SerDes Modeling:
IBIS-AMI Evaluation Toolkit

(Originally presented at the Sept 11th Summit in Beijing)

Presented by: Todd Westerhoff, SiSoft
twesterh@sisoft.com

IBIS Summit

Tokyo, Japan

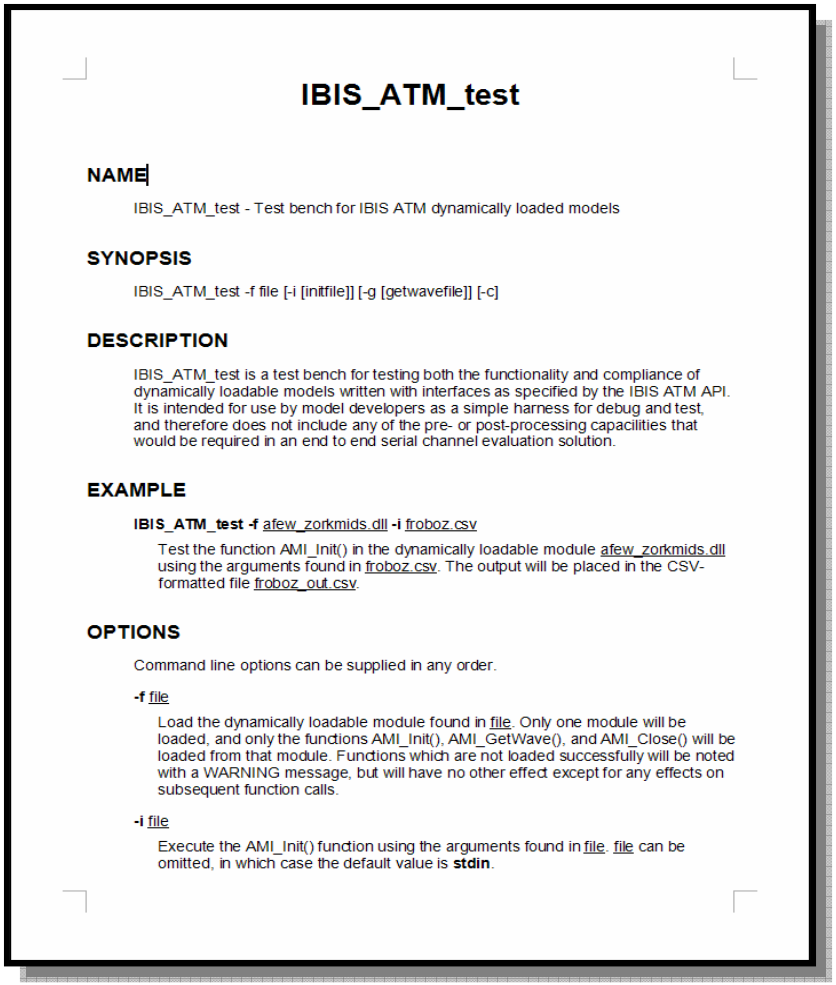
September 14, 2007

SiSoft
Signal Integrity Software, Inc.

Challenges

- IBISCHK cannot check compiled models
  - Similar problem to AMS model calls
- 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
- Authored by SiSoft, source code to be turned over to IBIS Open Forum
  - Executable will be publicly available

IBIS_AMI_Tx Model

AMI File

```
(IBIS_AMI_Tx
 (Dll
  (linux ibis_ami_tx_lnx.so)
  (solaris ibis_ami_tx_sun4.so)
  (windows ibis_ami_tx.dll)
)
 (Reserved_Parameters
  (Ignore_Bits (Type Integer) (Default 21))
  (Max_Init_Aggressors (Type Integer)(Default 25))
  (Init_Returns_Impulse (Type Bool)(Default True))
  (GetWave_Exists (Type Bool) (Default True))
 ) | End Reserved
(Model_specific
  (txtaps
    (tapid (Range -1 3) (type int) )
    (txtapcoeff
        (-1 (Range -0.1 0.1 0.1) (Type float) (Default 0))
        (0 (Range 1 1 0) (Type float) (Default 1))
        (1 (Lis
        (2 (Ran
        (Defaul
        (3 (Ran
        (Defaul
    )  | tapcoe
    )  | txtaps
  (tx_freq_offs
 ) | End User_D
)  | End IBIS_A
```

```
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];
}
```

