VCS使用

Revision

Enviroment

- VCS G-2012.09
- Nanosim D-2010.03-SP2
- Red Hat Enterprise Linux Server release 5.8

- Ref list:
- VCS doc : vcs.pdf, ucli_ug.pdf, dve_ug.pdf
- Nanosim doc: mixed_signal.pdf nanosimug.pdf
- others: VCS_Nanosim_CoSim_Training.pdf

内容

- 简介
- VCS 仿真流程 (UCLI & DVE)
- VCD/VPD
- Others
- VCS COSIM with NANOSIM

简介

- VCS是synopsys的仿真验证工具，支持Verilog、VHDL、SystemVerilog等
- 支持与nanosim的混合仿真(spice + verilog). 尤其是对于OTP这种行为比较复杂的IP，与nanosim的cosim能够check OTP的timing是否符合behavior model。
- 对于复杂的pattern (I2C等)，可以用verilog写，comsim仿真。方便灵活。
- VCS 包含 Discovery 可视化环境 (DVE)，这是一种高级的全功能调试和可视化环境. 可以代替verdi.

VCS help doc

- vcs -h/-help
- vcs -doc 如果有mozilla可打开网页版帮助

VCS仿真流程

- Compile Verilog source into an executable simulation

  % vcs [compile_options] design.v

  //编译 verilog 的源文件并且生成一个可执行文件 simv

- Run executable simv to perform simulation

  % ./simv [runtime_options]

  //运行 simv

- Debug Verilog design

  Text-based: Unified Command-line Interface (UCLI)

  GUI-based: Discovery Verification Environment (DVE)

VCS仿真流程

- % vcs [compile_options] Verilog_files
- 几个常用的**option**如下**:**
- -v filename -->Specifies a Verilog library file
- -y directory -->Specifies a Verilog library directory.
- -full64 -->Enables compilation and simulation in 64-bit mode
- -R -->Runs the executable file immediately after VCS links it together
- -l filename -->Specifies a file where VCS records compilation messages
- -f <filename> -->Specifies a file that contains a list of pathnames to source files and compile-time options.
- -F <filename> -->Same as the -f option but allows you to specify a path to the file and the source files listed in the file do not have to be absolute pathnames.
- -file filename -->This file can contain Specifies a file containing more compile-time options and different kinds of files. It can contain options for controlling compilation and PLI options and object files. (comment out entries: // /* */)

  EX: -CFLAGS '-I$VCS_HOME/include'

  /my/pli/code/$PROJECT/treewalker.o

  -P /my/pli/code/$PROJECT/treewalker.tab

VCS仿真流程

- +incdir+ directory+ Specifies the directory or directories that VCS searches for include files used in the `include compiler directive.

```
// Test file
`include "parameter.v"
module test (...);
...
endmodule
```

- +libext+extension+ -->Specifies that VCS search only for files with the specified file name extensions in a library directory.

  Ex: +libext+.v+.V

- -gui -->When used at compile time, always starts DVE at runtime.

- -ucli --> Forces runtime to go into UCLI mode, by default.

- -pvalue+parameter_hierarchical_name=value -->Changes the specified parameter to the specified value.

  Ex: -pvalue+udut.uchip.ucore. IDLE =0

- -parameters filename -->Changes parameters specified in the file to values specified in the file.

  Ex: assign 33 test/d1/param1

VCS仿真流程

- +v2k --> Enables the use of new Verilog constructs in the 1364-2001 standard.

Std 1364-2001 结构	Require +v2k
逗号分割敏感列表中的变量： always @(r1,r2,r3)	yes
基于名称的参数传递： modname #(.param_name(value)) inst_name(sig1,...);	yes
ANSI 标准端口和参数列表： module dev (output reg [7:0] out1,input wire [7:0] w1);	yes
在寄存器生命阶段进行初始化： reg [15:0] r2 = 0;	yes
条件编译控制: `ifdef ...`elseif	yes
除能默认的线类型: `default_nettype	yes
有符号算法扩展： reg signed [7:0] r1;	no
输出文件任务： $fopen $fscan $scanf...	no
从 runtime 命令行传递参数： $value$plusarg 系统功能	yes
部分索引功能： reg [8+:5] = 5`b11111;	yes
多维数组 reg [7:0] r1 [3:0] [3:0];	yes
持续文件名和行号： `line	yes
暗指敏感列表： always @*	yes
幂操作： r1=r2**r3;	yes
属性： (* optimize_power=1 *) module dev (res,out,clk,data1,data2);	yes
generate 段	yes
局部参数声名	yes
自动任务和函数 ： task automatic t1 ();	需要-sverilog 编译选项
常量函数： localparam lp1 = const_func(p1);	yes
带有位范围的参数： parameter bit [7:0] [31:0] P = {32`d1 , 32`d2 , 2`d3 , 32`d4 , 32`d5 , 32`d6 , 32`d7 ,32`d8};	编译选项中需要加入-sverilog 选项

VCS仿真流程

- +vcs+initreg+random --> Initializes all state variables (reg data type), registers defined in sequential UDPs, and memories including MDAs (reg data type) in the design, to random logic 0 or 1, at time zero.

  +vcs+initreg+0|1|x|z -->初始化成特定值

  +vcs+initreg 初始化正常的 memory 和多维的 reg 类型数组。例如：

  reg [7:0] mem [7:0] [15:0];

  +vcs+initreg 不会初始化 register 变量和 reg 类型以外的多维数组。

  EX: % vcs +vcs+initreg+random [other_vcs_options] file1.v file2.v file3.v

  % simv +vcs+initreg+0|1|random|<seed> [simv_options]

Syntax	Description
+vcs+initreg+0	Initializes all state variables (reg data type) and memories (reg data type) in the design, to random logic 0.
+vcs+initreg+1	Initializes all state variables (reg data type) and memories (reg data type) in the design, to random logic 1.
+vcs+initreg+random	Initializes all state variables (reg data type) and memories (reg data type) in the design, to random logic 0 or 1 (with default seed).
+vcs+initreg+100	Initializes all state variables (reg data type) and memories (reg data type) in the design, to random logic 0 or 1, with user-defined seed 100. Note: seed cannot be 1 or 0 and 1 or 0 has special meaning.

VCS交互模式

- 必须以交互模式编译你的设计，才能够用 VCS 的交互模式仿真
- 要在编译的过程中加入-debug/debug_all 选项,才能进入交互模式
- 如果没有以上选项，进入batch 模式。
- **调试模式也叫交互模式，一般是用在如下状况：**

设计的初始阶段，这个时候需要用调试工具例如： DVE/UCLI 进行调试

采用了 PLI 的时候

采用 UCLI 命令强制了一个信号，将值写入到了 register/net 中。

- VCS 在编译的时候有如下选项可以实现调试模式：

-debug_pp,-debug 和-debug_all

%vcs –debug_all [compile_options] top.v

VCS交互模式

- -debug_pp

  Gives best performance with the ability to generate the VPD/VCD file for post-process debug. It is the recommended option for post process debug.

- -debug

  Gives average performance and debug visibility/control i.e more visibility/control than –debug_pp and better performance than –debug_all.

  It provides force net and reg capabilities and set value and time breakpoints

- -debug_all

  Gives the most visibility/control and you can use this option typically for debugging with interactive simulation.

  it adds simulation line stepping and allows you to track the simulation line-by-line and setting breakpoints within the source code. With this option, you can set all types of breakpoints (line, time, value, event etc).

VCS UCLI

- the Unified Command-line Interface (UCLI)
- UCLI仿真其实是建立在TCL语言之上的。所以在UCLI结构中你可以使用TCL命令来控制UCLI的执行。下面的命令就是建立UCLI仿真：

  % simv [simv_options] –ucli

  注意：使用上述命令需要在编译的时候使用-debug_all,debug,debug_pp选项进行编译
- 在运行UCLI的时候可以通过命令的形式来控制UCLI的执行。如下：：

  % simv –ucli

  ucli % source file.cmds

  or

  % simv –ucli –do file.cmds

VCS DVE

- 下面的命令是用来调用 DVE 工具进行仿真的：

  % simv –gui

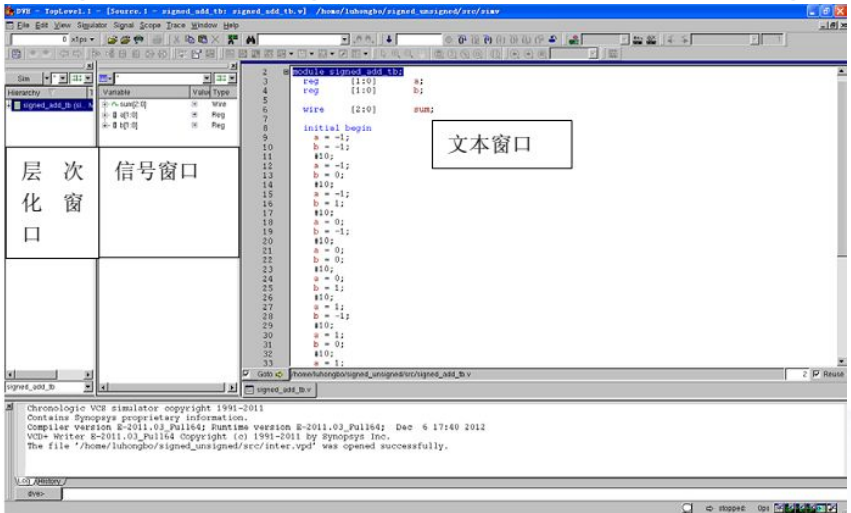
  调用了 DVE 的图形界面，但是前提是在编译的时候一定要采用-debug**类型的选项进行编译。

