

HSPICE® Integration to Cadence® Virtuoso® Analog Design Environment User Guide

Version E-2010.12, December 2010

SYNOPSYS®

Copyright Notice and Proprietary Information

Copyright © 2010 Synopsys, Inc. All rights reserved. This software and documentation contain confidential and proprietary information that is the property of Synopsys, Inc. The software and documentation are furnished under a license agreement and may be used or copied only in accordance with the terms of the license agreement. No part of the software and documentation may be reproduced, transmitted, or translated, in any form or by any means, electronic, mechanical, manual, optical, or otherwise, without prior written permission of Synopsys, Inc., or as expressly provided by the license agreement.

Right to Copy Documentation

The license agreement with Synopsys permits licensee to make copies of the documentation for its internal use only. Each copy shall include all copyrights, trademarks, service marks, and proprietary rights notices, if any. Licensee must assign sequential numbers to all copies. These copies shall contain the following legend on the cover page:

“This document is duplicated with the permission of Synopsys, Inc., for the exclusive use of _____ and its employees. This is copy number _____.”

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Registered Trademarks (®)

Synopsys, AMPS, Astro, Behavior Extracting Synthesis Technology, Cadabra, CATS, Certify, CHIPit, CoMET, Design Compiler, DesignWare, Formality, Galaxy Custom Designer, HAPS, HapsTrak, HDL Analyst, HSIM, HSPICE, Identify, Leda, MAST, METeor, ModelTools, NanoSim, OpenVera, PathMill, Physical Compiler, PrimeTime, SCOPE, Simply Better Results, SiVL, SNUG, SolvNet, Syndicated, Synplicity, the Synplicity logo, Synplify, Synplify Pro, Synthesis Constraints Optimization Environment, TetraMAX, UMRBus, VCS, Vera, and YELDDirector are registered trademarks of Synopsys, Inc.

Trademarks (™)

AFGen, Apollo, Astro-Rail, Astro-Xtalk, Aurora, AvanWaves, BEST, Columbia, Columbia-CE, Confirma, Cosmos, CosmosLE, CosmosScope, CRITIC, CustomExplorer, CustomSim, DC Expert, DC Professional, DC Ultra, Design Analyzer, Design Vision, DesignerHDL, DesignPower, DFTMAX, Direct Silicon Access, Discovery, Eclipse, Encore, EPIC, Galaxy, HANEX, HDL Compiler, Hercules, Hierarchical Optimization Technology, High-performance ASIC Prototyping System, HSIM^{plus}, i-Virtual Stepper, IICE, in-Sync, iN-Tandem, Jupiter, Jupiter-DP, JupiterXT, JupiterXT-ASIC, Liberty, Libra-Passport, Library Compiler, Magellan, Mars, Mars-Rail, Mars-Xtalk, Milkyway, ModelSource, Module Compiler, MultiPoint, Physical Analyst, Planet, Planet-PL, Polaris, Power Compiler, Raphael, Saturn, Scirocco, Scirocco-i, Star-RCXT, Star-SimXT, StarRC, System Compiler, System Designer, Taurus, TotalRecall, TSUPREM-4, VCS Express, VCSi, VHDL Compiler, VirSim, and VMC are trademarks of Synopsys, Inc.

Service Marks (sm)

MAP-in, SVP Café, and TAP-in are service marks of Synopsys, Inc.

SystemC is a trademark of the Open SystemC Initiative and is used under license.

ARM and AMBA are registered trademarks of ARM Limited.

Saber is a registered trademark of SabreMark Limited Partnership and is used under license.

All other product or company names may be trademarks of their respective owners.

Contents

Audience	9
Inside this User Guide	10
The HSPICE Documentation Set	12
Related Publications	13
Conventions	14
Customer Support	15

1. Quick-Start Tutorial	17
Task 1: Start Up the Tool	19
Task 2: Configure Your Design for Netlisting	23
Task 3: Review Available Analyses Windows	26
Task 4: Set Up Outputs and Explore the Plotting Assistant	32
Task 5: Load a State	36
Task 6: Create a Netlist File	38
Task 7: Run a Simulation and Use the Plotting Assistant	40
Task 8: Loading and Running a Verilog-A Example	44
Task 9: Set Up and Run a Monte Carlo Simulation	46
Task 10: Run a Corner Analysis	54
Task 11: Run an RF Analysis	61
Task 12: Run an Optimization Analysis	68

2. Updating Libraries and Tool Filter	75
Library Conversion Script	75
Running the Library Update Utility	75
HSPICE and analogLib	78
Splitting analogLib into 3 Output Libraries	80
HSPICE Components Added or Converted	80
Adding the HSPICE Simulator to the Tool Filter	92

Contents

3. Environment Setup	93
Environment Console	93
Menus — Environment Console	94
Verilog-A Support	106
Using the HSPICE Verilog-A Compiler by Default	107
4. Analysis Setup and Design Variables	109
HSPICE Integration Analyses	109
Multi-Dimensional Sweeps	110
Transient Analysis	111
Transient Noise Analysis	113
DC Analysis	117
AC Analysis	119
Operating Point Analysis	121
Noise Analysis	122
Discrete-Fourier Transform Analysis	124
Linear Network Parameter Analysis	127
.NOISE as Part of a .LIN Analysis	128
DCMatch Analysis	129
ACMatch Analysis	131
Loop Stability Analysis	132
Setting Up and Running a Simple LSTB Analysis	133
Running LSTB Monte Carlo, Corner, and Parametric Analyses	134
Pole/Zero Analysis	146
Simulating and Plotting P/Z Results	147
Other Features Supported in a PZ Analysis	148
Design Variables	148
5. Saving-Plotting Outputs	151
Setup	152
Delete	155
To Be Saved	155

To Be Plotted	155
Save Options	156
Saving and Restoring a List of Outputs	159
Saving a List of Outputs	159
Restoring a Saved List of Outputs	160
<hr/>	
6. Running Simulations and Using Control Options	161
Simulation Menu Overview	161
Netlist and Run.	162
Run.	163
Stop	163
Options.	164
Netlist	167
Output Logs	168
Convergence Aids	169
<hr/>	
7. Printing and Plotting Results	173
Results Menu Overview	173
HSPICE Plotting Assistant	174
Results > Plot Outputs	184
Results > Direct Plot	185
Results > Print	188
Results > Annotate.	189
Results > Save, Select, Delete.	194
Results > Printing/Plotting Options	196
<hr/>	
8. Monte Carlo in the HSPICE Integration	199
Accessing and Setting Up Monte Carlo Analysis.	200
Configuring the Setup Tab to Run a Monte Carlo Simulation	200
Setting Up Multiprocessing of Monte Carlo Runs	202
Using the Options Tab of the HSPICE Monte Carlo Analysis Form.	203
Setting up Measurements, Post-Plotting, and Analyzing the Simulation Results Using the Outputs Tab	205
Measurements Pane	207
Simulation Pane	207

Contents

Outputs Pane	208
Measurement Variation Statistics.	209
Other Features Supported in the Monte Carlo Analysis Form.	210
<hr/>	
9. Corners Analysis	213
Launching the Corner Analysis Tool	214
HSPICE Corner Analysis Window	214
Setting Up a Corners Analysis.	217
Specifying Design Variables.	217
Setting Up Measurements for Corners	224
Loading Measurements for Setup	225
Running Corner Simulations and Using the Outputs Tab	227
Simulation	227
Plotting Outputs	230
Getting Measurements, Plotting and Printing.	231
Other Features Supported in HSPICE Corner Analysis	233
<hr/>	
10. HSPICE RF Analysis	235
Accessing and Setting Up an HSPICE RF Analysis	235
Setting Up the Analysis Tab to Run an RF Analysis	236
Using the Options Tab of the HSPICE RF Analysis Form	239
Available Options	240
Using the Outputs Tab	243
Running and Post-Processing an RF Simulation.	245
Post-Simulation Plotting and Analysis	246
Using the Measure Utility on RF Simulations.	246
Other Features Supported in the HSPICE RF Analysis Form	247
Current RF Analysis Feature Limitations	248
<hr/>	
11. HSPICE Optimization Analysis	249
Accessing and Setting Up an HSPICE Optimization Analysis.	249
Using the Setup Tab.	249
Selecting the Model	250

Setting Optimization Variables	251
Setting Up Measurements	251
Using the Analysis Tab.	252
Choosing Analysis and Optimization Type	253
Other Features Supported in the HSPICE Optimization Form.	254
Current Optimization Analysis Feature Limitation	255

12. HSPICE Measurement Utility	257
Measurement Utility Overview	257
Opening and Using the HSPICE Measurement Utility	258
Trig/Targ Setup	261
General Functions Setup	262
Find/When Setup	263
Equation Setup	264
Error Function Setup	264
Using the Special-Meas. Tab	265
Optimization Syntax—Goal Setup	268
Report View Setup	269

13. HSPICE Reliability Analysis (MOSRA)	271
MOSRA Overview	271
Launching the HSPICE Reliability Analysis Window	272
Main Menus	275
Tabs and Pages	275
Setting Up Reliability Analysis	277
MOSRA Command Setup	278
Setting Up MOSRA Models	284
Setting up MOSRA-Related Options	288
Setting up the Outputs Page	288
Simulation Control	289
Using the Outputs Plot Control Section	289
Plotting and Printing Measurement Variable Results	294
Displaying Reliability Analysis Degradation Data	298
Other Features Supported in the HSPICE Reliability Analysis Window	301

Contents

14. HSPICE Violation Check (.BIASCHK)	303
Invoking and Setting up Violation Check	303
GUI Controls	305
Example Monitoring Setups	310
Setting Up .BIASCHK Commands and Options	313
Netlisting and Running .BIASCHK Simulations	314
Displaying/Printing Violation Check Results	314
HSPICE Violation Details Form	315
15. Distributed Mode—Monte Carlo/Corner Analyses	321
HSPICE Distributed Jobs Mode	321
HSPICE Monte Carlo Distributed Simulation	323
HSPICE Corner Analysis Distributed Simulation	328
A. Adding a CustomExplorer™ Menu to the HSPICE Integration	331
Updating the HSPICE.menus File	331
Sample HSPICE.menus Files for Versions 51 and 61	334
Version 51	335
Version 61	349
B. OCEAN API Functions for HSPICE Monte Carlo Analysis	365
Using the HSPICE-Provided OCEAN API for HSPICE Monte Carlo Runs	365
C. OCEAN API Functions for HSPICE RF Analysis	373
Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis	373
1. snpsAnalysis	373
2 snpsProbe	386
D. OCEAN API Functions for HSPICE Corner Analysis	391
Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis	391

E. OCEAN API Functions for HSPICE Optimization Analysis	407
Using the HSPICE-Provided OCEAN API for HSPICE Optimization Analysis.	407
F. OCEAN API Functions for the HSPICE Measure Utility	413
Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility	413
G. OCEAN API Functions for HSPICE MOSRA Analysis	425
Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility	425
Index	435

Contents

About this User Guide

This User Guide supports the Synopsys HSPICE integration to the Cadence® Virtuoso® Analog Design Environment. The content of this user guide provides information regarding the HSPICE integration and analysis/simulation setup.

For information on the Cadence tool, refer to the Virtuoso® Analog Design Environment User Guide documentation, available through that licensed tool.

All references to the Cadence® Virtuoso® Analog Design Environment platform are noted in this user guide as the “Environment.” The graphical interface for the HSPICE integration displayed in this manual is used with permission and is based on the version 6.14 of the native tool.

Audience

This user guide is written for the designers who use HSPICE for analog and/or mixed signal design. With this Synopsys HSPICE integration interface, users get a full-featured simulation environment that facilitates the efficient use of HSPICE in creating and verifying the performance of analog and mixed signal designs. The HSPICE integration provides an upgraded interface to the many long-standing features of HSPICE that are not supported in even the latest versions of the Virtuoso® Analog Design Environment.

Inside this User Guide

This guide contains the chapters described below. For descriptions of the other manuals in the HSPICE documentation set, see the next section, [The HSPICE Documentation Set](#).

Chapter	Description
Chapter 1, Quick-Start Tutorial	Presents a a quick-start tutorial to get you up and running with the Synopsys HSPICE integration to the Cadence® Virtuoso® Analog Design Environment.
Chapter 2, Updating Libraries and Tool Filter	Describes the library update utility for making the HSPICE simInfo function correctly and how to make HSPICE the default simulator.
Chapter 3, Environment Setup	Describes the setup steps to configure your environment to do HSPICE simulation and analysis on your design.
Chapter 4, Analysis Setup and Design Variables	Describes Analysis forms and capabilities available through the HSPICE integration interface and introduces the editor for design variables.
Chapter 5, Saving-Plotting Outputs	Describes the Outputs menu selections from the Setting Outputs form available to the HSPICE integration.
Chapter 6, Running Simulations and Using Control Options	Describes the simulation procedures according to the Simulations menu and details the HSPICE control options available through the custom HSPICE Analog Options forms.
Chapter 7, Printing and Plotting Results	Describes HSPICE integration post-processing actions using the Result menu selections and introduces the HSPICE Plotting Assistant.
Chapter 8, Monte Carlo in the HSPICE Integration	Describes the Monte Carlo Analysis capabilities available through the HSPICE integration interface.
Chapter 9, Corners Analysis	Describes the corners analysis capability in the HSPICE integration to the Cadence® Virtuoso® Analog Design Environment.
Chapter 10, HSPICE RF Analysis	Describes HSPICE RF capability in the HSPICE integration to the Cadence® Virtuoso® Analog Design Environment.

Chapter	Description
Chapter 11, HSPICE Optimization Analysis	Describes HSPICE Optimization capability in the HSPICE integration to the Cadence® Virtuoso® Analog Design Environment.
Chapter 12, HSPICE Measurement Utility	Describes the HSPICE Measurement Utility in the HSPICE integration to the Cadence® Virtuoso® Analog Design Environment.
Chapter 13, HSPICE Reliability Analysis (MOSRA)	Describes the HSPICE Measurement Utility in the HSPICE integration to the Cadence® Virtuoso® Analog Design Environment.
Chapter 14, HSPICE Violation Check (.BIASCHK)	Describes the HSPICE simulation violation (bias check) capability in the HSPICE integration to the Cadence® Virtuoso® Analog Design Environment.
Chapter 15, Distributed Mode—Monte Carlo/Corner Analyses	Describes the HSPICE distributed jobs solution in the HSPICE integration to the Cadence® Virtuoso® Analog Design Environment.
Appendix A, Adding a CustomExplorer™ Menu to the HSPICE Integration	Discusses how to modify the HSPICE.menus file to add the Synopsys CustomExplorer wave view tool to the Cadence® Virtuoso® Analog Design Environment console menu bar.
Appendix B, OCEAN API Functions for HSPICE Monte Carlo Analysis	Describes HSPICE-defined OCEAN functions provided to run Monte Carlo analyses.
Appendix C, OCEAN API Functions for HSPICE RF Analysis	Describes HSPICE-defined OCEAN functions provided to run HSPICE RF analyses.
Appendix D, OCEAN API Functions for HSPICE Corner Analysis	Describes HSPICE-defined OCEAN functions provided to run HSPICE Corner analyses.
Appendix E, OCEAN API Functions for HSPICE Optimization Analysis	Describes HSPICE-defined OCEAN functions provided to run HSPICE Optimization analyses.

Chapter	Description
Appendix F, OCEAN API Functions for the HSPICE Measure Utility	Describes HSPICE-defined OCEAN functions provided to run the HSPICE Measure utility.
Appendix G, OCEAN API Functions for HSPICE MOSRA Analysis	Describes HSPICE-defined OCEAN functions provided to run the HSPICE MOSRA model reliability analysis.

The HSPICE Documentation Set

This manual is a part of the HSPICE documentation set, which includes the following manuals:

Manual	Description
HSPICE User Guide: Simulation and Analysis	Describes how to use HSPICE to simulate and analyze your circuit designs, and includes simulation applications. This is the main HSPICE user guide.
HSPICE User Guide: Signal Integrity	Describes how to use HSPICE to maintain signal integrity in your chip design.
HSPICE User Guide: RF Analysis	Describes how to use special set of analysis and design capabilities added to HSPICE to support RF and high-speed circuit design.
HSPICE Reference Manual: Commands and Control Options	Provides reference information for HSPICE and HSPICE RF commands and options.
HSPICE Reference Manual: Elements and Device Models	Describes standard models you can use when simulating your circuit designs in HSPICE, including passive devices, diodes, JFET and MESFET devices, and BJT devices.
HSPICE Reference Manual: MOSFET Models	Describes available MOSFET models you can use when simulating your circuit designs in HSPICE.

Manual	Description
AMS Discovery Simulation Interface Guide for HSPICE	Describes use of the Simulation Interface with other EDA tools for HSPICE.
AvanWaves User Guide	Describes the AvanWaves tool, which you can use to display waveforms generated during HSPICE circuit design simulation.

Searching Across the HSPICE Documentation Set

You can access the PDF format documentation from your install directory for the current release by entering `-docs` on the terminal command line when the HSPICE tool is open.

Synopsys includes an index with your HSPICE documentation that lets you search the entire HSPICE documentation set for a particular topic or keyword. In a single operation, you can instantly generate a list of hits that are hyper-linked to the occurrences of your search term. For information on how to perform searches across multiple PDF documents, see the HSPICE release notes.

Note: To use this feature, the HSPICE documentation files, the Index directory, and the `index.pdx` file must reside in the same directory. (This is the default installation for Synopsys documentation.) Also, Adobe Acrobat must be invoked as a standalone application rather than as a plug-in to your web browser.

You can also invoke HSPICE and HSPICE RF command help by entering `-help` on your terminal command line when the HSPICE tool is open. This opens a browser-based help system for fast navigation to commands and options used in HSPICE and the HSPICE RF flow.

Related Publications

For additional information about HSPICE see:

- The documentation set installed with the HSPICE software distribution
- The Release Notes for the HSPICE integration are available in the HSPICE release notes published with each release and service pack, and is available on SolvNet (see [Accessing SolvNet on page 15](#)).
- Documentation on the Web, which provides PDF documents and is available on SolvNet (see [Accessing SolvNet on page 15](#))
- The Cadence™ Virtuoso® Analog Design Environment User Guide documentation is available through that licensed tool.

Conventions

The following conventions are used in Synopsys documentation.

Convention	Description
<code>Courier</code>	Indicates command syntax.
<i>Italic</i>	Indicates a user-defined value, such as <i>object_name</i> .
Bold	Indicates user input—text you type verbatim—in syntax and examples. Bold indicates a GUI element
[]	Denotes optional parameters, such as: <code>write_file [-f filename]</code>
...	Indicates that parameters can be repeated as many times as necessary: <code>pin1 pin2 ... pinN</code>
	Indicates a choice among alternatives, such as <code>low medium high</code>
\	Indicates a continuation of a command line.

Convention	Description
/	Indicates levels of directory structure.
Edit > Copy	Indicates a path to a menu command, such as opening the Edit menu and choosing Copy.
Control-c	Indicates a keyboard combination, such as holding down the Control key and pressing c.

Customer Support

Customer support is available through SolvNet online customer support and through contacting the Synopsys Technical Support Center.

Accessing SolvNet

SolvNet includes an electronic knowledge base of technical articles and answers to frequently asked questions about Synopsys tools. SolvNet also gives you access to a wide range of Synopsys online services, which include downloading software, viewing Documentation on the Web, and entering a call to the Support Center.

To access SolvNet:

1. Go to the SolvNet Web page at <http://solvnet.synopsys.com>.
2. If prompted, enter your user name and password. (If you do not have a Synopsys user name and password, follow the instructions to register with SolvNet.)

If you need help using SolvNet, click Help on the SolvNet menu bar.

The link to any recorded training is

<https://solvnet.synopsys.com/trainingcenter/view.faces>

Access recent release update training by going to

https://solvnet.synopsys.com/search/advanced_search.faces

Contacting the Synopsys Technical Support Center

If you have problems, questions, or suggestions, you can contact the Synopsys Technical Support Center in the following ways:

- Open a call to your local support center from the Web by going to <http://solvnet.synopsys.com/EnterACall> (Synopsys user name and password required).
- Send an e-mail message to your local support center.
 - E-mail support_center@synopsys.com from within North America.
 - Find other local support center e-mail addresses at http://www.synopsys.com/support/support_ctr.
- Telephone your local support center.
 - Call (800) 245-8005 from within the continental United States.
 - Call (650) 584-4200 from Canada.
 - Find other local support center telephone numbers at http://www.synopsys.com/support/support_ctr.

Quick-Start Tutorial

This chapter provides a quick-start tutorial to get you up and running with the Synopsys HSPICE integration to the Cadence® Virtuoso® Analog Design Environment.

Note: All references to the Cadence® Virtuoso® Analog Design Environment are noted in this user guide as “the Environment.”

All required files for this quick-start tutorial are available at:

`/demo/hspice/aa_integ/`

Depending on which software version you have, select either PLL_Demo_51/ or PLL_Demo_61/. This chapter’s screen shots are based on the version 6.1.

Licensing

The only licenses required for running the HSPICE Integration to Virtuoso are:

- From Cadence, one license each for: ADE-L and OASIS (the OASIS license is **ONLY** needed when doing a netlist in HSPICE-ADE before invoking HSPICE to execute an operation in IC6xx or IC5xx).
- From Synopsys: one HSPICE license

This tutorial assumes that you have already installed the software according to the installation directions README file, which is available in:

`/global/apps3/hspice_release_version_date/interfaces`

The purpose of this chapter is to get you started quickly in setting up a design for simulation using the full-range HSPICE simulator in an analog/mixed signal simulation environment. This integration minimizes the simulator language expertise required since the interface includes GUI selections that are the equivalent of the command-line statements and control options.

The HSPICE integration also recognizes that many designers specify their stimulus by placing library components directly in their designs using the

Environment Schematic Editor. Different types of voltage and current sources (dc, pwl, sinusoidal, behavioral) are provided, as are basic passive load components such as resistors, capacitors, and inductors. Users may apply the primitive definitions of these components, or, if HSPICE supports it for the given device type, couple the component to a model file. You can also specify text-based stimulus, either in the HSPICE netlist language or in a behavioral language such as Verilog-A.

HSPICE ships hundreds of examples for your use; see [Listing of Demonstration Input Files](#) for paths to demo files.

For batch processing using OCEAN scripts for the following features, see:

- [Appendix B, OCEAN API Functions for HSPICE Monte Carlo Analysis](#)
- [Appendix C, OCEAN API Functions for HSPICE RF Analysis](#)
- [Appendix D, OCEAN API Functions for HSPICE Corner Analysis](#)
- [Appendix E, Using the HSPICE-Provided OCEAN API for HSPICE Optimization Analysis](#)
- [Appendix F, OCEAN API Functions for the HSPICE Measure Utility](#)
- [Appendix G, OCEAN API Functions for HSPICE MOSRA Analysis](#)

Finally, for information on use of the HSPICE Measurement Utility see [Chapter 12, HSPICE Measurement Utility](#).

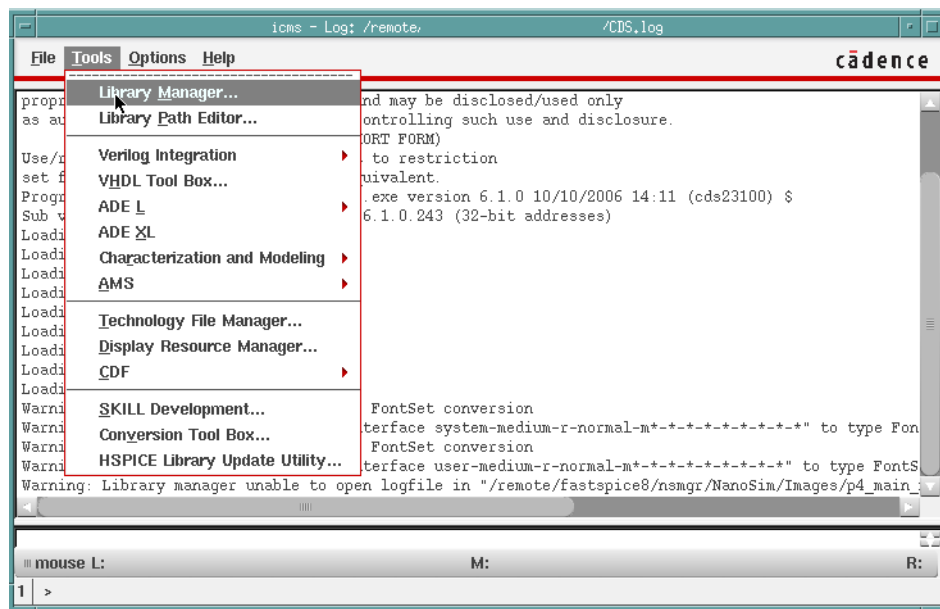
This quick-start tutorial contains the following tasks:

- [Task 1: Start Up the Tool](#)
- [Task 2: Configure Your Design for Netlisting](#)
- [Task 3: Review Available Analyses Windows](#)
- [Task 4: Set Up Outputs and Explore the Plotting Assistant](#)
- [Task 5: Load a State](#)
- [Task 6: Create a Netlist File](#)
- [Task 7: Run a Simulation and Use the Plotting Assistant](#)
- [Task 8: Loading and Running a Verilog-A Example](#)
- [Task 9: Set Up and Run a Monte Carlo Simulation](#)
- [Task 10: Run a Corner Analysis](#)
- [Task 11: Run an RF Analysis](#)
- [Task 12: Run an Optimization Analysis](#)

Task 1: Start Up the Tool

Locate the PLL_Demo_61.tar.gz tutorial database to start up the Environment.

1. Make a local copy of the compressed file and extract the contents.
2. Change directory to the PLL_Demo directory and start **icms &**. The CIW (Command Interpreter Window) is the first window to open.

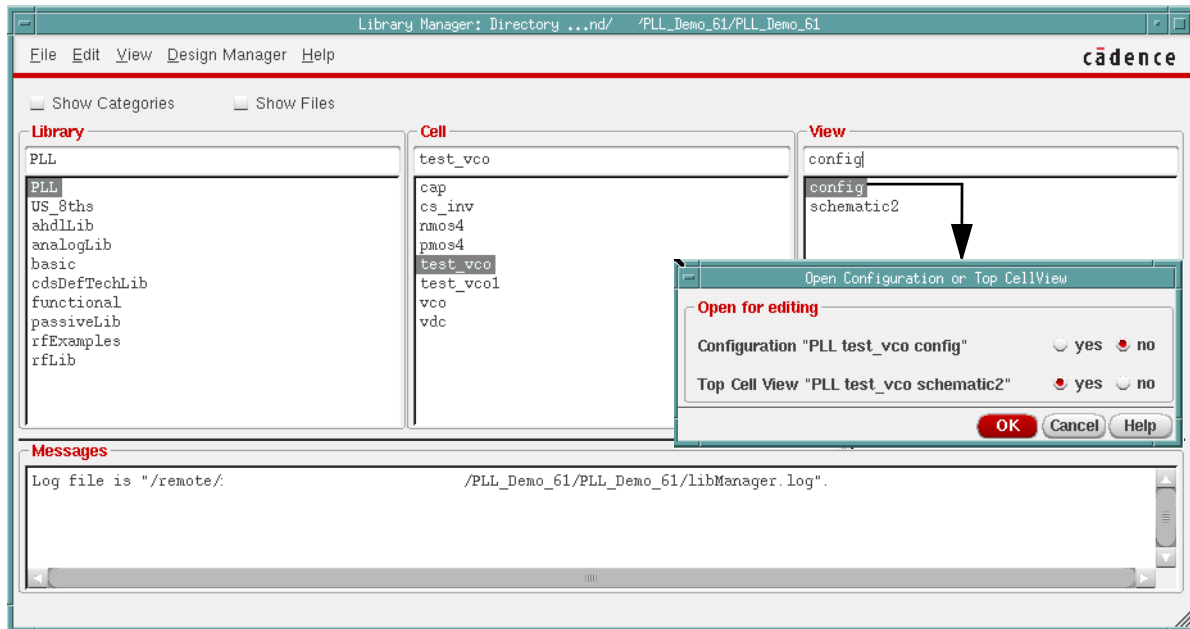


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

3. Select Tools > Library Manager to display the Library Manager dialog.
4. In this window, open PLL/test_vco/config by selecting **PLL** from the Library column, **test_vco** from the Cell column, and then double-click on **config** in the View column. A dialog opens asking if you wish to open both the schematic and configuration. Accept the default, just the schematic2 view, which displays the Schematic Editor window.

Chapter 1: Quick-Start Tutorial

Task 1: Start Up the Tool



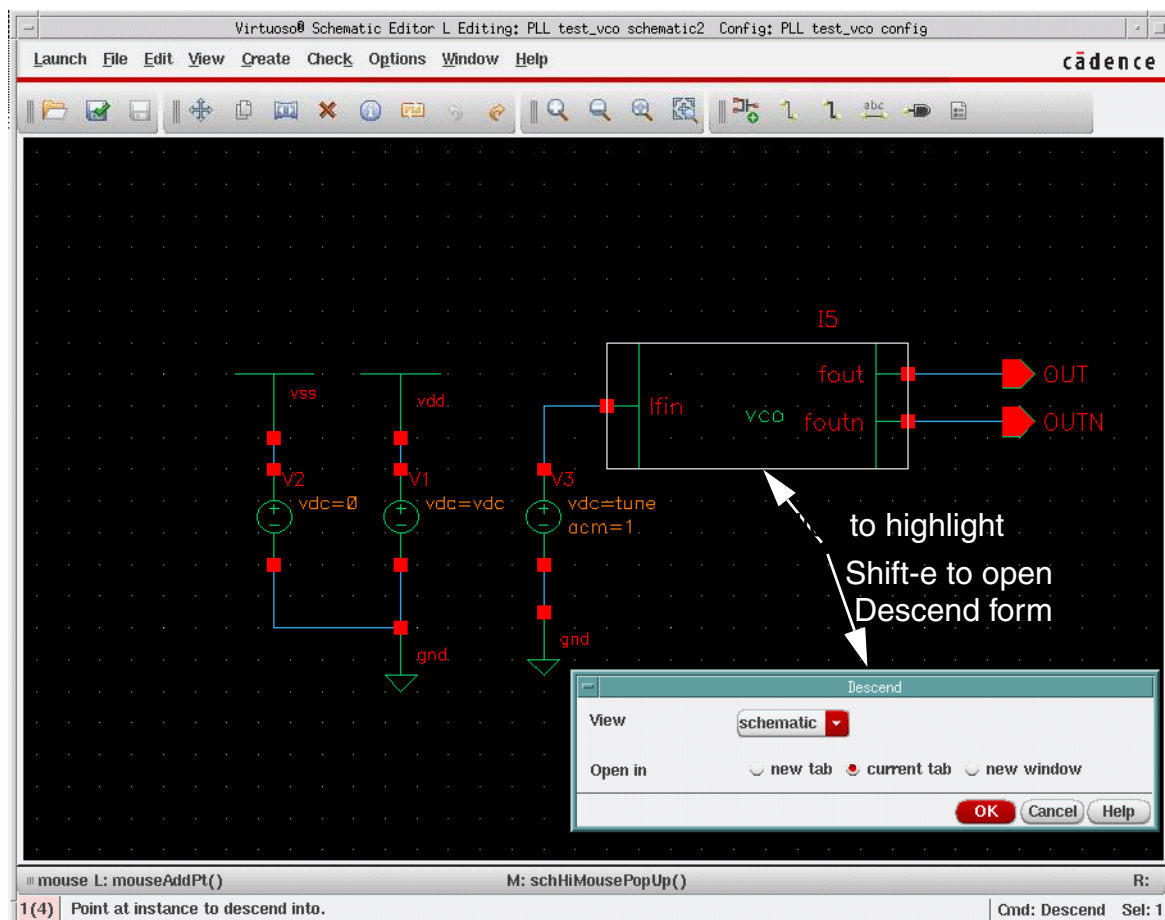
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

5. Descend into the vco schematic:


- Left-click to select the vco block.
- Shift-e to invoke the descend command. When the dialog appears, leave **schematic** as the View and **OK** the dialog.

Chapter 1: Quick-Start Tutorial

Task 1: Start Up the Tool



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- Examine the lower level of the design by repeating the descend command (Shift-e) for blocks lower in the hierarchy.
- Use the  controls and the keyboard arrows to move and view the areas of interest.

Chapter 1: Quick-Start Tutorial

Task 1: Start Up the Tool

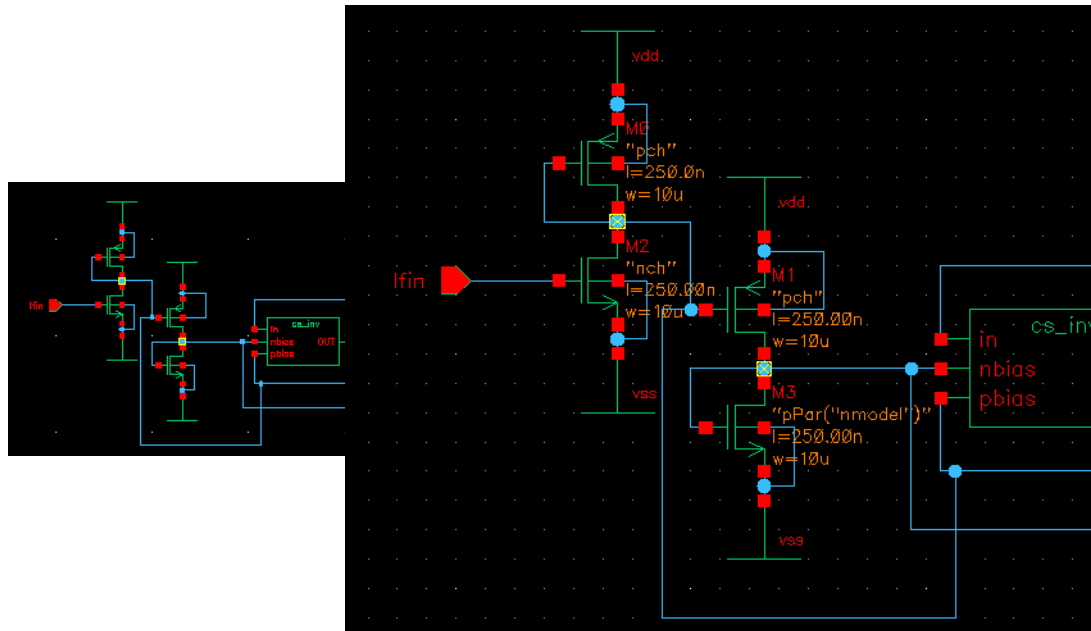
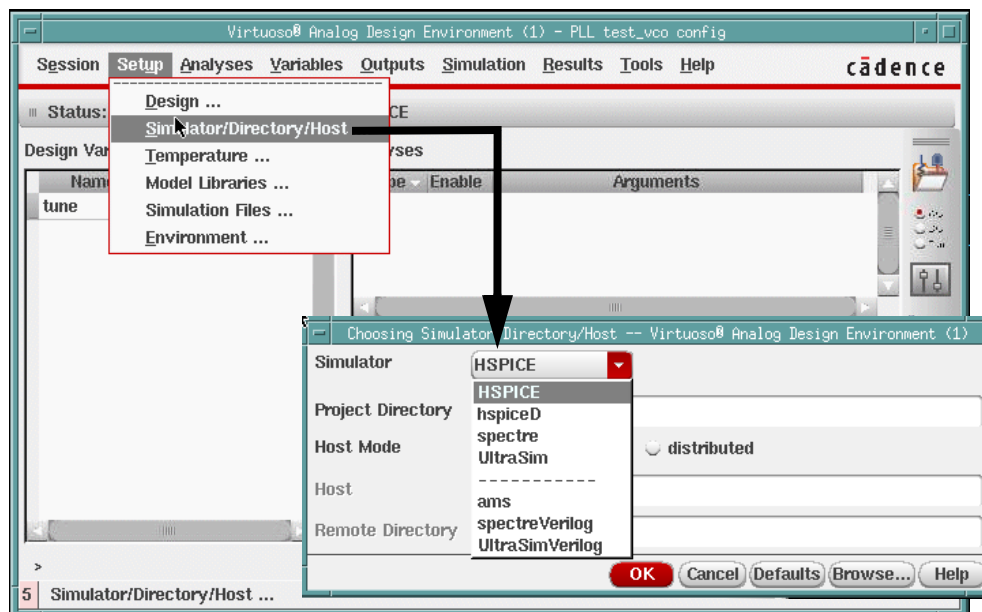


Figure 1 Descend and zoom to lower levels of layer stack

- Return to the top level by selecting Ctrl-e.
6. Open the Environment: In the Schematic window select Launch > ADE L. The Environment Console (with banner text “PLL test_vco config”) opens.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission

7. In the Environment Console, select Setup > Simulator/Directory/Host, and using the dropdown list, specify **HSPICE** as the Simulator. Leave the other fields as defaults and OK the form. The Environment session is now properly initialized.

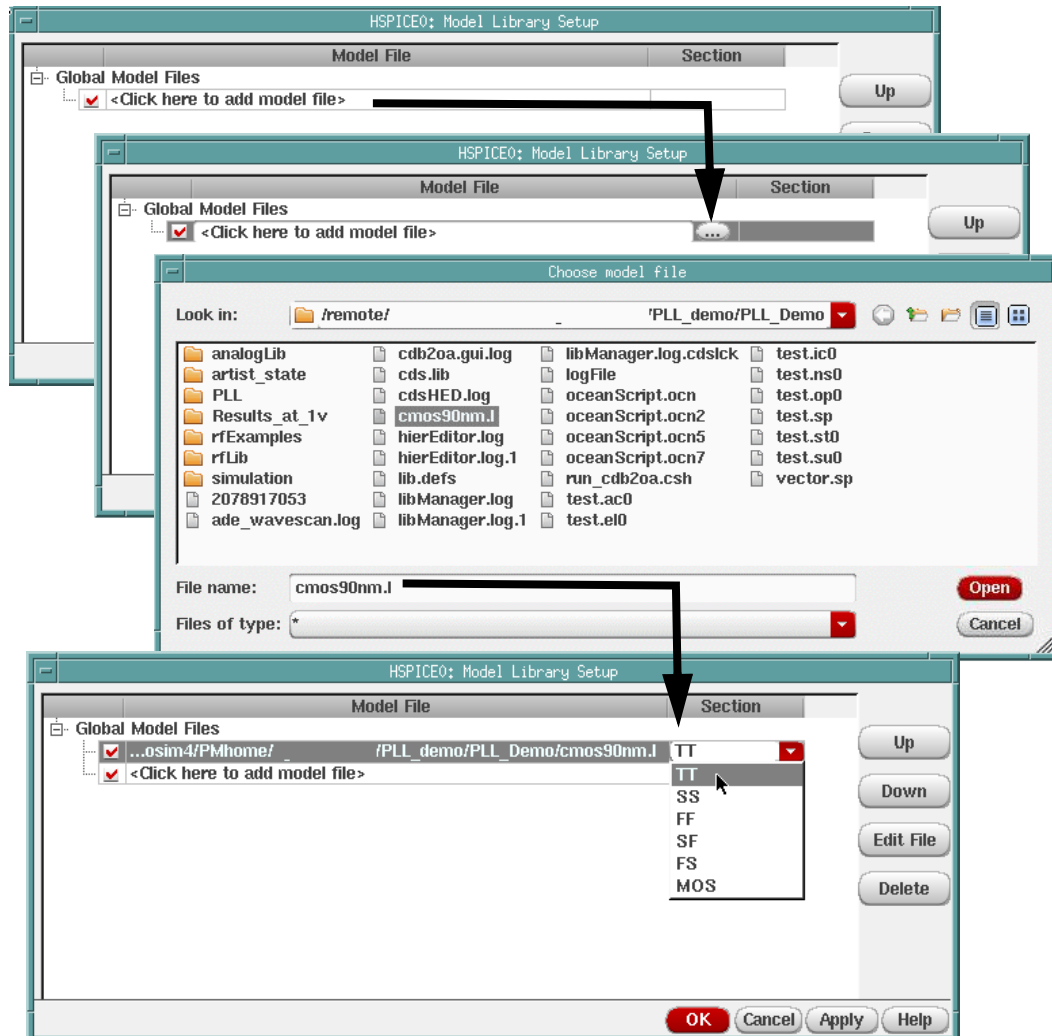
Task 2: Configure Your Design for Netlisting

This task helps you to set up model and simulation files, and introduces you to other setup specification forms.

1. In the Environment Console, select Setup > Model Libraries to display the HSPICE: Model Library Setup dialog.

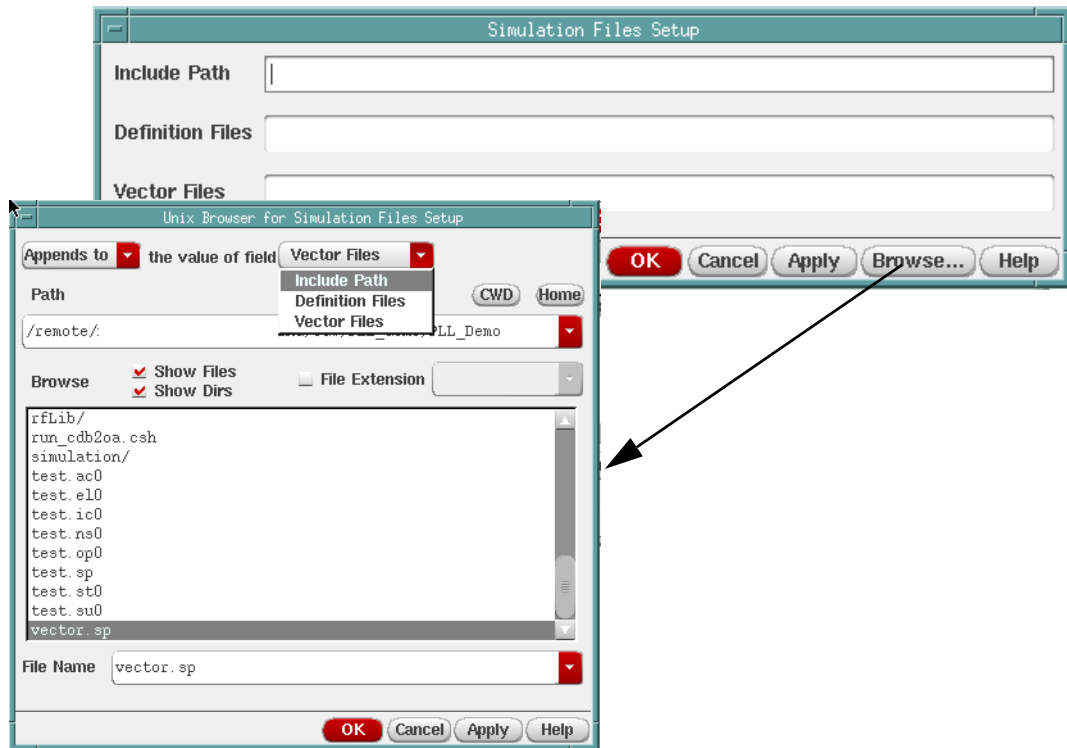
Chapter 1: Quick-Start Tutorial

Task 2: Configure Your Design for Netlisting



2. Double-click in the top row of the table and then click the browse button to open the Choose model file browser and locate the file *cmos90nm.l*, which is in the current directory. Click Open.
3. Click in the Section column cell to display the dropdown list and select the section TT. Then OK the dialog.
4. View the Simulation Files Setup dialog: Setup > Simulation Files.

Chapter 1: Quick-Start Tutorial
Task 2: Configure Your Design for Netlisting

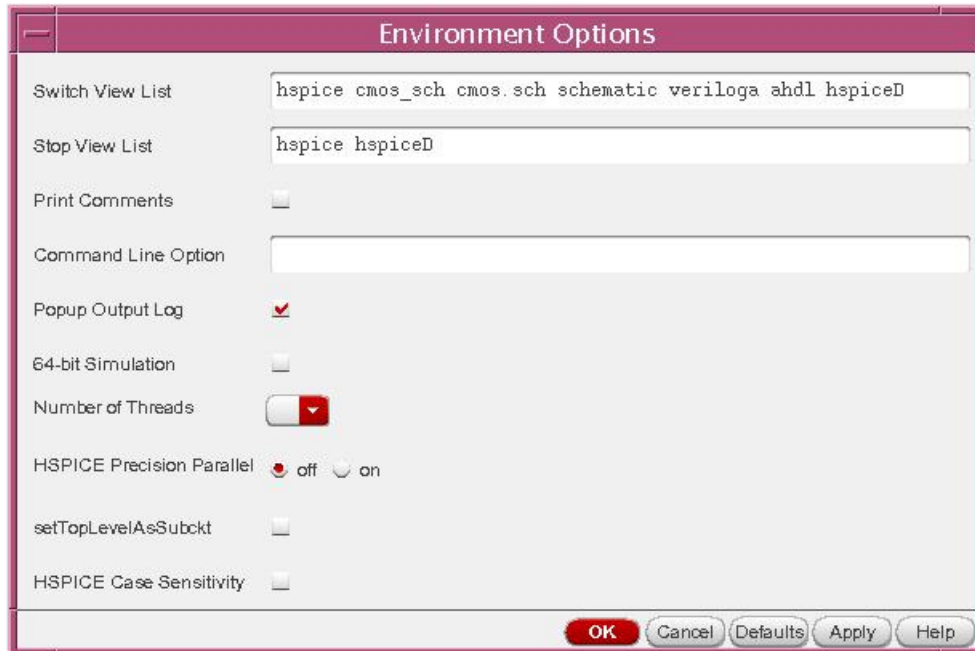


In addition to the Include Path and Definition Files fields available in traditional interfaces, the HSPICE integration introduces the Vector Files field to integration Environment users so you can include a file that is written in HSPICE's digital vector syntax. You can specify multiple files, and they will be written into individual `.VEC` statements.

- Click **Browse** to explore the Unix Browser for Simulation Files.
 - Cancel both the Unix Browser... and the Simulation Files Setup dialogs.
5. View the Environment Options dialog: Select **Setup > Environment** to open the Environment Options dialog.

Chapter 1: Quick-Start Tutorial

Task 3: Review Available Analyses Windows



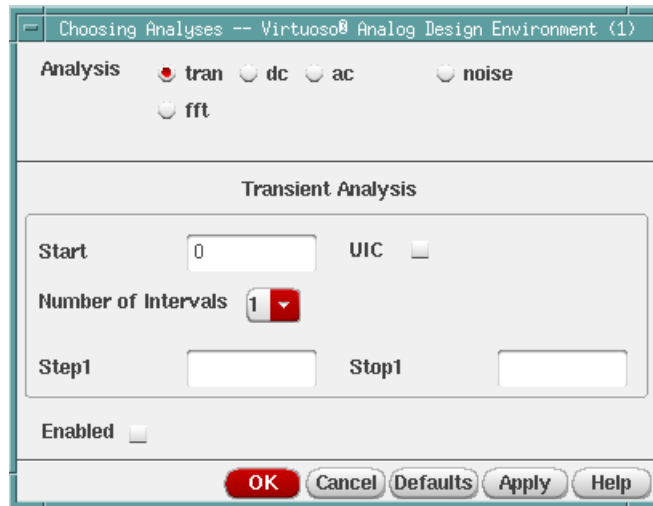
It provides the **Command Line Opt** field. Use this field to append HSPICE executable command line options to the command issued by the Environment. This is only necessary for command line options that are not supported by the GUI. In addition you can enable the **Print Comments** check box and **Popup Output Log** check box (selected *on* by default). See “Environment Options” in the [Setup](#) section of Chapter 3 for more information on this form. Leave the **Popup Output Log** checked and **Cancel** this dialog.

Task 3: Review Available Analyses Windows

Most HSPICE integration improvements are to be found in the analysis windows.

1. Under the Analyses menu of the Environment console, select Analyses > Choose. A dialog opens to display radio buttons that allow you to set up each analysis type. Note that the native Environment supports the specification of just one analysis per type. This is the same for all simulators, including HSPICE.

2. Transient Analysis: Click the **tran** radio button. The dialog provides the ability to specify multiple Intervals. Test this by selecting a different value from the Number of Intervals list. Enter a few step/stop values, and click the Enabled check box.

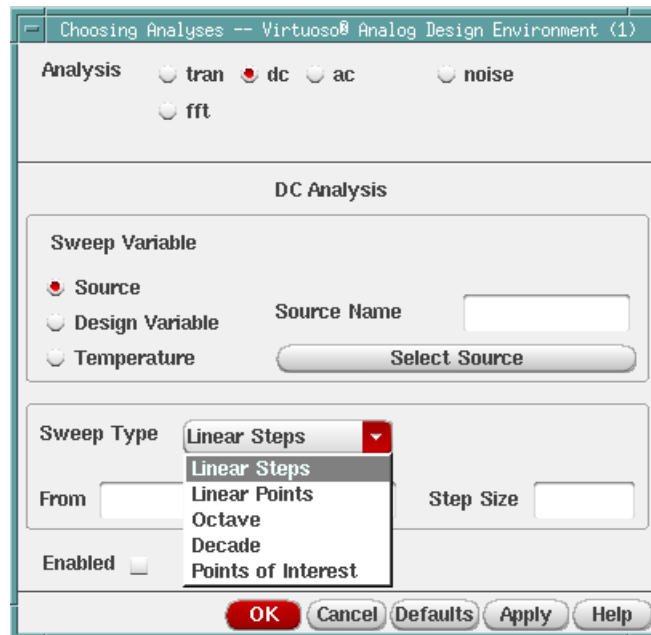


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

3. DC Analysis: The HSPICE integration specifies Linear Steps for the Sweep Type dropdown list. Click the **Select Source** button to automatically fill in the **Source** name by clicking on the component in the schematic.

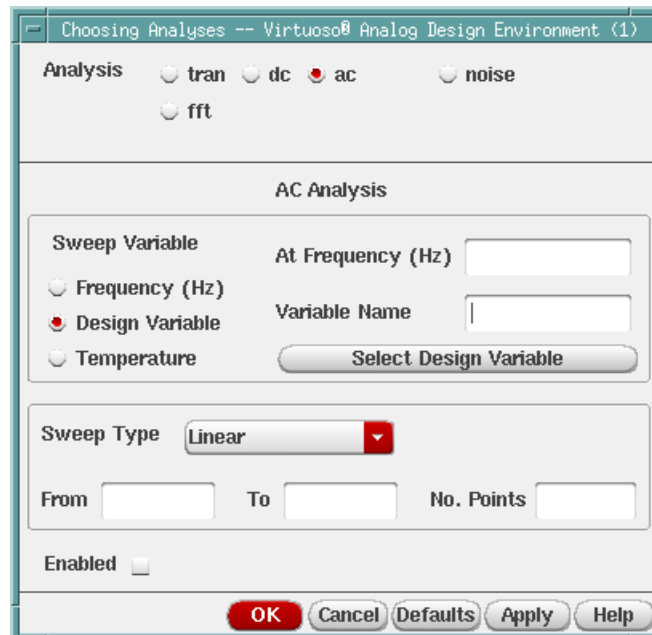
Chapter 1: Quick-Start Tutorial

Task 3: Review Available Analyses Windows



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

4. AC Analysis: The AC HSPICE user interface has the additional feature of the single point frequency sweep. The figure shows all GUI elements when the Design Variable radio button is selected.



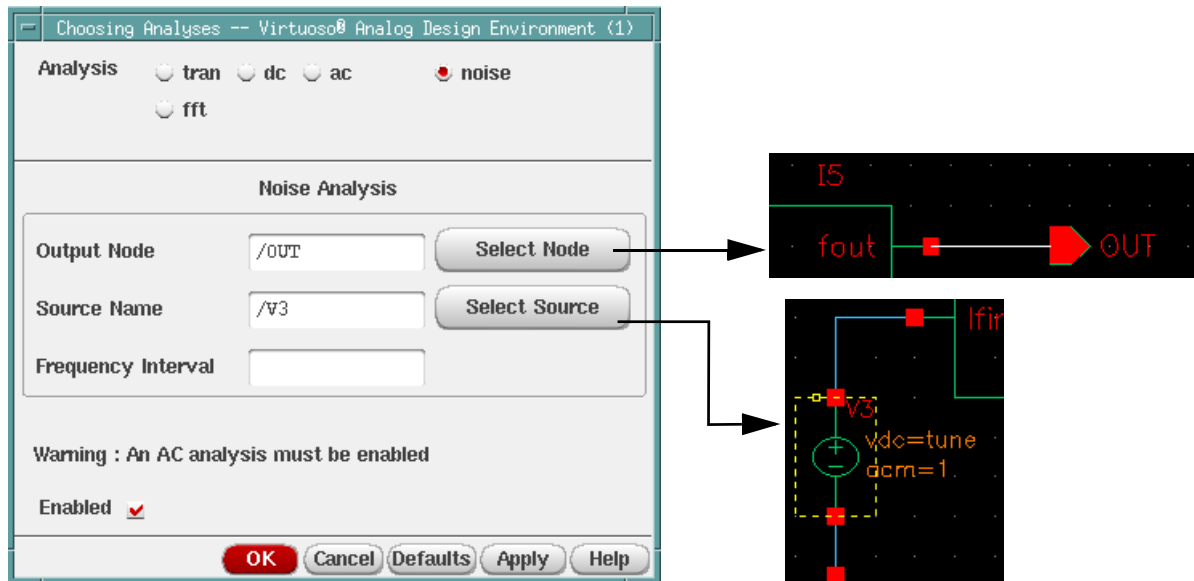
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Try setting up a single point temperature sweep by following these steps:

- Select **Temperature** as the Sweep Variable.
 - Enter a single frequency point in the **At Frequency** field.
 - If you select Design Variable as the sweep variable, click the **Select Design Variable** button and select a variable from the list dialog that opens.
 - Specify the sweep type and enable the analysis by clicking the **Enabled** check box.
5. Noise Analysis: NOISE is also similar to the hspiceD interface, although a reminder informs you that AC is also a required analysis. Noise data is saved to a PSF results file. Disable the AC analysis you just set up, and then return to this panel to show the warning message. Re-enable AC and return again to the noise setup form. You can redirect noise data results by using the HSPICE `-o` option to set results to a directory other than `/psf`. For example: `-o ./results/input`.

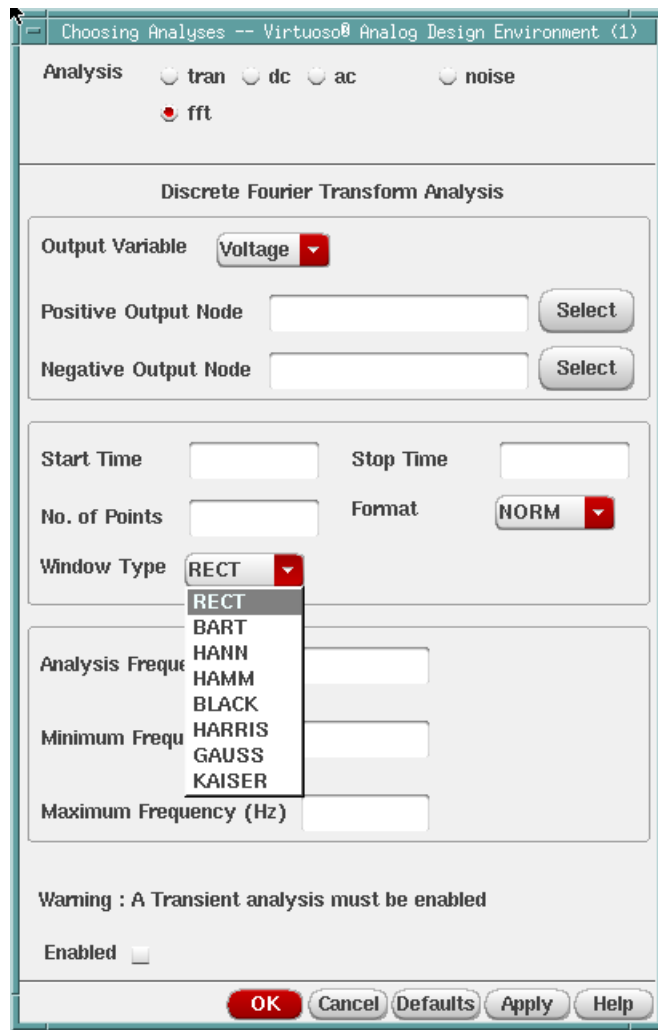
Chapter 1: Quick-Start Tutorial

Task 3: Review Available Analyses Windows



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- Select **OUT** as the output net by clicking the Select Node button and then left-click on the schematic to select the wire connecting the vco fout terminal with the pin OUT.
 - Repeat these steps to select **V3** as the input source.
 - Enter **10** as the Frequency Interval.
6. Full-Fourier Transform: The FFT Analysis window is a new form, specifically for the HSPICE functionality. Like NOISE, it is a dependent analysis, this time on TRAN.
- Select **/OUT** as the positive input node by clicking the Select Node button and then left-click on the schematic to select the wire connecting the vco fout terminal with the pin OUT.
 - Do not enter anything for the negative input node, as it defaults to gnd!.
 - Start and stop can be the same as what was specified for the tran analysis, and number of points can be 1024.
 - Do not make any other entries to allow HSPICE defaults be used.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

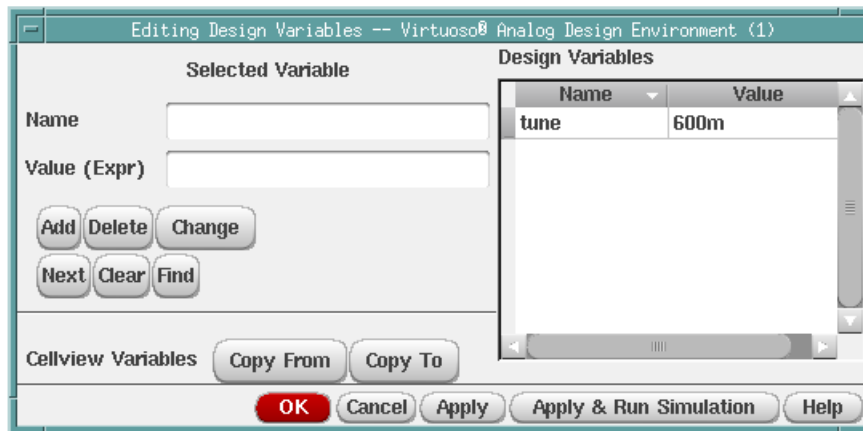
Note: Running the analysis at this point takes a significant amount of time because of the scrolling of all the noise data to the output log. So before running a simulation this tutorial will present a Task on loading states, and load one in that does not have the noise analysis enabled.

7. In the Environment Console, select the Variables pulldown to acquaint yourself with the design variables options.

Select Edit to display the Editing Design Variable dialog. The listed design variables and value (tune and 600m) are the same as the those shown on the Environment Console.

Chapter 1: Quick-Start Tutorial

Task 4: Set Up Outputs and Explore the Plotting Assistant



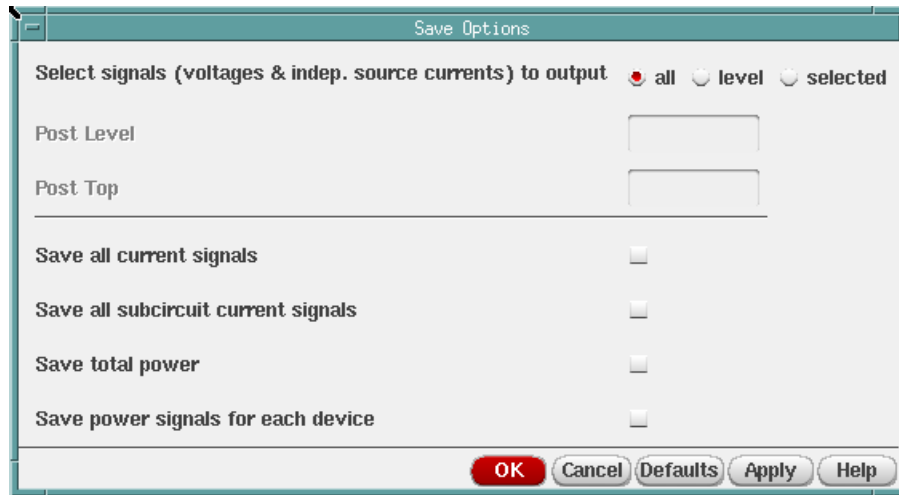
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

No improvements were made to the Variables pulldown selections over the existing hspiceD interface. These selections allow you to edit, delete, copy variables to, and retrieve variables from your design. You can also access a variables setup form that will let you define new variables. All design variables that are defined here are netlisted as global .PARAM statements.

Task 4: Set Up Outputs and Explore the Plotting Assistant

This task reviews the Save Options form and the specialized HSPICE Plotting Assistant for the HSPICE integration.

1. Choose Outputs > Save Options to display the Save options dialog.

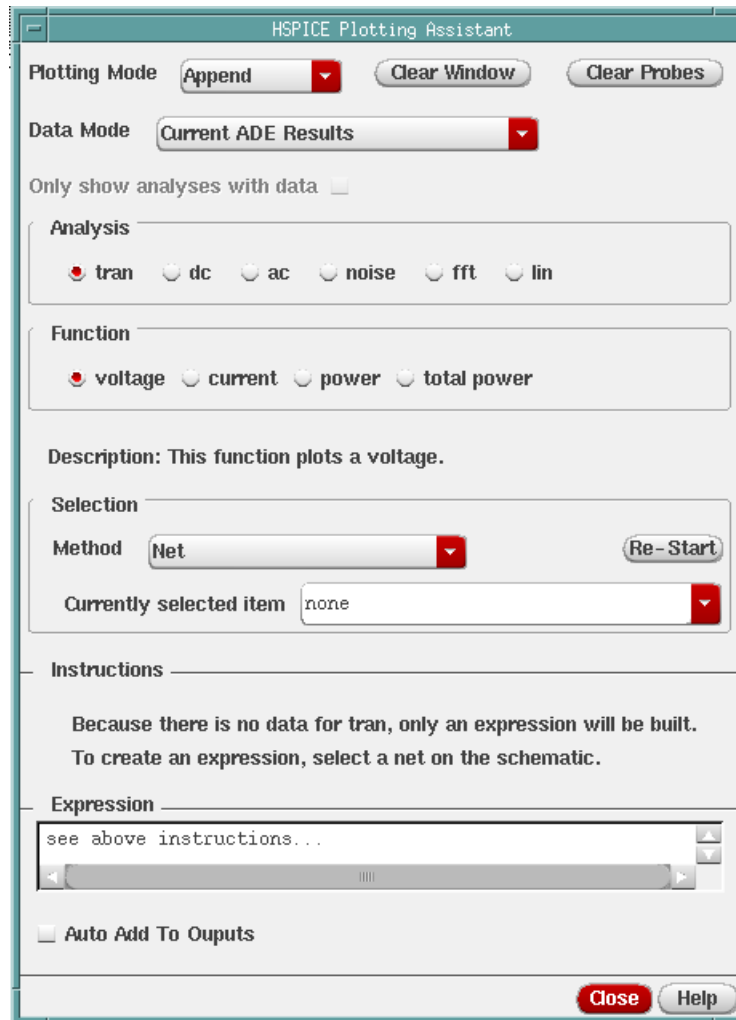


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- The radio button options: **all**, **level**, and **selected** control the value of the .OPTION PROBE command in HSPICE.
 - If you choose **level** for the first option you can specify the Post Level and Post Top options (.OPTION POSTLVL to limit data written to your waveform file to a specified level of nodes, and .OPTION POSTTOP to limit the data written to the waveform file to data from only the top *n* level nodes.
 - Review the other check box controls which are self-explanatory.
2. Cancel the dialog without making any changes.
 3. Open the HSPICE Plotting Assistant to explore the unique capability of letting you set up expressions ahead of time, as well as, plotting signals and expressions finishing the simulation. Select Results > Plotting Assistant.

Chapter 1: Quick-Start Tutorial

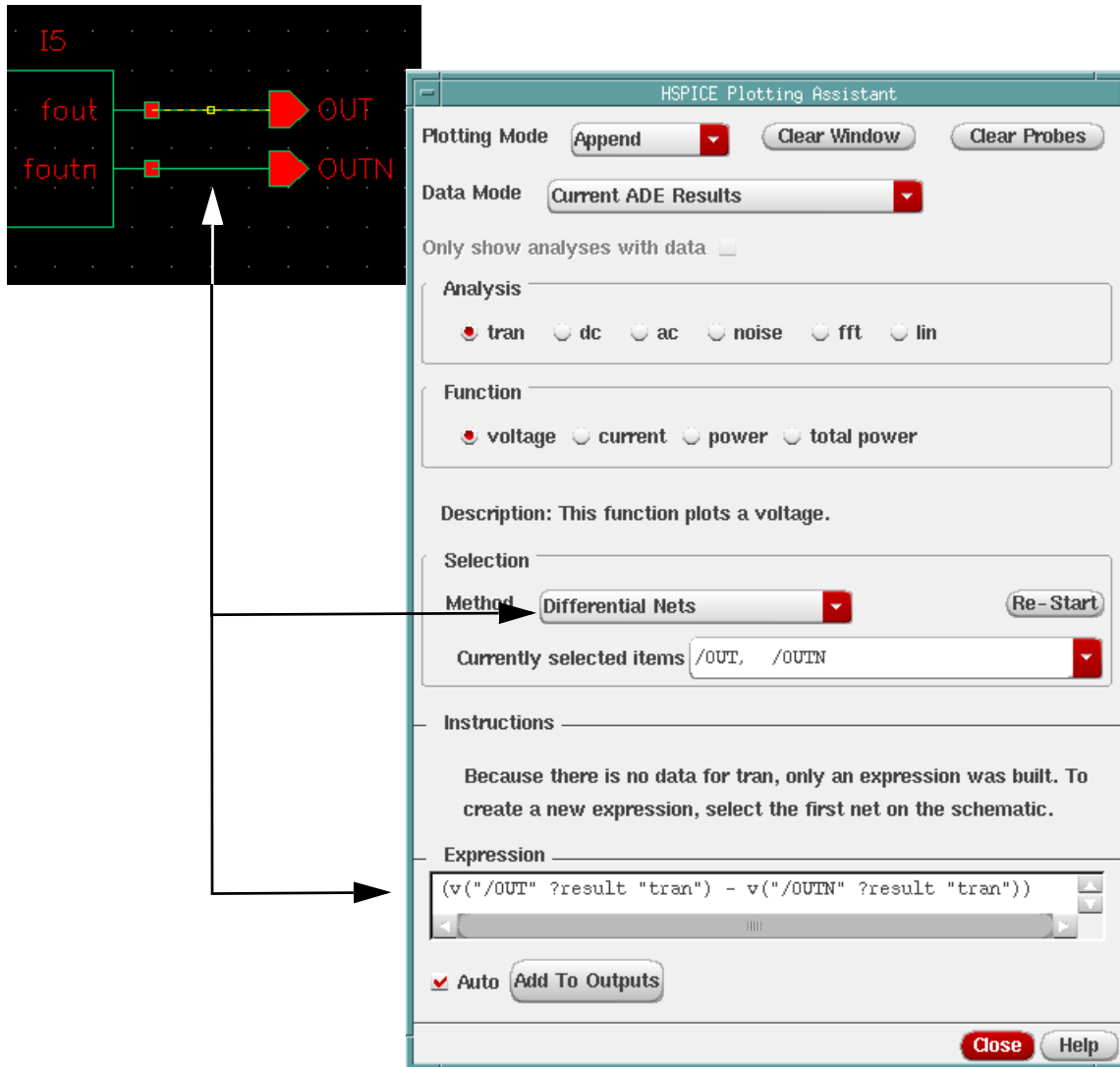
Task 4: Set Up Outputs and Explore the Plotting Assistant



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

4. Select Results > Plotting Assistant to open the HSPICE Plotting Assistant dialog and explore the dropdown lists:
 - Plotting Mode—Append: Do not change
 - Data Mode—Current ADE Results: Do not change
 - Selection Method —Change from Net to Differential Nets
 - On the schematic window, click on the wire attached to pin OUT, and then the wire attached to pin OUTN to display these two nets the Currently selected items listbox of the HSPICE Plotting Assistant. See also that the Expression $(v("/OUT" \text{ ?result "tran"}) - v("/OUTN" \text{ ?result "tran"}))$ is added to the Expression field

Chapter 1: Quick-Start Tutorial
Task 4: Set Up Outputs and Explore the Plotting Assistant

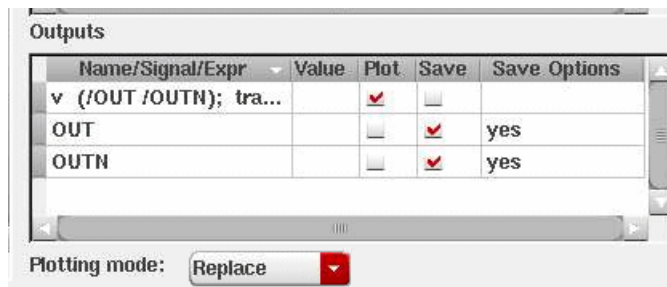


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- The built-in Environment waveform viewer ViVA Graph Window plots the waveform at the same time.
- Click the Add to Outputs button (not the check box) and notice that the signals as well as the expression are added to the Outputs table in the main ADE Console window.

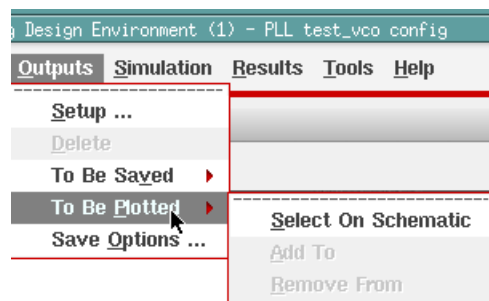
Chapter 1: Quick-Start Tutorial

Task 5: Load a State



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

5. In the Environment Console window, interactively specify additional outputs to be plotted by choosing Outputs > To Be Plotted. The outputs you want plotted must also be check-marked under the **Save** column. A plot will not be created in the HSPICE integration unless both check boxes are ticked.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Make several selections of other wires on the schematic, optionally descending into the vco block, and then press Esc on the keyboard when you have completed your selections. The additional signals, etc. are added to the Plot list of the Outputs table and their check boxes turned on. (This capability exists in all ADE integrations and is not unique to the HSPICE interface.)

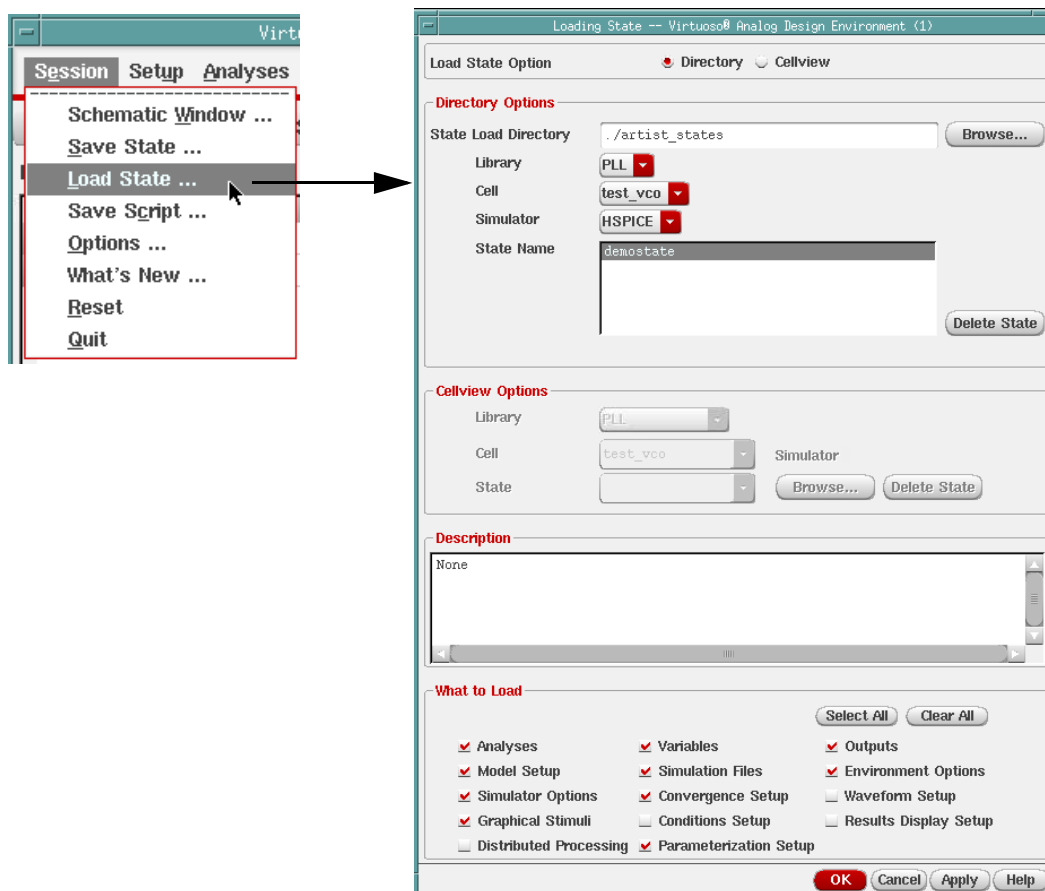
Task 5: Load a State

State files are used in the Environment to store setups. You just went through the steps of setting up a design for simulation. Since it can be time-consuming

to do this with every new session, the Environment provides the ability to save and restore state files.

Note: When you load a previously saved state, all current window settings are overwritten and the active design and data reflect the loaded state.

1. On the ADE Console window, select Session > Load State. Notice that States can be stored to directories or to cellviews so that they can be managed with design data. You are going to load a State from a directory.



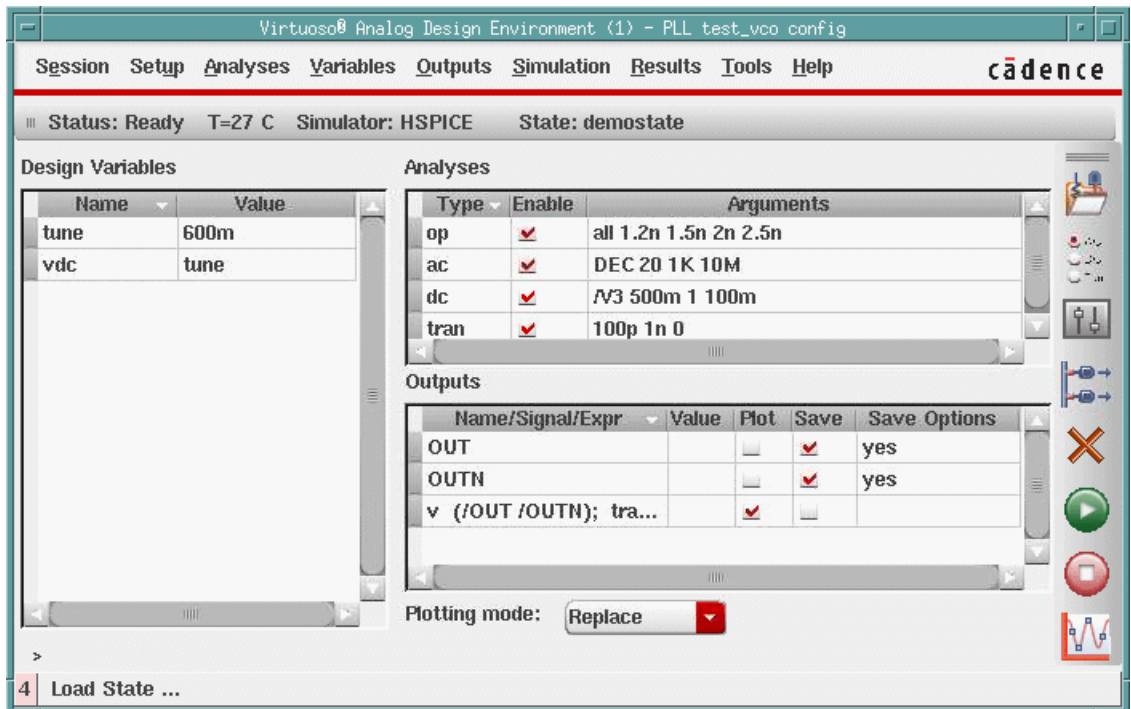
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

2. Change the State Load Directory to ./artist_state. Tab out of this field. Also ensure that the Library, Cell and Simulator are set to PLL, test_vco and HSPICE.
3. Select “demostate” from the list of states.

