

Automatic Mixed-Signal Design Verification Instrumentation with Observation Specification Language

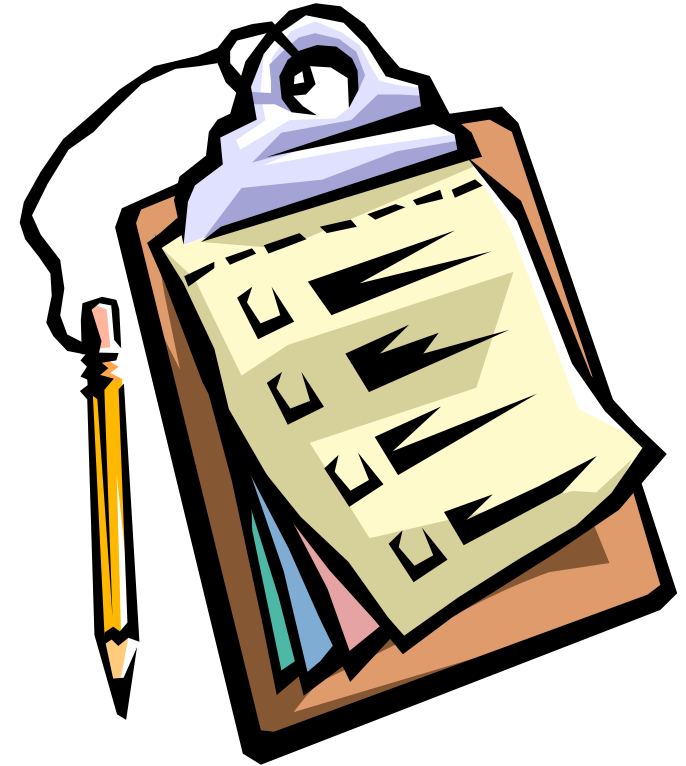
Jonathan David
Scintera Networks, Inc.

IEEE-BMAS 2007



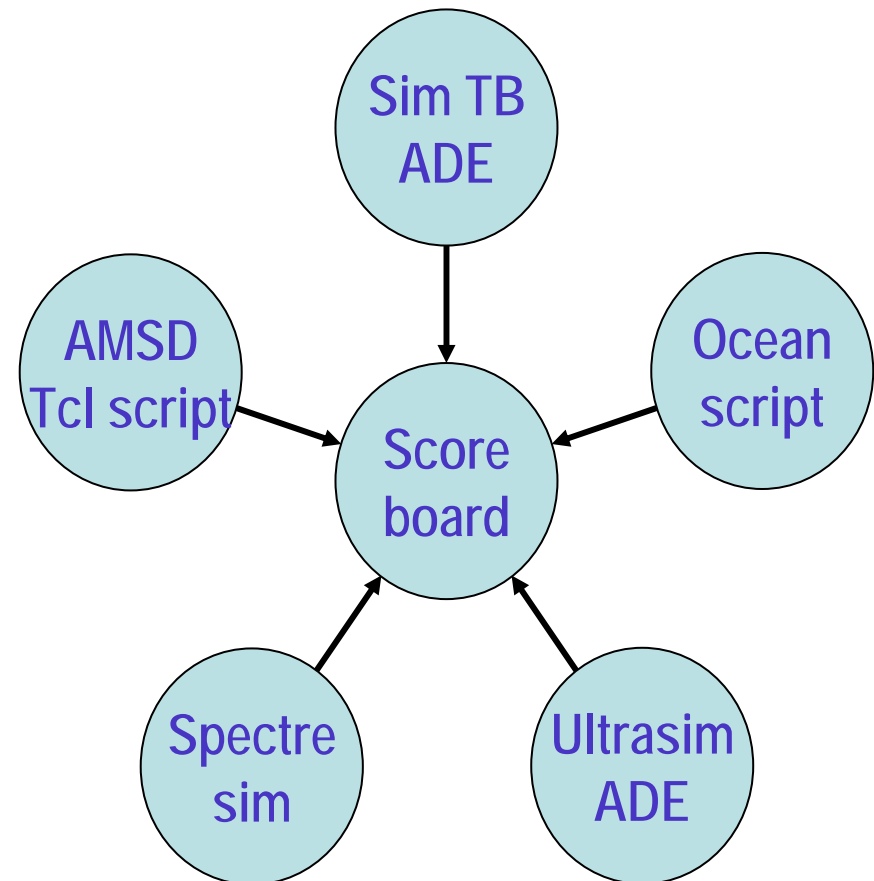
What's Missing from the simulation plan?