  采用如下命令来调用 DVE 实现后处理模式：

  %dve –vpd [VPD/EVCD_FILENAME]　(Note: 此种方式可直接查看最终dump的结果)

- 也可以直接编译仿真并打开DVE, 如下：(Note: 此种方式仿真会停在time 0，通过在DVE中添加相应的参数，让仿真继续)

  vcs –v signed_add.v signed_add_tb.v –full64 –debug_all –R –gui



VCS DVE

- Using the -session command

  The -session command invokes DVE with the design database file, test.vpd, and applies settings from the session file, mysession.tcl

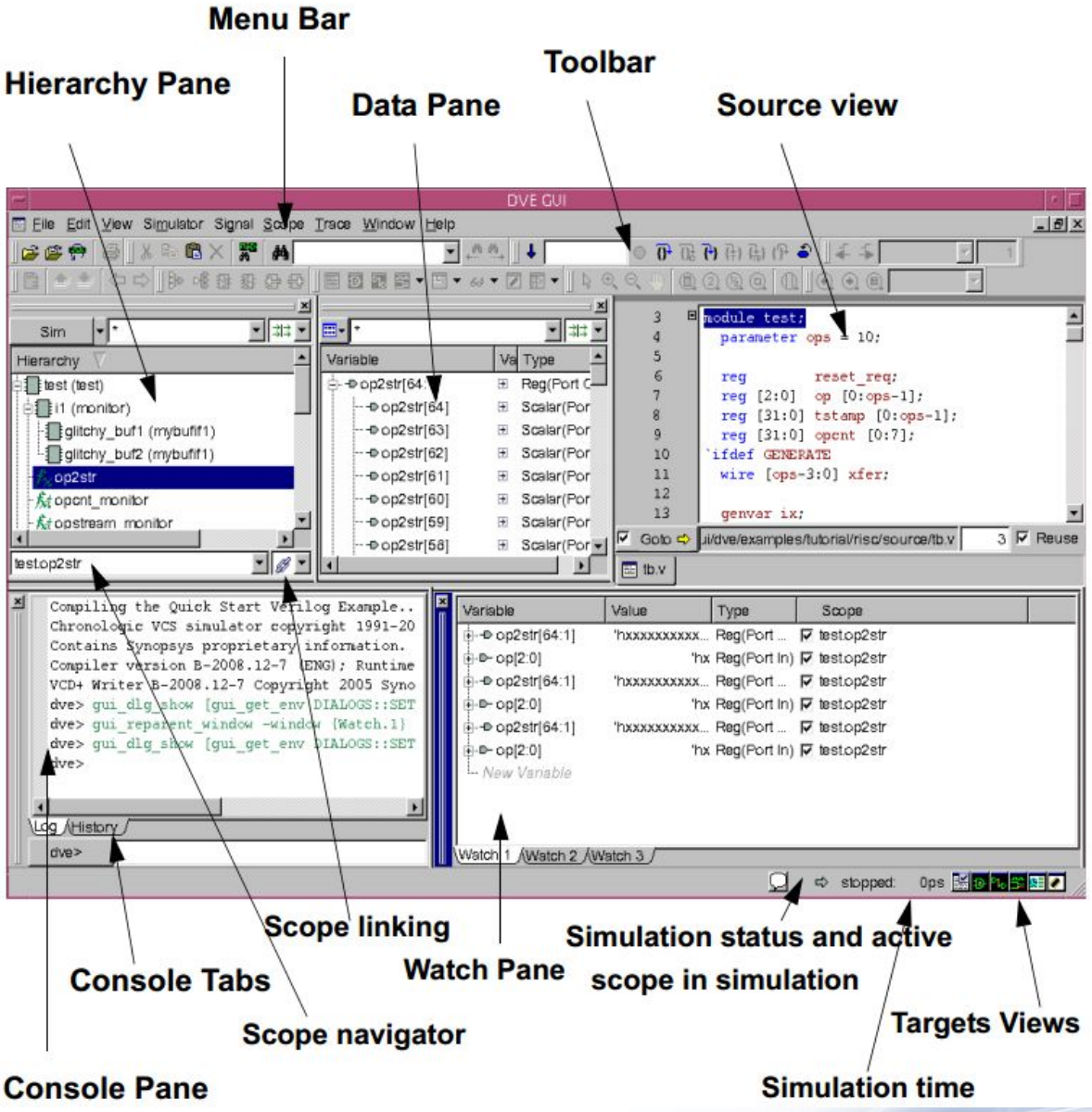
  Ex: % dve -vpd test.vpd -session mysession.tcl

- Using Tcl Scripts

  dve -cmd [tcl_cmd] -->Invokes DVE and executes the Tcl command enclosed in quotation marks.

  dve -script [tcl_file] -->Invokes DVE and reads the Tcl script specified as argument.

DVE Top-level Frame



DVE Target Views

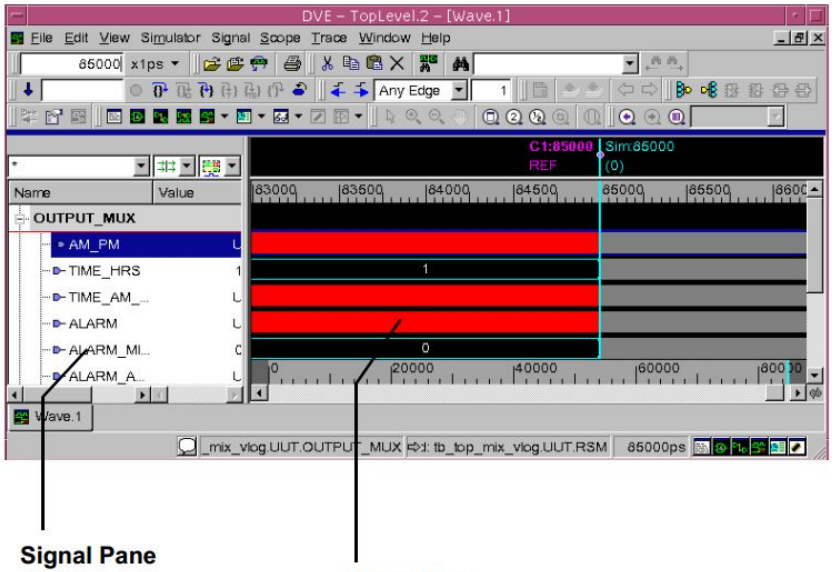
- DVE和Verdi的功能类似，熟悉Verdi上手DVE也很快。DVE和VCS的无缝连接，会优于Verdi .
- **PS: Synopsys 已收购springsoft，后续tool应该结合的更好.**

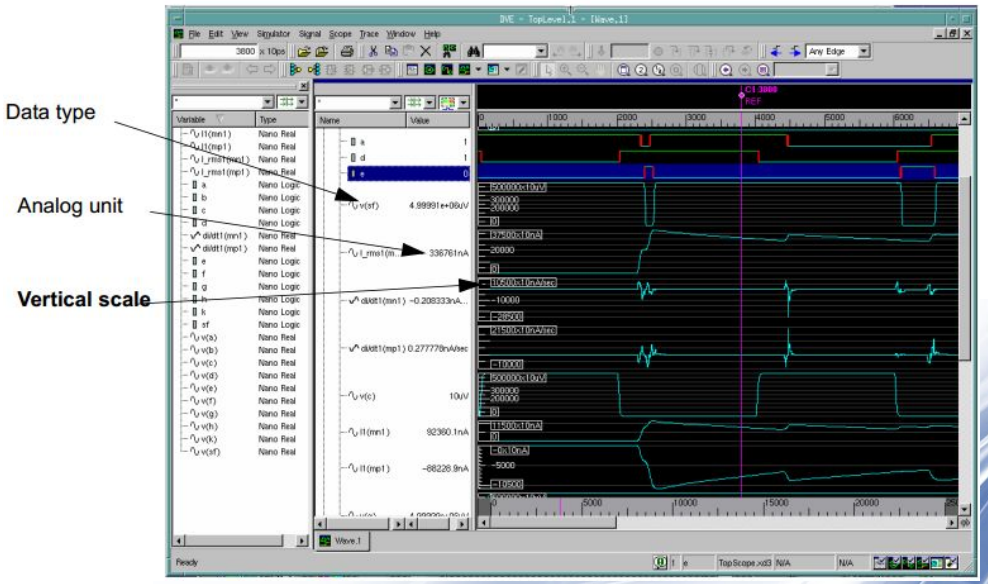
Icons	What it targets..
 **Source**	New Source view in a new TopLevel window.
 **Schematic**	New Schematic view in a new TopLevel window.
 **Path Schematic**	New Path Schematic view in a new TopLevel window.
 **Wave**	New Wave view in a new TopLevel window.
 **List**	New List view in a new TopLevel window.
 **Memory**	New Memory view in a new TopLevel window.

DVE Waveform

- Select a scope or object from the Hierarchy pane, Datapane,Source view, List view, Schematic view, or Assertion view.
- Click the Add to Waves icon in the toolbar.

