IBIS-AMI
Terminology Overview

Walter Katz, SiSoft
wkatz@sisoft.com
Mike Steinberger, SiSoft
msteinb@sisoft.com
Todd Westerhoff, SiSoft
twesterh@sisoft.com

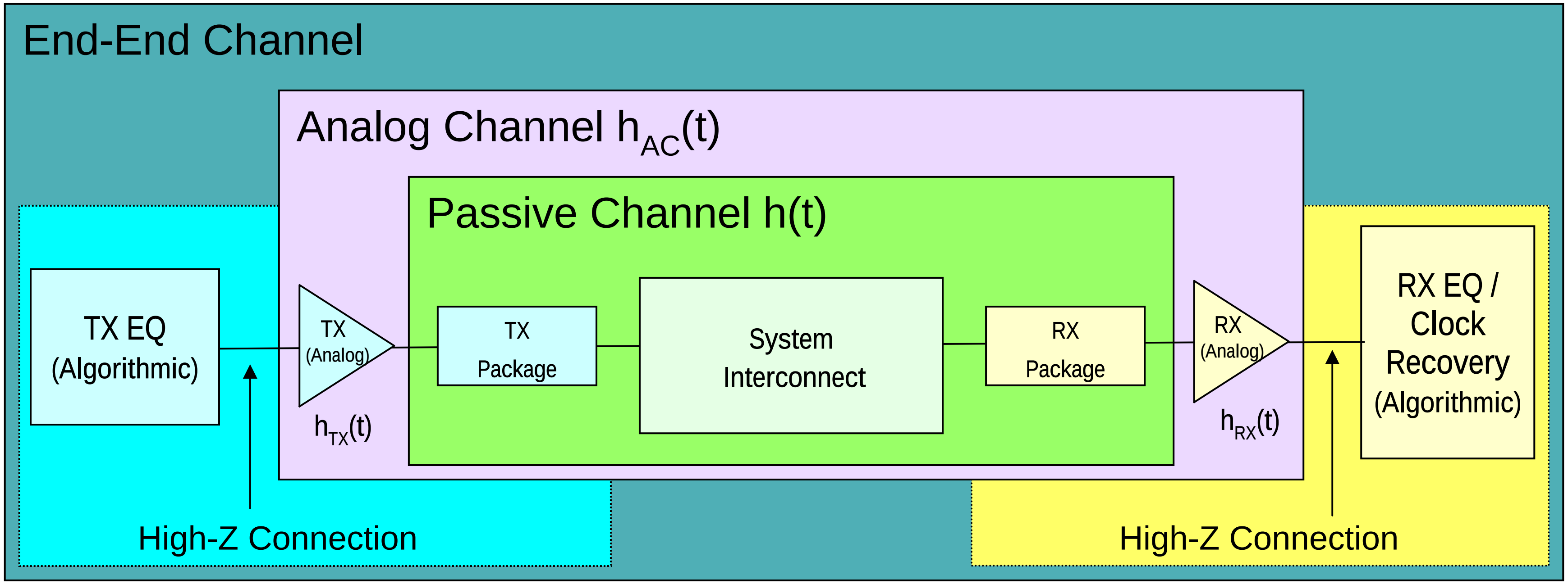
DAC 2009 IBIS Summit
San Francisco, CA
July 28, 2009

Signal Integrity Software, Inc.

IBIS-AMI Fundamental Assumptions

- Millions of bits of behavior are needed to adequately characterize serial links
- SERDES transmitters / receivers can be modeled as a combination of analog & algorithmic elements
  - TX output driver and RX termination network can be considered linear and time-invariant (LTI)
  - TX/RX equalization and clock recovery behavior can be modeled at the algorithmic level
- Serial channels can be characterized using S Parameter data and/or other passive interconnect models