- Typical AMS Design:
 - 30K+ transistors
 - 20% digital P&R
 - 40%+ High Bandwidth Analog
 - 20% High Speed CDR
 - 20% Bias, Telemetry, & Support
 - 3+ IP vendors
 - Internal IP from 2 or more prior projects
 - New circuits
- Which Circuits
 - Need more corners run, more control signals checked?
 - Haven't been simulated in their new context?
 - Haven't been simulated at ALL since the latest PDK change?
- A Good Plan is a great help.. But It can't tell you what you left out..



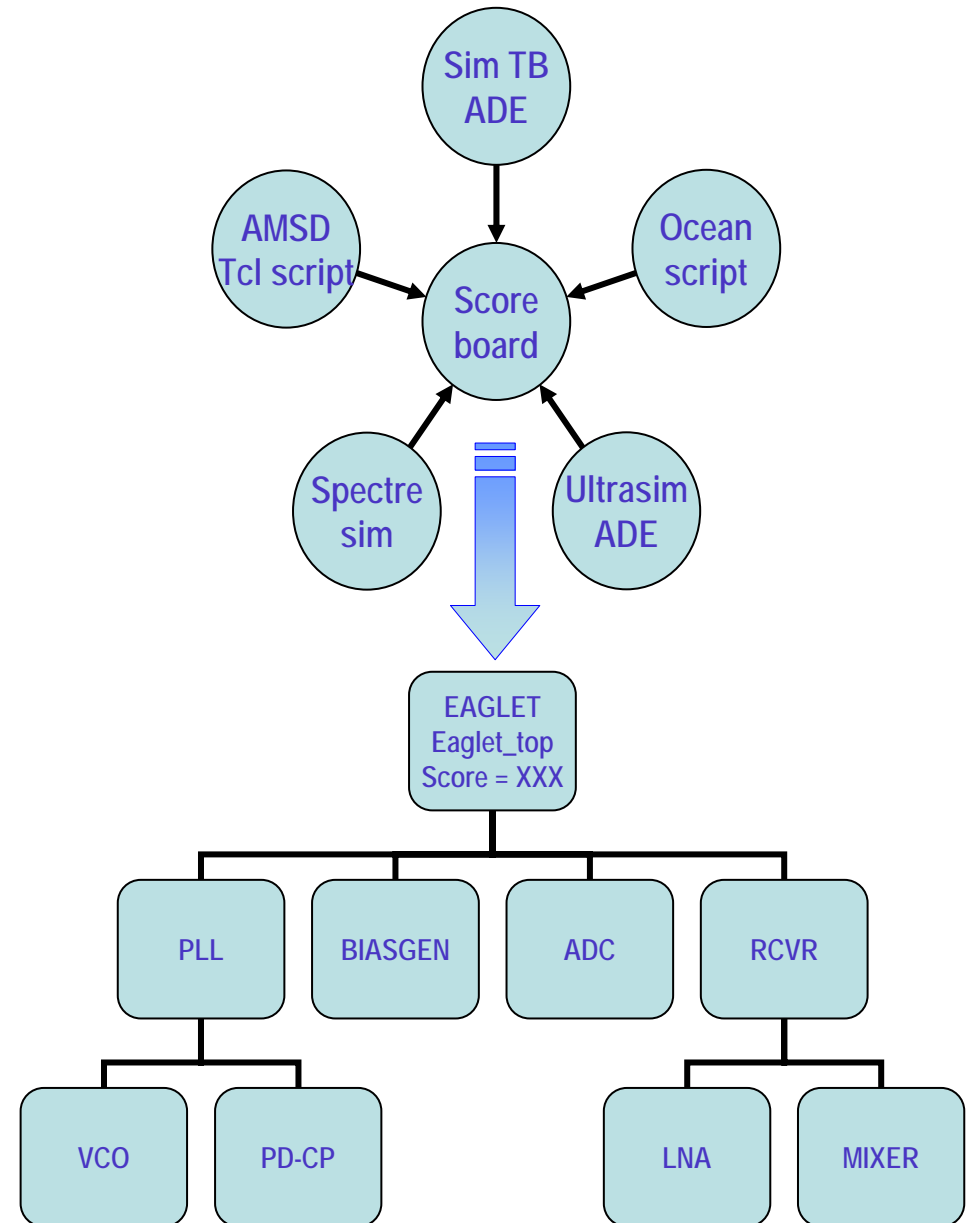
What data do we need? How to collect and organize it?

