SerDes Modeling: IBIS-AMI & Model Validation

July 2007

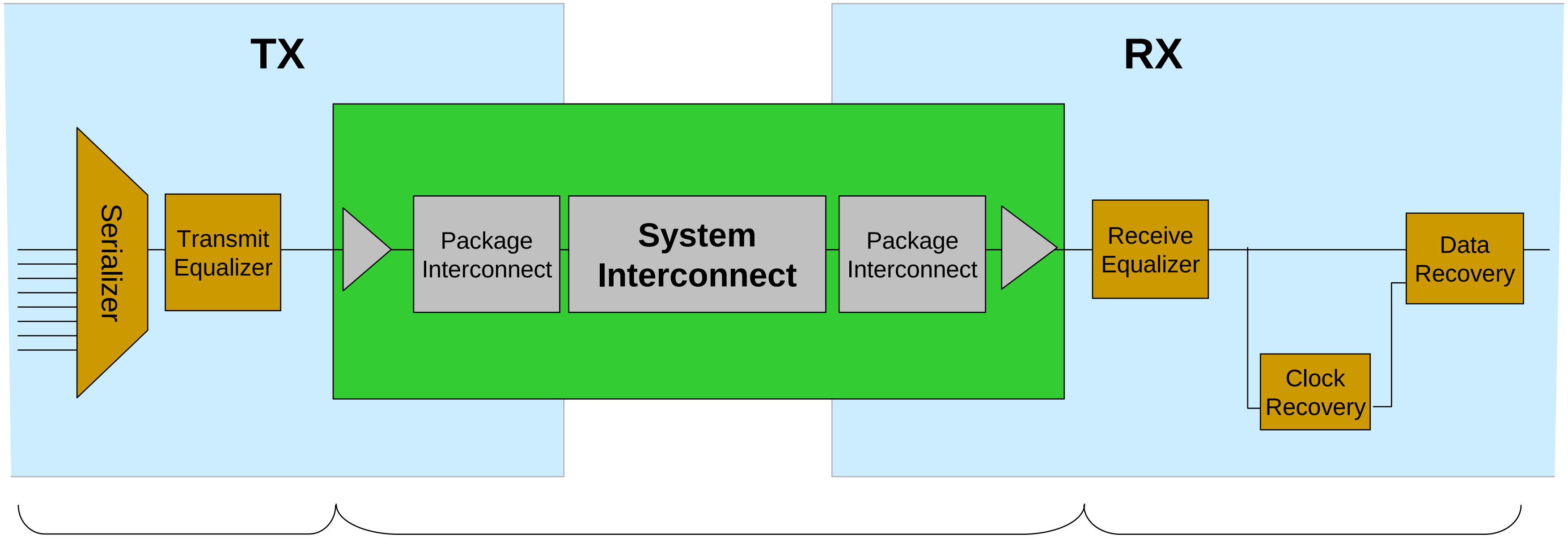
SiSoft™

Signal Integrity Software, Inc.

IBIS-AMI Effort

- Goal: SerDes Rx/TX model interoperability
  - Multiple EDA environments
  - Multiple SerDes vendor models
  - Protect SerDes vendor IP
- IBIS-AMI committee participation
  - EDA: SiSoft, Cadence, Mentor, Agilent
  - Semiconductor: IBM, TI, Intel, Micron, Xilinx, ST-Micro
  - System: Cisco
- Two part modeling standard
  - Electrical model: TX / RX analog characteristics
  - Algorithmic model: equalization, clock recovery, device optimization algorithms