Chapter 1: Quick-Start Tutorial

Task 6: Create a Netlist File

4. Leave all other settings as is and click **OK**.
5. Examine some of the dialogs visited earlier in this tutorial and see the effect of loading the state. For example, see the changes to the Environment Console when the “demostate” is invoked:



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

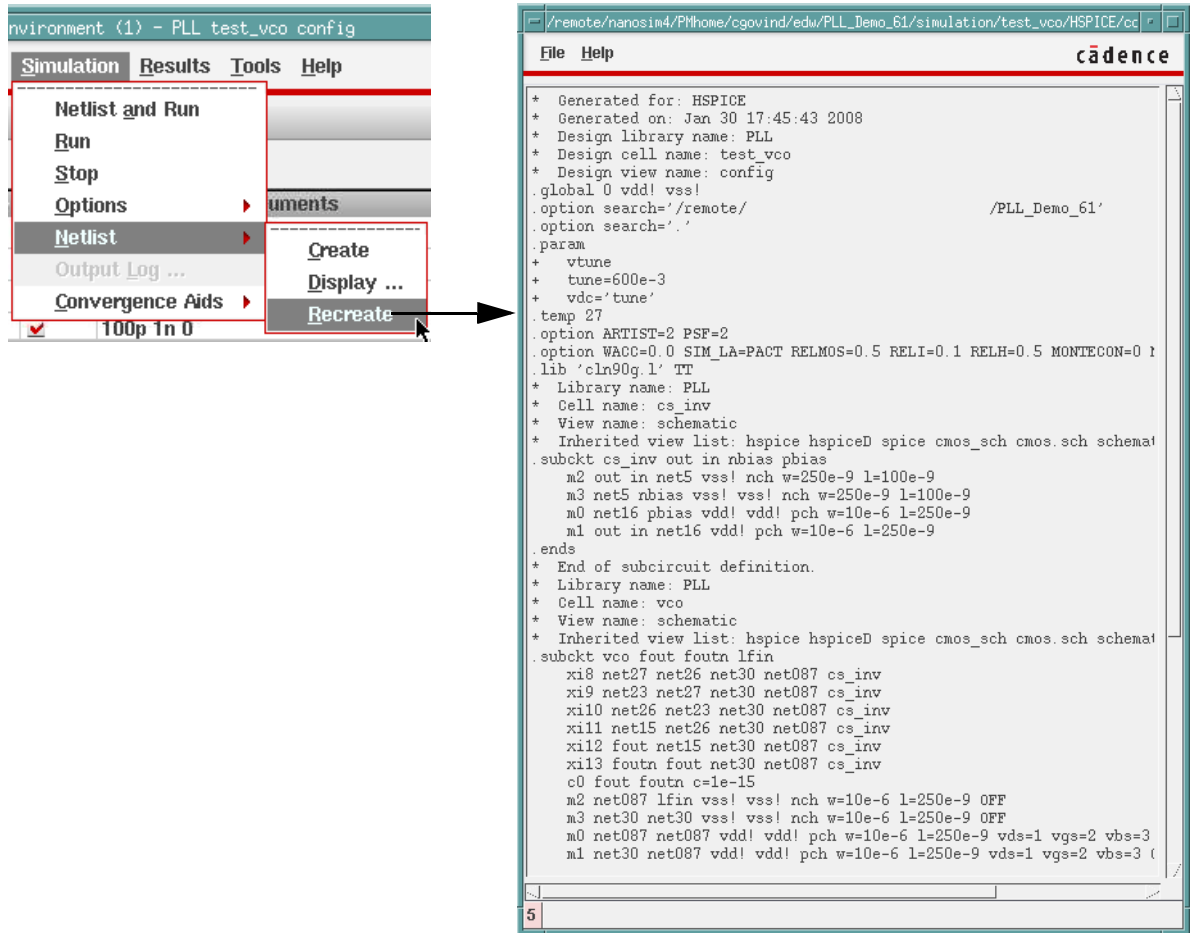
Task 6: Create a Netlist File

To construct a netlist file:

1. On the Environment Console, choose Simulation > Netlist > Recreate. A successful operation displays the netlist.

Chapter 1: Quick-Start Tutorial

Task 6: Create a Netlist File



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

If you receive an error message, it may be because your design needs to be explicitly re-extracted. If that happens follow these steps:

- Return to the schematic window.
 - Select File > Check and Save.
2. Check your netlist: When netlisting succeeds, the netlist appears in a text viewer. Examine the contents.
 3. Pay special attention to the instance statement for m3 within the vco subcircuit:

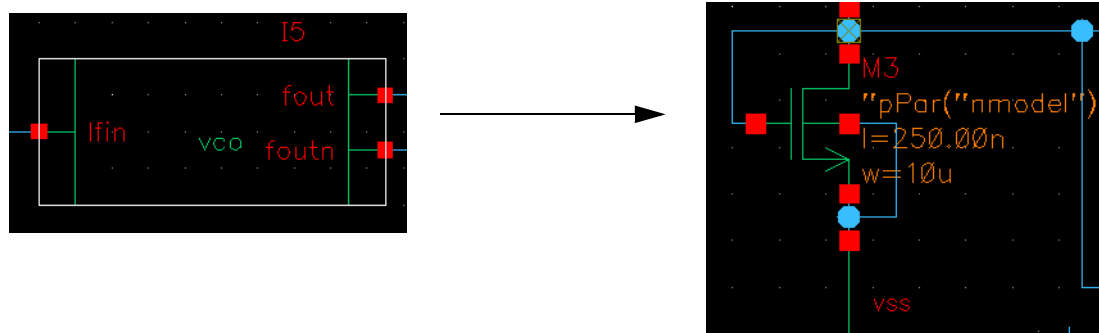
```
m3 net5 nbias vss! vss! str(nmodel) w=10e-6 l=250e-9
```

Chapter 1: Quick-Start Tutorial

Task 7: Run a Simulation and Use the Plotting Assistant

This statement shows the integration's support of model name passing, or specification of a model on a parent cell.

4. To see how this is done: descend into the vco block with the Shift-e binding (or **Edit > Hierarchy > Descend Edit**).
5. Use a combination of pans and zooms to locate M3 on the left side of the schematic.
6. Notice that the text “pPar(“nmodel”)” is displayed next to the device. This is an expression that tells the netlist to look on the parent block for the value of nmodel, and to use that value for the model field.



7. Return to the top design with the Ctrl-e binding, and select the vco block.
8. Use the “q” binding to bring up the Object Properties dialog. Notice that the value of nmodel is set to “nch”.
9. Now return to the netlist, and locate the instance statement for the vco:

```
xi5 out outn net034 vco nmodel=str('nch')
```
10. You can use the Property Editor (q bindkey) to change the nmodel value at the vco block level to control the model that is used by M3 in the vco design.

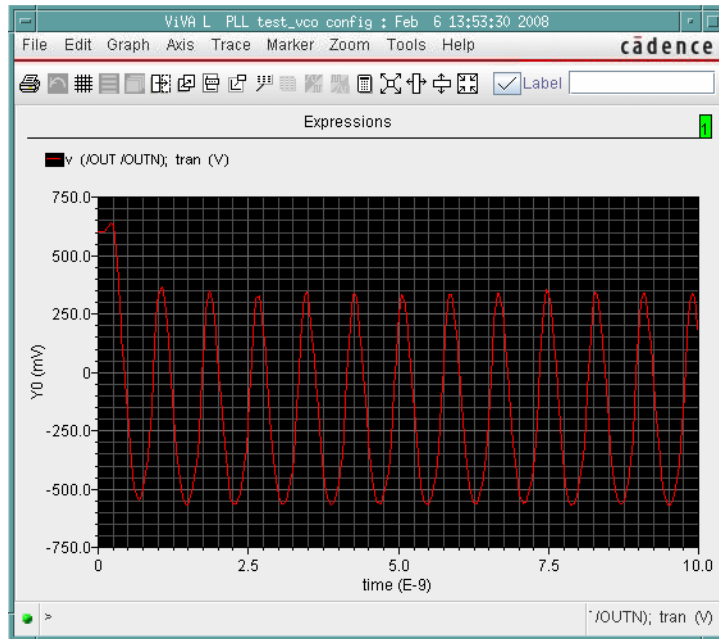
Task 7: Run a Simulation and Use the Plotting Assistant

You are now ready to run a simulation. You will be using the “demostate” for your first trial run.

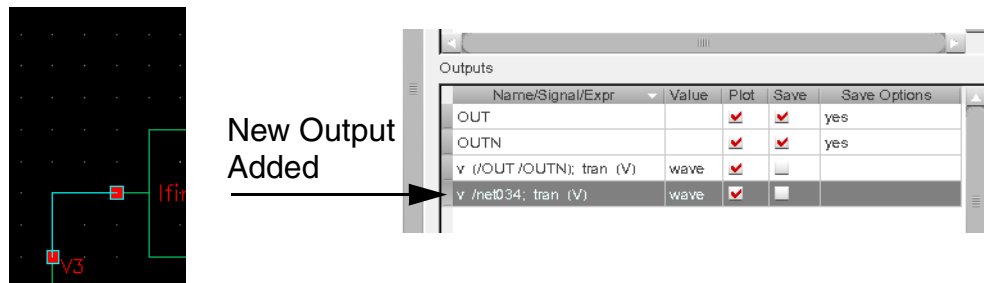
1. In the ADE Console, select Simulation > Run. Notice the display of a log file. At the end of the run your signals are plotted in the Environment’s ViVA waveform viewer.

Chapter 1: Quick-Start Tutorial

Task 7: Run a Simulation and Use the Plotting Assistant



2. In the Environment Console, open the HSPICE Plotting Assistant dialog (Results > Plotting Assistant).
 - In the window that opens, enable **Auto Add to Outputs** and plot a single voltage (select a voltage node on the schematic). For example:

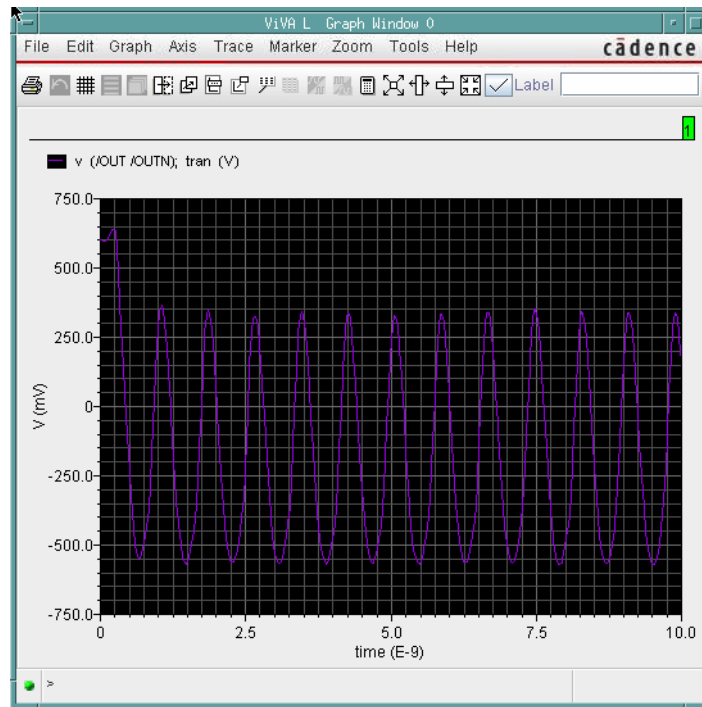


Notice that it is added to the Environment Console **Outputs** list.

- On the HSPICE plotting Assistant, disable (uncheck) **Auto Add to Outputs**. Close the current ViVa window by selecting File > Close. Plot a few more signals on a new graph window using the Plotting Assistant.
- Change **Method** in the **Selection** section of the Plotting Assistant to **Differential Nets** and plot a differential signal by selecting two wires on the schematic. (For example, OUT and OUTN.)

Chapter 1: Quick-Start Tutorial

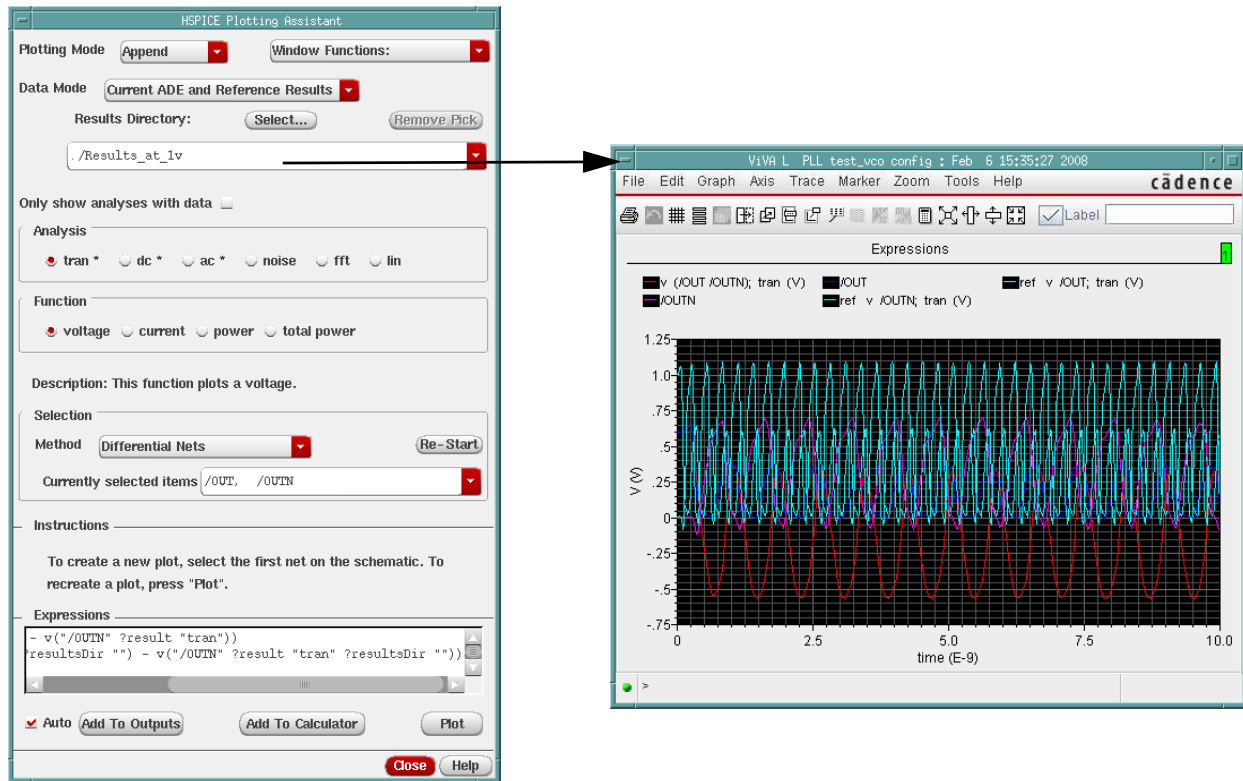
Task 7: Run a Simulation and Use the Plotting Assistant



- Then change **Method** back to **Net**. Click the down arrow of the dropdown list labeled **Currently selected item** to show the history of items plotted. Select any of the items listed and instantaneously view the plot.
3. In the Plotting Assistant, change the **Data Mode** to **Current ADE and Reference Results**. In the Results Directory field type `./Results_at_1v`.

Chapter 1: Quick-Start Tutorial

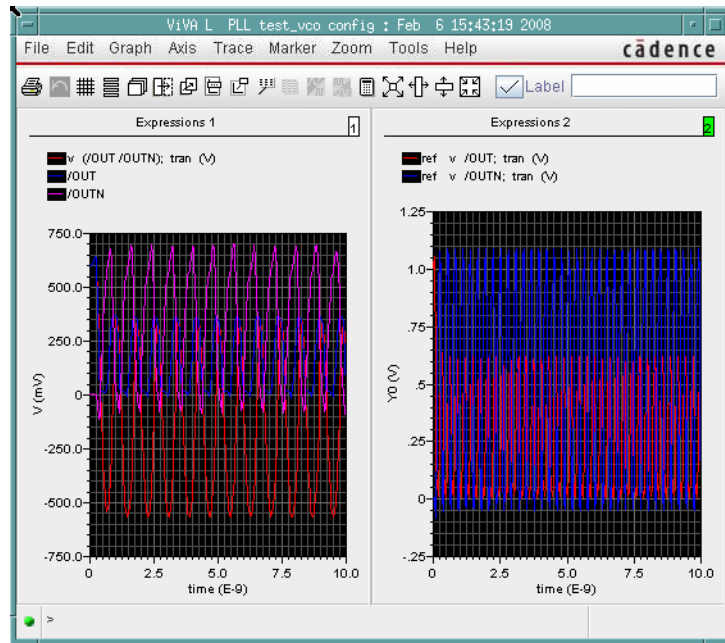
Task 7: Run a Simulation and Use the Plotting Assistant



4. Now plot the differential voltage again and see the difference in the frequency of oscillation.

Chapter 1: Quick-Start Tutorial

Task 8: Loading and Running a Verilog-A Example



Task 8: Loading and Running a Verilog-A Example

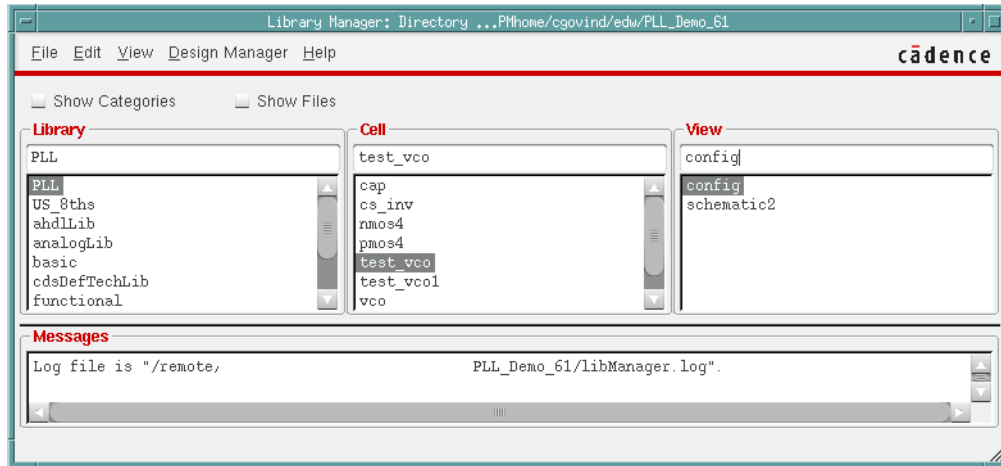
This task helps you to switch to and use a Verilog-A cell view in the HSPICE integration to the Analog Design Environment. This task assumes you have a Verilog-A cellview and a corresponding symbol.

Begin this task with no other view open.

1. In the CIW dialog, select Tools > Library Manager to display the Library Manager dialog.

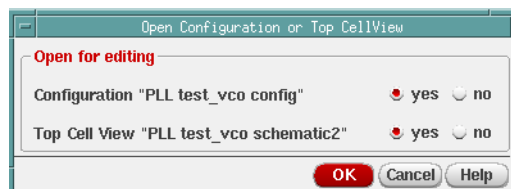
Chapter 1: Quick-Start Tutorial

Task 8: Loading and Running a Verilog-A Example



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

2. In the Library Manager window, select **PLL** from the **Library** column, **test_vco** from the **Cell** column, and then double-click on **config** in the **View** column.
3. In the Open Configuration or Top CellView window, click **yes** for both the **Configuration** and **Top Cell View** radio buttons and then click **OK** to open both the schematic and Hierarchy Editor.

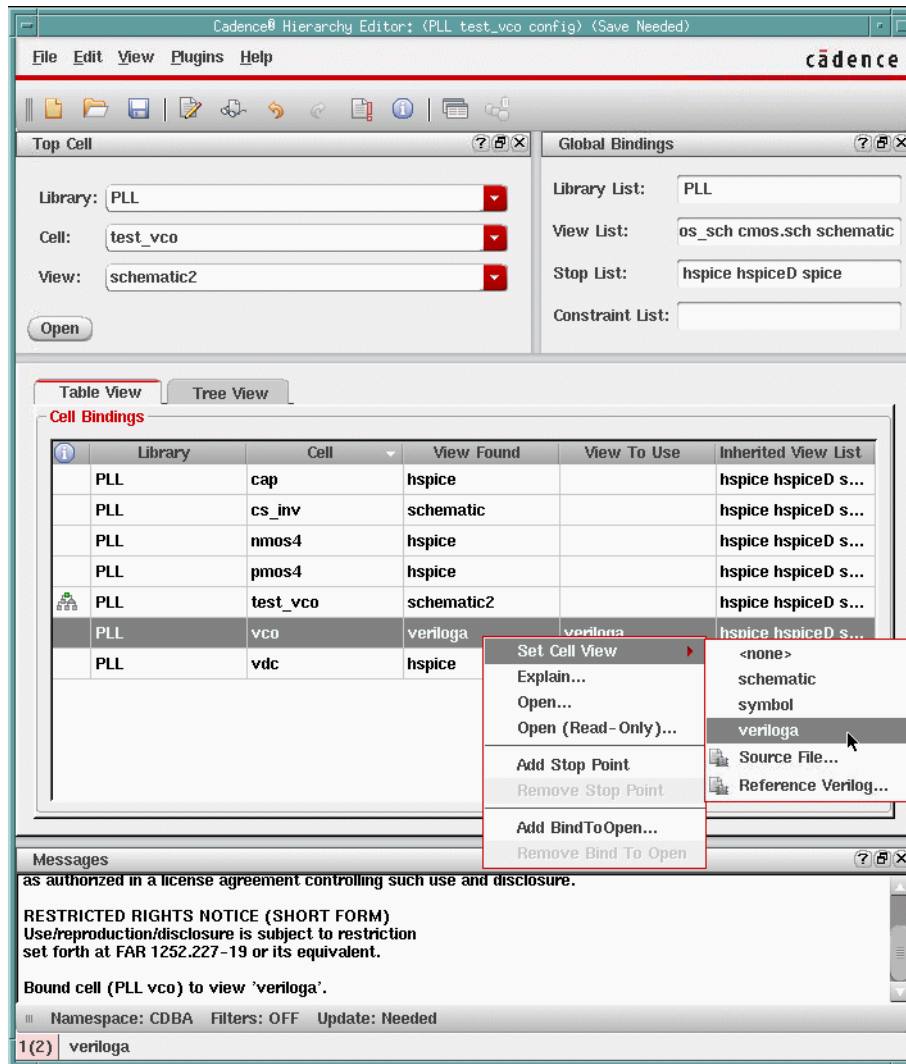


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

4. In the Hierarchy Editor, right-click the **View Found** column cell of the **vco** cell to display the menu, and select **Set Cell View**, then select **veriloga** in the expanded menu.

Chapter 1: Quick-Start Tutorial

Task 9: Set Up and Run a Monte Carlo Simulation



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

5. Continue on to launch ADE L (In the Schematic Editor, Launch > ADE L to set up HSPICE for simulation).
6. Save the Configuration.

Task 9: Set Up and Run a Monte Carlo Simulation

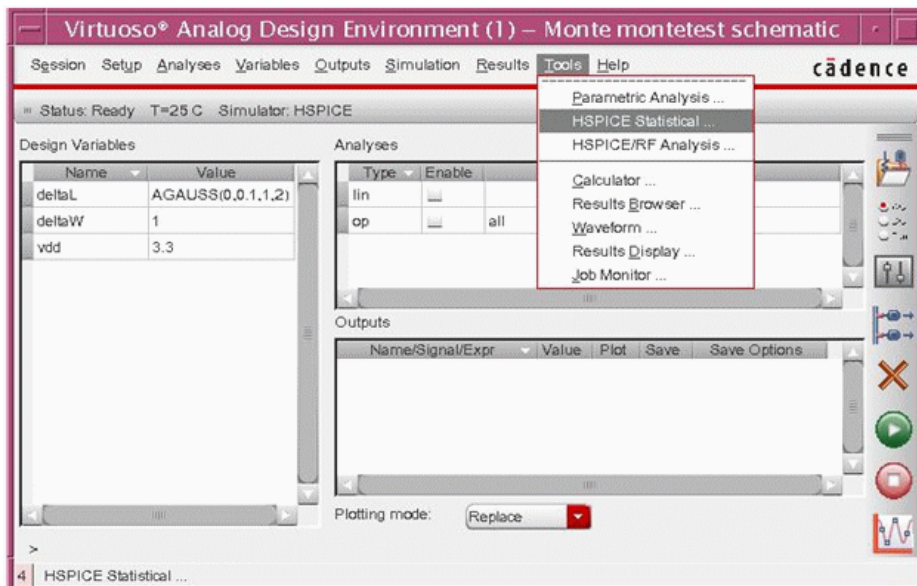
This task uses a demo case located in the HSPICE installation demo cases directory (`$installdir/demo/hspice/aa_integ/Monte_Demo_61`). Users can find a

corresponding IC51 demo case at `/$installdir/demo/hspice/aa_integ/Monte_Demo_51`.

The demonstration cases under directories `Monte_Demo_61` and `Monte_Demo_51` are provided to introduce the usage of HSPICE-ADE Integration HSPICE Monte Carlo Analysis. This feature is released with HSPICE version C-2009.03.

Follow these steps to invoke and run the demonstration case for Monte Carlo.

1. Locate `Monte_Demo_61` and open the case design:
 - Make a local copy of the demo case, change directory to the `Monte_Demo` directory, and start by typing **icms &**. The CIW (Command Interpreter Window) opens.
 - Select **Tools > Library Manager** to display the Library Manager dialog.
 - In this window, open `Monte/montetest/schematic` by selecting **Monte** from the Library column and **montetest** from the Cell column. Then double-click on **schematic** in the View column. The schematic view window is displayed.
2. Open the Environment: In the Schematic window select **Launch > ADE L**. The Environment Console (with banner text “Monte montetest schematic”) opens.



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Chapter 1: Quick-Start Tutorial

Task 9: Set Up and Run a Monte Carlo Simulation

In the Environment Console, select **Setup > Simulator/Directory/Host**, and use the dropdown list to specify **HSPICE** as the Simulator. Leave the other fields as defaults and OK the form. The Environment session is now properly initialized.

3. In the Environment Console, select **Tools > HSPICE Statistical** to display the HSPICE Monte Carlo Analysis Window.

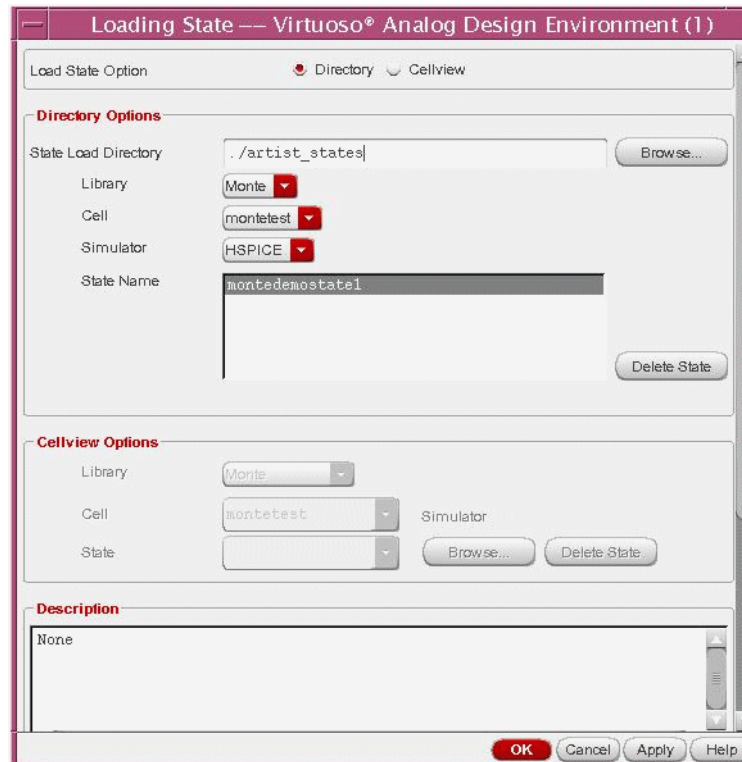
Name	Value
deltaL	AGAUSS(0,0.1,1,2)

© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

4. In the HSPICE Monte Carlo Analysis form, select **File > Load Setup** to open the Loading State window and select **montedemostate1**. After the state is loaded, the state information is displayed in the HSPICE Monte Carlo Analysis form. It contains “monte” information in the **Setup** tab for DC, AC and Tran analyses, the “AGAUSS” parameter “deltaL”, and includes a measurement file in the **Outputs** tab.

Chapter 1: Quick-Start Tutorial

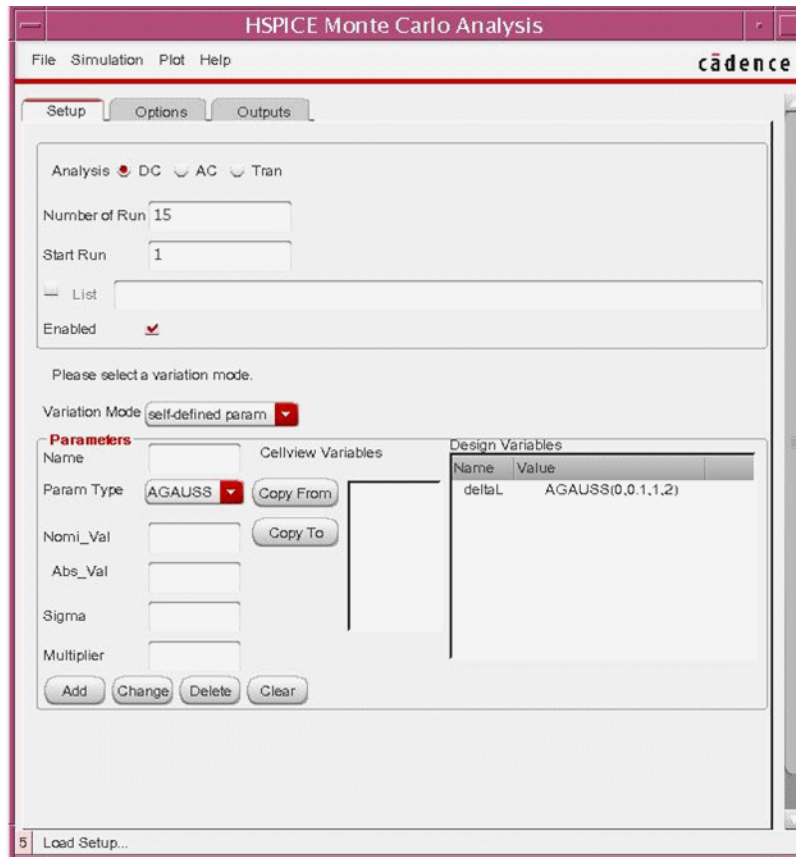
Task 9: Set Up and Run a Monte Carlo Simulation



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Chapter 1: Quick-Start Tutorial

Task 9: Set Up and Run a Monte Carlo Simulation

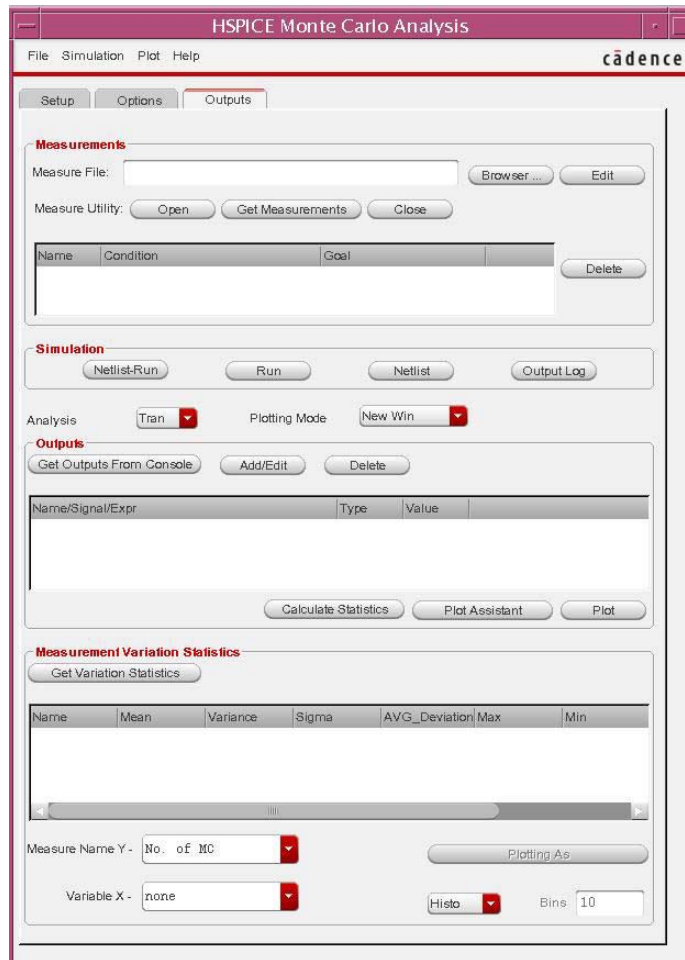


© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

5. Click the **Outputs** tab; then click **Get Outputs From Console**. The current setup of outputs (“/M1/D” and “/out”) is displayed. These outputs will be automatically plotted after the Monte Carlo simulation is completed.

Chapter 1: Quick-Start Tutorial

Task 9: Set Up and Run a Monte Carlo Simulation

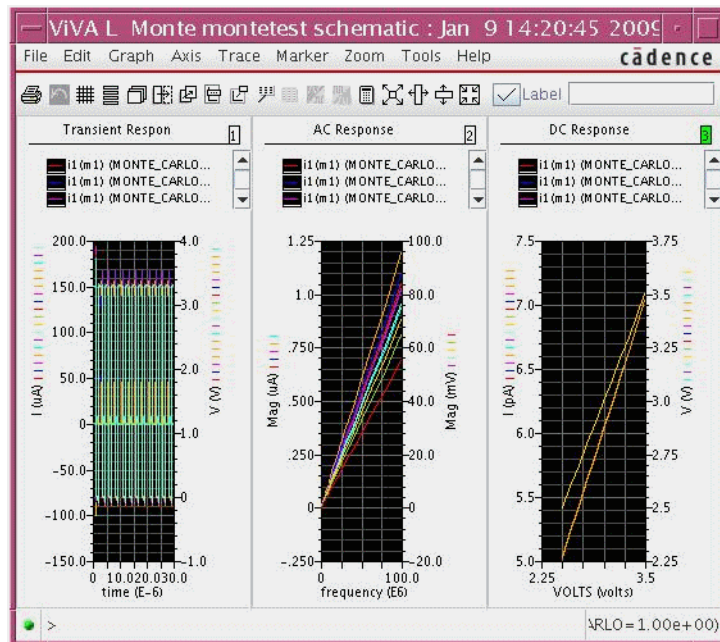


vide. Printed with permission.

6. In the **Outputs** tab, click the **Run Sim** button to run the Monte Carlo simulation. The tool starts netlisting and invokes the HSPICE simulator to run this Monte Carlo simulation. The netlist file and HSPICE simulation log are displayed during the run. After the simulation succeeds, the output “/M1/D” and “/out” waveforms are automatically plotted to a ViVA waveform window.

Chapter 1: Quick-Start Tutorial

Task 9: Set Up and Run a Monte Carlo Simulation

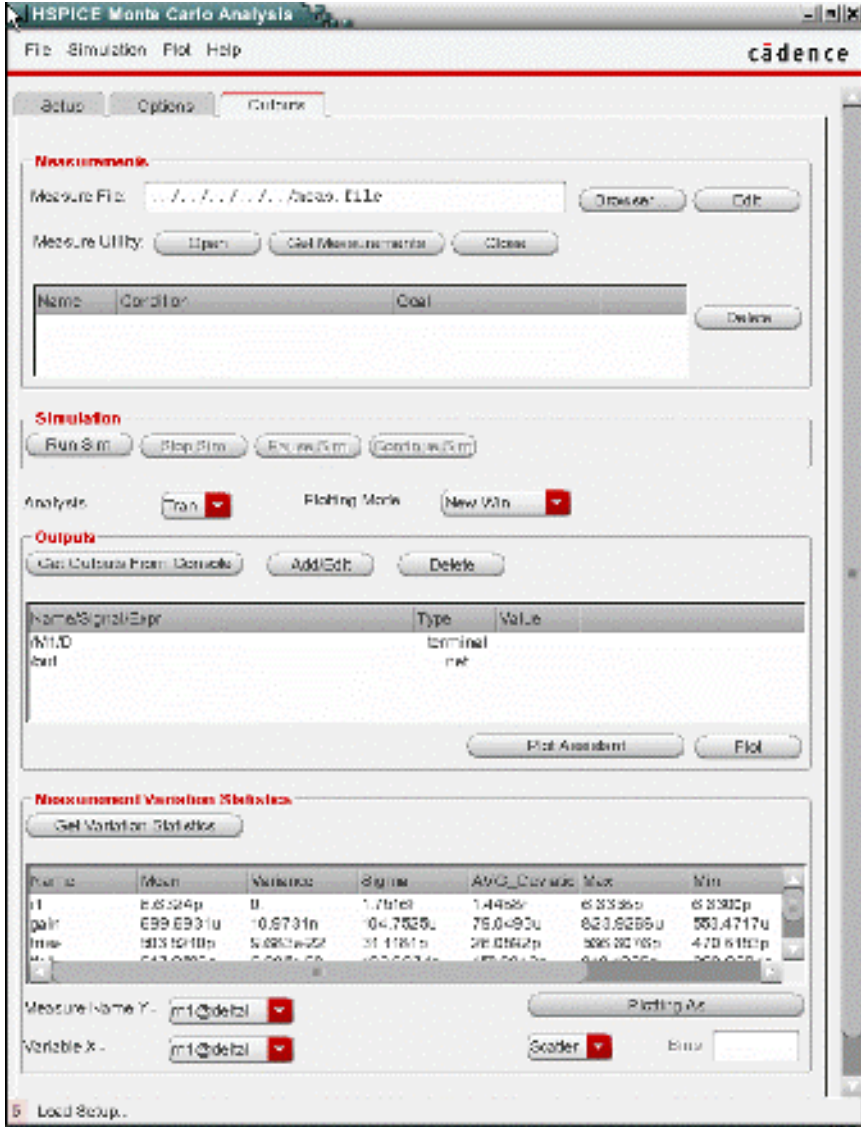


© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

7. On the **Outputs** tab, use the **Get Variation Statistics** button to load the simulation measurement results. To plot the measurement results, combine the use of the
 - **Analysis** section controls
 - **Measurement Variable - Y** dropdown menu
 - **Variable - X** dropdown menu
 - **Curve type**
 - **Bins** and
 - **Plotting As** button
 - Alternatively, launch the HSPICE Measurement Utility (added with the 2009.09 release) by clicking **Open**. See [Chapter 12, HSPICE Measurement Utility](#) for a discussion of this feature.

The variation statistics shown on the following **Outputs** tab are plotted in the succeeding waveform.

Task 9: Set Up and Run a Monte Carlo Simulation

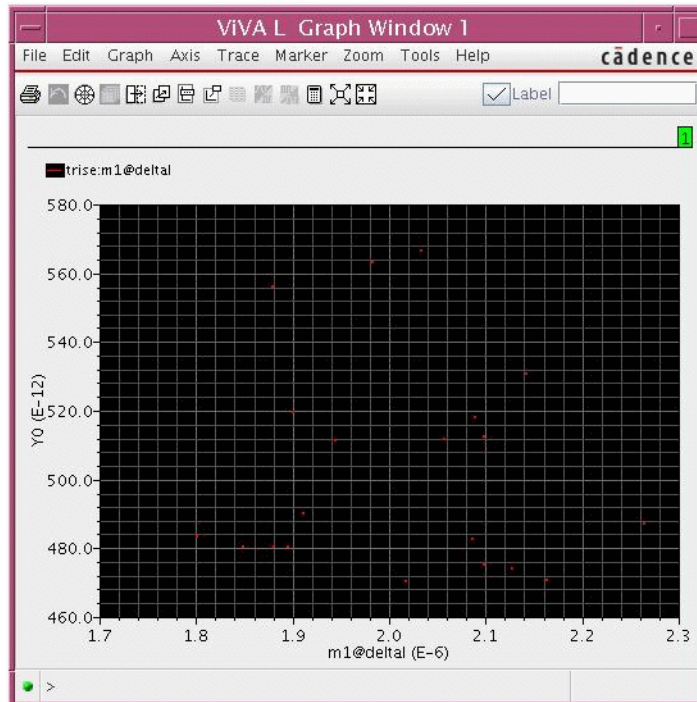


© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

In the following graphic, the scatter waveform of “trise” VS. “m1 @delta” with tran simulation results is plotted.

Chapter 1: Quick-Start Tutorial

Task 10: Run a Corner Analysis



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

8. In the HSPICE Monte Carlo Analysis form, select File > Save Ocean Script to save this entire simulation session to an OCEAN script (.ocn) in order to rerun it in a batch Ocean mode. See [Appendix B, OCEAN API Functions for HSPICE Monte Carlo Analysis](#).

Task 10: Run a Corner Analysis

This task uses a demo case located in the HSPICE installation directory (`$installDir/demo/hspice/aa_integ/Corner_Demo_61`). Users can find a corresponding IC51 demo case at `$installDir/demo/hspice/aa_integ/Corner_Demo_51`.

The demonstration cases under directories `Corner_Demo_61` and `Corner_Demo_51` are provided to introduce the usage of the HSPICE-ADE Corner Analysis tool. This feature is released with C-2009.09.

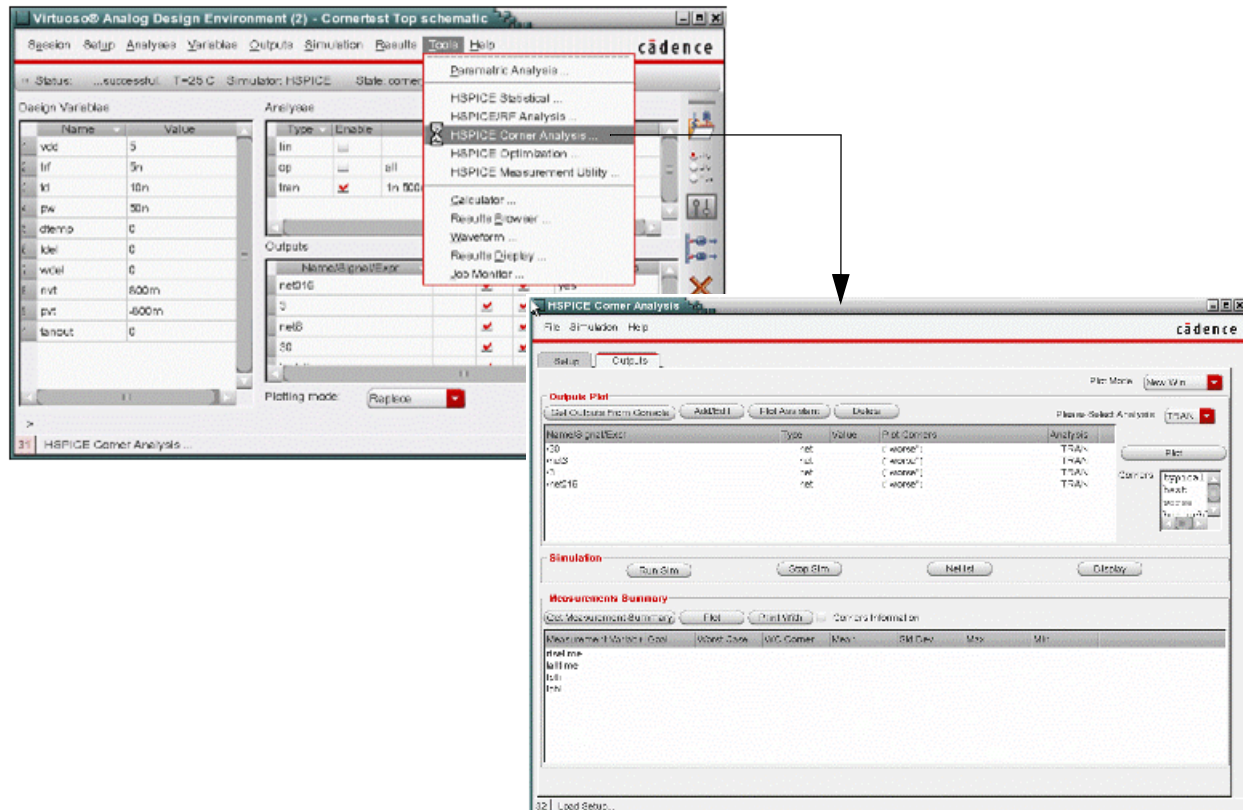
Follow these steps to invoke and run Corner analysis using HSPICE-ADE.

1. Locate `Corner_Demo_61` and open the case design:

- Make a local copy of the demo case, change directory to the Corner_Demo directory, and start by typing `icms &` to open the Command Interpreter Window (CIW).
 - Select **Tools > Library Manager** to display the Library Manager dialog.
 - In this window, open Cornertest/Top/schematic by selecting **Cornertest** from the **Library** column and **Top** from the **Cell** column. Then double-click on **schematic** in the **View** column to display the schematic view window.
2. Launch the Environment: In the Schematic window select **Launch > ADE L** to open the Environment Console (with banner text Cornertest Top schematic).
 3. To properly initialize the Environment session, in the Environment Console, select **Setup > Simulator/Directory/Host**, and use the dropdown list to specify **HSPICE** as the Simulator. Leave the other fields as defaults and **OK** the form.
 4. In the Environment Console, select **Tools > HSPICE Corner Analysis** to display the HSPICE Corner Analysis form.

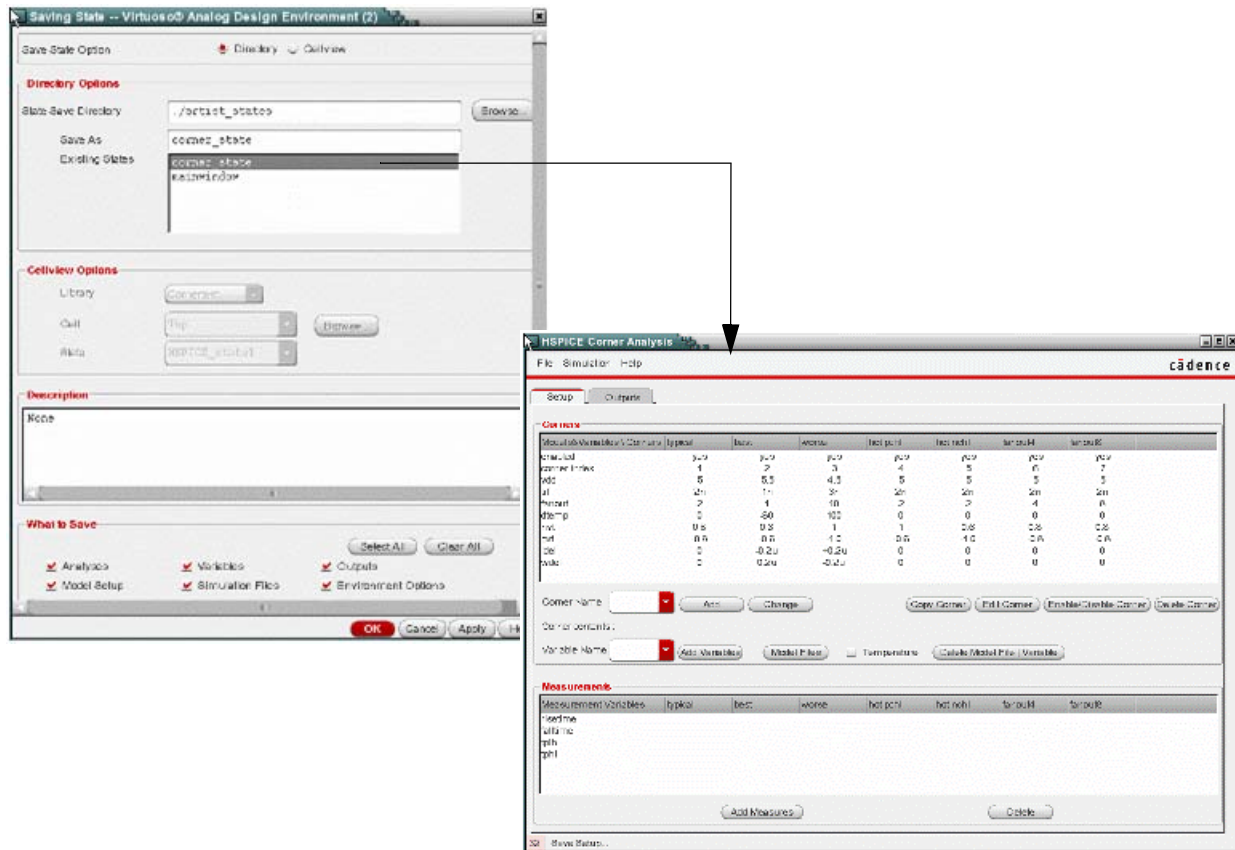
Chapter 1: Quick-Start Tutorial

Task 10: Run a Corner Analysis



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- In the HSPICE Corner Analysis form, select **File > Load Setup** to open the Loading State window. Select **corner_state** and click **OK** to load Corners and Measurements data into the fields of the **Setup** tab.



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

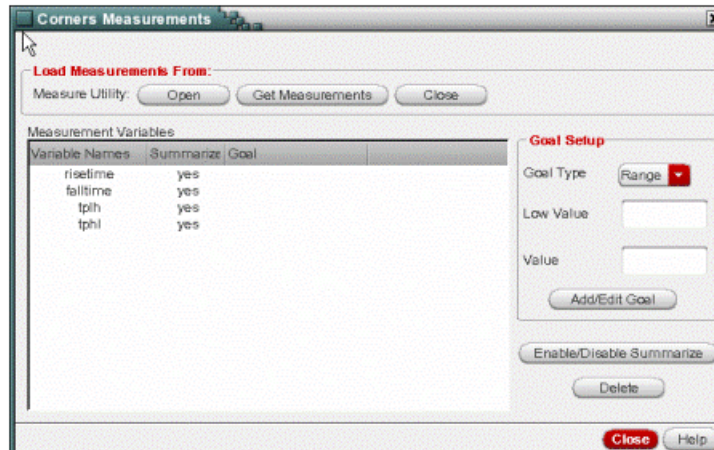
6. Edit or add measure statements by clicking the **Add Measures** button under the **Measurements** section (lower section in the setup window). The following measure statements become readily available in the **Measurements** section during the state load:

```
*----- measure statements for rise, fall, and propagation
delays
.meas risetime    trig par('v(3) -0.1*vdd')    val=0    rise=1
+               targ par('v(3) -0.9*vdd')    val=0    rise=1
.meas falltime    trig par('v(30)-0.9*vdd')    val=0    fall=1
+               targ par('v(30)-0.1*vdd')    val=0    fall=1
.meas tplh        trig par('v(2) -0.5*vdd')    val=0    fall=1
+               targ par('v(3) -0.5*vdd')    val=0    rise=1
.meas tphl        trig par('v(20)-0.5*vdd')    val=0    rise=1
+               targ par('v(30)-0.5*vdd')    val=0    fall=1
.measure delayr   trig at=0 targ v(voutr) val=2.5 rise=1
goal=100ns weight=10
```

Chapter 1: Quick-Start Tutorial

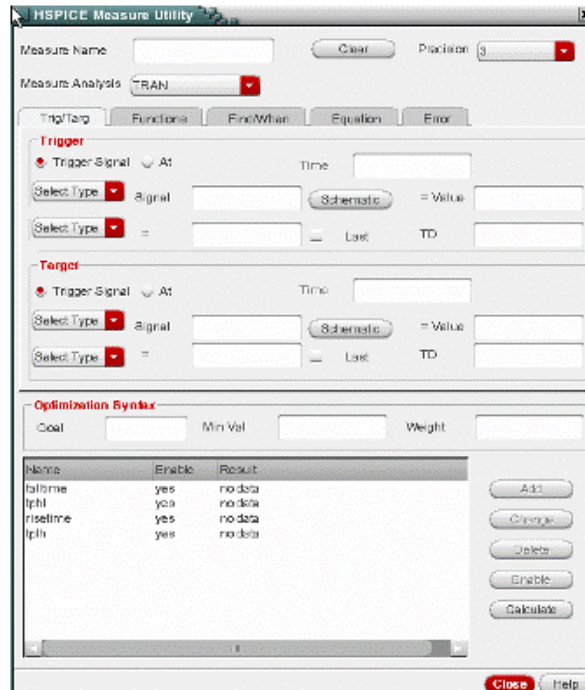
Task 10: Run a Corner Analysis

Click **Add Measures** to open the Corners Measurements form where you can add/edit measure goals for existing measures.



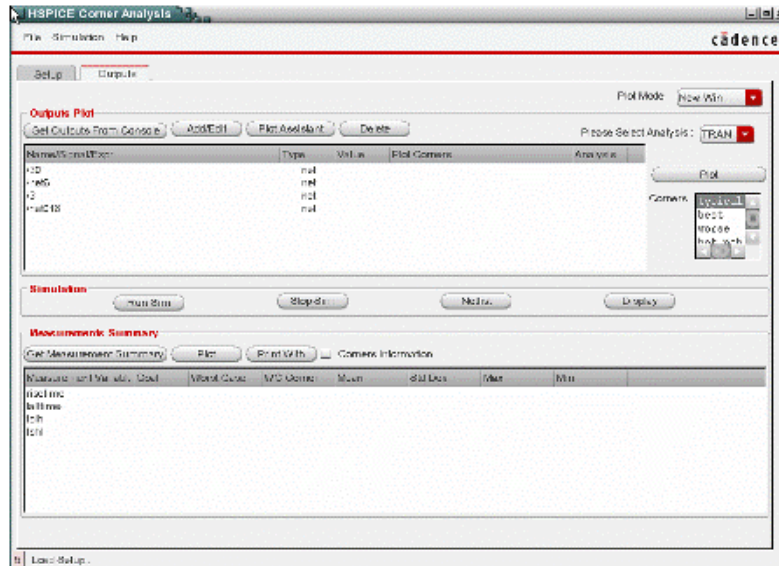
© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

7. Click the **Open** button to launch the HSPICE Measure Utility (also available from the Environment Console under the **Tools** menu) to input or modify the measures.



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

8. On the HSPICE Corner Analysis form, click the **Outputs** tab, then click the **Get Outputs From Console** button. The current setup of outputs (/30, /net6, /3 and /net016) is displayed. These outputs are automatically plotted after the Corner simulation is completed.

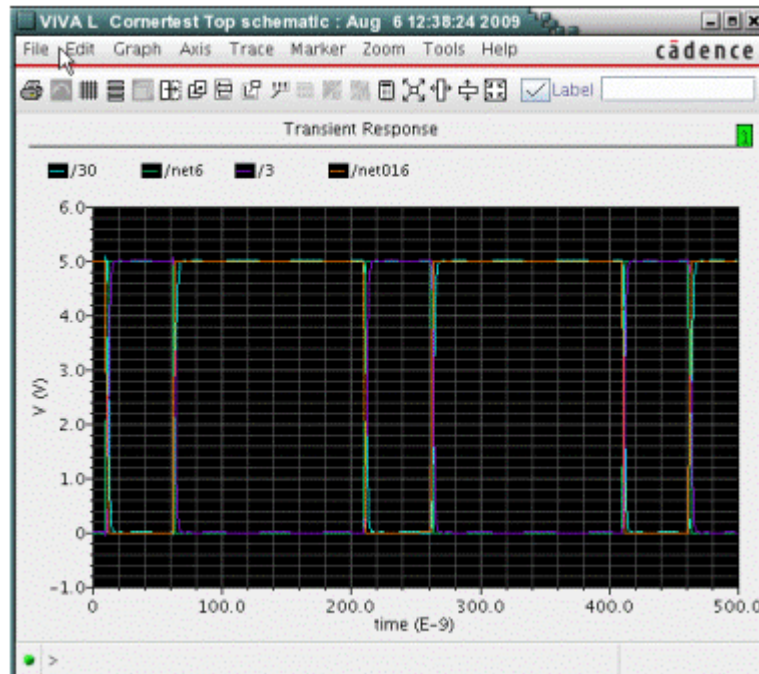


© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

9. On the **Simulation** pane of the **Outputs** tab, click the **Run Sim** button to run the Corner simulation. The tool starts netlisting and invokes the HSPICE simulator to run the simulation. The netlist file and HSPICE simulation log are displayed during the run. After the simulation succeeds, the output /30, /net6, /3 and /net016 waveforms are automatically plotted to a ViVA waveform window.

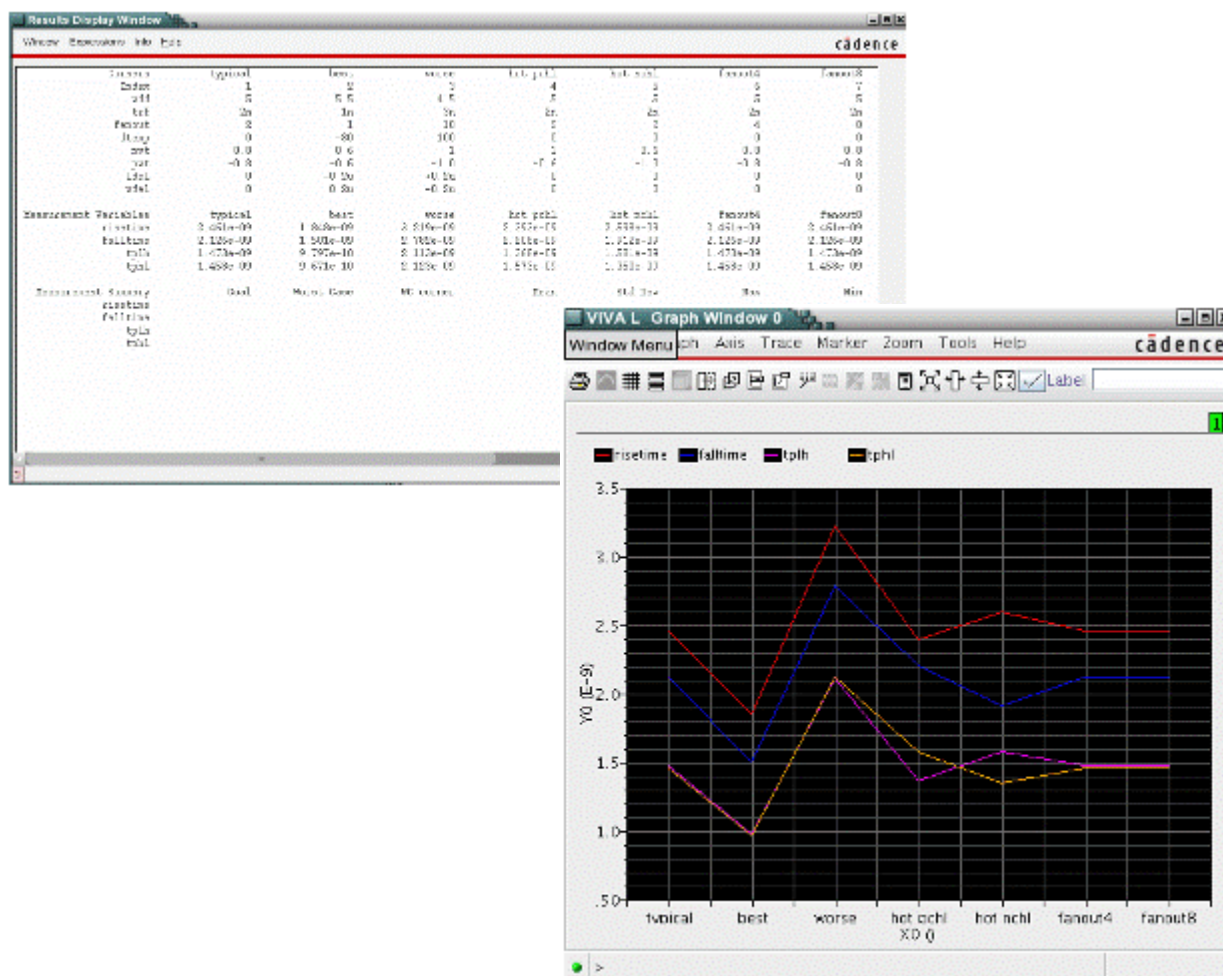
Chapter 1: Quick-Start Tutorial

Task 10: Run a Corner Analysis



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

10. In the **Outputs** tab, select multiple outputs and multiple corners by pressing the Ctrl key to plot them as required. Select the plot mode using the **Plot Mode** button.
11. Using the **Measurements** section of the **Outputs** tab, choose measurements from the **Measurements Summary** section to plot measurement variables and print measure values in separate windows (see below).



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

This concludes the Corners Analysis tutorial.

Task 11: Run an RF Analysis

This task uses a demo case located in the HSPICE installation demo cases directory (\$installdir/demo/hspice/aa_integ/Mixer_Demo_61).

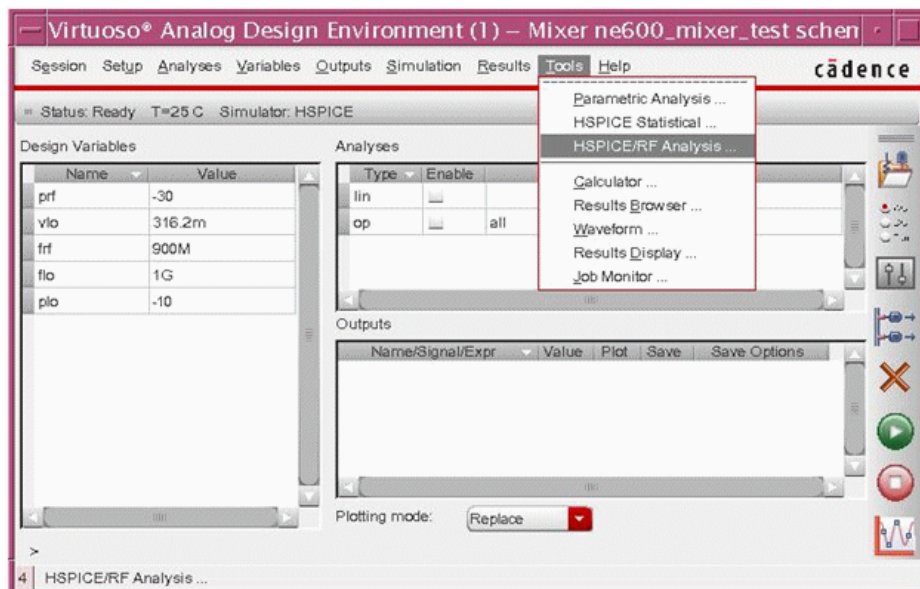
The demonstration cases under the directory Mixer_Demo_61 is provided to introduce the usage of HSPICE-ADE Integration HSPICE RF Analysis. This feature is released with HSPICE version C-2009.03.

Chapter 1: Quick-Start Tutorial

Task 11: Run an RF Analysis

Follow these steps to invoke and run the demonstration case for RF.

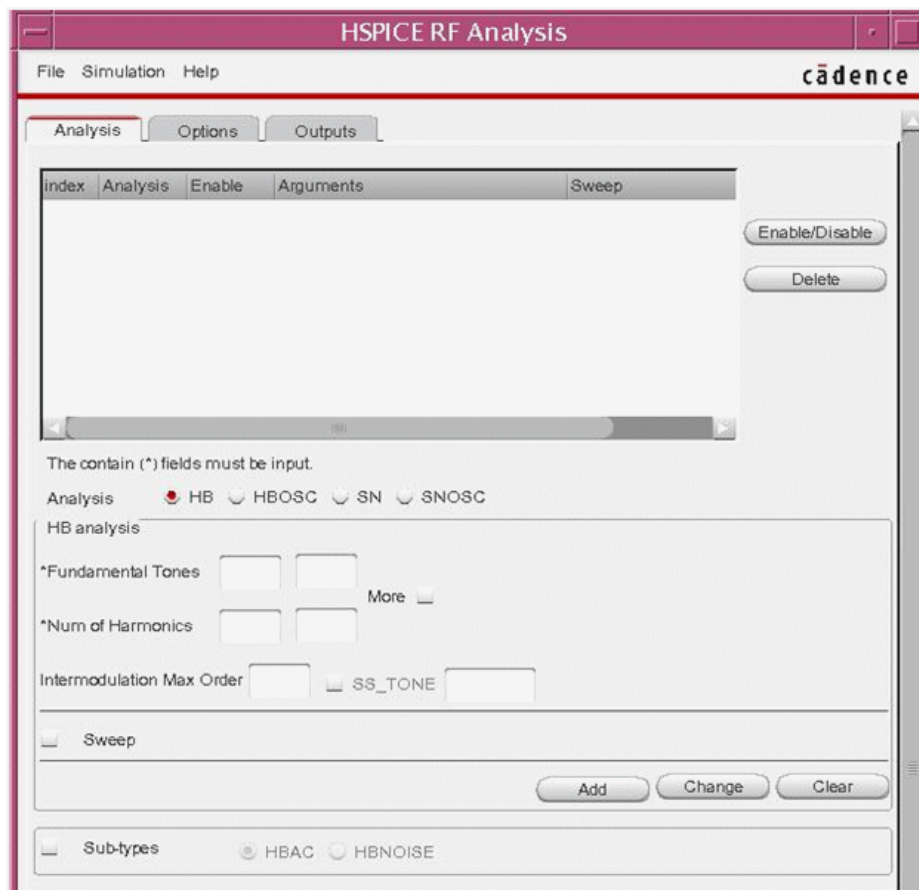
1. Locate Mixer_Demo_61 and open the case design:
 - Make a local copy of the demo case, change directory to the Mixer_Demo directory, and start by typing **icms &**. The CIW (Command Interpreter Window) opens.
 - Select Tools > Library Manager to display the Library Manager dialog.
 - In this window, open Mixer/ne600_mixer_test by selecting **Mixer** from the Library column and **ne600_mixer_test** from the Cell column. Then double-click on **schematic** in the View column. The schematic view window is displayed.
2. Open the Environment: In the Schematic window select Launch > ADE L. The Environment Console (with banner text “Mixer ne600_mixer_test schematic”) opens.



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

In the Environment Console, select Setup > Simulator/Directory/Host, and use the dropdown list to specify **HSPICE** as the Simulator. Leave the other fields as defaults and OK the form. The Environment session is now properly initialized.

3. In the Environment Console, select Tools > HSPICERF Analysis to display the HSPICE RF Analysis form.

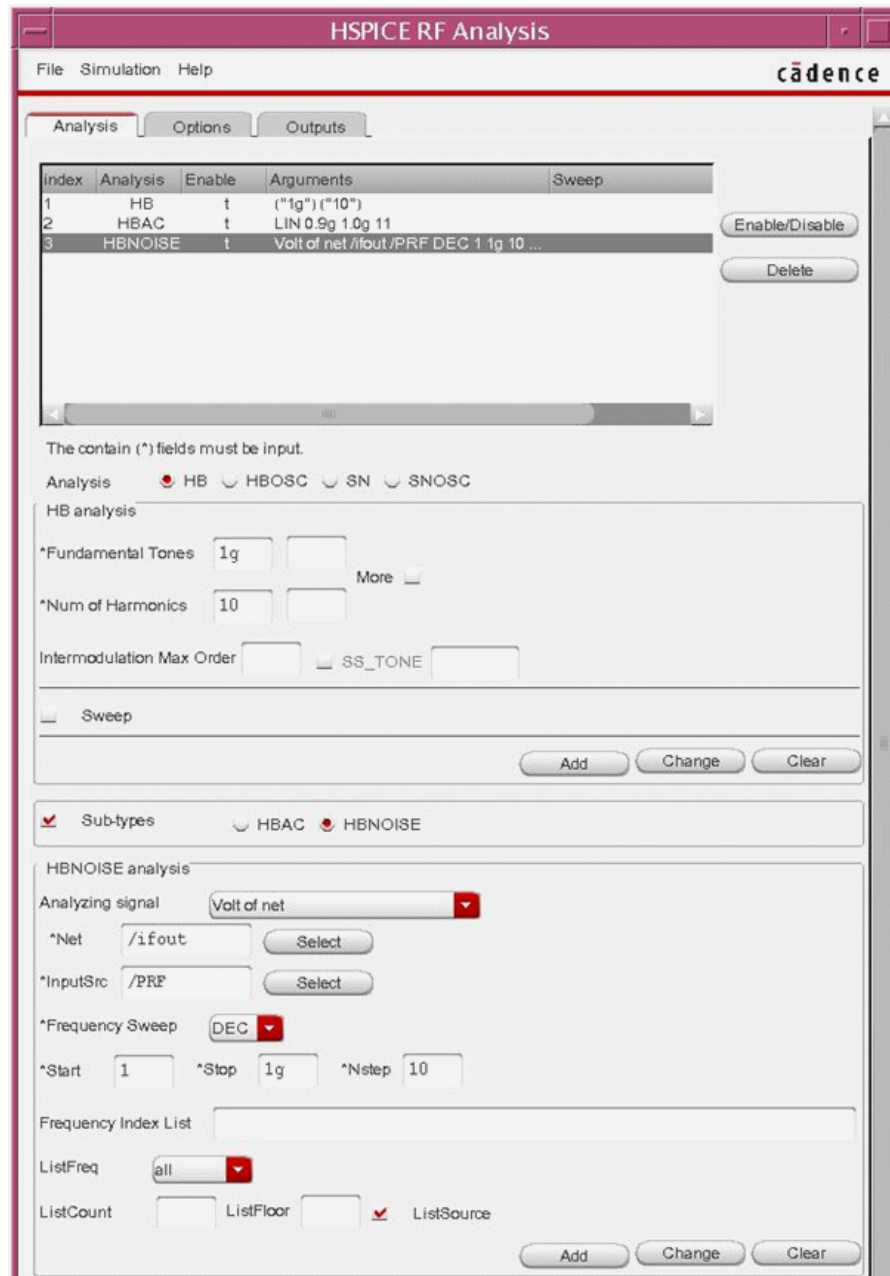


© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

4. In the HSPICE RF Analysis form, select File > Load Setup to open the Loading State window and select **rfdemohbac**. After the state is loaded, the states information will be displayed in the HSPICE RF Analysis form as displayed in the following graphic. It contains “hb”, “hbac” and “hbnoise” analyses in the Analysis tab for use in HSPICE RF simulation to come.

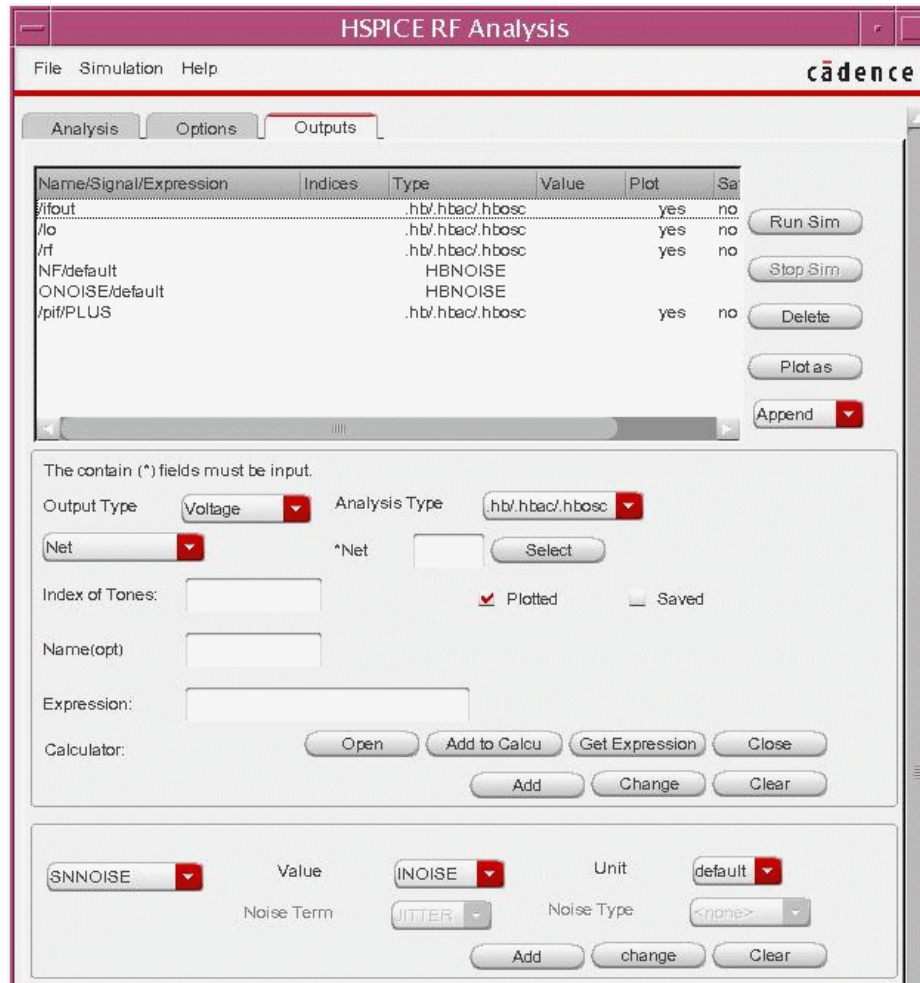
Chapter 1: Quick-Start Tutorial

Task 11: Run an RF Analysis



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

The Outputs tab also is populated with output data on the Outputs tab.

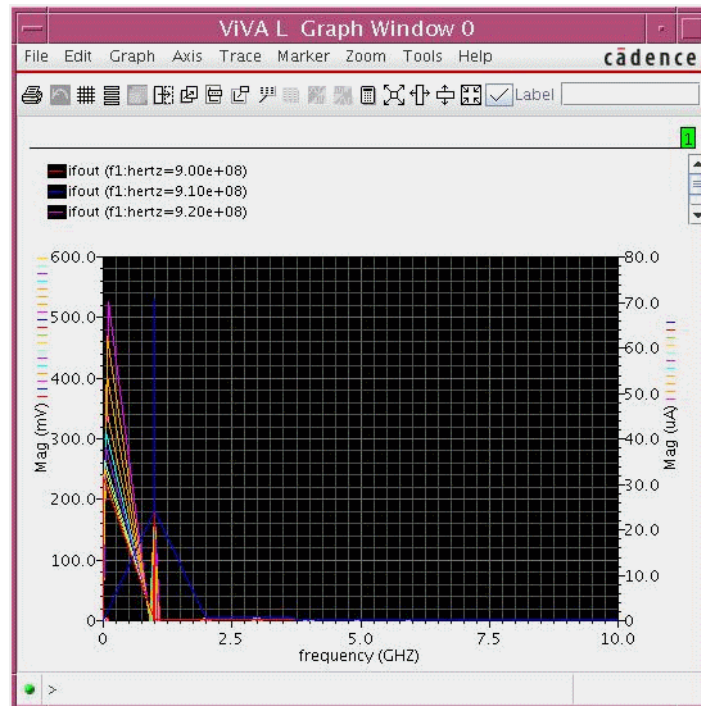


© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

5. In the Outputs tab, click the **Run Sim** button to run the HSPICE RF simulation. The tool starts netlisting and invokes the HSPICE simulator to run this HSPICE RF simulation. The netlist file and HSPICE simulation log are displayed during the run. After the simulation succeeds, the output waveforms are automatically plotted to a ViVA waveform window as shown in the following image.