IBIS-AMI Serial Channel

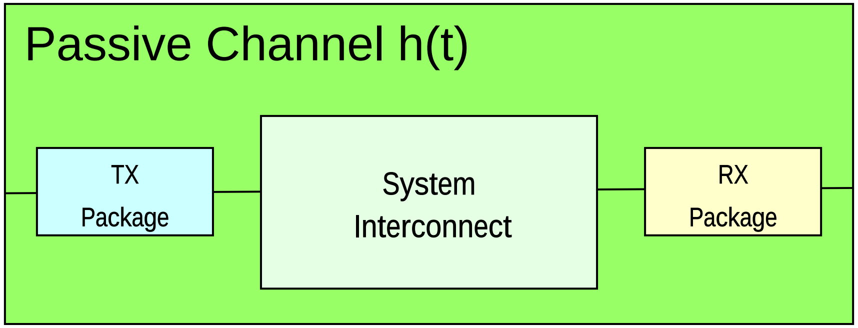


Passive Channel includes interconnect only

Analog Channel includes TX driver & RX termination network

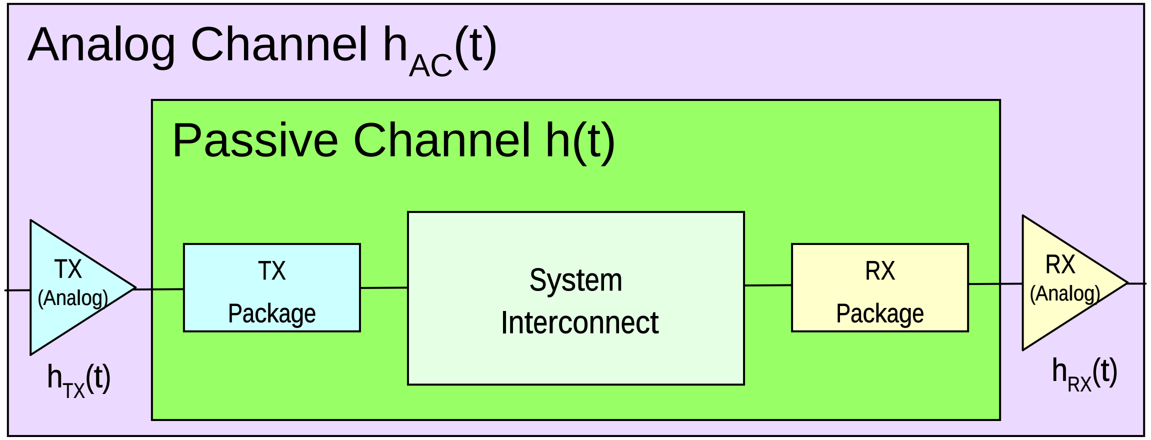
End-End Channel includes everything

Passive Channel – h(t)



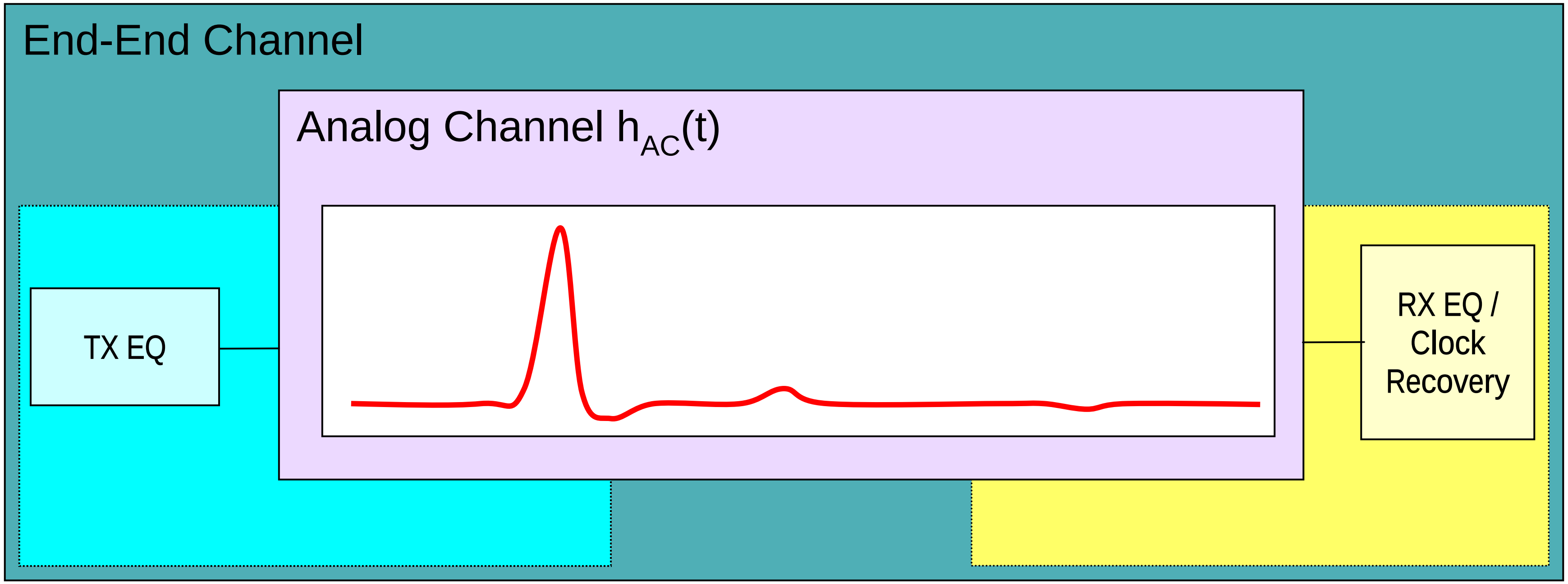
- Pad-pad passive interconnect (including TX/RX packages)
- Can be modeled using any method the simulator supports

