Macromodeling vs. AMS modeling in IBIS (update)

Behavioral Modeling Wrokgroup
April 14, 2005

Arpad Muranyi
arpad.muranyi@intel.com

intel

CHIPSET GROUP

Background

- This issue started with a presentation from Donald Telian (Cadence), first given at the January 31, 2005 IBIS Summit
- The presentation was repeated at the March 11, 2005 IBIS Summit and the March 23 Cadence webinar
- The subject has been vigorously discussed on the IBIS email reflectors and at various meetings within Intel

http://www.eda.org/ibis/summits/jan05/telian.pdf

http://www.eda.org/ibis/futures/ibis-futures-issues-mm.pdf

The real problem

- IBIS is running out of steam for advanced buffer modeling
  - it took a long time to get [Driver Schedule] to work for pre/de-emphasis buffers
  - other fancy buffers may follow
  - strict IBIS simulators can't do anything about it
  - HSPICE allows tricks around B-element
  - Cadence does the same tricks in DML (KSPICE)
- Many SI engineers don't want to fiddle with tricks, so they prefer transistor level models
  - this is amplified by the common belief that transistor models are more accurate than behavioral models
- The *-AMS extensions in IBIS are too advanced
  - not too many simulators support it yet
  - the new language is a deterrent factor
  - few *-AMS models exist

intel
CHIPSET GROUP

Cadence's idea and proposal

- Use Berkeley SPICE as if it was K-SPICE or HSPICE with the basic IBIS 2.1 engine (B-element)
  - Berkeley SPICE is one of the languages in IBIS 4.1
  - problem: Berkeley SPICE is very limited with its EFGH elements to be useful for anything like that
- Proposal: extend Berkeley SPICE's features
  - problem: Berkeley SPICE stopped being developed in 1993
  - proposal: Add missing SPICE features to the IBIS specification

Evaluating Cadence's proposal

- Provides useful solutions for the short term
- May not be able to model future buffers
  - CSI buffers with FIR filters, or decision feedback loops
- The changes require BIRDs for the IBIS spec
  - usually slow process
  - the faster we want to do it the fewer the features will be, and the more often BIRDs will need to be written
- This slows down the already slow acceptance rate of *-AMS modeling
- This is kind of standardizing SPICE
  - wouldn't be a bad idea after 20+ years…
  - who's syntax should it be?
  - Cadence may be willing to "donate" K-SPICE to public
  - K-SPICE has no transistor model capabilities!

Vendor neutral SPICE syntax?

- Invent a vendor neutral SPICE syntax from scratch to be fair to all SPICE vendors
  - this is to avoid favoritism with any vendor
  - makes it equally hard for each vendor to implement it
  - helps to avoid any copyright issues
- This will not address technical implementation differences
  - the "neutral" SPICE syntax may have certain features which exist in one tool, but not another, or
  - some features may resemble one tool's syntax more than another tool's syntax
- It would take a considerable amount of time to write such a standard SPICE syntax from scratch

intel
CHIPSET GROUP

Is there another way?

- Write a SPICE compatible library of primitives using the analog subset of *-AMS
  - the *-A(MS) primitives could be translated to tool specific SPICE equivalents with a script by each tool
  - tools understanding *-AMS could use them directly
- Use *-A(MS) netlisting for macromodeling
  - *-AMS “netlists” are very much SPICE like
  - the user of these primitives wouldn’t have to know about the underlying equations
- No changes are required in the IBIS spec
  - [External Circuit]s can be used to instantiate the macromodels in IBIS
  - this could be done as we speak

Pros and cons

+ No changes are required in the IBIS spec
  - [External Circuit]s can be used to instantiate these "macromodels" in IBIS
  - this could be done as we speak
+ Expandable
  - the *-A(MS) library of primitives can be extended without any spec changes
+ Does not conflict with the general direction
  - true *-AMS tools can also use these models
  - gets people used to the *-AMS idea
– Macromodeling would be written in *-A(MS), not the familiar SPICE syntax
  - model writers would need to learn a little bit
– Need to write two *-A(MS) libraries
  - Verilog-A and VHDL-A(MS)

VHDL-AMS netlist (pg. 1)

```
-- genhdl\test_004/test_004.vhd
-- Generated by SystemVision netlister 1.0 build 2005.25.1_SV
-- File created Thu Apr 07 09:59:57 2005

LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.electrical_systems.all;
USE ieee.mechanical_systems.all;
USE ieee.fluidic_systems.all;
USE ieee.thermal_systems.all;
USE ieee.radiant_systems.all;
LIBRARY edulib;
USE work.all;

entity TEST_004 is
end entity TEST_004;

architecture arch_TEST_004 of TEST_004 is
   terminal \$1N3\: ELECTRICAL;
   terminal OUT_N: ELECTRICAL;
   terminal OUT_P: ELECTRICAL;
   signal \$1N10\: STD_LOGIC;

begin
```

VHDL-AMS netlist (pg. 2)

```
begin

   R1 : entity EDULIB.RESISTOR(IDEAL)
      generic map ( RES => 100.0 )
      port map ( P1 => OUT_N,
                 P2 => OUT_P );

   \BUFFER\ : entity WORK.PXP_OUTPUT(MIN)
      port map ( IN_D => \$1N10\,
                 OUT_P => OUT_P,
                 OUT_N => OUT_N,
                 PC_REF => \$1N3\,
                 PD_REF => ELECTRICAL_REF,
                 GC_REF => ELECTRICAL_REF );

   V1 : entity EDULIB.V_CONSTANT
      generic map ( LEVEL => 1.8 )
      port map ( POS => \$1N3\,
                 NEG => ELECTRICAL_REF );

   DIG_PULSE1 : entity EDULIB.DIG_PULSE(IDEAL)
      generic map ( INITIAL_DELAY => 2.0NS,
                    PERIOD => 12.0NS )
      port map ( OUT_STATE => \$1N10\ );

end architecture arch_TEST_004;
```


intel

CHIPSET GROUP

HSPICE Verilog-A example

```
Verilog-A version of the SPICE JFET
.hdl jfet.va
.options post=1
VCC Drain 0 3.0
VG Gate 0 0.5
VS Source 0 0.0
X1 Drain Gate Source jfet Vto=-2.0 Beta=1.1e-4 Lambda=0.01
.dc VCC 0.0 4.0 0.01 VG -2.0 0.0 0.5
.print I(VCC)
.end
```

```
`include "constants.vams"
`include "disciplines.vams"
module jfet(d, g, s);
parameter real Vto = -2.0 from (-inf:inf); // Threshold voltage
parameter real Beta = 1.0e-4 from [0:inf);// Transconductance
parameter real Lambda = 0.0 from [0:inf); // Channel modulation
electrical d, g, s;
real Id, Vgs, Vds;
analog begin
Vgs = V(g,s);
Vds = V(d,s);
if (Vds <= Vgs-Vto)
Id = Beta*(1+Lambda*Vds)*Vds*(2*(Vgs-Vto)- Vds);
else if (Vgs-Vto < Vds)
Id = Beta*(1+Lambda*Vds)*(Vgs-Vto)*(Vgs-Vto);
I(d,s) <+ Id;
end
endmodule
```


intel
CHIPSET GROUP