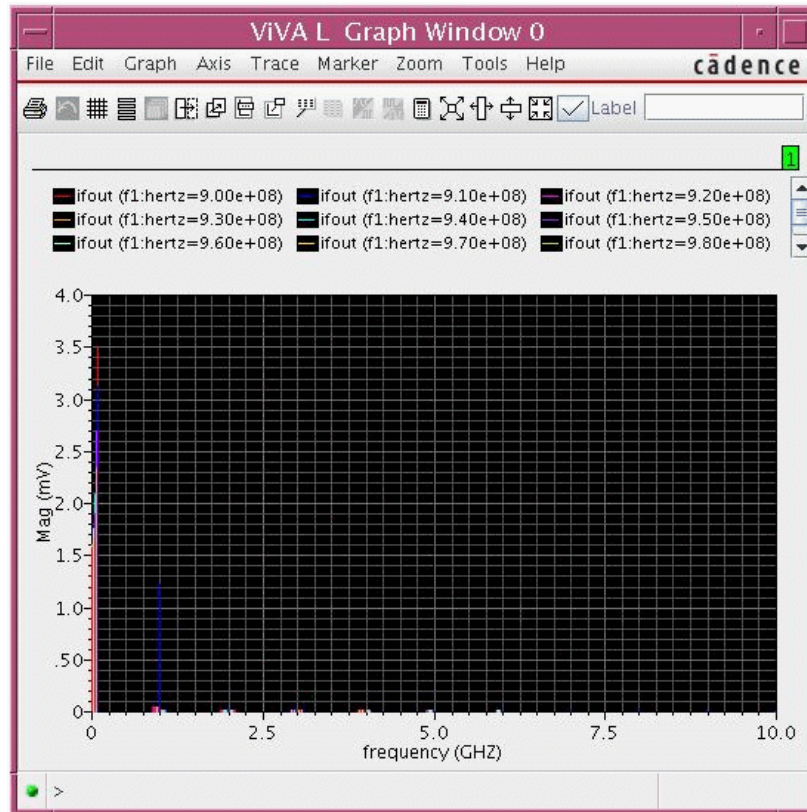
Chapter 1: Quick-Start Tutorial

Task 11: Run an RF Analysis



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

6. On the Outputs tab, select an output in the **Output Reports** field; then click the **Plot as** button to plot a single signal or expression. For example, the following graphic shows the “/ifout” waveform:



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

7. In the HSPICE RF Analysis form, select File > Save Ocean Script to save this entire simulation session to an OCEAN script (.ocn) in order to rerun it in a batch Ocean mode. See [Appendix B, OCEAN API Functions for HSPICE Monte Carlo Analysis](#).

Note: This RF demo also includes another state, **rfdemosnac**, which demonstrates an example for Shooting Newton, including .SN analysis, .SNAC analysis, and .SNNOISE analysis. In addition, the PLL_Demo_61 and PLL_Demo_51 cases also include an HSPICE RF state demonstrating examples of .HBOSC, .SNOSC and PHASENOISE HSPICE RF analyses. You can follow the preceding tutorial procedure to become familiar with use of the HSPICE RF feature in the HSPICE-ADE integration for these analyses.

Task 12: Run an Optimization Analysis

This task uses a demo case located in HSPICE installation directory (\$installDir/demo/hspice/aa_integ/Optimization_Demo_61). Users can find a corresponding IC51 demo case at \$installDir/demo/hspice/aa_integ/Optimization_Demo_51.

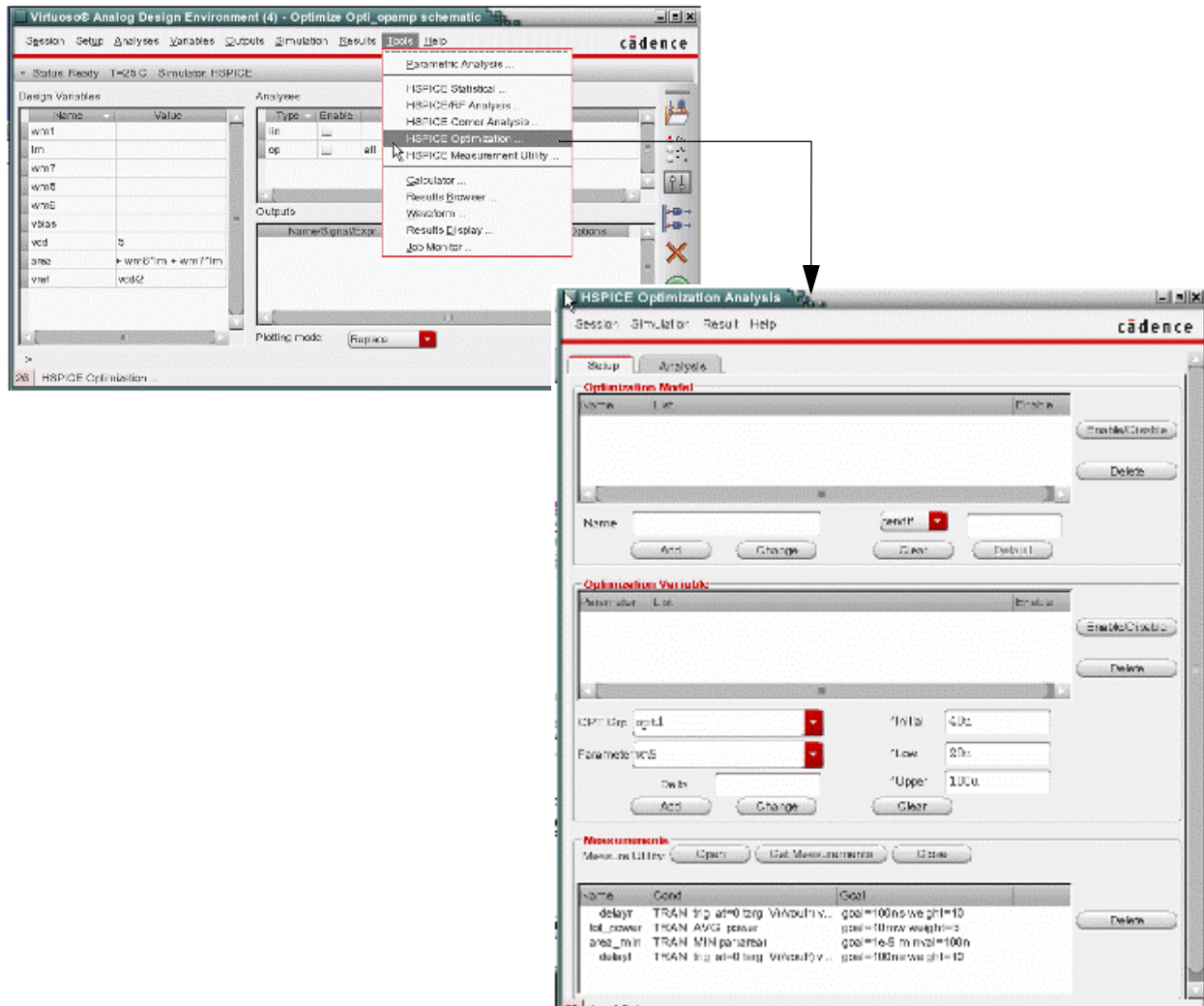
The demonstration cases under directories Optimization_Demo_61 and Optimization_Demo_51 are provided to introduce the usage of HSPICE-ADE Optimization Analysis tool. This feature is released with C-2009.09.

Follow these steps to invoke and run Optimization analysis using HSPICE-ADE.

1. Locate Corner_Demo_61 and open the case design:
 - Make a local copy of the demo case, change directory to the Corner_Demo directory, and start by typing `icms &` to open the Command Interpreter Window (CIW).
 - Select **Tools > Library Manager** to display the Library Manager dialog.
 - In this window, open Optimize/Opti_opamp/schematic by selecting **Optimize** from the **Library** column and **Opti_opamp** from the **Cell** column. Then double-click on **schematic** in the **View** column to display the schematic view window.
2. Launch the Environment: In the Schematic window select **Launch > ADE L** to open the Environment Console (with banner text Optimize Opti_opamp schematic).
3. To properly initialize the Environment session, in the Environment Console, select **Setup > Simulator/Directory/Host**, and use the dropdown list to specify **HSPICE** as the Simulator. Leave the other fields as defaults and **OK** the form.
4. In the Environment Console, select **Tools > HSPICE Corner Analysis** to display the HSPICE Optimization Analysis form.

Chapter 1: Quick-Start Tutorial

Task 12: Run an Optimization Analysis



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- In the HSPICE Optimization Analysis form, select **Session > Load Setup** to open the Loading State window. Select **Full_Demo_state** and click **OK** to import all setup data (optimization model and optimization variables and measurements into the **Setup** tab of the HSPICE Optimization Analysis form.

The Optimization model section corresponds to following line in the final netlist:

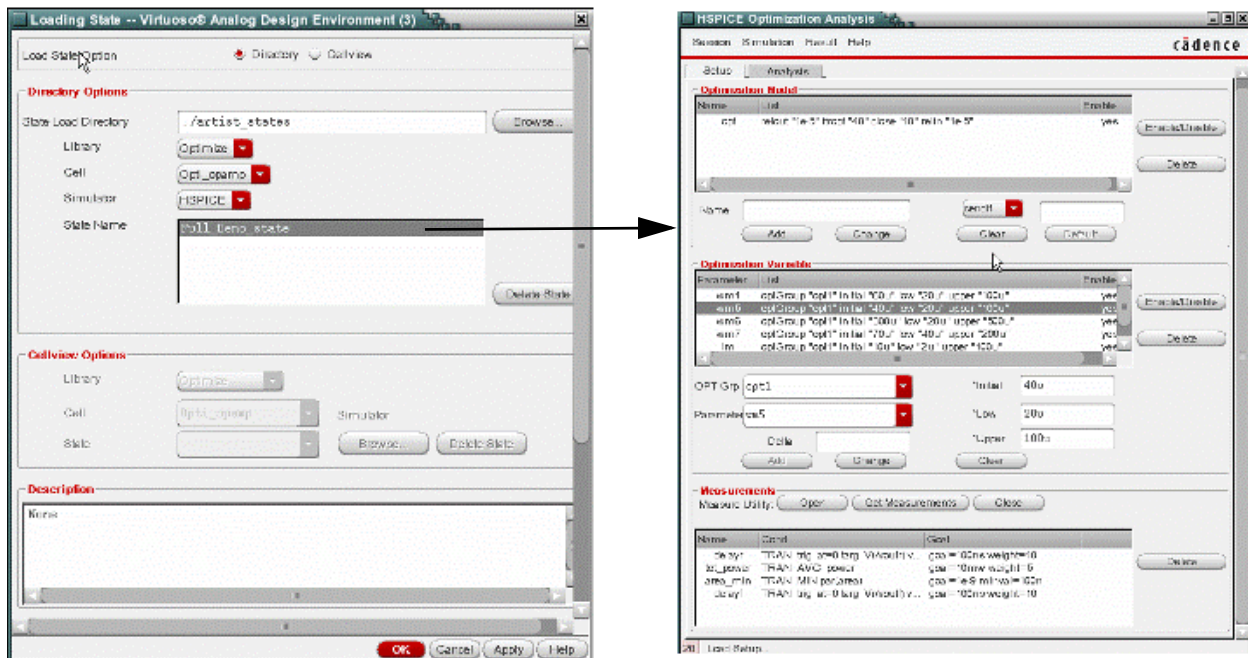
```
.model opt opt itropt=40 close=10 relin=1e-5 relout=1e-5
```

Chapter 1: Quick-Start Tutorial

Task 12: Run an Optimization Analysis

The Optimization variables section corresponds to following lines in the final netlist:

```
.param wm1=opt1(60u,20u,100u)
+      wm5=opt1(40u,20u,100u)
+      wm6=opt1(300u,20u,500u)
+      wm7=opt1(70u,40u,200u)
+      lm=opt1(10u,2u,100u)
+      bias=opt1(2.2,1.2,3.0)
```



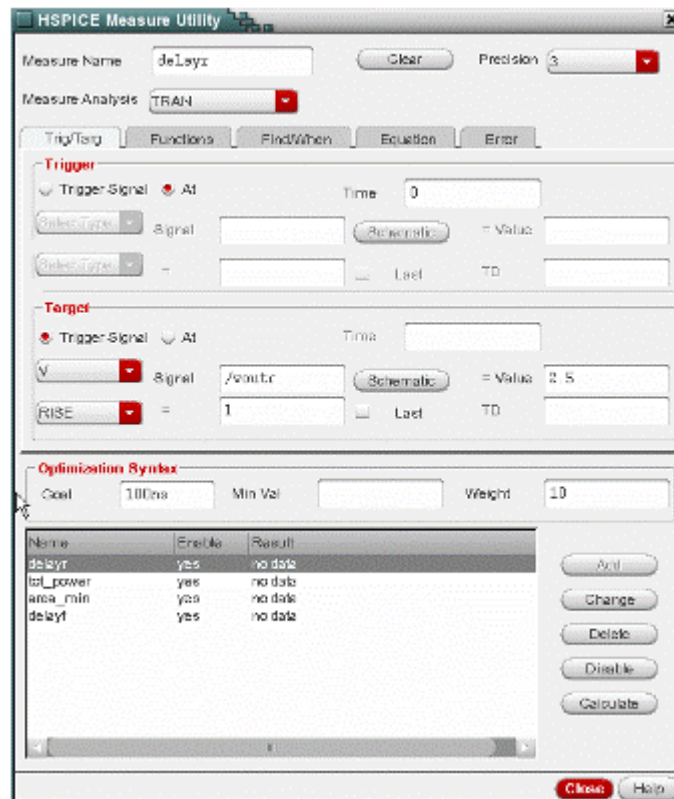
© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

6. Edit or add measure statements by selecting a statement and clicking on the **Open** button below the **Measurements** section (bottom section in the **Setup** tab). This action opens the HSPICE Measure Utility.

Using the tutorial state you find the following measure statements:

```
.measure delayr trig at=0 targ v(voutr) val=2.5 rise=1
goal=100ns weight=10
.measure delayf trig at=0 targ v(voutf) val=2.5 fall=1
goal=100ns weight=10
.measure tot_power avg power goal=10mw weight=5
.measure area_min min par(area) goal=1e-9 minval=100n
```

The HSPICE Measure Utility provides a convenient tool to edit or add more measurements. See [Chapter 12, HSPICE Measurement Utility](#) for a detailed description of this feature.

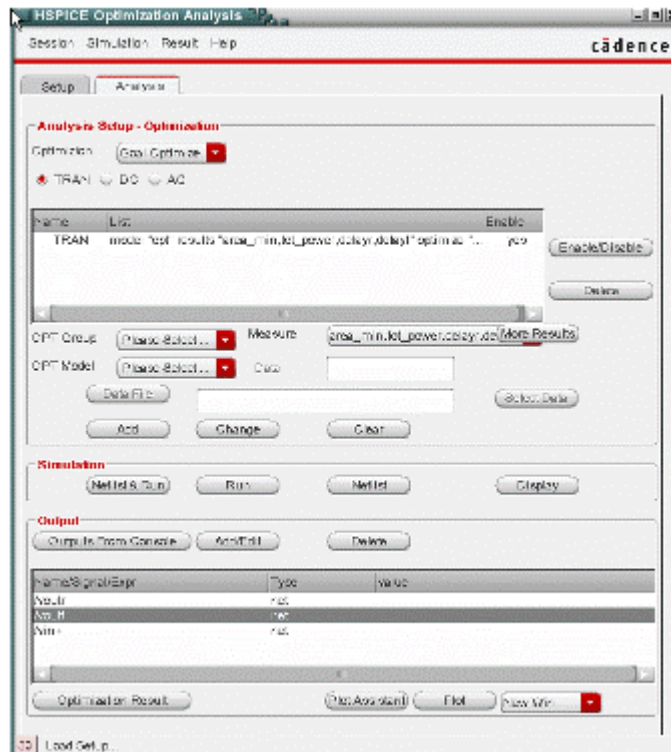


© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

7. In the HSPICE Optimization Analysis form, click the **Analysis** tab and then click on **Get Outputs From Console**. The current setup of outputs (/Voutf, /Voutf, and /Vin+) is displayed. These outputs are automatically plotted after the Optimization simulation is completed.

Chapter 1: Quick-Start Tutorial

Task 12: Run an Optimization Analysis



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

8. As shown in the **Analysis** tab, above, you can select multiple measure parameters in the optimization statement by using the **More Results** button. Press the Ctrl key and select the desired measure parameters from a list such as the following accessed from the tutorial load state you are using:

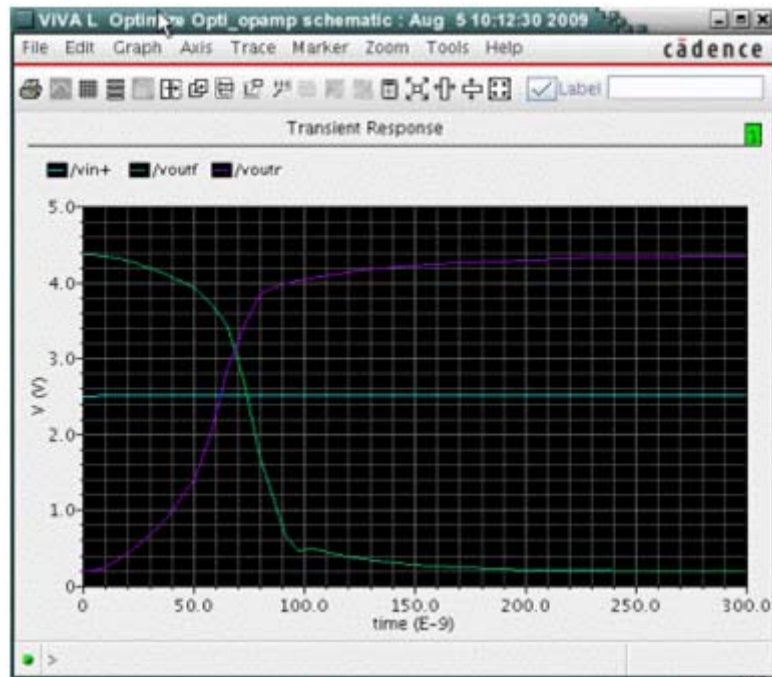
```
.tran 2.5n 300n sweep optimize=opt1
+ results=delayr,delayf,tot_power,area_min model=opt
```

In the printed measure report shown following Step 10 you can see resulting values for the selected measure parameters (above) in the section called Other measures.

9. In the **Analysis** tab, click **Netlist & Run** to run the Optimization simulation. The tool starts netlisting and invokes the HSPICE simulator to run the simulation. The netlist file and HSPICE simulation log are displayed during the run. After the simulation succeeds, the output /Voutr, /Voutf, and /Vin+ waveforms are automatically plotted to a ViVA waveform window.

Chapter 1: Quick-Start Tutorial

Task 12: Run an Optimization Analysis



© 2009, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- On the **Analysis** tab below the **Output** section, click the **Optimization Result** button to see optimization parameter values and measure values as shown below.

Optimization parameter(s)		
Name	Value	
vbias	2.9984	
lm	7.694e-06	
wm7	1.388e-04	
wm6	7.128e-05	
wm5	2.253e-05	
wm1	1.000e-04	
Optimization measure(s)		
Name	Value	Goal
Other measure(s)		
Name	Value	
delayr	6.230e-08	
tot_power	8.207e-03	
area_min	4.867e-09	
delayf	7.429e-08	

Chapter 1: Quick-Start Tutorial

Task 12: Run an Optimization Analysis

Updating Libraries and Tool Filter

This chapter discusses the HSPICE library conversion script, as well as the changes to the library and any migration issues (such as incompatibilities with existing states) It also addresses the need to add HSPICE to the Tool Filter.

These topics are discussed in the following sections.

- [Library Conversion Script](#)
- [HSPICE and analogLib](#)
- [Adding the HSPICE Simulator to the Tool Filter](#)

Library Conversion Script

A conversion script is provided to add HSPICE-Environment integration information to the existing Environment analogLib. The conversion script can take customer-created primitives or PDKs to a level of support that is needed in the HSPICE-Environment integration.

The script allows you to apply the conversion updates to an existing library. The conversion consists of adding missing CDF data to the cells and creating or updating the hspice stop views and HSPICE simInfo data.

The following topics are discussed in these sections:

- [Running the Library Update Utility](#)

Running the Library Update Utility

In order to run HSPICE within the Analog Design Environment, the 'analogLib' must have the necessary HSPICE simulator information. Use the following

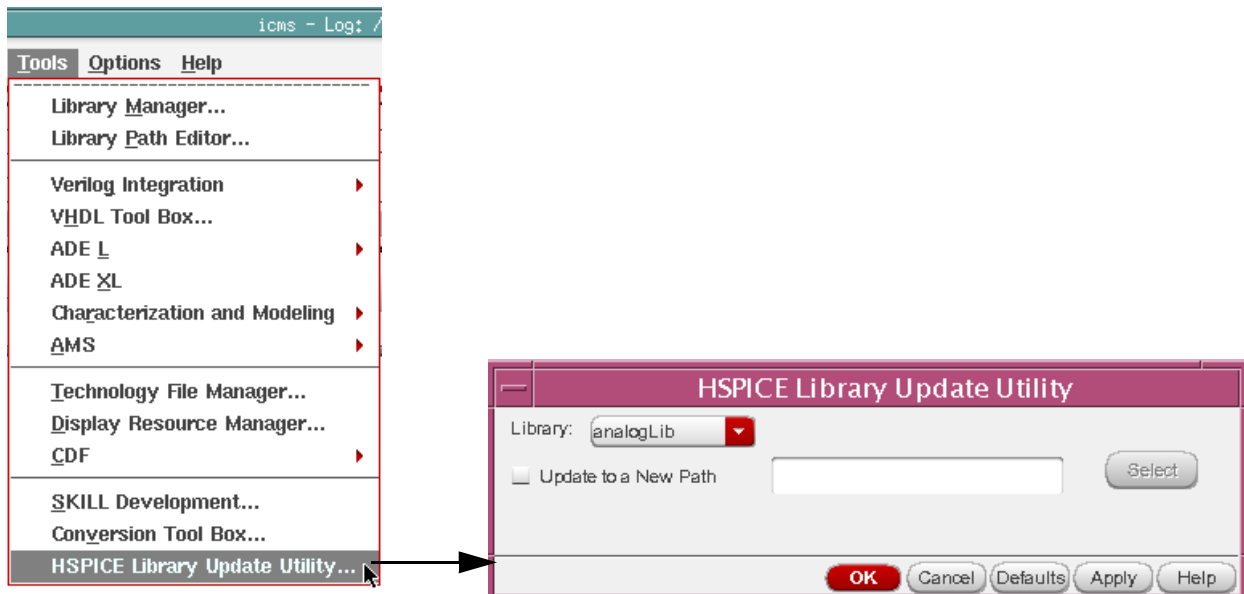
steps to update it from which you are instantiating simulation primitives in your designs.

Note: The HSPICE Library Update Utility updates recognized components with hspiceD simInfo. Most of the components in analogLib are recognized. If components don't fit this category or don't have hspiceD simInfo, you will need to update them manually for them to be netlisted. The update utility also adds HSPICE-specific components with HSPICE simInfo such as the HSPICE S-parameter model, PAT, and Pseudo Random BIT generator sources, etc.

Follow these steps to update your libraries from which you are instantiating simulation primitives in your designs.

1. Start `icms` and in the CIW, select **Tools > HSPICE Library Update Utility**.

Note: If you don't have this item in the **Tools** pulldown, your `.cdsinit` file may not be properly updated.



2. Select the name of the library that you wish to update in the **Library** dropdown list.

3. You can instruct the Library Update Utility directly to update existing 'analogLib' or Update to a new path before updating. To do this, enable **Update to a New Path** and specify a new path. Your library will be copied to the new path and updates will be applied to the new 'anaogLib'.

Note: A. If directly updating the 'analogLib' in the Cadence Virtuoso installation tree, make sure you have 'write' permission. (Normally, this is done by a CAE for a onetime conversion. Other end users need not update it.)
B. If you update the 'analogLib' through **Update to a New Path**, the 'analogLib' in cds.lib or lib.defs is automatically updated with the new path you specified.

For example, before updating your 'analogLib' in your cds.lib would read as:

```
DEFINE analogLib
/remote/apps/ic61/linux/tools/dfII/cds/
artist/analogLib
```

After updating, it would be

```
DEFINE analogLib
/remote/home/zyx/newpath/analogLib
```

4. If you want to include HSPICE-specific cells such as PAT/LFSR sources and so on, please select **Include HSPICE specific cells**.
5. Click **OK** or **Apply** to run the Update Utility.
6. Check the CDS.log file for unrecognized cells. If any were encountered, you will see a message such as:

Cell: nsoi unable to create HSPICE simInfo because there is no reference simInfo.
7. Select **Tools > Library Path Editor** to verify the 'analogLib' new path if you selected **Update to a New Path**.
8. Once the Library Update Utility has completed, you can continue to use icms or you may exit.

Troubleshooting

If you don't see **HSPICE** listed in the **Simulator** choices pulldown, your .cdsinit file may not be properly updated.

What the Script Does to the analogLib

The conversion process updates the 'hspice' view (not a symbol) and adds HSPICE-specific component p. The 'analogLib' must be converted *before* using the HSPICE integration in ADE environment. The conversion parameters to support HSPICE syntax ensure the correct netlisting for each cell in the 'analogLib'. If the source 'hspiceD' *netlistProcedure* and *componentName* are recognized by the script, it adds HSPICE simInfo with HSPICE integration equivalents. If the hspiceD source does not already exist, it also creates new cellviews 'hspice' and 'symbol' for the cell to support HSPICE primitives. For an unconverted library, no other data is touched or removed. If a library has been previously converted, the script overwrites the same parameters and HSPICE simInfo that were previously added.

After the conversion, other simulator simInfo and cellviews in 'analogLib' (such as spectre simInfo and spectre cellview) are never touched or removed.

What the Script Does on Libraries from 3rd Party PDKs

The update procedure for non-analogLib libraries only works on cells containing hspiceD simInfo and hspiceD cellviews. The script copies the hspiceD view to an hspice view and then copies the hspiceD simInfo to HSPICE simInfo.

Since the C-2009.09 release, the non-analogLib libraries such as a 3rd party PDK containing hspiceD simInfo and hspiceD cellview are supported directly without conversion. Users needn't use the library update script to convert these libraries.

HSPICE and analogLib

The standard component reference library is called analogLib, and the 'analogLib' must be converted by using the library update script to create HSPICE simInfo and hspice cellview with HSPICE primitive syntax to correctly netlist. The native library has been augmented to accommodate the HSPICE perspective. Missing HSPICE simulation views for components or missing components for HSPICE specific elements have been added for your simulation capability.

If the library update utility cannot convert a cell or encounters other issues, an output log groups the output information as follows:

During processing, the status of each cell conversion is given for each cell in the library as it is processed.

This status can be:

- Successfully converted - created HSPICE view
- Successfully converted - HSPICE view not created (no spectre, hspiceD or symbol view to create HSPICE view).
- Doesn't require conversion (schematic only symbol that doesn't netlist, such as gnd symbols).
- Unsupported cell
- Unsuccessfully converted - cell was unable to be matched to cell in library utility's database to be able to convert cell.

After processing, a summary is output detailing status of the library cells.

Example Conversion Messages

Cells successfully converted:

```
Cell ccvs: Successful conversion - created HSPICE simInfo AND  
hspice stop view
```

Cells that do not require conversion:

```
Copied cells for which conversion is not needed:  
gnd gnda gndd vcc vcca vccd vdd vdd_inherit vdda vddd  
vee veea veed vss vss_inherit vssa vssd
```

Cells that could not be converted:

```
(Corrective action: add the simInfo and stop view, if needed)  
MOS_a2d MOS_d2a TTL_a2d TTL_d2a bsim4 cmdmprobe corefragment delay  
dummy fourier fourier2ch fracpole ideal_balun msline nodeQuantity  
nsoi ntft phyres port3t psoi ptft scr pvccsp pvcvsp rcwireload  
scasubckt scccs sccvs simulinkCoupler spltswitch sp2tswitch  
sp3tswitch sp4tswitch
```

The following topics are discussed in these sections:

- [Splitting analogLib into 3 Output Libraries](#)
- [HSPICE Components Added or Converted](#)

Splitting analogLib into 3 Output Libraries

You can run the following SKILL routine from the Virtuoso CIW or batch to specify the names of the output libraries:

```
'snpsSplitAnalogLib(@optional (path "./") (anaLib  
"analogLib") (hspLib "hspiceLib") (speLib "spectreLib"))
```

If you do not specify names for the libraries, they default to the names in the routine. As indicated, “anaLib” is the common output library, while the other two receive only HSPICE or native simulator output.

HSPICE Components Added or Converted

Table 1 lists the devices that are either found in the Environment analogLib, or have been added to complete HSPICE support. Each of these components is listed in the Cell Name column. For each, the hspiceD netlisting procedure is listed, along with the componentName (if it exists) as found in the cell level CDF.

The last two columns list the HSPICE integration upgrade for the library component. In most cases, a library component existed in hspiceD, but has required augmentation to provide full HSPICE functionality. The HSPICE integration column notes augmentation.

Table 1 HSPICE Components in the Environment

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
bcs	hspiceDCompPrim	bcs	Augment	G-element: Behavioral Current Source
bjt504tnpn			Augment	Q-element: MEXTRAM 504 n-type BJT; Added HSPICE simInfo
bjt504tpnp			Augment	Q-element: MEXTRAM 504 p-type BJT; Added HSPICE simInfo
bsim4			Augment	M element: either n or p type, adding missing parameters

Table 1 HSPICE Components in the Environment (Continued)

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
bvs	hspiceDCompPrim	bvs		E-element: Behavioral Voltage Source; Added SCALE parameter
cap	hspiceDCompPrim	cap	Augment	C-element: Capacitor; Added all missing CDF parameters
cccap		cccap	New	C-element: Charge Conserving Capacitor
cccs	hspiceDCompPrim	cccs	Augment	F-element: Current Controlled Current Source; Added NPDELAY
ccvs	hspiceDCompPrim	ccvs	Augment	H-element: Current Controlled Voltage Source; Added NPDELAY
core	hspiceDCompPrim	core	Augment	K-element: Transformer core
delay		(vcvs) delay	Augment	E-element: vcvs delay, adding missing parameters
diode	hspiceDCompPrim	diode	Augment	D-element: Diode; Added simInfo and missing parameters
hspice_sparam			New	S-element: Added .model S-parameter statement into netlist
iam	hspiceDCompPrim	iam	Augment	I-element: Current Source Amplitude Modulation
idc	hspiceDCompPrim_idc	isrc	Augment	I-element: Current Source
ideal_balun		transformer	New	K-element: Balun Transformer
iexp	hspiceDCompPrim_iexp	isrc	Augment	I-element: Current Source Exponential
ibis_buffer		ibis_buffer	New	B-element: IBIS buffer
ind	hspiceDCompPrim	ind	Augment	L-element: Inductor; Added missing CDF parameters

Table 1 HSPICE Components in the Environment (Continued)

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
iopamp	hspiceDCompPrim	iopamp	Augment	E-element: OpAmp
ipat		ipat	New	I-element: Independent Current pattern source
iprbs		iprbs	New	I-element: Independent Current Pseudo Random-Bit Generator source; added Jitter source
iprobe	hspiceDCompPrim_vdc	vsrc	Augment	V-element: Current Probe
ipulse	hspiceDCompPrim_ipulse	isrc	Augment	I-element: Current Source Pulse
ipwl	hspiceDCompPrim_ipwl	isrc	Augment	I-element: Current Source Piece-Wise Linear
ipwlf	hspiceDCompPrim_pwlf	isrc	Augment	I-element: Current Source Piece-Wise Linear File
isffm	hspiceDCompPrim_isffm	isrc	Augment	I-element: Current Source Single Frequency FM
isin	hspiceDCompPrim_isin	isrc	Augment	I-element: Current Source Sine Wave
isource			Augment	I-element: Current Source Universal; Added simInfo and missing CDF parameters; add Jitter source
ivmrf		ivmrf	New	I-element: Independent Current Vector Modulated RF source
ixfmr	hspiceDCompPrim	ixfmr	Augment	I-element: Current Source Transformer
mind	hspiceDCompPrim	mind	Augment	K-element: Mutual Inductor
mos_a2d			Augment	Verilog-A support
mos_d2a			Augment	Verilog-A support

Table 1 HSPICE Components in the Environment (Continued)

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
mtline			Augment	W-element: Distributed Transmission Line; Added simInfo and missing CDF parameters
n1port			Augment	S-element: S-parameter Element 1-port; Added simInfo and missing CDF parameters
n2port			Augment	S-element: S-parameter Element 2-port; Added simInfo and missing CDF parameters
n3port			Augment	S-element: S-parameter Element 3-port; Added simInfo and missing CDF parameters
n4port			Augment	S-element: S-parameter Element 4-port; Added simInfo and missing CDF parameters
nbsim	hspiceDCompPrim	nmos	Augment	M-element: N-type BSIM; Added simInfo and missing CDF parameters
nbsim4	hspiceDCompPrim	nmos	Augment	M-element: N-type BSIM 4-terminal; Added simInfo and missing CDF parameters
ncs		ncs	New	G-element: Noise Current Source
njfet	hspiceDCompPrim	njfet	Augment	J-element: N-type JFET
nmes	hspiceDCompPrim	mesfet	Argument	J-element: N-type MESFET
nmes4	hspiceDCompPrim	mesfet	Augment	J-element: N-type MESFET 4-terminal
nmos	hspiceDCompPrim	nmos	Augment	M-element: N-type MOSFET; Added DELVTO

Table 1 HSPICE Components in the Environment (Continued)

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
nmos4	hspiceDCompPrim	nmos	Augment	M-element: N-type MOSFET 4-terminal; Added DELVTO
npn	hspiceDCompPrim	npn	Augment	Q-element: N-type BJT
nport			Augment	S-element: S-parameter Element N-port; Added simInfo and missing CDF parameters
nsoip		nsoip	New	M-element: N-type SOI Level57 BSIM3SOI
ntft		ntft	Augment	M element: Level 71, N type. adding missing parameters
nvs		nvs	New	E-element: Noise Voltage Source
pbsim	hspiceDCompPrim	nmos	Augment	M-element: P-type BSIM; Added simInfo and missing CDF parameters
pbsim4	hspiceDCompPrim	nmos	Augment	M-element: P-type BSIM 4-terminal; Added simInfo and missing CDF parameters
pcapacitor	hspiceDCompPrim	cap	Augment	C-element: Parasitic Capacitor; Added missing CDF parameters
pcccs			Augment	F-element: Polynomial Current Controlled Current Source; Added simInfo
pcccs2			New	F-element: Polynomial Current Controlled Current Source
pcccs3			New	F-element: Polynomial Current Controlled Current Source
pccvs			Augment	H-element: Polynomial Current Controlled Voltage Source; Added simInfo

Table 1 HSPICE Components in the Environment (Continued)

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
pccvs2			New	H-element: Polynomial Current Controlled Voltage Source; Added simInfo
pccvs3			New	H-element: Polynomial Current Controlled Voltage Source; Added simInfo
pdC			Augment	P-element: Port DC; Added simInfo and missing CDF parameters
pdiode	hspiceDCompPrim	diode	Augment	D-element: Parasitic Diode
pexp			Augment	P-element: Port Exponential; Added simInfo and missing CDF parameters
pinductor	hspiceDCompPrim	ind	Augment	L-element: Parasitic Inductor; Added missing CDF parameters
pjfet	hspiceDCompPrim	njfet	Augment	J-element: P-type JFET
pmes			New	J-element: P-type MESFET
pmes4			New	J-element: P-type MESFET 4-terminal
pmind	hspiceDCompPrim	mind	Augment	K-element: Parasitic Mutual Inductor
pmos	hspiceDCompPrim	pmos	Augment	M-element: P-type MOSFET; Added DELVTO
pmos4	hspiceDCompPrim	pmos	Augment	M-element: P-type MOSFET; Added DELVTO
pnP	hspiceDCompPrim	npn	Augment	Q-element: P-Type BJT

Table 1 HSPICE Components in the Environment (Continued)

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
port			Augment	P-element: Port; Added simInfo and missing CDF parameters; added Jitter source
ppulse			Augment	P-element: Port Pulse; Added simInfo and missing CDF parameters; added Jitter source
ppwl			Augment	P-element: Port Piece-Wise Linear; Added simInfo and missing CDF parameters
ppwlf			Augment	P-element: Port Piece-Wise Linear File; added simInfo and missing CDF parameters
presistor	hspiceDCompPrim	res	Augment	R-element: Parasitic Resistor; added simInfo and missing CDF parameters
psin			Augment	P-element: Port Sine Wave; added simInfo and missing CDF parameters
psoip		psoip	New	M-element: P-type SOI Level 57 BSIM3SOI
ptft		ptft	Augment	M element: Level 71, P type. added missing parameters
pvccs	hspiceDCompPrim	vccs_poly	Augment	G-element: Polynomial Voltage Controlled Current Source
pvccs2	hspiceDCompPrim	vccs_poly	Augment	G-element: Two-input Polynomial Voltage Controlled Current Source

Table 1 HSPICE Components in the Environment (Continued)

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
pvccs3	hspiceDCompPrim	vccs_poly	Augment	G-element: Three-input Polynomial Voltage Controlled Current Source
pvcvs	hspiceDCompPrim	vccs_poly	Augment	E-element: Polynomial Voltage Controlled Voltage Source
pvcvs2	hspiceDCompPrim	vccs_poly	Augment	E-element: Two-input Polynomial Voltage Controlled Voltage Source
pvcvs3	hspiceDCompPrim	vccs_poly	Augment	E-element: Three-input Polynomial Voltage Controlled Voltage Source
res	hspiceDCompPrim	res	Augment	R-element: Resistor; added missing CDF parameters
schottky	hspiceDCompPrim	diode	Augment	D-element: Schottky Diode
sp1tswitch		sp1tswitch	Augment	Verilog-A model support
sp2tswitch		sp2tswitch	Augment	Verilog-A model support
sp3tswitch		sp3tswitch	Augment	Verilog-A model support
sp4tswitch		sp4tswitch	Augment	Verilog-A model support
svccs		pole/laplace	Augment	G-element: added missing parameters
svcvs		pole/laplace	Augment	E-element: added missing parameters
switch		switch/relay	Augment	Verilog-A model support
tline	hspiceDCompPrim	tline	Augment	T-element: Ideal Transmission Line
u1wire	hspiceDCompPrim	u1wire	Augment	U-element: One Lumped Transmission Line

Table 1 HSPICE Components in the Environment (Continued)

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
u2wire	hspiceDCompPrim	uwire	Augment	U-element: Two Lumped Transmission Lines
u3wire	hspiceDCompPrim	uwire	Augment	U-element: Three Lumped Transmission Lines
u4wire	hspiceDCompPrim	uwire	Augment	U-element: Four Lumped Transmission Lines
u5wire	hspiceDCompPrim	uwire	Augment	U-element: Five Lumped Transmission Lines
usernpn	hspiceDCompPrim	bjt	Augment	Q-element: User Defined N-type BJT; added M and DTEMP
userpnp	hspiceDCompPrim	bjt	Augment	Q-element: User Defined P-type BJT; added M and DTEMP
vam	hspiceDCompPrim	vam	Augment	V-element: Voltage Amplitude Modulation
vbic			Augment	Q-element: VBIC BJT; added simInfo and missing CDF parameters
vccap	hspiceDCompPrim	vccap	Augment	G-element: Voltage Controlled Capacitor; added SMOOTH
vccs	hspiceDCompPrim	vccs	Augment	G-element: Voltage Controlled Current Source; added SMOOTH and NPDELAY
vccsp		vccsp	New	G-element: Voltage Controlled Current Source Multi-Input Gate
vclock			New	V-element: Voltage Source Clock Pulse
vcres	hspiceDCompPrim	vcres	Augment	G-element: Voltage Controlled Resistor; added SMOOTH

Table 1 HSPICE Components in the Environment (Continued)

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
vcvs	hspiceDCompPrim	vcvs	Augment	E-element: Voltage Controlled Voltage Source; added NPDELAY
vcvsp		vcvsp	New	E-element: Voltage Controlled Voltage Source Multi-Input Gates
vdc	hspiceDCompPrim_vdc	vsrc	Augment	V-element: Voltage Source
vexp	hspiceDCompPrim_vexp	vsrc	Augment	V-element: Voltage Source Exponential
vpas		vpas	New	V-element: Independent Voltage pattern source
vnbn			Augment	Q-element: Variable terminal N-Type BJT; added simInfo and missing CDF parameters
vpnp			Augment	Q-element: Variable terminal P-Type BJT; added simInfo and missing CDF parameters
vprbs		vprbs	New	V-element: Independent Voltage Pseudo Random-Bit Generator source; added Jitter source
vpulse	hspiceDCompPrim_vpulse	vsrc	Augment	V-element: Voltage Source Pulse; added Jitter source
vpwl	hspiceDCompPrim_vpwl	vsrc	Agument	V-element: Voltage Source Piece-Wise Linear
vpwlf	hspiceDCompPrim_pwlf	vsrc	Augment	V-element: Voltage Source Piece-Wise Linear File
vsffm	hspiceDCompPrim_vsffm	vsrc	Augment	V-element: Voltage Source Single Frequency FM
vsin	hspiceDCompPrim_vsin	vsrc	Augment	V-element: Voltage Source Sine Wave; added Jitter source

Table 1 HSPICE Components in the Environment (Continued)

Cell Name	hspiceD Netlist Process	Comp'nt	HSPICE integration	Description/Notes
vsouce			Augment	V-element: Voltage Source Universal; Added simInfo and missing CDF parameters; added Jitter source
vvmrf		vvmrf	New	V-element: Independent Voltage Vector Modulated RF source
winding	hspiceDCompPrim	winding	Augment	L-element: Winding; added M
xfmr	hspiceDCompPrim	xfmr	Augment	K-element: Transformer
zener	hspiceDCompPrim		Augment	D-element: Zener Diode
zvccs		ztrans	Augment	G-element: added missing parameters
zvcvs		ztrans	Augment	E-element: added missing parameters
hspice_vec_inc			Augment	Added .inc statement into netlist
hspice_veriloga_inc			Augment	Added .hdl statement into netlist

HSPICE Complex Sources

HSPICE supports complex independent sources including pattern, PRBS, VMRF, and Jitter in the analogLib. You can add them into a schematic and perform HSPICE simulation.

The following introduces those sources. See the *HSPICE User Guide: Simulation and Analysis*, [Sources and Stimuli](#) for detailed information on these sources.

Pattern Source

The pattern source uses four states, '1', '0', 'm' and 'z', which represent the high, low, middle voltage or current and high impedance state, respectively.

The syntax is:

```
Vxxx n+ n- PAT [(] vhi vlo td tr tf tsample data [RB=val]
+ [R=repeat] [<)]
```



```
Ixxx n+ n- PAT [() vhi vlo td tr tf tsample data [RB=val]
+ [R=repeat] [()]>
```

Note: The pattern name will be annotated in the UI with the input pattern value.

Pseudo Random-Bit Generator Source (PRBS)

The PRBS uses a linear Feedback Shift Register (LFSR) to generate a pseudo random bit sequence.

The syntax is:

```
Vxxx n+ n- LFSR [() vlow vhigh tdelay trise tfall rate seed
+ [taps] [rout=val] [ENCODE=DW8b10b] [RD_INIT=0|1] [()]]
Ixxx n+ n- LFSR [() vlow vhigh tdelay trise tfall rate seed
+ [taps] [rout=val] [ENCODE=DW8b10b] [RD_INIT=0|1] [()]]
```

Vector-Modulated RF Source (VMRF)

VMRF is generated by modulating an RF carrier with in-phase and out-phase signal.

The syntax is:

```
Vxxx n+ n- VMRF [() AMP=sa FREQ=fc PHASE=ph MOD=MOD
+ FILTER=FIL FILCOEF=filpar RATE=Rb BITSTREAM=data
+ [TRANFORHB=0/1] [()]]
Ixxx n+ n- VMRF [()
AMP=sa FREQ=fc
PHASE=ph
MOD=MOD
+ FILTER=FIL FILCOEF=filpar RATE=Rb BITSTREAM=data
+ [TRANFORHB=0/1] [()]]
```

For details, see [Vector-Modulated RF \(VMRF\) Source](#) in the *HSPICE User Guide: RF Analysis*.

Jitter Source

Jitter source can be embedded into other sources and analyses including pulse, sin, con, PRBS, and StatEye analysis.

To add a jitter source, use the keyword PERJITTER.

Example syntax for sin is:

```
Vxxx n+ n- SIN [() vo va [freq [td [q [j ]]]] [()]]
+ [PERJITTER=val SEED=val]]
Ixxx n+ n- SIN [() vo va [freq [td [q [j ]]]] [()]]
```

Chapter 2: Updating Libraries and Tool Filter

Adding the HSPICE Simulator to the Tool Filter

```
+ [PERJITTER=val SEED=val]
```

The parameter `SEED` enables you to generate different random number sequences. For details, see [Clock Source with Random Jitter](#) in the *HSPICE RF Analysis User Guide*, and [Periodic Jitter Effect](#) in Chapter 18, *Statistical Eye Analysis* in the *HSPICE User Guide: Simulation and Analysis*.

Adding the HSPICE Simulator to the Tool Filter

While it is not necessary to have the Tool Filter recognize the parameters that apply only to HSPICE, to be able to run the HSPICE Integration, it is vital to add HSPICE to the Tool Filter. There are several parameters that otherwise would be hidden from the users of analogLib and other libraries.

Once you perform this step in the installation hierarchy, all users pointing to the hierarchy will benefit from the Tool Filter.

1. Change directories to

```
cadence_install_dir/tools/dfII/etc/tools/auCore
```

Note: The *cadence_install_dir* is the one that you either created or updated when you installed the HSPICE integration to the Environment.

2. Add the following line to your *.cdsinit* file:

```
envSetVal("asimenv.startup" "simulator" 'string "HSPICE"
```

3. Make HSPICE your default simulator as follows: edit the *.cdsenv* file and add HSPICE to the list of default simulators. For example:

```
auCore.toolFilter defaultTools string "spectre spectreS auCdl  
auLvs HSPICE" nil
```

Environment Setup

This chapter describes the setup steps to configure an environment to do HSPICE simulation and analysis on your design.

For in-depth discussion of GUI components peripheral to the HSPICE integration, see the Cadence Virtuoso documentation available through the Environment Console dialog menu: Help > Cadence documentation > Analog Design Environment.

These topics are discussed in the following sections:

- [Environment Console](#)
- [Menus — Environment Console](#)
- [Session](#)
- [Setup](#)
- [Verilog-A Support](#)

For discussion of the Environment Console menus for Analyses, Variables, Outputs, Simulation, and Results, refer to the appropriate chapters.

Environment Console

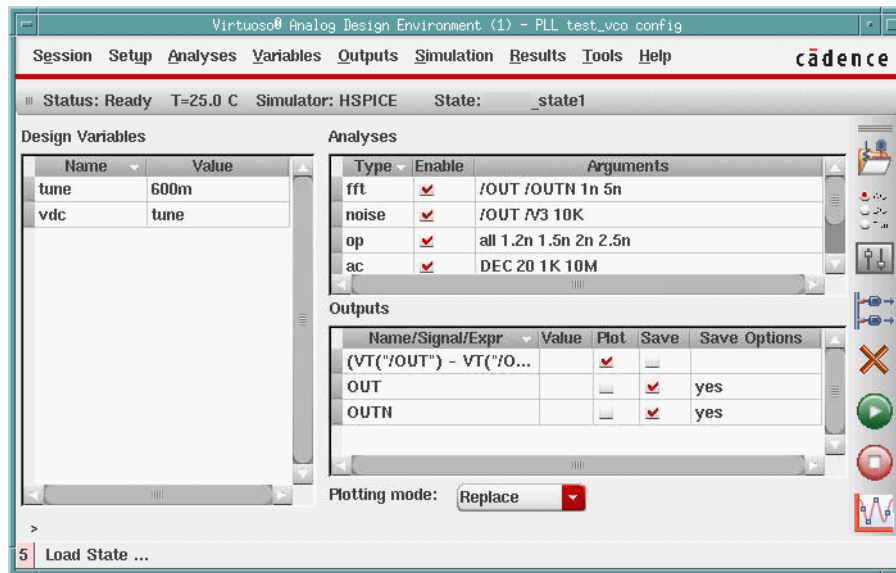
You can display the Environment Console from either the CIW window or from the Schematic Editor.

1. From the CIW:
 - For version 5.1xx... select Tools > Analog Environment > Simulation.
 - For version 6.1xx... select Tools > ADE L > Simulation
2. To display from the Schematic Editor:

Chapter 3: Environment Setup

Environment Console

- For version 5.1xx... select Tools > Analog Environment
- For version 6.1xx... select Launch > ADE L



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 2 Environment Console with HSPICE as the Simulator

If you open the Environment Console from the Schematic Editor, the resulting onscreen cell view in the Schematic Editor is the design simulation candidate.

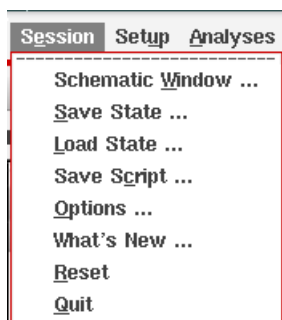
Menus — Environment Console

The following sections describe the Session and Setup Menus and menu items available in the Environment Console in the HSPICE integration to the Environment.

Note: The data you enter in many of the fields of the various menu dialogs can be saved and reloaded as State data under the HSPICE integration. You can also save these values to the `.cdsenv` file with the Options > Save Defaults selection in the CIW > Options > Save Defaults... menu dialog, and to initialize them at startup with that same file. In addition, the values in these fields can be set using the `.cdsinit` SKILL file.

Session

This HSPICE integration maintains the user interface with which users of the Environment are familiar.



- **Schematic Window:** This selection brings the design layout of the current state to the top of your display.
- **Save State and Load State:** Launches either the Save State or Load State form. You can save and restore all or part of the simulation environment setup using these menu options. States can be stored to directories or to cellviews so that they can be managed with design data.

Note: Users of Virtuoso hspiceD can expect to have their State data read by the new HSPICE integration.

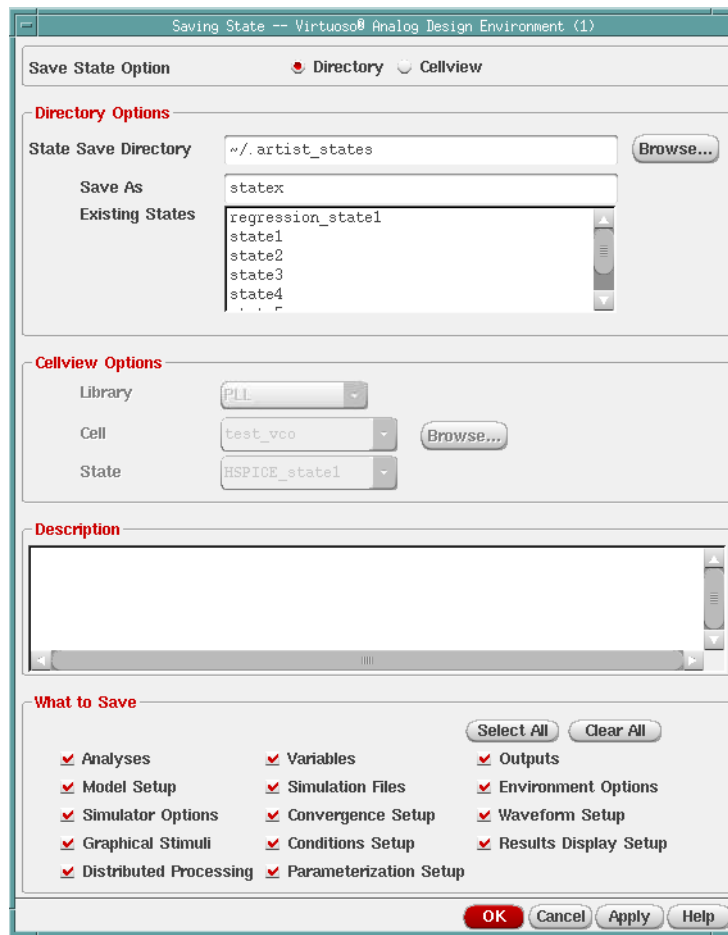
Saved states are simulator-dependent for analyses, simulator options, and convergence setup (if the convergence commands you saved are not supported by the other simulator).

You can restore saved states from different simulators. The analog circuit design environment restores as much as possible despite simulator-dependent settings. Saving the waveform setup using the Saving State form saves the same information as the File > Save as command in the waveform window. If AWD (Analog Waveform Display) is your waveform viewer, **Window > Save** is the command.

As seen in the bottom portion of the Save State form ([Figure 3 on page 96](#)) you can select which data you want to keep for the saved state. Before closing a session you are prompted as to whether or not you wish to save the current state.

Chapter 3: Environment Setup

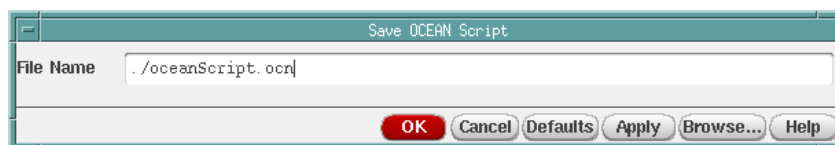
Environment Console



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 3 Save State Form

- **Save Script:** Enables you to save OCEAN (*.ocn) scripts.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Running OCEAN for the HSPICE integration requires an additional initialization step which can be performed by doing either of the following:

- 1) Create an `.oceanrc` file with the content:

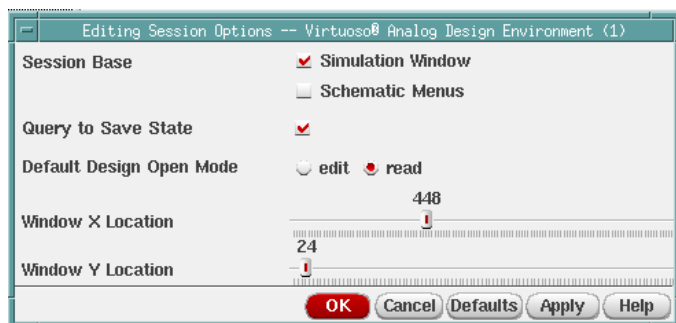
```
load (prependInstallPath("local/HSPICE/HSPICE.ini"))
```

in the directory from where you run “ocean”. When the command “ocean” is executed, it automatically loads the content of *.oceanrc*. It is similar to *.cdsinit* when you launch *icms/icfb*.

- 2) In the beginning of the OCEAN script, add the following line:
`load(prependInstallPath("local/HSPICE/HSPICE.ini"))`
- To have this statement created automatically at the head of the OCEAN script, add the following lines to your *.cdsinit* file:

```
envSetVal("asimenv.misc" "preSaveOceanScript"  
'string "myPreSaveProc")  
procedure(myPreSaveProc(session fp)  
  prog()  
  fprintf( fp "load( prependInstallPath( \"local/HSPICE/  
    HSPICE.ini\" ))\n" )  
  drain(fp)  
)  
)
```

- **Options:** Provides a dialog of the same Editing Session Options familiar to Environment users.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- **What's New:** Opens an information window describing the HSPICE integration to the Environment.
- **Reset:** Restores the original defaults.
Note: If you have data in this form, it is overwritten with default settings. Use the Session > Save command to save this information before resetting the defaults.
- **Quit:** Closes the Environment Console.

Setup

The **Setup** menu maintains the same menu items as the other simulators supported by the Environment. It also includes the HSPICE distributed processing solution for Monte Carlo and corner analyses. See [Chapter 15 on page 321](#).

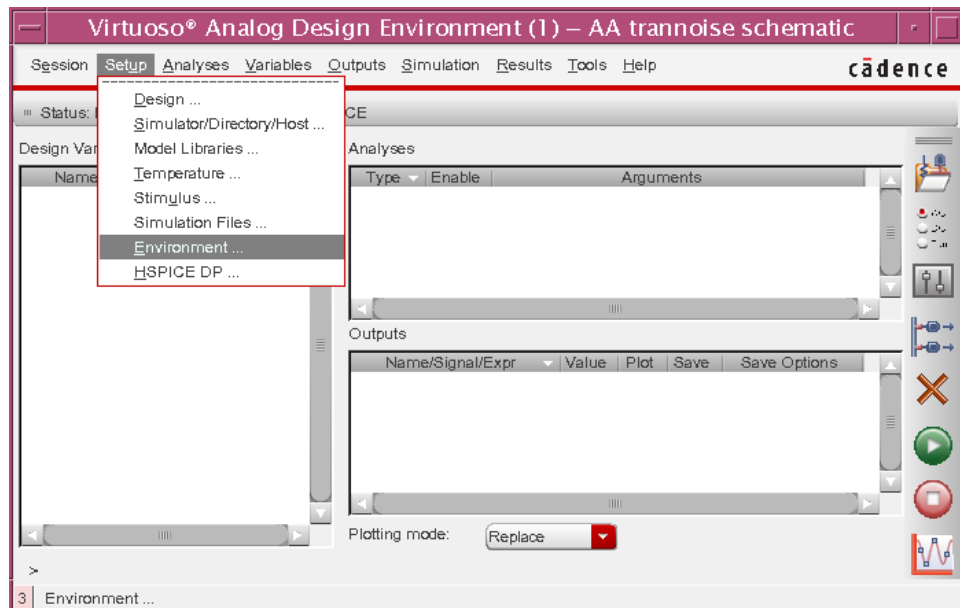
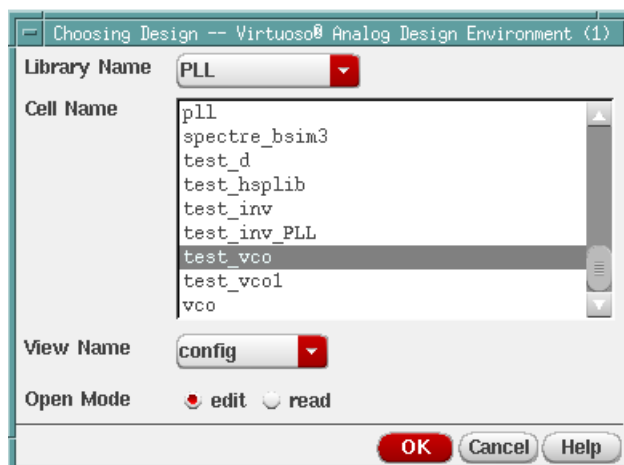


Figure 4 Setup menu

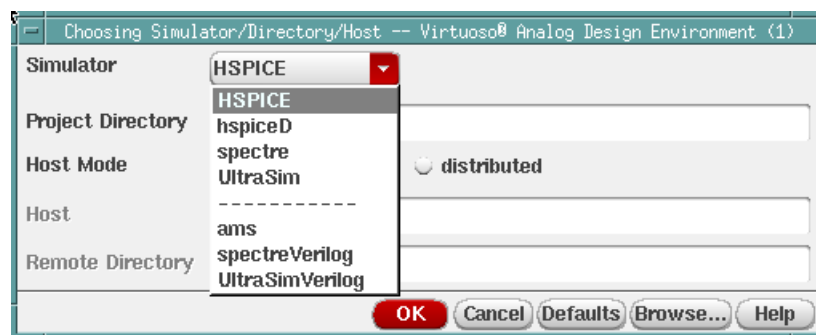
- **Design:** Opens the Choosing Design dialog for changing the design simulation candidate of the ADE session.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 5 Setup > Design Launches Choosing Design Form

- **Simulator/Directory/Host:** Opens a dialog for configuring session components, including choosing HSPICE as the simulator. Any State data in the HSPICE interface will be saved under this integration name.

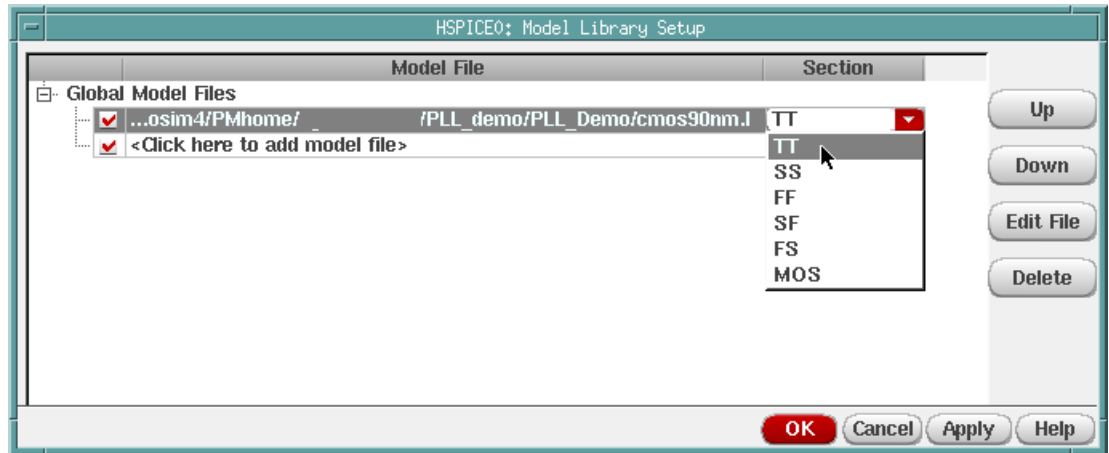


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- **Model Libraries:** Opens the Model Library Setup dialog to enable you to add a model file by browsing, manually typing, or copying and pasting a model file name and path. When you specify a model file, you can then select the library section from the **Section** column pulldown, which is populated by section names. Model library files will be printed as user-specified and HSPICE will resolve the paths using the `.option search` commands.

Chapter 3: Environment Setup

Environment Console



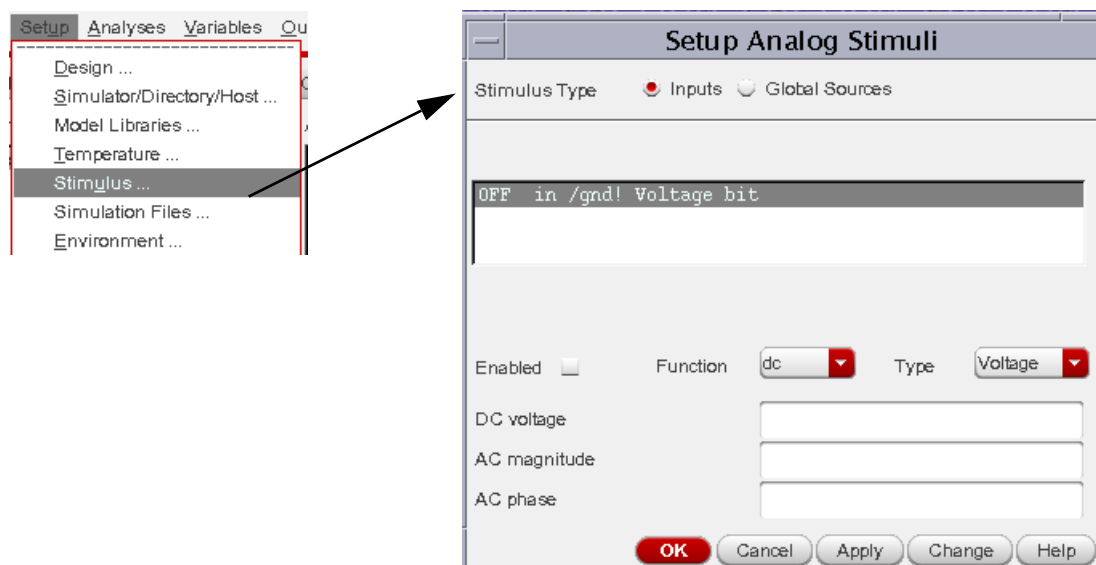
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- **Temperature:** Opens the Setting Temperature form. The form provides radio buttons for selecting the either the Celsius, Fahrenheit or Kelvin scale and setting default degrees.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

- **Stimulus:** Opens the Setup Analog Stimuli dialog for you to set inputs and global sources for your design. These pins and global nets must exist in your design and are activated as independent sources to drive your circuits through the stimuli setup. The form changes dynamically when you select a different input pin, function, or type.



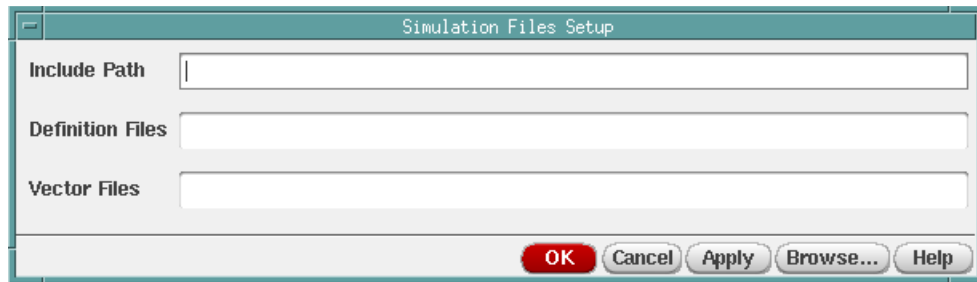
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

The following controls enable these usages:

- **Stimulus Type:** radio buttons to select either
Inputs: Sets the stimulus for the signals with input pins in the schematic.
Global Sources: Lets you assign DC voltages to global signals that represent power supplies in the design.
- Text box identifies the signal names with the currently selected highlighted.
- **Enabled** specifies whether each signal is ON or OFF.
- **Function** enables choice of the function for the selected signal with stimulus option properties and values. The possible functions include:

Function	Stimulus Option
dc	direct current
pulse	pulse stimulus
sin	sinusoidal
exp	exponential
pwl	piecewise linear
sffm	single frequency FM

- **DC voltage, AC magnitude, and AC phase** provide parameter input specific to the HSPICE simulator.
- **Simulation Files:** Opens the Simulation Files Setup dialog.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

The fields allow you to enter:

- **Include Path**, to directories containing necessary files, such as a path to an HDL file for simulation using Verilog-A, or other definition files. The netlister will print the `.option search` command for each path specified in the include path. If the path is specified relatively, it converts the path into an absolute path using the current working directory.
- **Definition Files**, which can specify functions and global variables that are not design variables. These files can contain model parameters or simulator parameters. Definition files will be printed as user-specified and the simulator will resolve the paths using the `.option search` commands.

An example of setting a definitions file follows:

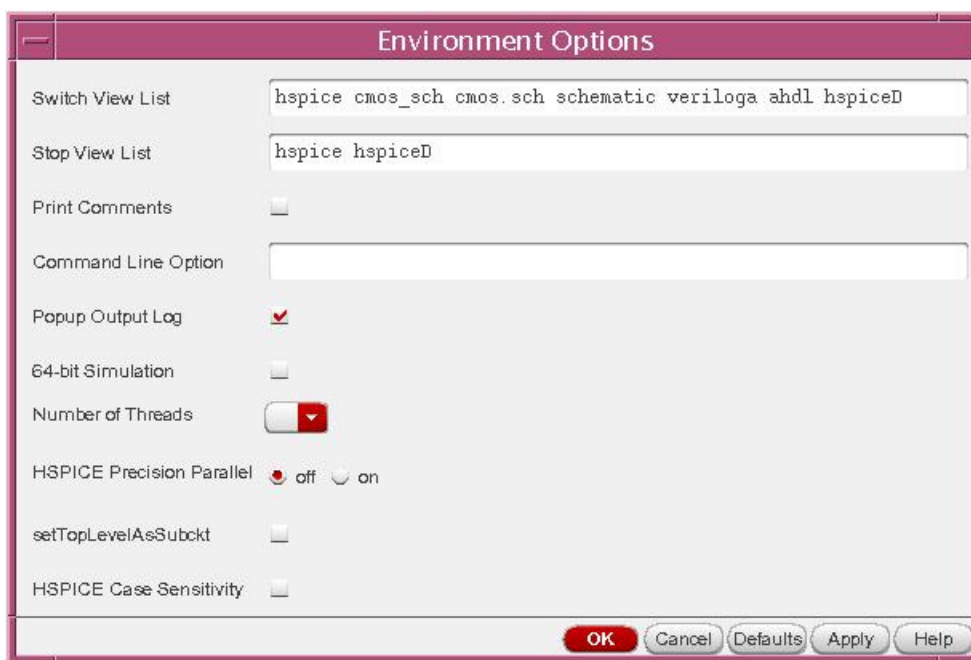
- In the Environment Console, select **Setup > Simulation Files**.
- In the Simulation Files Setup form, **Definition Files** field, enter the full path of the definitions file (including extension). For example:

```
/install_dir/tools/dfII/examples/artist/hspice/  
models/models.sp
```

- **Vector Files** (unique to the HSPICE integration). You can enter multiple space-delimited *.vec files in this field to be used on the `.VEC` card. Enter the vector file paths fully qualified. Vector files will be printed as user-specified and the simulator will resolve the paths using the `.option`

search commands. See [Digital Vector File Commands](#) in the *HSPICE Reference Manual: Commands and Control Options* for a listing of all HSPICE vector commands.

- **Environment Options:** Opens the dialog for you to specify command line options, check boxes to choose to print commands, automatically display an output log file (*on* by default) onscreen while the simulation is running, specify 64-bit simulation, and controls to set up a multithread count, HSPICE Precision Parallel technology, specify a top level schematic as a subcircuit, and enable case sensitivity in netlists.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 6 Environment Options

- The fields and check boxes include:
 - Switch View List:** Field to define string of space-separated views that the tool switches into when searching for design variables.
 - Stop View List:** Field to specify list of views that identify the stopping view to be netlisted. Both hspice and hspiceD are available by default.
 - Print Comments:** Check box — When off, comments are not printed. When on, extra comments are placed in the netlist regarding component location and name.

Chapter 3: Environment Setup

Environment Console

- **Command Line Opt:** Specify command line options that many not be supported by the GUI on the call to HSPICE which are appended to the list that the Environment normally sends to the simulator. For example: You can invoke multithreading:

```
% hspice -mt=2
```


You can specify Verilog-A files using the `-hdl` option in the Command Line Option field (see [Verilog-A Support](#) below). This field allows you to take advantage of current and future capabilities that HSPICE may offer. By default, the HSPICE invocation is done with the following command:

```
% hspice input.ckt >& ../psf/hspice.out
```
- **Popup Output Log:** Check box — When switched on, opens an HSPICE Log Files window that displays simulator messages. You can select from a **Logfiles** menu to display a listing file ***.lis**, standard out ***.st0**, or **warning/error** file. (See [Figure 7 on page 104](#).)



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 7 Available onscreen output files

- **64-bit Simulation:** Check box to turn on HSPICE 64-bit simulation and enable PSF output. For example: Before saving an OCEAN script select the **64-bit Simulation** environment option.

- **Number of Threads:** Control to specify the number of threads when using the HSPICE multithread feature (See [Running Multithread/Multiprocess HSPICE Simulations](#) in the *HSPICE User Guide: Simulation and Analysis*).
- **HSPICE Precision Performance:** Toggle radio button **off** (default) or **on** to invoke the HSPICE Precision Performance technology for medium to large size block (over 10 million elements) multithread simulations. See [HSPICE Precision Parallel \(-hpp\)](#) in the *HSPICE User Guide: Simulation and Analysis*.
- **setTopLevelAsSubckt:** Check box to set the top level schematic as a subcircuit. The option allows users of co-simulation to have HSPICE netlist using the `.subckt` statement without first creating the symbol for the schematic. For example, if you have an element named `v0` in the schematic, when netlisting using the Environment Console **Simulation > Netlist > Create** the tool generates the following netlist by default.

```
.global 0
.option search='.'
.temp 25
.option ARTIST=2 PSF=2
v0 vout 0
.end
```

Instead, you can netlist this as a `.subckt` if you have enabled **setTopLevelAsSubckt** on the Environment Options form.

```
.global 0
.option search='.'
.temp 25
.subckt vout
v0 vout 0
.ends vout
.end
```

You can also set the following environment option to enable this feature. Either of the following commands needs to be set in the `.cdsinit` or the `icms console`.

```
envSetVal("HSPICE.envOpts" "setTopLevelAsSubckt" 'boolean
t)
envSetVal("HSPICE.envOpts" "setTopLevelAsSubckt" 'boolean
nil)
```

However, this has to be set each time the session starts. Clicking the check box in the Environment Options form is the recommended method.

- **Case Sensitivity:** Check box to enable use of case sensitivity in netlists for:
 - Node Names
 - Parameter Names
 - Instance Names
 - Model Names
 - Subcircuit Names
 - Data Names
 - Measure Names
 - Library Entry Names
 - File Names and Paths (case sensitive by default)

Alternatively, you can use the SKILL function to set an environment variable in the initialization file to turn on/off this feature manually.

```
envSetVal("HSPICE.envOpts" "hspCaseSensitive 'boolean t/  
nil)
```

Verilog-A Support

HSPICE integrates the ability to use Verilog-A simulator capability. The HSPICE integration provides the *.hdl* statement that points to the module to include:

```
.hdl "/remote/home/usrA./cdn_oa/adeLib/res/veriloga/  
veriloga.va"  
xi7 0 net6 res r=500e3
```

The HSPICE integration supports inclusion of two same-named Verilog-A modules in the same netlist. This avoids issues when multiple Verilog-A views for a single cell may be accessed from two different libraries and the cells have the same name. (See the example below.) Conflicts can also arise if you partition one instance to use Verilog-A, and another to use a subcircuit with the same name as the module. (This occurs any time the same cell is used, since the subcircuit and Verilog-A module both have that cell name.) When this situation occurs, the HSPICE module name aliasing capability is employed:


```
.hdl "./cdn_oa/adeLib/res/veriloga/veriloga.va" res  
adeLib_res_veriloga  
.hdl "./cdn_oa/devLib/res/veriloga/veriloga.va" res  
devLib_res_veriloga  
xi6 0 net5 adeLib_res_veriloga r=500e3  
xi7 0 net6 devLib_res_veriloga r=500e3
```

In this example, one of the resistors is placed from the library `adeLib`, and the other from `devLib`. They both have the cell name `res`, which means that the default master name on the instance line, as well as the module name, will be `res`. Both have the view name `veriloga`. But since these two modules are found in different libraries, they may differ in functionality. To bind each module to the correct instance, an aliasing capability associates the real module name with a name that is formed by concatenating the library, cell, and view names together. This handles same-named modules in different libraries, as well as modules in the same library and cell, but different views.

Using Verilog-A files in the HSPICE integration assumes that you have brought your Verilog-A modules into the OpenAccess (OA) or the Cadence CDBA library structures. You can also use a method for including modules that are outside of the library structure, by creating a stopping view which causes an instance statement with the proper pinout and module name, and then invoking the `-hdl veriloga_file` option in the **Command Line Option** field. This method currently works in any integration, and continues to work in the HSPICE integration.

Using the HSPICE Verilog-A Compiler by Default

Users do not need to set anything to use the native HSPICE Verilog-A compiler as it is specified by default.

To change to the Cadence® Spectre® native VA compiler, follow these instructions:

1. Locate the environment variable `hsp_vacompiler`. By default it is set to `true`.
2. Change the setting to `false` to enable the native simulator VA compiler by adding the following line to your `.cdsenv` file:

```
envSetVal("HSPICE.envOpts" "hsp_vacompiler" boolean nil)
```


Analysis Setup and Design Variables

This chapter describes the Analysis capabilities available through the HSPICE integration interface and introduces the editor for design variables.