Analog Channel – $h_{AC}(t)$



- IBIS-AMI Analog models for TX driver & RX termination network
- Network Characterization uses Analog models & circuit analysis techniques to determine impulse response
- Analysis can be performed using any method the simulator supports

IBIS-AMI End-End Channel



End-End Channel

Analog Channel $h_{AC}(t)$

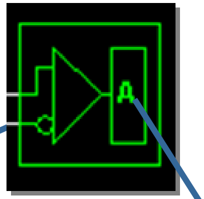
TX EQ

RX EQ / Clock Recovery

- Analog network is modeled as impulse response
- Channel Simulation uses IBIS-AMI Algorithmic models
- IBIS-AMI Reference Flow (IBIS 5.0 / Section 10 / Item 2.3) defines simulation results

IBIS-AMI Models

An IBIS-AMI model has two parts:



Analog Model

- Models unequalized analog device behavior
- Included in IBIS component model (.IBS) files
- Used to characterize analog network and derive impulse response
- Analog [Model] points to AMI [Algorithmic model]

Algorithmic Model

- Models equalization and clock recovery behavior
- Supplied as executable code
- Models can operate at two different levels:
  - INIT: impulse response processing
  - GETWAVE: time-domain waveform processing

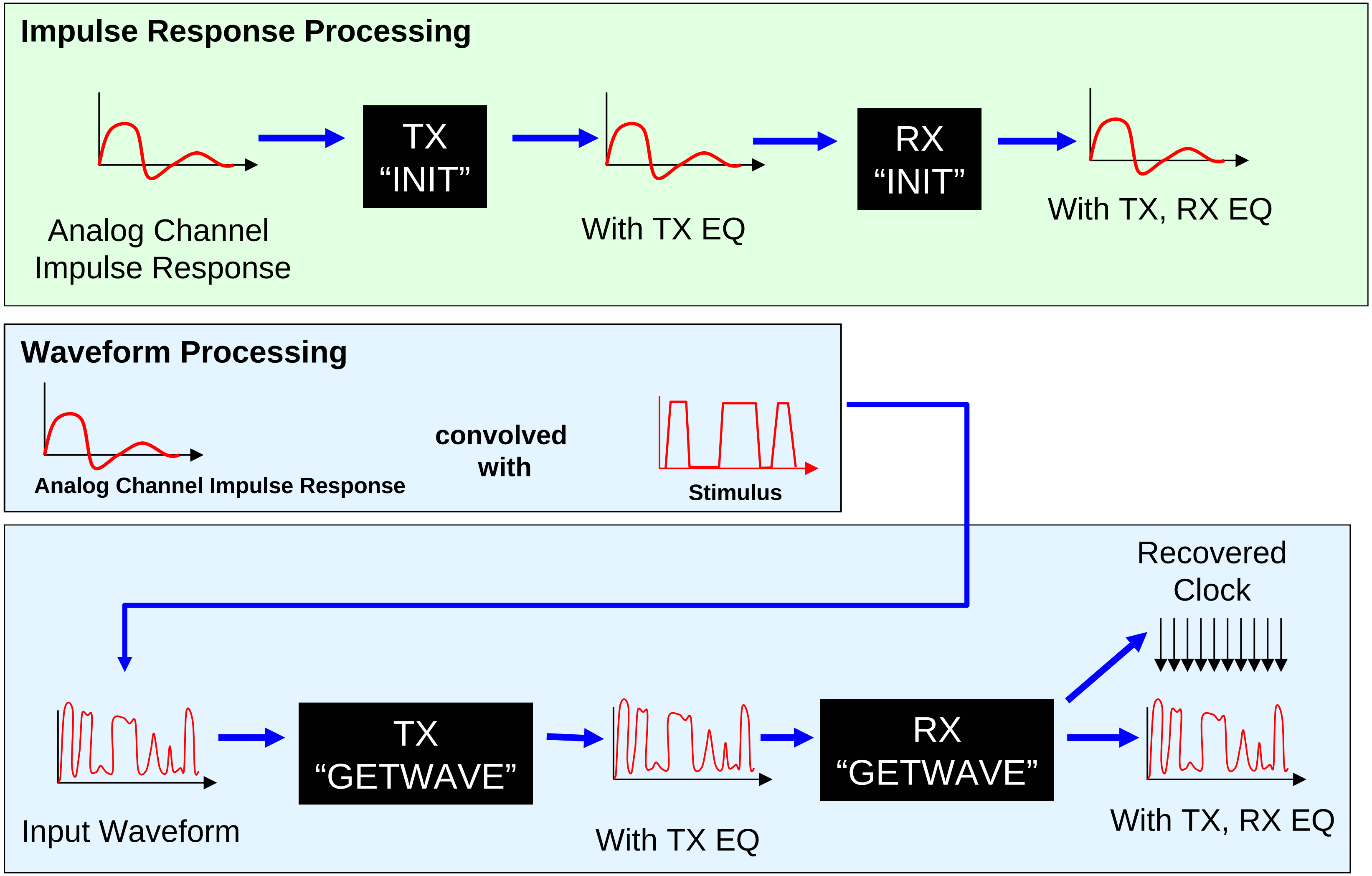
IBIS-AMI Algorithmic Models

- Provided as binary code
  - Fast, efficient execution
  - Protects vendor IP
  - Extensible modeling capability
  - Allows models to be developed in multiple languages

- Standardized execution interface
  - Defines .dll loading mechanism & call signature
  - Defines data input/output formats

- Standardized control (.AMI) file
  - Supplied as plain text file
  - “Reserved parameters” section - model capabilities
  - Model-specific parameters let users control the model