Serial Link Analysis



TX EQ
LTI or non-LTI

- TX Equalization
- TX Optimization

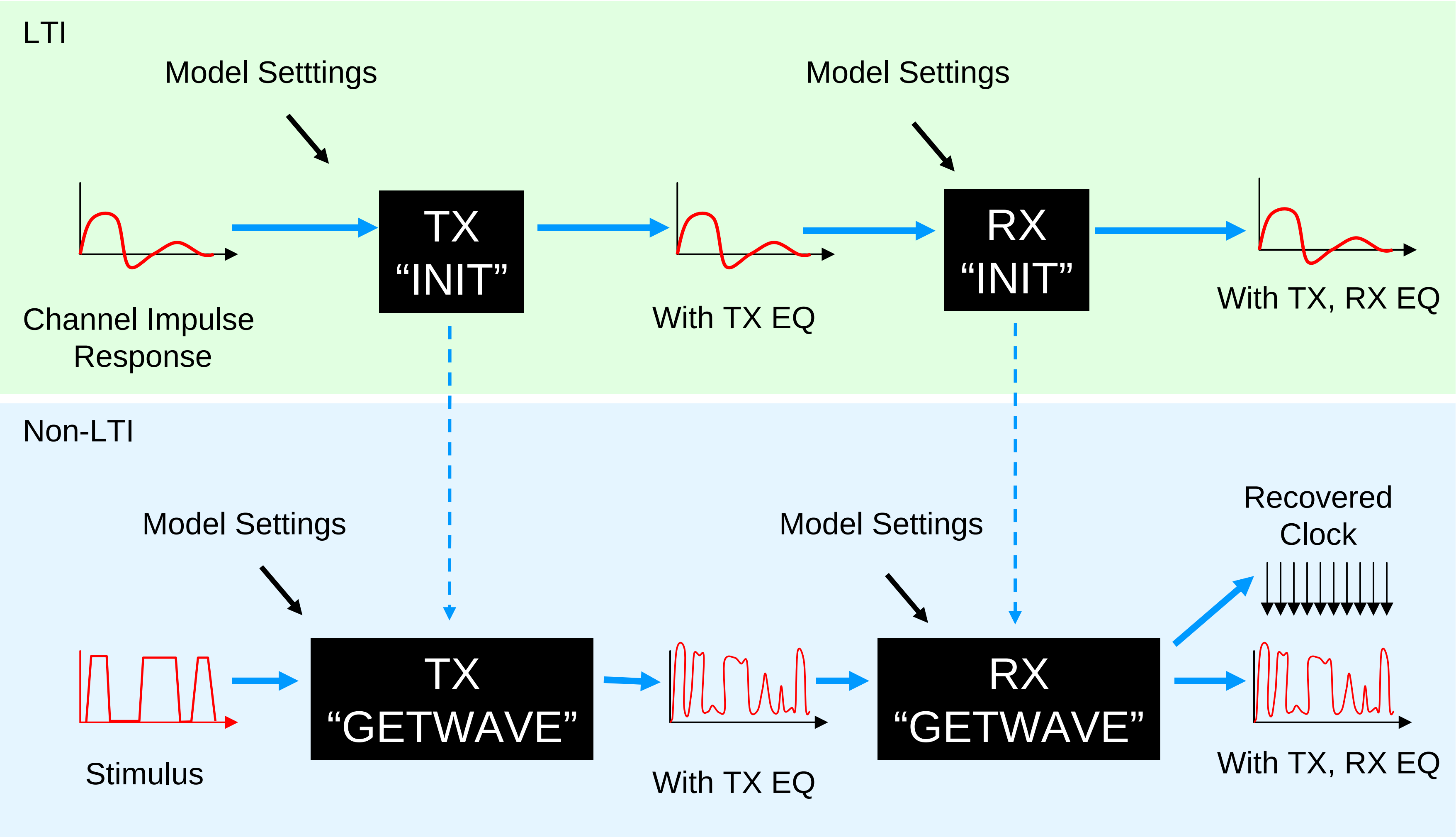
Channel & Analog I/O
Linear, Time-Invariant

- Channel Characterization (Impulse response)

RX EQ, CDR
LTI or non-LTI

- RX Equalization
- RX Clock Recovery
- RX Optimization

IBIS-AMI Algorithmic Models



LTI

Model Setttings

Model Settings

Channel Impulse Response

TX
“INIT”

With TX EQ

RX
“INIT”

With TX, RX EQ

Non-LTI

Model Settings

Model Settings

Recovered Clock

Stimulus

TX
“GETWAVE”

With TX EQ

RX
“GETWAVE”

With TX, RX EQ

IBIS-AMI Status

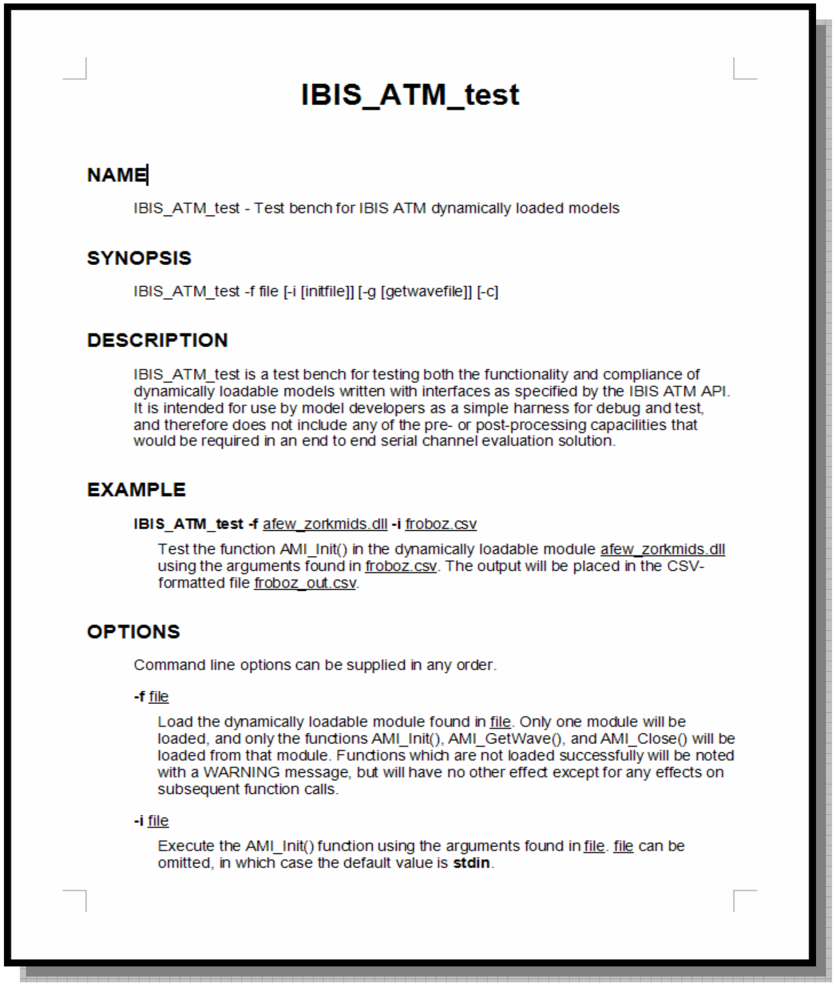
- Subcommittee work, presentations & BIRD available on-line:
  - http://www.vhdl.org/pub/ibis/macromodel_wip/
- First draft of BIRD approved by IBIS-AMI subcommittee for model & EDA platform development
- Sample models for public reference - 7/17/07