- Analog Simulation is typically Test Bench Oriented.
 - Netlists, simulations scripts, and results databases are organized by the Testbench Schematic/module at the TOP.
- Information we NEED is Design Block Oriented – for ALL the simulations run:
 - View used (model, sch, extracted etc..)
 - PVT corner, Control signals
 - Which Simulator?
 - Lib, Cell, View structure works.
- Methodology should cover ALL simulators and Environments (we use).



What data do we need? How to collect and organize it?

- Analog Simulation is typically Test Bench Oriented.
 - Netlists, simulations scripts, and results databases are organized by the Testbench Schematic/module at the TOP.
- Information we NEED is Design Block Oriented – for ALL the simulations run:
 - View used (model, sch, extracted etc..)
 - PVT corner, Control signals
 - Which Simulator?
 - Lib, Cell, View structure works.
- Methodology should cover ALL simulators and Environments (we use).



What's similar? What's Done?

- Digital Verification – Coverage Assertions
 - PSL assertions can be placed in model or separate vunit
 - Beyond “code” coverage to “design coverage”
 - Scoreboard concept
- Message passing to Databases
 - XML is not data “efficient” but
 - parsers are widespread (Firefox/IE can read most)
 - Readable format for troubleshooting
- Verilog-A/AMS can be used in most simulators
 - As language for wrapper, works in Spectre & AMSD.
- Ocean Scripting
 - Limited to ADE
 - Powerful
 - Postprocessing only
 - Could be used for non-transient data.
- Spice “Measure”
 - Not avail in spectre
 - Formatting?
 - Can stay with subcircuit?
 - Record Process Info?
- MDL
 - Limited to Spectre,
 - Need AMS flow.



Language – What's in a Name?

- PSL doesn't support what we need.
 - “Assertion” is too “logical” for analog.
 - Observe or Measure is more appropriate –
 - Measure taken by SPICE years ago.
 - We are not just “covering” gates/behavior but
 - Environment – Temp, Supply voltages, Bias currents
 - Process, normal, HV, MOS corners, Resistors, Capacitors Inductors.
 - Controls – both digital and Analog
 - All of these define a “point” in the design volume.
- We'll create Observation Specification Language (OSL)
- To measure the “Enclosure” of our Verification