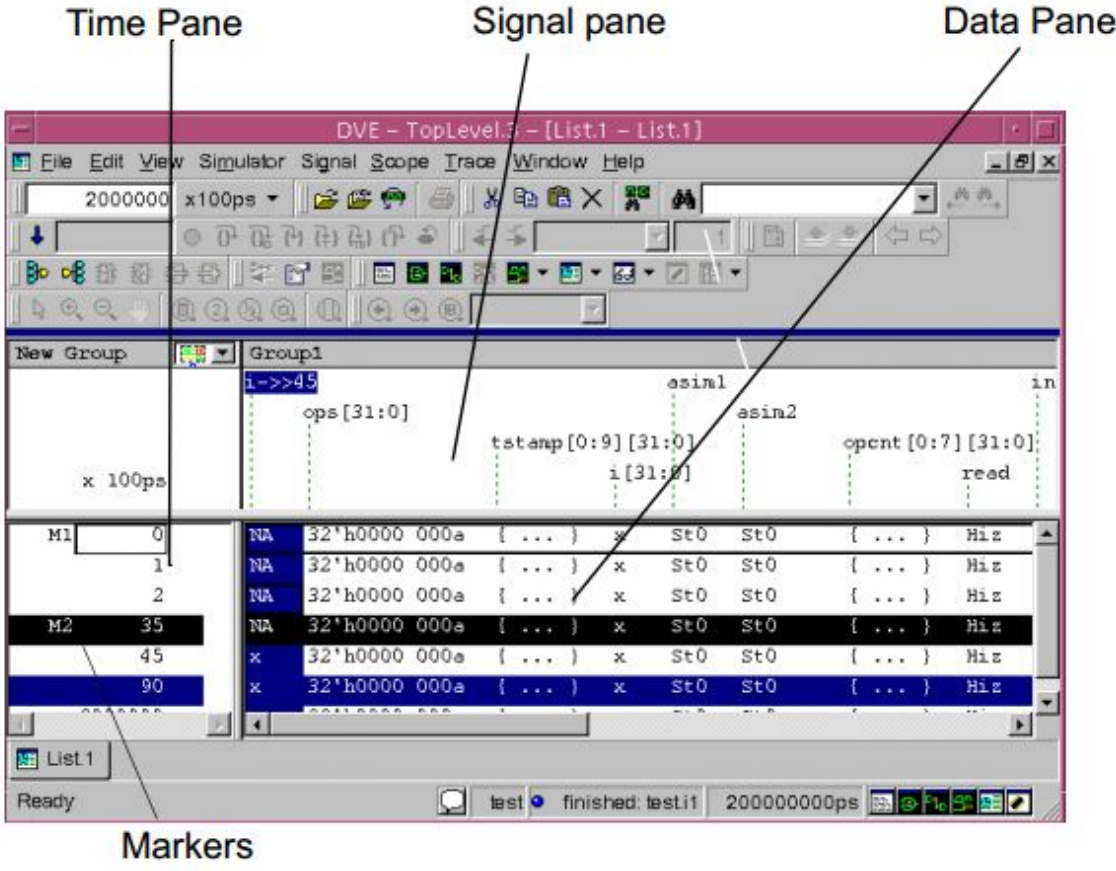
  按快捷键Ctrl+4会比较快
- DVE supports display of Nanosim signals dumped to a VPD file.



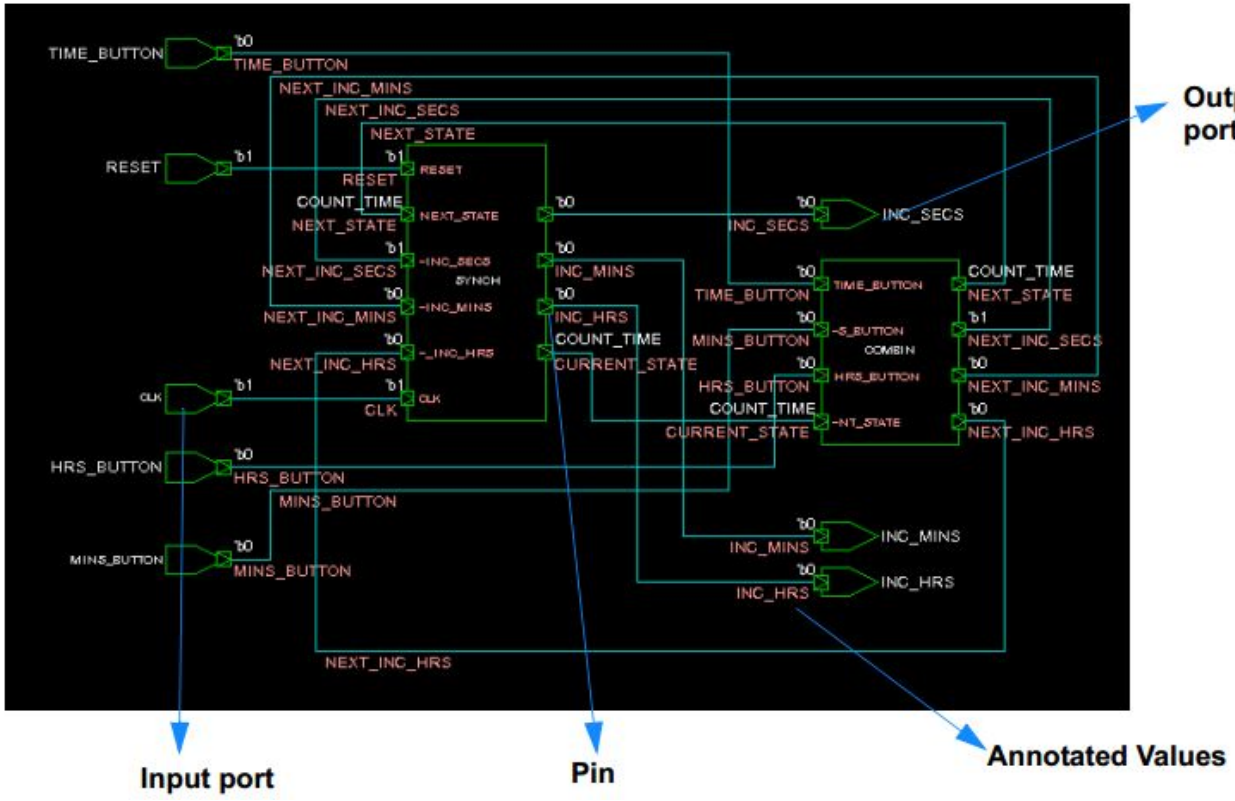


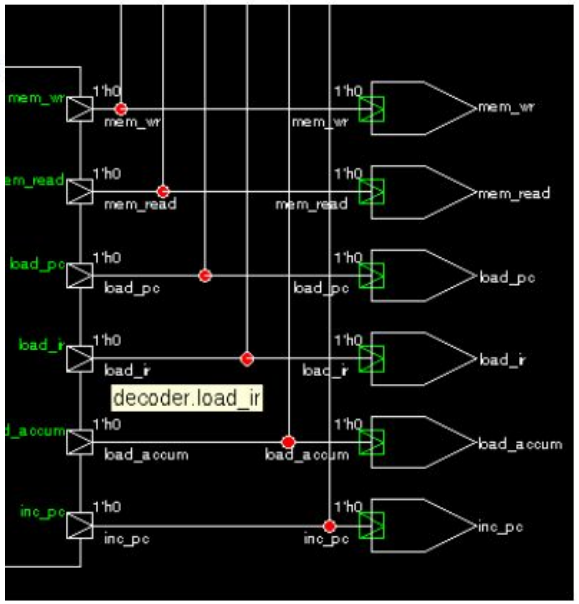
DVE List View

- View simulation data.
- Create and delete markers
- Set signal properties
- Associate signal with any database
- Compare signals
- Save signal values



DVE Design Schematic View





- Annotating Values

保存波形

- 三种波形文件

Value Change Dumping (VCD)

VCD is the IEEE Standard for Verilog designs. You can save your simulation history in VCD format by using the $dumpvars Verilog system task.

VCDPlus Dumping (VPD) <-- suggestion

VPD is a Synopsys propriety dumping technology. Provides a compressed binary format.To dump a VPD file, use the $vcdpluson Verilog system task.

大小大概是VCD的1/8倍，使得仿真速度，以及波形打开速度更快.

Extended VCD (EVCD)

EVCD dumps only the port information of your design.