These topics are presented in the following sections.

- [Transient Analysis](#)
- [DC Analysis](#)
- [AC Analysis](#)
- [Operating Point Analysis](#)
- [Noise Analysis](#)
- [Discrete-Fourier Transform Analysis](#)
- [Linear Network Parameter Analysis](#)
- [DCMatch Analysis](#)
- [ACMatch Analysis](#)
- [Loop Stability Analysis](#)
- [Pole/Zero Analysis](#)
- [Design Variables](#)

HSPICE Integration Analyses

The HSPICE integration to the Environment currently supports nine analyses. As more analysis types are added to the HSPICE integration they will be documented in this chapter. Basic analysis types include TRAN, DC, AC, OP,

NOISE, FFT, LIN, DCMATCH, ACMATCH, LSTB, and PZ. (Postprocessing for these analyses is discussed later in this user guide.)

Users can save field values to the `.cdsenv` file with the Options > Save Defaults menu in the session Environment Console, and initialize them at startup with that same file. The values in these fields can be set via the `.cdsinit` SKILL file.

To open the analyses GUI, in the Environment Console window select Analyses > Choose to display the Choosing Analyses dialog window.

Note: PSF output is supported by all current HSPICE analyses in the HSPICE integration, except DCMatch and ACMatch. You use the [HSPICE Plotting Assistant](#), described in Chapter 7 to plot results requiring post-processing using the PSF data. The maximum PSF file name with path is 1024 characters.

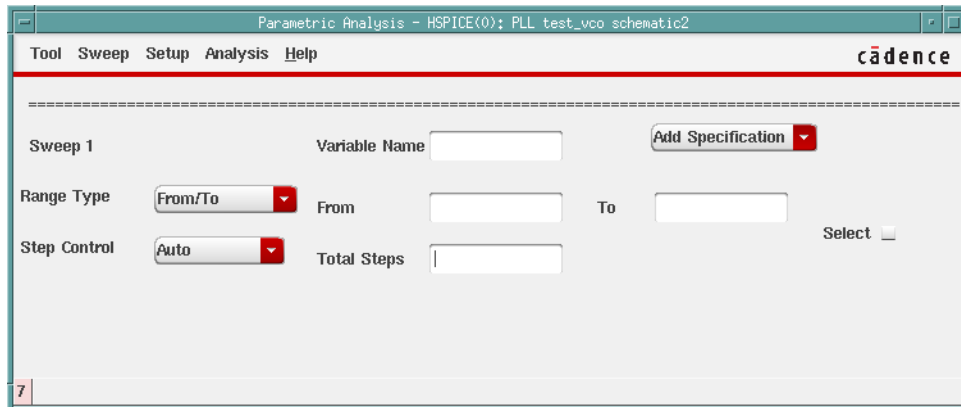
The following sections describe the supported HSPICE analyses and the enhancements to the existing Environment simulation implementations.

For Monte Carlo trials and HSPICE RF analyses see [Chapter 8, Monte Carlo in the HSPICE Integration](#) and [Chapter 10, HSPICE RF Analysis](#).

Multi-Dimensional Sweeps

Although the HSPICE simulator itself supports multi-dimensional sweeps directly on some analysis lines, a multi-dimensional sweep via ADE requires the use of the parametric analysis tool.

In order to use a second sweep, you must use parametric analysis, which is available from the Tools menu (**Tools > Parametric Analysis**) ([Figure 8](#)).



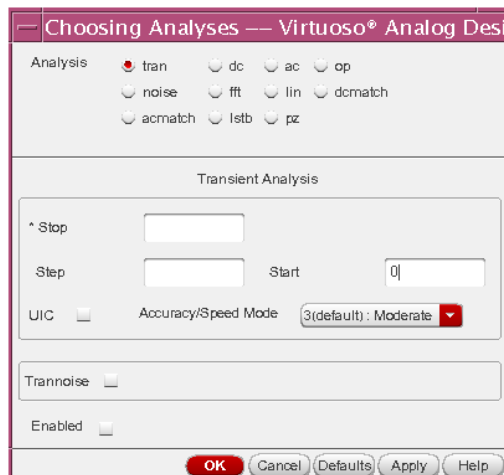
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 8 Parametric Analysis Form for Multiple Sweeps

Transient Analysis

The HSPICE integration for transient analysis supports multiple timestep increments in a single analysis. This reflects the Synopsys HSPICE ability to specify multiple time steps (max time step) values on the .tran analysis command line. For example:

```
.TRAN 1e-9 100e-9 1e-12 150e-9 1e-9 200e-9 START=0.0
```



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

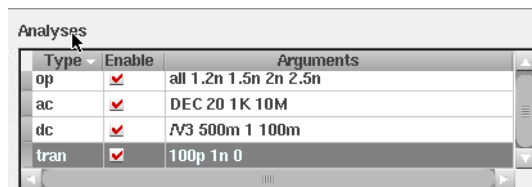
Figure 9 Transient Analysis Form

The HSPICE integration Transient Analysis form provides the following controls and fields:

- Multiple timesteps are useful for saving time in simulating circuits where start-up behavior is not of interest, or for tightening up the timestep in regions of the transient response where areas of high curvature exist. **Step 1, 2, 3**, etc. and equivalent **Stop 1, 2, 3**, etc. fields are provided to specify the number of timestep regions (intervals) that you wish to set up. Manually enter multiple step and stop values in these fields.
- The **Start** field allows you to enter a time in the simulation when printing or plotting begins.
- If you turn the **UIC** button on, you specify the UIC parameter in the .TRAN statement, HSPICE does not calculate the initial DC operating point, but directly enters transient analysis. When you use .TRAN UIC, the .TRAN node values (at time zero) are determined by searching for the first value found in this order: from .IC value, then IC parameter on an element statement, then .NODESET value, otherwise use a voltage of zero.
- The **Accuracy/Speed Mode** spinner allows you to select the equivalent of the runlvl option in HSPICE. The default of **3 Moderate** is recommended for almost all simulations.
- The **Trannoise** check box is discussed in the [Transient Noise Analysis](#) section.

Making entries for a standard transient analysis requires no additional postprocessing capabilities. Transient signals that result from this multi-timestep analysis are plotted as if the analysis was run with a single timestep.

When the value of any field is changed, the **Enabled** button is automatically set to *on*. When you click **OK** or **Apply**, the form values are applied to the environment.



Type	Enable	Arguments
op	<input checked="" type="checkbox"/>	all 1.2n 1.5n 2n 2.5n
ac	<input checked="" type="checkbox"/>	DEC 20 1K 10M
dc	<input checked="" type="checkbox"/>	/V3 500m 1 100m
tran	<input checked="" type="checkbox"/>	100p 1n 0

Figure 10 Analyses Table on Environment Console Updated with Analysis Types and Parameter Values

In addition, the values for the most common fields are displayed in the **Analyses** table on the Environment Console. This behavior occurs for all the analysis forms.

Since an FFT analysis requires that a transient analysis must be specified and enabled, a warning appears on the FFT Analysis form. The warning does not appear if a transient analysis has been enabled.

Monte Carlo analysis can be run with transient analysis either through the HSPICE Integration UI (see [Chapter 8, Monte Carlo in the HSPICE Integration](#)) or by creating an OCEAN script for non-GUI batch runs (see [Appendix B, OCEAN API Functions for HSPICE Monte Carlo Analysis](#)).

For full information regarding HSPICE Transient Analysis, see the **.TRAN** command in the *HSPICE Reference Manual: Commands and Control Options* and [Transient Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

The following section discusses

- [Transient Noise Analysis](#)

Transient Noise Analysis

HSPICE Transient Analysis shows the effect of noise on the signal magnitude.

You can use either:

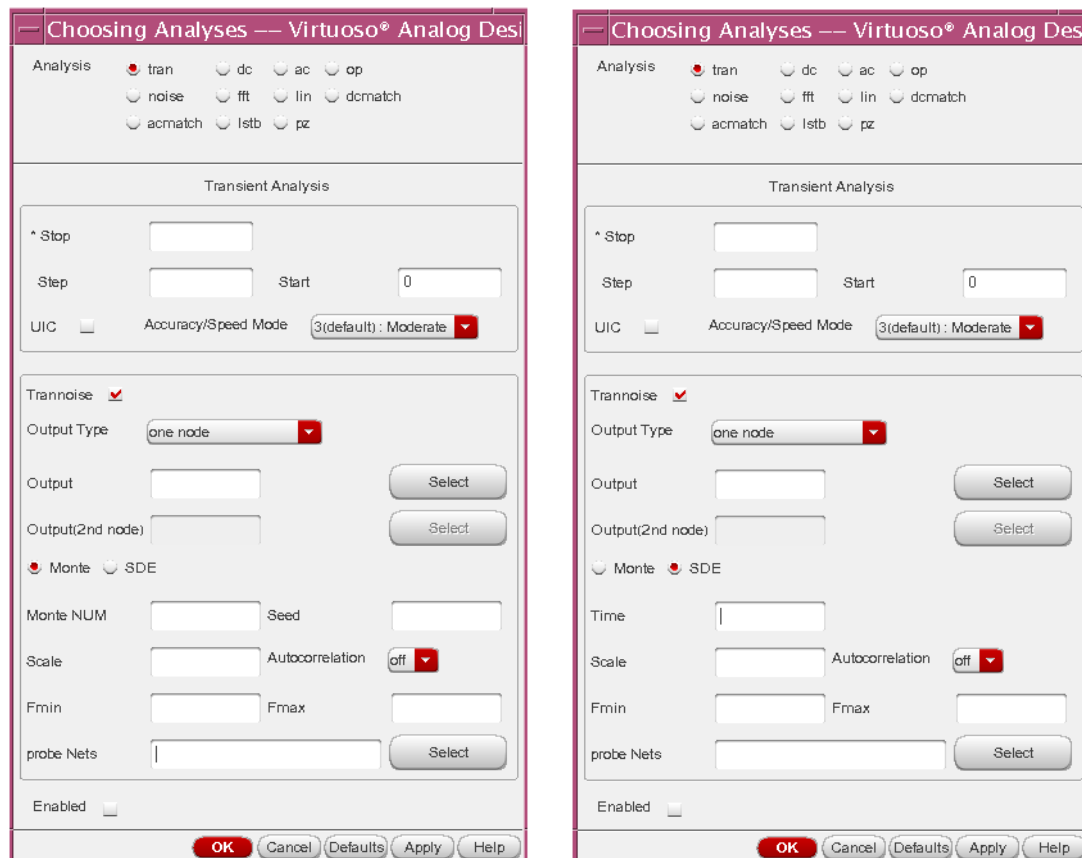
- Monte Carlo (default) method, where the device noise is modeled as uncorrelated random signal sources to predict the statistical characteristics of the circuit performance due to device noise.
- Stochastic Differential Equation (SDE) method, for advanced users, which makes a direct prediction of the actual statistics of the output waveforms.

See [Transient Noise Analysis](#) in the *HSPICE User Guide: RF Analysis* for details on these methods.

To invoke the HSPICE Trannoise Analysis form, select **Analyses > Choose** on the Environment Console to open the Choosing Analyses form. The Transient Analysis form opens with **tran** as the default-selected analysis. To initiate a transient noise analysis, click the **Trannoise** check box. The Transient Analysis form expands to allow setup of transient noise analysis, as shown [Figure 11 on page 114](#).

Chapter 4: Analysis Setup and Design Variables

Transient Analysis



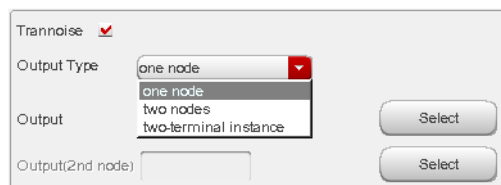
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 11 Expanded Transient Analysis form to set up a Transient Noise Analysis for a Monte Carlo or SDE simulation

Setting Up a Trannoise Analysis

Follow these steps to set up a transient noise simulation:

1. Select an **Output Type** from the three available types: **one node** (default), **two nodes**, and **two-terminal instance**.



The **Output** field entry is the source node if the **Output Type** is not selected to be **two-terminal instance**, otherwise, it is filled with the source instance name.

If **Output Type** is **two nodes**, the **Output (2nd node)** field becomes enabled and is filled with the 2nd node of the output source.

The **Select** button can be used to choose the node/instance from the design schematic editor.

2. Choose the method of trannoise simulation from the Monte Carlo single run (Default), multiple runs or **SDE** (Stochastic Differential Equation) for advanced usage. As shown in [Figure 11 on page 114](#) the method is defaulted to **Monte**.
 - Under the Monte Carlo method, you can enter values for **Monte NUM** and **Seed**. The seeded “first run” sets the first simulation sample. Use **Monte NUM** to specify the simulation samples. For example, for **Monte NUM**, you can enter 20 30 40 or 20:30 40. 20:30 40 means samples from 20 to 30 inclusive and sample 40.
 - Under the **SDE** method, you can also set values for **Time** (time points) for the stochastic simulation.
3. Optionally, for both methods, you can set **Scale** (amplification factor for noise), **Fmin** (minimum frequency for frequency dependent noise), and **Fmax** (maximum frequency for frequency dependent noise).
4. **Autocorrelation** “on” enables autocorrelation function calculation at the specified output (default is “off”).
5. **probe Nets** (optional) Lists the nets that can be probed for noise with the “VRMS” function (VRMS: The output of RMS noise voltages at other nodes (i.e., the output for general nodal noise voltage values).
6. The **Select** button allows you to automatically select autocorrelation outputs and/or nets to be probed using VMRF from the design schematic editor.

Limitation

The HSPICE ADE Trannoise feature currently does not support transient noise summary.

Plotting Trannoise Curves Using the Plotting Assistant

For full information on using the [HSPICE Plotting Assistant](#), see [Chapter 7 on page 173](#).

Chapter 4: Analysis Setup and Design Variables

Transient Analysis

In the Environment console select **Results > Plotting Assistant** to display the HSPICE Plotting Assistant utility.

Select the **tran** radio button for the **Analysis**.



Figure 12 HSPICE Plotting Assistant for Tran Noise

To plot output noise using the Plotting Assistant:

1. Select **ONOISE** in the **Function** section.
2. Click **Plot** or **Replot** to generate the “onoise” curve.

To plot noise for selected nets:

1. Select **VMRS** in the **Function** section.

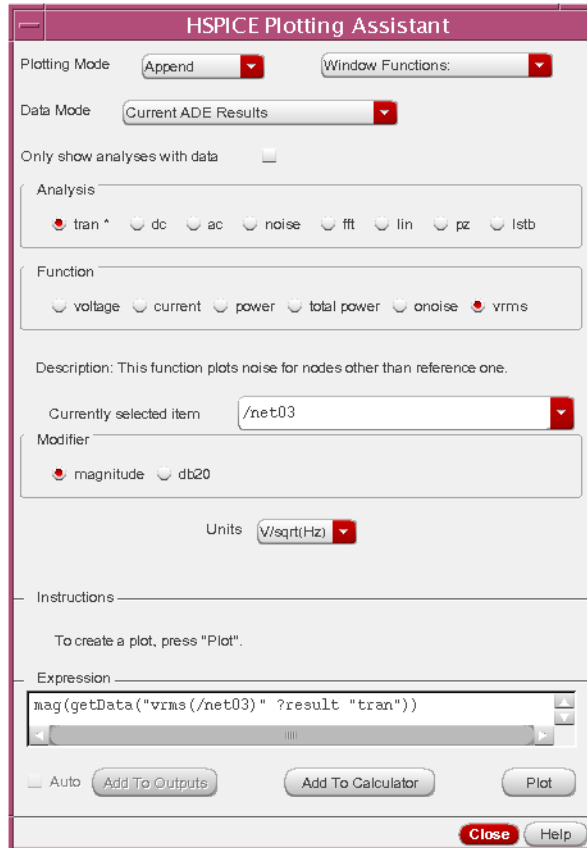


Figure 13 Setup for vrms noise curve

2. Click **Plot** or **Replot** to generate the selected net noise curve.
3. Select a different net using the **Currently selected item** selection box.

Note: You can select any of the nets that are listed in the **probe Nets** field in the Trannoise setup form.

DC Analysis

The interface for DC analysis supports the sweeps of temperature, design variables, or unparameterized voltage sources (voltage value is swept).

Chapter 4: Analysis Setup and Design Variables

DC Analysis

Sweep Type fields change depending on selected sweeps

Clicking Design Variable changes form options

Points of Interests selection replaces all fields with a single field for space-separated entries

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 14 DC Analysis form

- Selecting the **Source** (default) radio button allows you to pick a source by clicking the **Select Source** button and then clicking on a legitimate component on the schematic to auto-fill the **Source Name** field. You can also manually specify the source name.
- Selecting the **Design Variable** radio button expands the DC analysis form for you to specify a design variable from the list of currently established design variables, as shown on the Environment Console. (See the [Design Variables](#) section later in this chapter for more information.) Click the **Select Design Variable** button to open the Select Design Variable for dc window. Highlight the variable of choice and click **OK** to auto-fill the **Variable Name** field on the DC Analysis form. Alternatively, you can manually enter a design variable acceptable to HSPICE.

- To sweep temperature, select the **Temperature** radio button. To change the temperature value use the **tnom** option (see the TNOM option on the **Simulation > Options > All Analog Options** form). Refer to [Options](#) in Chapter 6, Running Simulations and Using Control Options.
- You can select the **Sweep Type** as a range of linear steps, linear points, points per octave, points per decade, or space-separated points of interest.

Monte Carlo analysis can be run with DC analysis either through the HSPICE Integration UI (see [Chapter 8, Monte Carlo in the HSPICE Integration](#) or by creating an OCEAN script for non-GUI batch runs (see [Appendix B, OCEAN API Functions for HSPICE Monte Carlo Analysis](#)).

For full information regarding HSPICE DC Analysis, see the [.DC](#) command in the *HSPICE Reference Manual: Commands and Control Options* and [Initializing DC-Operating Point Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

AC Analysis

In addition to a simple frequency sweep, the AC Analysis form allows you to run the AC analysis at a specific frequency in conjunction with a temperature or parameter sweep.

Chapter 4: Analysis Setup and Design Variables

AC Analysis

If Sweep type is changed to Points of Interest the fields change to a single field to specify white-space separated points

Selecting Design Variable expands the form to show Variable Name and At Frequency (Hz) fields

Selecting Temperature expands the form to display At Frequency (Hz) field

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 15 AC Analysis Form

- The **Frequency** (default) radio button specifies the sweep variable in Hz.
- Selecting the **Design Variable** radio button expands the DC analysis form for you to specify a design variable from the list of currently established design variables, as shown on the Environment Console. (See the [Design Variables](#) section later in this chapter for more information.) Click the **Select Design Variable** button to open the Select Design Variable for dc window. Highlight the variable of choice and click **OK** to auto-fill the **Variable Name** field on the DC Analysis form. Alternatively, you can manually enter a design variable acceptable to HSPICE. You can add design variables one at a time by this selection process and click **OK** or **Apply** on the analysis form to add this design variable data for the simulation.

- To sweep temperature, select the **Temperature** radio button. To change the temperature value use the tnom option (see the TNOM option of the Simulation > Options > All Analog Options form). Refer to [Options](#) in Chapter 6, Running Simulations and Using Control Options.
- You can select the **Sweep Type** as a range of linear steps, linear points, points per octave, points per decade, or space-separated points of interest.

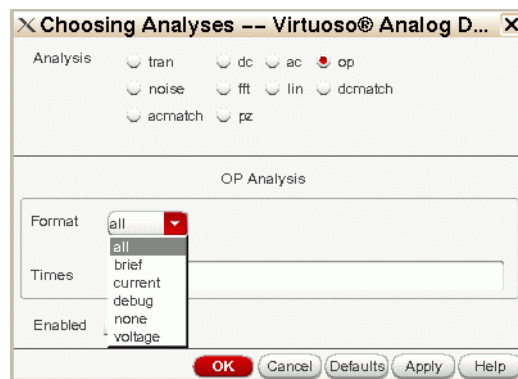
Since noise and linear networks analyses analysis require an AC analysis to be enabled, a warning advises of the requirement. The warning does not appear if an AC analysis has been enabled.

Monte Carlo analysis can be run with AC analysis either through the HSPICE Integration UI (see [Chapter 8, Monte Carlo in the HSPICE Integration](#) or by creating an OCEAN script for non-GUI batch runs (see [Appendix B, OCEAN API Functions for HSPICE Monte Carlo Analysis](#)).

For information regarding HSPICE AC Analysis, see the [.AC](#) command in the *HSPICE Reference Manual: Commands and Control Options* and [AC Small-Signal and Noise Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

Operating Point Analysis

The .OP Analysis interface calculates the DC operating point of the circuit and saves circuit values at multiple timesteps.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 16 Operating Point Analysis Form

The **Format** drop-down provides options for: all, brief, current, debug, none, and voltage. Only one format can be run per simulation.

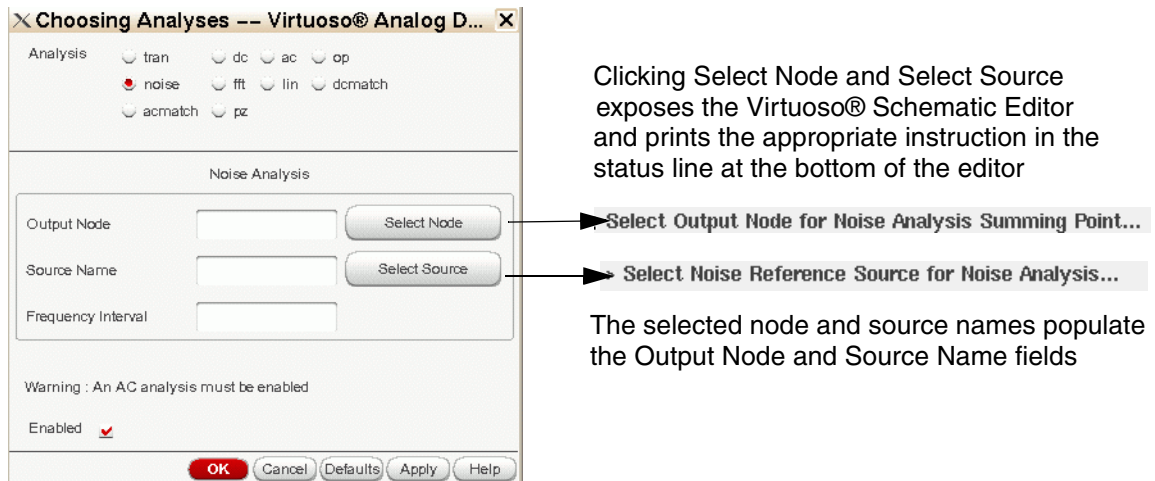
- **all**: Full operating point, including voltage, currents, conductances, and capacitances. This parameter outputs voltage/current at the specified times.
- **brief**: Produces a one-line summary of each element's voltage, current, and power. Current is stated in milliamperes and power is in milliwatts.
- **current**: Produces a summary voltage table of element currents and power.
- **debug**: Usually invoked only if a simulation does not converge. Debug prints the non-convergent nodes with the new voltage, old voltage, and the tolerance (degree of non-convergence). It also prints the non-convergent elements with their tolerance values.
- **none**: Inhibits node and element printouts.
- **voltage**: Produces a voltage table only.

Use the **Times** field to specify the time points for which HSPICE prints the report. The times must be space-separated, for example: 1.2n 1.5n 2n.

For full information, see the [.OP](#) command in the *HSPICE Reference Manual: Commands and Control Options* and [Initializing DC-Operating Point Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

Noise Analysis

A noise analysis calculates the output noise generated based on the contributions from all noise sources within the circuit. Noise may be from passive elements, such as thermal noise in resistors, or from sources such as shot, channel, and flicker noise present within transistors.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 17 Noise Analysis Form

- The **Output Node** and **Source Name Select...** buttons make the Schematic Editor active for selection of wires or components; after clicking the **Select Node** or **Select Source** button and then clicking on an appropriate component in the schematic, the name value/value is automatically written to the fields in the analysis form. Alternatively, you can manually enter the node and source names.
- The **Frequency Interval** requires manual entry (but is not mandatory).

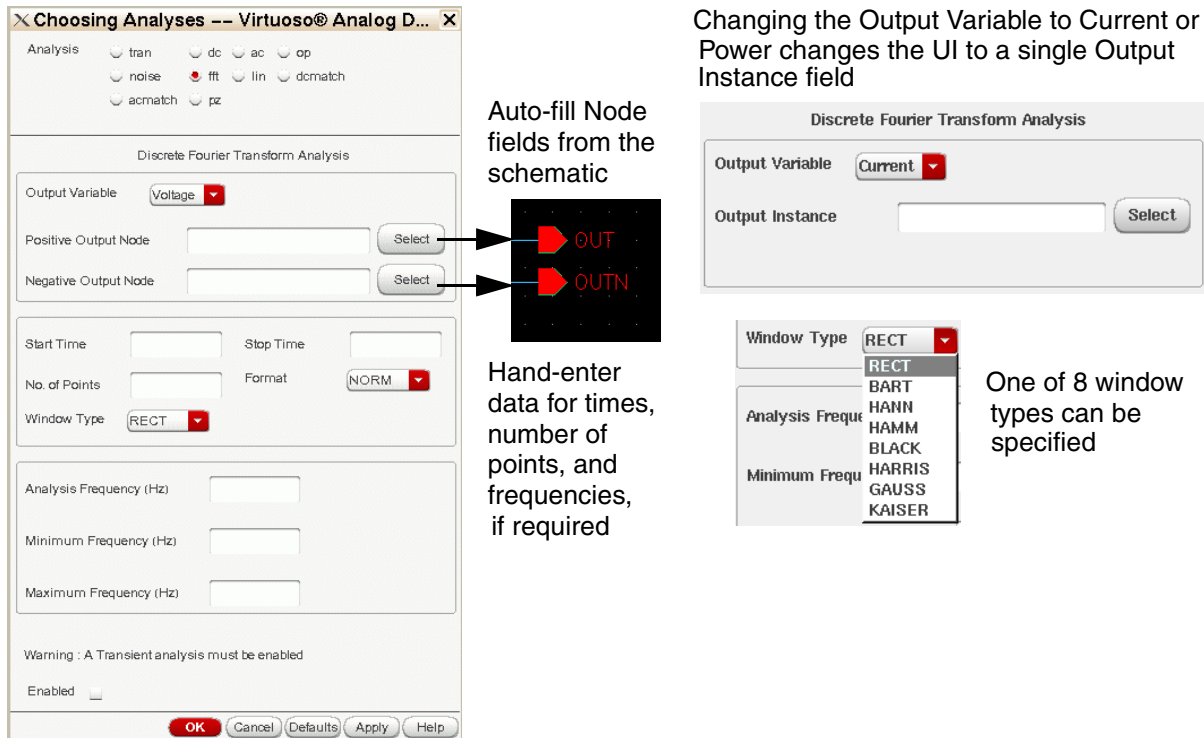
Note: HSPICE noise analysis does not support the use of variables.

Since noise analysis requires an AC analysis to be enabled, a warning advises of the requirement. No warning appears if an AC analysis has been enabled.

For full information regarding HSPICE Noise Analysis, see the [.NOISE](#) command in the *HSPICE Reference Manual: Commands and Control Options* and [Simulation of Random Noise](#) and [Using .NOISE for Small-Signal Noise Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

Discrete-Fourier Transform Analysis

The FFT analysis calculates the Discrete Fourier Transform (DFT) value used for spectrum analysis. Numerical parameters (excluding string parameters) can be passed to the .FFT statement.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 18 Discrete Fast-Fourier Transform Analysis Form

The **Output Variable** can be selected from the dropdown and can be either:

- **Voltage:** The UI for Voltage provides fields for **Positive Output Node** and **Negative Output Node**. Clicking the accompanying **Select** buttons brings the Schematic Editor to the top of the window stack and you can make selections directly on the schematic nodes to auto-fill the fields; or you can manually enter the component node names.
- **Current:** Selecting Current changes the node fields to a single **Output Instance** field. Clicking select results in the same auto-fill sequence as above.
- **Power:** Shows the same UI as the previous selection for current (the single **Output Instance** field).

The next pane in the form specifies data selection having to do with how the analysis and plot will be constructed.

- **Start** and **Stop** fields: start of the output variable waveform to analyze (defaults to the START value in the .TRAN statement, which defaults to 0) and end of the output variable waveform to analyze (defaults to the TSTOP value in the .TRAN statement).
- **No. of Points:** number of points to use in the FFT analysis. NP must be a power of 2. If NP is not a power of 2, HSPICE automatically adjusts it to the closest higher number that is a power of 2. The default is 1024.
- Format can be either:
 - **NORM:** normalized magnitude (default)
 - **UNORM:** un-normalized magnitude

- **Window Type**

One of eight window types can be selected:

- **RECT**—simple rectangular truncation window (default)
- **BART**—Bartlett (triangular) window
- **HANN**—Hanning window
- **HAMM**—Hamming window
- **BLACK**—Blackman window
- **HARRIS**—Blackman-Harris window
- **GAUSS**—Gaussian window

- **KAISER**—Kaiser-Bessel window
When either the Gaussian or Kaiser-Bessel window is chosen, the UI adds a field called **Alfa** for the parameter to control highest side-lobe level and bandwidth. The Alfa parameter has a legal range of 1.0 to 20.0 inclusive; the default is 3.0.

If there is no transient analysis run as part of the simulation, then the .FFT computation is not run. By default, .FFT will use the time points calculated by the transient analysis to perform a DFT, which extracts frequency content from the circuit. As an alternative, users can run a transient analysis and use the DFT function of the Environment calculator.

Note: Using the calculator will always be slower because it is a post-processing step. The .FFT calculates the Fourier Transforms on-the-fly. If you forgot to set up the .FFT and you performed a very long transient simulation you can use the calculator or a third party application to retrieve the FFT.

To avoid interpolation errors caused by this post-processing approach, HSPICE automatically creates the FFT_ACCURATE option in simulation input file. You can set the ACCURATE option by selecting on the Environment Console: Simulation > Options > Commonly Used. The ACCURATE option also turns on FFT_ACCURATE, which forces HSPICE to insert computed time points for every time value that is used in the .FFT. See the [Options](#) section Chapter 6, [Running Simulations and Using Control Options](#).

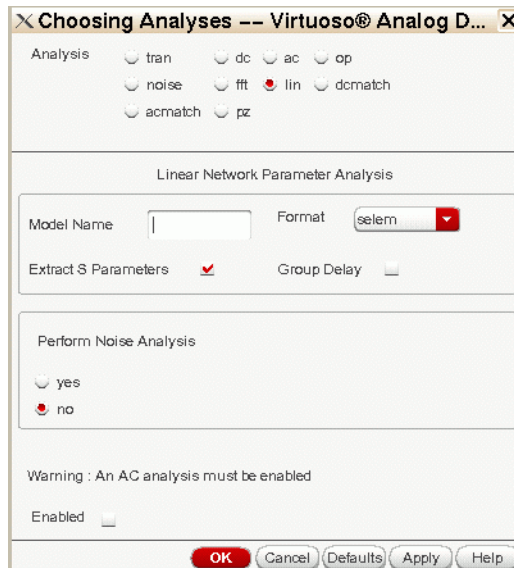
The interface to the .FFT analysis supports the ability to specify all of the parameters associated with the analysis. These include the output variable (only one can be specified per .FFT), the time period over which to calculate the FFT, the number of points to use, and the windowing function.

Note: You cannot use variables with either the **Window Type** or **Format** options in an FFT analysis.

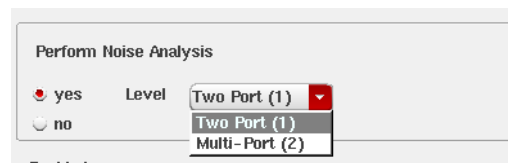
For full information regarding HSPICE Spectrum Analysis, see the .FFT command in the *HSPICE Reference Manual: Commands and Control Options* and [Spectrum Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

Linear Network Parameter Analysis

The Linear Network Parameter analysis extracts noise and linear transfer parameters for a general multi-port network.



On the Perform Noise Analysis pane
Selecting yes expands the UI by adding
a Level dropdown list to select 2-port or
multi-port noise analysis



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 19 Linear Network Analysis Form

- The **Model Name** field is for specifying the model name to be listed in the .MODEL statement in the .sc# model output file.
- You can specify the **Format** to save the extracted parameters as a S-element (*.sc#), Citi, or Touchstone file.
- The **Extract S Parameters** check box is turned *on* by default.
- A **Group Delay** check box allows you to extract group delay (perform group delay analysis).

The **Perform Noise Analysis** pane offers **yes** and **no** options for Noise analysis to be run as part of a .LIN analysis. The .LIN analysis can compute the

equivalent two-port noise parameters or multi-port noise parameters for a network.

- The .LIN analysis can be run with or without the extraction of noise parameters.
- Selecting **yes** expands the pane to include a **Level** dropdown list for:
 - **Two-port (1):**
 - **Multi-port (2)**

The 1 and 2 refer to the HSPICE argument `noisecalc` values.

As with Noise analysis, .LIN requires an .AC analysis to be run. A warning notifies that an AC analysis has not been enabled. The warning is not displayed if an AC analysis has been enabled.

.NOISE as Part of a .LIN Analysis

When a noise analysis is run as part of a .LIN analysis, the type of results generated by HSPICE is different than when just the .NOISE analysis is run. The .LIN analysis generates two-port and multi-port noise parameters such as noise figure, noise equivalent resistance, and maximum power gain. The .LIN analysis with `noisecalc` enabled does not generate noise voltage values, and suppresses their generation if a separate .NOISE analysis is present in the netlist. This means that any voltage signals that are set up to be automatically plotted after the simulation completes will result in an error message similar to the following:

```
*Error*   Trying to plot expression <VN2()>, which does not
evaluate to an object that can be plotted, like waveform or
parametric wave. Please refer to the ViVA User Guide to see what
all types of objects can be plotted in ViVA. Expression which
evaluates to those type(s) of objects can only be plotted.
```

This message can be ignored, but be aware that noise voltages will not be plotted under these conditions.

Similarly, trying to plot any of the following from the Results > Direct Plot menu will produce an error message similar to the one shown above:

- Equivalent Output Noise
- Equivalent Input Noise
- Squared Output Noise
- Squared Input Noise

For full information regarding HSPICE Linear Networks Analysis, see the [.LIN](#) command in the *HSPICE Reference Manual: Commands and Control Options* and [Linear Network Parameter Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

DCMatch Analysis

The DCMatch Analysis form is unique to the HSPICE integration to make use of the Variation Block functionality. In order for this analysis to be functional a library file with a Variation Block section must be set up for the session (see Model Libraries in the [Setup](#) section).

Choosing Analyses -- Virtuoso® Analog D...

Analysis

☐ tran ☐ dc ☐ ac ☐ op

☐ noise ☐ fft ☐ lin ☒ dcmatch

☐ acmatch ☐ pz

DCMatch Analysis

Output Variables

Type	Node/Instance	Node
------	---------------	------

Output Type: Voltage

Positive Output Node: Select

Negative Output Node: Select

Add Delete Change

Threshold: Interval:

Warning : A DC analysis must be enabled

Enabled ☐

OK Cancel Defaults Apply Help

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 20 DCMatch Form

The **Output Variables** field is populated (Type Node/Instance, and Node) either:

- Automatically, by the controls below the columns, according to the output node or instance name selected on the schematic. Click **Add** to send data to the **Output Variables** list field.
- Or by manually entering the output variables in the editable field(s) and clicking **Add**.

When you change the output type to **Current** the two voltage fields change from **Positive** and **Negative Output Node** to an **Output Instance** field.

- When voltage is the output variable, the **Select** buttons for **Positive Output Node** and **Negative Output Node** make the Schematic Editor active for selection of nodes; after clicking the **Select** button and then clicking on an output node in the schematic, the value is automatically written to the field in the analysis form.
- When current is the output variable, the **Select** button for **Output Instance** enables the same behavior as above.

Each set of values is transferred one at a time by clicking the **Add** button so that you can create a matrix of values for **Output Variables**.

The **Delete** button removes the output variable data if it is highlighted.

If you need to change a value either manually or by selecting a component on the schematic, use the **Change** button to replace a highlighted row of output variable data as follows:

- Click **Select** then select a valid node on the schematic and click **Change**.
- Alternatively, you can type the name of the node in the **Positive** or **Negative Output Node** field and then click **Change**.

The **Threshold** and **Interval** fields can be optionally filled in manually.

Note: The DCMatch **Interval** setting requires that a DC sweep is defined (see [DC Analysis](#)).

For full information regarding HSPICE DCMatch Analysis, see the [.DCMATCH](#) command in the *HSPICE Reference Manual: Commands and Control Options* and [DCMatch Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

ACMatch Analysis

The ACMatch Analysis form is unique to the HSPICE integration to make use of the Variation Block functionality. In order for this analysis to be functional a library file with a Variation Block section must be set up for the session (see “Model Libraries” in the [Setup](#) section).

Choosing Analyses -- Virtuoso® Analog D...

Analysis: ☐ tran ☐ dc ☐ ac ☐ op
☐ noise ☐ fft ☐ lin ☐ dematch
☒ acmatch ☐ pz

ACMatch Analysis

Output Variables

Type	Modifier	Node/Instance	Node

Output Type: Modifier:

Positive Output Node:

Negative Output Node:

Threshold: Interval:

Warning : An AC analysis must be enabled

Enabled ☐

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 21 ACMatch Form

- The **Output Variables** field is populated (Type Node/Instance, and Node) either:
 - Automatically, by the controls below the columns, according to the output node or instance name selected on the schematic. Click **Add** to send data to the **Output Variables** list field.

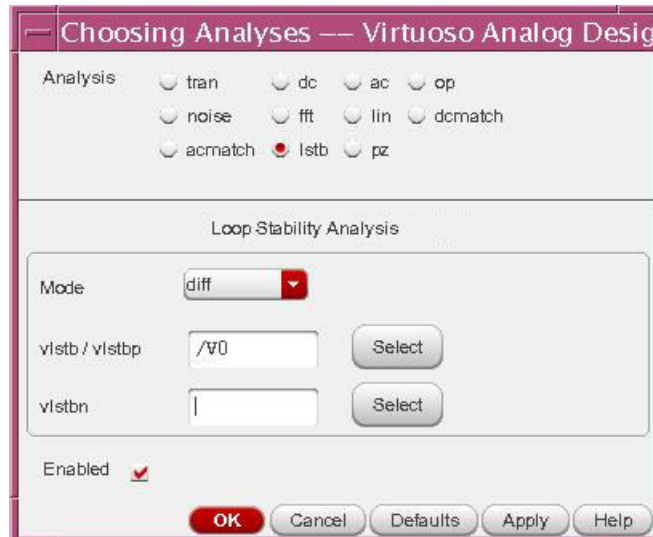
- Or by manually entering the output variables in the editable field(s) and clicking **Add**.
- The **Modifier** pulldown allows you to select a *quantity* type of output voltage or current. For example, you can select from several output voltages, difference voltages, or branch current through an independent voltage source. In the HSPICE syntax, the voltage or current specifier is followed by an identifier of the AC quantity of interest: **magnitude**, **phase**, **real**, or **imaginary**.
- When voltage is the output variable, the **Select** buttons for **Positive Output Node** and **Negative Output Node** allow you to make the Schematic Editor active for selection of nodes; after clicking the **Select** button and then clicking on an output instance in the schematic, the value is automatically written to the fields in the analysis form.
- When current is the output variable, the **Select** button for **Output Instance** enables the same behavior as above.
- The **Delete** button removes the output variable data if it is highlighted.
- The **Change** button replaces a highlighted row of output variable data as follows:
 - Click **Select** then click on a valid node on the schematic and click **Change**.
 - Alternatively, you can type the name of the node in the **Positive** or **Negative Output Node** field and then click **Change**.
- The **Threshold** and **Interval** fields can be optionally filled in manually.

For full information regarding HSPICE ACMatch Analysis, see the [.ACMATCH](#) command in the *HSPICE Reference Manual: Commands and Control Options* and [ACMatch Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

Loop Stability Analysis

To invoke the HSPICE-ADE Loop Stability Analysis (lstb) Setup Form ([Figure 22 on page 133](#)): select **Analyses > Choose** on the Environment console.

Note: An AC Analysis must be enabled to activate LSTB analysis. A warning is issued if no AC analysis is enabled.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 22 Loop Stability Analysis setup form

For detailed information on HSPICE loop stability analysis see [Using .LSTB for Loop Stability Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

Setting Up and Running a Simple LSTB Analysis

The following controls and fields are available on the Loop Stability Analysis form:

- The **Mode** types include **single** (default), **diff** (differential), and **common**.
- The **vlstb/vlstbp** field allows you to input a **single** mode vsource or invoke either a **diff** or **comm** mode as one of the two vsources.
- The **vlstbn** input field can be used to input either diff or comm mode as the other of the two vsources. When in **single** mode, this field is disabled.
- The **Select** buttons allow you to select the voltage source from the schematic editor which will input the sources directly from the schematic.

Complete the analysis setup by clicking **Apply** or **OK**.

To netlist and run a loop stability simulation select **Simulation > Netlist and Run** in the Environment console.

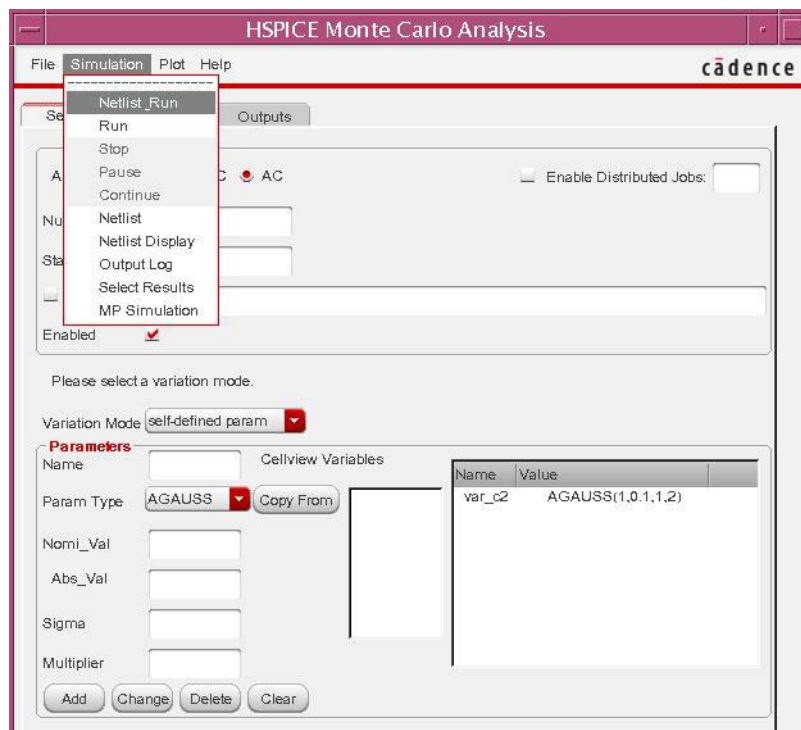
Running LSTB Monte Carlo, Corner, and Parametric Analyses

Once you have set up the loop stability analysis you can run Monte Carlo, corner, and parametric analyses as follows:

Monte Carlo Analysis with LSTB Analysis

With the setup Loop Stability Analysis form on screen, select **Tools > HSPICE Statistical Analysis** in the Environment console to open the HSPICE Monte Carlo Analysis form ([Figure 23](#)).

Verify that the **AC** radio button is selected and proceed to set up an HSPICE Monte Carlo Analysis. Refer to [Chapter 8, Monte Carlo in the HSPICE Integration](#) for details on setting up a Monte Carlo run. When you have completed the setup, select **Simulation > Netlist_Run** to netlist and run a simulation.

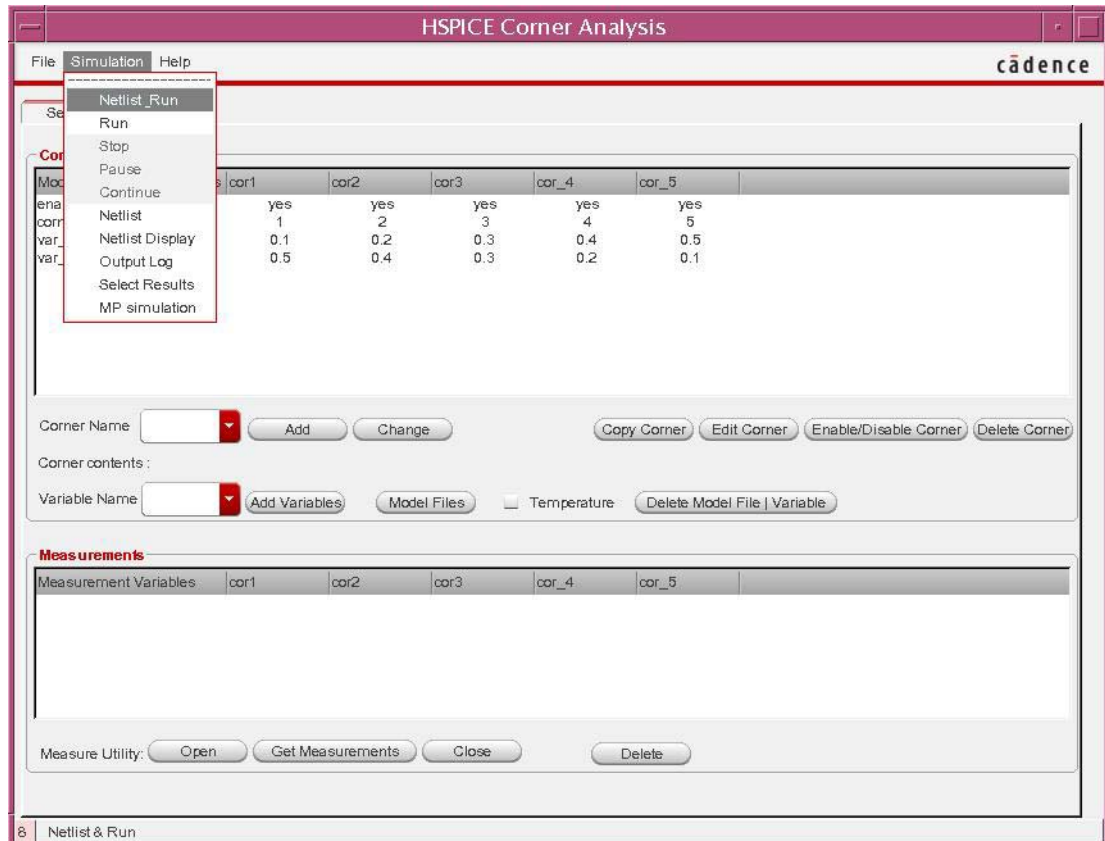


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 23 HSPICE Monte Carlo Analysis form

Running a Corner Analysis with a LSTB Analysis

Select **Tools > HSPICE Corner Analysis** in the Environment console to open the HSPICE Corner Analysis form (Figure 24) to set up an HSPICE Corner Analysis. See [Chapter 9, Corners Analysis](#) for details on setting up and running a corner analysis.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 24 HSPICE Corner Analysis form

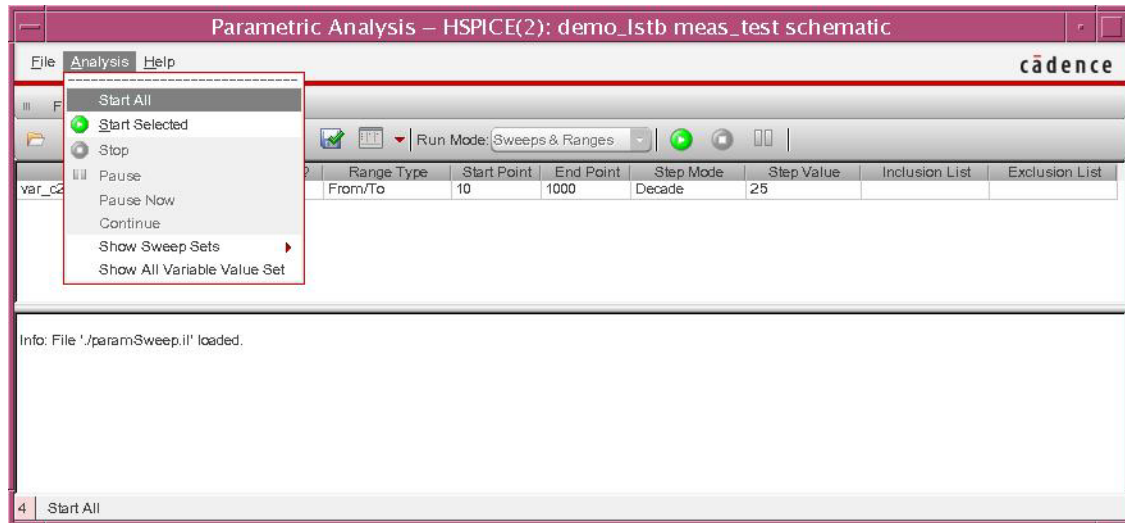
Select **Simulation > Netlist_Run** to launch a simulation.

Running a Parametric Analysis with a LSTB Simulation

Select **Tools > Parametric Analysis** in the Environment console to open the Parametric Analysis form.

Chapter 4: Analysis Setup and Design Variables

Loop Stability Analysis



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 25 Parametric Analysis form

After setup, select **Analysis > Start All** or **Start Selected** to netlist and run a simulation.

Plotting LSTB Analysis Results Using the Plotting Assistant

Select **Results > Plotting Assistant** in the Environment console to launch the utility that helps you plot the waveform of HSPICE output after a simulation completes.

The Plotting Assistant form can be used to plot LSTB analysis output and measurement results after running a simulation. [Figure 26 on page 137](#) shows the **1stb** radio button selected in the **Analysis** group.

The **Function** group displays six functions. The **Loop Gain** choice is used to plot LSTB analysis outputs with the **Modifier** group. The other functions are used to use to display loop stability measurement results (or plot a waveform if LSTB analysis is run with Parametric analysis).

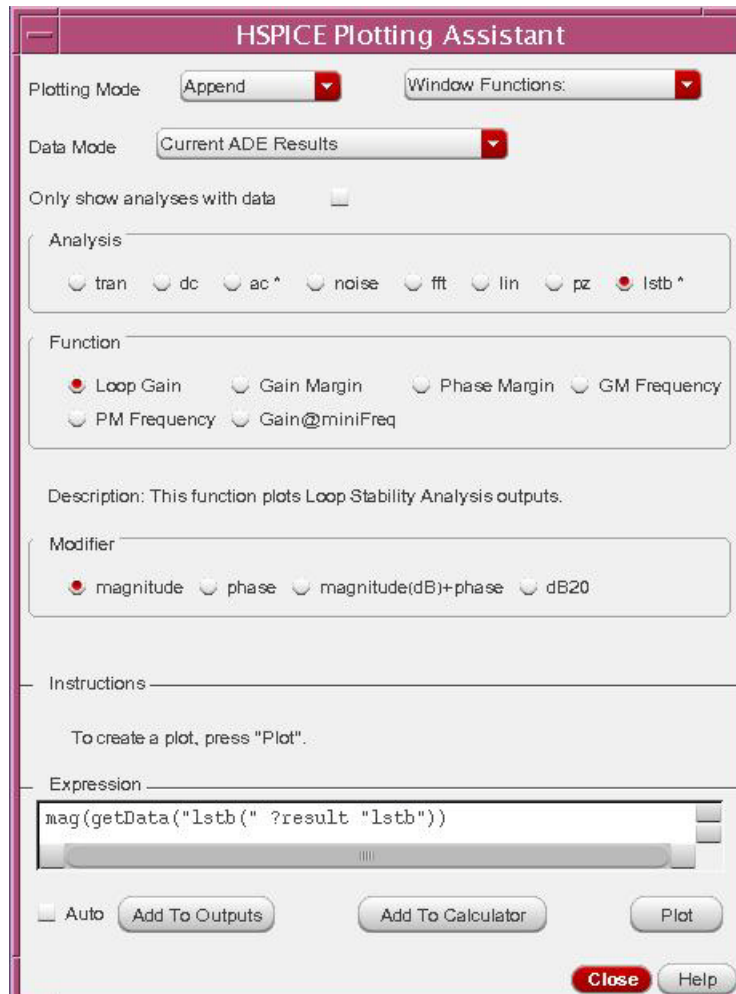
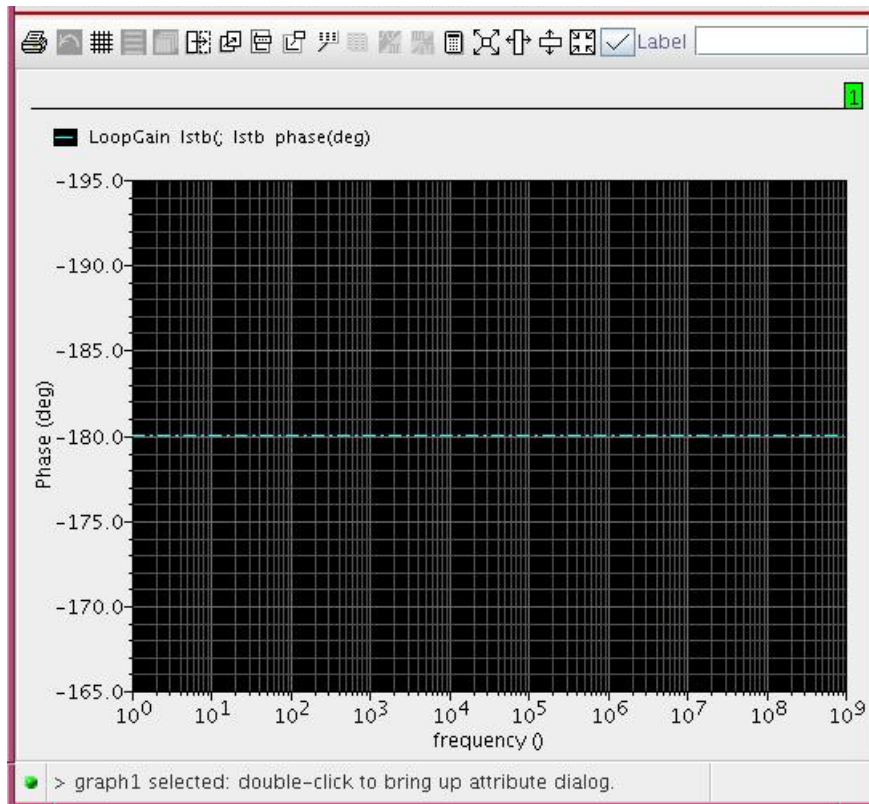


Figure 26 HSPICE Plotting Assistant showing Loop Analysis functions

When you click the **Plot** button with the setup shown in [Figure 26](#) you can plot the waveform of **Loop Gain** with any of the different **Modifiers**. [Figure 27 on page 138](#) shows the waveform phase of Loop Gain.

Chapter 4: Analysis Setup and Design Variables

Loop Stability Analysis



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 27 LSTB analysis Loop Gain phase waveform

When you select other LSTB functions such a **Gain Margin**, **Phase Margin**, **GM Frequency**, **PM Frequency**, and **Gain@miniFreq**, the Plotting Assistant form is prepared to generate the related measure results of LSTB analysis.

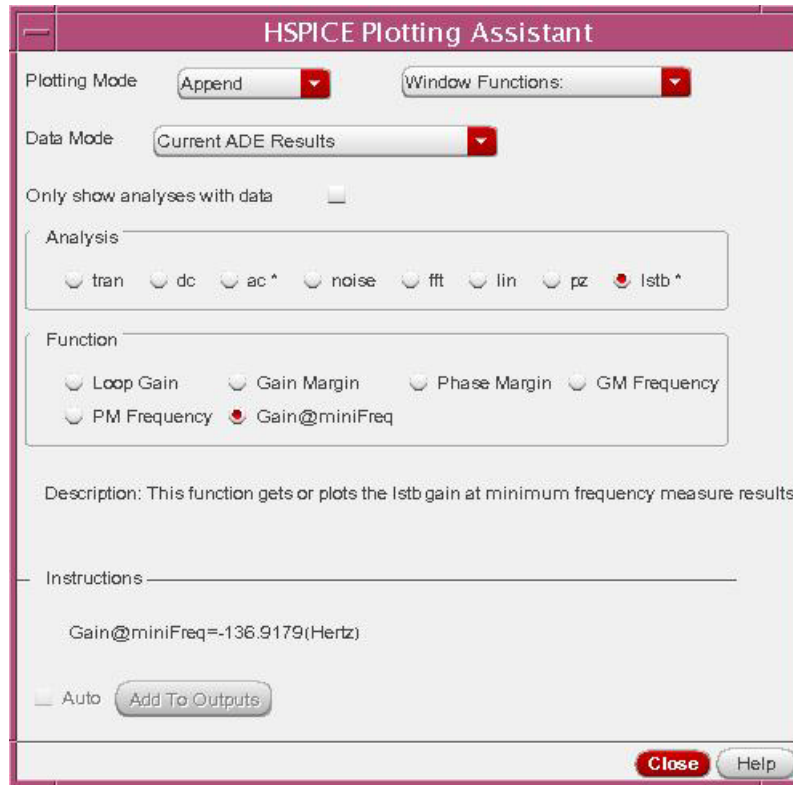


Figure 28 Plot Assistant showing LSTB analysis Gain@miniFreq measure variable value

You can print LSTB analysis measure results using the Environment console by selecting **Results > Print > Loop Stability Summary**. Figure 29 shows the calculated results in the Results Display Window. Note that when you use the Environment console to run a simulation of LSTB analysis measure results, the print window can only show one row of results.

The Results Display Window shows the following results:

Gain Margin(dB)	Phase Margin(deg)	GM Frequency(Hertz)	PM Frequency(Hertz)	Gain@miniFreq(Hertz)
166.9577	failed	1.024e+05	failed	-136.9179

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 29 Results Display Window showing measurements for the various Functions

Checking a Monte Carlo Analysis of LSTB Results

After running a LSTB analysis simulation using the HSPICE Monte Carlo tool, open the HSPICE Plotting Assistant form and select **lstb** analysis. (See [Chapter 8, Monte Carlo in the HSPICE Integration](#) for details on this tool). As seen in [Figure 30](#) the **Group Data** section is shown with **Monte Carlo** selected. Use the **Replot** button in this form to generate the waveform of 'Loop Gain' with different modifiers. [Figure 30](#) shows the Plotting Assistant window and waveform window displaying the magnitude Loop Gain simulation results of an HSPICE Monte Carlo/LSTB analysis.

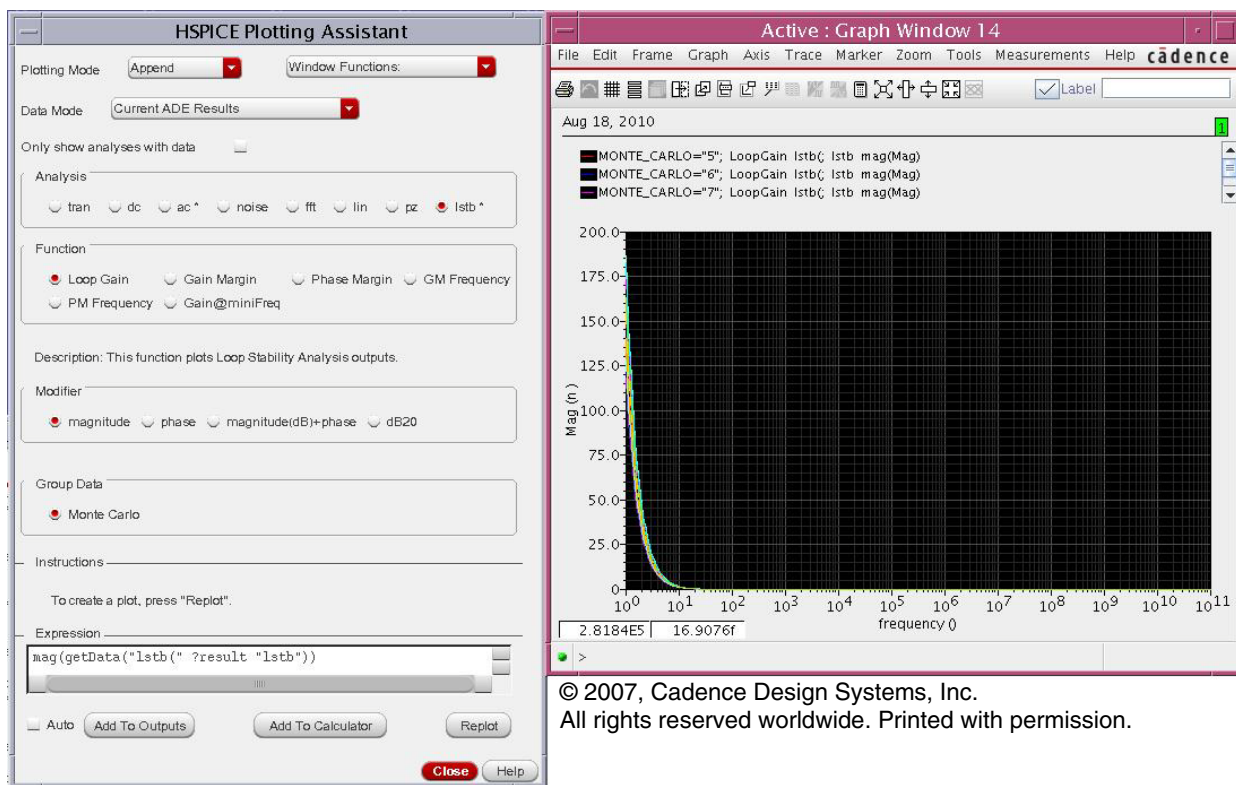
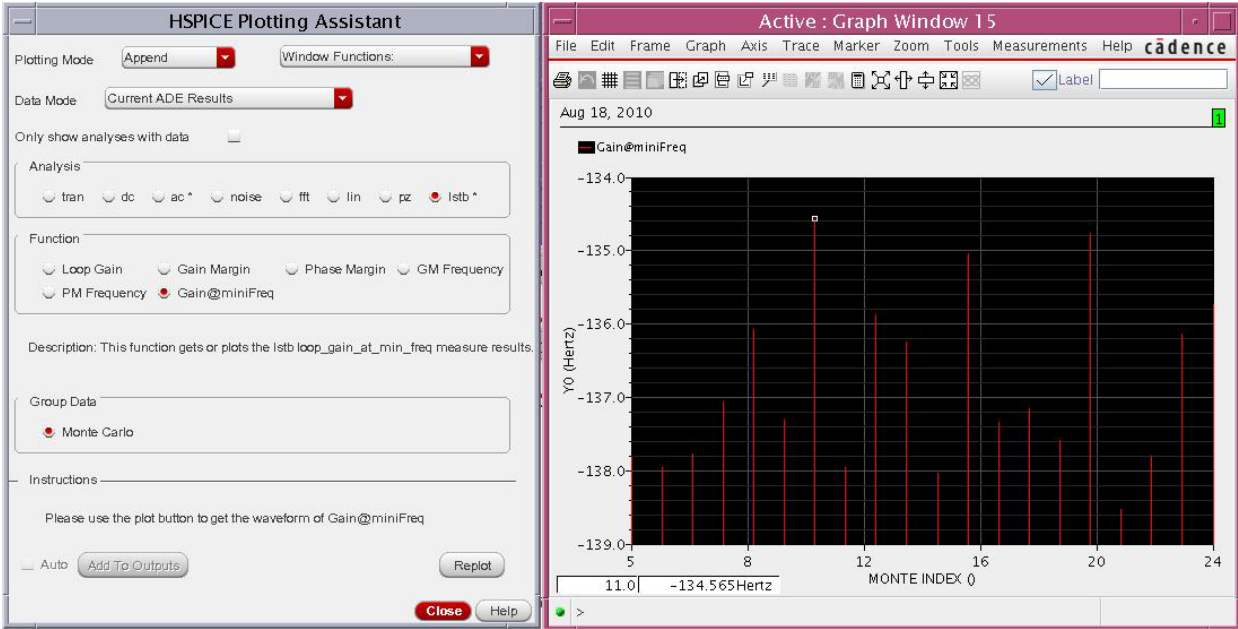


Figure 30 Plotting Assistant for Monte Carlo run and LSTB analysis Loop Gain magnitude waveform

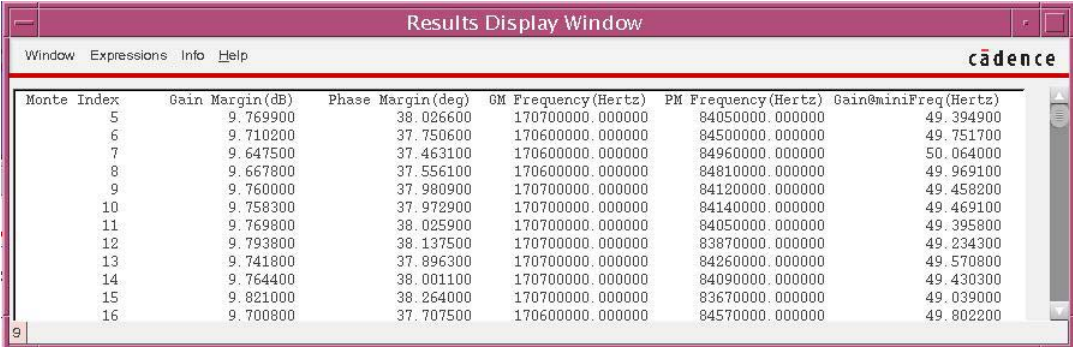
The LSTB analysis measurement variable results can be plotted in Plotting Assistant with HSPICE Monte Carlo **lstb** analysis simulation results. The Plotting assistant form In [Figure 31 on page 141](#). The **Instructions** advise clicking the **Replot** button to plot the measurement results waveform.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 31 HSPICE Monte Carlo lstdb analysis Plotting Assistant form and Gain@miniFreq waveform

You also can print the HSPICE Monte Carlo simulation lstdb analysis measurement results summary; Figure 32 shows the print window:



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 32 HSPICE Monte Carlo lstdb analysis print results window

Unlike generating only a single line of measure results through the Environment console (Figure 29 on page 139), the HSPICE Monte Carlo/LSTB

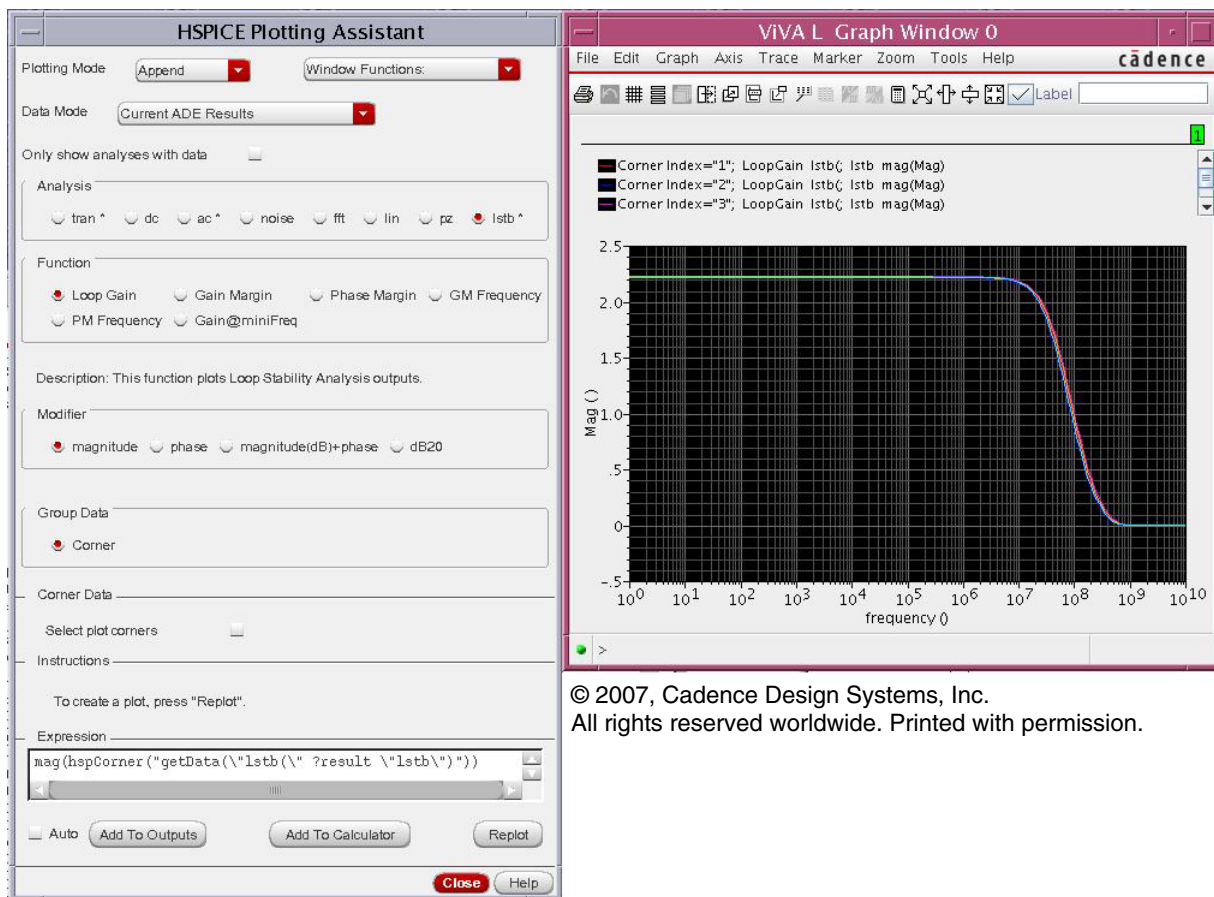
Chapter 4: Analysis Setup and Design Variables

Loop Stability Analysis

analysis measure results shows a print window that displays more **Monte Index** columns and the print results include many more rows.

Checking HSPICE Corner Analysis Simulation LSTB Analysis Results

After running an LSTB analysis simulation using the HSPICE Corner Analysis tool (see [Chapter 9, Corners Analysis](#)), open the HSPICE Plotting Assistant form and select **lstb** analysis. The form looks like [Figure 33](#) with the **Group Data** and **Corner Group** shown. Use the **Replot** button in this form to plot the waveform of Loop Gain with different modifiers. [Figure 33](#) shows the Plotting Assistant window and waveform window of the magnitude Loop Gain of HSPICE corner analysis/LSTB analysis simulation results.



© 2007, Cadence Design Systems, Inc.
All rights reserved worldwide. Printed with permission.

Figure 33 HSPICE Corner Analysis with lstb analysis simulation Plotting Assistant form and waveform of Loop Gain

The LSTB analysis measurement variable results can be prepared in the Plotting Assistant using HSPICE corner analysis and lstb analysis simulation

results. The **Instructions** advise clicking the Replot button to plot the measurement results waveform.

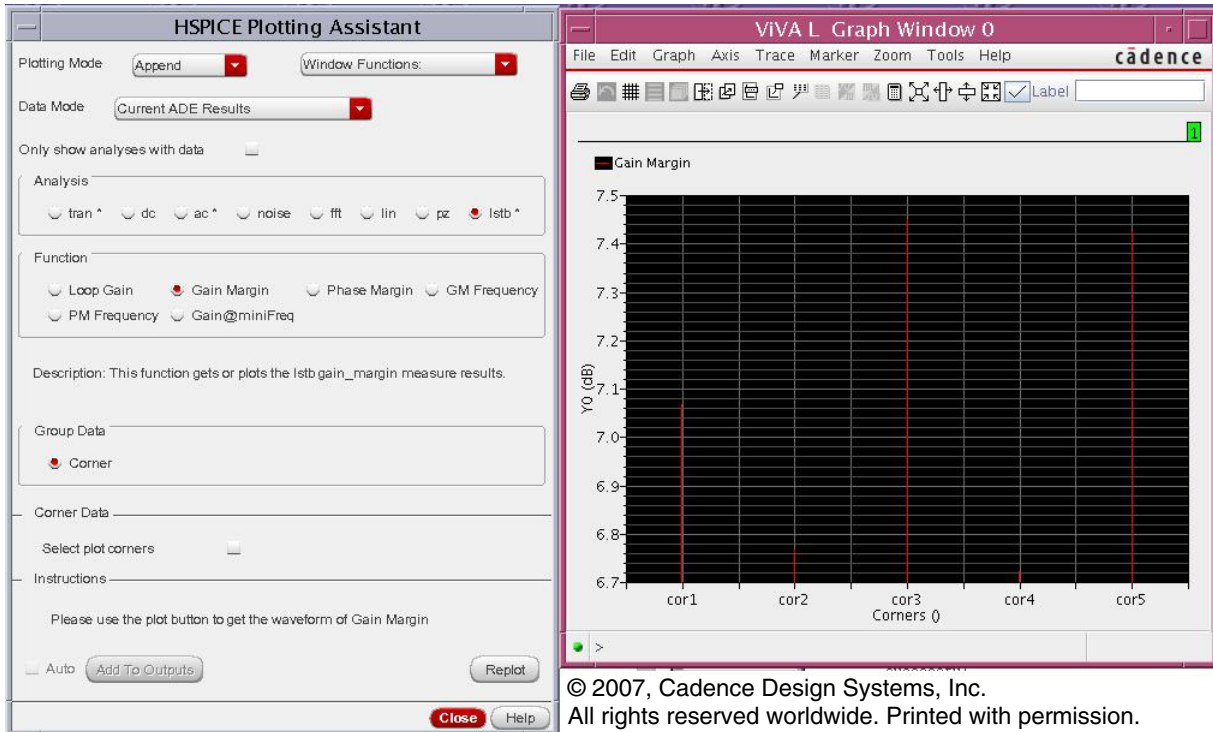


Figure 34 HSPICE Corner Analysis with lstdb analysis simulation Plotting Assistant form and waveform of Gain Margin

You also can print the HSPICE Corner analysis simulation lstdb analysis measurement results summary; Figure 35 shows the print window:

The figure shows the 'Results Display Window' with a table of measurement results for five corners. The table has columns for Corners, Gain Margin(dB), Phase Margin(deg), GM Frequency(Hertz), PM Frequency(Hertz), and Gain@miniFreq(Hertz). The data is as follows:

Corners	Gain Margin(dB)	Phase Margin(deg)	GM Frequency(Hertz)	PM Frequency(Hertz)	Gain@miniFreq(Hertz)
cor1	7.0673	19.3037	1.811e+08	9.262e+07	6.9184
cor2	6.7663	18.3640	1.637e+08	8.606e+07	6.9015
cor3	7.4493	20.3376	2.030e+08	1.003e+08	6.9365
cor4	6.7208	18.1500	1.831e+08	9.668e+07	6.9095
cor5	7.4233	20.4741	1.792e+08	8.868e+07	6.9289

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 35 Measurement results for Corner/LSTB analyses

Chapter 4: Analysis Setup and Design Variables

Loop Stability Analysis

Unlike generating only a single line of measure results through the Environment console (Figure 29 on page 139), the HSPICE Corner/ Istb analysis measure results generates a print window that displays more **Corners** columns and the print results include several more rows.

Checking Parametric Simulation and LSTB Analysis Results

After running an LSTB analysis simulation using the Parametric Analysis tool, open the HSPICE Plotting Assistant and select **Istb** analysis as shown in Figure 36. Note that the **Group Data** section is shown with Parametric selected. Use the **Replot** button in this form to plot the waveform of 'Loop Gain' with different modifiers and sweep variables. Figure 36 shows the Plotting Assistant window and waveform window phase of Loop Gain of Parametric Istb analysis.