Assertions
Cover



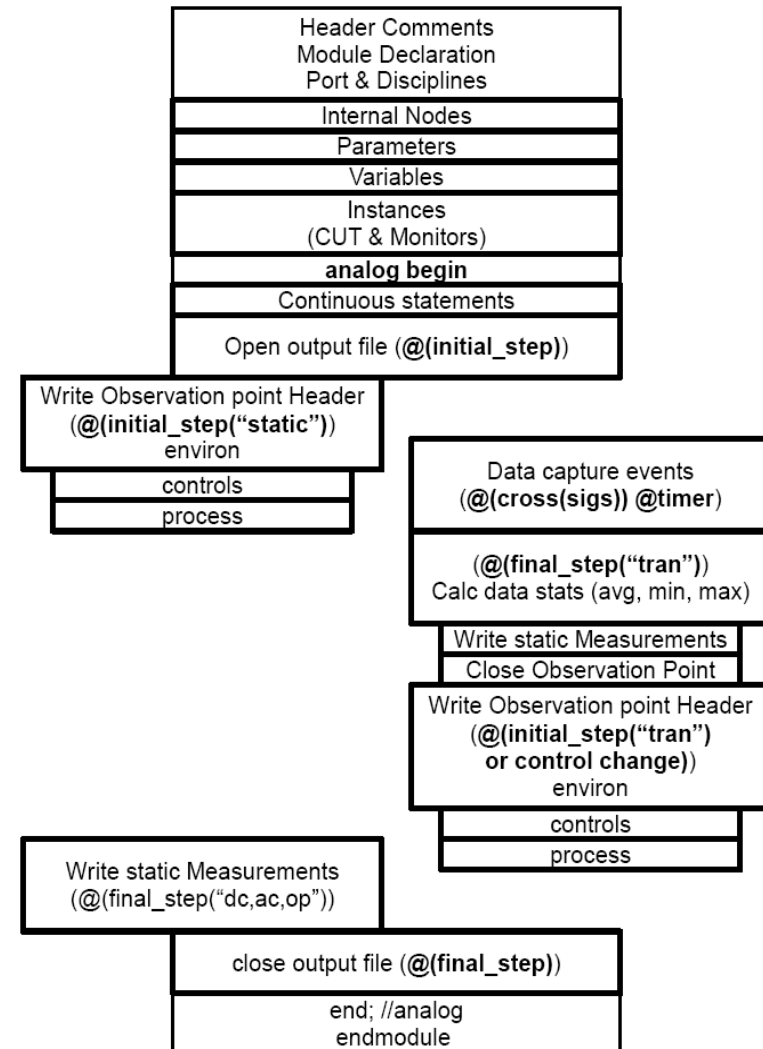
Observations
Enclose

- Specify info to capture
 - Vaunit – OSL in comments
 - Netlister can find all devices outside instrumented blocks
- Build wrapper view
 - Same pins/Symbol as design block
 - Perl automation
- Use wrapper view in simulations
 - Records data in xml files
- Import xml “messages” into Enclosure database
 - Score the design
 - View results to identify where additional simulations are needed [jbdavid:CDNLive-2007]

```
// VerilogA for CHRONOS_top_sim, CML_demobuf, vaunit
// msdv dut CHRONOS_top_sim.CML_demobuf:schematic ;
`include "constants.vams"
`include "disciplines.vams"
module CML_demobuf(out_n, out_p, gnd_a, vdd33, vdd_a, in_n, in_p, iref500u,
    pwrdn);
output out_n, out_p;
inout gnd_a, vdd33, vdd_a;
input in_n, in_p, iref500u, pwrdn;
electrical out_n, out_p;
electrical gnd_a, vdd33, vdd_a;
electrical in_n, in_p, iref500u, pwrdn;
// msdv Tsub: observe environ.temperature;
// msdv Vgnd_a: observe environ.reference #(.units("V")) (gnd_a);
// msdv Vdd_a: observe environ.supply #(.units("V")) (vdd_a);
// msdv Iq_vdd_a: measure dcamps #(.units("A"),.scalar("m"))
// when(analysis("static","tran")) (vdd_a);
// msdv Ibref500u: observe environ.bias #(.units("A"),.scalar("u"))
// (iref500u);
// msdv Vbref500u: measure dcvolts #(.units("V")) when(analysis("static"))
// (iref500u,gnd_a);
// msdv mos: observe process.cmos #(.tox_units("Ang"),.cj_units("pf/um^2"),
// .dvth_units("V")) mos_pmonitor;
// msdv scn_vars: observe process.passive #(.count(3),
// .what({captol,indtol,restol})) passives_monitor;
// msdv pwrdn: observe control.binsig #(.vth(0.75)) (pwrdn);
// msdv out_in: measure sigpath.gain_delay #(.samples(25),.diff(TRUE),
// .inv(FALSE),.firstsample(10), .units("none"), .timeaccy(5p) ,
// .dly_units("s"),.ampl_units("V"),.dscalar("p"),.stdy_dly(50p),
// .start_time(100p)) while(!pwrdn) from(in_p,in_n) to(out_p,out_n);
endmodule
```

Anatomy of a Wrapper Module

- defining a standard structure for the wrapper
 - Script parses OSL statements
 - Looks up template for each part
 - Substitutes parameters
 - And append to that part
- Start with Structural
 - Circuit Under Test (CUT)
 - Process monitor subcircuits
- Initial step
 - Open file – write headers
 - Capture static observation point
- Measure Events
 - Capture & process info
- Control Events
 - Add observation point
- Final step
 - Write data, and close file