VCD文件产生

- 准备工作: 修改testbanch

```
initial
        begin
                $dumpfile("div_wave.vcd");
                $dumpvars;// $dumpvars(0,div_tb)
        end
```

(其他系统任务: $dumplimit $dumpoff $dumpon $dumpflush …)

- 直接运行VCS生成VCD文件

Ex: vcs *.v –R

VPD文件产生

- 准备工作:修改testbench文件

```
initial
  begin
    //$vcdplusfile (" div_wave.vpd"); 这个要放在$vcdpluson前面。也可以通过+vpdfile+的方式产生.
    $vcdpluson(0,div_tb);
    #5 $vcdplusoff ;
  end
```

- 生成VPD文件

```
vcs *.v –R +vpdfile+div_wave.vpd
```

（这种方式方便更改产生的名称，不需要更改testbech中的code）

(若不加+vpdfile+…, 默认产生vcdplus.vpd文件)

PLI

- 连接C程序与VCS仿真器的接口
- 用途:

  编写自己的系统函数

  在testbench中产生激励(动态指令发生器)

  Verilog模块与C model联合仿真

- 3个文件

  .c文件: c函数

  .tab文件: 将c函数映射为verilog中可调用的系　　统函数

  .v文件: 调用.tab中的系统函数(实际上是调用其映射的c函数)

- 目前未用到暂不详细描述

coverage

- 4种coverage:

```
Line                    //vcs  -cm line …
Condition               //vcs  -cm cond …
Toggle                  //vcs  -cm tgl …
FSM                     //vcs  -cm fsm …
                        //vcs  -cm line+cond+fsm
```

- 结果存在simv.cm文件夹中,用cmView命令可以观察各种测试覆盖率
- 目前未用到暂不详细描述

POSTSIM

- Testbench中添加:

  $sdf_annotate("../HARDCORE/mClkGen_eco.sdf",U_TOP.U_ClkGen,,"ClkGen.log");

  或者通过添加option的方式来反标注

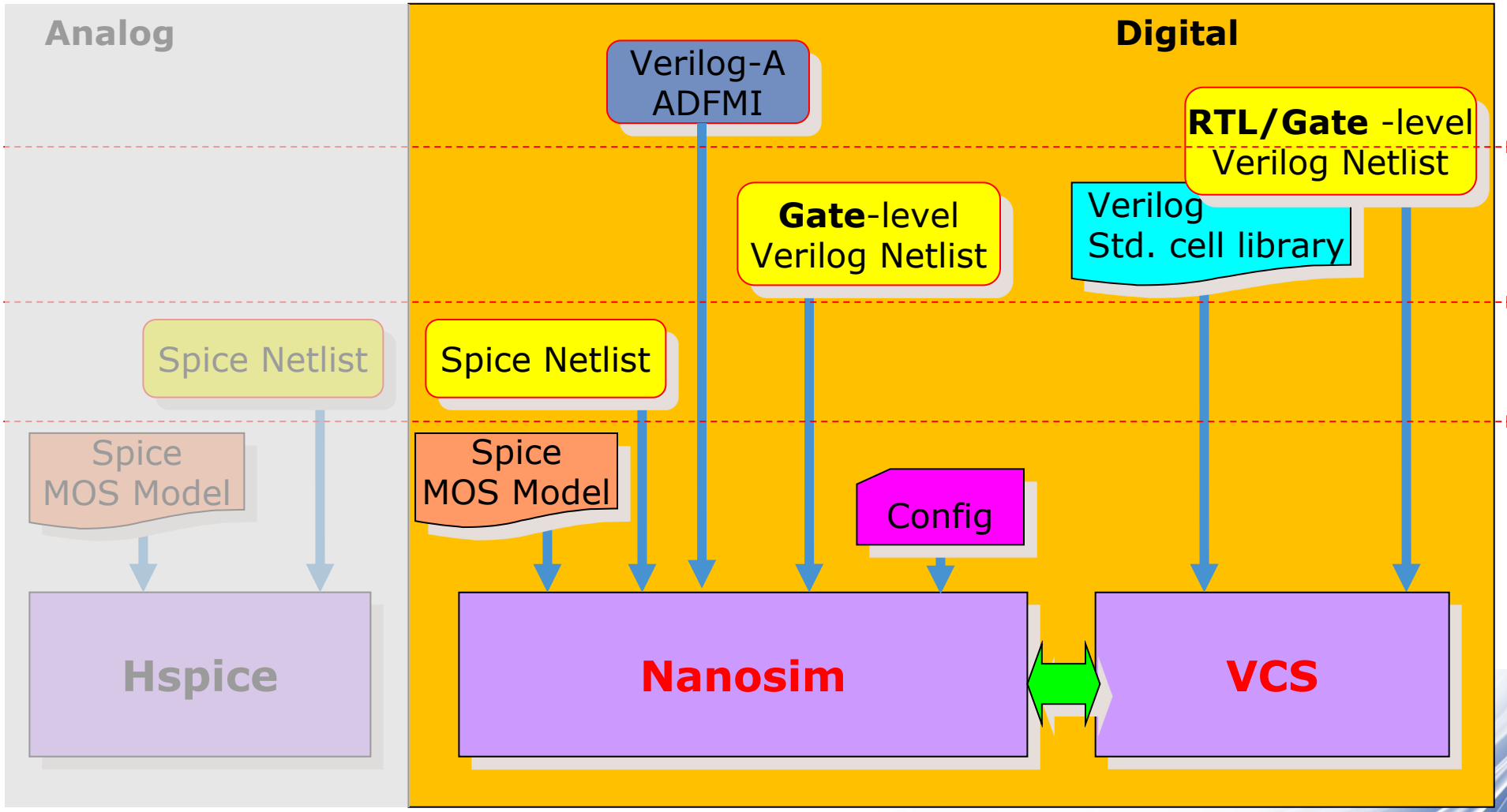
  % vcs -sdf min|type|max:instance_name:file.sdf [compile_options]

- 命令

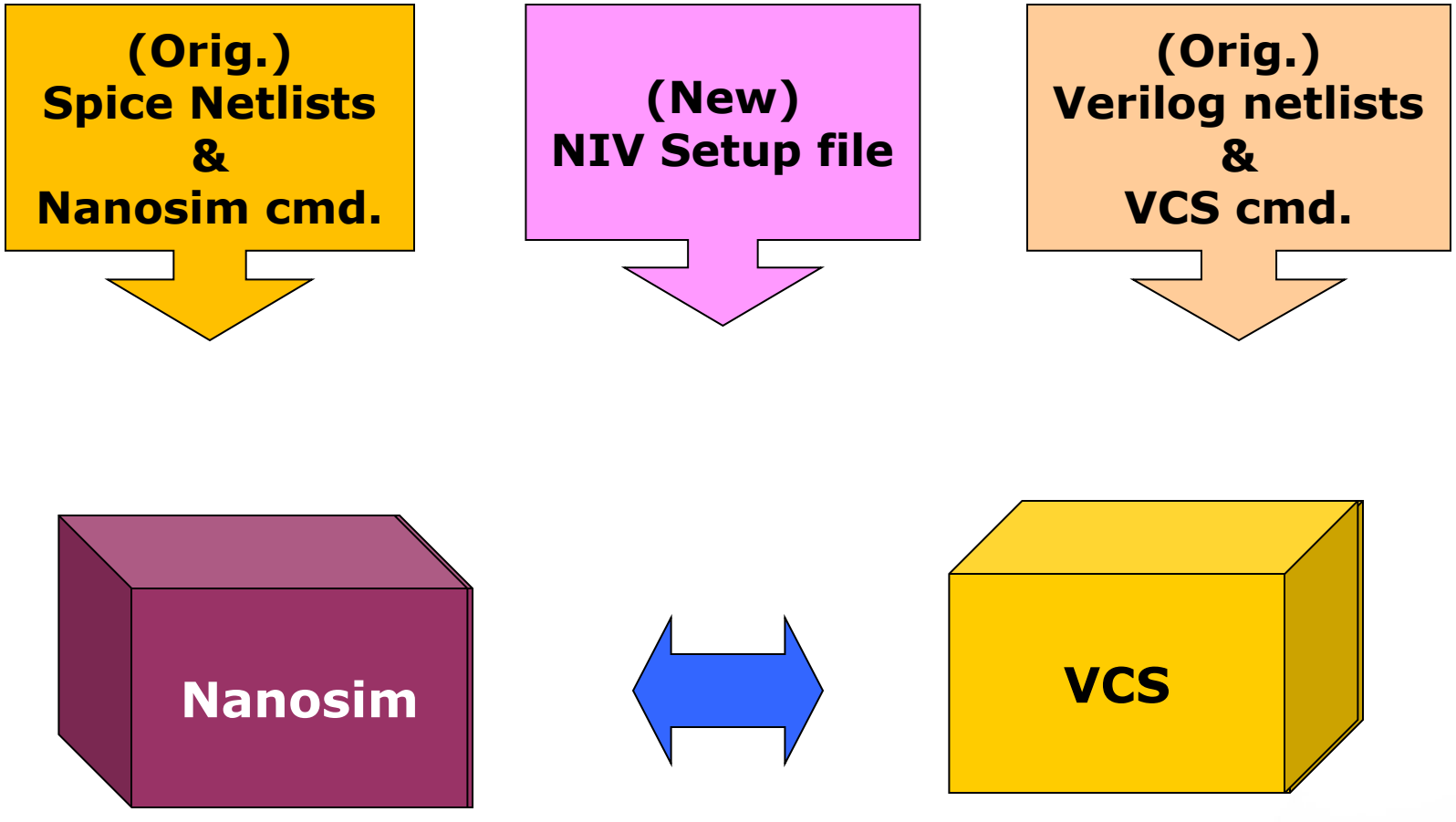
  vcs *.v -R +notimingcheck +maxdelays …

  +notimingcheck: Tells VCS to ignore timing check system tasks when it compiles your design. This option can moderately improve simulation performance.