Algorithmic Model Code

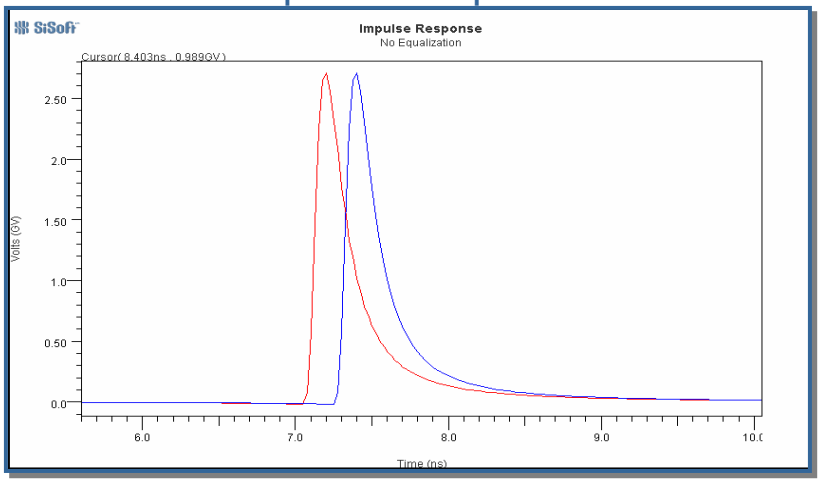
- Reference IBIS & AMI files
- Reference Algorithmic model
  - Impulse response and waveform processing
  - 4 tap equalizer
    - Pre-cursor tap
    - Cursor tap
    - 2 post-cursor taps
    - Model normalizes tap sum
  - Scalable transmit swing
  - Executable and source code publicly available

IBIS-AMI Evaluation Toolkit

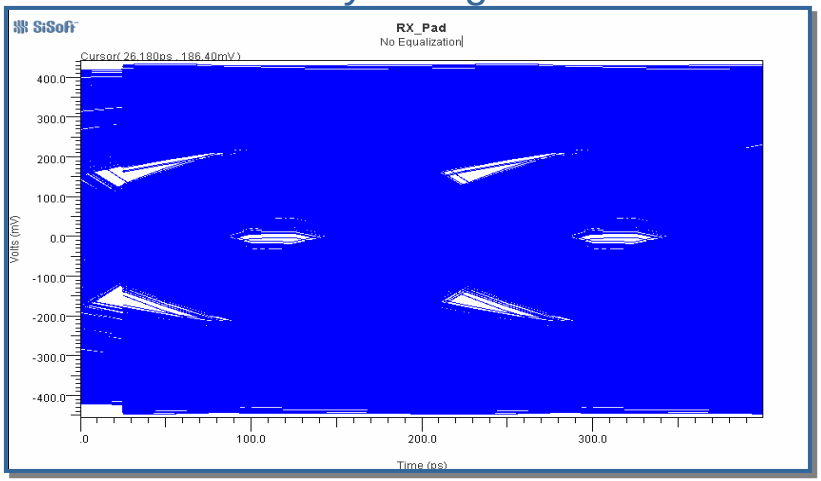
- Goal: allow interested parties to evaluate & develop IBIS-AMI models
- Available on-line from IBIS-ATM task group website and from www.sisoft.com
- Contents
  - IBIS_AMI_Test utility
  - Sample TX model and source code
  - Sample input data, scripts, documentation
- Email discussion group established: ibis-ami-toolkit@freelists.org