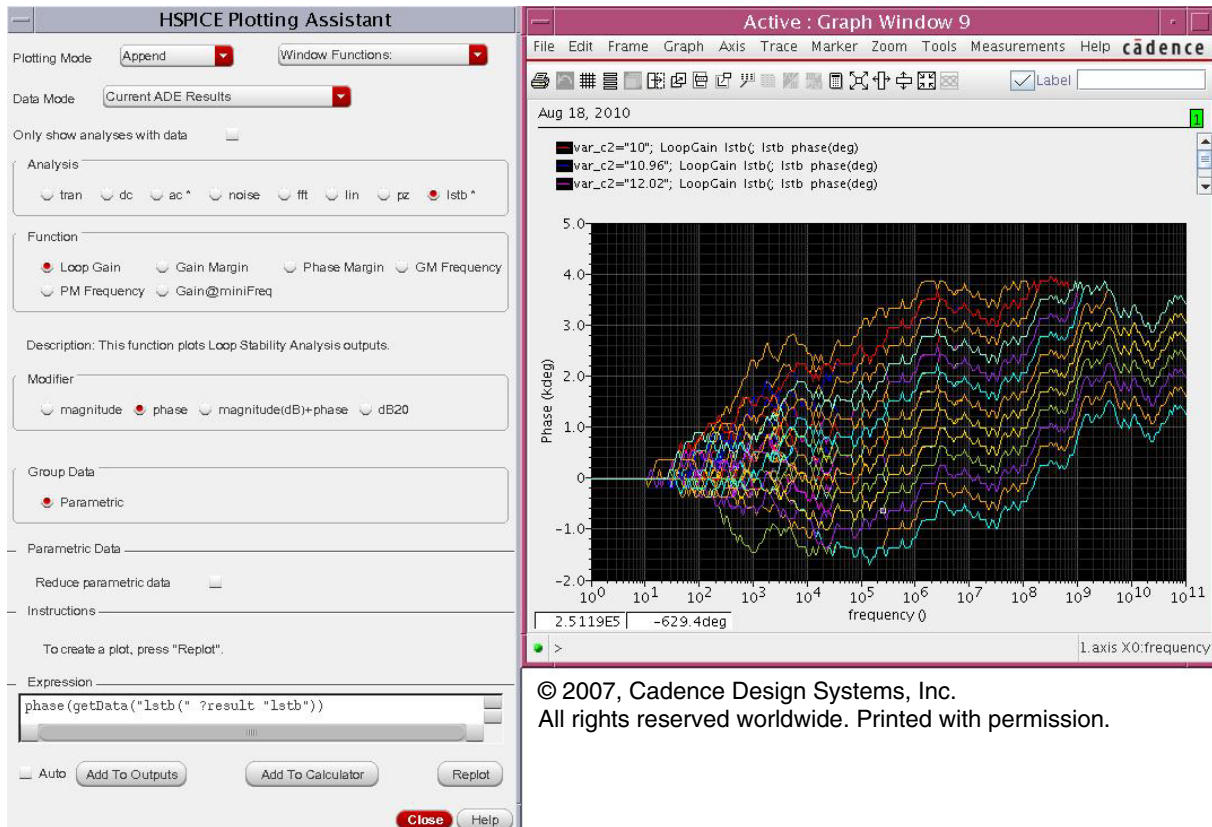
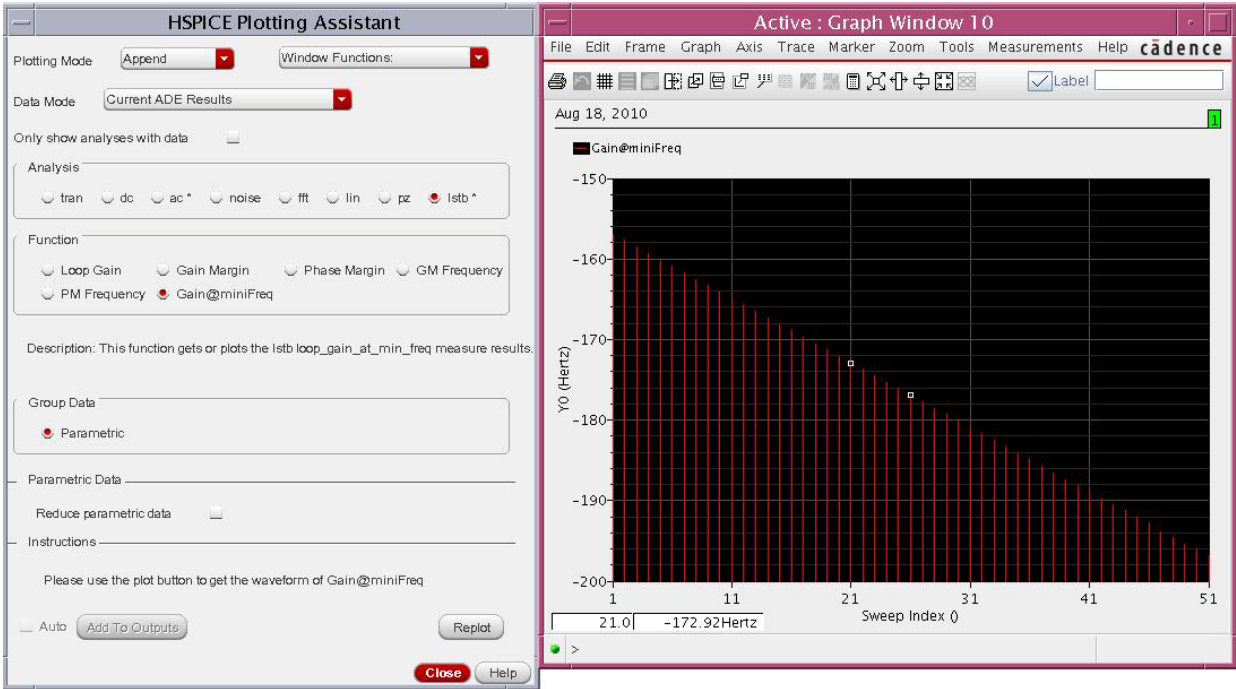


Figure 36 Parametric /LSTB analysis Plotting Assistant and waveform of Loop Gain.

You can use the Plotting Assistant to generate a plot of the Parametric/LSTB analysis measurement variable results. The **Instructions** advise to click the **Replot** button to plot the measurement results waveform. See [Figure 37](#).



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 37 Measure results plotted for parametric/lstb analyses

You also can print the Parametric Lstb analysis measure results summary ([Figure 38](#)).

The figure shows the 'Results Display Window' from the Cadence Virtuoso environment. It contains a table with the following data:

Sweep Index	var_c2	Gain Margin(dB)	Phase Margin(deg)	GM Frequency(Hertz)	PM Frequency(Hertz)	Gain@miniFreq(Hertz)
1	10	265.6101	failed	35.0126	failed	-156.9169
2	10.9648	-3.579e+03	failed	0.	failed	-157.7195
3	12.0226	failed	failed	failed	failed	-158.5182
4	13.1826	-5.616e+03	failed	0.	failed	-159.3183
5	14.4544	-453.8762	failed	1.030e-09	failed	-160.1175
6	15.8489	failed	failed	failed	failed	-160.9194
7	17.378	-6.059e+03	failed	0.	failed	-161.7184
8	19.0546	238.7858	failed	719.9328	failed	-162.5151
9	20.893	failed	failed	failed	failed	-163.3186
10	22.9087	222.8996	failed	226.6224	failed	-164.1174
11	25.1189	254.2828	failed	149.8835	failed	-164.9156

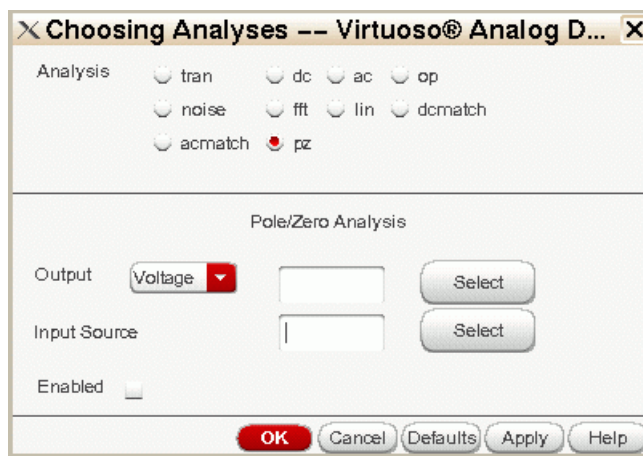
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 38 Parametric/Lstb analysis measure results

For Parametric/LSTB analysis measure results, the print window shows multiple **Sweep Index** and sweep variables columns, and the print results include several rows.

Pole/Zero Analysis

On the Choosing Analyses form click the **pz** radio button to set up a Pole/Zero analysis.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 39 Pole/Zero Analysis form

Set up a Pole/Zero analysis by filling in the form's two fields.

Use the following controls:

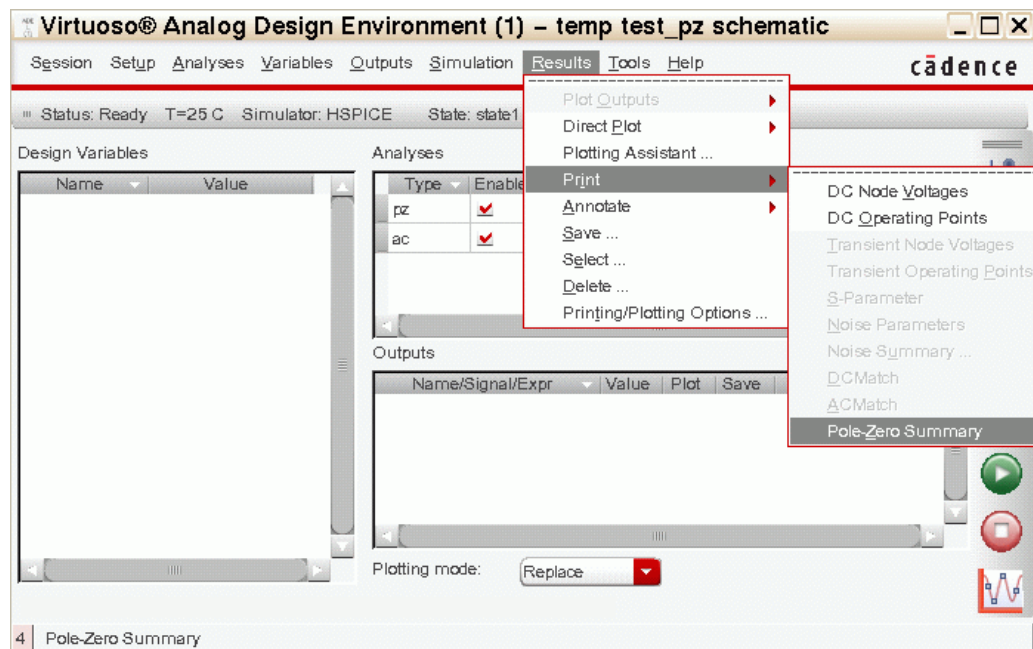
1. For **Output** select either **Voltage** or **Current**.
2. Use the Output **Select** button to identify appropriate voltages or currents of interest in the design.
3. For the **Input Source**, manually enter the source of interest or use the **Select** button to define the source.
4. If not already enabled, click the **Enabled** check box and then click **OK**.

Simulating and Plotting P/Z Results

After setting up your Pole/Zero analysis, simulate in the usual manner (Environment console: **Simulation** menu) and the log file will record the simulation process. There are two ways of viewing the PZ results: through the Print Summary or through plotted waveforms.

Using the Print Summary

On the Environment console menu bar, select **Results > Print > Pole-Zero Summary** (Figure 40 on page 147).



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 40 Console: Results > Pole-Zero Summary

Plotting the Pole/Zero Result

To plot the Pole/Zero result:

- Select **Results > Plotting Assistant**.
- In the HSPICE Plotting Assistant form **Analysis** section, choose the **pz** radio button, then select the **Function** of your choice **Poles/Zeros**, **Poles** (only) or **Zeros** (only).

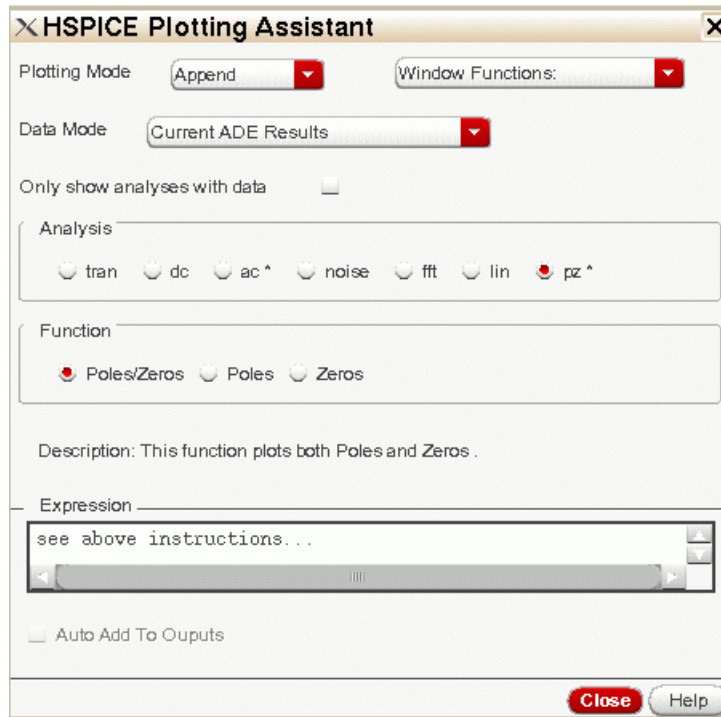


Figure 41 Plotting Assistant for Pole/Zero results

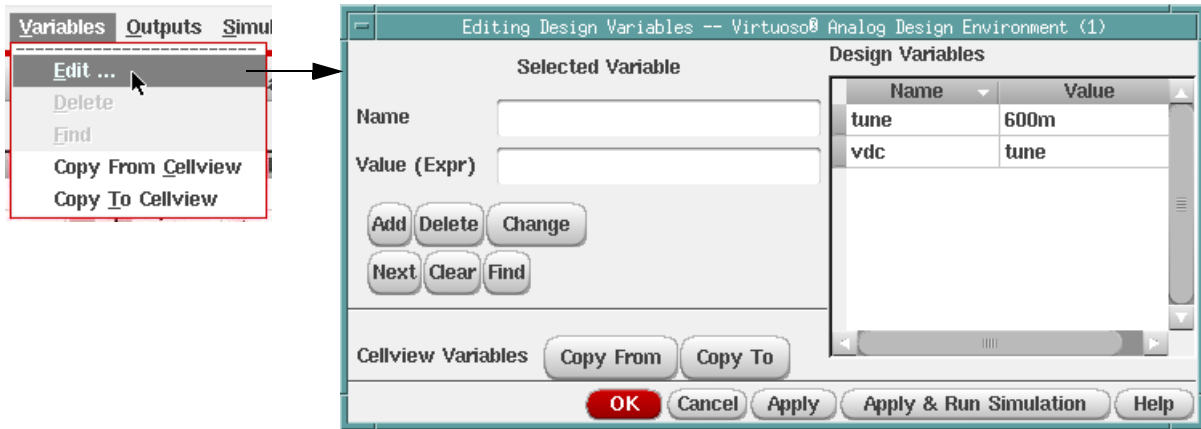
Other Features Supported in a PZ Analysis

As with other analyses, you can save or load a session state. Use the Environment console **File** menu. In addition you can save to save and load all user operations for simulation, and create an OCEAN script based on all operations performed in a HSPICE PZ session through the UI.

For more information on Pole-Zero analysis see [Pole-Zero Analysis](#) in the *HSPICE User Guide: Simulation and Analysis*.

Design Variables

The **Variables** menu item provides options to edit and copy to or from the cellview in the Schematic Editor. To open the Editing Design Variables form, on the Environment Console select **Variables > Edit**.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 42 Editing Design Variables Form

The Editing Design Variables form provides the following controls:

- The **Name** field allows you to enter a name for the variable, which is listed in the Design Variables Name column when you click **Apply** or **OK**.
- The **Value (Expr)** field is for entering a value or expression for the design variable, and it too is registered in the Design Variables table under the Value when applied or OK'd.
- **Add** creates the variable you specified in the Selected Variable fields.
- **Delete** removes a highlighted variable. (Click on a variable to highlight it.) This button is active only when a variable is selected in the table, and will delete the variable with no confirmation.
- **Change** updates the highlighted variable from the Selected Variable fields to the Design variable table.
- **Next** highlights the following signal or expression in the Design Variables table.
- **Clear** erases the Selected Variable fields so you can enter a new variable.
- **Find** works with selection in the table and will hierarchically highlight the instance that uses that variable.
- **Copy From** copies the variables in the schematic cellview into the Environment Console.

Chapter 4: Analysis Setup and Design Variables

Design Variables

- **Copy To** copies the variable values in the Environment Console to the schematic cellview.
- The **Design Variables** table lists the name and value of each variable in the design. Note that variables can be set to real values, to string values, and even to built-in expressions supported by HSPICE. The netlister will netlist what is there and let the simulator handle any evaluation.

Clicking the **Apply & Run Simulation** button initiates a simulation with the enabled or netlisted analyses and any design variables on the form.

Saving-Plotting Outputs

This chapter describes the Outputs menu selections from the Setting Outputs form available to the HSPICE integration.

Before you simulate a circuit, you can decide which data outputs you want to save and plot or save only. The Outputs menu lets you save a selected set of voltages and currents from the schematic. For example, in [Figure 43](#) the names of the signals or expressions to be saved and plotted are listed in the **Outputs** table along with designations of **Plot**, **Save**, and **Save Options**, up to 999 outputs. Outputs you want to plot *must be saved* to disk in the HSPICE integration.

Note: Saving all the node voltages and terminal currents for a large design produces a very large data set.

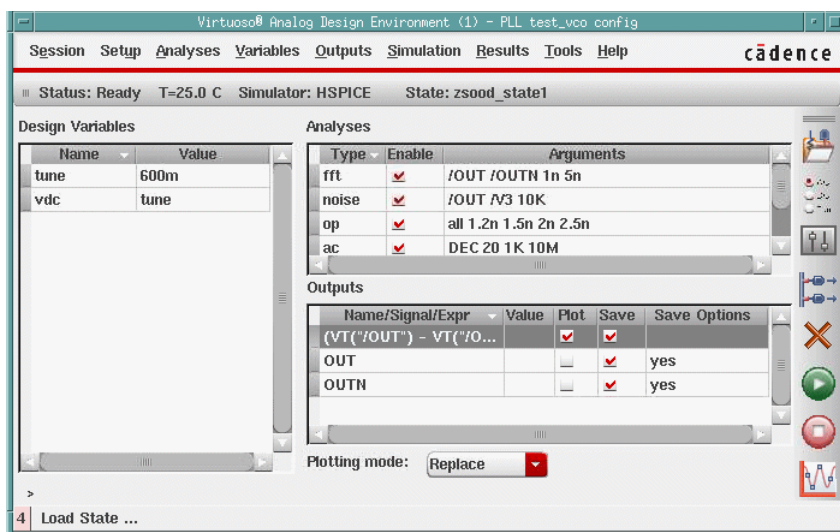


Figure 43 Environment Console Listing Save/Plot Outputs

Chapter 5: Saving-Plotting Outputs Setup

The Outputs menu on the Environment Console offers the following options:

- [Setup](#)
- [Delete](#)
- [To Be Saved](#)
- [To Be Plotted](#)
- [Save Options](#)

This chapter also describes [Saving and Restoring a List of Outputs](#).

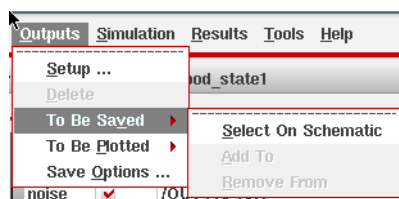


Figure 44 Outputs Menu

Setup

The HSPICE integration to the Cadence® Virtuoso® Analog Design Environment keeps track of two sets of nets and terminals:

- The saved set, for which simulation data is written to disk.
- The plotted set, which is automatically plotted after simulation in the Waveform window (only if **Save** is also marked in the Outputs matrix); this set can also contain expressions.

The content of these sets of outputs is listed in the Outputs matrix of the Environment Console ([Figure 43 on page 151](#)), and in the Setting Outputs form in the Table Of Outputs. Signals marked as **Plot** must also be marked **Save** in the **Outputs** matrix before they can be plotted.

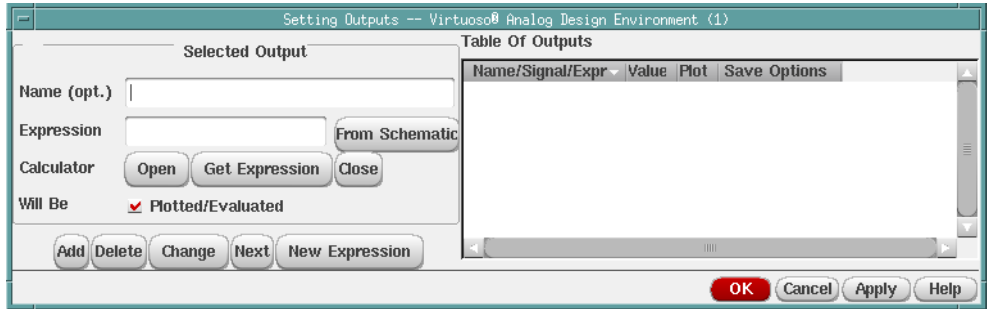


Figure 45 Setting Outputs Form

The following fields and controls are available:

- The **Name** field is optional for manual entry, since a set of output nodes and terminals you select from the schematic can be automatically entered in the Table of Outputs. A name entered here appears on the Table of Outputs and the Waveform window.
- The **Expression** field lets you build an expression using one of the following methods:
 - Type the expression in the **Expression** field.
 - Use the calculator to build the expression and then retrieve it by clicking **Get Expression**.
 - The **From Schematic** button allows you to select an expression on the schematic and auto-fill the form for the selected net or terminal. The schematic GUI circles pins when you choose a current and highlights wires when you choose a net. For example: Selecting OUTN on the schematic changes the **Expression** and **Calculator** fields to **Signal** and **Type** fields with /OUTN and voltage as the auto-filled entries.
 - The **Plotted** check box is automatically checked for the /OUTN signal.
 - **Expression** is the calculator expression to save or save/plot.
 - You need to also check **Saved** to create a successful plot.



- The **Open** button launches the tool's built-in calculator. See Cadence documentation for information.
- The **Get Expression** button retrieves an expression from the calculator buffer and pastes it into the **Expression** field.
- The **Close** button dismisses the calculator.
- The **Will Be Plotted/Evaluated** check box changes to two check boxes depending on whether an expression or a signal is selected. **Plotted** and **Saved** check boxes replace the single check box if a signal for a net or terminal is entered.
- Click **Add** to add the expression the Table Of Outputs list.
- The **Delete** button clears one or more selected expressions from the Table of Outputs and from the Selected Outputs area.
- The **Change** button helps you suppress the plotting of an expression without deleting it.

To do this:

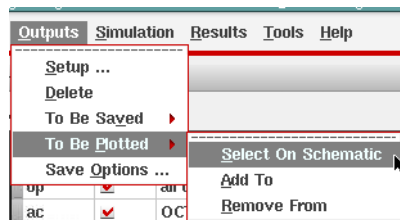
- Click an existing expression in the Table of Outputs list.
- Deselect the **Will be Plotted/Evaluated** check box.
- Click **Change**. As a result, when you select **Results > Plot Outputs** following the simulation, the expression you deactivated remains unplotted.
- If you have multiple signals or expressions entered in the Table Of Outputs and one or more is highlighted, the **Next** button shifts the selection to the next lower row of the list. You then can perform any of the appropriate preceding actions on the selected row. This allows you to make changes to consecutive entries in the Table Of Outputs list without clicking on each entry.
- The **New Expression** button clears all previous entry the Selected Output section so you can type in a new output.
- Clicking **Apply** or **OK** sets up the specified outputs for saving and plotting in the results phase of post-processing.

Delete

Clicking **Delete** when a row is selected in either the Table of Outputs of the Setting Outputs form or the Environment Console Outputs matrix, immediately removes the entry from both forms.

To Be Saved

The **To Be Saved** menu option expands to present three sub-options.



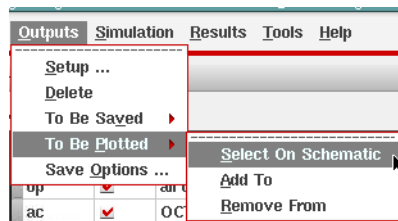
- Clicking **Select on Schematic** activates the schematic so you can select a net or terminal to immediately add it to both the Setting Outputs form Table Of Outputs and the Environment Console Outputs matrix as an output to be saved along with check marks for the data to be saved.
- **Add To** checks off the check box to schedule the schematic-selected net or terminal to be saved.
- **Remove From** deselects a net or terminal that was scheduled to be saved. The net or terminal remains listed, but unchecked.

To Be Plotted

The **To Be Plotted** menu option expands to present a three sub-options.

Chapter 5: Saving-Plotting Outputs

Save Options



- Clicking **Select on Schematic** activates the schematic so you can select a signal node or net to immediately add it to both the Setting Outputs form Table Of Outputs and the Environment Console Outputs matrix as an output to be plotted along with check marks for the data to be plotted.
- Clicking **Add To** to a schematic-selected net or terminal schedules the signal to be plotted.
- **Remove From** deselects a net or terminal that was scheduled to be plotted. The net or terminal remains listed, but unchecked.

Save Options

Selecting the **Save Options** menu option launches the Save Options form. This dialog allows you to make multiple choices prior to simulation for the saving of voltages, and current and power signals. The **Save Options** column on the Environment Console adds a “yes” for each signal that is scheduled to be saved. Making selections and clicking **Apply** or **OK** will change the standing of previously checked Plot or Save options in the Setting Outputs Table of Outputs and Environment Console Outputs matrix. You must click the **OK** or **Apply** button for your choices to take effect.

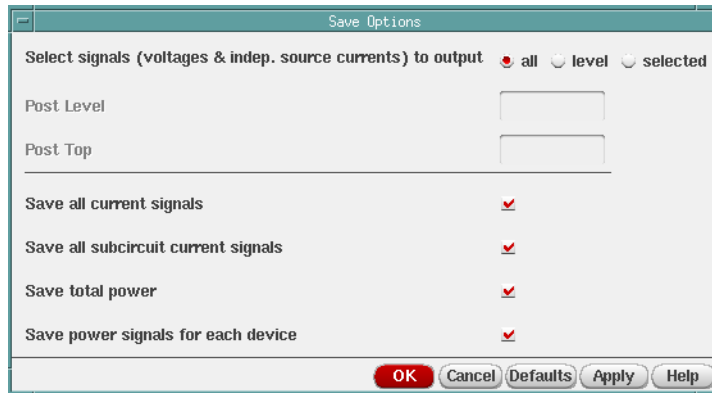


Figure 46 Save Options Checklist for Outputs

The dialog consists of two sections. The upper section provides three radio buttons that allow you to specify which voltages and independent source currents signals to save to output according to the categories of **all**, **level**, or **selected**. These buttons only refer to *voltages and independent source currents signals*.

The lower section provides check boxes for choices to:

- Save all current signals
- Save all subcircuit current signals
- Save total power
- Save power signals for each device

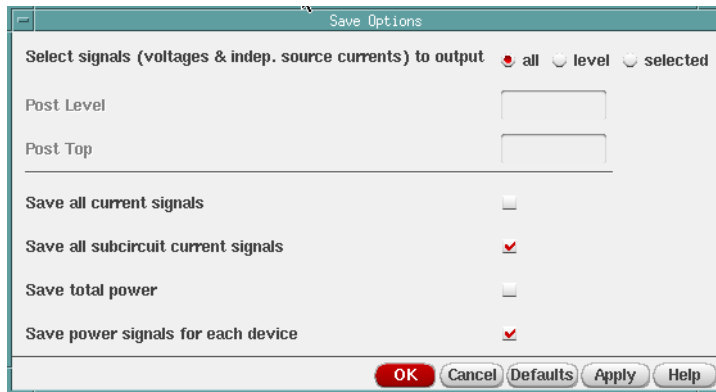
You can save a combination of signals by using both portions of the Save Options dialog.

In the scenario below, the Save Options form specifies that the following signals will be saved to output:

- All voltages and independent source current signals
- All subcircuit current signals across the design
- Power signals for each device in the design
- All individual current signals listed in the Outputs matrix on the Environment Console.

Chapter 5: Saving-Plotting Outputs

Save Options

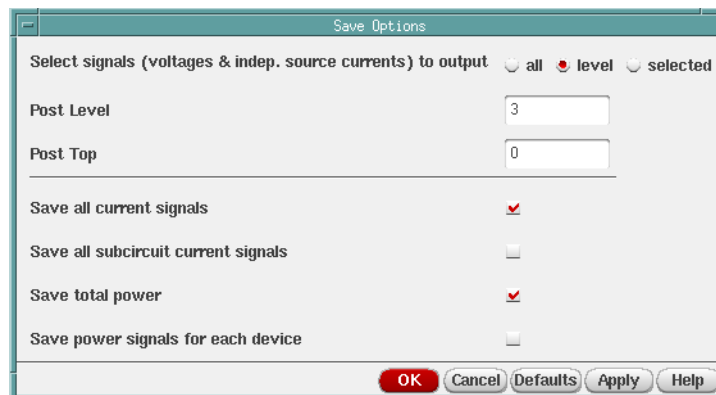


The 'Save Options' dialog box is shown with the 'all' radio button selected. The 'Post Level' and 'Post Top' fields are empty. The checkboxes for 'Save all current signals', 'Save total power', and 'Save power signals for each device' are unchecked. The checkbox for 'Save all subcircuit current signals' is checked.

Option	State
Select signals (voltages & indep. source currents) to output	all (selected)
Post Level	
Post Top	
Save all current signals	<input type="checkbox"/>
Save all subcircuit current signals	<input checked="" type="checkbox"/>
Save total power	<input type="checkbox"/>
Save power signals for each device	<input checked="" type="checkbox"/>

In the scenario below with **level** selected:

- The Save Options form limits the data written to your waveform file to the third level of nodes for voltages and independent source currents, plus the other checked options, below. (Setting Post Top to 1 limits the saved data to the top level nodes only.)
- All current signals across the design (not limited to the top three levels)
- Total power
- All individual subckt current signals listed in the Outputs matrix

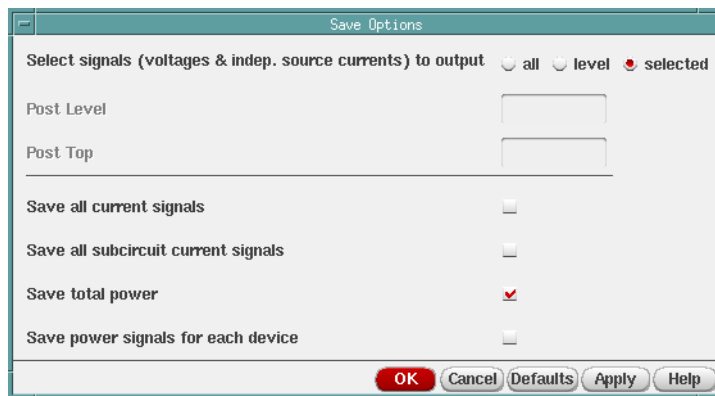


The 'Save Options' dialog box is shown with the 'level' radio button selected. The 'Post Level' field is set to 3 and the 'Post Top' field is set to 0. The checkboxes for 'Save all current signals', 'Save total power', and 'Save power signals for each device' are unchecked. The checkbox for 'Save all subcircuit current signals' is checked.

Option	State
Select signals (voltages & indep. source currents) to output	level (selected)
Post Level	3
Post Top	0
Save all current signals	<input checked="" type="checkbox"/>
Save all subcircuit current signals	<input checked="" type="checkbox"/>
Save total power	<input checked="" type="checkbox"/>
Save power signals for each device	<input type="checkbox"/>

In the scenario below, because the **selected** radio button has been selected, the Save Options form is scheduled to save:

- Those signals already listed in the Outputs matrix of the Environment Console
- Total power
- The individual subckt current signals and the individual current signals specified in the Outputs matrix



Saving and Restoring a List of Outputs

This section describes how to save and restore lists of outputs to be saved and plotted.

Saving a List of Outputs

You can save both

- The data for a set of outputs
- The list of saved and plotted outputs itself (including output expressions).

To save this list:

1. In the Environment Console, select Session > Save State.
2. In the Saving State form appears, type a name for the saved simulation state.

3. Verify that the **Outputs** box is selected and click **OK**.

Note: Saved states are simulator-dependent if you save the analyses. Otherwise, you can restore states from a different simulator.

Restoring a Saved List of Outputs

To restore a saved set of outputs:

1. In the Environment Console, select Session > Load State.

Explanation: The Loading State form displays the state files in the run directory identified by the **Cell** and **Simulator** fields.

2. Select a run directory with the **Cell** and **Simulator** fields to display the saved states for the cell and simulator combination.
3. Click on a state name and verify that Outputs is selected.
4. If Outputs is selected, click **OK**.

Running Simulations and Using Control Options

Describes the interface for running simulations and use of control options.

The following topics are covered in this chapter.

- [Simulation Menu Overview](#)
- [Netlist and Run](#)
- [Run](#)
- [Options](#)
- [Netlist](#)
- [Output Logs](#)
- [Convergence Aids](#)

Simulation Menu Overview

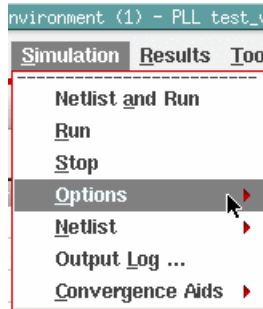
The Simulation menu on the Environment Console displays commands to run a simulation based on previously selected analyses and netlisted simulation parameters for your design, and the means to change simulator options and review the generated text and graphical output files. [Figure 47 on page 162](#) shows the menu with the expanded Options list.

There are two kinds of files generated for simulation:

- The netlist, which contains component information but no simulation control data. This file is named *netlist*.
- The simulator input file, *input.ckt*, contains both the netlist and the simulator control information required. The *input.ckt* data is passed to the HSPICE simulator.

The simulator input file is created with a simulator-specific extension. For example, this file is deposited in the following directory:

/projectdirectory/topCellName/HSPICE/view/netlist/input.ckt



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 47 Menu Options for the Simulation Menu

Netlist and Run

The Netlist and Run menu option runs the incremental hierarchical netlister (ihnl) and runs the simulation. If you have made changes to the schematic, the ihnl will update the */projectdirectory/topCellName/HSPICE/view/netlist/netlist* file with the new component/structural information. It will update the *input.ckt* file with analyses setup changes, simulator options etc., and run simulation. If the Automatic Output Log option (Setup > Environment Setup) is turned on in the Environment Options form, the output log window is displayed and the window is updated dynamically.

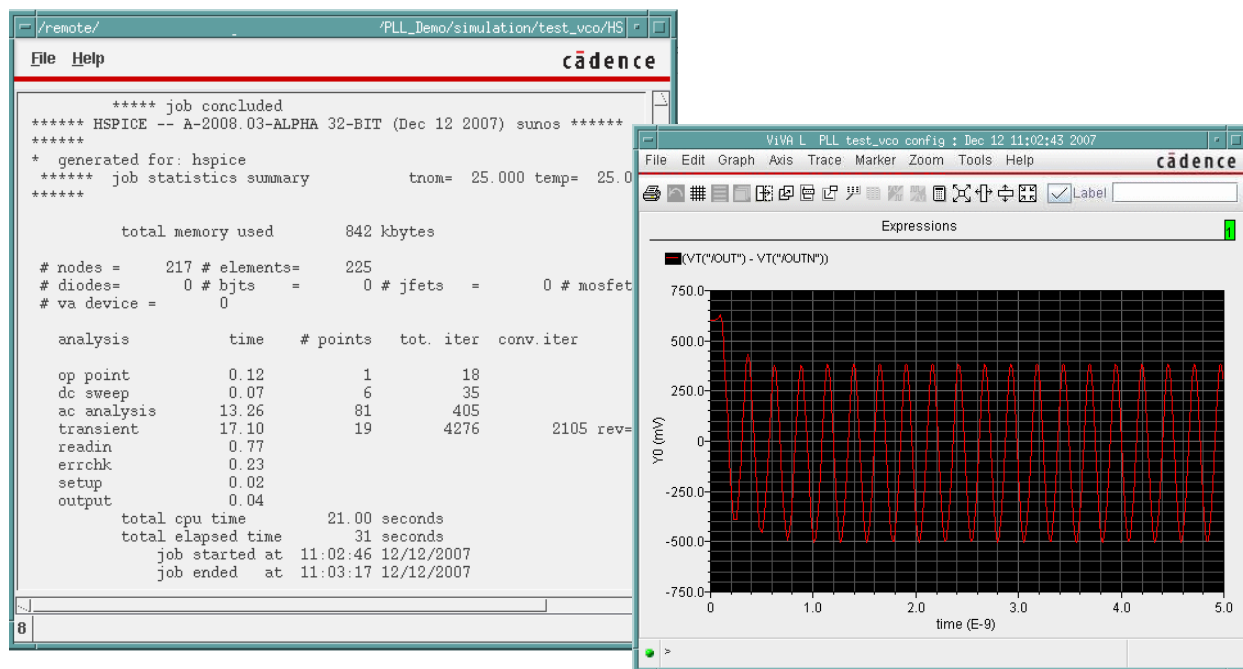
Note: You can use a function provided by HSPICE to run a script after the netlist has been created, but before it is simulated so you can post-process the netlist. The function is `customerHook`. You can redefine it and override it after loading the HSPICE integration to ADE interface. The usage is as follows:

```
load("hook_function_file")
```

Then click **Netlist** and run in ADE. It automatically runs after the netlist has been created, but before it is simulated. For example:

```
procedure(customerHook())
print(asiGetNetlistDir(session))
)
```


In [Figure 48 on page 163](#), note that four analyses were run, Operating Point, DC sweep, AC and transient analyses. shows the waveform generated for the transient sweep of voltage for the expression /OUT and /OUTN.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 48 Output File Job Concluded Section and the Waveform Plot

Run

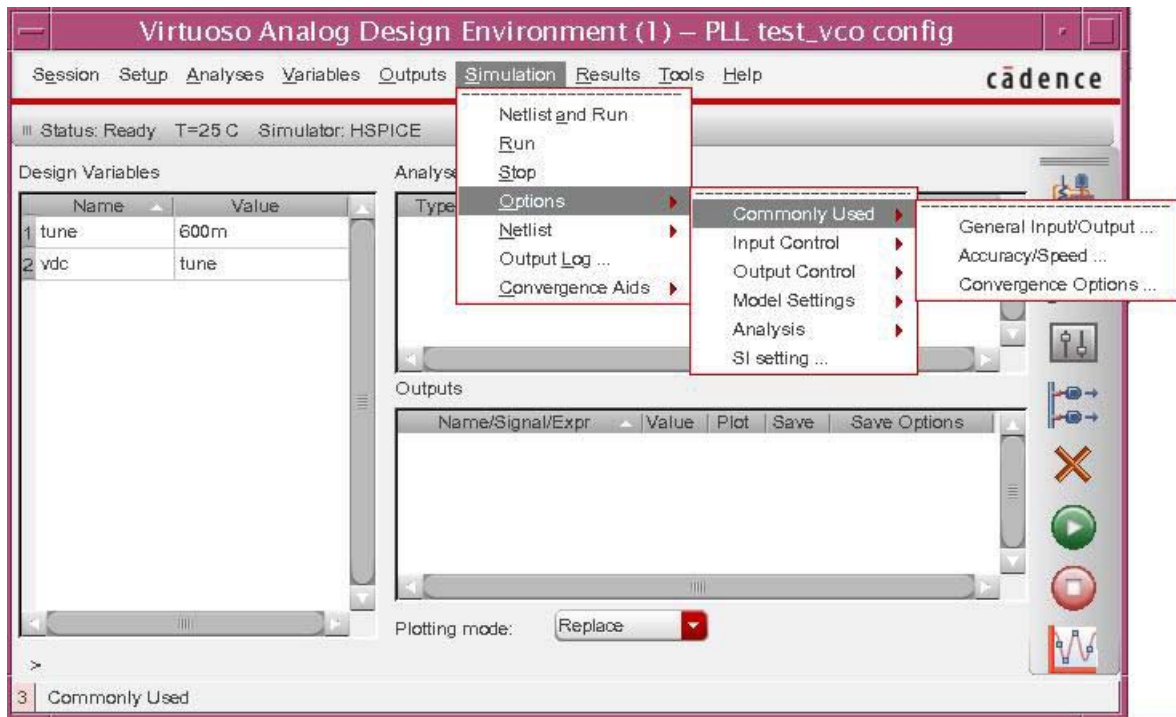
The Run menu option does not run the ihnl. The `/projectdirectory/topCellName/HSPICE/<view>/netlist/netlist` file will not change. Run updates the input.ckt file with environment changes(i.e., analyses setup, design variable changes, simulator option changes, etc.). It does not update any changes in the schematic nor does it update the changes in the stop and switch view lists in Environment Options form.

Stop

The Stop menu command aborts a running simulation.

Options

Selecting any of the **Simulation > Options** sub-menus opens the **Simulation Options** form with the listing of HSPICE options set to their defaults.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 49 Simulation Option menu cascading showing Commonly Used HSPICE Control options

The defaults can be of two types, HSPICE simulator defaults and defaults that come from the Cadence environment via the `.cdsenv` preferences files. The Analog Options form includes the buttons **Cdsenv** and **HSPICE**, which will restore the values in the form to the default values set by the Cadence environment or the Synopsys HSPICE tool. For anyone unfamiliar with the Cadence environment, the `cdsenv` value is the result of a `.cdsenv` file loading process. This is done based on a default sequence (installation, then project, then user), but can be impacted by setting the `CDS_LOAD_ENV` environment variable.

The **Options** sub-menus ([Figure 49 on page 164](#)) reflect categories of the HSPICE simulator control options:

- **Commonly Used** — includes the categories: **General Input/Output**, **Accuracy/Speed**, and **Convergence Options**.
- **Input Control** — includes the categories: **Input value limits**, **Parameter Setting Control**, **Verilog-A module**, **Compatible mode**, and **Miscellaneous**
- **Output Control** — includes the categories: **Log file formatting**, **Waveform files**, and **.Meas Options**
- **Model Settings** — includes the categories: **General**, **MOSFET model**, **Custom CMI model**, **TMI model**, **BJT**, and **Diode**
- **Analysis**

The expanded **Analysis** menu item includes:

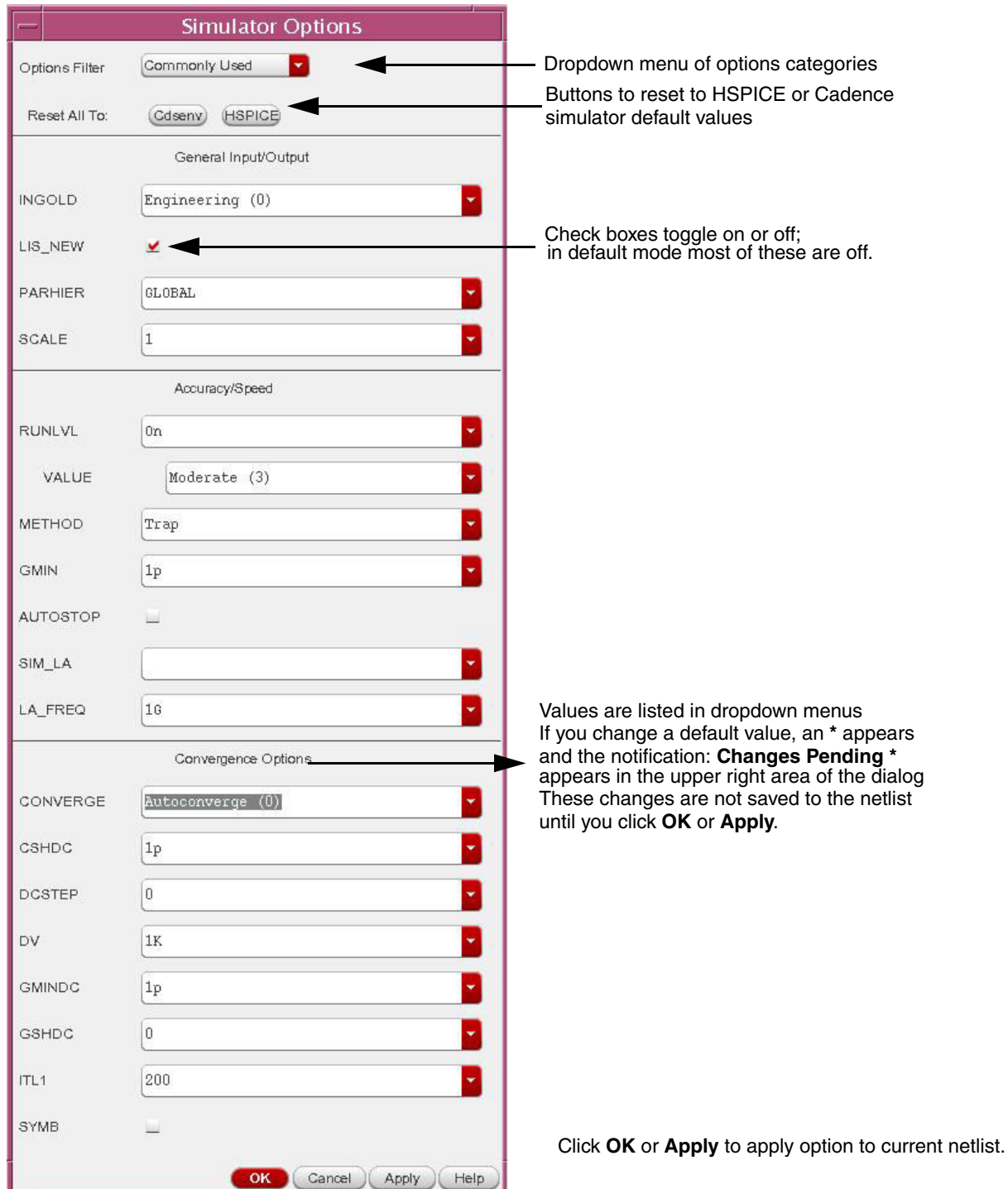
- Transient
 - DC/OP
 - Matrix Control
 - Tolerance
 - Pole/Zero
 - RC Reduction
- **SI setting**

Online html help is available for each labeled option by clicking the **Help** button of the Simulator Options form.

Clicking **OK** or **Apply** adds the control option and value to the simulation input file. To verify that your selected options are added, select **Simulation > Netlist > Recreate** and view the circuit's netlist file. Each of the Simulator Options forms contain the UI controls shown and described in [Figure 50 on page 166](#):

Chapter 6: Running Simulations and Using Control Options

Simulation Menu Overview



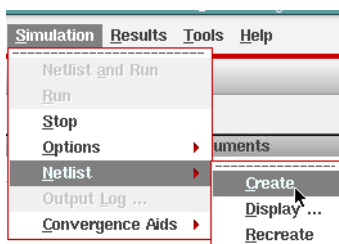
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 50 Analog Options dialog for Commonly Used Options

Netlist

The netlist contains component information but no simulation control data. (The simulator input file, which contains both the netlist and the required simulator control information, is passed to the HSPICE simulator.)

You can Create, Display, and Recreate the simulation input file using the menu options for Netlist.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 51 Netlist Menu Options

- **Create** allows you to create the hierarchical netlists incrementally. The incremental netlisting is faster than full hierarchical netlisting because the environment updates the netlist only for those schematics that have changes since it created the previous netlist.
- **Display** opens an existing simulation input file.

Chapter 6: Running Simulations and Using Control Options

Simulation Menu Overview



```
* Generated for: HSPICE
* Generated on: Jan 24 13:34:53 2008
* Design library name: PLL
* Design cell name: test_vco
* Design view name: config
.global 0 vdd! vss!
.option search='.'
.param
+ tune=600e-3
+ vdc='tune'
.temp 25
.option ARTIST=2 PSF=2
.option
.lib '/remote/PLL_demo/PLL_Demo/cmos90nm.l
* Library name: PLL
* Cell name: cs_inv
* View name: schematic
* Inherited view list: hspiceD spice cmos_sch cmos.sch schematic
.subckt cs_inv out in nbias pbias
m2 out in net5 vss! nch L=100e-9 W=250e-9
m3 net5 nbias vss! vss! nch L=250e-9 W=10e-6
m0 net16 pbias vdd! vdd! pch L=250e-9 W=10e-6
m1 out in net16 vdd! pch L=100e-9 W=500e-9
.ends
* End of subcircuit definition.
* Library name: PLL
* Cell name: vco
* View name: schematic
* Inherited view list: hspiceD spice cmos_sch cmos.sch schematic
.subckt vco fout foutn lfin
+ rmodel=str('defModel')
c0 fout foutn 1e-15
xi8 net27 net26 net30 net087 cs_inv
xi9 net23 net27 net30 net087 cs_inv
xi10 net26 net23 net30 net087 cs_inv
xi11 net15 net26 net30 net087 cs_inv
xi12 fout net15 net30 net087 cs_inv
xi13 foutn fout net30 net087 cs_inv
m2 net087 lfin vss! vss! nch L=250e-9 W=10e-6 OFF
m3 net30 net30 vss! vss! str(rmodel) L=250e-9 W=10e-6
m0 net087 net087 vdd! vdd! pch L=250e-9 W=10e-6 OFF IC=1 2 3
m1 net30 net087 vdd! vdd! pch L=250e-9 W=10e-6
.ends
```

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 52 Sample Netlist

- **Recreate** allows you to create the simulation input file using full hierarchical netlisting (all schematics are forced to re-netlist).

Output Logs

When the **Popup Output Log** option is turned on in the Environment Options form (**Setup > Environment**), the tool automatically opens an HSPICE Log Files window that displays simulator messages. You can select from a **Logfiles** menu to display a listing file ***.lis**, standard out ***.st0**, or **warning/error** file.

Convergence Aids

When convergence issues arise, users of the HSPICE integration can employ various techniques to bias or force a circuit towards a particular solution. Convergence issues occur frequently in designs where MOSFETs have gate voltages below the threshold needed to turn them on (such as an inverter with an input voltage at midway between the supplies) and therefore the internal node(s) might have an infinite number of solutions.

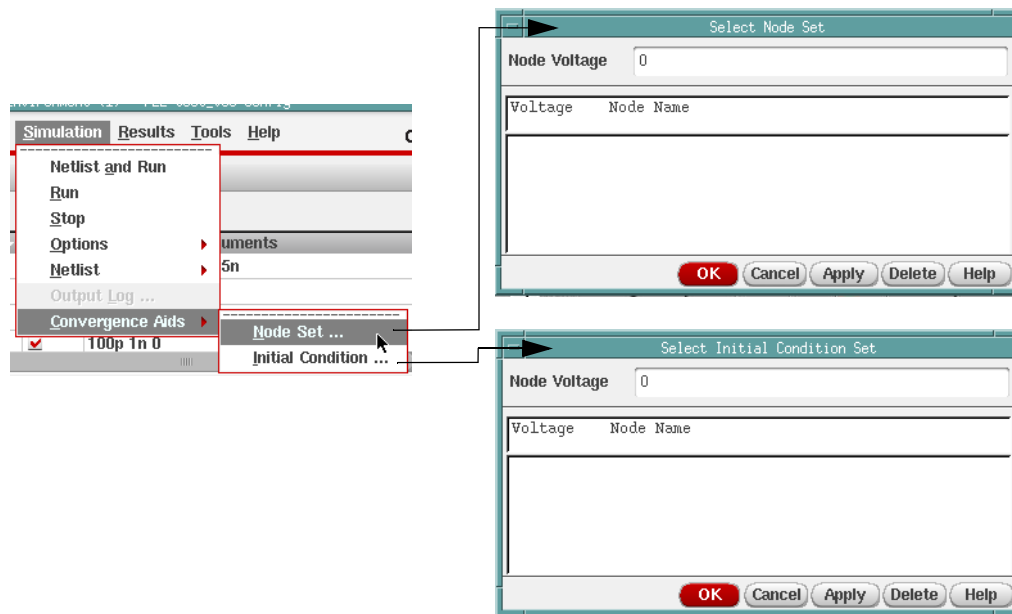
The ability to bias a node to a particular value is also important in analysis of oscillators, as it enables the designer to start the oscillation. The convergence aids options may help the simulator find a solution when it fails to achieve convergence. Once you get the simulation to converge, you can save the DC and Transient solutions. When you re-simulate, you can save time by restoring the saved solutions.

For the HSPICE integration, the interface for setting up nodesets (.NODESET) and initial conditions (.IC) is opened on the Environment Console by Selecting Simulation > Convergence Aids. **Node Set**, provides an initial guess for nodes in any DC analysis or the initial condition calculation for the transient analysis and **Initial Condition** provides initial conditions for nodes in a transient analysis.

For detailed information solving convergence issues in HSPICE see the section [Diagnosing Convergence Problems](#) in the *HSPICE User Guide: Simulation and Analysis*.

Chapter 6: Running Simulations and Using Control Options

Simulation Menu Overview



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

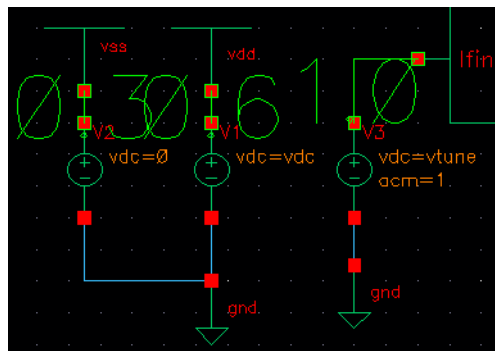
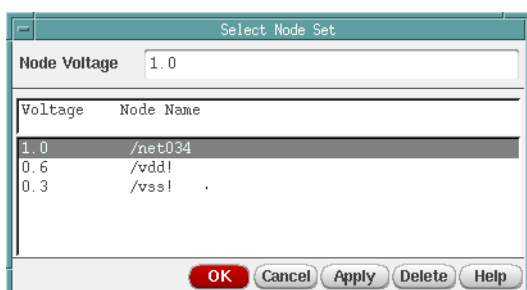
Figure 53 Convergence Aids Menu Options

Selecting Node Sets or Setting Initial Conditions

To set new voltages on specified nodes:

1. Type your required **Node Voltage** on the Select Node Set form.
2. On the schematic, select the node for the new voltage to auto-fill the **Voltage** and **Node Name** columns in the form.
3. Repeat Steps 1 and 2 until you have created a new set of node voltages in the form. An annotation of the voltage is printed on the schematic after each voltage change and node selection, as seen in [Figure 54 on page 171](#).

Note: For split nets, the UI labels only the driving cell.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 54 Annotated Schematic Voltages for Specified Nodes

- To set other nodes to the same voltage, select them, one at a time, on the schematic.
- To set other nodes to a different voltage, change the voltage, and select other nodes.
- OK** the form before you run or re-run your simulation. (Click **Cancel** to clear the form and dismiss it.)

You can select nodes at any level of the hierarchy. When you select a node at a lower level in the schematic, the node is highlighted, but the voltage value is annotated only on the higher-level.

While the select node commands are active, the system highlights selected nodes and labels them with the voltages you set. After you close the form, the UI removes the highlighting and the labels from the schematic. Re-opening a form with voltage entries restores the highlights and annotation.

Changing Node Set or Initial Conditions Voltages

To change the node set or initial condition node voltage settings:

- On either the Set Node Set or Set Initial Conditions Set, click on the node name to highlight it.
- Type the new voltage value, and click **Apply**.
- Click **OK** or **Cancel** when you are finished changing voltages.
- To release voltages, remove the values on the form and click **Apply**.

Chapter 6: Running Simulations and Using Control Options

Simulation Menu Overview

If the node voltage setting forms are closed during your session and you want to change or release voltage values, re-open the form by selecting Simulation > Convergence Aids > Node Set *or* Initial Condition.

For further information about the available convergence assistance through the UI, refer to the Cadence documentation.

Printing and Plotting Results

Describes HSPICE post-processing actions using the Result menu selections and introduces the HSPICE Plotting Assistant.

As analog/mixed signal designers, you spend considerable time examining the results of your simulations. Examination can take several forms: back-annotating results to the schematic, sending saved plot sets to the waveform viewer, generating reports of scalar outputs, node voltages and operating points, interactively plotting results, and making measurements of simulation data.

These sections introduce the HSPICE Plotting Assistant dialog and describe the other options on the Results menu:

- [Results Menu Overview](#)
- [HSPICE Plotting Assistant](#)
- [Results > Direct Plot](#)
- [Results > Print](#)
- [Results > Annotate](#)
- [Results > Save, Select, Delete](#)
- [Results > Printing/Plotting Options](#)

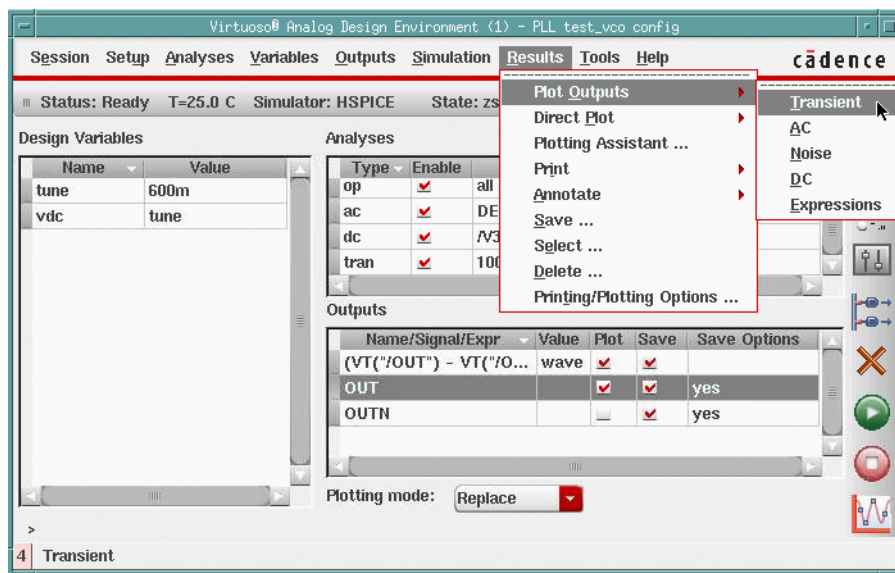
Results Menu Overview

The successful completion of a simulation activates all the menu selections available on the Environment Console > Results menu. The menu is active when simulation results are presently loaded with the following options:

Chapter 7: Printing and Plotting Results

Results Menu Overview

- Plot Outputs
- Direct Plot
- Plotting Assistant
- Print
- Annotate
- Save
- Select
- Delete
- Printing/Plotting Options



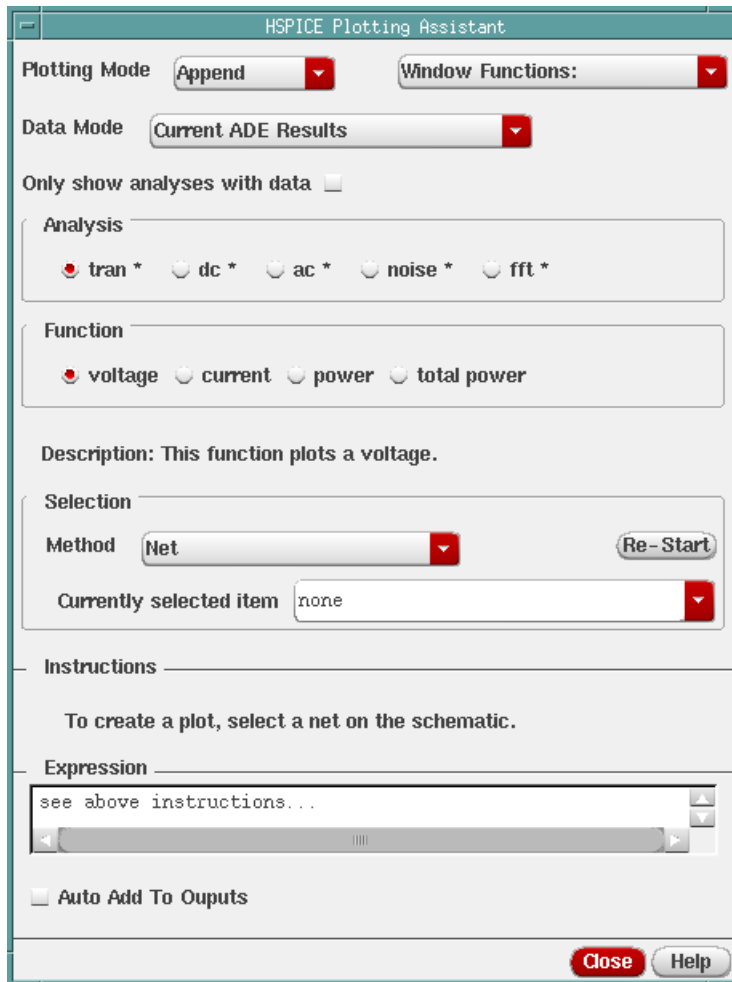
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 55 Results menu

HSPICE Plotting Assistant

The HSPICE Plotting Assistant is especially tailored for HSPICE simulator users. It is invoked by selecting Results > Plotting Assistant. This tool is used to facilitate the quick creation and plotting of the most commonly used waveform expressions depending on the selected analysis. This dialog changes dynamically depending on which analysis you select for post-processing. The Plotting Assistant is a UI customized to support the analyses that HSPICE

simulates. For example, the window for Transient Analysis is shown in [Figure 56 on page 175](#). The Plotting Assistant also acts as a cockpit for managing several plot-related tasks.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 56 Plotting Assistant for Transient Analysis

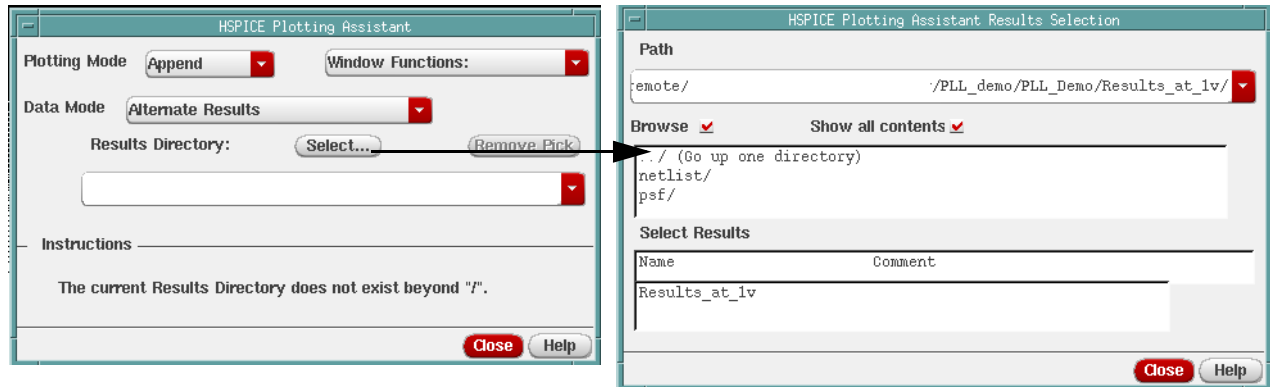
User Interface Controls

The UI controls for the Plotting Assistant allow you to customize the plotting results. The upper portion of the window provides plot management options.

Figure 57 Plotting Assistant Results Selection

Chapter 7: Printing and Plotting Results

Results Menu Overview



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

The **Plotting Mode** pulldown consists of these choices:

- **Append** — adds the new expression or waveform to the current waveform window
- **Replace**—replaces any current waveform with the new expression or waveform
- **New SubWin**— plots the new expression or waveform in a new sub window
- **New Win**—plots the expression or waveform in a new window
- **Off**—turns off the plotting mode function; this feature allows you to generate waveform expressions without creating a plot. This is typically used when the expression needs to be further defined within either the console **Outputs** section or the calculator.

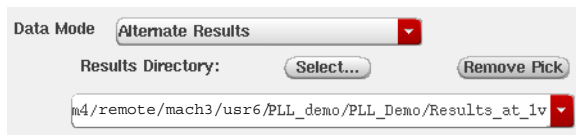
The **Window Functions** pulldown list provides controls to manage the various windows and plotting dialogs on-screen:

- **Schematic - Raise** brings the Schematic Editor to the top of the stack of windows.
- **Schematic - Clear probes** deselects all selections on the schematic design.
- **Waveform - Raise** brings the waveform window to the top of the stack of windows.
- **Waveform - Clear Subwindow** clears all waveforms from the current sub-window of the current waveform window.
- **Calculator - Raise** invokes the Environment built-in calculator for verifying or building a new expression.
- **ADE - Raise** brings the Environment Console to the top of the stack of windows.

The **Data Mode** dropdown allows you to select the simulation results on which to operate:

- **Current ADE Results** mode uses the data currently loaded for the current ADE session. This is the default mode.
- **Alternate Results** mode references data that is outside what is currently loaded for the current ADE session. In this mode the user is required to specify the specific results directory to reference.

The **Results Directory** field is where the user specifies the path to the desired results directory. In addition, any valid results directories previously declared for this field are added to the pulldown choices of this combo field. Clicking the **Remove Pick** button removes the alternate results path field value from both the field and its combo choices list and restores the path to the first found previous results specified.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

To facilitate declaring a proper results directory, a **Select** button is provided to open a browser tool for this purpose. Clicking this button opens the HSPICE Plotting Assistant Results Selection dialog to the current working path. The **Results Directory** field is populated and the form automatically updates according to the selected results data.

A legitimate results directory must have both a `netlist/` and a `psf/` directory included.

- **Current ADE Results and Reference Results** is the third **Data Mode** dropdown item. This mode is a combination of the other two modes. In this case, two waveforms are created where one references the current ADE results and the other the Reference Results. The waveforms generated from each data set are appended on the same graph window, to allow for convenient results comparison.

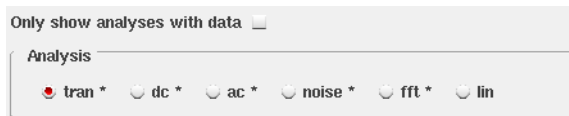
Plotting Analysis-Specific Data

The selections in the Analyses portion of the assistant represent all of the supported HSPICE analyses. As you select various analyses, the Plotting Assistant will update with a display that is specific to the current analysis.

Chapter 7: Printing and Plotting Results

Results Menu Overview

The Analyses that have an asterisk (*) appended to their name are the ones that currently have data in the selected results. Analyses without corresponding data are provided as valid picks so that you can use the Plotting Assistant to build expressions prior to running a simulation. A check box allows you to filter these analyses, to show only the collection with valid current results.



Common Form Sections Shared Across Functions

The following HSPICE Plotting Assistant labels and sections are generally shared by each of the analysis plotting forms. For any chosen plot function, a subset of these sections will be found.

- **Function** — this field allows the user to choose from a set of most commonly used plot function expressions. Each analysis has its own set of plot functions.
- **Description** — this field provides a synopsis of what the function plots.
- **Selection** — this section provides the user the ability to choose from several methods of selecting the signals on the schematic. The set of methods to choose from varies on a per-plot function basis.

In addition, the **Currently Selected Item(s)** field shows the currently selected signal set for the function. The choices pulldown for this combo field shows the five previous selection sets used on this function. This combo pulldown allows the user to quickly redo previous plots without having to go to the schematic for re-selection. This section also warns when selected schematic items are not found in the simulation data, thus informing the user when a plot cannot be produced.

Because the Plotting Assistant shares the same core selection engine with the other plotting utilities in Direct Plot, it is possible for the PA to get out of sync with the selection engine when multiple plotting utilities are used concurrently. When the Plotting Assistant does not appear to respond to schematic selections, the **Re-Start** button is provided to restore the selection synchronization of the PA.

- **Plot Type** — this field allows the user to specify the format in which a plots axis are displayed.
- **Modifier** — this field allows the user to select a modifier function that wraps the generated waveform expression.
- **Instructions** — this label helps guide the user through the necessary steps needed to create a successful waveform expression. This label will sequentially explain selection requirements, form field value requirements, and the status of the required simulation data.
- **Expression** — this section displays the actual SKILL data access expression used to create the waveform being plotted. When an expression evaluates to an error, the error is reported just below the expression field (See [Figure 62 on page 184](#), [FFT Analysis Output](#) for an example).
- **Add To Outputs** — clicking this button sends the current waveform expression(s) to the **Outputs** table on the Environment Console. When the **Auto** lock boolean is activated, then each successively created plot will also result in its waveform expression being set to the **Outputs** pane.
- **Add to Calculator** — clicking this button sends the current waveform expression(s) to the calculator. This is most useful when the user wishes to use the calculator to build upon the basic waveform expression created by the Plotting Assistant.
- **Plot/Replot** — these mutually exclusive buttons are used for two main purposes. Use **Plot** when a plot is not dependent on performing a selection in the schematic. Use **Replot** to quickly repeat a plot using the previous selection set.

Analysis-Specific Plot Functions

Plotting Transient Analysis Signals

When the Transient Analysis is selected in the Plotting Assistant, the Plotting Assistant allows for four plotting **Functions**: **voltage**, **current**, **power**, and **total power**. Selecting a particular function changes the rest of the dialog as shown in the following figure. Where appropriate, the **Method** cyclic field is expanded in [Figure 58 on page 180](#) to show the currently available selection methods for each function.

Chapter 7: Printing and Plotting Results

Results Menu Overview

Function: Voltage

Function

☒ voltage ☐ current ☐ power ☐ total power

Description: This function plots a voltage.

Selection

Method **Net** Re-Start

Currently **Differential Nets**

Instance With Two Terminals

Instructions

To create a plot, select a net on the schematic.

Expression

see above instructions...

☐ Auto Add To Outputs

Function: Current

Function

☐ voltage ☒ current ☐ power ☐ total power

Description: This function plots a current.

Selection

Method **Terminal** Re-Start

Currently **Differential Terminals**

Instance 'PLUS' Terminal

Instructions

To create a plot, select a terminal on the schematic.

Expression

see above instructions...

☐ Auto Add To Outputs

Function: Power

Function

☐ voltage ☐ current ☒ power ☐ total power

Description: This function plots power.

Selection

Method **Terminal** Re-Start

Currently **Terminal**

Instance With Two Terminals

Terminal and Voltage Reference Terminal

Terminal and Voltage Reference Net

Power Dissipated By An Instance

Instructions

To create a plot, select a terminal on the schematic.

Expression

see above instructions...

☐ Auto Add To Outputs

Function: Total Power

Function

☐ voltage ☐ current ☐ power ☒ total power

Description: This function plots total power.

Instructions

To create a plot, press "Plot".

Expression

getData("power()" ?result "tran")

☐ Auto Add To Outputs Add To Calculator Plot

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 58 Dynamic Plotting Assistant Changes Based on Function Selected

UI Changes Based on Selected Function

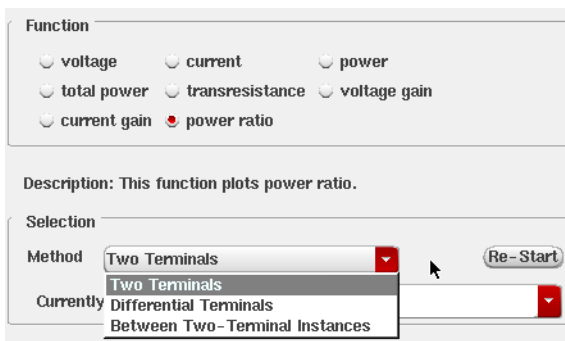
The four functions available for a transient analysis ([Figure 58](#)) provide a variety of selection methods for plotting.

- Voltage Selection Methods:
 - Net

- Differential Nets
- Instance with Two Terminals
- Current Selection Methods:
 - Terminal
 - Differential Terminals
 - Instance “PLUS” Terminal
- Power Selection Methods
 - Terminal
 - Instance With Two Terminals
 - Terminal and Voltage Reference Terminal
 - Terminal and Voltage Reference Net
 - Power Dissipated by an Instance
- Total Power: Since the total power function does not require schematic canvas input, the plot is created by clicking the **Plot** button.

Plotting DC Analysis Signals

The section of **Function** buttons for a DC analysis increases the options with **transresistance**, **voltage gain**, **current gain**, and **power ratio**. See [Figure 58](#) on page 180.



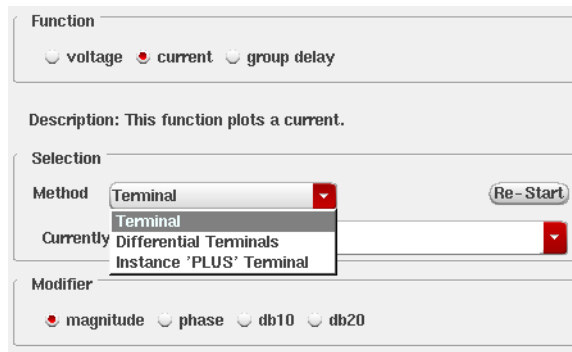
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 59 DC Analysis Signal Function

The **Selection Method** presents alternative modes in a pulldown list similar to the other analyses.

Plotting AC Analysis Signals

The AC Analysis functions include **voltage**, **current**, and **group delay**.



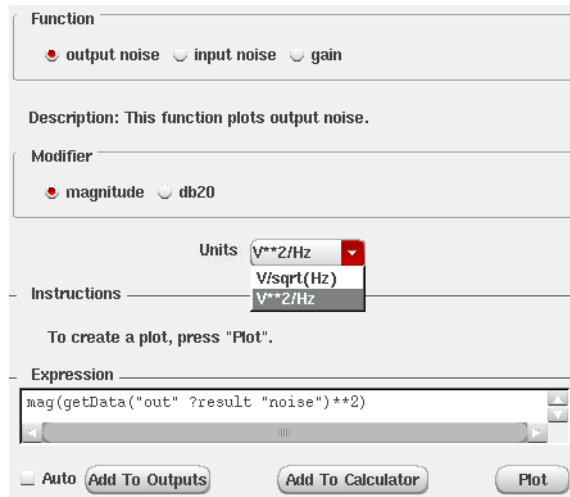
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 60 AC Analysis for Current

As with all functions, multiple selection methods are available to further define what is to be plotted. For example, with current selected, three choices are available through the pulldown list of selection methods. Four modifier choices are available.

Plotting Noise Analysis Signals

The Noise Analysis functions include **output noise**, **input noise**, and **gain**. See [Figure 61 on page 183](#).



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 61 Noise Analysis for Output Noise

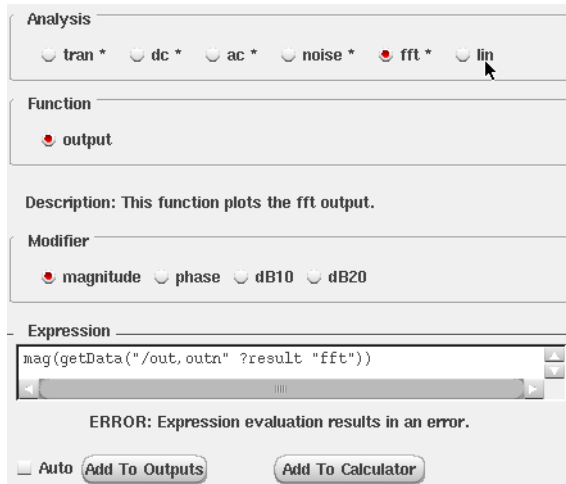
With **output noise** or **input noise** selected, the **Modifier** choices of **magnitude** and **db20** are available as radio buttons. The **Units** pulldown list offers the option of plotting equivalent input noise, equivalent output noise, squared input noise, and squared output noise. These are the same quantities that are available from the Direct Plot cascade menu.

Plotting FFT Analysis Signals

Post-processing of FFT signals offers an *output* function only. In this case it is the output signal as specified in the FFT analysis dialog. See [Figure 62 on page 184](#).

