IBIS 4.1 Macromodel Library for Simulator Independent Modeling

Arpad Muranyi, Mike LaBonte, Todd Westerhoff
Sam Chitwood, Ian Dodd, Barry Katz,
Scott McMorrow, Bob Ross, Ken Willis
Agenda

- History – How did we get here?
- Macromodeling concepts
- A call to action
- The working group
- Current building block library
- Current issues
- Next steps
History – How Did We Get Here?

• IBIS isn’t keeping up with the demands of new I/O technologies
  - SPICE use in SI analysis is increasing

• IBIS 4.1 allows AMS models to be called from IBIS, but adoption has been slow

• Macromodeling proposed at January 2005 summit by Donald Telian of Cadence as an alternative to full AMS implementations
  - Proposal was for Berkeley SPICE extensions
History – How Did We Get Here?

- Study group formed (Intel, Cisco, Teraspeed, SiSoft, Cadence) to explore macromodeling concept
- Macromodeling in AMS proposed as a bridge to full AMS models, without requiring IBIS specification changes that would be required for SPICE extensions
- Proof-of-concept models built and presented at DAC 2005 IBIS Summit
Mar 2005: First Attempts at Mixing Spice/AMS
The Macromodel Concept

- Create a library of AMS “elements” that can be instantiated and interconnected to create complex buffer models
- Model AMS elements after sources and elements found in popular SPICE tools
- Express the reference definitions of element behavior using an AMS language.
- Ensure elements can be implemented by substitution in SPICE engines without native AMS support
- Standardize AMS element library across all model providers
- IBIS files reference netlists constructed from reference “templates”, which instantiate the AMS elements
Macromodel Hierarchy

IBIS File

[External Circuit] or [External Model] call macro model templates

Macro model templates call elements from common AMS library

AMS Elements are written using the analog only features of the *-AMS languages, and can be substituted with native SPICE elements in SPICE tools if necessary
Macromodel example

Pre-emphasis buffer template

Template port

AMS Element Callouts

IN

TX+

TX-

main
delay
boost
scaler

main
delay
boost
scaler

Adder

Adder
AMS Element Mapping

Native AMS Simulator

IBIS_BUFFER_OUT ioB #( .ibis_file("mybuf.ibs"), .ibis_model("mybuff"))
(InB, En, RcvB, IoB, PCrefB, PUrefB, PDrefB, GCrefB);

SPICE Simulator

b_io PUrefB PDrefB IoB InB En PCrefB GCrefB
+ file='mybuf.ibs' model='mybuff' +power=on buffer=2
Macromodels in Verilog-A simulators

IBIS 4.1 Macromodel Library for Simulator-independent models

Part.ibs

IC website

IBIS website

EXTRACT/ MERGE DATA

Netlists.va
Customized buffer modules

Bld_blks.va

Verilog-A simulator
Macromodels in VHDL-AMS simulators

Part.ibs → EXTRACT NETLIST, TRANSLATE → Netlists.vhd

IC website

IBIS website

Bld_blks.vhd → VHDL-AMS simulator
June 14, 2005: A Clear Strategy

AMS models that can be translated into SPICE implementations
A Call to Action

- IBIS/AMS macromodeling proposal presented at IBIS Summit, June 2005
- Letters sent out to major simulator vendors inviting them to participate in library definition
- Working group formed
  - Arpad still does most of the coding and testing ...
The Working Group

- Intel – Arpad Muranyi
- Cadence Design Systems – Ken Willis
- Cisco Systems – Mike LaBonte, Todd Westerhoff
- Mentor Graphics – Ian Dodd
- Sigrity – Sam Chitwood
- SiSoft – Barry Katz
- Teraspeed - Scott McMorrow, Bob Ross
## Current Library - Basic Elements

<table>
<thead>
<tr>
<th></th>
<th>Fixed</th>
<th>Voltage Controlled</th>
<th>Current Controlled</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>IBIS_R</td>
<td>IBIS_VCR</td>
<td>IBIS_CCR</td>
</tr>
<tr>
<td>L</td>
<td>IBIS_L</td>
<td>IBIS_VCL</td>
<td>IBIS_CCL</td>
</tr>
<tr>
<td>C</td>
<td>IBIS_C</td>
<td>IBIS_VCC</td>
<td>IBIS_CCC</td>
</tr>
</tbody>
</table>
## Current Library - Sources

<table>
<thead>
<tr>
<th>Voltage Source</th>
<th>Fixed</th>
<th>Voltage Controlled</th>
<th>Current Controlled</th>
<th>Voltage Controlled with Delay</th>
<th>Current Controlled with Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBIS_V</td>
<td>IBIS_VCCVS</td>
<td>IBIS_CCVS</td>
<td>IBIS_VCVS_DELAY</td>
<td>IBIS_CCVS_DELAY</td>
<td></td>
</tr>
<tr>
<td>IBIS_I</td>
<td>IBIS_VCCS</td>
<td>IBIS_CCCS</td>
<td>IBIS_VCVS_DELAY</td>
<td>IBIS_CCCS_DELAY</td>
<td></td>
</tr>
</tbody>
</table>
### Current Library – Operators