Challenges

- IBISCHK cannot check compiled models
  - Similar problem to AMS model calls
- API interface is complex by IBIS standards
- Several possible sources of platform/model incompatibility
  - Incorrect EDA tool implementation
  - Incorrect model implementation
  - Incompatible run-time libraries

- A “reference standard” for IBIS-AMI is needed
  - Reference platform implementation
  - Reference model implementation

IBIS_AMI_Test



IBIS_ATM_test

NAME

IBIS_ATM_test - Test bench for IBIS ATM dynamically loaded models

SYNOPSIS

```
IBIS_ATM_test -f file [-i [initfile]] [-g [getwavefile]] [-c]
```

DESCRIPTION

IBIS_ATM_test is a test bench for testing both the functionality and compliance of dynamically loadable models written with interfaces as specified by the IBIS ATM API. It is intended for use by model developers as a simple harness for debug and test, and therefore does not include any of the pre- or post-processing capacilities that would be required in an end to end serial channel evaluation solution.

EXAMPLE

IBIS_ATM_test -f afew_zorkmids.dll **-i** froboz.csv

Test the function AMI_Init() in the dynamically loadable module afew_zorkmids.dll using the arguments found in froboz.csv. The output will be placed in the CSV-formatted file froboz_out.csv.

OPTIONS

Command line options can be supplied in any order.

-f file

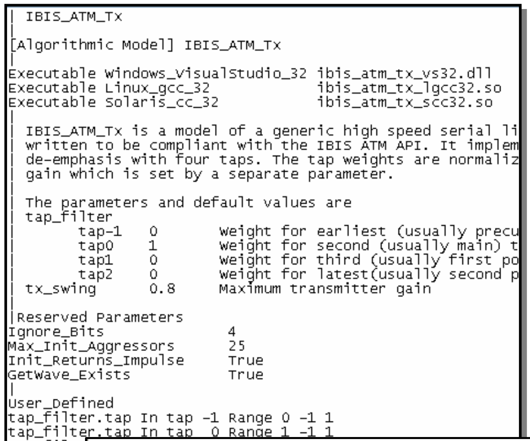
Load the dynamically loadable module found in file. Only one module will be loaded, and only the functions AMI_Init(), AMI_GetWave(), and AMI_Close() will be loaded from that module. Functions which are not loaded successfully will be noted with a WARNING message, but will have no other effect except for any effects on subsequent function calls.

-i file

Execute the AMI_Init() function using the arguments found in file. file can be omitted, in which case the default value is **stdin**.

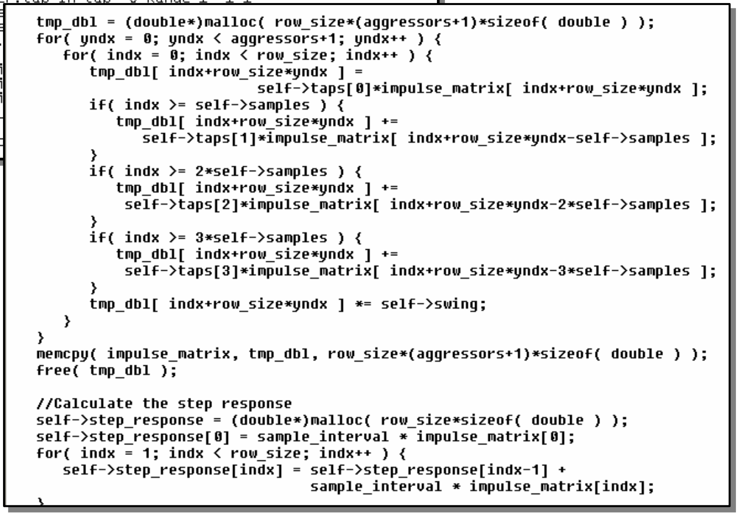
- Allows IBIS-AMI .dll models to be run as standalone “executables”
  - Facilitates model debug
  - Provides standard environment for testing model compliance
  - Can be supplied as part of IP vendor model “kit”
- Authored by SiSoft, source code to be turned over to IBIS Open Forum
  - Executable to be widely available