IBIS-AMI Algorithmic Models



Impulse Response Processing

Analog Channel
Impulse Response

TX
“INIT”

With TX EQ

RX
“INIT”

With TX, RX EQ

Waveform Processing

Analog Channel Impulse Response

convolved with

Stimulus

Input Waveform

TX
“GETWAVE”

With TX EQ

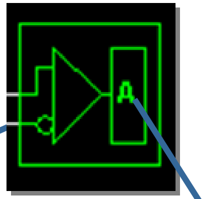
RX
“GETWAVE”

Recovered
Clock

With TX, RX EQ

IBIS-AMI Simulation

IBIS-AMI analysis has two stages:



Network Characterization

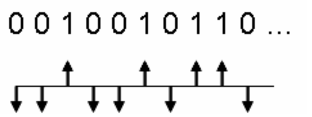
- Uses the passive network and IBIS analog models as input
- Performs either time-domain or frequency-domain circuit analysis
- Derives the impulse response for the analog network
- All analog effects considered are included in the impulse response

Channel Analysis

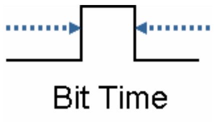
- Uses impulse response to represent Analog Channel
- Links IBIS-AMI Algorithmic models into simulator
- Users may vary model-specific settings
- Statistical and/or Time-Domain analysis depending on simulator and model capabilities

IBIS-AMI Terminology

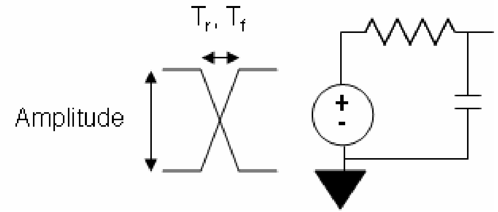
- Bit stream b(t)
  - Sum of delta functions



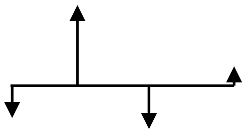
- Data symbol p(t)
  - Single bit width pulse



- TX analog characteristic $h_{TX}(t)$ *
  - Rise/fall time
  - Voltage swing
  - Drive impedance
  - Capacitance



- TX “Init” equalization $h_{TEI}(t)$ *
  - Sum of weighted delta functions
    - Coefficients & delays



- TX “Getwave” equalization $g_{TEG}()$ *
  - Not considered LTI
  - Waveform in, waveform out

* Similar terms exist for RX

LTI Channel Math

- Channel impulse response
  $h_{AC}(t) = h_{TX}(t) \otimes h(t) \otimes h_{RX}(t)$