Integrating VCS with NanoSim



NIV input files



How to run NIV

Run NIV script :

Put all of your VERILOG netlists in VCS

```
#!/bin/csh
vcs -full64 testbench.v –f verilog.list +ad=vcsAD.init -R
```

NIV setup file : vcsAD.init

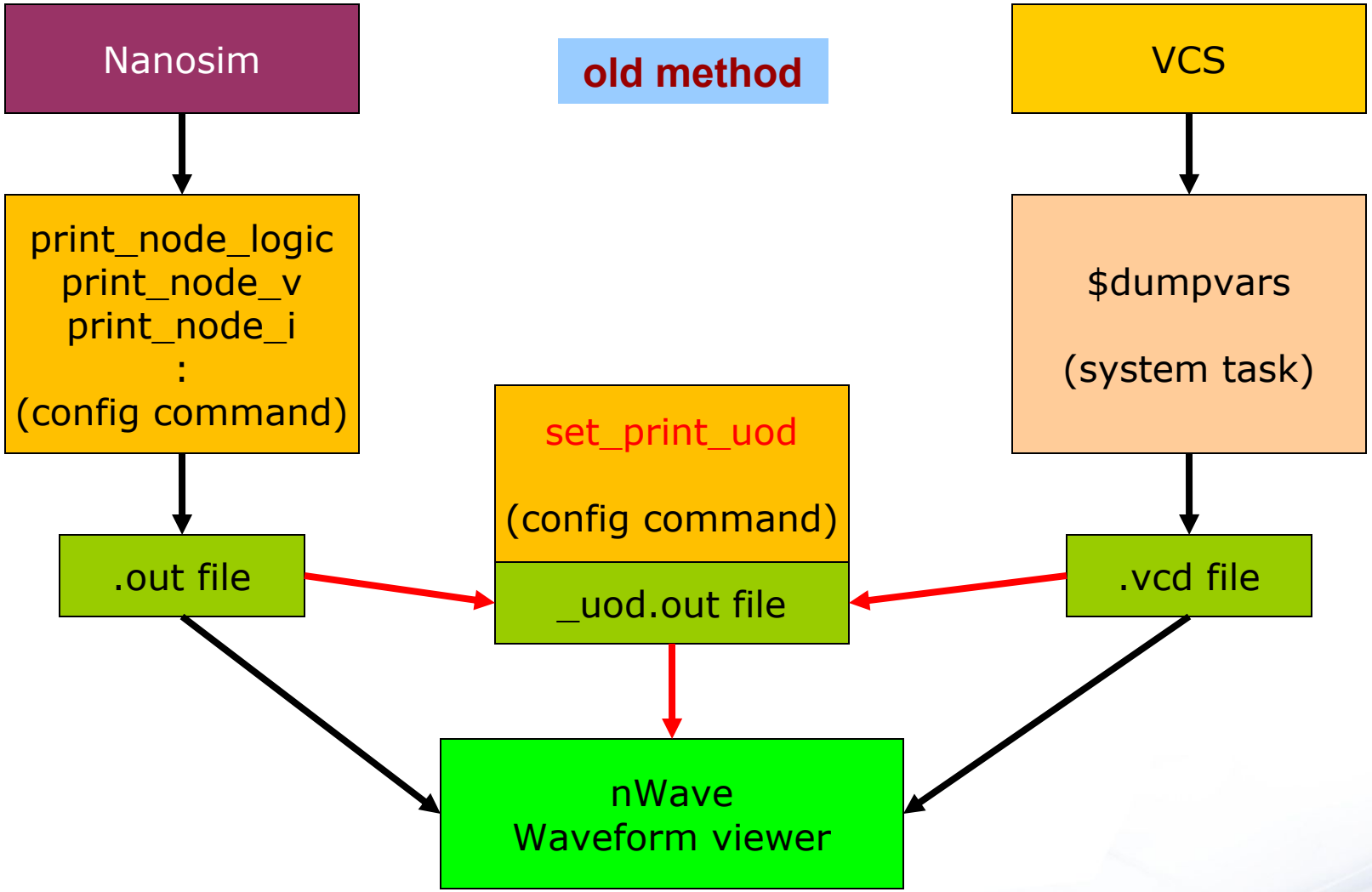
Put all of your SPICE netlists in Nanosim

```
choose nanosim –nspi umc018.cdl rom.spi –c config –o runresult ;
```

Nanosim Config file : config

```
use_sim_case
print_node_v *
print_node_i  VDD
set_print_uod uod=all
set_sim_eou sim=3 model=3
```

Generating a Unified Output



Generating a Unified Output

set_print_uod

Combine Nanosim .out and VCS .vcd to a new waveform file , its name is *_uod.out (prefix is same as Nanosim .out)

Usage :

set_print_uod [out=uod | all]

(only 'set_print_uod' is same as 'set_print_uod out=uod')

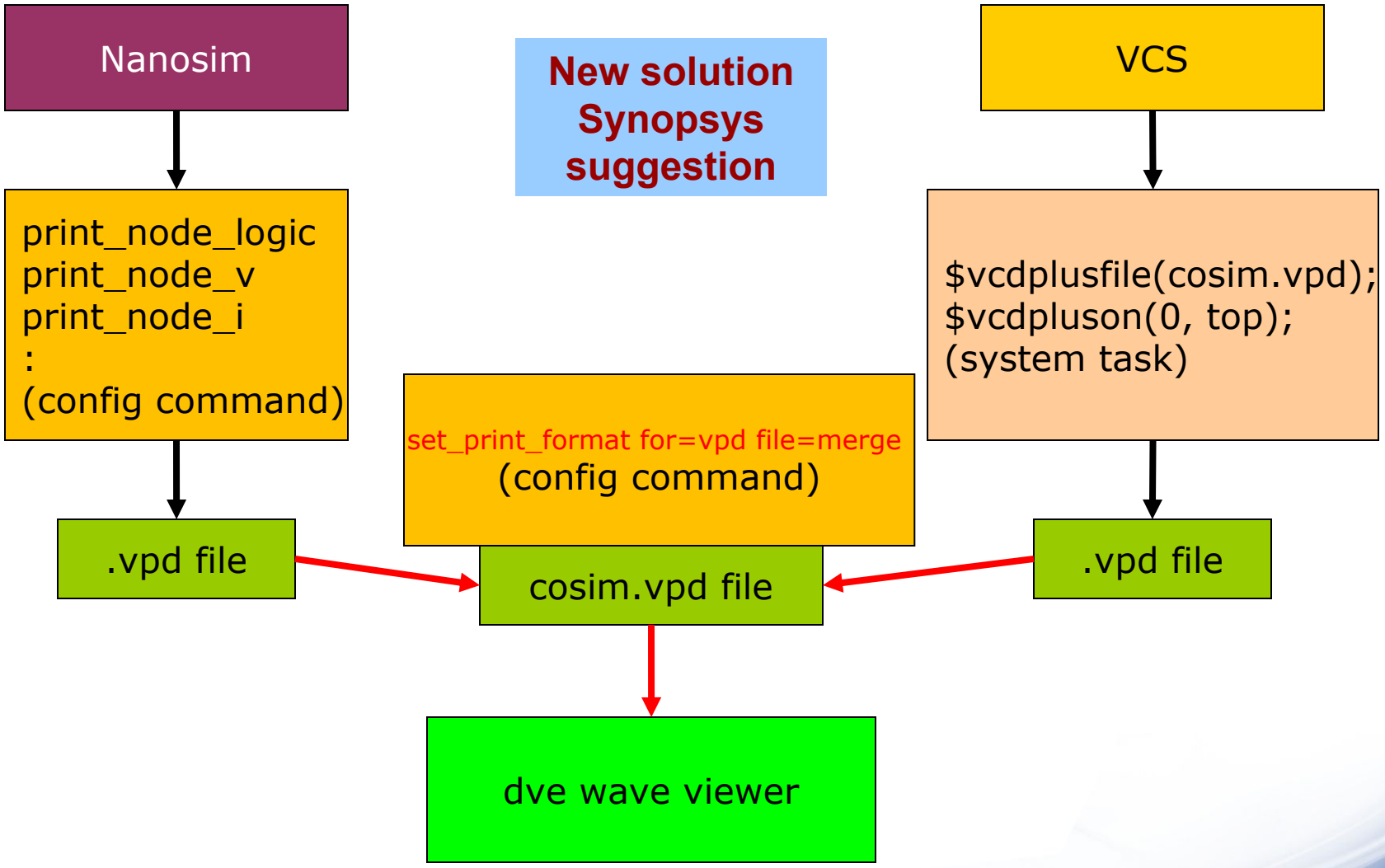
(all 为保留结合之前的out file及vcd file)

```
; this is Nanosim Config file
print_node_v level=0 *
Print_node_logic *

set_print_uod
```

UOD Limitations

- Because UOD files are created through post processing of digital and analog outputs at the end of the simulation, they cannot be generated in interactive mode.
- Viewing waveforms during simulation (marching waveforms) is not possible using UOD files, because the file is not generated until after the completion of the simulation.
- UOD files are in ASCII format, which have a much larger size than a compact binary format such as VPD.

Simulation Results



Old and New Mixed-Signal Commands

Old Mixed-Signal Commands (to be phased out after 2009.06)	New Mixed-Signal Commands (starting from 2009.06)
choose	**choose**
set bus_format	**bus_format**
N/A	**a2d** (replaces NanoSim config command `"set_node_thresh"` and the use of HSIM `".hsimvcs_intfparam"` command for a2d interface elements)
set interface_opt	**d2a** (also replaces the use of HSIM `".hsimvcs_intfparam"` command for d2a interface nets)

Old and New Mixed-Signal Commands

Old Mixed-Signal Commands (to be phased out after 2009.06)	New Mixed-Signal Commands (starting from 2009.06)
N/A	**insert_cell** (new in 2009.06)
set irmap (obsolete, replaced by XMR system tasks in 2009.06)	N/A **rmap_file**
set mview_vlog_noportswap	**mview_vlog_noport_swap**
set opt_shadowfile	**optimize_shadowfile**
set print_thru_net	**no_thru_net_opt**
N/A	**param_pass** (new in 2009.06)
set remove_interface	**remove_d2a**
set res_by_node	**rmap_by_node**
set spice_port_order_as_vlog	**spice_port_order_as_vlog**
set wrapper_dir	**shadow_file_dir**
spice_top	**spice_top**
use_spice	**use_spice**
use_verilog	**use_verilog**
use_vhdl	**use_vhdl**

Terminology

- View

  A Verilog module, Spice subckt representation of a block.