SiSoft IBIS_AMI TX Model



```
| IBIS_ATM_Tx
|
[Algorithmic Model] IBIS_ATM_Tx
|
Executable Windows_VisualStudio_32 ibis_atm_tx_vs32.dll
Executable Linux_gcc_32            ibis_atm_tx_lgcc32.so
Executable Solaris_cc_32           ibis_atm_tx_scc32.so
|
| IBIS_ATM_Tx is a model of a generic high speed serial li
| written to be compliant with the IBIS ATM API. It implem
| de-emphasis with four taps. The tap weights are normaliz
| gain which is set by a separate parameter.
|
| The parameters and default values are
| tap_filter
|       tap-1   0       Weight for earliest (usually precu
|       tap0    1       Weight for second (usually main) t
|       tap1    0       Weight for third (usually first po
|       tap2    0       Weight for latest(usually second p
| tx_swing      0.8     Maximum transmitter gain
|
|Reserved Parameters
Ignore_Bits                 4
Max_Init_Aggressors         25
Init_Returns_Impulse        True
GetWave_Exists              True
|
User_Defined
tap_filter.tap In tap -1 Range 0 -1 1
tap_filter.tap In tap  0 Range 1 -1 1
tap_filte
tap_filte
tx_swing.
|
Descripti
Descripti
Descripti
End_User_
|
[End Algo
```

IBIS Model

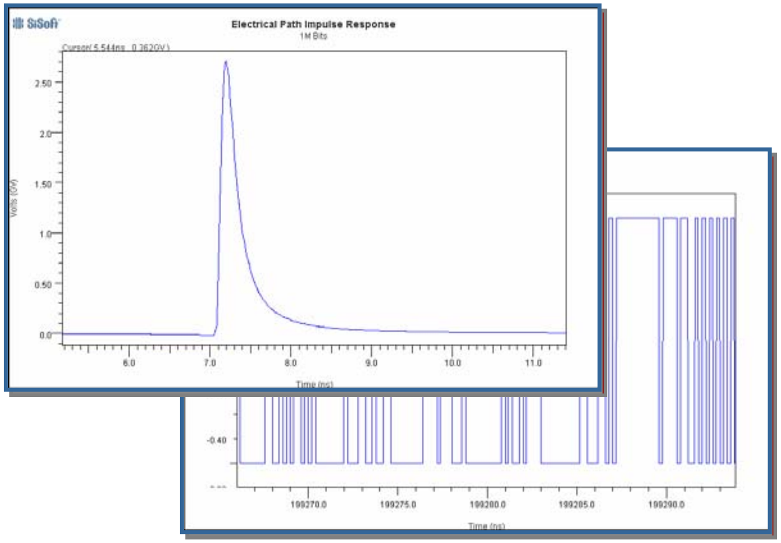


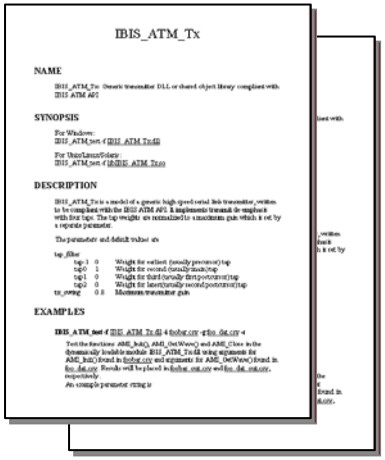
```
tmp_dbl = (double*)malloc( row_size*(aggressors+1)*sizeof( double ) );
for( yndx = 0; yndx < aggressors+1; yndx++ ) {
   for( indx = 0; indx < row_size; indx++ ) {
      tmp_dbl[ indx+row_size*yndx ] =
                           self->taps[0]*impulse_matrix[ indx+row_size*yndx ];
      if( indx >= self->samples ) {
         tmp_dbl[ indx+row_size*yndx ] +=
            self->taps[1]*impulse_matrix[ indx+row_size*yndx-self->samples ];
      }
      if( indx >= 2*self->samples ) {
         tmp_dbl[ indx+row_size*yndx ] +=
          self->taps[2]*impulse_matrix[ indx+row_size*yndx-2*self->samples ];
      }
      if( indx >= 3*self->samples ) {
         tmp_dbl[ indx+row_size*yndx ] +=
          self->taps[3]*impulse_matrix[ indx+row_size*yndx-3*self->samples ];
      }
      tmp_dbl[ indx+row_size*yndx ] *= self->swing;
   }
}
memcpy( impulse_matrix, tmp_dbl, row_size*(aggressors+1)*sizeof( double ) );
free( tmp_dbl );

//Calculate the step response
self->step_response = (double*)malloc( row_size*sizeof( double ) );
self->step_response[0] = sample_interval * impulse_matrix[0];
for( indx = 1; indx < row_size; indx++ ) {
   self->step_response[indx] = self->step_response[indx-1] +
                               sample_interval * impulse_matrix[indx];
}
```