- Channel pulse response
  $= h_{TX}(t) \otimes h(t) \otimes h_{RX}(t) \otimes p(t)$

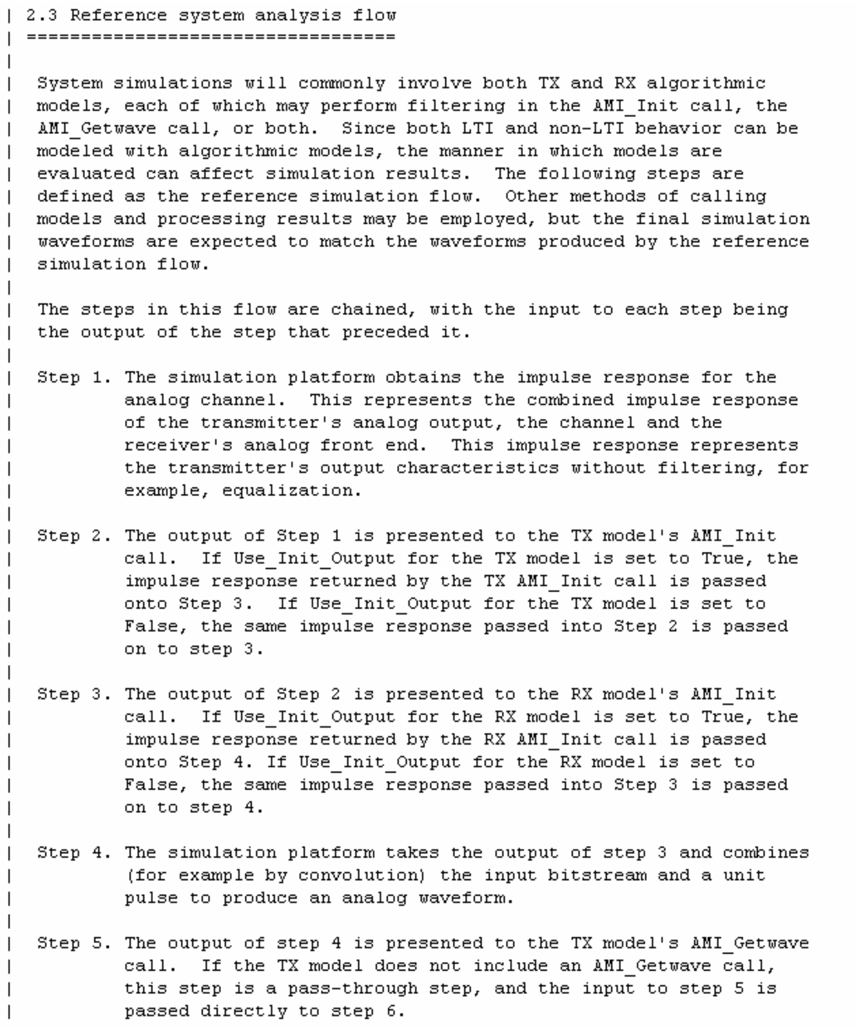
- Unequalized waveform @ RX pad
  $= h_{TX}(t) \otimes h(t) \otimes h_{RX}(t) \otimes p(t) \otimes b(t)$

- Channel pulse response with TX & RX “Init” EQ
  $= h_{TX}(t) \otimes h(t) \otimes h_{RX}(t) \otimes p(t) \otimes h_{TEI}(t) \otimes h_{REI}(t)$

- Waveform @ RX pad with TX “Init” EQ
  $= h_{TX}(t) \otimes h(t) \otimes h_{RX}(t) \otimes p(t) \otimes b(t) \otimes h_{TEI}(t)$

- Waveform @ RX sampler with TX & RX “Init” EQ
  $= h_{TX}(t) \otimes h(t) \otimes h_{RX}(t) \otimes p(t) \otimes b(t) \otimes h_{TEI}(t) \otimes h_{REI}(t)$

Reference Flow



```
| 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.
```

- IBIS 5.0 Spec, Section 10, Item 2.3
- Lists model evaluation order to define reference simulation waveforms
- Can be used with IBIS-AMI toolkit to produce reference waveforms
- Other evaluation techniques may be used as long as the resulting waveforms are the same

Use_Init_Output

- Controls linkage between Init() and Getwave() calls when both entry points are present
- 4 possible cases:

Case	TX Use_Init_Output	RX Use_Init_Output
1	False	False
2	False	True
3	True	False
4 (Default)	True	True