Chapter 7: Printing and Plotting Results

Results Menu Overview



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 62 FFT Analysis Output

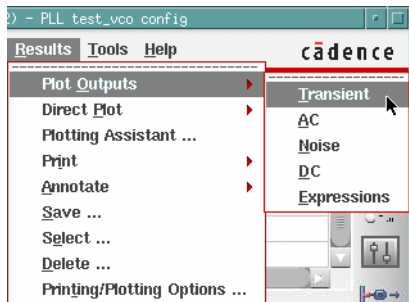
Four **Modifier** choices are available.

For the sake of example, an error message is shown below the **Expression** field.

Depending on the function selected, the UI provides **Plot Type** and **Modifier** choices appropriate to the selected function.

Results > Plot Outputs

The first Results menu option is Plot Outputs. Each individual submenu pick for this menu is active depending on unique criterion as described below.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

The following picks are available using the Plot Outputs sub-menus

- Transient Plots: the transient response for the transient signal(s), etc. For this menu item to be active, transient data must be present and some signals on the Environment Console Outputs table must have both Plot and Save options activated.
- AC Plots: the AC response for the AC signal(s). For this menu item to be active, ac data must be present and some signals on the Environment Console Outputs table must have both Plot and Save options activated.
- DC Plots: the DC sweep response for the DC signal(s). For this menu item to be active, dc data must be present and some signals on the Environment Console Outputs table must have both Plot and Save options activated.
- Noise Plots: the squared noise voltage for the noise signals. For this menu item to be active, noise data must be present.
- Expressions: Plots the waveforms for expressions you define in the Setting Outputs form (see [Chapter 5, Saving-Plotting Outputs](#)).

Results > Direct Plot

You can plot common waveforms using the Direct Plot menu picks [Figure 63 on page 185](#). With these commands, you can plot basic signals without the need for a primary plotting dialog (like the calculator or Plotting Assistant). Note that these commands will not add signals to the Outputs section of the Environment Console.

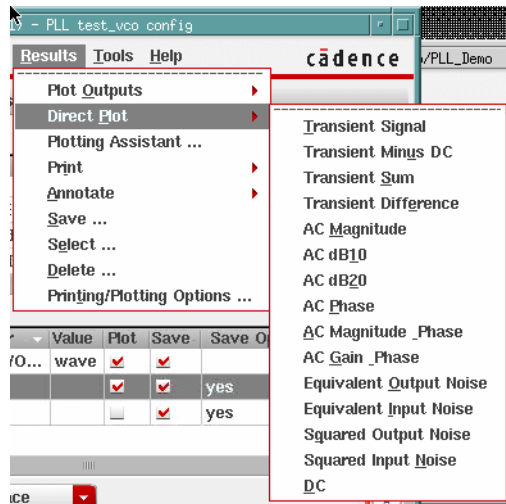
To use the Direct Plot menu options:

1. Choose a menu option from the Results > Direct Plot menu [Figure 63](#). (For those options where a form is required, it is displayed.) The waveform window opens. If a waveform window was already open, it becomes active. This behavior depends on the value of the Plotting Mode cyclic field below the **Outputs** list on the Environment Console. Plotting may create a new waveform window, a sub-window, or append/replace an existing window's contents.
2. Check the status line in the Schematic window for a prompt, which advises what to select in the schematic.

Figure 63 Direct Plot options

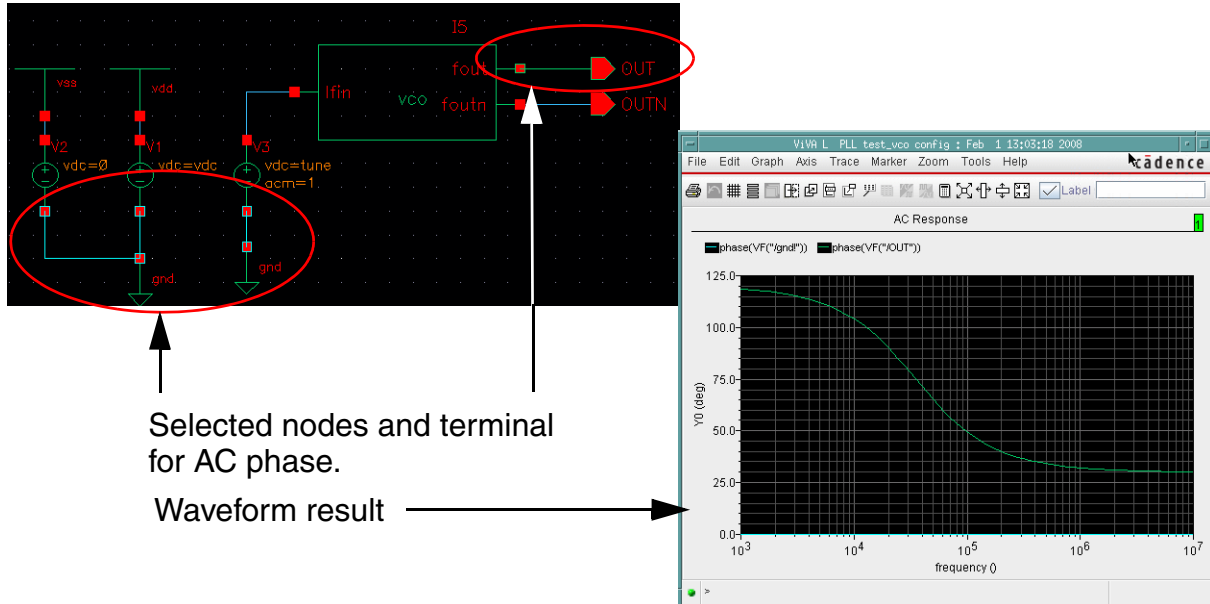
Chapter 7: Printing and Plotting Results

Results Menu Overview



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

For example, using the state shown in the Environment Console in [Figure 55 on page 174](#), if you select AC Phase on the Direct Plot menu the message on the Schematic Editor status line instructs you to select nodes and terminals and then press the Esc key. A waveform such as the one below is generated based on selecting the ground nodes and the OUT terminal.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

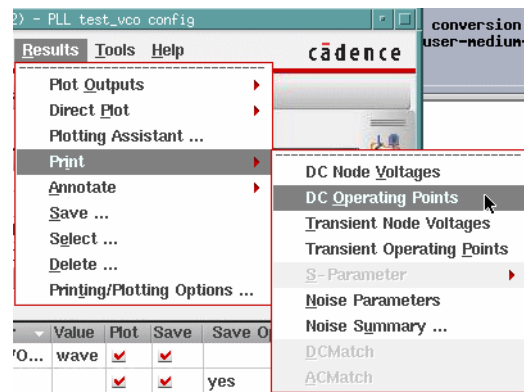
Figure 64 Schematic nodes selected and waveform of AC phase

The Direct Plot options and the curves they will plot are:

- Transient Signal —Transient voltage or current waveforms
- Transient Minus DC —Transient voltage or current waveforms without the DC offset
- Transient Sum — Multiple signals added together and plotted (the status line message prompts you for the signals)
- Transient Difference—Two signals subtracted (sig1- sig2) and plotted (the status line message prompts you for two signals)
- AC Magnitude—AC voltage or current gain waveform
- AC db10—The magnitude on a decibel scale
- AC db20 The magnitude of selected signals on a decibel scale $20\log(V1)$
- AC Phase—AC voltage or current phase waveform
- AC Magnitude & Phase—The db20 gain and phase of selected signals are plotted together
- AC Gain & Phase—The differences between two magnitudes and two phases; you are prompted for two signals, V1 and V2, and the expression $20\log(V2)-20\log(V1)$ is plotted, which is equivalent to $20\log(V2/V1)$
- Equivalent Output Noise—Output noise voltage or current signals selected in the analysis form; the curve plots automatically and does not require selection
- Equivalent Input Noise—Input noise waveform, which is the equivalent output noise divided by the gain of the circuit; the curve plots automatically and does not require selection
- Squared Output Noise—Squared output noise voltage or current signals selected in the analysis form; the curve plots automatically and does not require selection
- Squared Input Noise—Input noise waveform, which is the equivalent output noise divided by the gain of the circuit squared; the curve plots automatically and does not require selection
- Noise Figure—Noise figure of selected signals according to the input, output, and source resistance; the curve plots automatically and does not require selection
- DC—DC sweep voltage or current waveform

Results > Print

The Results > Print menu shows available print options for those analyses that have been simulated and run. The Schematic Editor must be displayed on screen for this operation. The Print menu also provides access to scalar results such as voltages and operating points; the schematic editor need not be displayed for this to work.

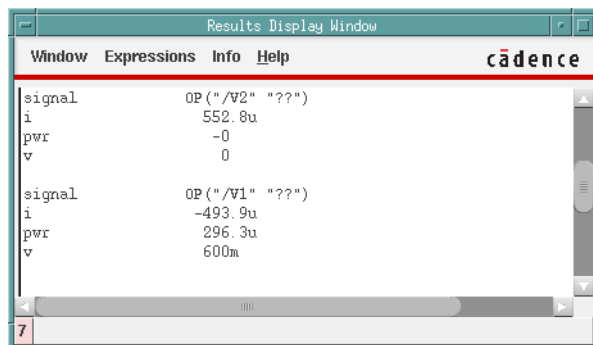


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

An active menu option implies that a data file has been generated. Click an available option for the results to be displayed. Then select the nodes for which you want the data displayed. For example:

To print data for DC Operating points:

1. Select DC Operating Points on the Print menu
2. On the Schematic Editor, select any number of appropriate nodes to populate the display window with the nodes' DC operating point data.

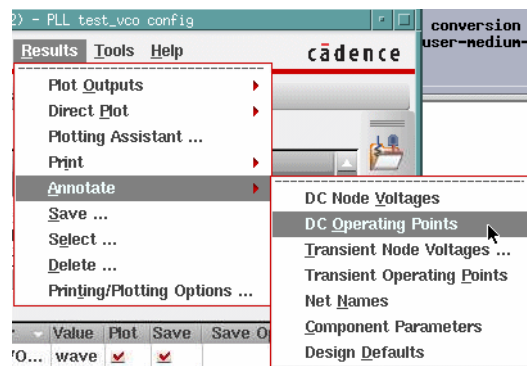


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

For further information about the Print menu, refer to the Cadence documentation.

Results > Annotate

Back-annotation is the process of annotating values to the schematic canvas. The values could be node voltages, terminal currents, or device operating points that are defined per primitive by each simulator. This annotation is done automatically when a simulation completes, and is done through the entire hierarchy. These annotations are not database changes, allowing them to be done on read-only designs as well as those that are editable, and allowing multiple users to annotate the same schematic. Annotations are visible on hardcopy plots as well as on images made by exporting plots to other formats. This basic functionality exists across all simulators in the Environment. You can further annotate your results by selecting the **Results > Annotate** menu option.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Annotation of operating points gives you information as to potential device malfunctions in the context of the schematic. The typical Environment device library has three cdsParam() and several cdsTerm() “slots” for back-annotation, although this can be changed by anyone with write access to the placed device masters (usually the symbol cellview):



Figure 65 Annotated Symbol cellview

Back-annotation in the HSPICE interface can be performed with the **Results > Annotate > DC Operating Points** for time=0 results, or **Transient Operating Points** for final transient timepoint results.

Once annotated with operating points, a device is displayed as in [Figure 66 on page 191](#):

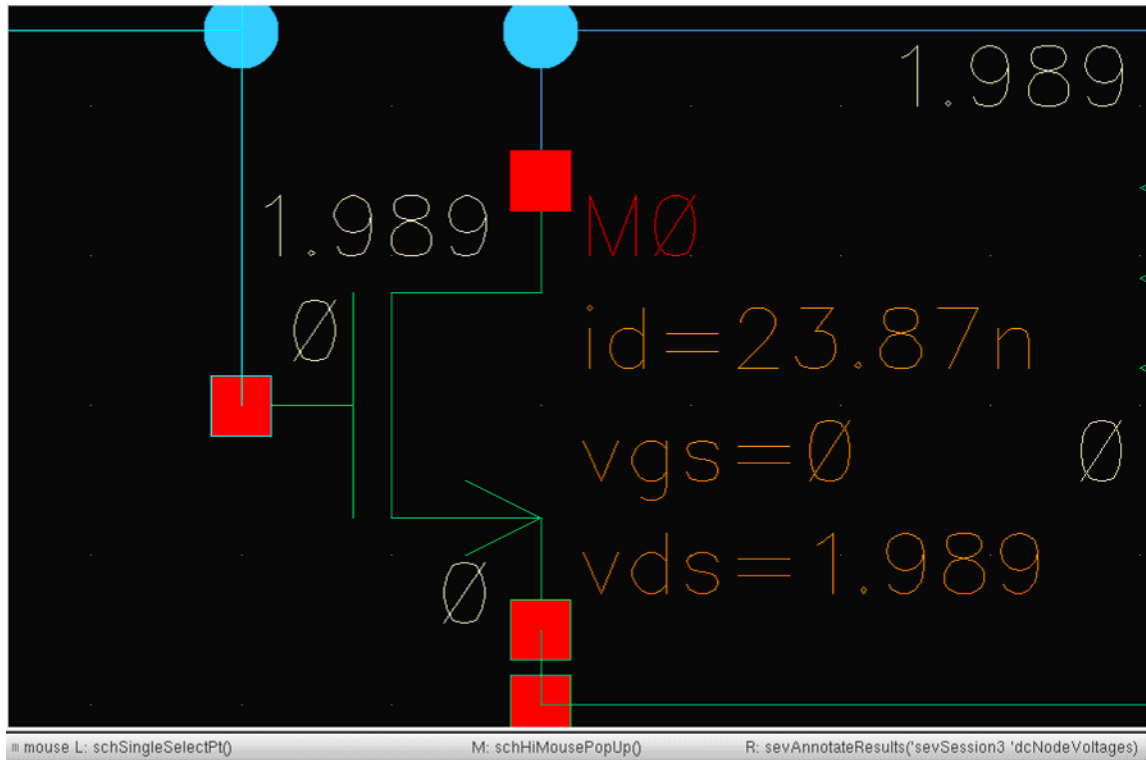


Figure 66 Operating Points Annotation

The three operating points to display (in the example above i_d , v_{gs} , and v_{ds}) can be defined on the cell CDF, and can be changed on a per-instance/master/library level in the Component Display dialog. The functionality of that dialog is not impacted by the HSPICE integration.

Annotating/Printing DC or Transient Operating Points on a Macro-Model-Based Device Instance

Because there are data name differences between HSPICE outputs in a PSF file and annotation function calls in ADE, users need to provide a name mapping function to avoid annotate function failures in HSPICE-ADE. The HSPICE integration to ADE provides the 'name-mapping' function based on user inputs to transfer the data correctly from HSPICE PSF output file to the instance on schematic.

Background

HSPICE treats the macro-model-based device instance as subckt instance and generates operating point information for one subckt level. For example:

Chapter 7: Printing and Plotting Results

Results Menu Overview

assume that one macro-model device instance is named 'm0' on the schematic. Its macro model name is `nch_mac` which is defined by a `nch_mac` cell in the PDK library. If you check the model file provided by the vendor, you might find there is one real definition of the macro model `nch_mac` as one subckt.

```
.subckt nch_mac d g s b
.param xxx=xxx
...
nch1 d g s b nch w='w' l='l' m='mf' ...
.model nch.1 NMOS Level=54 Wmin=xxx Wmax=xxx Lmin=xxx Lmax=xxx ...
.model nch.2 NMOS Level=54 Wmin=xxx Wmax=xxx Lmin=xxx Lmax=xxx ...
...
.ends
```

Note that there is one real device instance `nch1` defined in this macro model. Then HSPICE outputs the operating point information based on the real device instance `nch1` as follows:

```
'Xm0.nch1:Vgs', 'Xm0.nch1:Vth', 'Xm0.nch1:gds' ...
```

While you may have expected it to be annotated on schematic as:

```
'm0:Vgs', 'm0:Vth', 'm0:gds' ...
```

So there is data name difference here between HSPICE outputs in the PSF file and annotation function calls in ADE.

Use either of the following methods to create a mapping to handle this issue.

Using the Edit Properties Form

This method is designed to annotate a current selected instance ('m0') only on a schematic for DC or transient operating points.

Use the Edit Object Properties form to add an instance property of `m0` named `mapDeviceName` as a **string** type, then fill in the actual device instance name `nch1` as its **Value** as shown in [Figure 67 on page 193](#). Access the Edit Properties form using the Schematic Editor menu: **Edit > Properties > Objects**.

The screenshot shows the 'Edit Object Properties' dialog box. At the top, there are two dropdown menus for 'Apply To' (set to 'only current') and 'instance' (set to 'instance'). Below these is a 'Show' button and a sub-dialog titled 'Add Property'. The 'Add Property' dialog has fields for 'Name' (mapDeviceName), 'Type' (string), 'Value' (nch1), and 'Choices' (empty). It also has 'OK', 'Cancel', 'Apply', and 'Help' buttons. Below the 'Add Property' dialog, there is an 'Instance Name' field with 'm0' and a 'Display' dropdown set to 'off'. At the bottom, there are 'Add', 'Delete', and 'Modify' buttons. Below these buttons is a table with three columns: 'CDF Parameter', 'Value', and 'Display'.

CDF Parameter	Value	Display
Model name	nch_mac	off
Width	1u M	off
Length	100.0n M	off
Source diffusion area	1p	off
Drain diffusion area	1p	off
Source diffusion periphery	3u M	off
Drain diffusion periphery	3u M	off

Figure 67 Add mapDeviceName as an object property

Using the Edit Component CDF Form

This method is designed for a user to annotate the DC or transient operating points on all of the device instances based on one cell. Also, if the modification is based on CDF type 'Base', then the annotation will always work. (See the caution below.) This method to map the device name adds a CDF parameter to the defined cell `nch1_mac` in the PDK through the Edit Component CDF form and the Add Parameter CDF form. The added CDF parameter is named `mapDeviceName` as a **string** type and you fill in the actual device instance name `nch1` as its **defValue** ([Figure 68 on page 194](#)). Access this form by going to Virtuoso console window (icms) and select **tools > CDF > Edit**.

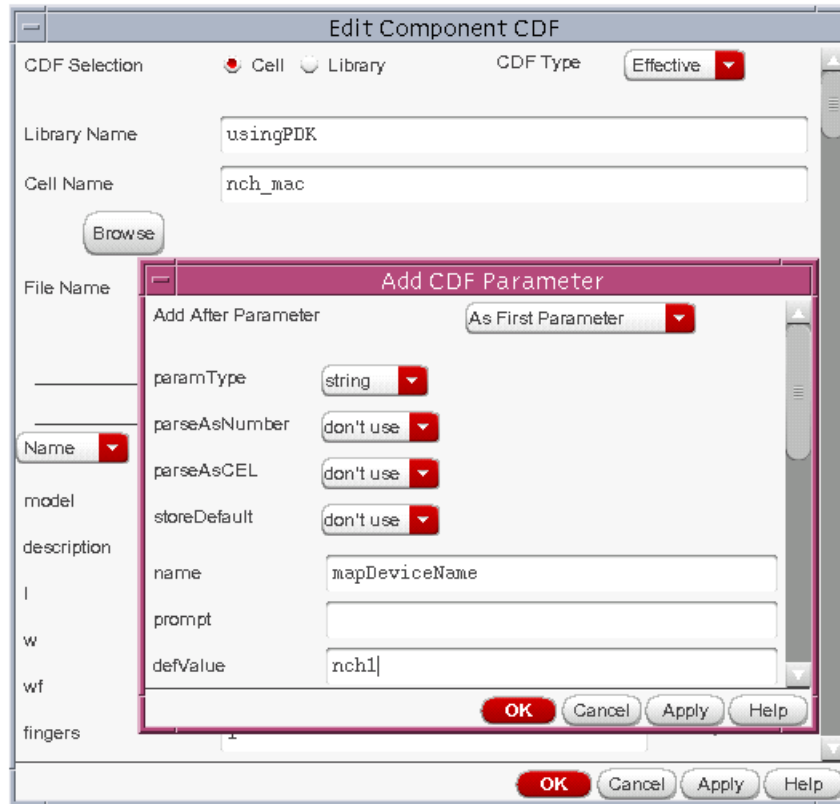


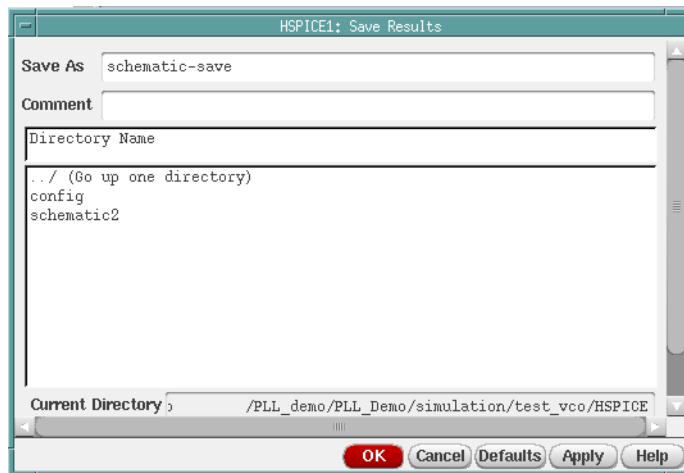
Figure 68 Add mapDeviceName as a CDF parameter

Note: Be careful regarding the current **CDF Type** selection. The **Base** CDF type means that you have *write* permission to change and save the added parameter to the cell database in the PDK (which could be used by any others). The **User** or **Effective** CDF type means that the added parameter is stored in memory and is discarded after the current Virtuoso session. You have to manually save the changes to another CDF file for next usage.

Results > Save, Select, Delete

The **Save**, **Select**, and **Delete** options on the **Results** menu all help you manage your results directories.

Selecting Results > Save launches the Save Results form.



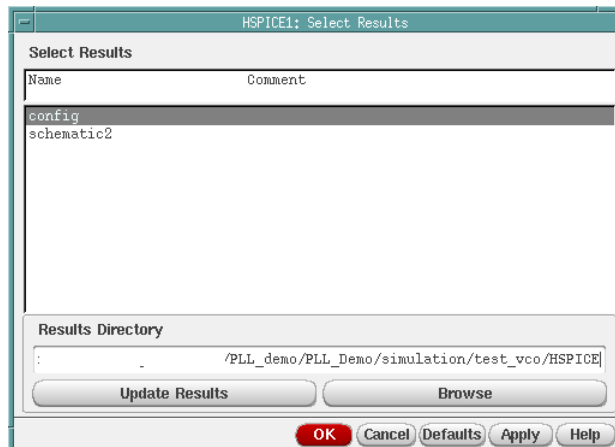
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Save Results

The Save Results form shows the results of the latest simulation with the default name `schematic-save`. To save a results directory under a different name, manually change the name and add a comment if you wish, and navigate to a directory of your choice, then click **OK**. For more information regarding the Save Results form, see the Cadence documentation.

Select Results

Selecting Results > Select launches the Select Results form. You can use this form to update and restore results.



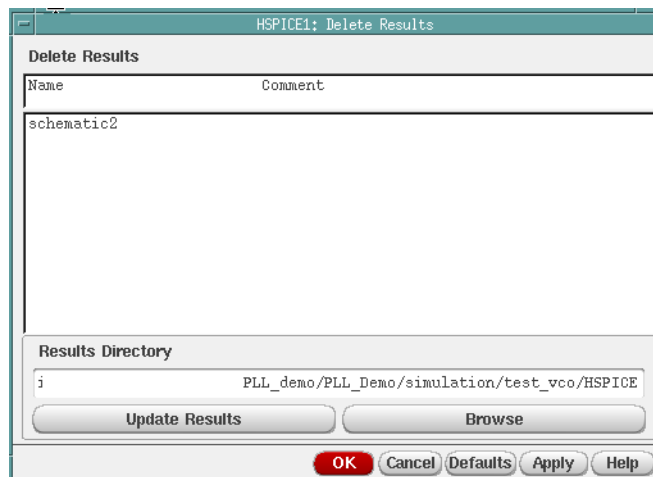
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Check that the Results Directory field displays the correct information. You can select results in a different location by clicking the **Browse** button and navigating to the proper directory.

Note: The proper directory is two levels up from the `psf/` directory. For example, if your results directory is: `simulation/test_vco/HSPICE/schematic/psf/`, use the browser to navigate to `simulation/test_vco/HSPICE/`.

Delete Results

You can clear your existing results by invoking the Delete Results form. The **Results Directory** field lists the default directory in which results are saved. To delete a set of simulation results, select the name of the results in the list field and click **OK**. If the results you want to delete are in a different location, click the **Browse** button to open the Unix Browser and navigate to the target directory, highlight and click **OK**. Clicking the Update Results button



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Results > Printing/Plotting Options

Selecting Results > Printing/Plotting Options launches the Setting Plot Options form. You can set preferences according to categories presented in the form for the current session, including directions of when plots are printed, auto plotting, annotation options, waveform window preferences for font-size, height and width, and X-Y location offsets.

Setting Plotting Options -- Virtuoso® Analog Design Environment (2)

Print After ☒ Each Selection
☐ All Selections Are Made

Auto Plot Outputs After Simulation ☒

Direct Plots Done After ☐ Each Selection
☒ All Selections Are Made

Annotations ☒ Design Name ☒ Simulation Date
☐ Temperature ☐ Design Variables
☐ Scalar Outputs

Waveform Window

Font Size 11

Width 564

Height 500

X Location 577

Y Location 345

OK Cancel Defaults Apply Help

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 69 Setting Plot Options form

For detailed discussion of this form, see the documentation available directly from Cadence.

Chapter 7: Printing and Plotting Results

Results Menu Overview

Monte Carlo in the HSPICE Integration

Describes the Monte Carlo Analysis capabilities available through the HSPICE integration interface.

The Monte Carlo Analysis form in the HSPICE Interface to the Cadence™ Virtuoso® Analog Design Environment allows easy access to HSPICE's advanced analysis capabilities without additional license control.

Note: The HSPICE Variation Block syntax is not supported in the current version.

For paths to Monte Carlo analysis demonstration cases available with this release, see [HSPICE Integration to ADE Demonstration Examples](#).

For information on using HSPICE distributed processing for Monte Carlo simulations, see [HSPICE Monte Carlo Distributed Simulation](#) in [Chapter 15, Distributed Mode—Monte Carlo/Corner Analyses](#).

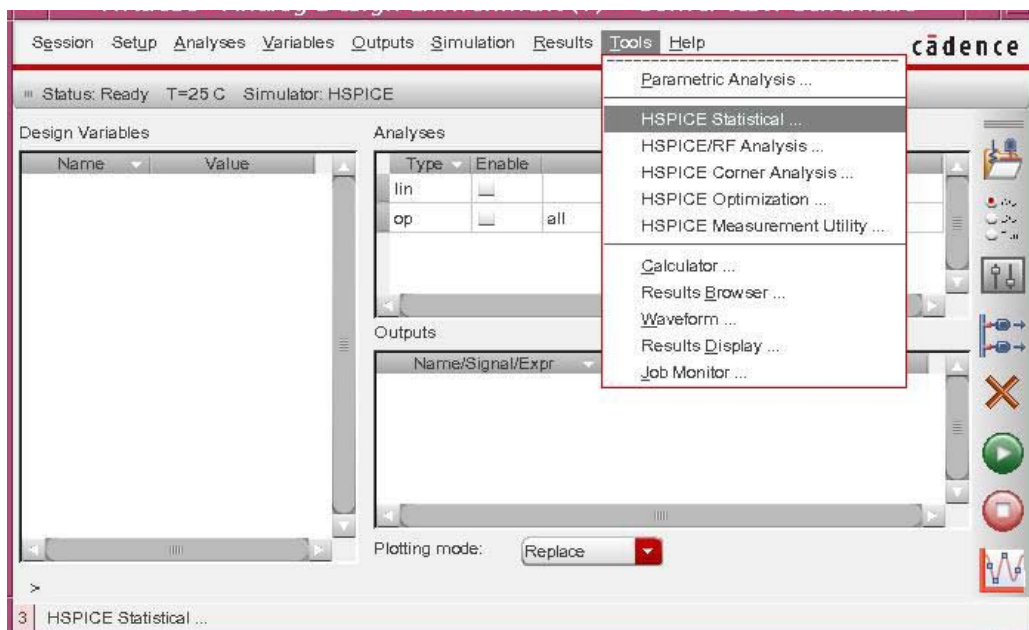
The following topics are discussed in these sections:

- [Accessing and Setting Up Monte Carlo Analysis](#)
- [Configuring the Setup Tab to Run a Monte Carlo Simulation](#)
- [Using the Options Tab of the HSPICE Monte Carlo Analysis Form](#)
- [Setting up Measurements, Post-Plotting, and Analyzing the Simulation Results Using the Outputs Tab](#)
- [Other Features Supported in the Monte Carlo Analysis Form](#)

Accessing and Setting Up Monte Carlo Analysis

Before running a Monte Carlo Analysis, you need to set up an analysis such as DC, Tran, or AC as you would normally (see [Chapter 4, Analysis Setup and Design Variables](#)).

To invoke the Monte Carlo Analysis form, select the **HSPICE Statistical** option under the Tools menu of the Environment Console to open the Monte Carlo Analysis form.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 70 HSPICE Statistical option on Environment Console Tools menu

Configuring the Setup Tab to Run a Monte Carlo Simulation

In the **Setup** tab of HSPICE Monte Carlo Analysis form:

1. Select the corresponding analysis (**DC**, **AC**, or **Tran**) to do a Monte Carlo analysis.

Chapter 8: Monte Carlo in the HSPICE Integration
Configuring the Setup Tab to Run a Monte Carlo Simulation

2. Enter the number of sweeps (**Number of Run**) to specify the Monte Carlo sweeps.
3. Specify the starting sweep number in the **Start Run** field.
4. Click the **List** check box to enable the input field to type in the effective points in the simulation.
5. Click the **Enabled** check box to enable the Monte Carlo for the selected analysis.

HSPICE Monte Carlo Analysis

File Simulation Plot Help

cadence

Setup Options Outputs

Analysis ☒ DC ☐ AC ☐ Tran

Number of Run

Start Run

☐ List

Enabled ☐

Please select a variation mode.

Variation Mode self-defined param

Parameters

Name Cellview Variables

Param Type AGAUSS

Nomi_Val

Abs_Val

Sigma

Multiplier

Design Variables

Name	Value
------	-------

4

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 71 Setup Tab of the Monte Carlo Analysis Form

6. For **Variation Mode**, select either the **self-defined param** for variation parameters used in the circuit, or the **statistical model** to do variations. To do variation on the self-defined parameter, use the variation parameters according to the HSPICE-supported variation functions/arguments.

Function	Description
UNIF	Uniform distribution function by using relative variation.
AUNIF	Uniform distribution function by using absolute variation.
GAUSS	Gaussian distribution function by using relative variation.
AGAUSS	Gaussian distribution function by using absolute variation
LIMIT	Random-limit distribution function by using absolute variation. Adds +/- <i>abs_variation</i> to <i>nominal_val</i> based on whether the random outcome of a -1 to 1 distribution is greater than or less than 0.
Nomi_val	Nominal value in Monte Carlo analysis and default value in all other analyses.
Abs_val	AUNIF and AGAUSS vary the nominal_val by +/- <i>abs_variation</i> .
Sigma	Specifies <i>abs_variation</i> or <i>rel_variation</i> at the <i>sigma</i> level. For example, if <i>sigma</i> =3, then the standard deviation is <i>abs_variation</i> divided by 3.
Multiplier	If you do not specify a multiplier, the default is 1. HSPICE recalculates many times and saves the largest deviation. The resulting parameter value might be greater than or less than <i>nominal_val</i> . The resulting distribution is bimodal.

7. The Design Variables controls (**Add**, **Change**, **Delete**, **Clear**) function as described in Chapter 4, [Design Variables](#).

Note: To set up measurements see [Setting up Measurements, Post-Plotting, and Analyzing the Simulation Results Using the Outputs Tab](#).

Setting Up Multiprocessing of Monte Carlo Runs

To set up a multiprocessing simulation for Monte Carlo, select **Simulation > MP simulation** in the HSPICE Monte Carlo Analysis window (see [Figure 72 on](#)

[page 203](#). You can set up the multiprocessing count according to the instructions in the form. For details on HSPICE multiprocessing, see [Running Multithread/Multiprocess HSPICE Simulations](#) in the *HSPICE User Guide: Simulation and Analysis*. The count selected in the window is in effect only for Monte Carlo session. See also, [HSPICE Monte Carlo Distributed Simulation on page 323](#).

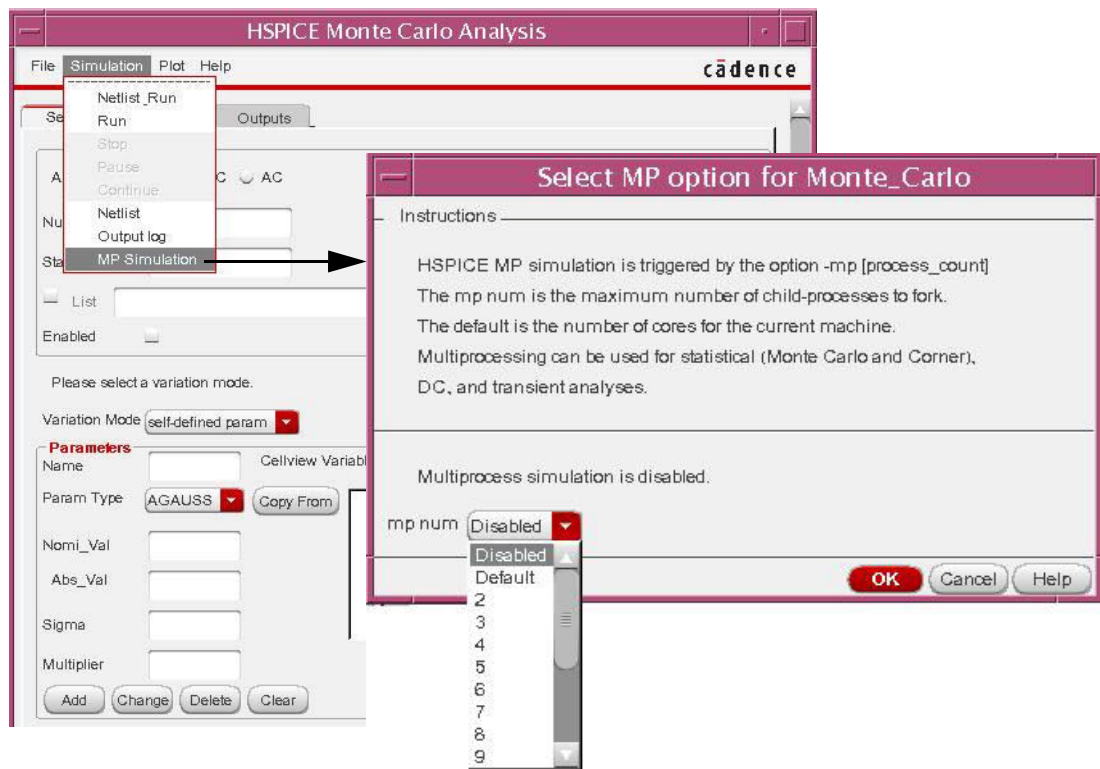


Figure 72 Multiprocessing for Monte Carlo

Using the Options Tab of the HSPICE Monte Carlo Analysis Form

Use the **Options** tab the same way you would use any simulation options provided by HSPICE as described in [Chapter 6, Running Simulations and Using Control Options](#).

Chapter 8: Monte Carlo in the HSPICE Integration

Using the Options Tab of the HSPICE Monte Carlo Analysis Form

The screenshot shows the 'HSPICE Monte Carlo Analysis' dialog box with the 'Options' tab selected. The dialog has a menu bar (File, Simulation, Plot, Help) and the Cadence logo. Below the tabs, there is a table with columns 'Name', 'Value', and 'Default status'. The table contains the following options:

Name	Value	Default status
MCBRIEF	0	Default
MODMONTE	<input type="checkbox"/>	Cdsenv
MONTECON	<input checked="" type="checkbox"/>	Default
RANGEN	0	Default
SEED	specified	Cdsenv
VALUE	1	Default

Below the table, there is a message: 'Please click "Apply" button to save your option settings.' and an 'Apply' button.

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 73 Options tab for HSPICE Monte Carlo Analysis

The available options for Monte Carlo are:

- MCBRIEF, which controls how HSPICE outputs Monte Carlo parameters.
 - MCBRIEF=0: Outputs all Monte Carlo parameters
 - MCBRIEF=1: Suppresses the MC parameters in the *.mt# and *.lis files.
 - MCBRIEF=2: Outputs the Monte Carlo parameters into a .lis file only.
 - MCBRIEF=3: Outputs the Monte Carlo parameters into the measure files only.
- MODMONTE, controls how random values are assigned to parameters with Monte Carlo definitions.

- If MODMONTE=1: within a single simulation run, each device that shares the same model card and is in the same Monte Carlo index receives a different random value for parameters that have a Monte Carlo definition.
- If MODMONTE=0: within a single simulation run, each device that shares the same model card and is in the same Monte Carlo index receives the same random value for its parameters that have a Monte Carlo definition.
- MONTECON, if check box is on (default), continues a Monte Carlo analysis by retrieving the next random value, even if non-convergence occurs.
- RANDGEN, specifies the number of the random generator in Monte Carlo analysis.
- SEED, starting seed number for the random generator; minimum value is 1; the maximum value of is 259200.

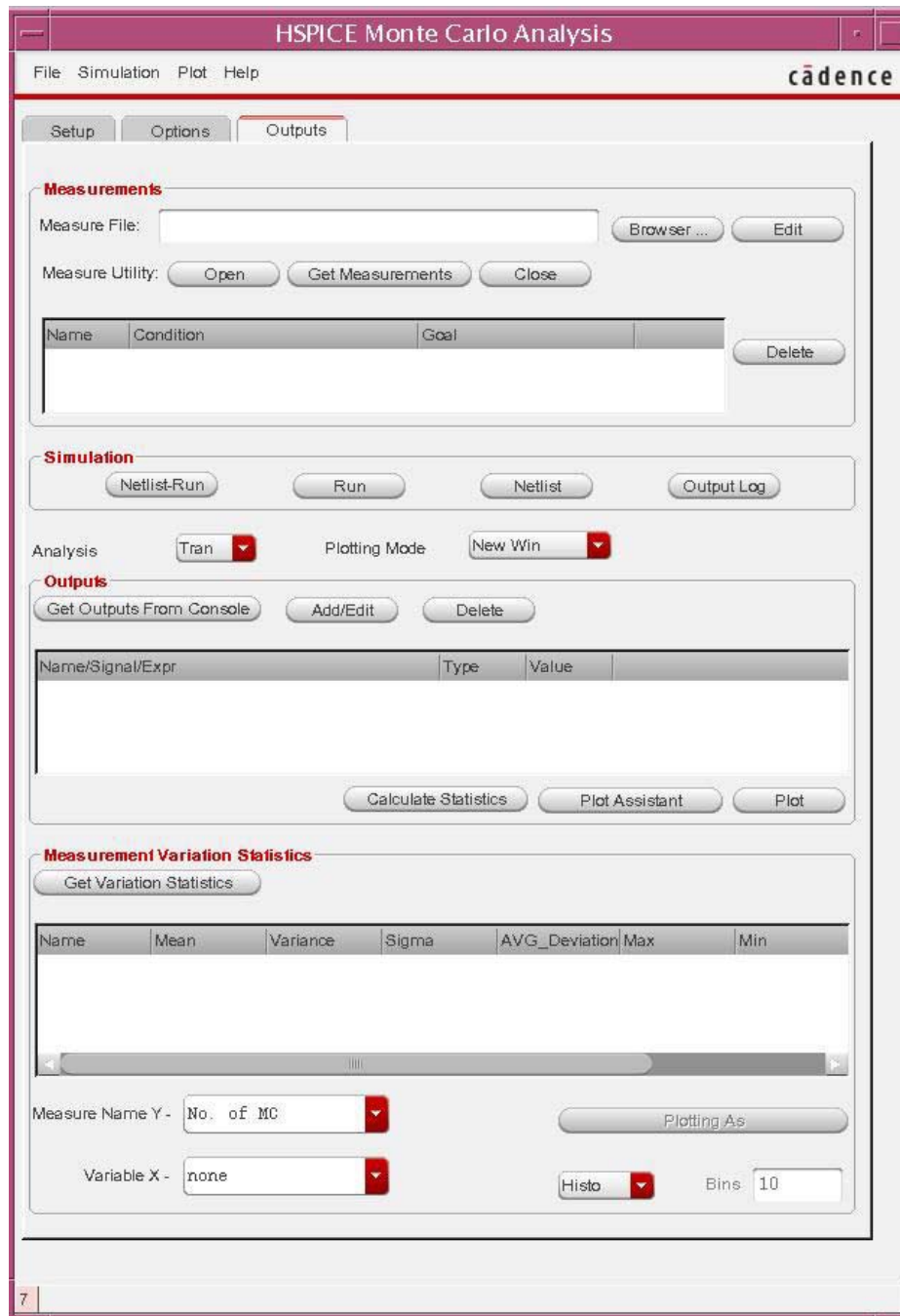
Setting up Measurements, Post-Plotting, and Analyzing the Simulation Results Using the Outputs Tab

The **Outputs** tab ([Figure 74 on page 206](#)) enables you to:

- Use HSPICE measurement statements to measure specific circuit attributes, manually edit the measure statements, and save as a single file for Monte Carlo analysis as an input file. You can use the HSPICE Measurement Utility to add, edit, or delete measurement variables.
- Run a simulation
- Select a type of analysis
- Specify a plotting mode
- Get, edit, or delete signal or expression outputs from the Environment Console
- Invoke the HSPICE Plotting Assistant window or do a Direct Plot
- Get and view measurement variation statistics and plot them as curves, scatter plots, or histograms.

Chapter 8: Monte Carlo in the HSPICE Integration

Setting up Measurements, Post-Plotting, and Analyzing the Simulation Results Using the Outputs Tab



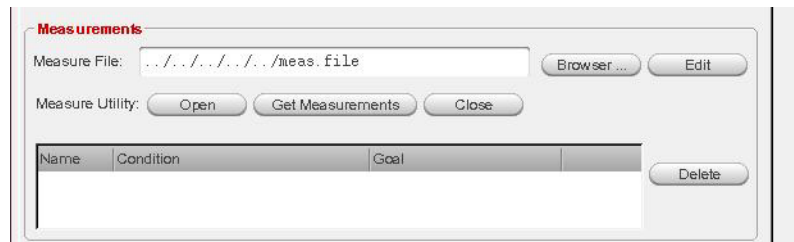
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 74 Outputs tab

Measurements Pane

Use the **Measurements** pane as follows:

1. Click the **Browser** button to navigate to a measurement file.



2. Click **Edit** to modify the measure file, as required.
3. Click **Open** to launch the Measure Utility window. (see [Chapter 12, HSPICE Measurement Utility](#) for a detailed description of the utility).
4. Click **Get Measurements** to load all enabled measurement variables into the measurement report field.
5. Click **Close** to dismiss the HSPICE Measure Utility window.
6. To omit a measurement from the netlist, select the measurement on the report field and click **Delete** (to the right of the measurement report field). All remaining measurement variables are netlisted when you do an HSPICE Monte Carlo simulation.

Simulation Pane

Control the simulation from the Monte Carlo form as follows:

1. Click **Netlist-Run** or **Run** to begin a simulation.
2. Use the **Netlist** button to generate a netlist only from the **Setup** tab.
3. Click the **Output Log** to generate an on-screen running account of the progress of the simulation.
4. Select the type of analysis to be run from the **Analysis** selection box.
5. Select the **Plotting Mode** for waveforms from three options: open a new graph window, append to existing window, or replace an existing graph window.
6. Pre-set the plotting mode option for the waveview window.

Chapter 8: Monte Carlo in the HSPICE Integration

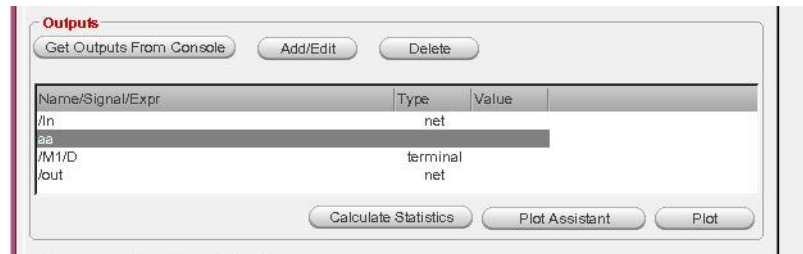
Setting up Measurements, Post-Plotting, and Analyzing the Simulation Results Using the Outputs Tab

Result: The HSPICE simulation log file is displayed after the simulation finishes and the `job concluded` message in the log file indicates a successful simulation.

Outputs Pane

Use the **Outputs** controls to specify output attributes.

1. Click **Get Outputs from Console** to use either the output signals or the expressions from the Environment Console which are set up previously, and which will populate the field below.



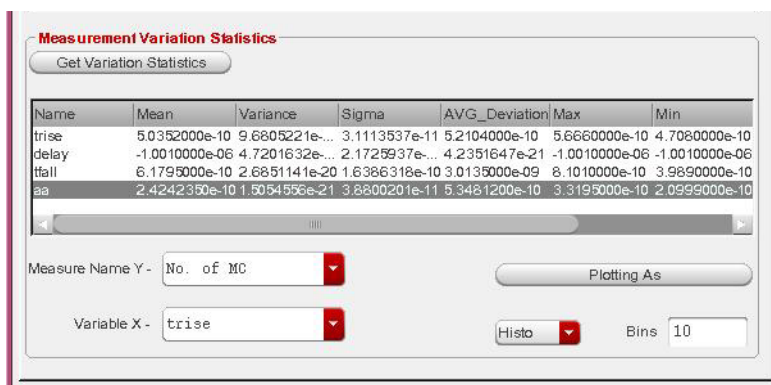
2. Click **Add/Edit** to modify the selected outputs.
3. Click **Delete** to remove selected outputs listed in the outputs field to ensure the signal is not plotted.
4. Click the **Calculate Statistics** button to calculate selected expression output variation statistical results which will automatically enter them in the Measurement Variables Statistics report field as measurement variables. Added measurement variables also can be used to plot waveforms and be printed in a text file. (See [Measurement Variation Statistics on page 209](#).)
5. Select the output signal or expression in the report field, then click either the **Plot Assistant** button to launch the HSPICE plotting utility (see [HSPICE Plotting Assistant on page 174](#)) or click the **Plot** button to directly plot all Monte Carlo waveforms.

Note: Remember to select the correct analysis type and plotting mode through the selection controls noted in [Simulation Pane on page 207](#).

Measurement Variation Statistics

Following a simulation, the **Measurement Variation Statistics** pane provides controls as follows:

1. Click the **Get Variations Statistics** button after selecting the appropriate analysis type. The Variation Statistics field displays all variation information that HSPICE calculates after doing Monte Carlo analysis for specific measurement statements.



Note: You must have input the measurement file for HSPICE to calculate the statistical information. Otherwise, no statistical information is displayed, even if you click the button.

2. Select the X and Y axes names from the selection boxes and enter the number of **Bins** to be plotted.
3. Click **Plotting As** and plot the arbitrary combination of variation parameters and measured results as a **Histogram**, **Scatter**, or **Curve** graph to analyze the Monte Carlo simulation results.
4. Alternatively, use the **Plot** menu on the HSPICE Monte Carlo Analysis form to plot the combination of variation parameters and measured results.



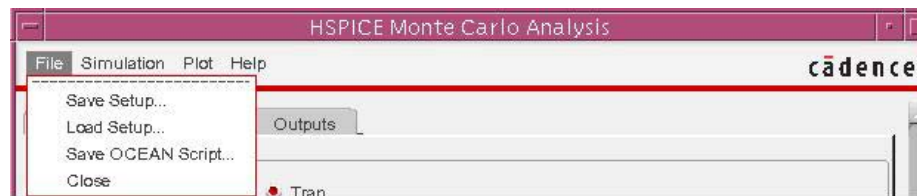
Figure 75 Plot menu on HSPICE Monte Carlo Analysis form

Other Features Supported in the Monte Carlo Analysis Form

You can do save state and load state operations from the Monte Carlo Analysis form.

Saving or loading session statistics: Use the Monte Carlo Analysis form to save and load all the user operations for simulation as a standard ADE simulation, and create an OCEAN script of based on all operations performed in an a Monte Carlo Analysis session.

Note: A limitation in IC51 disallows the plotting of a Histogram waveform.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 76 File menu

1. To Save/Load Statistics, under the File menu, select either the **Save Setup** or **Load Setup** option. Alternatively, select the **Save State** or **Load State** option in the Environment Console to perform the same operation.
2. Create an OCEAN script based on your Monte Carlo session by either:
 - Selecting File >XXXXX and Saving all the operations as a single OCEAN script to do a post batch run simulation with Monte Carlo analysis, or
 - Use the provided OCEAN API functions directly to write your own OCEAN script to do batch simulation. For detailed OCEAN API functions, please refer to the descriptions in [Appendix B, OCEAN API Functions for HSPICE Monte Carlo Analysis](#).
 - Alternatively, use the **Help** button on the HSPICE Monte Carlo Analysis form to assess an html user guide for the Monte Carlo feature or the reference guide for OCEAN API functions

Chapter 8: Monte Carlo in the HSPICE Integration

Other Features Supported in the Monte Carlo Analysis Form



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 77 Help menu

Chapter 8: Monte Carlo in the HSPICE Integration

Other Features Supported in the Monte Carlo Analysis Form

Corners Analysis

Describes the corners analysis capability in the HSPICE integration to the Cadence™ Virtuoso® Analog Design Environment.

Corners analysis enables a convenient way to measure the circuit performance while simulating a circuit with sets of parameter values that represent the most extreme variations expected in the process, voltage, and the temperature values.

The Corners analysis tool in HSPICE ADE works in graphics mode, by using the user interface, and in non-graphics mode, by using the specialized OCEAN commands provided by HSPICE ADE.

For information on running corner simulations using distributed jobs, see [HSPICE Corner Analysis Distributed Simulation on page 328](#).

For paths to corner analysis demonstration cases available with this release, see [HSPICE Integration to ADE Demonstration Examples](#).

The following topics are discussed in these sections:

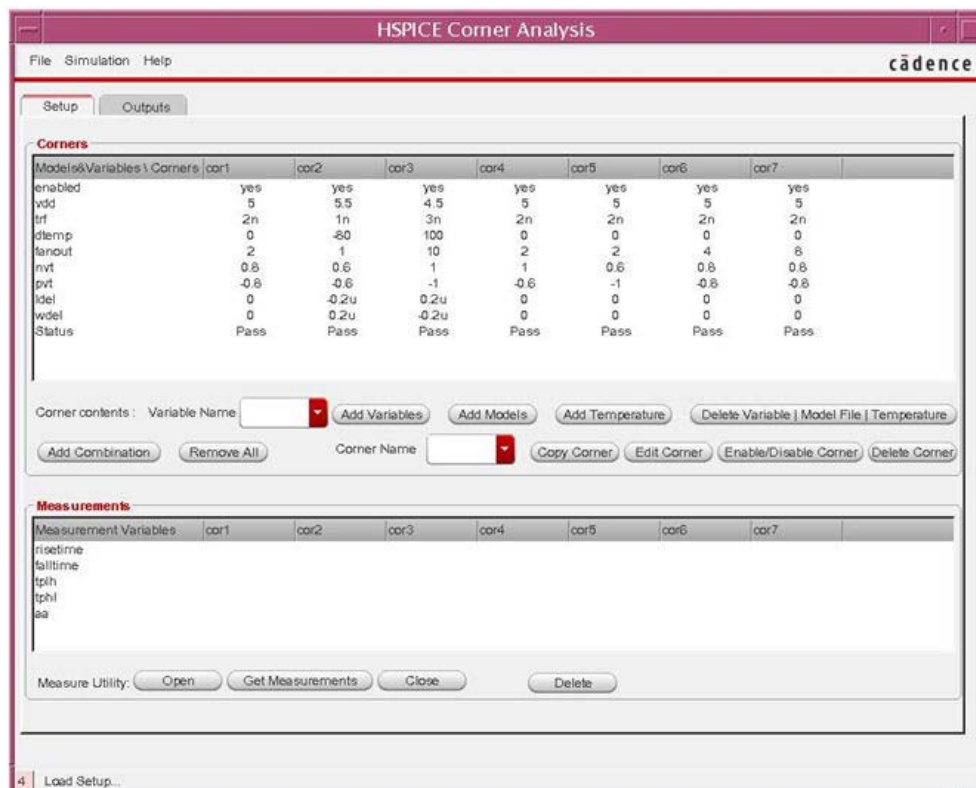
- [Launching the Corner Analysis Tool](#)
- [Setting Up a Corners Analysis](#)
- [Setting Up Measurements for Corners](#)
- [Running Corner Simulations and Using the Outputs Tab](#)
- [Other Features Supported in HSPICE Corner Analysis](#)

Launching the Corner Analysis Tool

In the Environment Console, select **Tools > HSPICE Corner Analysis**. No additional licences are check out during the opening and running of the Corners Analysis tool in the HSPICE integration to ADE.

HSPICE Corner Analysis Window

HSPICE Corner Analysis provides an interface for setting up the corners and the measurements for the corners analysis to be run. The **Setup** tab displays two panes: **Corners** for setup information and **Measurements**.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 78 Corner Analysis Set up tab (populated after a successful simulation)

The **Corners** setup tab displays information about the currently defined corners and output measurements, and setup function controls.

Each column in the *Corners* report field characterizes a corner; each row characterizes an environment variable. The column headings identify each corner name. The name of the environment variable is displayed on the left side of each row. For example, if the environment variable represents the model file, the row states the name of the model file followed by the sections to be used in each corner. If the environment variable represents the design variable, then the row lists the name of the design variable followed by the values of the design variable to be used in each corner. The temperature variable appears as one row by default.

The **Measurements** field displays the information about the currently defined measurements.

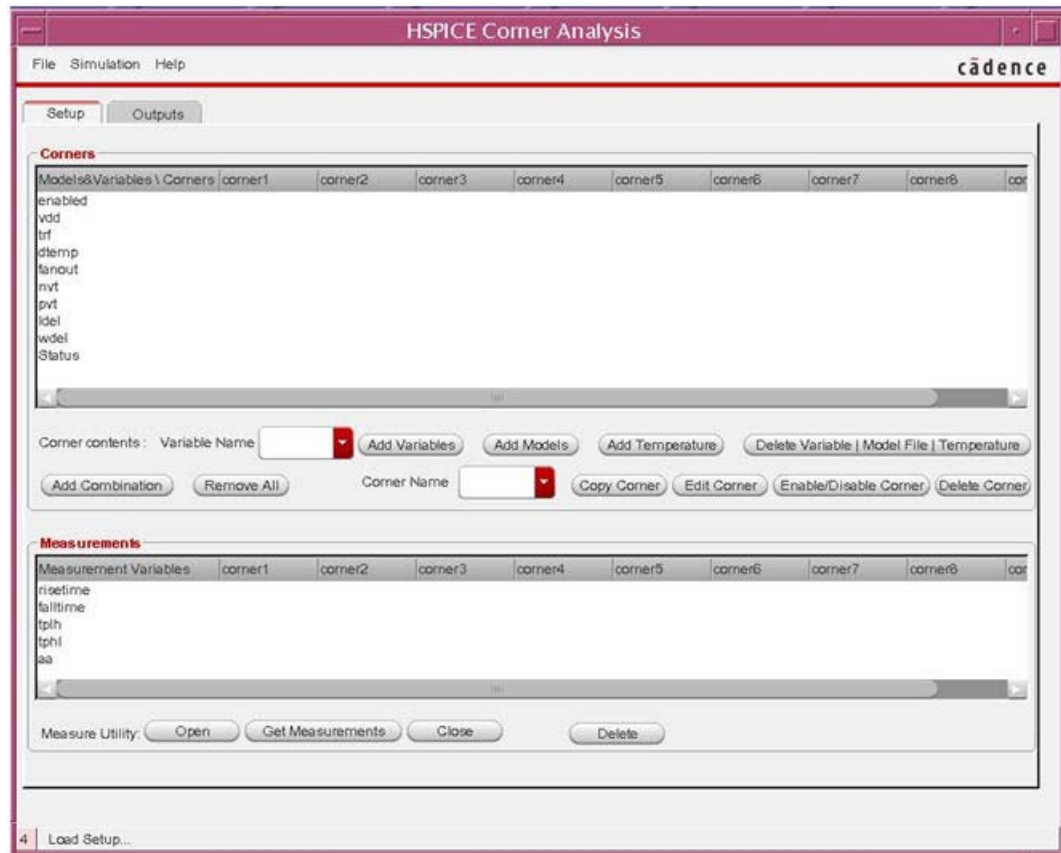
Corner Menus

The menus on the Corner Analysis Window contains the commands needed to setup, run and plot the results of a corners analysis.

- **File**
 - **Load Setup**—Load one existing corners setup and measurements setup information from HSPICE-ADE states.
 - **Save Setup**—Save current corners setting to HSPICE-ADE states.
 - **Save Script As**—For saving the OCEAN script to a specific file.
 - **Close**—Closes the Corners Analysis UI
- **Simulation**
 - **Run_Netlist**—Redo netlist and run the corner analysis simulation.
 - **Run**—Run the corners analysis.
 - **Stop**—Stop the corners analysis running on a single machine.
 - **Netlist**—Create netlist files.
 - **Output log**—Show simulation log file.
- **Help**
 - **Ocean API Reference**—Select to display the corners analysis OCEAN API reference.
 - **HSPICE Corner Analysis User Guide**—Select to display the usage guide for HSPICE-ADE corners analysis.

Setup Tab GUI Elements

The **Corners** pane on the **Setup** tab consists of a table to display the corners and the environment variables, such as temperature, model files, design variables, and measurements. The table is automatically populated through use of controls on the form.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 79 Setup tab with Corners and Measurements panes

The **Corner contents** section displays the following controls and buttons:

- **Variable Name** selector and buttons to:
 - **Add Variables**
 - **Add Models**
 - **Add Temperature**
 - **Delete Variable | Model File |Temperature**

- **Add Combination**
- **Remove All**
- Corner controls include a **Corner Name** selector, and buttons to:
 - **Edit Corner** (opens the Corners Edit form).
 - **Enable/Disable Corner**
 - **Delete Corner**

The **Measurements** pane displays a summary of the corners results in tabular form similar to the Corners report above. The buttons include

- **Open** (launches the HSPICE Measure Utility form)
- **Get Measurements** (from the utility) and **Delete** (to remove any selected row).

Setting Up a Corners Analysis

The **Setup** tab consists of a **Corners** and **Measurements** panes.

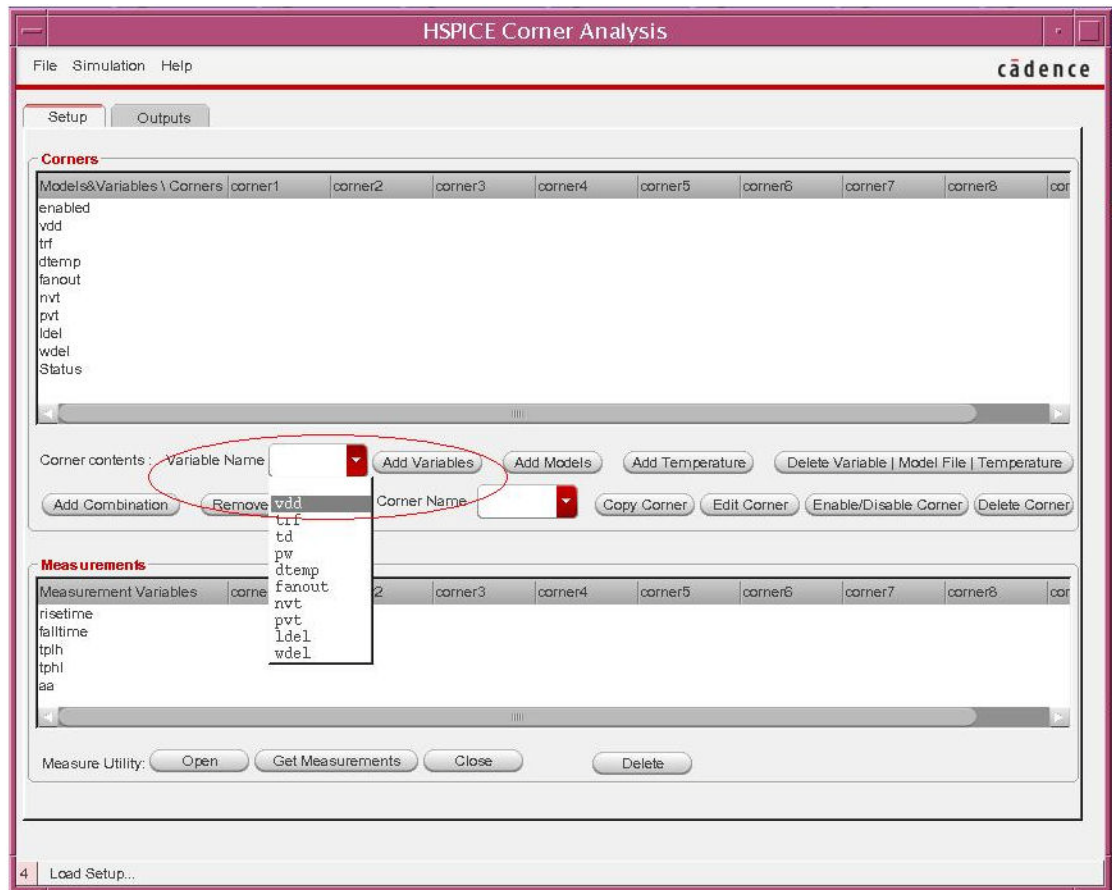
Specifying Design Variables

The **Variable Name** field initially is filled by the Environment Console's design variables as its choices so that you can select them and add to the corner analysis. Set up corner contents as follows:

1. Click **Add Variables** to add design variables into the HSPICE Corner Analysis environment. For example, [Figure 80 on page 218](#) shows adding the “vdd” design variable to a corner analysis.

Chapter 9: Corners Analysis

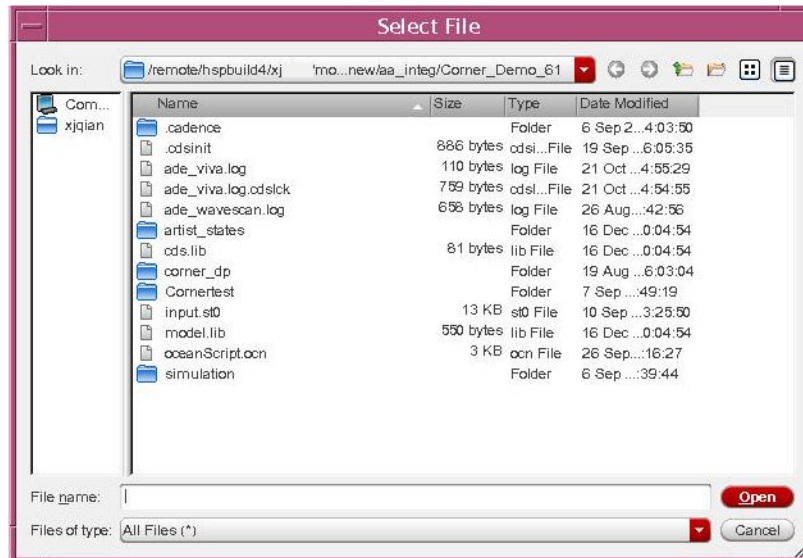
Setting Up a Corners Analysis



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 80 Add Variable operation

2. Click **Add Models** to bring a model file into the HSPICE Corner Analysis environment. The Select File browser opens ([Figure 81 on page 219](#)).



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 81 Model Library Setup form

3. Add a single model file if all models for all corners are located in a single file or add multiple model files if the models are located in the multiple model files. Your model files are added to the **Corners** report field on the **Setup** tab. If you have already added some corners, the added model file values will be “nil.”
4. Use the **Add Temperature** button to add a temperature parameter. When Temperature parameter exists, this button is disabled. The ADE environment default includes the Temperature parameter.
5. To delete a design variable row (model file, design variable, or temperature), select it on the report field, then click **Delete Model File|Variables|Temperature**. A deletion removes these variables from the Corners environment, but they remain in the ADE environment.

Setting up Corners

Use the Add Combination Form to change or add values to the variables parameters.

To make changes or modify values to variables/parameters:

1. Click **Add Combination** to open the Add Combination Form ([Figure 82 on page 220](#)).

Chapter 9: Corners Analysis

Setting Up a Corners Analysis

Add Combination Form

Please input variable or parameter values to add corners.

Use commas ',' or empty spaces to delimit values when adding combination values.

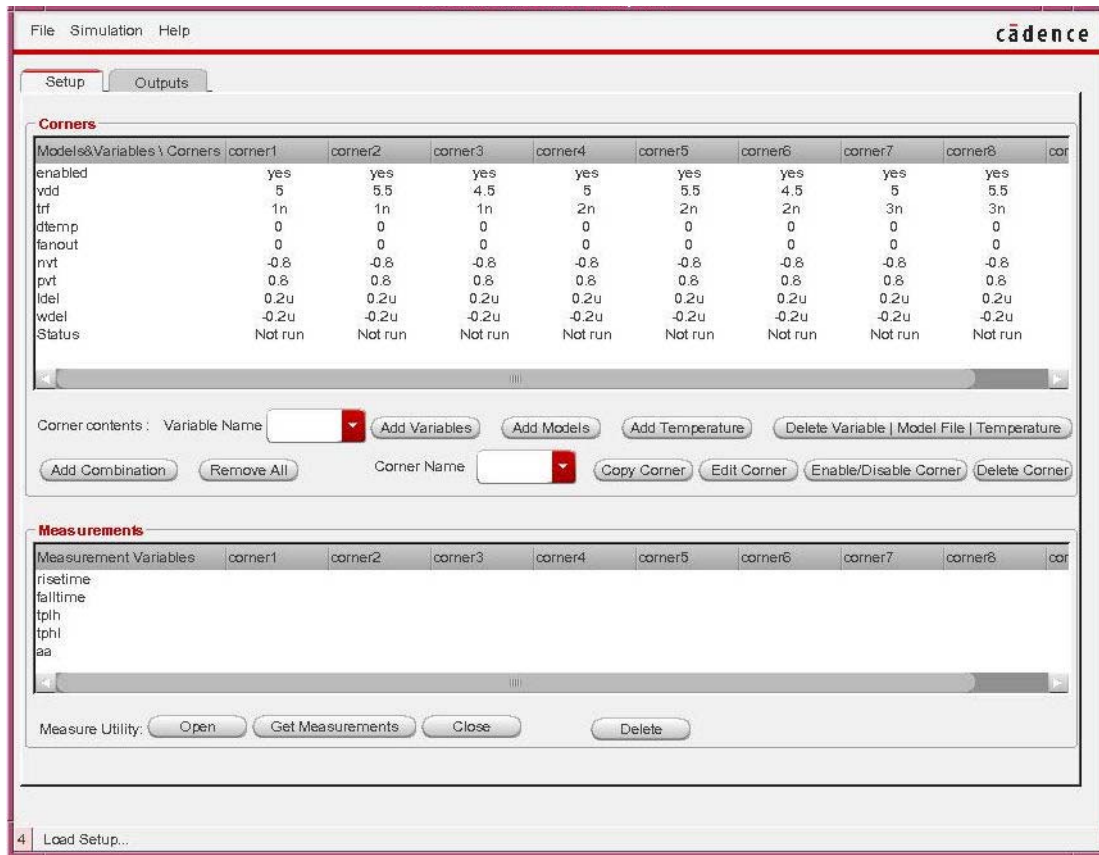
vdd	5, 5, 5, 4, 5
trf	1n 2n 3n
dtemp	0
fanout	2
nvt	0.8
pvt	-0.8
ldel	0
wdel	0.2

Add Clear Close Help

Figure 82 Add Combination Form with delimited values

2. Enter values for each parameter; when you input multiple values for a parameter, use commas “,” or empty spaces as delimiters between values.
3. On the Add Combination Form, click **Clear** if you wish to delete all entries or click **Add** to create corners based on the values you input and add them to the **Corners** report field ([Figure 83 on page 221](#)).

Note: The added corners are named “cornerXX” in which XX is the corner index number.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 83 Added Corners via the Add Combination Form

- To remove all corners, parameters, and their values from the **Corners** report field, click the **Remove All** button.

Modifying Corners

You can further modify a corner by using the group of controls beginning with the **Corner Name** selection box.

Uses the Corner buttons as follows:

- Click **Edit Corner** to set all the values for its model files (or section of the file), design variables, and temperature. See [Editing a Corner on page 222](#) for discussion of methods to edit a corner.
-

3. Select an existing corner name in the **Corner Name** field and modify it by using these buttons:
 - **Copy Corner:** Select a corner's contents and values. The default for the new corner's name is *Corner_name_Copy*. For example, if the selected corner name is "cor1," then the copied corner name is "cor1_Copy." To copy a corner, select an existing corner from the **Corner Name** selection box and click the **Copy Corner** button. The copied corner appears in both the **Corners** report and **Corner Name** selection fields.
 - **Enable/Disable Corner:** Allows you to set an enabling flag for a existing corner so that it can be netlisted for a Corner Analysis simulation. If a corner is disabled, it is not netlisted and used for simulation. View all corners' statuses in the "Enabled" row of the **Corners** report field. To change a corner's status, select the corner in the **Corner Name** field and click the **Enable/Disable Corner** button.
 - **Delete Corner:** Used to delete a corner from the current corner analysis environment. To delete a corner, select it in the **Corner Name** field and click the **Delete Corner** button.

Editing a Corner

There are two methods to edit a corner:

- 1) Select a corner and edit all values to do with model files, design variables, and Temperature; or
- 2) Edit different corners within the model file, design variable, or temperature parameter.

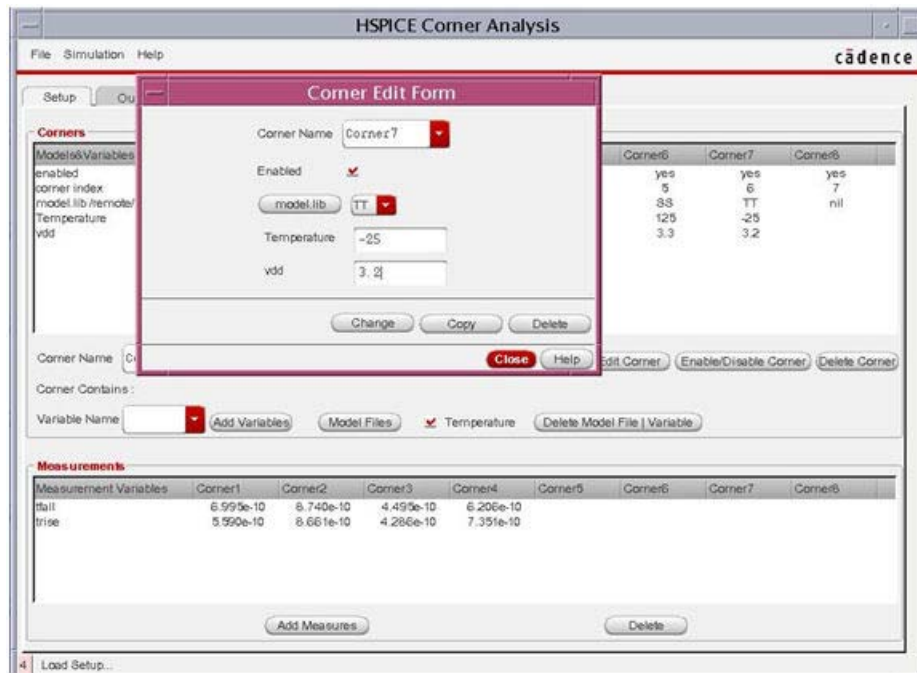
Corner Editing Method 1	Corner Editing Method 2
1. Select the corner in the Corner Name field and click the Edit Corner button to open the Corner Edit Form.	1. Either click Edit Corner after selecting a row in the Corners report field or double-click the row to open the Corners Edit form (Figure 85 on page 224).
2. Modify all values shown on this form. For example, Figure 84 shows the edit to the Corner7 vdd value to "3.2"	2. Modify any values shown on this form and click Apply or OK on the form to save.

Corner Editing Method 1

3. Click **Change** to repopulate the **Setup** tab with the updated value.

Corner Editing Method 2

3. Use the radio buttons at the top of the form to edit corners for other Variables, Models, or Temperature. After successfully editing, the corners information in the **Corners** report field automatically updates.

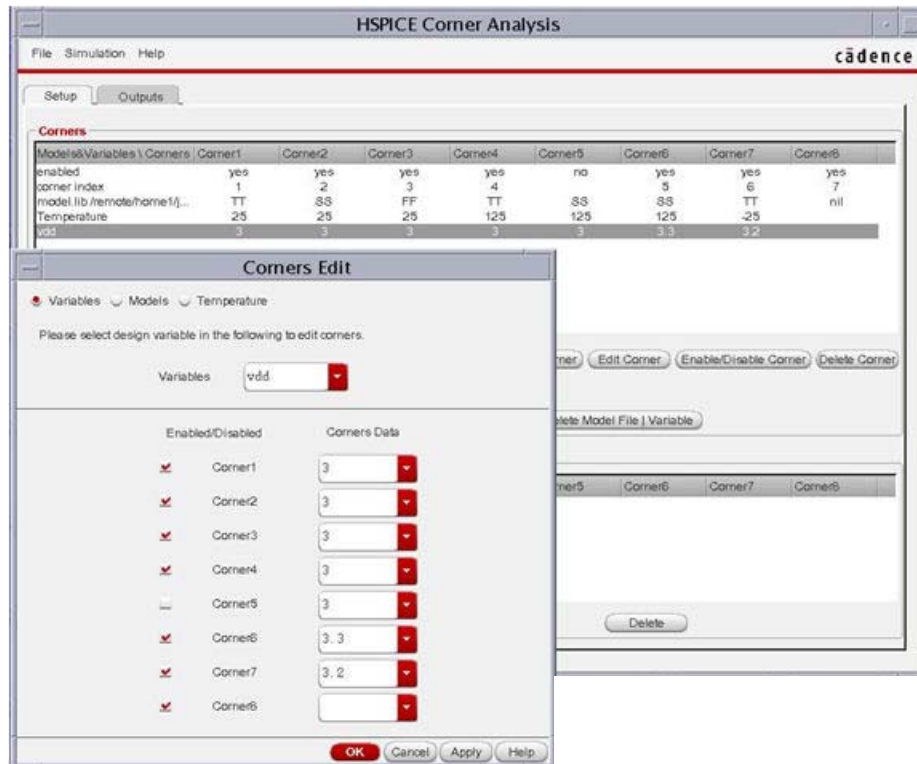


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 84 Method 1: Corner Edit form

Chapter 9: Corners Analysis

Setting Up Measurements for Corners



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 85 Method 2: Corners Edit form

Setting Up Measurements for Corners

This section discusses adding measurement variables and setting up measurements summarization. HSPICE corner analysis measurements information includes two parts: different corners measurement variables results and the measurement variables results summary information, displayed on the **Measurements** report field on the **Setup** tab and **Measurements Summary** report field on the **Outputs** tab. (See [Figure 86 on page 225](#)).

These fields display the currently-defined measurements variables and output results after corners analysis simulation. The **Measurements Summary** report field retrieves and displays the summary scalar results of all corners' measurements variables results for enabled corners.