API Model Code

- Reference IBIS file
- Reference API model
  - Impulse response and waveform processing
  - 4 tap equalizer
    - Pre-cursor tap
    - Cursor tap
    - 2 post-cursor taps
    - Model normalizes tap sum
  - Scalable transmit swing
  - Executable and source code to be widely available

Supporting Data





IBIS_ATM_Tx

NAME

IBIS_ATM_Tx: Generic transmitter DLL or shared object library compliant with IBIS ATM API

SYNOPSIS

For Windows:
IBIS_ATM_test -f IBIS_ATM_Tx.dll

For Unix/Linux/Solaris:
IBIS_ATM_test -f libIBIS_ATM_Tx.so

DESCRIPTION

IBIS_ATM_Tx is a model of a generic high-speed serial link transmitter, written to be compliant with the IBIS ATM API. It implements transmit de-emphasis with four taps. The tap weights are normalized to a maximum gain which is set by a separate parameter.

The parameters and default values are

```
tap_filter
        tap-1   0       Weight for earliest (usually precursor) tap
        tap0    1       Weight for second (usually main) tap
        tap1    0       Weight for third (usually first postcursor) tap
        tap2    0       Weight for latest (usually second postcursor) tap
tx_swing        0.8     Maximum transmitter gain
```

EXAMPLES

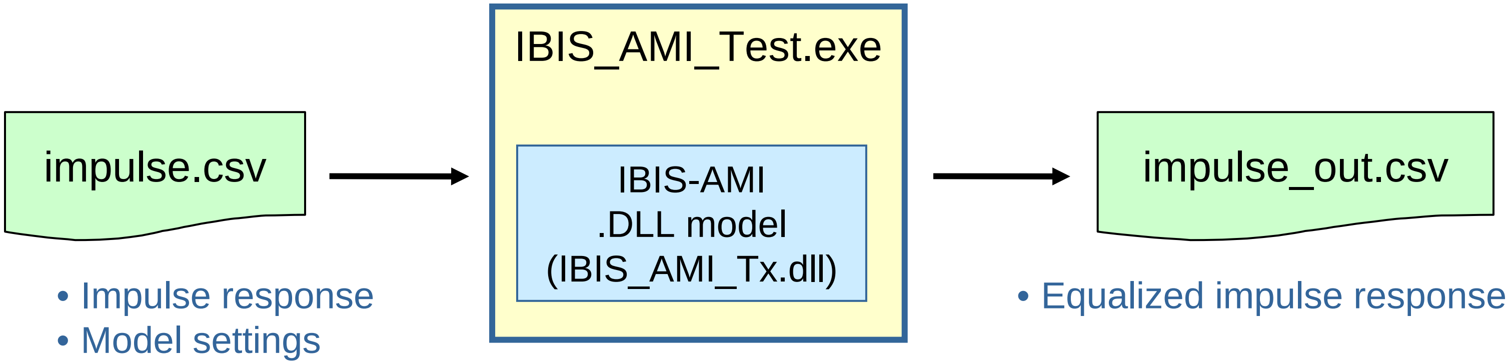
IBIS_ATM_test -f IBIS_ATM_Tx.dll -i foobar.csv -g foo_dat.csv -t

Test the functions AMI_Init(), AMI_GetWave() and AMI_Close in the dynamically loadable module IBIS_ATM_Tx.dll using arguments for AMI_Init() found in foobar.csv and arguments for AMI_GetWave() found in foo_dat.csv. Results will be placed in foobar_out.csv and foo_dat_out.csv, respectively.
An example parameter string is

- Sample impulse response
- Sample stimulus data
- Batch files
- Documentation

Impulse Response Processing

IBIS_AMI_test -f IBIS_AMI_Tx.dll **-i** impulse.csv



impulse.csv

IBIS_AMI_Test.exe

IBIS-AMI
.DLL model
(IBIS_AMI_Tx.dll)