<table>
<thead>
<tr>
<th></th>
<th>Current Controlled Current (with scaler)</th>
<th>Voltage Controlled Voltage (with scaler)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adders</td>
<td>IBIS_CCCS_SUM</td>
<td>IBIS_VCVS_SUM</td>
</tr>
<tr>
<td>Multipliers</td>
<td>IBIS_CCCS_MULT</td>
<td>IBIS_VCVS_MULT</td>
</tr>
<tr>
<td>Dividers</td>
<td>IBIS_CCCS_DIV</td>
<td>IBIS_VCVS_DIV</td>
</tr>
</tbody>
</table>
## Current Library – Operators (Cont.)

<table>
<thead>
<tr>
<th></th>
<th>Voltage Controlled Current (with scaler)</th>
<th>Current Controlled Voltage (with scaler)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Adders</strong></td>
<td>IBIS_VCCS_SUM</td>
<td>IBIS_CCVS_SUM</td>
</tr>
<tr>
<td><strong>Multipliers</strong></td>
<td>IBIS_VCCS_MULT</td>
<td>IBIS_CCVS_MULT</td>
</tr>
<tr>
<td><strong>Dividers</strong></td>
<td>IBIS_VCCS_DIV</td>
<td>IBIS_CCVS_DIV</td>
</tr>
</tbody>
</table>
## Current Library – Active Devices

<table>
<thead>
<tr>
<th>Buffer Type</th>
<th>IBIS_BUFFER_IO</th>
<th>IBIS_BUFFER_OUT</th>
<th>IBIS_BUFFER_IN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input / Output</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output Only</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input Only</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Buffer Type</th>
<th>IBIS_BUFFER_OD</th>
<th>IBIS_BUFFER_OS</th>
<th>IBIS_BUFFER_SERIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Drain</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Open Source</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Series</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Current Library – Other Models

<table>
<thead>
<tr>
<th>Inductive Coupler</th>
<th>IBIS_K</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmission Line</td>
<td>IBIS_T</td>
</tr>
</tbody>
</table>
Sample AMS Element Code

Inductive Coupler

module IBIS_K (p1, n1, p2, n2);
  output p1, n1, p2, n2;
  electrical p1, n1, p2, n2;
  branch (p1, n1) Out1;
  branch (p2, n2) Out2;
  parameter real Lval_1 = 1.0;
  parameter real Lval_2 = 1.0;
  parameter real Kval = 0.0;
  parameter real I0_1 = 0.0;
  parameter real I0_2 = 0.0;
  parameter real Scale = 1.0;

  // This should be declared as "localparam" so that it
  // couldn't be changed from the outside
  //localparam real M = Kval * sqrt(Lval_1 * Lval_2);

  parameter real M = Kval * sqrt(Lval_1 * Lval_2);

  analog begin
    if (Scale * Lval_1 > 0.0) begin
      I(Out1) <+ (id(V(Out1), I0_1*Scale*Lval_1) - M*(I(Out2)-I0_2)) / (Scale*Lval_1);
    end else begin
      V(Out1) <+ 0.0;
    end
    if (Scale * Lval_2 > 0.0) begin
      I(Out2) <+ (id(V(Out2), I0_2*Scale*Lval_2) - M*(I(Out1)-I0_1)) / (Scale*Lval_2);
    end else begin
      V(Out2) <+ 0.0;
    end
  end
endmodule

Simple Resistor

module IBIS_R (p, n);
  electrical p, n;
  branch (p, n) Out;
  parameter real Rval  = 1.0;
  parameter real Scale = 1.0;

  analog begin
    V(Out) <+ Scale * Rval * I(Out);
  end
endmodule
Current Issues

- Verilog-A(MS) does not allow a string to be passed as a file name to the $table_model keyword
  - Every instance of the building block would have to be duplicated with a hard coded file name, or
  - we could pass the data into the building block as parameters from the netlist when instantiating it
- Verilog-A(MS) provides NO file parsing capabilities to read data from external files
  - The data will have to be extracted from the IBIS file manually or by a separate script (VHDL-AMS doesn’t have this limitation)
- Verilog-A(MS) has very limited array features
- VHDL-AMS and Verilog-A(MS) do not allow expressions to be passed as parameters like (H)SPICE
Next Steps

- Create more templates to find out what additional building blocks need to be written
- We need semiconductor vendor participation – using the library to model current parts
- Translator implementation for EDA companies without native Verilog-A(MS) / VHDL-AMS support
Final Thoughts

• This approach should help speed the adoption of AMS as a modeling language for signal integrity purposes

• Full AMS models (as opposed to templates based on macromodels) will appear once all EDA vendors support AMS