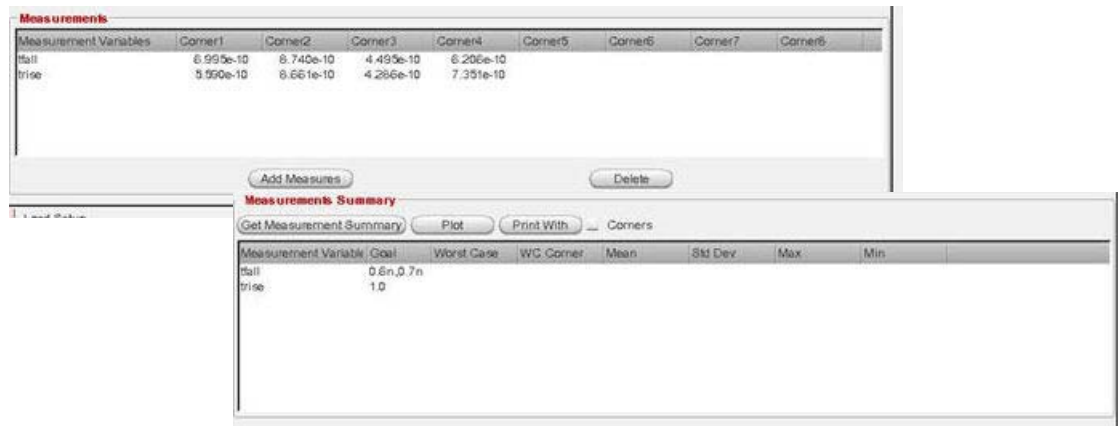


Figure 86 Measurements fields for Setup and Outputs tabs

Loading Measurements for Setup

The **Corner Measurements** form can be used to load measurement variables from HSPICE Measure Utility and configured to do measurement summarization. Click the **Add Measurements** button in the **Setup** tab to display the Corners Measurements form [Figure 87 on page 225](#).

Variable Names	Summarize	Goal
delay	no	+/-1u
tfall	yes	200p, 500p
trise	yes	0.3

Goal Type	Range
Low Value	200p
Value	500p

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 87 Corners Measurements form

The Corners Measurements form is used to:

- Load measurement variables (including goal information) from the HSPICE Measure Utility window
- Set the loaded measurement variables' summarization
- Set/edit measurement variables' goal information
- Enable or disable summarization

Using the Corners Measurements form

Use the Corners Measurements controls as follows:

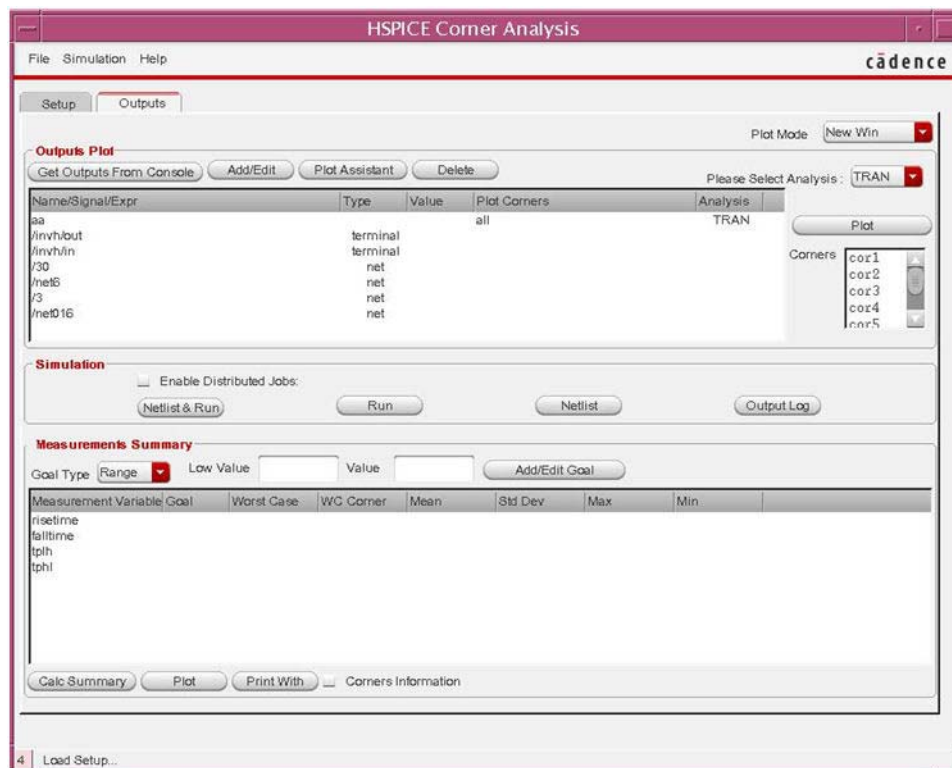
1. Click **Open** to display the HSPICE Measurements Utility window in order to add/edit/delete measurement variables.
2. Click **Get Measurements** to retrieve all defined enabled measurement variables and display them in a table in the lower pane of the **Setup** tab (the **Measurements Variables** report field).
3. Click **Close** to dismiss the HSPICE Measure Utility window.
4. Use the **Goal Setup** pane to set the measurement variable "goal" to do measurement summarization after a HSPICE Corner Analysis simulation ends. Select measurement variable row in the measurement variable report field before creating a goal setup.

Note: For the **Goal Type Range**, both the **Low Value** and **Value** fields require input. For other goal types, only the **Value** field needs to be input.

5. Click **Enable/Disable Summarize** to enable/disable the summarization of the selected measurement variable in the report field.
6. Click **Delete** to disable netlisting of the selected measurement variable in **Measurement Variables** report field on the **Outputs** tab. If a measurement variable is deleted here, the measurement variable is not netlisted when you run a corners analysis.

Running Corner Simulations and Using the Outputs Tab

After completing entries in the corners **Setup** tab, you can use the Outputs tab set plot outputs, run corners simulations, and create a **Measurements Summary** listing all variables, including plotting measurement results.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 88 Outputs tab

Simulation

You can use either the **Simulation** pane or the **Simulation** menu on the Corners Analysis form to prepare a netlist and run a corners simulation.

The **Simulation** pane buttons include:

- **Netlist & Run Sim**—Recreates the netlist file and starts a simulation.
- **Run**—Starts a simulation.
- **Netlist**—Recreates the netlist including any updates.
- **Outputs Log**—Displays Corner Simulation Outputs Log View window.

Selecting Simulation Mode

HSPICE corner analysis simulation supports three modes:

- Sequential run - create netlist for each corner and simulation them one by one
- Multiprocess run — create one netlist file making use of HSPICE “.alters” and use the HSPICE “-mp” option to simulate the netlist file (see [Setting Up Multiprocessing of Corner Analysis](#)).
- Distributed Process run — create a netlist file for each corner and assign corners simulation to several jobs to finish all corner simulations (see [Chapter 15, Distributed Mode—Monte Carlo/Corner Analyses](#)).

Setting Up Multiprocessing of Corner Analysis

To set up a multiprocessing simulation for Corner analysis:

1. Select **Simulation > MP simulation** in the HSPICE Corner Analysis window to open the Select MP option for Corner form (see [Figure 89](#)).
2. Set up the multiprocess count according to the instructions in the form. The count selected in the window is in effect for the Corner analysis session only. (For details on HSPICE multiprocessing, see [Running Multithread/Multiprocess HSPICE Simulations](#) in the *HSPICE User Guide: Simulation and Analysis*.)
3. Click **OK** in the Select MP option... form.
4. In the HSPICE Corner Analysis window click **Netlist & Run** to produce a netlist with a series of .alter commands and simulation log file similar to those shown in [Figure 90 on page 229](#).

Chapter 9: Corners Analysis

Running Corner Simulations and Using the Outputs Tab

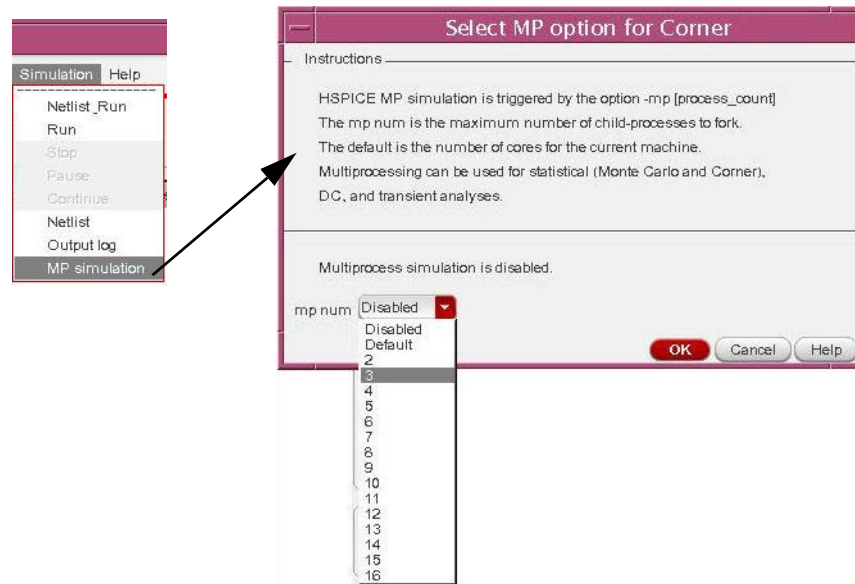
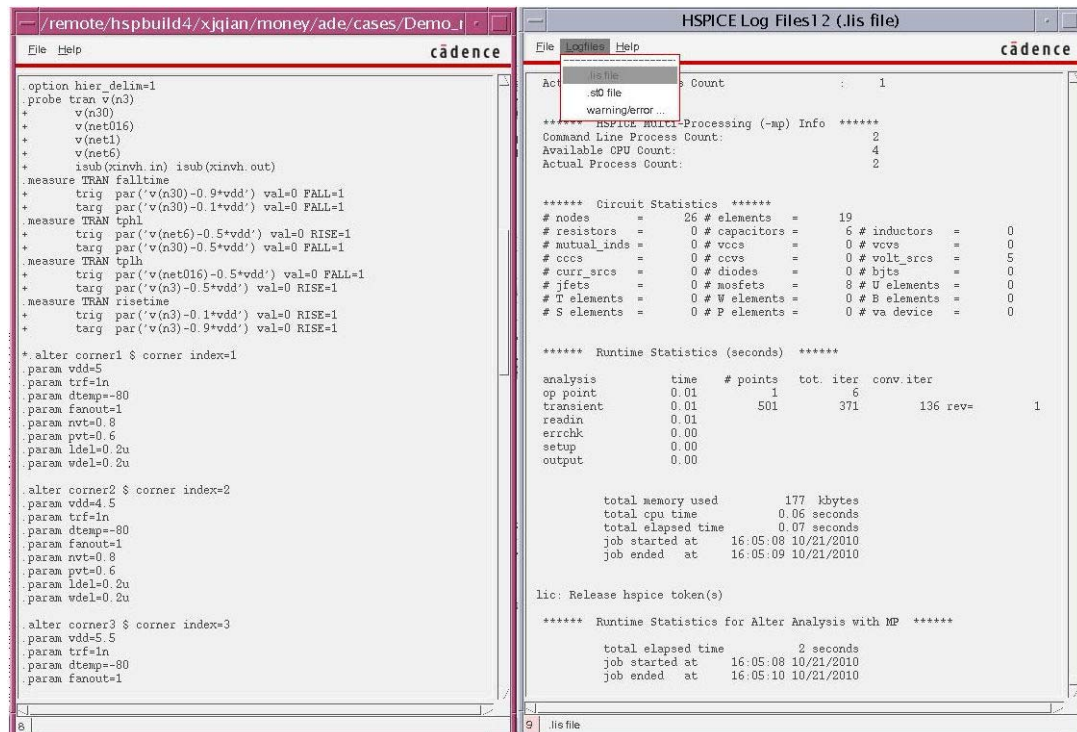


Figure 89 Multiprocessing for Corner Analysis



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 90 Netlist and log file for multiprocessed Corner analysis simulation

Chapter 9: Corners Analysis

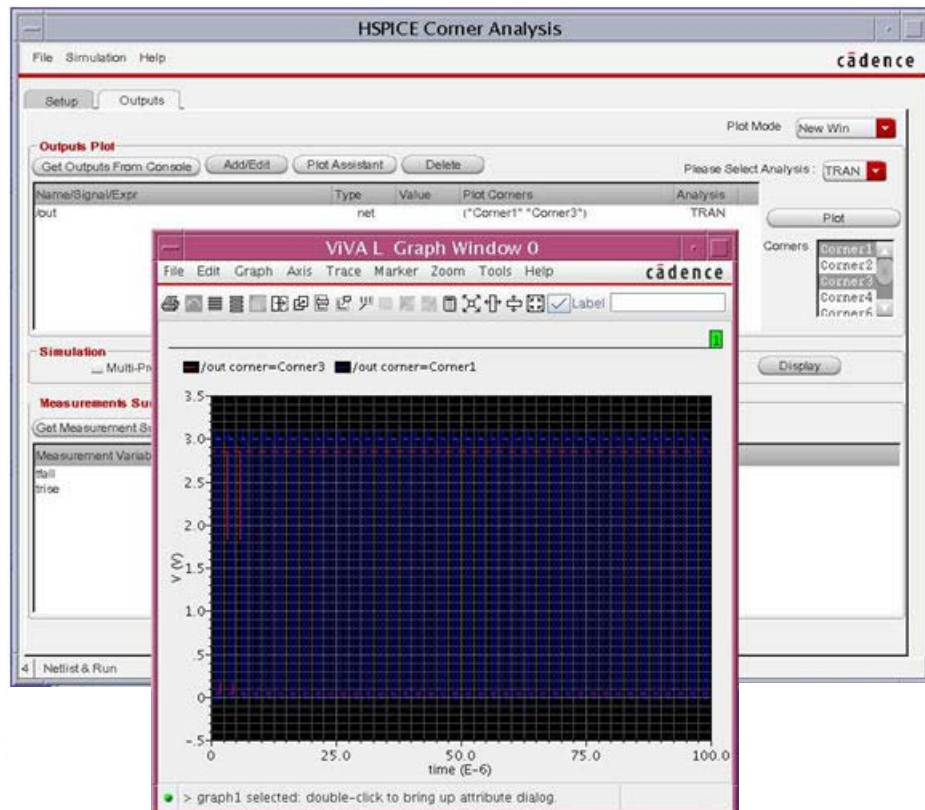
Running Corner Simulations and Using the Outputs Tab

Plotting Outputs

After successfully netlisting and simulation, you can plot outputs in the using the **Output Plots** pane.

To plot an output signal or expression:

1. Click **Get Outputs From Console** or **Plot Assistant** to load the existing outputs into the **Outputs** report field
2. Select one row in the Outputs report field.
3. Select the required corners in the **Corners** list box and click the **Plot** button. The tool plots the output in a waveform. For example, [Figure 91](#) plots the “/out” signal for Corner1 and Corner3.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 91 Waveform for selected corners

4. When the selected output is a measure expression, the output is automatically added to the **Measurements** and **Measurements Summary** fields after plotting. Figure 92 shows “aa” (delay(VT("/net1") 2.5 1 "either" VT("/net6") 2.5 1 "either" 0 0 nil nil)) added to the **Measurements Summary** report field after it is plotted.

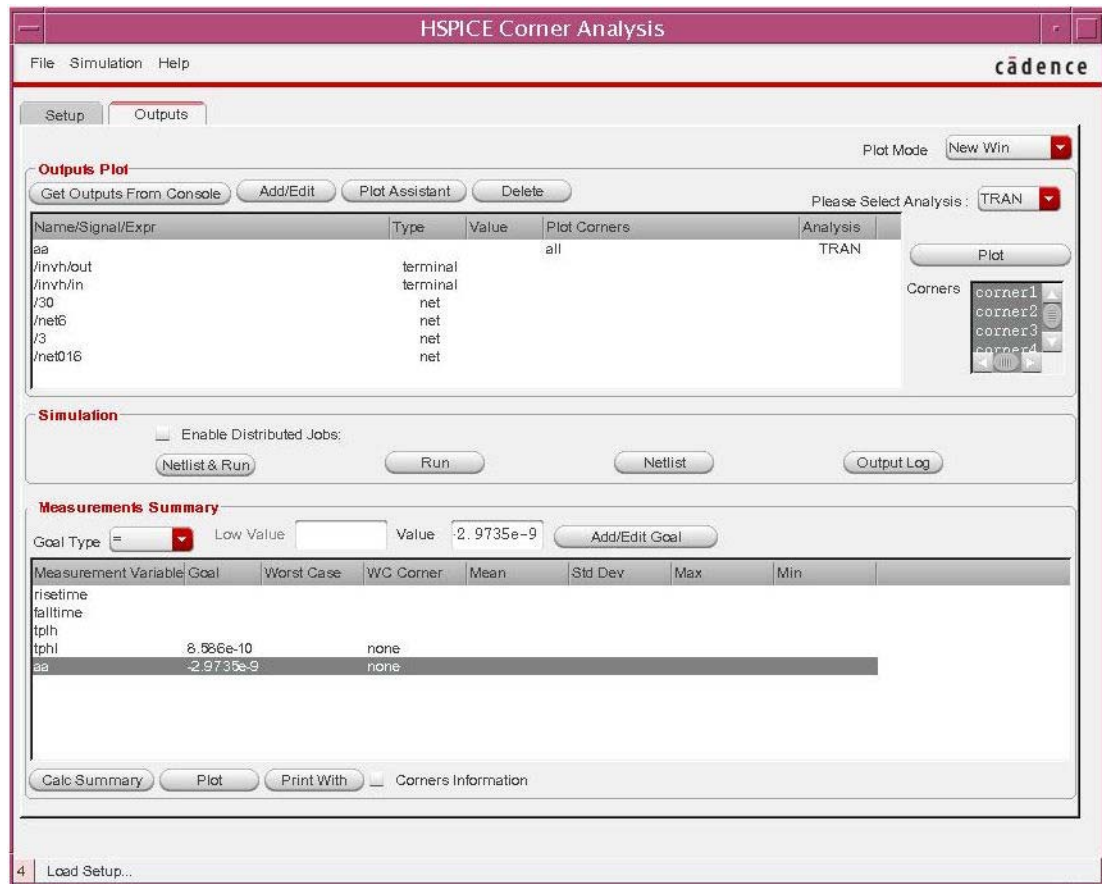


Figure 92 Measurement Variable aa expression added to Measurements Summary report

Getting Measurements, Plotting and Printing

After successfully netlisting and simulation, the measurements variables results and summarized results are automatically loaded and displayed to the **Measurements** report field in on the **Setup** tab and the **Measurements Summary** report field on the **Outputs** tab.

Chapter 9: Corners Analysis

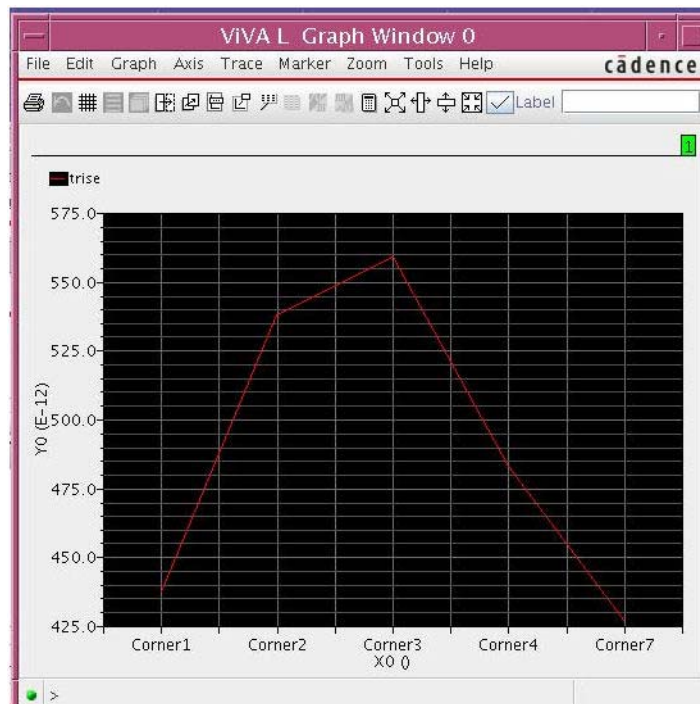
Running Corner Simulations and Using the Outputs Tab

You can also click the **Get Measurements Summary** button to reload them.

Plotting a Measurement Variable

To plot a measurement variable simulation result based on corners:

1. Select the measurement variable row in the **Measurement Summary** report field.
2. Click the **Plot** button.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 93 Plot of trise over seven corners

3. Click **Print With** to generate a text file containing all measurements results.
4. Enable the **Corners** check box to include corners information in the generated text file, as shown in [Figure 94](#).

Models&Variables \ Corners	Corner1	Corner2	Corner3	Corner4	Corner5	Corner7
enabled	yes	yes	yes	yes	no	yes
corner	1	2	3	4		5
model lib	TT	SS	FF	TT	SS	TT
Temperature	25	25	25	125	125	-25
vdd	3	3	3	3	3	3.2

Measurement Variables	Corner1	Corner2	Corner3	Corner4	Corner7
tfall	2.480e-10	6.871e-10	6.995e-10	8.255e-10	4.501e-10
trise	4.373e-10	5.379e-10	5.590e-10	4.830e-10	4.273e-10

Measurement Variables	Goal	Worst Case	WC corner	Mean	Std Dev	Max	Min
tfall	0.6n, 0.7n	2.480e-10	Corner1	5.820e-10	1.837e-10	2.480e-10	8.255e-10
trise	1.0	4.273e-10	Corner7	4.889e-10	4.944e-11	4.273e-10	5.590e-10

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 94 Results Display Window for Corners Measurement Variables

Other Features Supported in HSPICE Corner Analysis

You can do save state and load state operations from the HSPICE Corners Analysis form.

Saving or loading session statistics: Use the HSPICE Corners Analysis form to save and load all the user operations for simulation as a standard ADE simulation, and create an OCEAN script of based on all operations performed in an Corners Analysis session.

1. To Save/Load Statistics, under the File menu, select either the **Save Setup** or **Load Setup** option. Alternatively, select the **Save State** or **Load State** option in the Environment Console to perform the same operation.
2. Create an OCEAN script based on your Corners Analysis session by either:
 - Selecting File >XXX and Saving all the operations as a single OCEAN script to do a post batch run simulation with corner analysis, or
 - Use the provided OCEAN API functions directly to write your own OCEAN script to do batch simulation. For detailed OCEAN API functions, refer to the descriptions in [Appendix D, OCEAN API Functions for HSPICE Corner Analysis](#).
 - Alternatively, use the **Help** button on the HSPICE Corners Analysis form to assess an html user guide for the corners feature or the reference guide for OCEAN API functions

Chapter 9: Corners Analysis

Other Features Supported in HSPICE Corner Analysis

HSPICE RF Analysis

Describes HSPICE RF capability in the HSPICE integration to the Cadence™ Virtuoso® Analog Design Environment.

For the path to a suite of examples of RF analyses using the HSPICE Integration to ADE, see Mixer_Demo_61 in [HSPICE Integration to ADE Demonstration Examples](#).

The following sections discuss these topics:

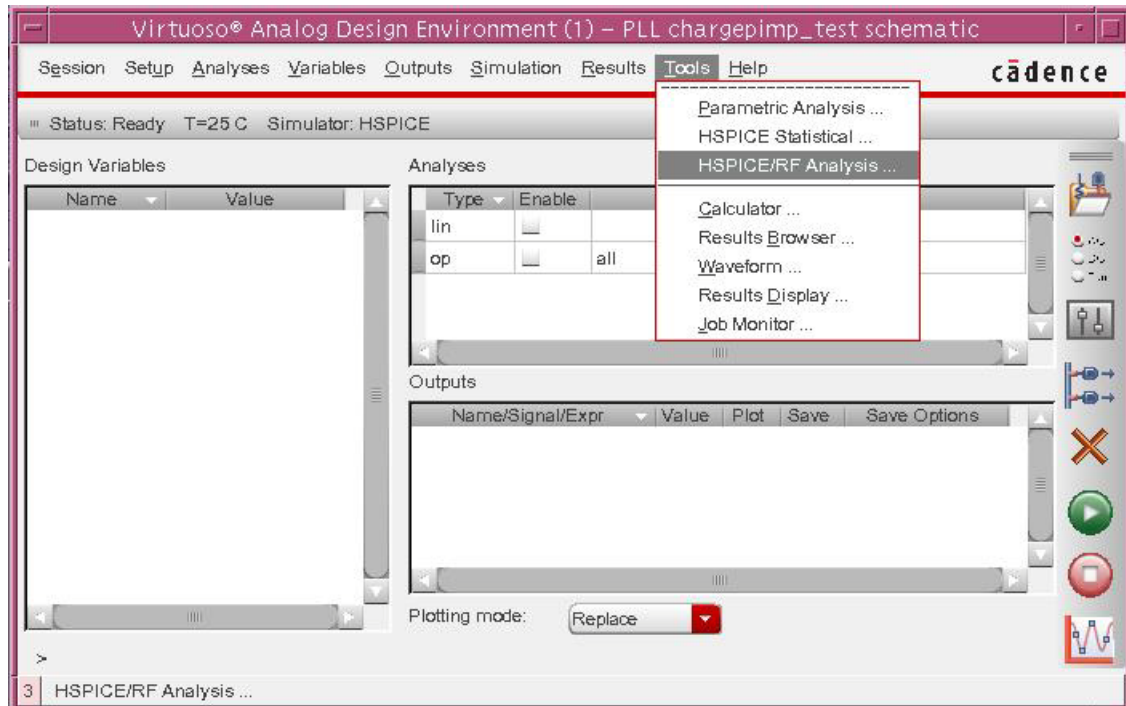
- [Accessing and Setting Up an HSPICE RF Analysis](#)
- [Setting Up the Analysis Tab to Run an RF Analysis](#)
- [Using the Options Tab of the HSPICE RF Analysis Form](#)
- [Using the Outputs Tab](#)
- [Running and Post-Processing an RF Simulation](#)
- [Post-Simulation Plotting and Analysis](#)
- [Other Features Supported in the HSPICE RF Analysis Form](#)
- [Current RF Analysis Feature Limitations](#)

Accessing and Setting Up an HSPICE RF Analysis

HSPICE RF analysis provides harmonic balance and Shooting Newton algorithm-based RF simulation methodologies. To invoke the Monte Carlo Analysis form, select the **HSPICE RF Analysis** option under the **Tools** menu of the Environment Console to open the Monte Carlo Analysis form.

Chapter 10: HSPICE RF Analysis

Setting Up the Analysis Tab to Run an RF Analysis



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 95 HSPICE RF option on Environment Console Tools menu

Setting Up the Analysis Tab to Run an RF Analysis

Use the **Analysis** tab to specify an HSPICE RF Steady State or Small Signal Analysis to launch an HSPICE RF simulation. See the *HSPICE User Guide: RF Analysis* for details on HSPICE RF simulation types.

Note: All fields containing an asterisk (*) must be input to successfully add an analysis.

Chapter 10: HSPICE RF Analysis

Setting Up the Analysis Tab to Run an RF Analysis

The screenshot shows the 'HSPICE RF Analysis' dialog box with the 'Analysis' tab selected. The dialog is organized into several sections:

- Analysis Report section:** Located at the top, it contains a table with columns 'Index', 'Analysis', 'Enable', 'Arguments', and 'Sweep'. Below the table are 'Enable/Disable' and 'Delete' buttons.
- Main analysis input section:** This section includes:
 - A note: 'The contain (*) fields must be input.'
 - Analysis type selection: ☒ HB, ☐ HBOSC, ☐ SN, ☐ SNOSC.
 - HB analysis sub-section:
 - *Fundamental Tones: Two input fields and a 'More' button.
 - *Num of Harmonics: Two input fields.
 - Intermodulation Max Order: One input field and a checkbox for 'SS_TONE'.
 - Sweep settings:
 - ☒ Sweep, Design Variables dropdown, and a 'Select' button.
 - Sweep Type: LIN dropdown.
 - Start, Stop, and Nstep input fields.
 - Buttons: 'Add', 'Change', and 'Clear'.
- Sub-type analysis input section:** This section includes:
 - Sub-types: ☒ HBAC, ☒ HBNOISE.
 - HBNOISE analysis sub-section:
 - Analyzing signal: Volt of net dropdown.
 - *Net: Input field and 'Select' button.
 - *InputSrc: Input field and 'Select' button.
 - *Frequency Sweep: LIN dropdown.
 - *Start, *Stop, and *Nstep input fields.
 - Frequency Index List: Input field.
 - ListFreq: none dropdown.
 - ListCount, ListFloor, and ListSource input fields.
 - Buttons: 'Add', 'Change', and 'Clear'.

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 96 Detailed Analysis Tab of the HSPICE RF Analysis Form

In the **Analysis** tab of HSPICE RF Analysis form:

Chapter 10: HSPICE RF Analysis

Setting Up the Analysis Tab to Run an RF Analysis

1. Select the desired analysis option (**HB**, **HBOSC**, **SN**, or **SNOSC**) from the Analysis radio buttons. The appropriate analysis-related fields are displayed for required user input. For example, [Figure 97](#) shows the **Analysis** form for an HBOSC simulation.

The screenshot shows the 'HSPICE RF Analysis' dialog box. The 'Analysis' tab is selected, displaying a table with columns: Index, Analysis, Enable, Arguments, Sweep. Below the table, there are radio buttons for 'HB', 'HBOSC' (selected), 'SN', and 'SNOSC'. The 'HBOSC analysis' section includes fields for '*Approximate Osc Freq', '*Num of Harmonics', 'Intermodulation Max Order', '*ProbeNodes Positive', '*Negative', '*Value', 'FSTPS: Num', 'Min', 'Max', 'Stability', 'SS_TONE', and 'SubHarm'. The 'Sweep' section has a checked 'Sweep' checkbox, 'Design Variables', 'Sweep Type' (LIN), 'Start', 'Stop', and 'Nstep' fields. The 'Sub-types' section is checked and shows 'PHASENOISE' analysis. The 'PHASENOISE analysis' section includes 'Analyzing signal' (Volt of net), '*Net', '*Frequency Sweep' (LIN), '*Start', '*Stop', '*Nstep', 'Method', 'Carrier Index', 'ListFreq' (none), 'ListCount', 'ListFloor', 'ListSource', and 'Spurious' fields. Buttons for 'Add', 'Change', and 'Clear' are present at the bottom of each section.

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 97 HBOSC Analysis Form

2. Input Steady state analysis information in the main analysis input section.
3. Check the **Sweep** button to display and input analysis sweep information.
4. Click the **Add** button in the main analysis input section to add a steady state analysis to HSPICE RF simulation.

Result: If the steady-state analysis is successfully added, the detail is added to the analysis report field and all the input fields are cleared; if the steady-state analysis is not added successfully, an error or warning message appears in the `icms` window. You can correct and continue add the analysis.

5. Click the **Sub-types** button to display the Sub-type analyses input section for you to set up a small-signal analysis. Analyses types include HBAC, HBNOISE, SNAC, and SNNOISE.
6. Click the **Add** button in the **Sub-types** analysis section to add a small-signal analysis to HSPICE RF simulation.

Result: If the small-signal analysis is successfully added, the detail is added to the analysis report field and all the input fields are cleared; if the steady-state analysis does not add successfully, an error or warning message appears in the `icms` window. The related steady-state analysis must be added before a sub-type small signal analysis is added. You can correct and continue the add procedure.

7. Select the analysis item in the analysis report field to restore its detail information back to the input fields so that you can change it.
8. In the analysis report (top) section, use the **Enable/Disable** or **Delete** controls to enable, disable, or delete the selected analysis.

Using the Options Tab of the HSPICE RF Analysis Form

Use the **Options** tab to set or reset HSPICE RF simulation options the same way you would use any simulation options provided by HSPICE as described in [Chapter 6, Running Simulations and Using Control Options](#).

Select the **Options** tab to display the **Options Filter** page which opens by default to the **RF commonly used** command options ([Figure 98](#)). This window includes only the options applicable to HSPICE RF simulations.

1. To select a different group in the options filter, Select from two other lists, **HB & SN subtypes** and **OSC & phasenoise**.

Result: Selecting a different group in the Options Filter displays its related group options in the **Options** tab with their default value in the input field. You can modify or input each displayed option with appropriate values.

2. To change the default values, use the **Reset All From Cdsenv** or **HSPICE** buttons to reset all options field values to either the HSPICE integration default values (Cdsenv) or HSPICE options default values.
3. When you complete any changes to the option defaults, click **Apply** to ensure that the change is added to the simulation run.

Available Options

The **Commonly used** options for RF are:

- **TRANFORHB:** Forces HB analysis to recognize/ignore specific V/I sources.
- **PROBE:** Limits post-analysis output to only variables specified in `.PROBE` and `.PRINT` commands.
- **HB_GIBBS:** Option for HBTRAN output to minimize Gibbs' phenomena.
- **SIM_ACCURACY:** Sets and modifies the size of timesteps.
- **HTOL:** Specifies the absolute error tolerance for determining convergence.
- **HBACTOL:** Specifies the absolute error tolerance for determining convergence and overrides the corresponding PAC option if specified.
- **SNACCURACY:** Sets/modifies timestep size for SN simulation.
- **PHASENOISETOL:** Specifies the error tolerance for the phase noise solver.
- **HBSOLVER:** Specifies a preconditioner for solving nonlinear circuits.
- **PHNOISELORENTZ:** Turns on a Lorentzian model for phase noise analysis.
- **HBTRANINIT:** Selects transient analysis for initializing all state variables for HB analysis of a ring oscillator.
- **LOADSNINIT:** Loads operating point saved at the end of SN initialization.
- **SAVESNINIT:** Saves operating point at the end of SN initialization (sninit).
- **LOADHB:** Loads state variable information from a specified file.
- **SAVEHB:** Saves the final-state variable values from an HB simulation.

Name	Value	Default status
Input options		
TRANFORHB	0	Default
Output options		
PROBE	0	Default
HB_GIBBS	0	Default
Accuracy & Performance		
SIM_ACCURACY	1	Default
HBTOL	1n	Default
HBACTOL	10n	Default
SNACCURACY	10	Default
PHASENOISETOL	10n	Default
Algorithm selection		
HBSOLVER	1	Default
PHNOISELORENTZ	1	Default
Initialization conditions		
HBTRANINIT	0	Default
LOADSNINIT	'/path/SNinitial.input'	Default
SAVESNINIT	'/path/SNinitial.output'	Default
LOADHB	'/path/filename'	Default
SAVEHB	'/path/filename'	Default

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 98 Options tab for HSPICE RF Analysis

The **HB & SN and Subtypes** options are as follows:

- HBSOLVER: Specifies a preconditioner for solving nonlinear circuits.
- HBTOL: Specifies the absolute error tolerance for determining convergence.
- HBMAXITER: Specifies the maximum number of Newton-Raphson iterations performed by the HB engine.

Chapter 10: HSPICE RF Analysis

Using the Options Tab of the HSPICE RF Analysis Form

- HBCONTINUE: Specifies whether to use the sweep solution from the previous simulation as the initial guess for the present simulation.
- HBACKRYLOVDIM: Specifies the dimension of the Krylov subspace used by the Krylov solver.
- HBKRYLOVTOL: Specifies the error tolerance for the Krylov solver.
- HBKRYLOVMAXITER: Specifies the maximum number of GMRES solver iterations performed by the HB engine.
- HBLINESEARCHFAC: Specifies the line search factor.
- HBJREUSE: Controls when to recalculate the Jacobson matrix.
- HBJREUSETOL: Determines when to recalculate Jacobian matrix if $HBJREUSE=1$. 0.
- TRANFORHB: Forces HB analysis to recognize/ignore specific V/I sources.
- HBACTOL: Specifies the absolute error tolerance for determining convergence and overrides the corresponding PAC option if specified.
- HBACKRYLOVDIM: Specifies the dimension of the Krylov subspace used by the Krylov solver.
- HBACKRYLOVITER: Specifies the number of GMRES solver iterations performed by the HB engine.
- SNACCURACY: Sets and modifies the size of timesteps.
- SNMAXITER: Sets the maximum number of iterations for a Shooting Newton analysis.
- LOADSNINIT: Loads operating point saved at the end of SN initialization.
- SAVESNINIT: Saves operating point at the end of SN initialization (sninit).

Oscillator and phase noise analysis options are listed on the **OSC&phasenoise** selection box selection on the **Options** tab:

- HBFREQABSTOL: Specifies the maximum relative change in frequency between solver iterations for convergence.
- HBFREQRELTOL: Specifies the maximum relative change in frequency between solver iterations for convergence.
- HBPROBETOL: Searches for a probe voltage at which the probe current is less than the specified value.
- HBMAXOSCITER: Specifies the maximum number of outer-loop iterations for oscillator analysis.

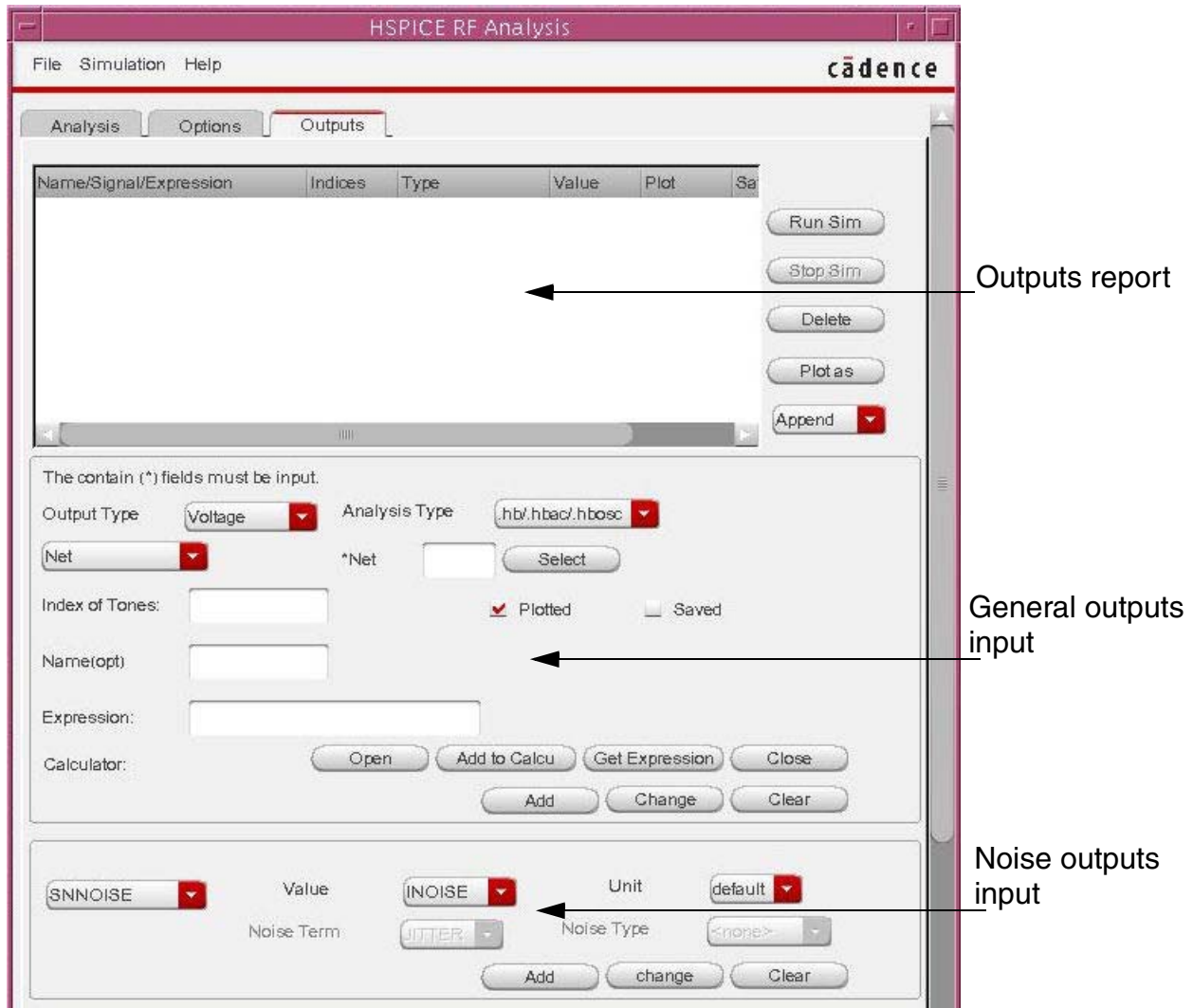
- BPNMATCHTOL: Determines the minimum required match between the NLP and PAC phase noise algorithms in HSPICE RF.
- PHASENOISEKRYLOVDIM: Specifies the dimension of the Krylov subspace that the Krylov solver uses.
- PHASENOISEKRYLOVITER: Specifies the maximum number of Krylov iterations that the phase noise Krylov solver takes.
- PHASENOISETOL: Specifies the error tolerance for the phase noise solver.
- PHASENOISELORENTZ: Turns on a Lorentzian model for the phase noise analysis.
- PHNOISEAMPM: Separates amplitude modulation and phase modulation components in a phase noise simulation.

Using the Outputs Tab

Selecting the **Outputs** tab displays a form which includes three sections: Outputs report, General outputs input, and Noise outputs input. Any field with an asterisk (*) must be input for a simulation to be run.

Chapter 10: HSPICE RF Analysis

Using the Outputs Tab



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 99 Outputs tab

Use the controls for the **Outputs** tab as follows:

1. Data in the general outputs input section adds or changes data to the outputs report field. The data you enter here is added to the netlist; the HSPICE RF simulation generates psf data information based on this input for the outputs to plot waveforms.
2. Use the **Select** button to open/activate and select a net, terminal, or instance from the schematic editor. The data is entered in both the inputs and report fields.

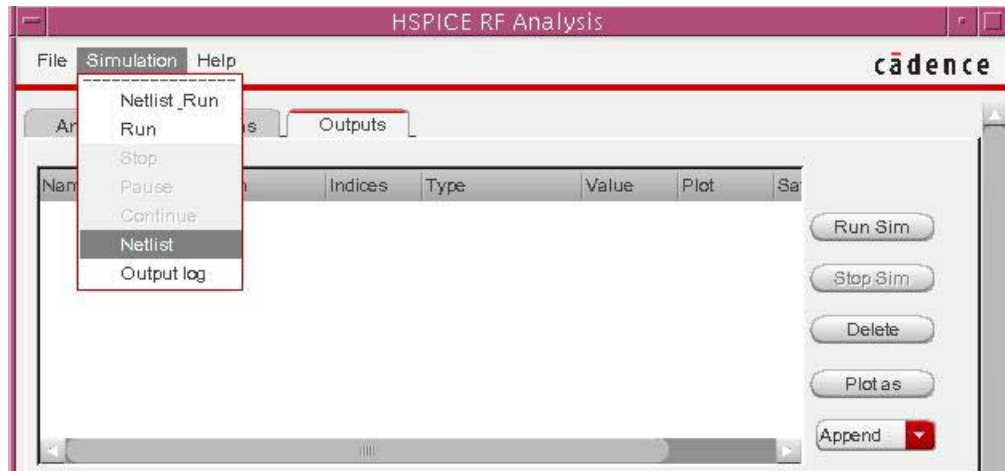
3. Use the Noise outputs input section to add or change Noise-specific outputs in the outputs report field. These input outputs are added to the netlist and the HSPICE RF Simulation generates psf data for the outputs to plot waveforms.
4. In the outputs report field select and data you wish to exclude and click **Delete** to permanently removed them form the simulation.
5. Use **Run Sim** or **Stop Sim** button to directly invoke or abort an HSPICE RF simulation.
6. Use the **Plot As** button and **Waveform** window mode to plot a waveform of outputs after successfully running an HSPICE RF simulation. The window options include: open a new graph window, append to existing window, or replace an existing graph window.
7. Enable the **Plotted** check box to automatically let HSPICE RF simulation generate output related psf date information and plot an output waveform after running a successful HSPICE RF simulation.
8. Use the **Save** check box to have the HSPICE RF simulation generate output related psf so that you can manually plot the output after running a successful HSPICE RF simulation.
9. Click the **Add to Calcu** button to add the input output to the Environment calculator tool; then import the edited output from the calculator by clicking the **Get Expression** button.
10. By selecting a data line in the outputs report section you can call its detailed information which is then displayed in the input section.

Running and Post-Processing an RF Simulation

After creating the settings listed in the previous section, call an HSPICE RF Analysis simulation by clicking the **Run Sim** button. Alternatively, you can select **Netlist** under the **Simulation** menu of the HSPICE RF Analysis form.

Note: Be sure you have enabled the simulation on the **Analysis** tab.

The HSPICE RF simulation log file is displayed after the simulation finishes; and the `hspicerf done` message in the output log file indicates a successful simulation.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 100 Simulation menu on the HSPICE RF Analysis form

Post-Simulation Plotting and Analysis

Use the **Outputs** tab to analyze your results:

After a successful HSPICE RF simulation:

1. Select the output signal or expression in the Outputs box.
2. Click the **Plot** button directly to plot its waveforms. (Remember to select the correct analysis type and plotting mode through the selection controls noted in the previous sections).
3. To redisplay output information, click the “output” item in the outputs report field, then click the **Add to Calcu** button to launch the Calculator tool and proceed to plot the waveform.

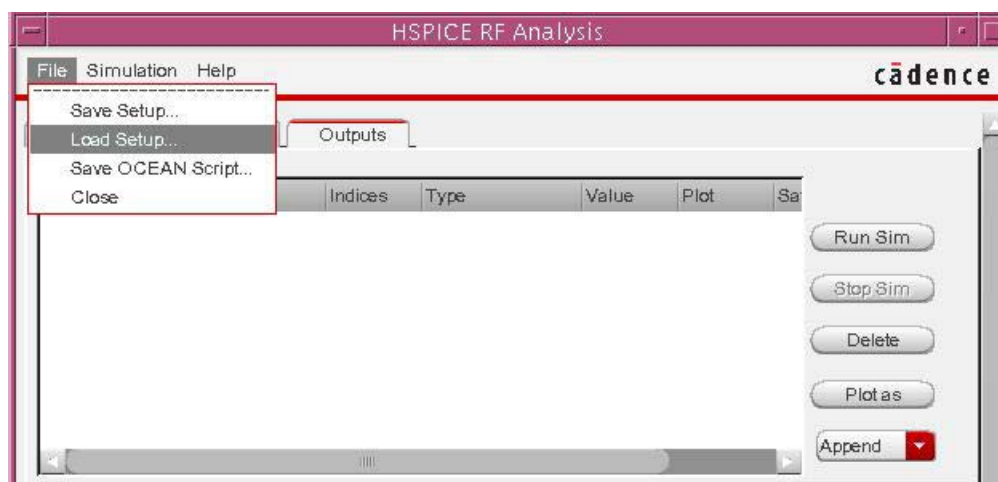
Using the Measure Utility on RF Simulations

The HSPICE Measure Utility supports measurements for harmonic balance-based and Shooting-Newton-based analyses, as well as PHASENOISE analysis. For details, see [Chapter 12, HSPICE Measurement Utility](#).

Other Features Supported in the HSPICE RF Analysis Form

You can do save state and load state operations from the HSPICE RF Analysis form.

Saving or loading session statistics: Use the HSPICE RF Analysis form to save and load all the user operations for simulation as a standard ADE simulation, and create an OCEAN script of based on all operations performed in an a Monte Carlo Analysis session.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

1. To Save/Load Statistics, under the File menu, select either the **Save Setup** or **Load Setup** option. Alternatively, select the **Save State** or **Load State** option in the Environment Console to perform the same operation.
2. Create an OCEAN script based on your RF Analysis session by either:
 - Selecting File >XXXXX and Saving all the operations as a single OCEAN script to do a post batch run simulation with RF analysis, or
 - Use the provided OCEAN API functions directly to write your own OCEAN script to do batch simulation. For detailed OCEAN API functions, please refer to the descriptions in [Appendix C, OCEAN API Functions for HSPICE RF Analysis](#).
 - Alternatively, use the **Help** button on the HSPICE RF Analysis form to assess an html user guide for the RF feature or the reference guide for OCEAN API functions

Current RF Analysis Feature Limitations

The HSPICE RF Analysis feature currently doesn't support the following HSPICE RF analyses:

- .HBLSP, .ENV, .ENVOSC
- .HBLIN, .HBXF, .SNFT, .SNXF, .PTDNOISE, .ENVFFT
- .ACPHASENOISE, .STATEYE
- In a frequency sweep analysis such as .HBAC, some sweep types are not supported in the current release. For example: sweepblock, data sweep: DATA=dataname.

HSPICE Optimization Analysis

Describes HSPICE Optimization capability in the HSPICE integration to the Cadence™ Virtuoso® Analog Design Environment.

For paths to corner analysis demonstration cases available with this release, see [HSPICE Integration to ADE Demonstration Examples](#).

These topics are discussed in the following sections.

- [Accessing and Setting Up an HSPICE Optimization Analysis](#)
- [Using the Setup Tab](#)
- [Using the Analysis Tab](#)
- [Other Features Supported in the HSPICE Optimization Form](#)
- [Current Optimization Analysis Feature Limitation](#)

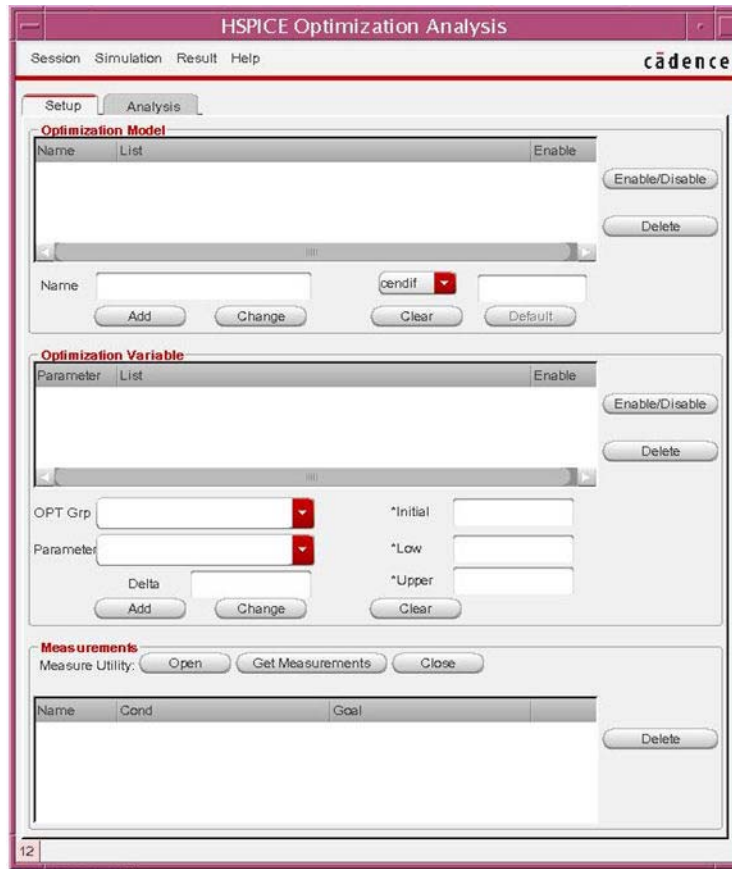
Accessing and Setting Up an HSPICE Optimization Analysis

HSPICE Optimization analysis provides HSPICE optimization methodologies. To invoke the HSPICE Optimization Analysis form, select the **Optimization** option under the Tools menu of the Environment Console.

Using the Setup Tab

Use the **Setup** tab to specify an HSPICE Optimization model, parameters, and measure statements. The form contains these panes:

- **Optimization Model**
- **Optimization Variable**
- **Measurements**



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 101 HSPICE Optimization Setup tab

Selecting the Model

On the **Setup** tab of the HSPICE Optimization Analysis form:

1. In the **Name** field enter the model name, including path.
2. Select the optimization model parameters one at a time and set each with a value. (See [.MODEL](#) in the *HSPICE Reference Manual: Commands and Control Options* for details on the optimization parameters. Click **Add** to add them into report.

3. You can modify the selected statement in the report by clicking **Change**.
4. Click **Delete** to delete one useless selected statement or **Enable/Disable** to disable/enable the selected statement.
5. Click **Clear** to remove the model from the UI.
6. When the **Default** button is enabled, click it to use the default parameter value.

Setting Optimization Variables

To set an optimization parameter:

1. Select or enter a user-defined optimize group name (**OPT Grp**). For example, “opt1” in the statement “.dc data=mos1 sweep optimize=opt1 results=verr1 model=optmod” is one optimization group.
2. **Select or enter a user-defined Parameter** name. For example, “wn1” in the statement “.param wn1=opt1(40,20,60)”. Enter values for **Initial**, **Low**, and **Upper** bounds (required).
3. Click **Add** to add the parameter to the report.
4. Click the **Enable/Disable** button to disable/enable the selected statement.
5. Optionally enter a **Delta** value.
6. Click **Delete** to remove a selected statement.
7. Click **Change** to modify the selected statement in the report.
8. Click **Clear** to remove all existing entries from the UI.

Setting Up Measurements

To launch the Measurement Utility:

1. In the **Measurements** pane click **Open** (see [Chapter 12, HSPICE Measurement Utility](#) for a detailed description of the utility).
2. Click **Get Measurements** to load all enabled measurement variables into the measurement report field.
3. Click **Close** to dismiss the HSPICE Measure Utility window.

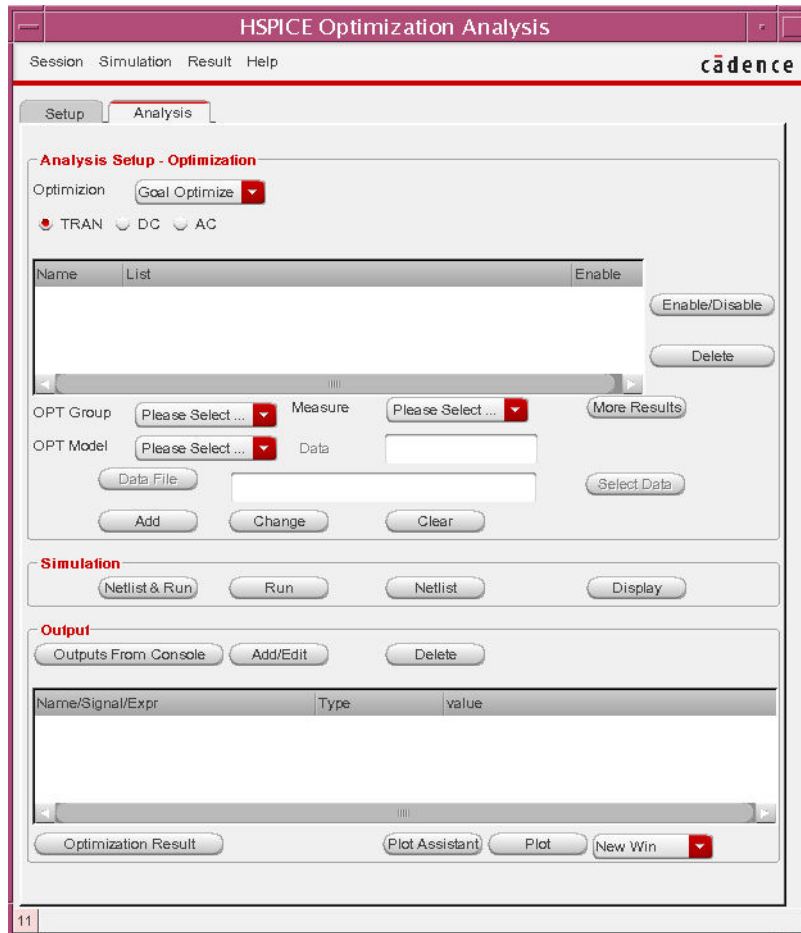
4. To omit a measurement from the netlist, select the measurement on the report field and click **Delete** (to the right of the measurement report field). All remaining measurement variables are netlisted when you do an HSPICE Monte Carlo simulation.

Using the Analysis Tab

Selecting the **Analysis** tab opens the HSPICE Optimization Analysis form ().

The form consists of these panes:

- **Analysis Setup - Optimization**
- **Simulation**
- **Output**



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 102 Analysis tab HSPICE Optimization

Choosing Analysis and Optimization Type

The form enables you to select the following:

1. **Optimization:** Select which type of optimization is to be performed from **Curve Fit**, **Goal Optimization**, **Bisection**, and **Passfail**.
2. Select the **OPT group** from optimization variable you named on the **Setup** tab earlier.
3. Select the **Measures** you set up on the **Setup** tab

Chapter 11: HSPICE Optimization Analysis

Other Features Supported in the HSPICE Optimization Form

4. Select the **OPT model** which you defined in the optimization model field on the **Setup** tab.
5. Click **Add** to add the Analysis setup into the report.
6. Click **Delete** to delete a selected statement.
7. Click **Enable/Disable** to disable/enable the selected statement.
8. Click **More Results** to choose more results.
9. Click **Data Files** to load the data file. Use the **Select Data** to choose one data to be included in the **Data** field.
10. Click **Change** to enable modification of the selected setup.
11. Click **Clear** to remove all entries on the Analysis UI.

Other Features Supported in the HSPICE Optimization Form

You can do save state and load state operations from the HSPICE Optimization Analysis form.

Saving or loading session statistics: Use the HSPICE Optimization Analysis form to save and load all the user operations for simulation as a standard ADE simulation, and create an OCEAN script of based on all operations performed in an Optimization Analysis session.

1. To Save/Load Statistics, under the **Session** menu, select either the **Save Setup** or **Load Setup** option. Alternatively, select the **Save State** or **Load State** option in the Environment Console to perform the same operation.
2. Create an OCEAN script based on your Optimization Analysis session by either:
 - Selecting File >XXXXX and Saving all the operations as a single OCEAN script to do a post batch run simulation with optimization analysis, or
 - Use the provided OCEAN API functions directly to write your own OCEAN script to do batch simulation. For detailed OCEAN API functions, please refer to the descriptions in TBD.

- Alternatively, use the **Help** button on the HSPICE RF Analysis form to assess an html user guide for the RF feature or the reference guide for OCEAN API functions

Current Optimization Analysis Feature Limitation

The current version only supports DC, AC, and TRAN analysis for optimization.

Chapter 11: HSPICE Optimization Analysis

Current Optimization Analysis Feature Limitation

HSPICE Measurement Utility

Describes the HSPICE Measurement Utility in the HSPICE integration to the Cadence™ Virtuoso® Analog Design Environment.

The following sections discuss these topics:

- [Measurement Utility Overview](#)
- [Opening and Using the HSPICE Measurement Utility](#)
- [Trig/Targ Setup](#)
- [General Functions Setup](#)
- [Find/When Setup](#)
- [Error Function Setup](#)
- [Optimization Syntax—Goal Setup](#)
- [Report View Setup](#)

Measurement Utility Overview

The HSPICE measurement utility enables you to modify information and define the results of successive HSPICE simulations. This utility helps to set up user-defined electrical specifications of circuits. It supports all the functions and features that the .MEASURE command in HSPICE does. The Measure Utility can be combined with several tools such as Corners, Optimization, and Monte Carlo. It can be also used independently.

The utility includes the Trig/Targ, General functions, Find/When, Equation, Error, Goal, and Special Measurement setup pages. The following describes the utility's forms and setups.

Chapter 12: HSPICE Measurement Utility

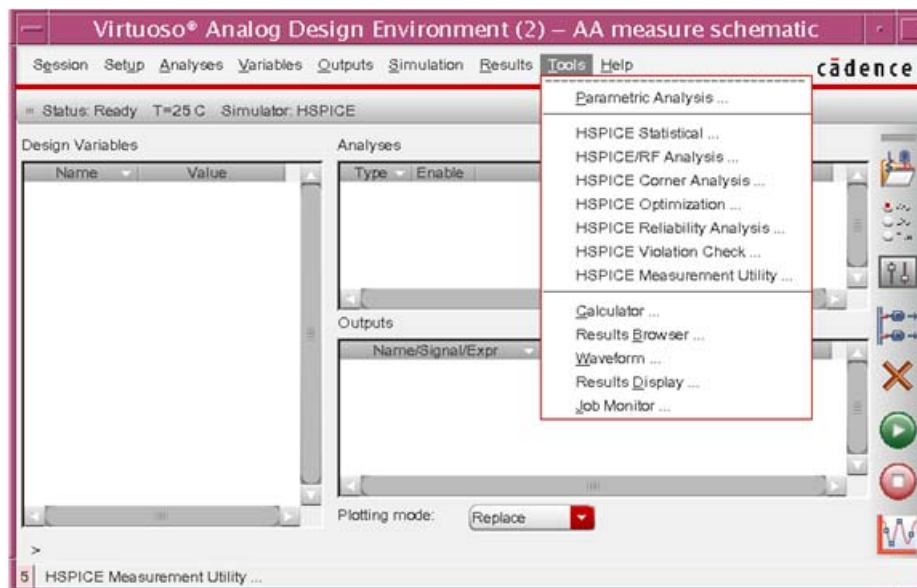
Opening and Using the HSPICE Measurement Utility

For detailed information on the measurements functionality in HSPICE, see [Specifying User-Defined Analysis \(.MEASURE\)](#) in the *HSPICE User Guide: Simulation and Analysis* and extensive command line syntax coverage for the [.MEASURE \(or\) .MEAS](#) command in the *HSPICE Reference Manual: Commands and Control Options*.

Opening and Using the HSPICE Measurement Utility

To open and begin using the Measure Utility:

On the Environment Console menu bar, select **Tools > Measurement Utility**.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 103 Tools menu—HSPICE Measurement Utility

The main tab of the utility window opens without checking out any additional licenses. The main page of the HSPICE Measure Utility opens as shown in [\(Figure 104 on page 259\)](#).

Chapter 12: HSPICE Measurement Utility

Opening and Using the HSPICE Measurement Utility

HSPICE Measure Utility

Measure Name Clear Precision 4

Measure Analysis TRAN

Trig/Targ Functions Find/When Equation Error Special - Meas.

Trigger

☒ Trigger Signal ☐ At Time

Select Type Signal Schematic = Value

Select Type = Last TD

Target

☒ Target Signal ☐ At Time

Select Type Signal Schematic = Value

Select Type = Last TD

Optimization Syntax

Goal Min Val Weight

Name	Analysis	Enable	Result
------	----------	--------	--------

Add Change Delete Enable Calculate Summary

Close Help

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 104 Measurement Utility

The utility form consists of these sections:

1. **Measure Name, Precision** (optional, Default: 4), a button to **Clear** all entries on the form, and selectable analysis that the measurement requires from a list of TRAN, DC, and AC HSPICE analyses, plus RF analyses using harmonic balance-based and Shooting-Newton-based algorithms ([Figure 105 on page 260](#)).

Chapter 12: HSPICE Measurement Utility

Opening and Using the HSPICE Measurement Utility

The figure displays three stacked instances of the HSPICE Measure Utility form. Each instance contains a 'Measure Name' text field, a 'Clear' button, a 'Precision' dropdown menu (set to 3), and a 'Measure Analysis' dropdown menu. The top instance has 'TRAN' selected. The middle instance has 'HB' selected, and an 'HBTRAN' checkbox is visible and unchecked. The bottom instance has 'SNOSC' selected, and an 'SNFD' checkbox is visible and unchecked.

Figure 105 Upper section Measure Utility form; lower two show HSPICE RF analyses

- **Measure Name:** Input a string as the measure name in the measurement statement
 - **Measure Analysis:** Select an analysis for the current measure. If **HB** is selected for analysis then the **HBTRAN** check box is also added. If this control is checked, then the HB time domain will be measured. Similarly, if **SNOSC** is selected, the window is updated with an additional **SNFD** check box. If this control is checked, then the SNOSC frequency domain will be measured.
 - **Precision:** Select a value for the “measdgt” option (output of significant digits can be 1-10; Figure 105 shows the **Precision** setting changed to 3).
 - **Clear:** Removes all the setups in the Measure Utility.
2. Measurement function details
Type of measurement functions include:
 - Trigger-Target
 - Function
 - Find/When
 - Equation
 - Error
 - Special-Meas.
 3. Optimization Syntax for measure commands and arguments
 4. Utility controls and the measurement report view

Trig/Targ Setup

The **Trig/Targ** setup panel (Figure 106) is divided into two sections, **Trigger** and **Target**.

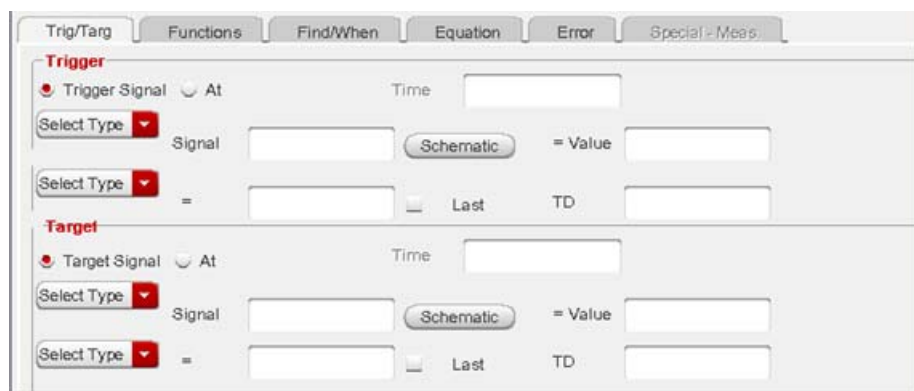


Figure 106 Trig/Targ setup pane

The form displays the following options:

1. **Trigger Signal** or **At**: Select **At** if your measurement is a special case, otherwise select **Trigger Signal**. If you select **At**, input the value in the empty field to the right. If **TRAN** analysis is selected in the **Measure Analysis** selection box, the empty field to the right is for **Time**; for **AC** analysis, the field is for **Frequency**, for **DC** analysis, it is for **Parameter**. If you do not select **At**, continue to set up the following:
2. Select the trigger signal type and input the signal. You can use the **Schematic** button to choose the signal name from schematic. After that, the signal value must be specified in the right field.
3. Select the trigger event type. If it is the last event, click the **Last** check box, otherwise input the value in the field to the left of the **Last** check box.
4. Optionally, input the time delay (**TD**) before HSPICE enable the measures.

Target controls are identical to **Trigger** controls.

General Functions Setup

The Function tab ([Figure 107](#)) provides the general function setup under the measure.

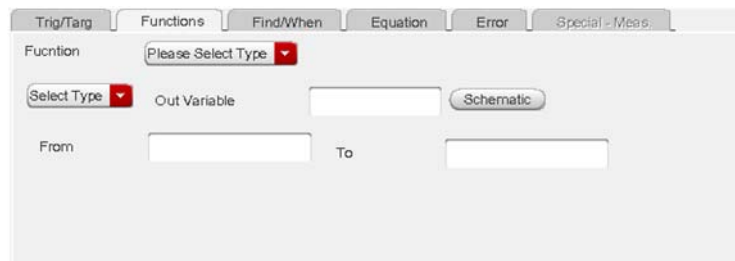


Figure 107 Functions tab

The **Functions** pane includes a list box of functions.

1. Select the out variable type and input the variable. You can use the **Schematic** button to choose the signal name from the schematic editor.
2. Optionally, input the initial and end value of the variable that is measured.
3. If the selected function type is **DERIV**, you need to set up the derivative specification as you would the “Trigger” specification ([Figure 108](#)).

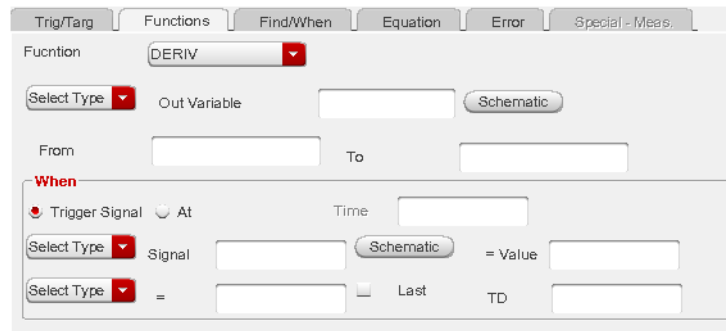


Figure 108 DERIV function with Trigger Signal setup requirements

Find/When Setup

The **Find/When** tab (Figure 109) displays controls and parameters as noted.

The figure shows two screenshots of the HSPICE Measurement Utility interface. The top screenshot shows the 'Find/When' tab with radio buttons for 'When', 'Find/When', and 'Find/At'. The 'Find/When' radio button is selected. There are input fields for 'Time', 'Variable', and 'Signal', each with a 'Select Type' dropdown and a 'Schematic' button. There is also a 'Last' button and a 'TD' input field. The bottom screenshot shows the 'Find/When' tab with the 'Find/At' radio button selected. There are input fields for 'Freq.', 'Variable', and 'Signal', each with a 'Select Type' dropdown and a 'Schematic' button. There is also a 'Last' button and a 'TD' input field. The 'Measure Name' field is empty, and the 'Measure Analysis' dropdown is set to 'FFT'.

Figure 109 Find/When setup form (top) and when doing an FFT measurement (bottom)

Use the following steps:

1. Select the measure type.
 - If selecting **Find/At**, complete Steps 2 and 3.
 - If selecting **Find/When**, complete Step 3, 4, 5, and 6.
 - If selecting **When**, complete Steps 4, 5, and 6.
2. Input the value on the right blank text field. If **TRAN** analysis is selected in the **Measure Analysis** selection box, the right blank is for **Time**; for **AC** analysis, the right blank is for frequency (**Freq**), for **DC** analysis, it is for **Parameter**.
3. Select the variable type and input the variable name. You can choose to use the **Schematic** button to choose the variable name from the design schematic editor.

Chapter 12: HSPICE Measurement Utility

Opening and Using the HSPICE Measurement Utility

4. Select the signal type and input the signal. You can use the **Schematic** button to choose the signal name from the design in the schematic editor. After that, the signal value must be specified on the right blank field.
5. Select the event type. If it is the last event, click the **Last** check box, otherwise input the value in the field to the left of the **Last** check box.
6. Optionally, input the time delay (**TD**) before HSPICE enable the measures.
7. If you select **FFT** in the **Measure Analysis** selection box, then the **Find/When** tabbed section changes to the lower section of [Figure 109 on page 263](#). You only need to set the **Freq** value and select the signal type. The signal is reserved during FFT setup.

Equation Setup

Use the Equation setup ([Figure 110](#)) for measurement to specify the HSPICE parameter measure statements directly into the text field.

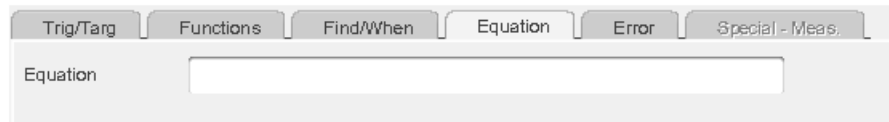


Figure 110 Equation setup pane

Error Function Setup

The **Error** setup ([Figure 111](#)) enables the report of the relative difference between two output variables.

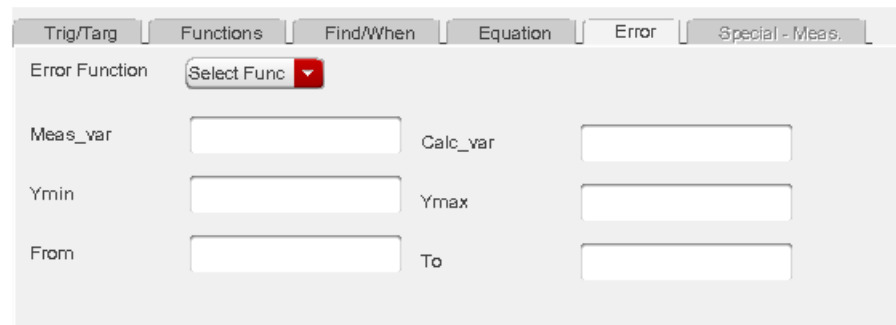


Figure 111 Error setup pane

The Error setup controls are as follows:

1. Select the Error function (**ERR**, **ERR1**, **ERR2**, **ERR3**). See [ERROR Function](#) in the *HSPICE User Guide: Simulation and Analysis* for a discussion of the equations used to calculate the relative and absolute relative error values.
2. Input the **Meas_var** (name of any output variable or parameter in the data) and **Calc_var** variable (name of the simulated output variable or parameter in the .MEASURE command).
3. Select the error measure profile type and input the value. The default values are Ymin=1.0e-12 and Ymax =1.0e+15.
4. Input the begin value and end value of error calculation using the **From** and **To** fields.

Using the Special-Meas. Tab

Use the **Special-Meas.** tab to specify HSPICE special measure statements. ([Figure 112 on page 266](#)). Currently, only FFT and PHASENOISE analyses use this tab.

Measuring Full Spectrum (FFT) Analysis Results

With **FFT** as the selected as the **Measure Analysis**, the **Special-Meas.** tab is enabled and selections can be made to measure: total harmonic distortion (**THD**), signal to noise ratio (**SNR**), signal to noise-plus-distortion ratio (**SNDR**), effective number of bits (**ENOB**), spurious free dynamic range (**SFDR**), and to specify the harmonic up to which to carry out the measurement (**nbharm**).

For details on HSPICE measurement of FFT results see [.MEASURE FFT](#) in the *HSPICE Reference Manual: Commands and Control Options*.

Chapter 12: HSPICE Measurement Utility

Opening and Using the HSPICE Measurement Utility

The screenshot shows the 'Special - Meas.' tab of the HSPICE Measurement Utility. At the top, there is a 'Measure Name' text box, a 'Clear' button, and a 'Precision' dropdown menu set to '3'. Below this is a 'Measure Analysis' dropdown menu set to 'FFT'. A row of tabs includes 'Trip/Targ', 'Functions', 'Find/When', 'Equation', 'Error', and 'Special - Meas.'. Under the 'Special - Meas.' tab, there are five radio buttons: 'THD' (selected), 'SNR', 'SNDR', 'ENOB', and 'SFDR'. Below the radio buttons is a text box labeled 'nbharm'.

Figure 112 Special-Measurements form for FFT

Spectrum Analysis Measurements

When FFT is specified as the Measure Analysis, the UI controls are displayed according to the following function settings:

1. **THD** is the default setting with the **nbharm** field empty. Optionally, set a value for this harmonic (the default is the highest harmonic in the FFT result).
2. If you select the function to be **SNR**, **SNDR**, or **ENOB**, the setup form is updated as in [Figure 113](#). Input values for **nbharm**, **binsiz**, and **maxfreq**, as necessary.

The screenshot shows the 'Special - Meas.' tab of the HSPICE Measurement Utility. At the top, there is a 'Measure Name' text box, a 'Clear' button, and a 'Precision' dropdown menu set to '3'. Below this is a 'Measure Analysis' dropdown menu set to 'FFT'. A row of tabs includes 'Trip/Targ', 'Functions', 'Find/When', 'Equation', 'Error', and 'Special - Meas.'. Under the 'Special - Meas.' tab, there are five radio buttons: 'THD', 'SNR' (selected), 'SNDR', 'ENOB', and 'SFDR'. Below the radio buttons are three text boxes: 'nbharm', 'binsiz', and 'maxfreq'.

Figure 113 Setup form for measurement of SNR in FFT result

3. If you select the function **SFDR**, the setup form is refreshed to [Figure 114](#). Input values for **maxfreq** and **minfreq**, as necessary.

Figure 114 Setup form for measurement of SNDR in FFT result

Measuring PHASENOISE Analysis Results

Selecting **PHASENOISE** as the **Measurement Analysis** dynamically refreshes the **Special-Meas.** form according to the functions being measured.

For details on HSPICE measurement of PHASENOISE analysis results see [.MEASURE PHASENOISE](#) in the *HSPICE Reference Manual: Commands and Control Options*.

When **PHASENOISE** is specified as the **Measure Analysis**, the UI controls are displayed with **PERJITTER** as the default function.