  Verilog-view， SPICE-view

- Multiple-view block

  A block that has two or more “views” available during simulation.

- Parent block

  The module or subckt that contains the instance of a specific block.

- Verilog wrapper module

  The corresponding Verilog-view to a Spice-view.

  For successful VCS compile, a Verilog-view must exist for any instantiation made in Verilog.

Verilog-top/SPICE-top Flows and Donut Configurations

- In a SPICE-top configuration, the spice_top command must be used in the mixed-signal simulation setup file, which by default is assumed to be called vcsAD.init.
- For Verilog-top or VHDL-top configuration, no specific command is required.
- 简单的区分是spice top还是verilog top，是看输入激励是用spice输入还是testbench输入，如果是后者则为verilog top。
- 目前我们是用spice top，后续重点介绍spice top相关，Verilog top后续有需求再添加。

Mixed-signal simulation commands

spice_top

Use the spice_top command to indicate that the top-level of the design is described in SPICE. This command is required for simulating a SPICE-top design.

spice_top [name=unique_name];

The unique_name option assigns a name (of your choice) to the top-level. Default is "snps_sptop"

```
//this is vcsAD.ini file
spice_top name=my_top;
```

Mixed-signal simulation commands

use_spice

Use the use_spice command to explicitly instruct the tool to use the SPICE view of a multi-view cell for a child cell under a Verilog parent.

use_spice -cell subcircuit_names [-C cfg_file_name];

use_spice instance_names [-C cfg_file_name];

如果用的是spice_top, 这个指令应该不太会用。

```
//this is vcsAD.ini file
use_spice -cell memory;
```

Mixed-signal simulation commands

use_verilog

Use the use_verilog command to explicitly instruct the tool to use the Verilog view of a multi-view cell for a child cell under a SPICE parent.

use_verilog [use_vcs] -module module_name;

use_verilog [use_vcs] instance_name;

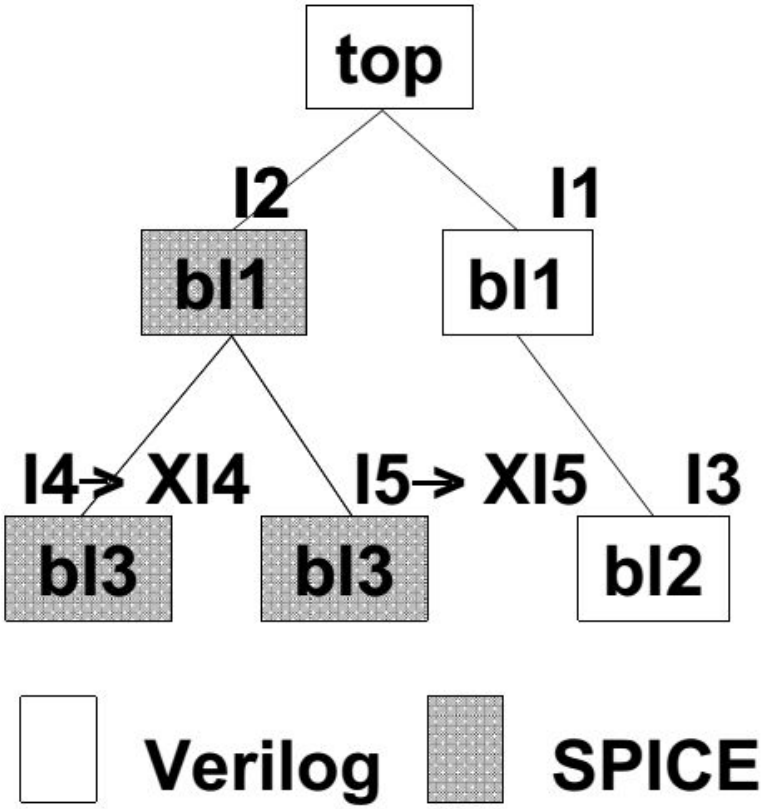
如果用的是spice_top, 这个指令对于verilog写的behavior model会比较有用。

```
//this is vcsAD.ini file
use_verilog -module memory;
//use_vcs -module memory;
```

Instance based view switching

Example:

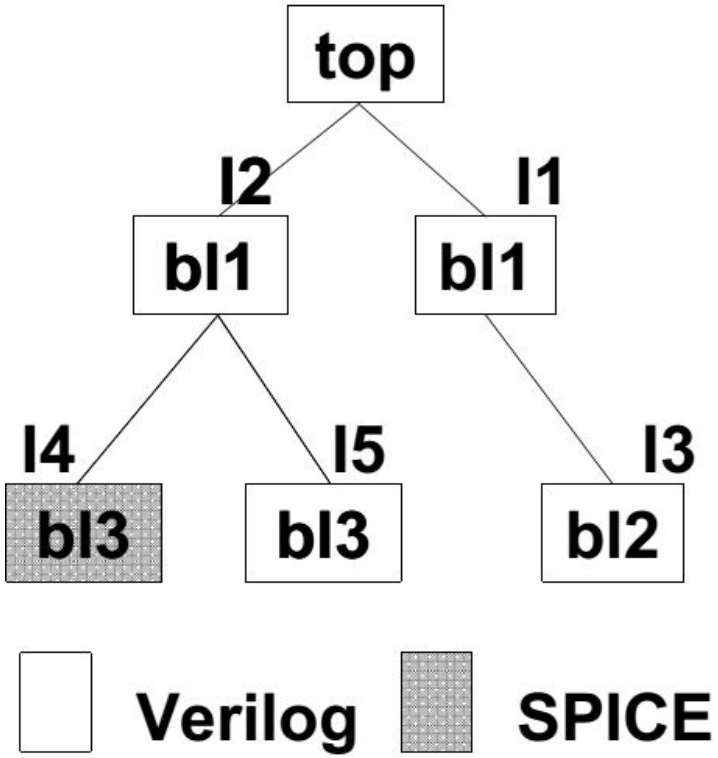
use_spice top.I2



Instance based view switching

Example:

use_spice top.I2.I4

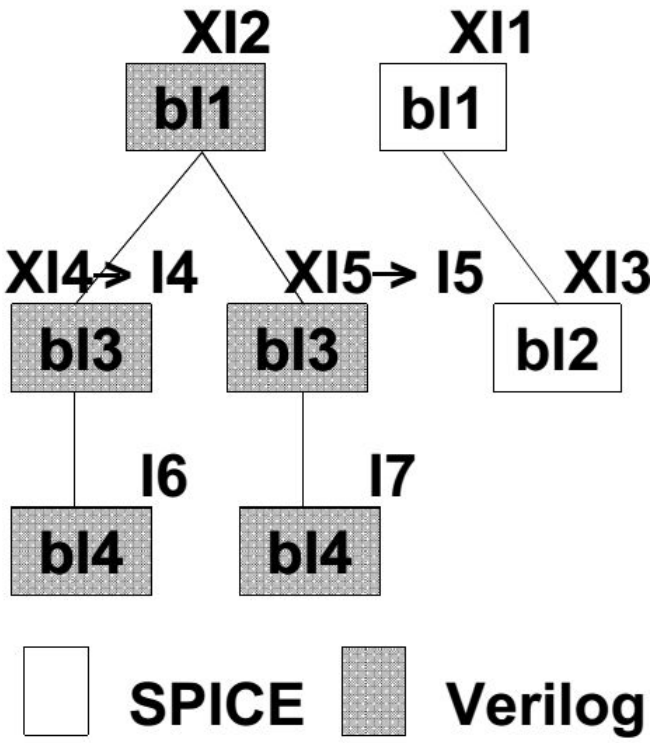


Instance based view switching

Example:

spice_top;

use_verilog XI2;