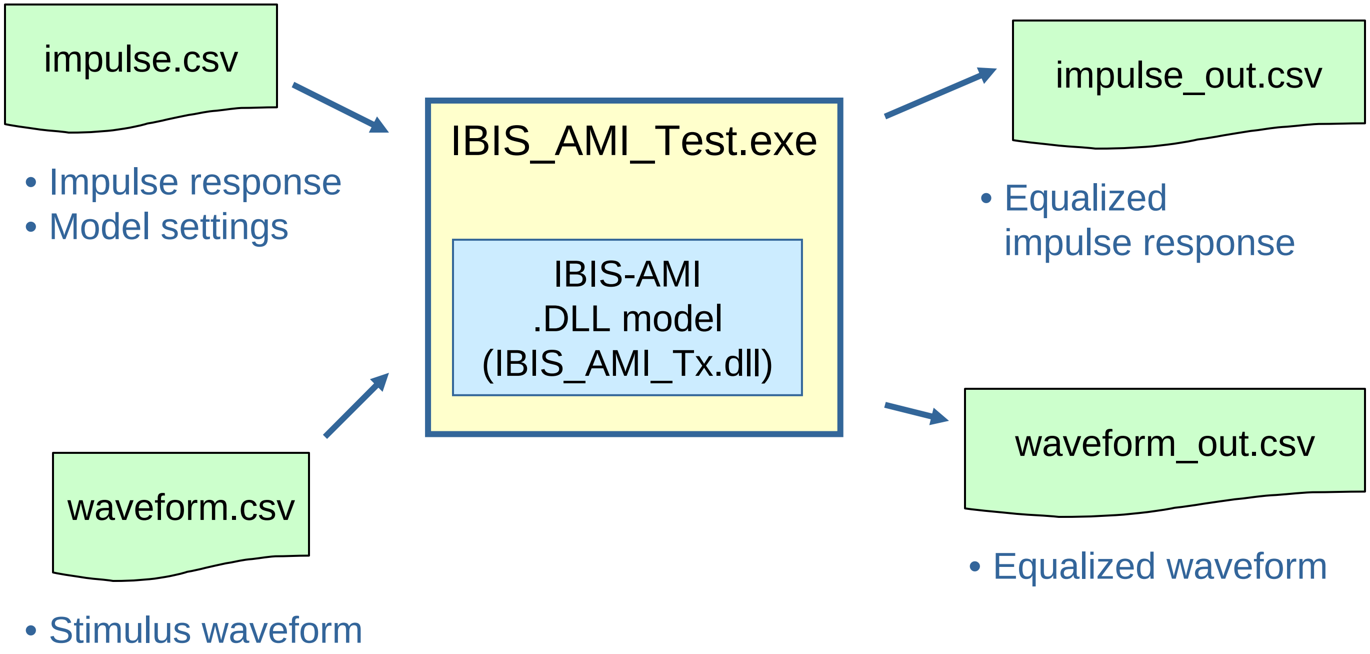
impulse_out.csv

- Impulse response
- Model settings

- Equalized impulse response

Waveform Processing

```
IBIS_AMI_test -f IBIS_AMI_Tx.dll -i tx_impulse.csv -g waveform.csv –c > waveform_out.csv
```



impulse.csv

- Impulse response
- Model settings

waveform.csv

- Stimulus waveform

IBIS_AMI_Test.exe

IBIS-AMI
.DLL model
(IBIS_AMI_Tx.dll)