- The following slides consider cases where TX and RX models implement both AMI_Init and AMI_Getwave – other cases are simpler

Case 1: TX = False, RX = False

- Impulse response input to TX AMI_Init
  - $h_{AC}(t)$
- Impulse response input to RX AMI_Init
  - $h_{AC}(t)$
- Impulse response after RX AMI_Init
  - $h_{AC}(t)$
- Waveform input to TX AMI_Getwave
  - $p(t) \otimes b(t) \otimes h_{AC}(t)$
- Waveform input to RX AMI_Getwave
  - $g_{TEG}(p(t) \otimes b(t) \otimes h_{AC}(t))$
- Waveform output from RX AMI_Getwave
  - $g_{REG}(g_{TEG}(p(t) \otimes b(t) \otimes h_{AC}(t)))$

Case 2: TX = False, RX = True

- Impulse response input to TX AMI_Init
  - $h_{AC}(t)$
- Impulse response input to RX AMI_Init
  - $h_{AC}(t)$
- Impulse response after RX AMI_Init
  - $h_{AC}(t) \otimes h_{REI}(t)$
- Waveform input to TX AMI_Getwave
  - $p(t) \otimes b(t) \otimes h_{AC}(t) \otimes h_{REI}(t)$
- Waveform input to RX AMI_Getwave
  - $g_{TEG}(\ p(t) \otimes b(t) \otimes h_{AC}(t) \otimes h_{REI}(t)\ )$
- Waveform output from RX AMI_Getwave
  - $g_{REG}(\ g_{TEG}(\ p(t) \otimes b(t) \otimes h_{AC}(t) \otimes h_{REI}(t)\ )\ )$

Case 3: TX = True, RX = False

- Impulse response input to TX AMI_Init
  - $h_{AC}(t)$
- Impulse response input to RX AMI_Init
  - $h_{TEI}(t) \otimes h_{AC}(t)$
- Impulse response after RX AMI_Init
  - $h_{TEI}(t) \otimes h_{AC}(t)$
- Waveform input to TX AMI_Getwave
  - $p(t) \otimes b(t) \otimes h_{AC}(t) \otimes h_{TEI}(t)$
- Waveform input to RX AMI_Getwave
  - $g_{TEG}(\ p(t) \otimes b(t) \otimes h_{AC}(t) \otimes h_{TEI}(t)\ )$
- Waveform output from RX AMI_Getwave
  - $g_{REG}(\ g_{TEG}(\ p(t) \otimes b(t) \otimes h_{AC}(t) \otimes h_{TEI}(t)\ )\ )$

Case 4: TX = True, RX = True

- Impulse response input to TX AMI_Init
  - $h_{AC}(t)$
- Impulse response input to RX AMI_Init
  - $h_{TEI}(t) \otimes h_{AC}(t)$
- Impulse response after RX AMI_Init
  - $h_{TEI}(t) \otimes h_{AC}(t) \otimes h_{REI}(t)$
- Waveform input to TX AMI_Getwave
  - $p(t) \otimes b(t) \otimes h_{AC}(t) \otimes h_{TEI}(t) \otimes h_{REI}(t)$
- Waveform input to RX AMI_Getwave
  - $g_{TEG}(p(t) \otimes b(t) \otimes h_{AC}(t) \otimes h_{TEI}(t) \otimes h_{REI}(t))$
- Waveform output from RX AMI_Getwave
  - $g_{REG}(g_{TEG}(p(t) \otimes b(t) \otimes h_{AC}(t) \otimes h_{TEI}(t) \otimes h_{REI}(t)))$

Observations

- In all cases, the best-case (least distorted) input to TX AMI_GetWave is $p(t) \otimes b(t) \otimes h_{AC}(t)$
  - Input bit stream and bit time boundaries are not recoverable, makes modeling Dj and DCD nearly (if not completely) impossible
- In cases 2 and 4, the input to TX AMI_GetWave includes the RX AMI_Init equalization $h_{REI}(t)$
  - How could the writer of TX AMI_GetWave be expected to write code that compensates for an unknown external equalization function?
- The current “analog waveform” input to TX AMI_Getwave doesn’t really work
  - SiSoft will address this is an upcoming BIRD

Summary

- Reviewed IBIS-AMI fundamental assumptions
- Introduced channel and model terminology
- Reviewed LTI channel math and IBIS-AMI reference simulation results
- Identified issue for future resolution
- Comments welcome and appreciated!