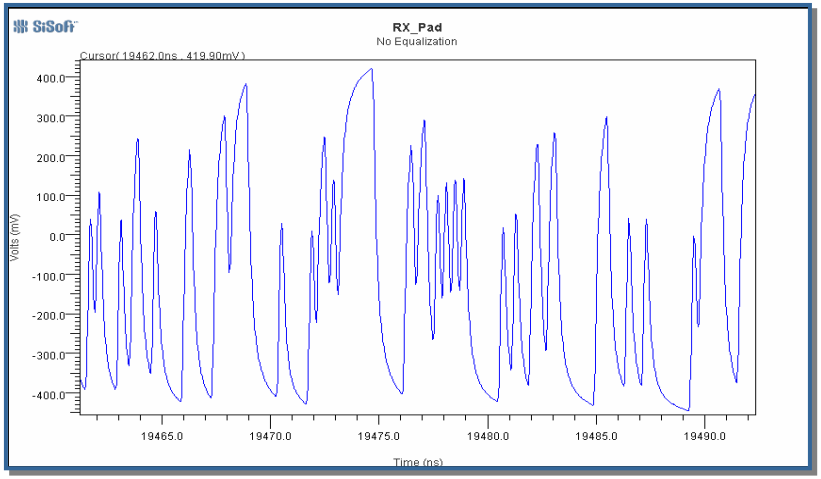
Sample Results: No TX EQ

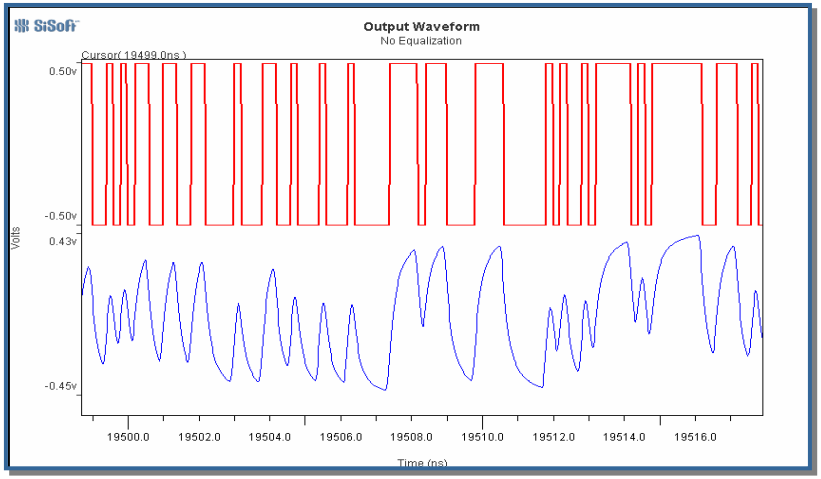
Impulse Response



Eye Diagram



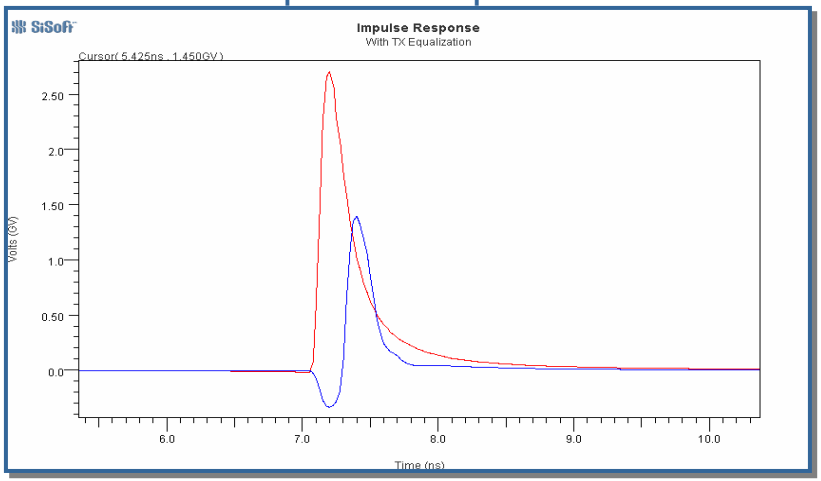




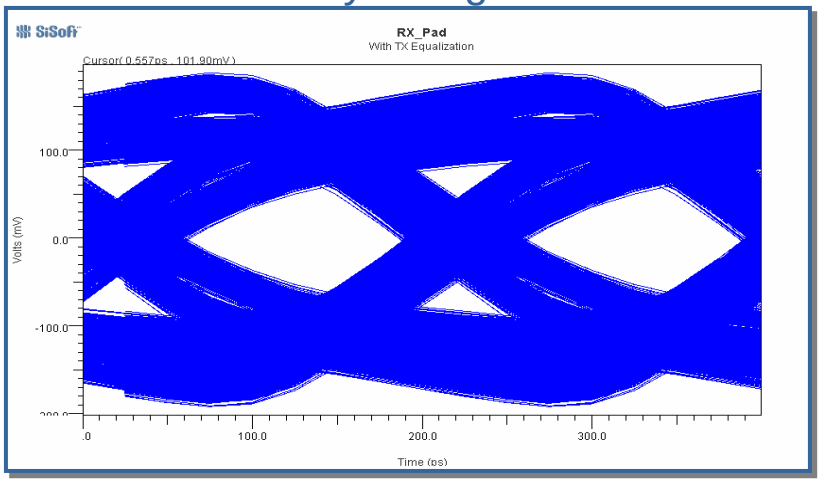
Signal @ Rx pad, Stimulus

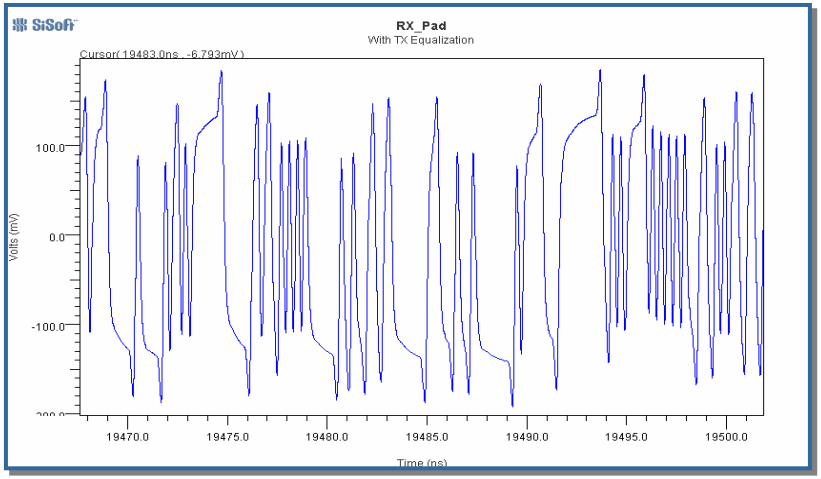
Sample Results:
TX EQ: (-.15, .7,-.125,-.025)*0.8