impulse_out.csv

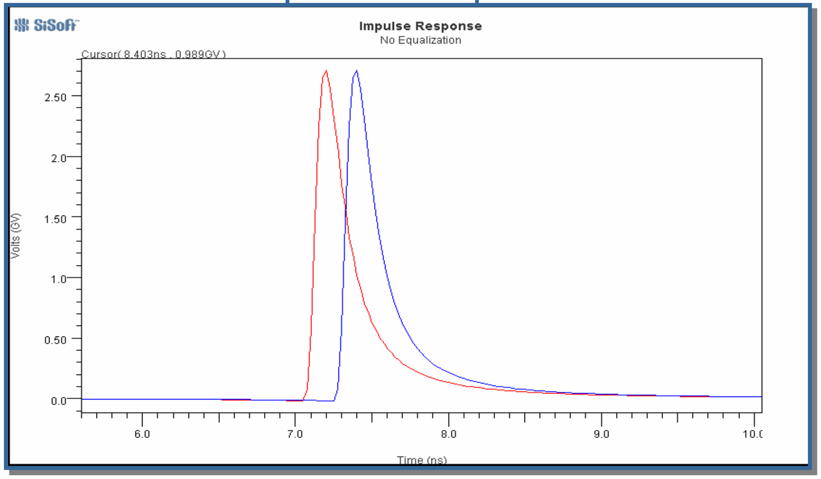
- Equalized impulse response

waveform_out.csv

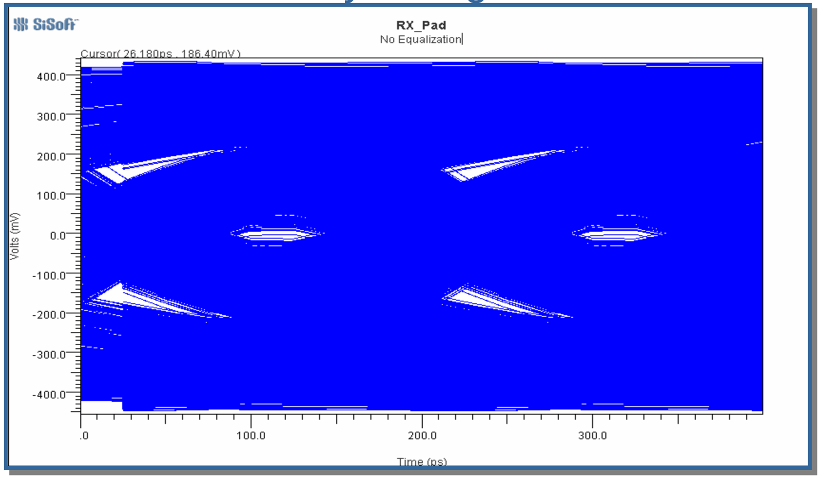
- Equalized waveform

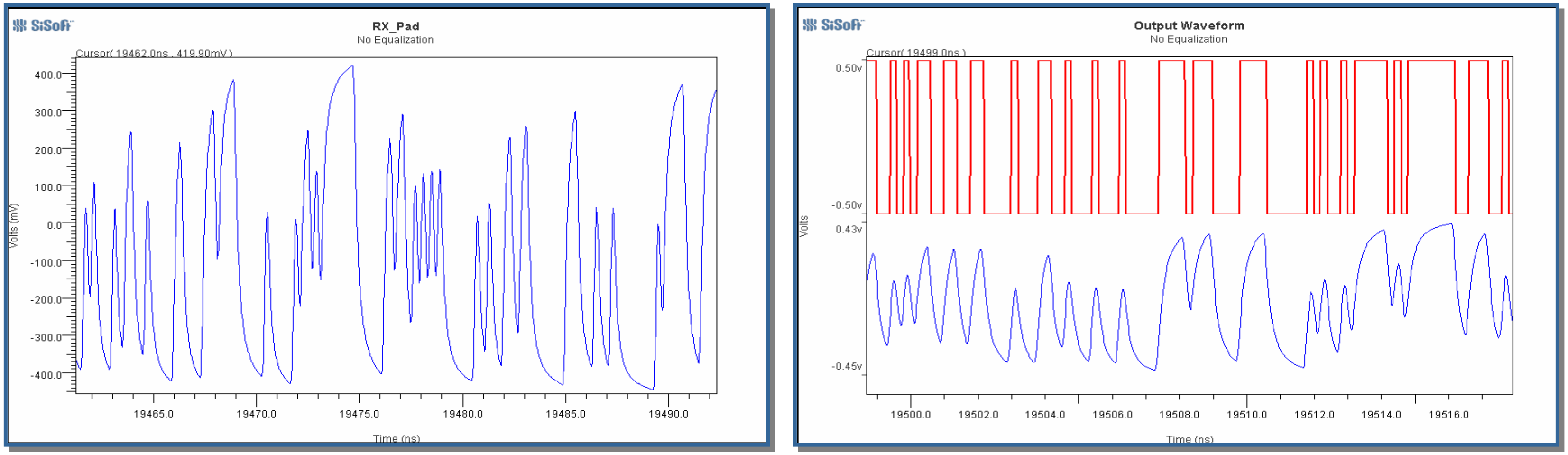
No TX EQ

Impulse Response



Eye Diagram

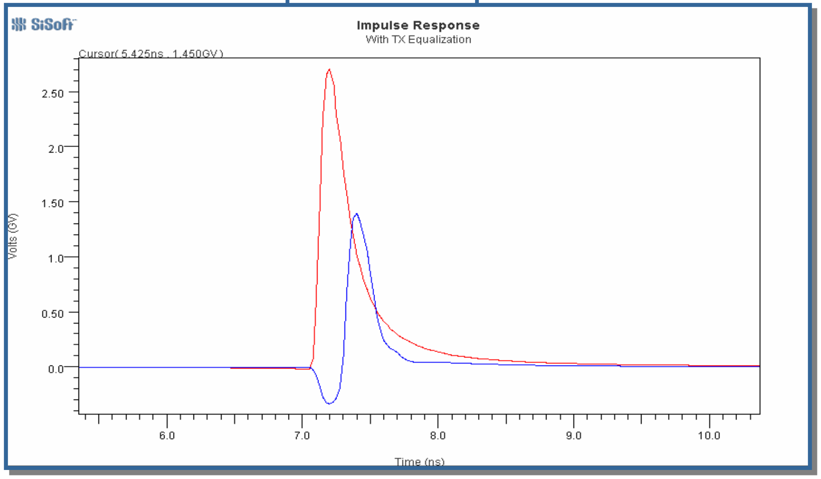




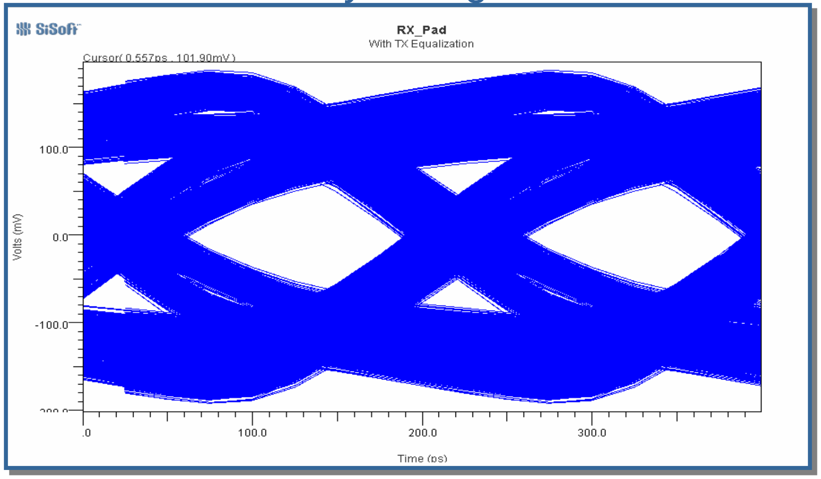
Signal @ Rx pad, Stimulus

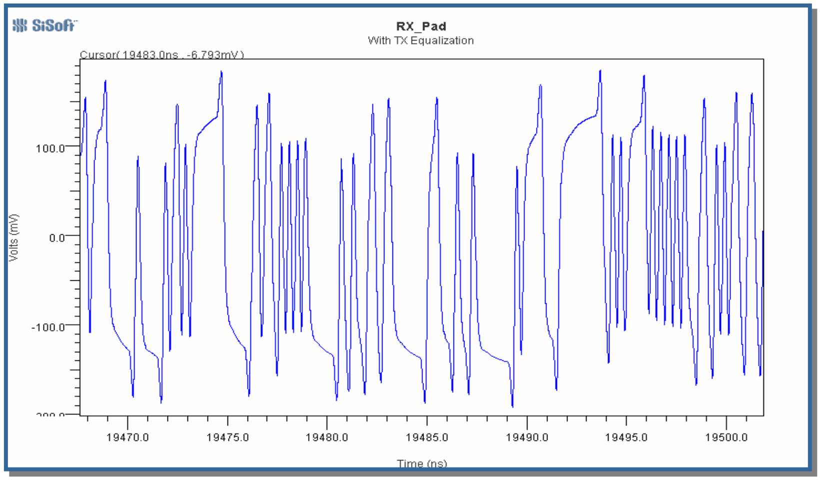
TX EQ: (-.15, .7,-.125,-.025)*0.8

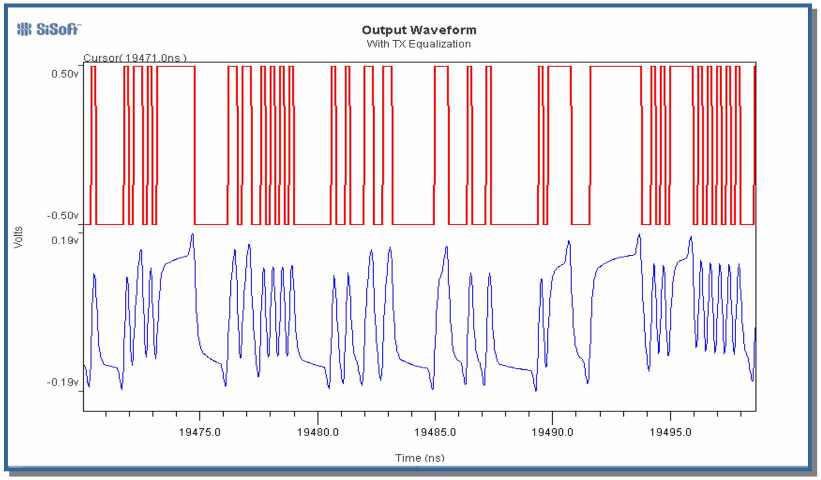
Impulse Response



Eye Diagram







Signal @ Rx pad, Stimulus

IBIS-AMI Evaluation Toolkit

- Goal: allow interested parties to evaluate & develop IBIS-AMI models
- Initially available on-request from SiSoft
  - Will reassess distribution model once support requirements are better understood
- Contents
  - IBIS_AMI_Test utility
  - Sample TX model and source code
  - Sample input data, scripts, documentation
- IBIS_AMI_Test source will be turned over to IBIS Open Forum (similar to IBISCHK)