Writing XML from Verilog-A

- Formatted print statements
- Order of print statements must be right
 - Especially if element is spread across lines.
- Use a browser to check.
- File Append can be a problem
 - If simulator doesn't create a new file at the right time.
 - Appending a properly formatted XML file to another complete XML file creates an INVALID XML file.
- This is an area for extending the language with special print statements.
 - Describing an Element structure may be interesting
 - Sub elements
 - Attributes
 - Values

```

$fstrobe(VFILE,
  "\t\t\t<gain name=\"out_in\" analysis=\"tran\"
  fromport=\"in\" toport=\"out\" units=\"none\"
  samples=\"%d\" diff=\"true\" ><r> %g </r><f> %g
  </f></gain>",
  Niout_in_m+1, Gain_out_in_m_r_sum/Nsout_in_m,
  Gain_out_in_m_f_sum/Nsout_in_m );
$fstrobe(VFILE,
  "\t\t\t<delay name=\"out_in\" analysis=\"tran\"
  fromport=\"in\" toport=\"out\" units=\"s\"
  samples=\"%d\" diff=\"true\" ><r> %g p </r><f> %g p
  </f></delay>",
  Niout_in_m+1, Dly_out_in_m_r_sum/Nsout_in_m * 1T,
  Dly_out_in_m_f_sum/Nsout_in_m * 1T );
$fstrobe(VFILE,
  "\t\t\t<amplitude name=\"out_in_in\"
  analysis=\"tran\" atport=\"in\" units=\"V\"
  samples=\"%d\" diff=\"true\" ><diff><r> %g </r><f>
  %g </f></diff><comn><r> %g </r><f> %g
  </f></comn></amplitude>",
  Niout_in_m+1,
  Vdm_from_out_in_m_r_sum/Nsout_in_m,
  Vdm_from_out_in_m_f_sum/Nsout_in_m,
  Vcm_from_out_in_m_r_sum/Nsout_in_m,
  Vcm_from_out_in_m_f_sum/Nsout_in_m );

```

Templatizing Verilog-A

- Wrapper definition file
 - uses comments +begin/end
 - Associates observe/measure type with section of the template
- Tokens (
 - designated by %%TOKENNAME%%)
 - substituted with values from the OSL statement

```
// begin observe.bias vardeclare
real %%VAR%%;
// end observe.bias vardeclare
// begin observe.bias nodedeclare
electrical %%SIG_INT%%;
// end observe.bias nodedeclare
// begin observe.bias analogmain
V(%%SIG%%,%%SIG_INT%%) <+ 0;
%%VAR%% = I(%%SIG%%,%%SIG_INT%%);
// end observe.bias analogmain
// begin observe.bias tagbody
$fstrobe(VFILE,
"\t\t\t<bias name=\"%%LABEL%%\" units=\"%%UNITS%%\" atport=\"%%SIG%%\" > %g %%SCALAR%% </bias>",
%%VAR%%*%%SCALEFCTR%% ); // print the current on DC runs
// end observe.bias tagbody
```

Passing the Message - XML

- All measures are taken at a POINT in the design volume.
 - Requires consistent coordinate system.
- A transient sim can traverse multiple (control) points.
 - Multiple points per message
- Points are classified by
 - Application – Instance name, view used
 - Environment (Voltage, current, Temp)
 - Process (from model sections)
 - Controls – affect operation of circuit
- Control signal or Environment voltage?
 - Control Changes Intended
- Measures may depend on
 - Analysis, run time
 - Stimulus
 - Controls

```

<?xml version='1.0'?><observation>
<dut><library>CHRONOS_top_sim</library> <cell>CML_demobuf</cell>
<view>schematic</view></dut> <instance> DUT </instance>
<vwrapper> $Header: ../CML_demobuf_vwrp/.. 1.6 2007/07/20 jbdavid $
</vwrapper>
<point count="0" > <environ>
  <temperature name="Tsub" units="C"> 100.0 </temperature>
  <reference name="Vgnd_a" units="V" atport="gnd_a" > 0 </reference>
  <supply name="Vdd_a" units="V" atport="vdd_a" > 1.5 </supply>
  <supply name="Vdd33" units="V" atport="vdd33" > 3.3 </supply>
  <bias name="Ibref500u" units="A" atport="iref500u" > 500 u </bias>
</environ> <process>
  <section type="cmos" name="mos">
    <tox units="Ang"><p> 27.83 </p><n> 27.43 </n></tox>
    <Cj units="pf/um^2"><p> 0.0013008 </p><n> 0.0013072 </n></Cj>
    <dVth units="V"><p> 0.026 </p><n> -0.025 </n></dVth> </section>
  <section type="passive" name="scn_vars">
    <captol> 1 </captol> <indtol> 1 </indtol> <restol> 1 </restol> </section>
</process> <control>
  <signal name="pwrdrn" type="binlogic" size="1" > 0 </signal>
</control> <measures>
  <lq name="Iq_vdd_a" analysis="static tran" units="A" > 2.12915 m </lq>
  <lq name="Iq_vdd33" analysis="static tran" units="A" > 0.00562618 m </lq>
  <Vdc name="Vbref500u" analysis="static" units="V" > 0.409153 </Vdc>
  <amplitude_static name="out_in_in" analysis="static" atport="in" units="V"
    diff="true" ><diff> 0 </diff><comn> 1.35 </comn></amplitude_static>
  <amplitude_static name="out_in_out" analysis="static" atport="out" units="V"
    diff="true" ><diff> 0 </diff><comn> 1.19136 </comn></amplitude_static>
  <gain_static name="out_in" analysis="static" fromport="in" toport="out"
    units="none" diff="true" > undef </gain_static>
</measures> </point > </observation>

