulse response by modifying
| the first column of impulse_matrix. If the impulse response is modified,
| the new impulse response is expected to represent the filtered response. The
number
| of items in the matrix should remain unchanged.

------------------------------------------------------------------------------

The following paragraph in 3.1.2.6:

| Memory for AMI_parameters_in is allocated and de-allocate by the EDA.  The
| memory pointed to by AMI_parameters_out is allocated and by the model.

shall be modified as follows:

| Memory for AMI_parameters_in is allocated and de-allocated by the EDA platform.
The
| memory pointed to by AMI_parameters_out is allocated and de-allocated by the
model.

------------------------------------------------------------------------------

The following paragraph in 3.2.2.1:

| A vector of a time domain waveform, sampled uniformly at an interval
| specified by the 'sample_interval' specified during the init call.  The
| wave is both input and output.  The EDA platform provides the wave.  The
| algorithmic model is expected to modify the waveform in place.

shall be modified as follows:

| A vector of a time domain waveform, sampled uniformly at an interval
| specified by the 'sample_interval' specified during the init call. The
| wave is both input and output. The EDA platform provides the wave.
| The algorithmic model is expected to modify the waveform in place by
| applying a filtering behavior, for example, an equalization function,
| being modeled in the AMI_Getwave call.

*****************************************************************************

ANALYSIS PATH/DATA THAT LED TO SPECIFICATION

Comparisons of simulation results from the reference toolkits revealed differences
that were due
```

to different assumptions about the input waveform presented at a TX AMI_Getwave call, and the
relationship of the outputs from AMI_Init and AMI_Getwave calls.  Further discussion identified that
two different styles of modeling were possible and should be supported.  In the default case, the
AMI_Init and AMI_Getwave calls represent filtering performed by sequential stages of a device, and
the results should therefore be chained together.  In the second case, the AMI_Init and AMI_Getwave
calls each represent the overall device.  For example, the AMI_Init call could provide an LTI model
for the device while the AMI_Getwave call provides a time-varying model.  In this case, results from
the AMI_Init and AMI_Getwave calls should be treated as independent.

*****************************************************************************

ANY OTHER BACKGROUND INFORMATION:

The thoughts captured in this BIRD were discussed at the April 1, 2008 meeting of the
IBIS-ATM Working Group.  A slide presentation of this material is available at:
http://tinyurl.com/28ouvx

*****************************************************************************

The thoughts captured in this BIRD were discussed at the April 1, 2008 meeting of the
IBIS-ATM Working Group. A slide presentation of this material is available at:
<http://tinyurl.com/28ouvx>