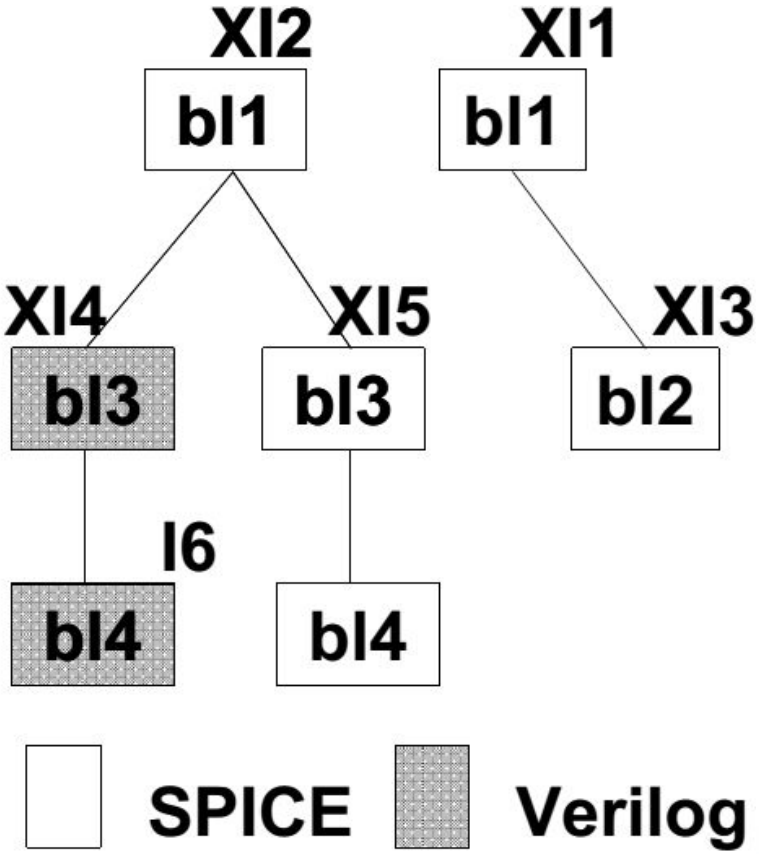


Instance based view switching

Example:

spice_top;

use_verilog XI2.XI4;



XMR (Cross Module Referencing) Across Analog-Digital Boundary

1.Logic XMR Access to Analog Nodes

EX1:Assigning the Logic Value Corresponding to the Voltage of an Analog Node to a Verilog Wire.

```
assign verilog_wire = top.i1.i2.x1.clk;
```

EX2:Assigning the Logic Value Corresponding to the Voltage of an Analog Node to a Verilog Register.

```
initial begin
...
verilog_reg = top.i1.i2.x1.strb;
...
end
```

EX3:Assigning Logic Values of a Verilog Register to an internal Analog Node as Voltages.

```
reg rst_reg;
assign top.i1.i2.x1.rst = rst_reg;
initial begin
...
rst_reg = 1'b0;
#5 rst_reg = 1'b1;
...
end
```

XMR (Cross Module Referencing) Across Analog-Digital Boundary

2.Real XMR Access to Analog Nodes

$snps_force_volt()

This system task allows Verilog to force a voltage to any analog node, even one connected to an ideal voltage source. In such a case this system task must override the ideal voltage source.

```
$snps_force_volt (top.i1.spcell.n1, 3.3);
```

$snps_release_volt()

This system task removes the voltage source applied by a previous $snps_force_volt task, and from that point on, allows the analog node to assume voltages determined by the analog circuit.

```
$snps_release_volt (top.i1.spcell.n1);
```

$snps_get_volt()

This is a Verilog function that allows sampling of voltage values for internal analog nodes.

```
real_var = $snps_get_volt(top.i1.spcell.n2);
 if($snps_get_volt(top.i1.i2.sp1_node > 2.5)
…
 else
…
end
```

Mixed-signal simulation commands

bus_format

Use the bus_format command to specify the bus format used in the SPICE netlist.

bus_format open_char %d close_char;

char support: { } < > [ ] _

```
//this is vcsAD.ini file
bus_format <%d>;
```

```
*SPICE subckt
.subckt addr4 a<3> a<2> a<1> a<0>
+b<3> b<2> b<1> b<0> cin
+s<3> s<2> s<1> s<0> cout
.ends
```

Mixed-signal simulation commands

a2d

Use this command to control all aspects of the A2D interfaces in the NS/HSIM/XA-VCS and NS/HSIM/XA-VCS-MX solutions. If this command is not specified, then by default all a2d events will be triggered at 50% of the local VDD.

a2d [loth=lo_thrsh[V | %]] [hith=hi_thrsh[V | %]] [hiz_off | hiz_on] <cell=cell_name port=port_name | node=hier_name>

```
//this is vcsAD.ini file
//The following example sets the a2d low and high thresholds for node
//"top.i1.ctl"  to 0.2V and 1.7V respectively.
a2d loth=0.2 hith=1.7 node=top.i1.ctl;
//a2d loth=20% hith=80% port=*;
```

Mixed-signal simulation commands

d2a

Use this command to control all aspects of the D2A interfaces in the NS/HSIM/XA-VCS and NS/HSIM/XA-VCS-MX solutions.

d2a [powernet][rf_time=slope_time] | [rise_time=rise_time]

[fall_time=fall_time][delay=delay_time][x2v=0|1|2|3|4]
[hiv=high_voltage lov=low_voltage] <cell=cell_name port=port_name | node=hier_name>

```
//this is vcsAD.ini file
d2a hiv=1.8 lov=0.1 node=top.i1.ctl;
d2a powernet hiv=3 lov=0 node=top.vdd;
d2a powernet hiv=3 lov=0 node=top.vss;
```

Mixed-signal simulation commands

insert_cell

This command allows the insertion of a 2-port SPICE netlist on the analog side of an interface net.

insert_cell [model= ADFMI_model_name [param=parameter_list]] [subckt= subcircuit_name apin=port_name dpin=port_name] <cell=cell_name port=port_name | node=hier_name>subckt= subcircuit_name apin=port_name dpin=port_name

```
//this is vcsAD.ini file
//analog和digital界面的地方插入2port ckt, 加入电路进行电压逻辑转换
//.subckt multiplier in out
//...
//.ends
//At the interface net "top.rst" , the following command must be used:
insert_cell subckt=multiplier apin=in dpin=out node=top.rst;
```

Mixed-signal simulation commands

optimize_shadowfile

Use the optimize_shadowfile command to optimize the shadow Verilog hierarchy generated for SPICE blocks.

optimize_shadowfile;

save compile time

Note:
Using this command prevents VCS from probing inside the SPICE hierarchy; this can prevent optimization of D2D through-nets, potentially increasing the number of interface nodes.

Mixed-signal simulation commands

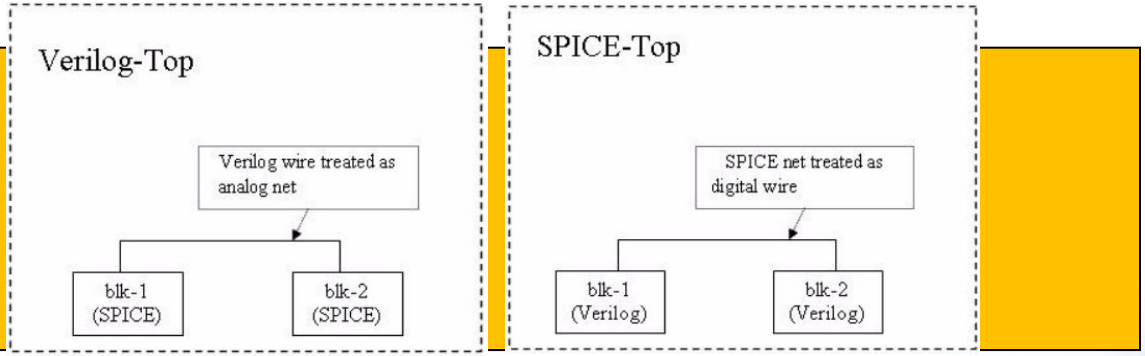
print_thru_net

Use the print_thru_net command to force NanoSim/HSIM/XA to instantiate a dummy A/D or D/A converter at the given mixed-net to make an image of through-nets visible in the other domain.

```
print_thru_net a2a|d2d|all;
```

When the print_thru_net command is added, extra A/D or D/A converters are inserted, which could potentially result in a slower simulation.

```
//this is vcsAD.ini file
print_thru_net all;
```



Mixed-signal simulation commands

rmap_file

Use the rmap_file command to specify the resistance map file or the path to where the resistance map file is located.

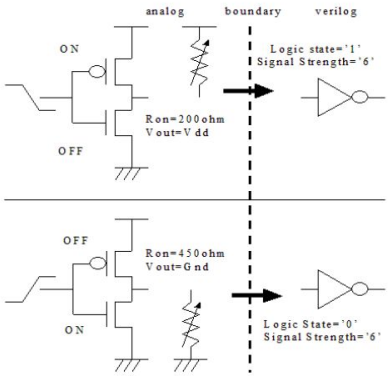
rmap_file resistance_map_file_name [options];

The rmapAD.init default resistance map file is available in the directory

/nanosim/amd64/ns/interface/vcsace/rmapAD.init

- Level 7: supply0, supply1
- Level 6: strong0, strong1 (default)
- Level 5: pull0, pull1
- Level 4: large, large
- Level 3: weak0, weak1
- Level 2: medium, medium
- Level 1: small, small
- Level 0: highz0, highz1

```
resistance_map 90000.2-1e32 0 ;
resistance_map 70000.2-90000.1 1 ;
resistance_map 50000.2-70000.1 2 ;
resistance_map 7000.2-50000.1 3 ;
resistance_map 6000.2-7000.1 4 ;
resistance_map 1000.2-6000.1 5 ;
resistance_map 1.2-1000.1 6 ;
resistance_map 0-1.1 7 ;
```




```
//this is vcsAD.ini file
//Applying user-defined resistance map to the specified port in the
//specified instance
rmap_file r3.map inst=top.i1.abc port=Z;
```

Summary of Mixed-signal Simulation Commands

Command	Use	Definition
`choose analog_engine_name` *`command-line options;`*	required	Specifies NanoSim, HSIM, or XA as the analog engine and its command-line options.
`bus_format` *`open_char %d close_char;`*	optional	Specifies bus format used in the SPICE netlist.
`set interface_opt [`*`option`*`] node=`*`node_name(s);`*	optional	Models the digital-to-analog interface.
`mview_vlog_noportswap;`	optional	Applies to multi-view Verilog blocks instantiated under SPICE. This command forces the use of the original SPICE port order instead of the default Verilog port order.
`optimize_shadowfile;`	optional	Optimizes Verilog dummy modules and hierarchy under the SPICE view.
`no_thru_net_opt [a2a | d2d | all];`	optional	Instructs NanoSim/HSIM/XA to instantiate dummy A/D or D/A inverters to generate the digital image of an a2a through-net and/or the analog image of a d2d through-net.
`remove_d2a [dc=`*`dc_voltage_source_value`*`] node=`*`node_name;`*	optional	Specifies a d2a mixed-net to be removed from the Verilog and transistor-level boundary.
`rmap_by_node r=` *`resistance_value`* `node=`*`node_name(s);`*	optional	Applies the resistance value to the interface resistor connected to the node at the digital-to-analog interface.