```

Calculating a Volume

Bldg. Zero
wrkg. group

- It's a simple calculation if ALL the vertices exist
 - IE Total Volume of Design Space
- Otherwise it's a weird shape
 - Back to Calculus...
 - Analog but impractical
- Convert to Recursive Summation
 - Borrow the Trapezoidal Method of integration.
- What if Zero result?
 - Return % dimensions enclosed
 - % enclosure for those dimensions

$$\prod_{d=0}^N length_d$$



$$\iiint_{X,Y,Z} fshape(x, y, z) dx dy dz$$

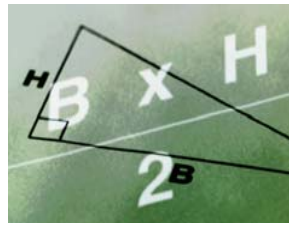
$$\sum_{i^M=0}^{N^M} \Delta D_{i^M}^M \cdots \sum_{i^1=0}^{N^1} \Delta D_{i^1}^1 \cdot \sum_{i^0=0}^{N^0} \Delta D_{i^0}^0 \cdot \left\{ \left\langle d_{i^0}^0, d_{i^1}^1, \dots, d_{i^M}^M \right\rangle \in Observed \right\}$$

The math's too hard – lets just count

Bldg. Zero
wrkg. group

- Counting would be possible if all the runs were done at typical, max or min values..
 - but the axes of the design “space” are real numbers
 - The potential is an infinite number of points.
- A count doesn't give same the insight into the interplay between the variables as a volume.
 - Some parameters are more important to vary
 - Which extra points would actually increase the volume of enclosure, or the density of points with-in the enclosure?
- Besides, the math isn't REALLY that hard..

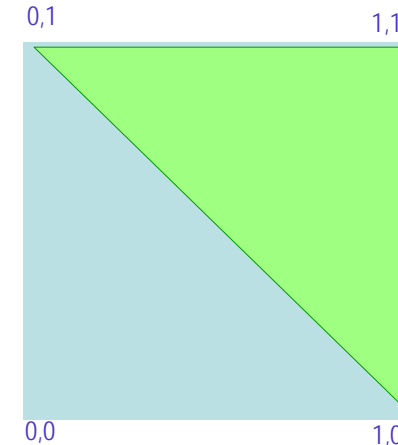
A simple example



Bldg. Zero
wrkg. group

- 3 points observed in 2 dimensions..