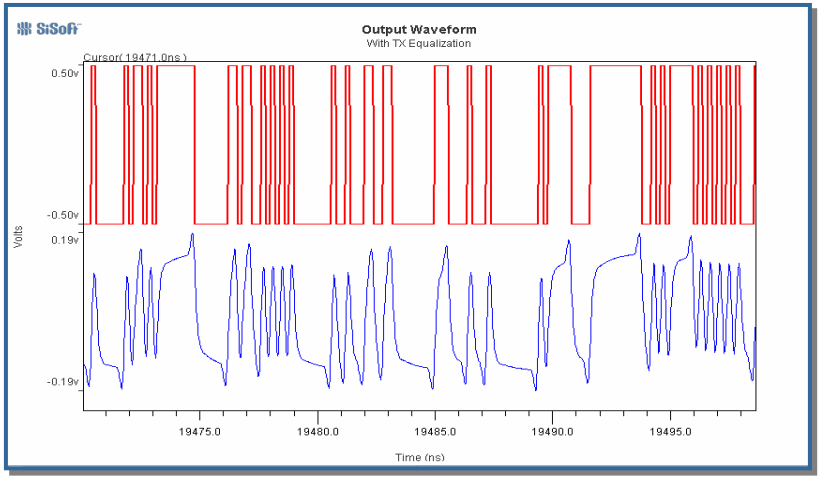
Impulse Response



Eye Diagram







Signal @ Rx pad, Stimulus

IBIS-AMI Toolkit

- Provides a “reference implementation” for testing EDA platforms and IBIS-AMI models
- Allows users to assess IBIS-AMI model functionality and performance
- Users can run analyses based on their own designs and channel impulse responses