Summary of Mixed-signal Simulation Commands

Command	Use	Definition
`rmap_file` *`resistance_map_file_name`* OR `rmap_file` *`resistance_map_file_path_name`* `[option];`	optional	Specifies the resistance map file to be used.
`spice_port_order_as_vlog;`	optional	Applies to multi-view SPICE blocks instantiated under Verilog. It forces Discovery-AMS to apply the port order in the Verilog view of the cell to the SPICE instantiation.
`shadow_file_dir` *`directory_path`*`;`	optional	Specifies the directory where mixed-signal intermediate files will be stored.
`spice_top [name=`*`unique_name`*`];`	optional	Specifies that the top level of the design is SPICE
`use_spice -cell` *`subcircuit_names -C cfg_file_name;`* OR `use_spice` *`instance_names -C cfg_file_name;`*	optional	Specifies cells or instances to be modeled and simulated in analog. The "-C" option can only be used in NS-VCS and NS-VCS-MX.
`use_verilog -module` *`module_name;`* OR `use_verilog` *`instance_name;`*	optional	Specifies cells or instances to be modeled and simulated in digital.

Cosim sample with generated a single output file in VPD

TOP (testbech.v)

addr4 dut (.a(a[3:0]), .b(b[3:0]), .cin(cin), .s(s[3:0]), .cout(cout));

addr4

adder.v

```
module addr4 (a, b, cin, s, cout);
input [3:0] a, b;
input cin;
output [3:0] s;
output cout;
assign {cout, s} = a + b + cin;
endmodule
```

addr4.spi

```
.subckt addr4 a_3 a_2 a_1 a_0
+b_3 b_2 b_1 b_0 cin
+s_3 s_2 s_1 s_0 cout
x1 a_3 b_3 cout s_3 n1 addr
x2 a_2 b_2 n1 s_2 n2 addr
x3 a_1 b_1 n2 s_1 n3 addr
x4 a_0 b_0 n3 s_0 cin addr
.ends
```

Cosim sample with generated a single output file in VPD

```
;This is cfg file
print_node_v *
print_node_logic *
set_print_format for=vpd file=merge
```

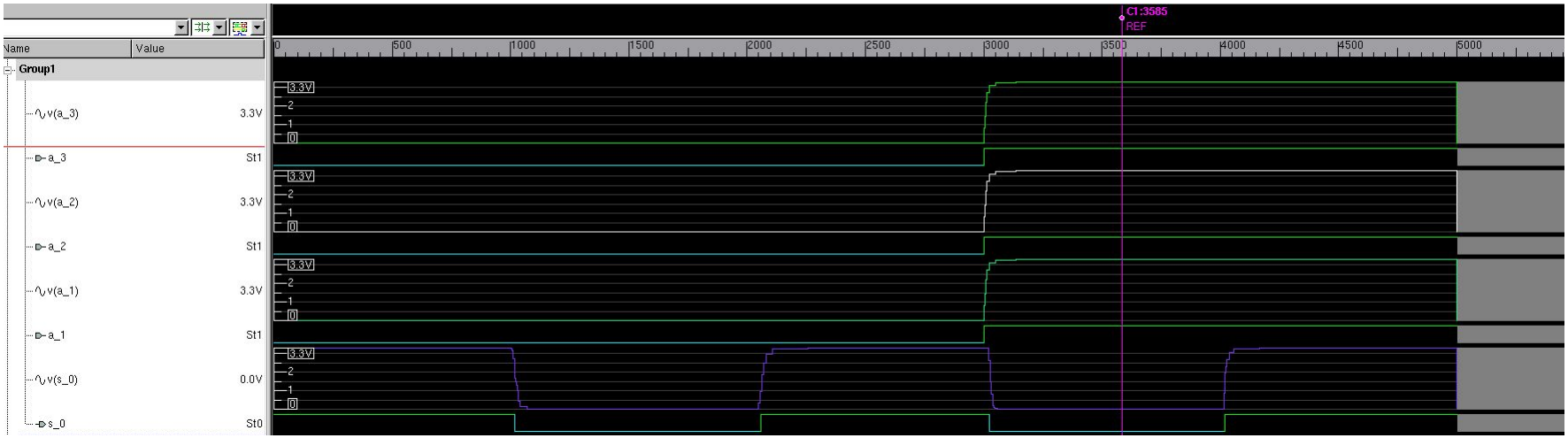
```
#This is run file
vcs -full64 -debug_pp testbench.v adder.v -l comp.log +ad
simv -l sim.log
```

```
//this is vcsAD.ini file
use_spice -cell addr4;
choose nanosim -n addr4.spi -C cfg ;
bus_format _%d;
rmap_file resis_map;
```

```
#this is run_dve file
dve -full64 -vpd ./addr4.vpd &
```

Cosim sample with generated a single output file in VPD

Analog and digital signal message saved in a same vpd file.



Spice top 演示-->

Cosim experience

- 发现加参数产生VPDFILE的方式

@simv -l sim.log +vpdfile+$(RUNW_PATX).vpd;

在presim hierarchy netlist仿真时， 通过spice外灌的信号，无法在dve中查看，都出现hiz状态。如果改成code中dump是ok的。如下:

$vcdplusfile("my.vpd");

- 产生一个DP_VPD.v，内部只是单纯的vcdplusfile宣告，需要在spice中调用一下，XDP_VPD DP_VPD

这样就不需要在其它的behavior module中宣告.效率会比较高，

也不会导致其它的behavior module维护混乱。

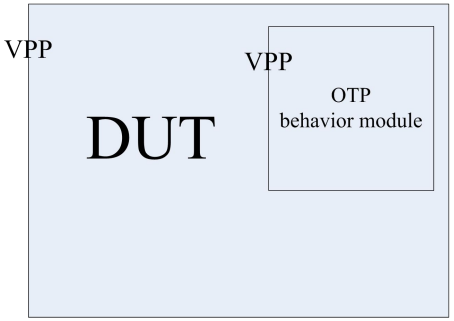
```
module DP_VPD(
              );

initial
begin
          $vcdplusfile("DUT.vpd");
          $vcdpluson(0, DP_VPD);
//        $vcdpluson(0, E0064K10FN180SK02);

end


endmodule
```

Cosim experience

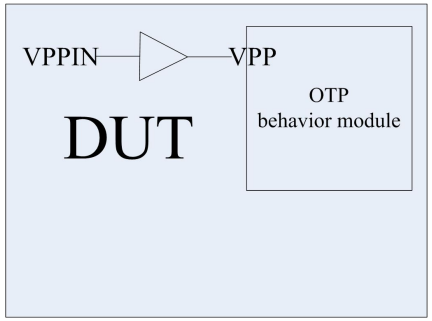
- 宣告为spice top的结构，对于内部的verilog block的信号，如果未经过任何spice device，直接在spice中强灌信号将无法识别。比如底层的otp的behavior model中的VPP，直接在SPI中强灌无法识别，有两种方法解决，一个是将VPP拉到顶层，一个是内部加spice的buffer。

- cosim中的use_verilog use_spice的选项，会因为pin对应关系，转换会不太顺。须保证Pin脚能够一一对应起来。

VPP VPP 0 PWL(0 LOW 100u HIGH)
XDUT VPP PAX PBX ... DUT



VPP XDUT.VPPIN 0 PWL(0 LOW 100u HIGH)
XDUT PAX PBX ... DUT



%make run_1
%make dve_1

Makefile ENV

```
#==================Sim pattern=============================
PATTERN        = TX1 F_ksleep ISB F_oisb
RUN_PATX       = $(word $(subst run_,,$(@)), $(PATTERN))
DVE_PATX       = $(word $(subst dve_,,$(@)), $(PATTERN))
RUN_PAT        = $(subst run_,,$@)
WV_PAT         = $(subst verdi_,,$@)
MAINTEST       = maintest.v
run_%:
    @sed -f sedver $(TB_DIR)$(RUN_PATX).ver > romdat.ver
    @cp $(TB_DIR)$(RUN_PATX).v $(MAINTEST)
    @vcs -f vfile.f  -v $(LIB_DIR)$(LIB1) -v $(LIB_DIR)$(LIB2) -R -full64 -l vcs_run.log \
    -debug_pp \
    +vpdfile+../Wave/$(RUN_PATX).vpd

dve_%:
    @dve -full64 -vpd ../Wave/$(DVE_PATX).vpd &
```

Thank you!