Dimension	point		
	0	1	2
X	0	1	1
Y	1	1	0



X = 0 number points with Y = 1, span = 0 return (0,1)

X = 1 number of points in common with Y = 2, span = 1, return (1,2)

Foreach \$x (0 .. maxx-1) {

delx = x[0+1]-x[0]; # = 1-0 = 1

trapy = avg(0, 1); # (since both have a point) = 0.5

vol += delx*trapy;

} # result = 1/2 which is correct for a triangle with base and height = 1

$$\int_Y x dy \approx \sum_{i_0}^{N-1} \frac{x_i + x_{i+1}}{2} \cdot (y_{i+1} - y_i)$$

Result for our example block

Bldg. Zero
wrkg. group

MSDV verification Enclosure for CHRONOS_top_sim . CML_demobuf : schematic

Dimension category	Vol Enclosed	Vol Total	% Enclosure	Dimension	Dimension	% Dimensions Enclosed	Dimensions lacking Enclosure
All Major Dimensions	0.25	1	25%	4	7	57.14%	applic.instance
Application Major Dimensions	0	1	0%	0	1	0%	instance
Environment Major Dimensions	20	30	66.67%	2	2	100%	
Process Major Dimensions	0.0025	0.01	25%	2	3	66.67%	section.mos.dVth.p
Control Major Dimensions	1	1	100%	1	1	100%	

Dimension category	Minor Dim	Span	Values..
applic	"vwrapper"	"Invariant"	"\$Header: vs 1.6 2007/07/20 11:58:35 jbdavid Exp \$ "
process	"section.scv_var:	"0.2"	"0.9 " "1.1 " "1 "
process	"section.mos_33	"12"	"71.5 " "65.5 " "77.5 "
process	"section.mos.Cj.l	"0.0001369"	"0.001437 " "0.001300 " "0.00136925 "
process	"section.mos.Cj.l	"0.0001376"	"0.001444 " "0.001307 " "0.001376 "
process	"section.mos_33	"12"	"71.5 " "65.5 " "77.5 "
process	"section.mos_33	"9.099999999999999e-	"0.000864 " "0.000955 " "0.00091 "
process	"section.mos_33	"0.0001267"	"0.001330 " "0.001203 " "0.001267 "
process	"section.mos_33	"0.2"	"0.12 " "-0.08 " "0.02 "
process	"section.mos.tox	"1.34"	"29.17 " "27.83 " "28.5 "
process	"section.mos_33	"0.16"	"-0.08 " "0.08 " "0 "
process	"section.mos.tox	"1.34"	"27.43 " "28.1 " "28.77 "
process	"section.scv_var:	"0.1"	"1.1 " "1 "
process	"section.dis_rpol:	"Invariant"	"260.09 "
process	"section.dis_rpol:	"Invariant"	"319.55 "
process	"section.dis_rpol:	"Invariant"	"319.55 "
process	"section.dis_rpol:	"Invariant"	"260.09 "
environ	"supply.Vdd33"	"0.4"	"3.3 " "3.1 " "3.5 "
environ	"bias.lbref500u"	"Invariant"	"500 u "
environ	"reference.Vgnd	"Invariant"	"0 "

Hierarchical Scoreboard

- Starting from Any point in the design
 - (eventually the Top) show:
- EAGLET.eaglet_top:schematic
 - xxx transistors xxx instrumented (20%)
 - xxx blocks xxx have observations recorded (10%)
 - Design Volume is NN dimensions, 3% enclosed
 - Observation Density Ratio is XX:1
 - Receiver (Lib.Cell) 5% Instrumented
 - ADC (Lib.Cell) 0% instrumented
 - DTMF (Lib.Cell) 50% observed
 - PLL (Lib.Cell) 30% enclosed
 - BIAS (Lib.Cell) 8:1 density
 - ...
 - Inst Lib.Cell Score Rollup

Where we are

- Vamsunits converted to Verilog-A wrappers with perl script
- Wrappers work in ADE/Spectre simulations
 - write valid observation data files
- Observation data files have been scored with perl scripts
 - Volume Calculations work
 - Generating CSV score sheets

A Metric for Determining the Verification Gap in Mixed-Signal designs has been defined: “Enclosure”

A prototype methodology for Calculating and Displaying a design’s Enclosure has been demonstrated.

Design teams adopting this methodology are adding observation specifications to their designs.

Future work: (not done yet!)

- Complete the Roll-Up scripts.
- Convert score sheets to XML
 - For more reliable parsing during the roll-up
 - and create XSLT for use in browsing the scoreboard
- Expand Wrapper generation capability to Verilog-AMS

Possibles:

- Expand Roll-Up script to Auto-Generate testbench Ocean script to write Observation data files .
- Begin extending simulators/environments with a native OSL capability
 - reduce/eliminate impact of “wrappers” on simulation time.