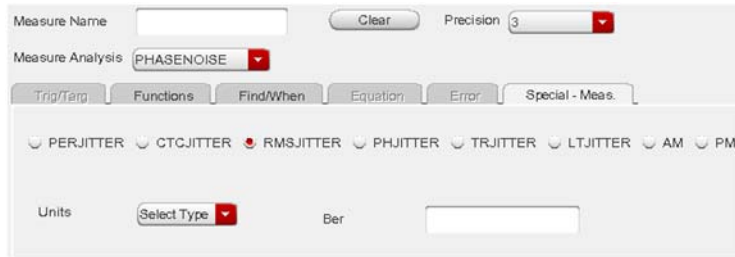
1. Optionally, set **From**, **To**, **Units**, and bit error rate (**Ber**) parameters, as necessary. These parameters are the same on the form when you select **CTCJITTER** as the function.

Figure 115 Form opens with default function set as PERJITTER

2. Setting the function to **RMSJITTER**, **PHJITTER**, **LTJITTER**, or **TRJITTER** refreshes the **Special-Meas.** form as shown in [Figure 116](#). Select unit type and input bit error rate, if necessary.

Chapter 12: HSPICE Measurement Utility

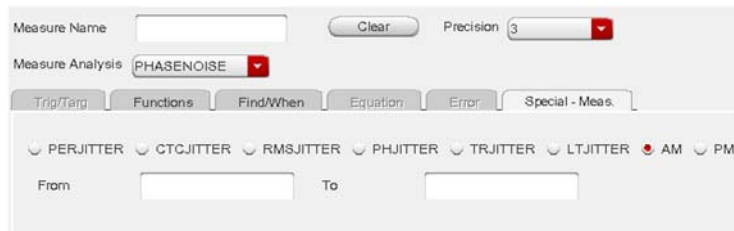
Opening and Using the HSPICE Measurement Utility



The screenshot shows the HSPICE Measurement Utility dialog box. At the top, there is a 'Measure Name' field, a 'Clear' button, and a 'Precision' dropdown set to 3. Below this is the 'Measure Analysis' dropdown, which is set to 'PHASENOISE'. A row of tabs includes 'Trip/Targ', 'Functions', 'Find/When', 'Equation', 'Error', and 'Special - Meas.'. Under the 'Functions' tab, several radio buttons are visible: 'PERJITTER', 'CTCJITTER', 'RMSJITTER' (which is selected), 'PHJITTER', 'TRJITTER', 'LTJITTER', 'AM', and 'PM'. At the bottom, there is a 'Units' dropdown set to 'Select Type' and a 'Ber' field.

Figure 116 RMSJITTER function selected

3. If the function is set to amplitude modulation noise (**AM**) or phase modulation noise (**PM**), the setup form refreshes as shown in [Figure 117](#). Enter the range, if required.

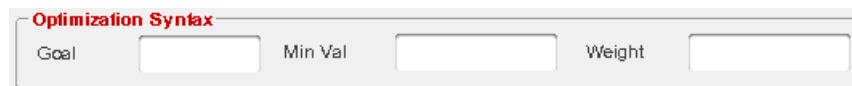


This screenshot shows the same dialog box as Figure 116, but with the 'AM' radio button selected instead of 'RMSJITTER'. The 'From' and 'To' input fields are now visible below the radio buttons, indicating that a range must be specified for this function.

Figure 117 AM function selected

Optimization Syntax—Goal Setup

The Goal setup enables the reporting of the relative difference between two output variables. The **Optimization Syntax** pane is shown in [Figure 118](#).



The screenshot shows the 'Optimization Syntax' pane. It contains three input fields labeled 'Goal', 'Min Val', and 'Weight', each with a corresponding text label to its left.

Figure 118 Goal setup pane

The fields for user input are:

1. **Goal:** Input the optimum value
2. **Minval:** Input the lowest acceptable value
3. **Weight:** Input the value of the weight parameter

Report View Setup

The report view setup pane (Figure 119) enables reporting of the relative difference between two output variables.

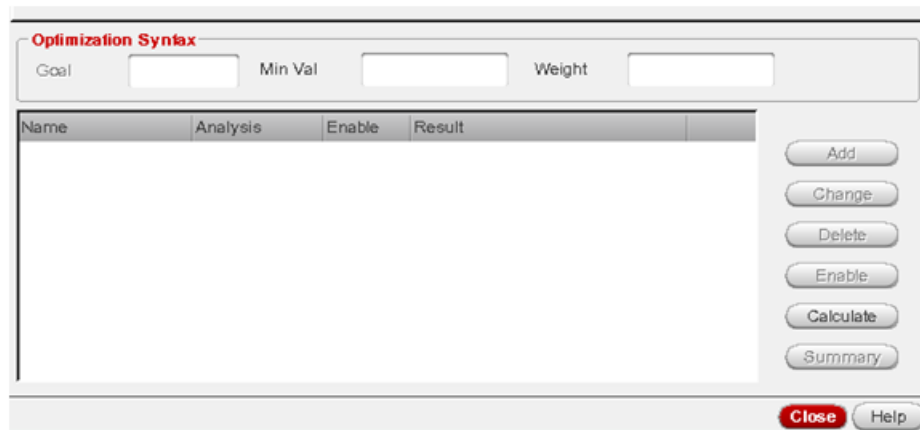


Figure 119 Report view setup pane

When measure statements are created, they appear in the measurement report view field. The buttons include the following:

1. **Add:** Use to add each setup created in the form above into the report view.
2. **Change:** Allows you to edit a selected statement.
3. **Delete:** Removes the selected measure statement.
4. **Enable/Disable:** Enables or Disables the selected statement.
5. **Calculate:** Click to display the measure result in the report view.
6. **Summary:** Summarizes the selected measure result and displays it in a separate window.

Chapter 12: HSPICE Measurement Utility

Opening and Using the HSPICE Measurement Utility

HSPICE Reliability Analysis (MOSRA)

Describes the HSPICE Reliability Analysis (MOSRA) feature in the HSPICE integration to the Cadence™ Virtuoso® Analog Design Environment.

HSPICE reliability analysis allows circuit designers to predict the reliability of their design to allow enough margin for their circuits to function correctly over their lifetimes.

A unified custom reliability modeling MOSRA API is available. Contact your Synopsys technical support team for more information.

For paths to reliability analysis demonstration cases available with this release, see [HSPICE Integration to ADE Demonstration Examples](#).

The following sections discuss these topics:

- [MOSRA Overview](#)
- [Launching the HSPICE Reliability Analysis Window](#)
- [Setting Up Reliability Analysis](#)
- [Setting up MOSRA-Related Options](#)
- [Setting up the Outputs Page](#)
- [Other Features Supported in the HSPICE Reliability Analysis Window](#)

MOSRA Overview

MOSFET reliability analysis (MOSRA) is an HSPICE advanced feature to simulate MOSFET lifetime and reliability after an extended period of service. This feature is essential in maintaining the long-term reliability of these devices for the highest quality IC designs. MOSRA is also available in

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Launching the HSPICE Reliability Analysis Window

Synopsys HSIM simulator and is one of the HSPICE differentiated features from other competitors.

The HSPICE Reliability Analysis window in the HSPICE Interface to the Cadence™ Virtuoso® Analog Design Environment allows easy access to HSPICE Reliability analysis capabilities without additional license control. For full discussion of the MOSRA functionality contact your Synopsys Technical Support team for access to the *HSPICE User Guide: MOSRA API Implementation*; an overview of the MOSRA functionality is available in [MOSFET Model Reliability Analysis \(MOSRA\)](#), Chapter 27 of the *HSPICE User Guide: Simulation and Analysis*.

Launching the HSPICE Reliability Analysis Window

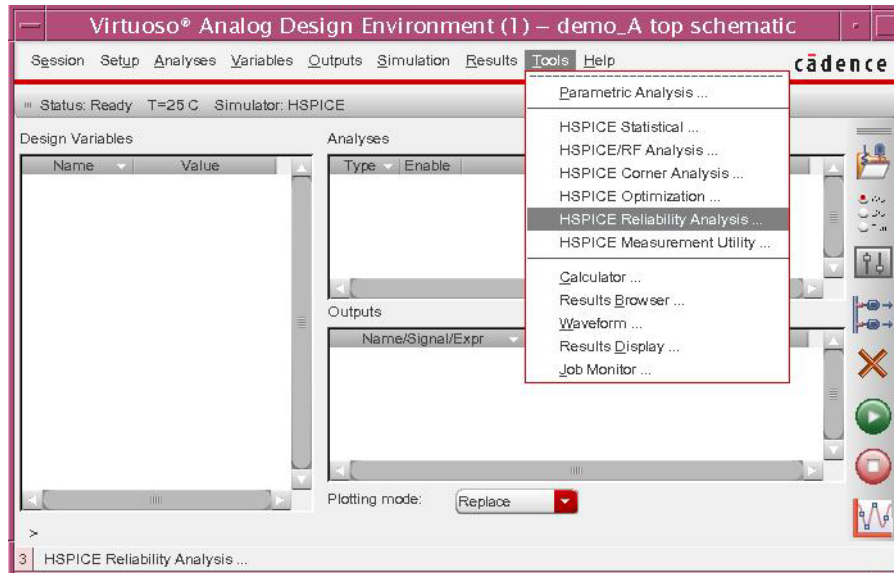
Note: First install HSPICE ADE interface and configure the 'cdsinit' file to load the HSPICE context file. After invoking ADE, check for the current configured simulator. If the default simulator is not HSPICE, you must change it to HSPICE for this feature.

To open and begin using the Measure Utility:

On the Environment Console menu bar, select **Tools > HSPICE Reliability Analysis**.

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Launching the HSPICE Reliability Analysis Window



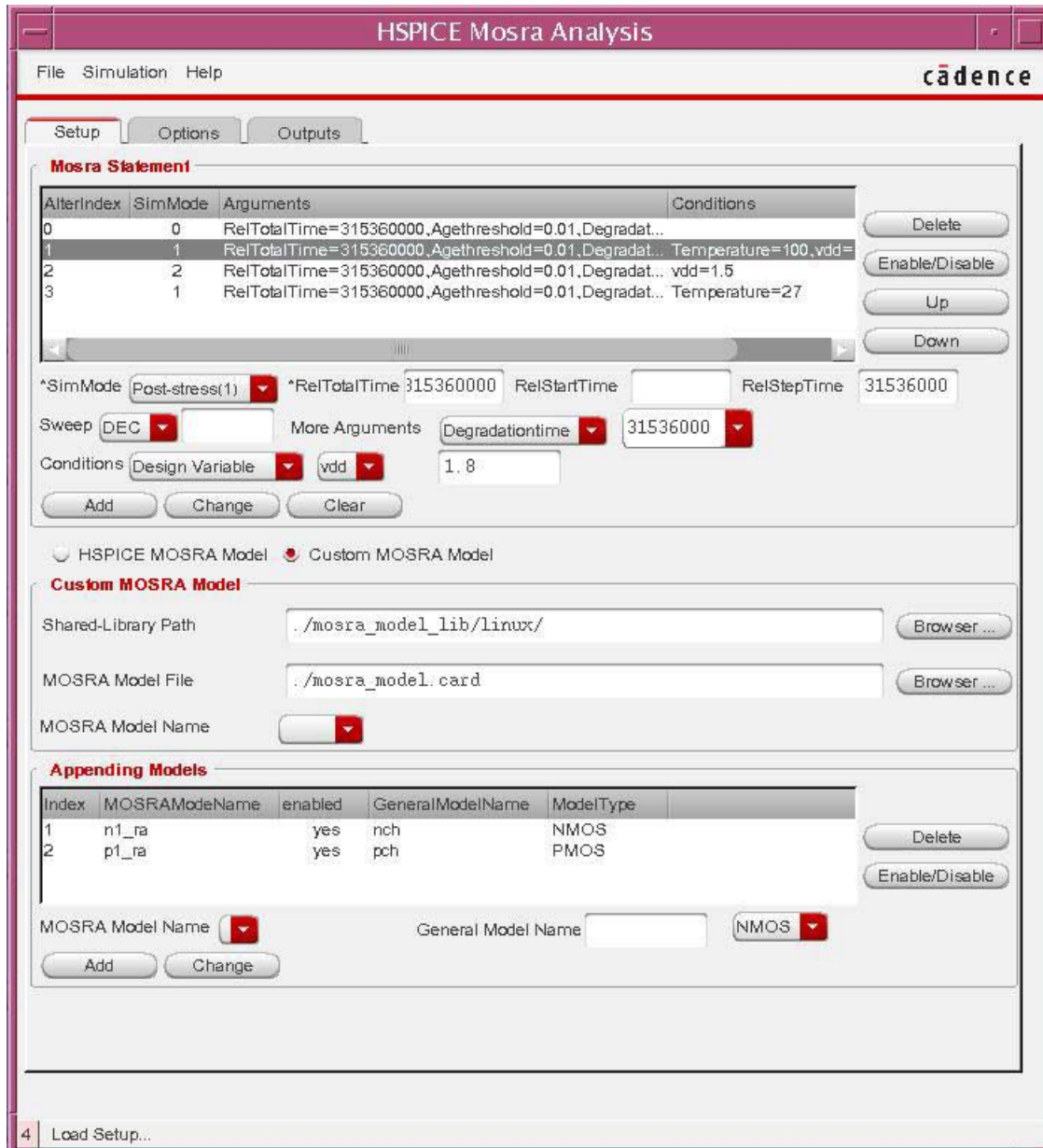
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 120 Environment Console Tools menu with Reliability Analysis selected

The HSPICE Mosra Analysis window opens ([Figure 121](#)).

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Launching the HSPICE Reliability Analysis Window



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 121 MOSRA Analysis main window showing Setup page

Main Menu

The three menus on the HSPICE Mosra Analysis window are **Session**, **Simulation**, and **Help**.

Session menu options include the following. The first two options invoke the **Save/Load States** command in the Environment console.

- **Save Setup** — Saves the user setup states generated in the Mosra Analysis window
- **Load Setup** — Loads the user setup states previously saved.
- **Save Script** — Saves all user operations into one OCEAN script file for use in non-graphic mode.
- **Quit** — Quits and closes the MOSRA utility window

The **Simulation** menu replicates the functionality of the **Simulation** section of the **Outputs** page.

- **Netlist and Run** — Netlists current analysis setup and invokes HSPICE to run simulation.
- **Netlist** — Creates a netlist only and display the file.
- **Run** — Invokes an HSPICE simulation with existing netlisted result file.
- **Stop** — Aborts an HSPICE simulation of it is running.
- **Log** — Opens the simulation log file.

Use the **Help** menu to access:

- **OCEAN APIs** — Provides OCEAN API functions to do batch analysis in non-GUI mode.
- **HSPICE Reliability Analysis User Guide** — Displays the manual for HSPICE MOSRA analysis.

Tabs and Pages

The HSPICE Mosra Analysis window consists of three tab pages: **Setup**, **Options**, and **Outputs**.

Setup Page

The **Setup** page displays three setup sections: **Mosra Statement**, **MOSRA Model**, and **Appending Models** (APPENDMODEL commands setup section).

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Launching the HSPICE Reliability Analysis Window

The **MOSRA model** setup section provides two options: **HSPICE MOSRA models** (built-in, default models) and **Custom MOSRA models** (user-supplied).

Options Page

The **Options** page (Figure 122 on page 276) lists all MOSRA analysis-related command options used for running an HSPICE simulation.

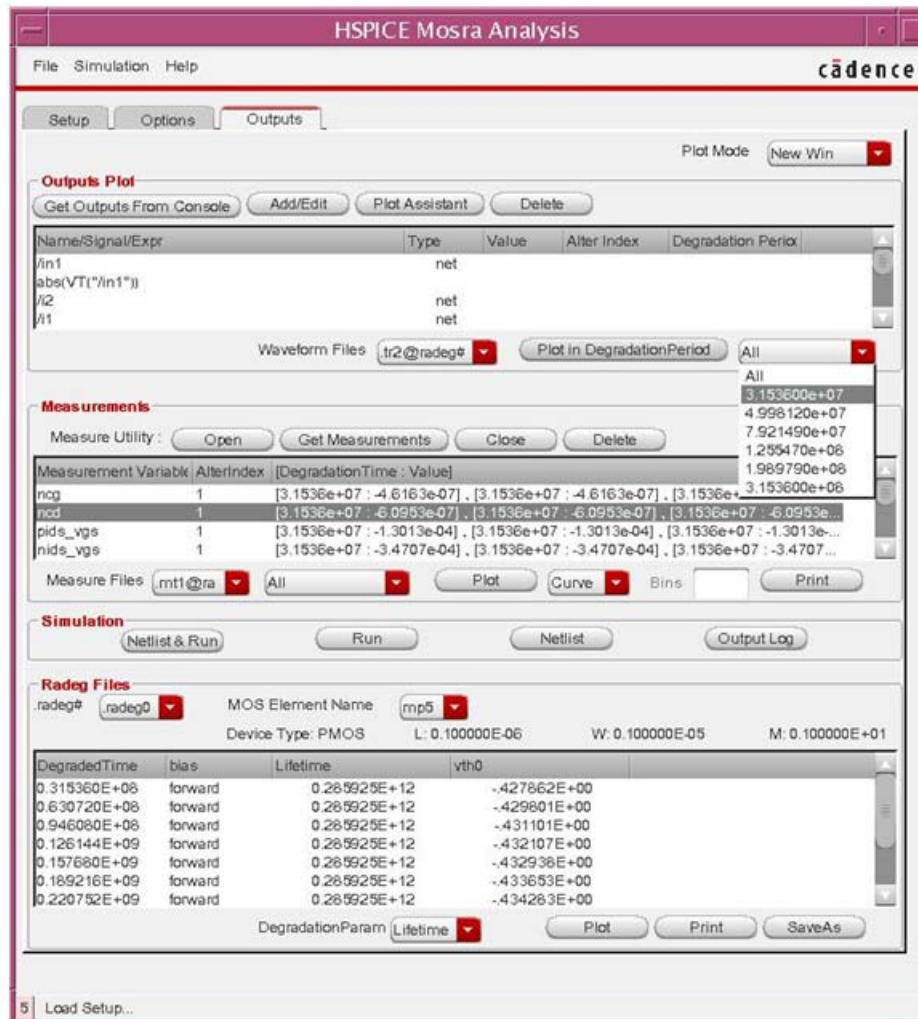
Name	Value	Default status
APPENDALL	0	Cdsenv
DEGFN	100m	Default
DEGFP	100m	Default
MOSRALIFE	nolife	Cdsenv
MOSRASORT	delvth0	Cdsenv
MRAEXT	<input type="checkbox"/>	Default
MRAPAGED	<input type="checkbox"/>	Default
RADEGOUTPUT	off (0)	Default

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 122 Options page

Output Page

The **Outputs** page displays four sections: **Outputs Plot**, **Measurements** (measure results), **Simulation**, and **Radeg Files** information. Use these sections to plot analysis outputs, measure and print results, netlist and run simulations, and view radeg (Reliability Analysis Degradation) file parameters information in a table format.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 123 Outputs page

Setting Up Reliability Analysis

The HSPICE Reliability Analysis setup includes:

- Invoking MOSRA commands, models, and APPENDMODEL commands
- Specifying options and their values
- Adding measure variables

MOSRA Command Setup

The MOSRA commands group of controls enables you to specify the .MOSRA commands for the MOSRA analysis. It also allows you to specify different simulation conditions for pre-stress or post-stress simulation. The **Mosra Statement** section changes dynamically depending on which of the **Conditions** is selected.

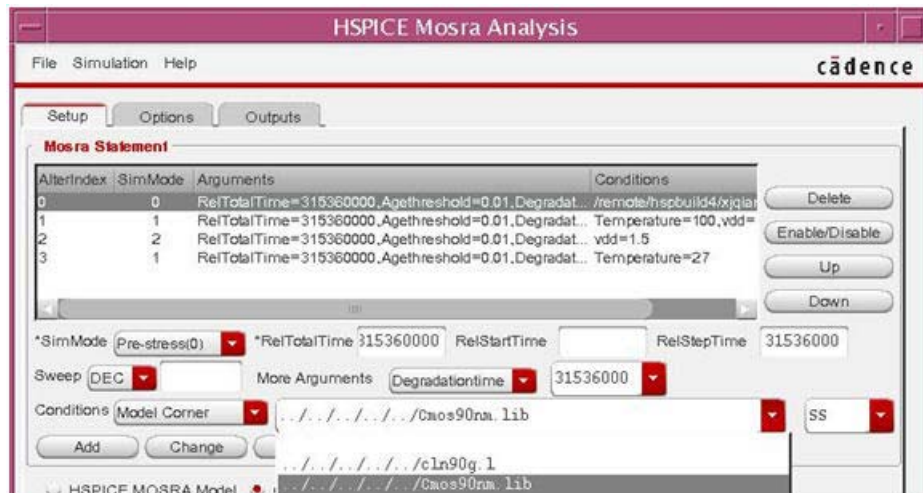


Figure 124 Mosra Statement section

The section contains the following fields and controls:

- **Mosra Statement** report field — Displays the MOSRA setup commands.
- **SimMode** — Select from the following simulation flows: pre-stress, post-stress, both flows (default), or the simulation mode to support battery degradation in a stress calculation.
 - **Pre-stress(0)** — Select pre-stress simulation only.
 - **Post-stress(1)** — Select post-stress simulation only.
 - **Default-both(2)** — Select pre- and post-stress simulation.
 - **Cont-deg(3)** — Select continual degradation integration through alters.

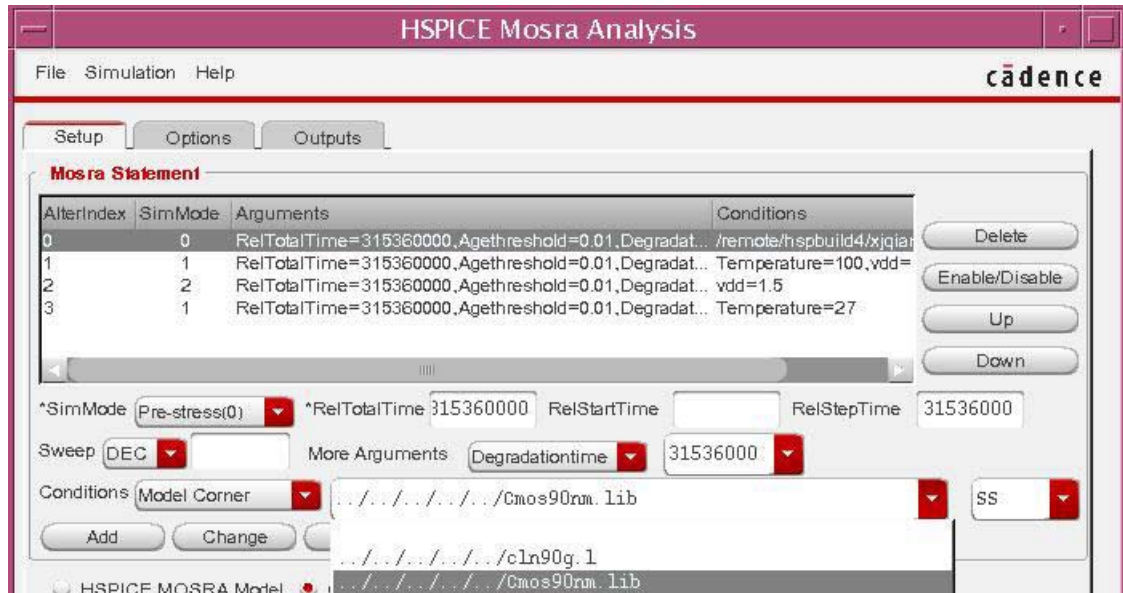
Note: For HSPICE, when SimMode=3, if .option radegfile is *not* specified in the top level netlist, the simulation starts from a fresh device. If .option radegfile is specified in the top level netlist, HSPICE reads in the last suite degradation in the radeg file and

continues the degradation integration and extrapolation from the corresponding circuit time in the *radeg* file. In consecutive alters, the *radeg* generated from the previous alter run is read in.

- **Conditions** — Select from the following choices:
 - **Temperature**
 - **Design Variables** — This field can be automatically populated according to definitions in the Environment console for different pre-stress or post-stress simulations; the values of the variables can be changed individually here.
 - **Model Corner** — Use different model parameter files for pre-stress and post-stress simulation. When selected, the section displays a **Model File** select field and a **Model Section** field ([Figure 125](#)). You can use these two fields to define models for each MOSRA simulation. The choices in the **Model File** select field come from the settings in Environment console. Select **Setup > Model Files** in the Environment console to open the Model Library Setup form ([Figure 126 on page 280](#)). Locate the model **Section** for the **Model Files** in this form to populate the Mosra Analysis **Model Corner** field.

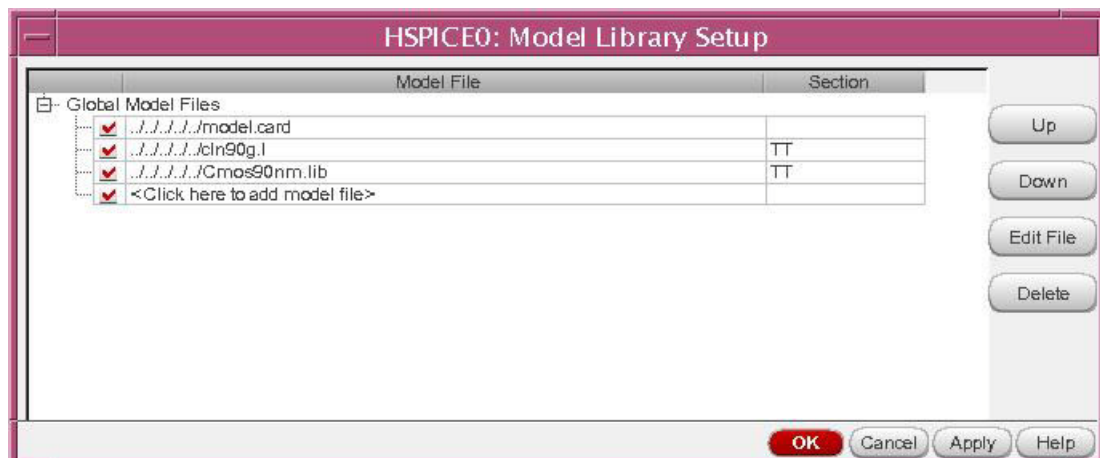
Chapter 13: HSPICE Reliability Analysis (MOSRA)

Setting Up Reliability Analysis



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 125 Model Corner selected, model files populated from Model Library Setup form



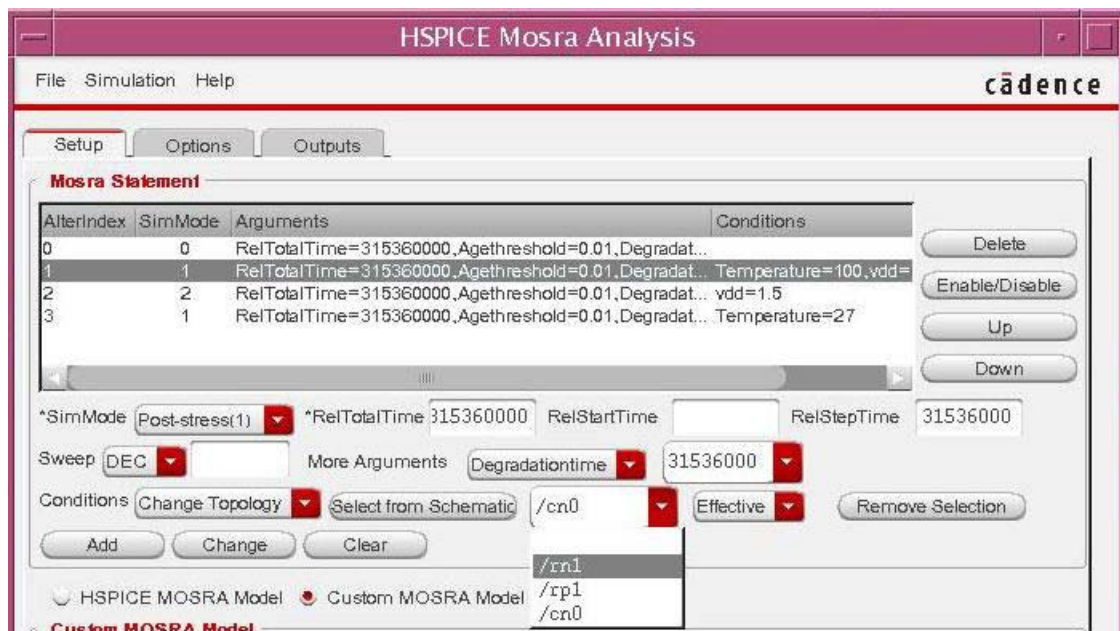
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 126 Model Library Setup form

- **Change Topology** — Changes circuit topology between different simulation modes. **Change Topology** (Figure 127 on page 281) displays the **Select From Schematic** button, **Topology Elem** select

field, **Status** field, and **Remove Selection** button.

- **Select From Schematic** selects a level instance from the schematic and adds to the element choices of the **Topology Elem** select field.
- The **Status** field controls whether the selected instance is **Effective** (added to the netlist) or **Discarded** in (will not be netlisted).
- **Remove Selection** returns the selected instance to normal circuit netlisting.



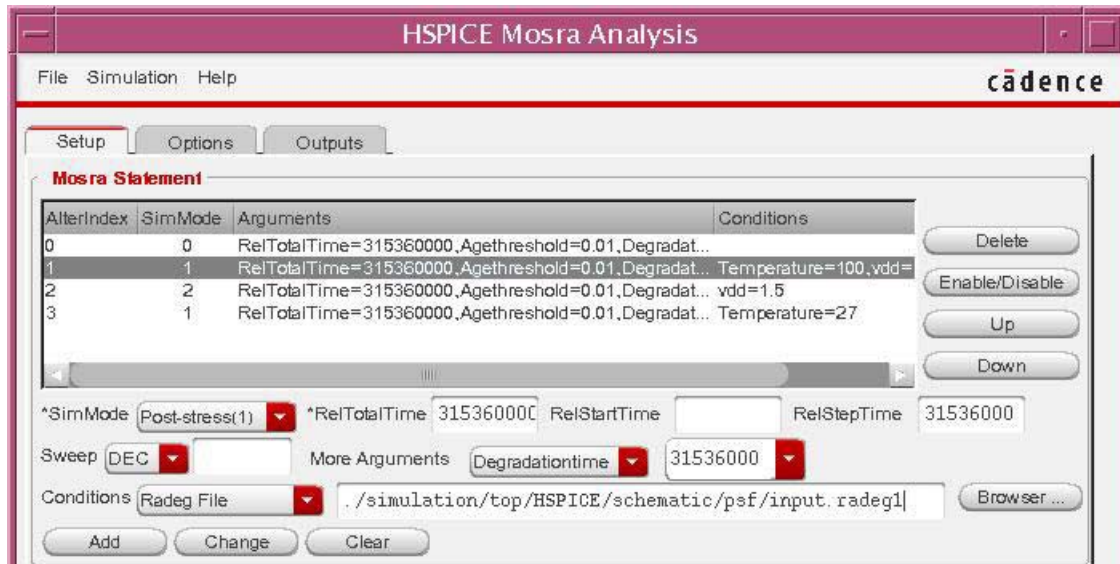
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 127 Change Topology view of the Mosra Command section

- **Radeg File** — Enables selection of a specific degradation file for post-stress MOSRA simulation. The **Radeg File** choice displays a file path input field and **Browser** button.

Chapter 13: HSPICE Reliability Analysis (MOSRA)

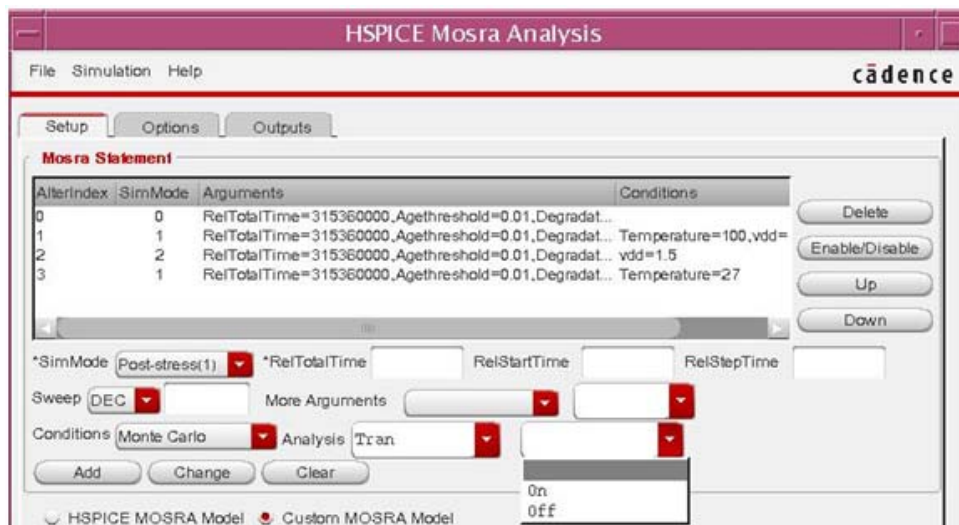
Setting Up Reliability Analysis



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 128 Radeg File condition setup of UI

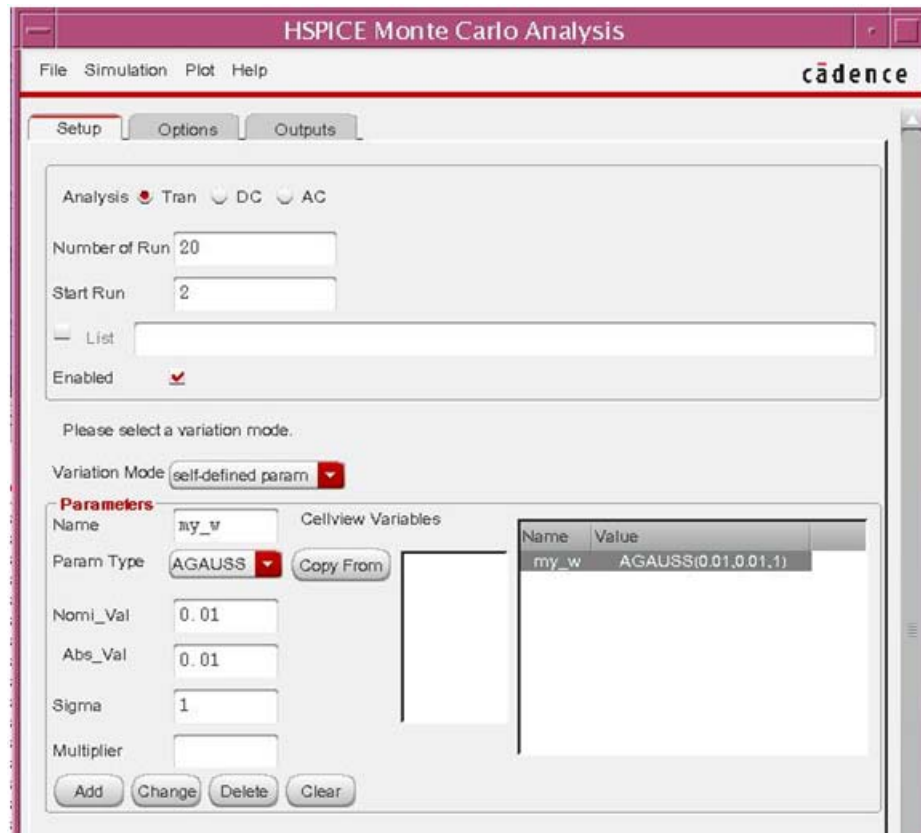
- Monte Carlo** — Allows the invocation of a Monte Carlo simulation. Selection of this option adds a **Status** field indicating the Monte Carlo run to be on or off. See [Chapter 8, Monte Carlo in the HSPICE Integration](#) for complete discussion of this feature.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 129 Monte Carlo selected in the Conditions combination box

The basis for the Monte Carlo simulation is determined by the setup in the HSPICE Monte Carlo Analysis window where a transient analysis and 20 trials have been specified in the example (Figure 130).



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 130 Monte Carlo Analysis utility setup for MOSRA run

- **RelTotalTime** — (Required) Final reliability test time to use in post-stress simulation phase; the field needs one user-specified essential value for MOSRA analysis. **Note:** **RelTotalTime** cannot be used with **Sweep** (below).
- **RelStartTime** — Time point of the first post-stress simulation; Default is 0.
- **RelStepTime** — Input RelStep argument value.
- **Sweep** — Select and input DEC (decade variation) or LIN (linear variation) value.
- **More Arguments** — Simplifies the setting of .MOSRA statement arguments.

- **Add** — Adds the MOSRA command into the simulation environment after user input of the above information. The MOSRA command is added to the **MOSRA Statement** report field and assigned a value of **AlterIndex**.
- **Change** — Changes an existing MOSRA command using the following procedure: (1) Select a MOSRA command in the MOSRA Statement report to populate the MOSRA command into input fields; (2) change any arguments in each input field; (3) click the Change button to reload the MOSRA statement in the **Mosra Statement** report field.
- **Clear** — Removes all the information in input fields.
- **Delete** — Deletes any selected MOSRA command in the MOSRA Statement report field.
- **Enable/Disable** — Enables or disables the selected MOSRA command in the **Mosra Statement** report field. If a MOSRA command is enabled, it is reassigned an alter index; if it is disabled, the value in the **AlterIndex** column is “X” and it is moved to the bottom of the report field. **Note:** in the screen capture showing the **Mosra Statement** report, while three MOSRA commands are displayed, the lower two are disabled (X).
- **Up** — Manually move the selected MOSRA command to the top of the report.
- **Down** — Manually move the selected MOSRA command to the bottom of the report field.

Setting Up MOSRA Models

HSPICE Reliability Analysis supports both HSPICE default MOSRA models and Custom MOSRA models. You can only select one of the two radio buttons to run a simulation.

The following sections discuss these topics:

- [HSPICE MOSRA Models \(Built-in\)](#)
- [Custom MOSRA Models](#)
- [Appending Model Section](#)

HSPICE MOSRA Models (Built-in)

HSPICE MOSRA model (Level 1) uses HSPICE internal default MOSRA degradation equations.



Figure 131 HSPICE MOSRA Models

If you select the HSPICE MOSRA Model radio button this section includes the following fields and controls:

- **MOSRA Model** report — Displays MOSRA models currently set up.
- **Name** — User-defined MOSFET reliability model name.
- **HCI Params** — Reliability model parameter for HCI; (1) select the parameter name from the choices; (2) input its value.
- **NBTI Params** - Reliability model parameter for NBTI; (1) select the parameter name from the choices; (2) input its value.
- **Add** — Make additions to the MOSRA model following inputs to the above fields; successful additions are shown in the MOSRA Model report.
- **Change** — Use to modify the selected model in the MOSRA Model report.
- **Clear** - Removes all the values in the input fields.
- **Default All** — Removes only the values in the **HCI Params** or **NBTI Params** fields.
- **Delete** - Deletes the selected MOSRA model in the report.
- **Enable /Disable** — Enables/disables selected MOSRA models in the report field (which will not be netlisted after HSPICE runs a simulation).

Custom MOSRA Models

The user-defined Custom MOSRA model defines the degradation equations through MOSRA API and MOSRA models. The following is the screen capture of the **Custom MOSRA models** fields group and its related APPENDMODEL commands.

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Setting Up Reliability Analysis

Figure 132 Custom MOSRA Model

The following fields and controls are displayed in the **Custom MOSRA Model** section:

- **Shared-Library Path** — Selects the custom MOSRA shared library path, which will be set as the value of `hspice_mosra_models` environment variable
- **MOSRA Model File** — Selects the MOSRA models definition file. **Note:** The relative path in **Shared-Library Path** and **MOSRA Model File** is based on the current working directory.
- **MOSRA Model Name** — Use to select from the choices of the MOSRA Model Name field in the **Appending Models** section.

After you select the MOSRA Model File, the file is parsed to load all the defined MOSRA libraries as choices for the **MOSRA Model Name** field.

If the value of **MOSRA Model Name** in this section is **All**, then the choices of the **MOSRA Model Name** field in the **Appending Models** section are all the defined MOSRA libraries in the **MOSRA Model File**.

The **Appending Models** report shows all the Custom MOSRA models' related `.APPENDMODEL` commands.

Appending Model Section

The `APPENDMODEL` command binds the MOSRA model to normal MOSFET models.

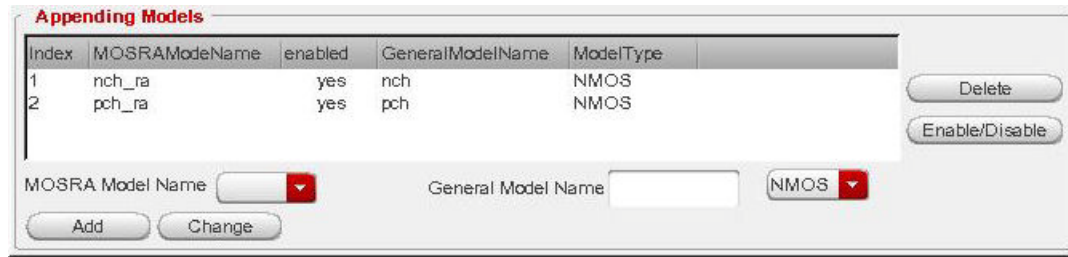


Figure 133 Appending Models section

The following are the Appending Models section fields and controls:

- **Appending Models** report — Displays a table of the defined .APPENDMODEL commands information. Its values change according to the contents of the section of **HSPICE MOSRA Model** or **Custom MOSRA Model** in the **MOSRA Models** section.
- **MOSRA Model Name** - The MOSRA model name, which is defined in **MOSRA Models** section.
- **General Model Name** — User-defined name; needs to be filled in with the same name defined in user model files that the foundry shipped. The **General Model Name** can be associated with the .appendmodel command only once.
- **NMOS/PMOS** — Model type for the **General Model Name**.
- **Add** — Adds the APPENDMODEL command into the simulation environment.
- **Change** — Enables changes the content of the selected APPENDMODEL command in the **Appending Models** report.
- **Delete** — Deletes the selected .APPENDMODEL command information in the **Appending Models** report.
- **Enable/Disable** — Enables or disables the selected APPENDMODEL command information in the **Appending Models** report. If the APPENDMODEL command is disabled, it will not be netlisted when the simulation is run.

Setting up MOSRA-Related Options

The **Options** page ([Figure 122 on page 276](#)) lists a subset of HSPICE control options that support MOSRA-related commands and simulation.

Use the Options page to specify the values for the following options:

- APPENDALL
- DEGFN
- DEGFP
- MOSRALIFE
- MOSRASORT
- MRAEXT
- MRAPAGED
- RADEGOUTPUT

The values displayed in the fields are the defaults supported by HSPICE. For the detailed information on the options, refer to [HSPICE and RF Netlist Simulation Control Options](#) in the *HSPICE Reference Manual: Commands and Control Options*.

Alternatively, clicking the button on the left side of each option pops up the help information about the option. When you change the value of an option, a changes pending asterisk, “*” appears on the upper right side of the page.

Note: You must click **Apply** to add the change into simulation environment.

See [Options](#) in Chapter 6 of this manual for full discussion of HSPICE simulation option usage.

Setting up the Outputs Page

The HSPICE Mosra Analysis window **Outputs** page enables you to set enables you to set preferences for **Output Plots**, and **Measurements**, operate **Simulation** controls, and view **Radeg Files** (reliability analysis degradation results files).

Simulation Control

After setting up the MOSRA commands and models, adjusting options, and specifying measurements, you can do netlisting and simulation.



Figure 134 Simulation controls

Use either the **Simulation** menu in the HSPICE Mosra Analysis window or the **Simulation** controls section in the **Outputs** page. The following are the simulation control buttons:

- **Netlist & Run** — Recreates netlisting and runs a simulation.
- **Netlist** — Recreates netlist file, without running a simulation.
- **Run** — Directly runs a simulation based on the existing netlisting file.
- **Display** — Displays the existing netlist file.

Note that you need to set up a Tran, DC, or AC analysis before running HSPICE Reliability analysis. See [Chapter 4, Analysis Setup and Design Variables](#) for discussion of the three analyses in the HSPICE integration to ADE.

Using the Outputs Plot Control Section

Use the **Outputs Plot** section to plot waveforms of different voltage/current signals or expressions. This section can be used to also evaluate an expression to a value for both pre-stress and post-stress simulation. You can select different waveforms associated with multiple .ALTER statements generated by HSPICE and then plot a signal's waveform during the user-defined degradation period by using the **Plot in DegradationPeriod** button. This plot can be used to monitor the real circuit response after degradation parameters are in effect. Typically, these user-defined voltage/current signals can be imported from the Environment console.

Following a successful simulation, the choices in the **Waveform Files** and **Degradation Period** fields are automatically loaded in the **Outputs Plot** section. (Alternatively, you can load them by clicking **Get Outputs from Console**.) The choices of the **Degradation Period** are the **retime** which also automatically loads with **Waveform Files**. If you plot pre-stress simulation outputs, the **Degradation Period** need not be selected. If you want to plot all

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Setting up the Outputs Page

degradation period waveforms, select the **All** value below the **Degradation Period** field.



Figure 135 Output Plots control section

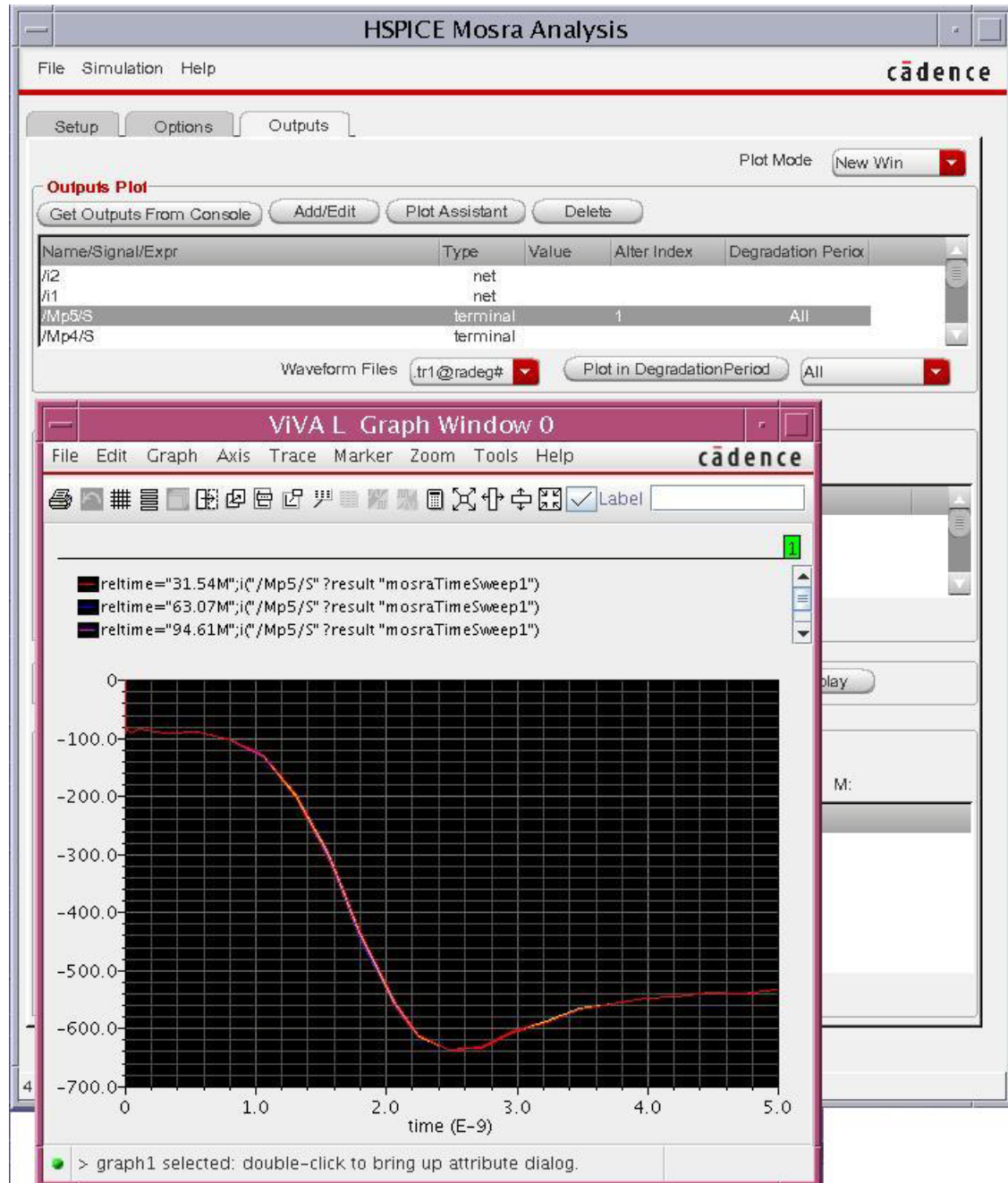
Following a successful netlisting and simulation, plot outputs from the **Outputs** page using the procedure below.

To plot an output signal or expression:

1. Click **Get Outputs from Console** or **Plot Assistant** to load the existing outputs into the **Outputs** report field. (See [HSPICE Plotting Assistant](#) in [Chapter 7, Printing and Plotting Results](#) for details on using the Plotting Assistant.)
2. Select one output row in the **Outputs** report field
3. Select **Waveform Files, Degradation Period**. The **Waveform Files** field value is a psf output file which combines the `.tr/.sw/.ac` psf file suffix string with **AlterIndex** and **SimMode**. (`*.tr` is the transient analysis psf file suffix, `*.sw` is the DC analysis psf file suffix, and `*.ac` is the AC analysis psf file suffix.) For example: `.tr0` means alter 0, pre-stress simulation psf output file; `.tr1@ra` means alter1, post-stress simulation psf output file.
4. Click either the **Plot** or **Plot in DegradationPeriod** button.

Result: The tool plots the results waveform in the waveview window.

[Figure 136 on page 291](#) shows the waveforms of a plotted signal. It plots the degradation periods waveforms of signal **/Mp5/S** in alter 1 transient analysis. The *mosraTimeSweep1* means “alter 1 MOSRA transient analysis.” If it were a “dc” or “ac” analysis, it would be “mosraSrcSweep1” or “mosrFrequencySweep1.”



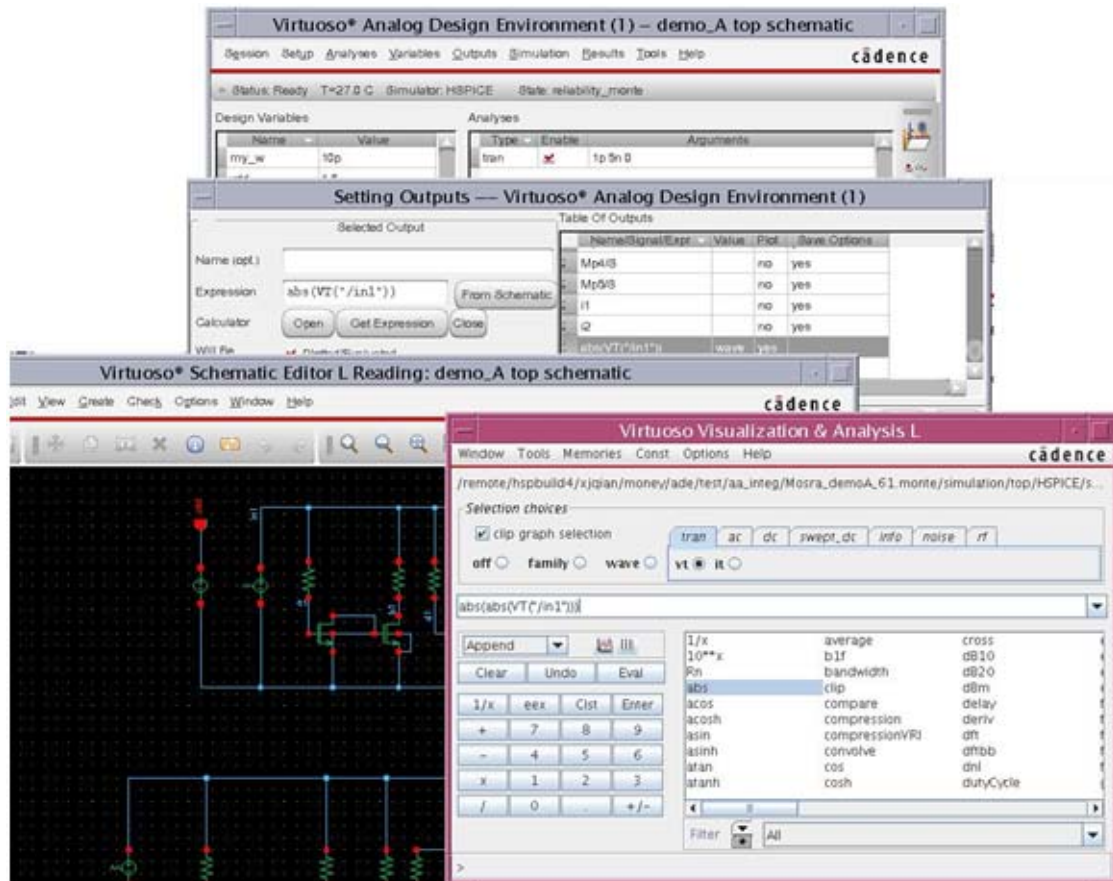
© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 136 Waveform based on the signal Mp5/S

- To plot or evaluate expressions, use the ADE Calculator to add the expression.

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Setting up the Outputs Page



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 137 Sequence to Evaluate an Expression using the ADE Calculator

Note: Only VT, IT, VF, IF, VS, and IS expressions are supported in MOSRA outputs plots.

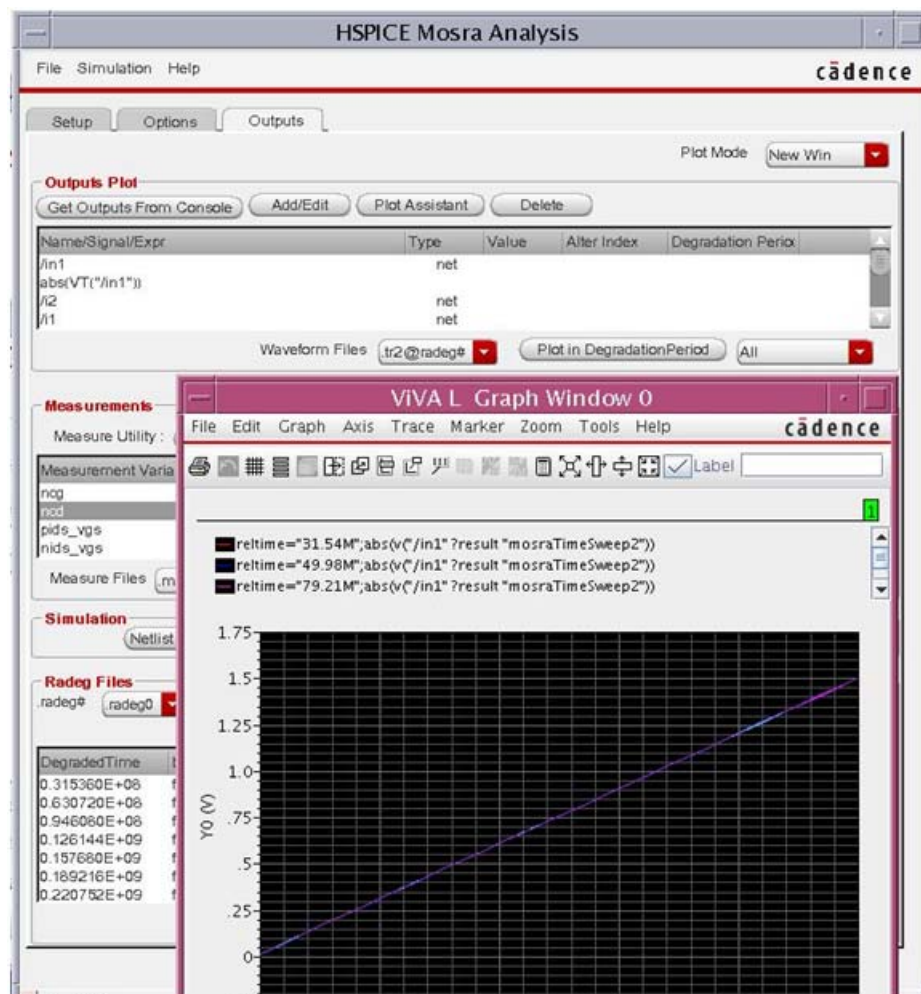
Adding Expressions Using the Calculator

Use the following procedure to add expressions into MOSRA with the Calculator.

1. In the Environment console, select the **Outputs** menu and open the Setting Outputs form.
2. Enter the expression by selecting the signal on the schematic editor.
3. In the Setting Outputs form **Calculator** click the **Get from Schematic** button.

4. Perform calculations in ADE Calculator window on your selected expression. (See [Setup on page 152](#) in [Chapter 5, Saving-Plotting Outputs](#) for details on using the calculator.)
5. After expression being added, on the **Outputs** tab of the HSPICE Mosra click the **Get Outputs from Console** button to load it into MOSRA **Outputs Plot** report field.
6. Plot the expression using the plot controls in the Outputs tab.

Result: The following screen capture shows the expression “abs(VT("in1")) waveform:



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 138 Waveform for abs(VT("in1"))

After a successful netlisting and simulation, the measurement variables results information loads automatically. You can select the different measure output files associated with multiple .ALTER statements generated by HSPICE and then you can print or plot such measure results at all degradation times as a waveform. The choices in **Measure Files** field are automatically loaded after a successful simulation. You can also load them by clicking **Get Measurements**.



Take advantage of the HSPICE Measurement Utility to efficiently set up measurement variables on the **Outputs** page. Click the **Open** button in the Measurements section to launch the HSPICE Measure Utility [Figure 140](#)). (See [Chapter 12 on page 257](#) for a full discussion of this feature.) After finishing setup of the measurement variables in this utility window, return to the HSPICE Mosra Analysis window **Outputs** page.

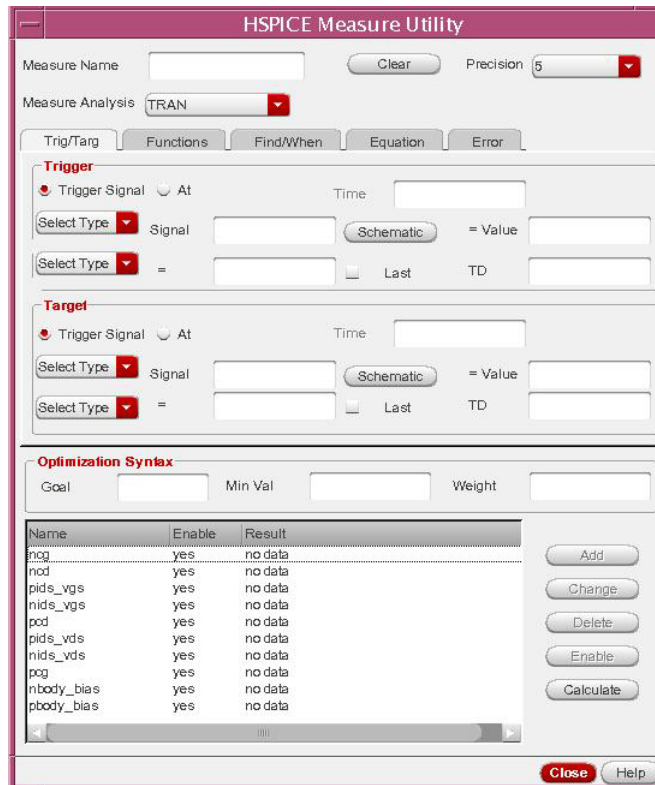


Figure 140 HSPICE Measure Utility window

The **Measure Files** field value is a measure results file which combines the *.mt/.ms/.ma* measure result file suffix string with the alter index and simMode. (**.mt* is the transient analysis measure results file suffix, **.ms* is the DC analysis measure results file suffix, and **.ac* is the AC analysis measure results file suffix.) For example: *.mt0* means alter 0, pre-stress simulation measure results file; *.mt1 @ra* means alter1, post-stress simulation measure result file.

The **Reltime** field to the right of the **Measure Files** field is used for MOSRA + Monte Carlo co-simulation, which outputs multiple measure results file. This field shows the suffix of these results files. If there are no Monte Carlo simulations, the field is disabled. (The default selection is **All**.)

The **Waveform type** field and **Bins** field are used to specify how to plot measure variable results. Possible choices are **Curve**, **Histo**, and **Scatter**. For **Histo**, the **Bins** field becomes enabled and requires input.

Select **Measure Files** to show measure variables results in the **Measurements** report. Select a measure variable row in the report field and click **Plot** to

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Setting up the Outputs Page

generate a waveform of measurement variable results. [Figure 141 on page 296](#) shows such a waveform. This waveform plots the measurement variable “pids_vgs” results of an alter 2 post-stress transient analysis.

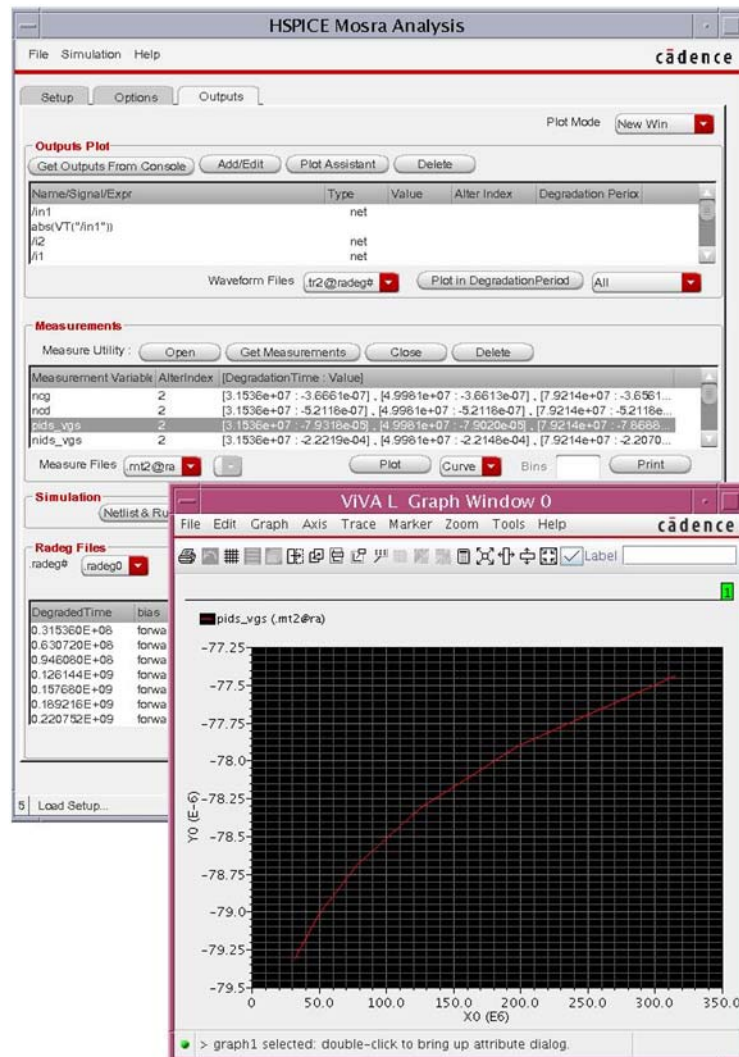


Figure 141 Waveform of Measure results

The screen capture in [Figure 142](#) shows a plot for a MOSRA + Monte Carlo measure variable of alter 1 which runs a transient Monte Carlo simulation:

Chapter 13: HSPICE Reliability Analysis (MOSRA) Setting up the Outputs Page

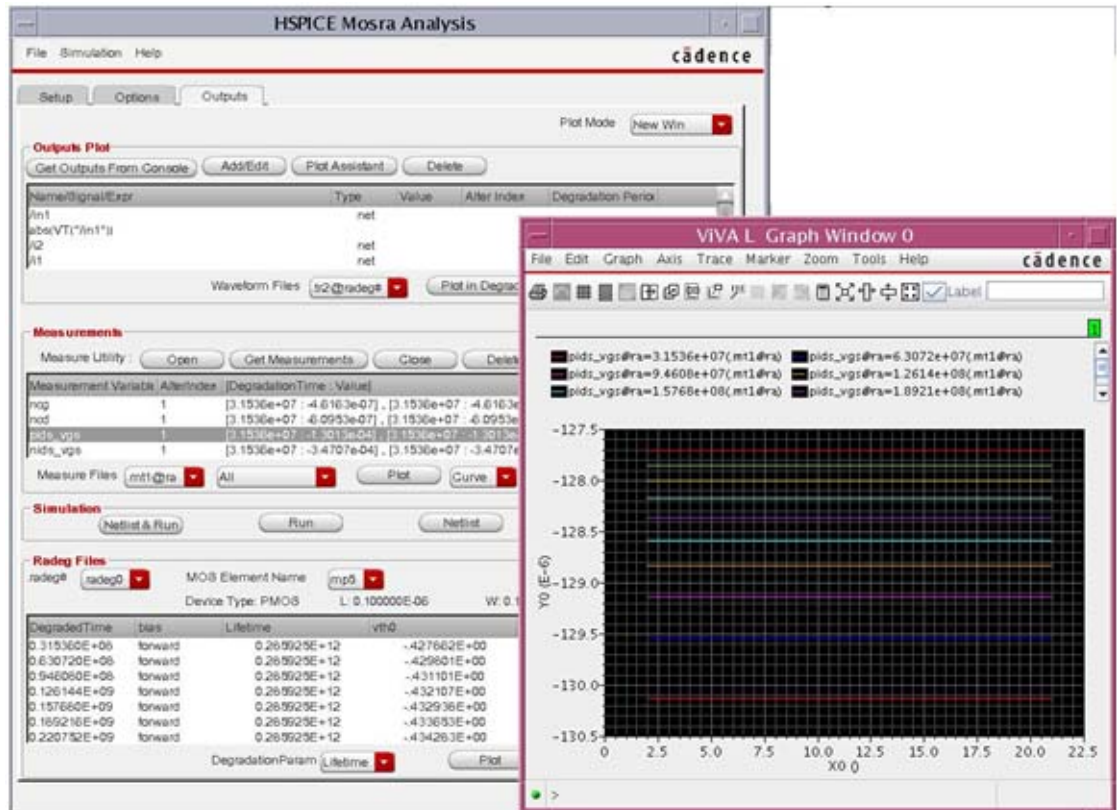


Figure 142 MOSRA-Monte Carlo measurement results

To print all the measurement results in a measure file, select **Measure Files** and click **Print**. The tool prints measure results information from the measure file in a Results Display Window using a table format (Figure 143 on page 298).

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Setting up the Outputs Page

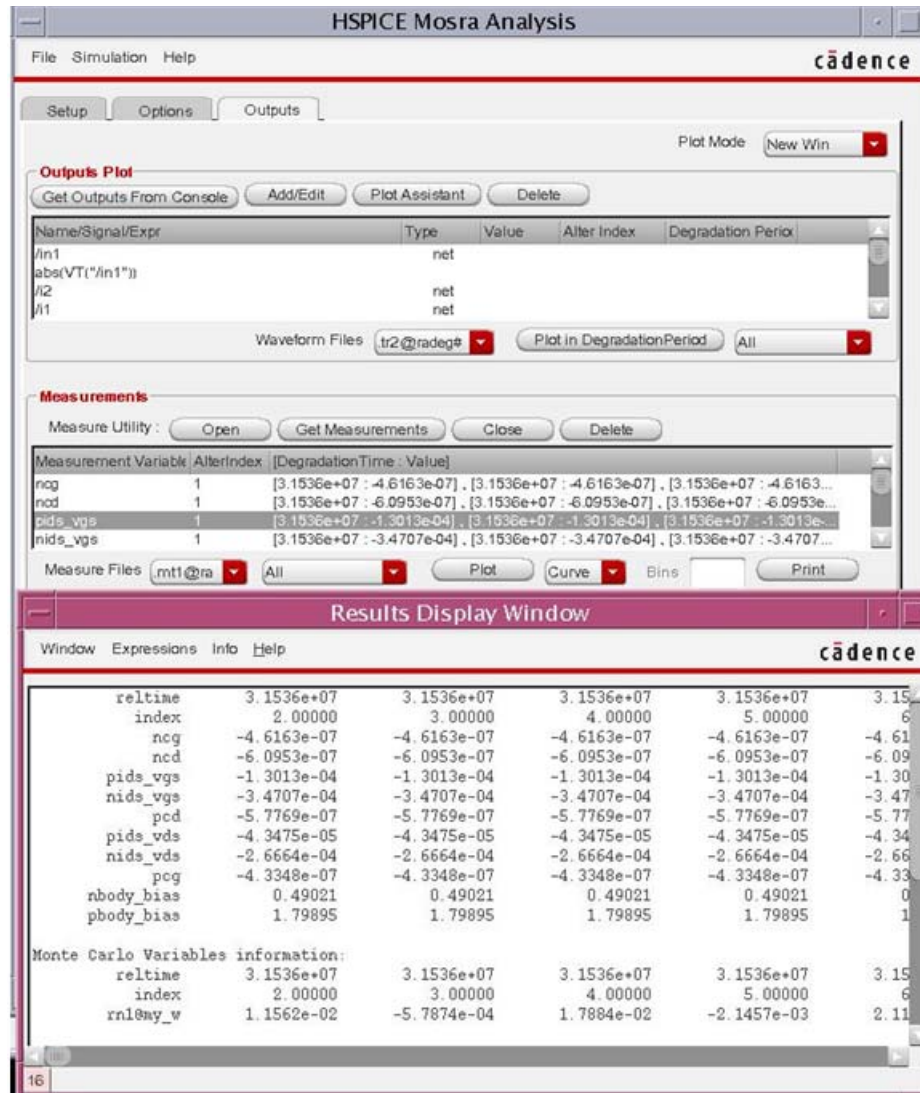


Figure 143 Result Display Window with measure results in tabular format

Displaying Reliability Analysis Degradation Data

Use the **Outputs** page to display, plot, and print information from Radeg Files.

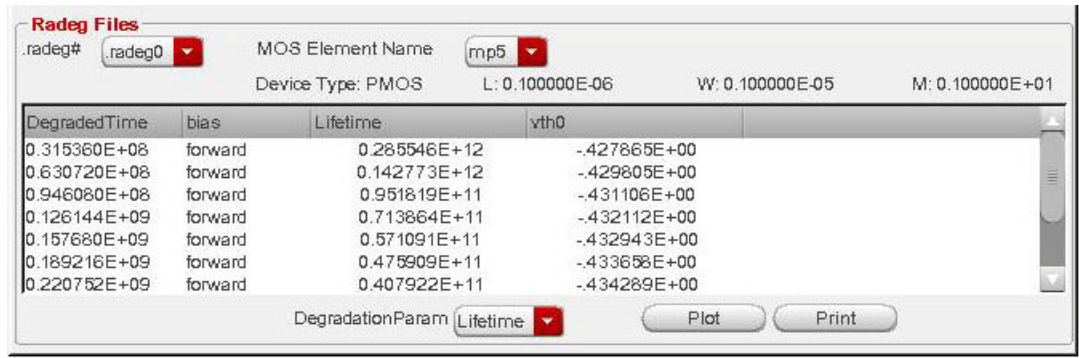


Figure 144 Radeg Files section on the Outputs page

To show and plot the degradation parameters for each user-specified MOS element after doing MOSRA analysis use the **Outputs** page controls. The Radeg Files section enables you to easily evaluate and analyze the degraded parameters. Multiple degradation files (.radeg#) can be generated by HSPICE through multiple .MOSRA statements set up in the previous tabs. Here, one selection list is available for user to select actual degradation file to analyze and plot.

The choices in the **Radeg Files** field are automatically loaded after a successful simulation.

The **Radeg Files** field value is composed of the radeg file suffix combined with the .radeg suffix string with the alter index. For example: “.radeg0” means alter 0, radeg file.

To show the element information in the radeg file in the Radeg Information report:

1. Select **Radeg Files** and **MOS Element Name** to show the element information in the radeg file in the **Radeg Files** information report.
2. Select **Degradation Param.**
3. Click **Plot** to plot the element parameter information with **Degradation Time**.

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Setting up the Outputs Page

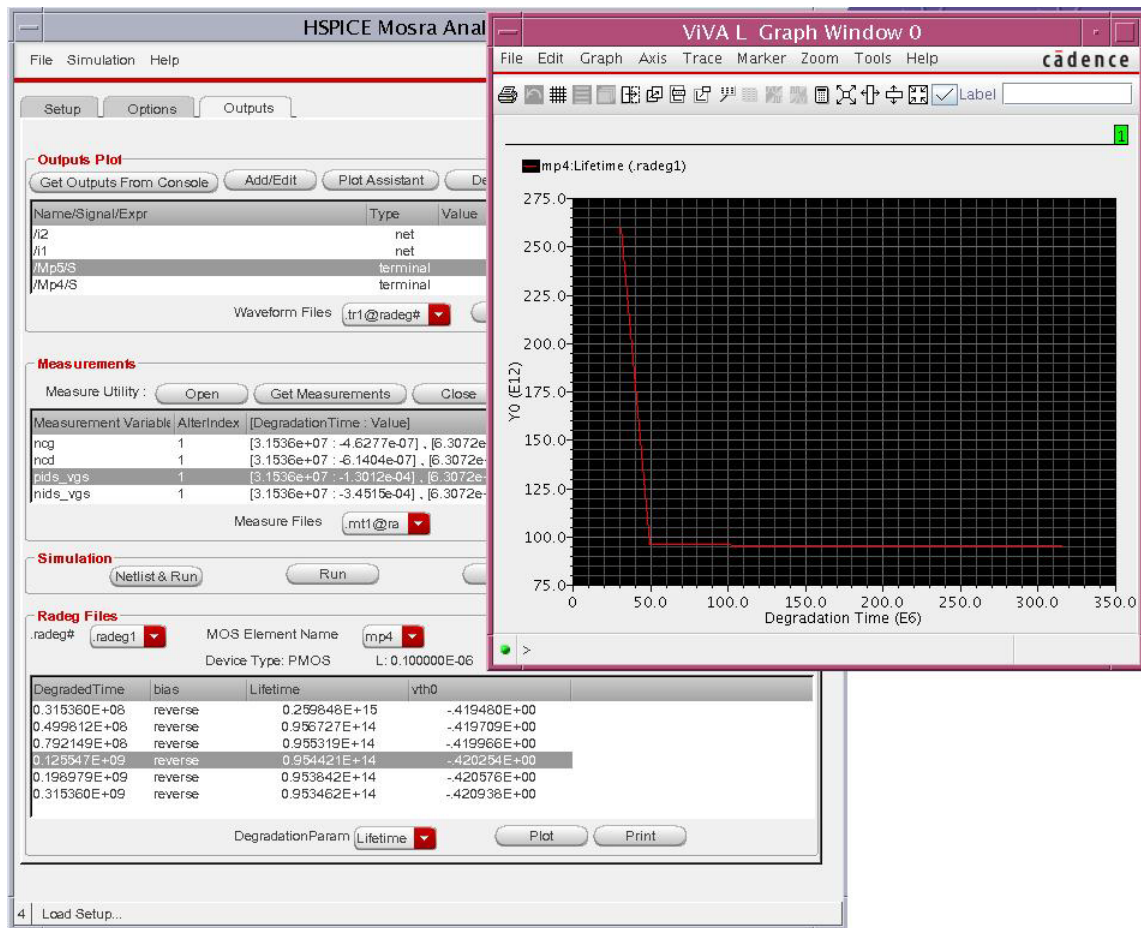
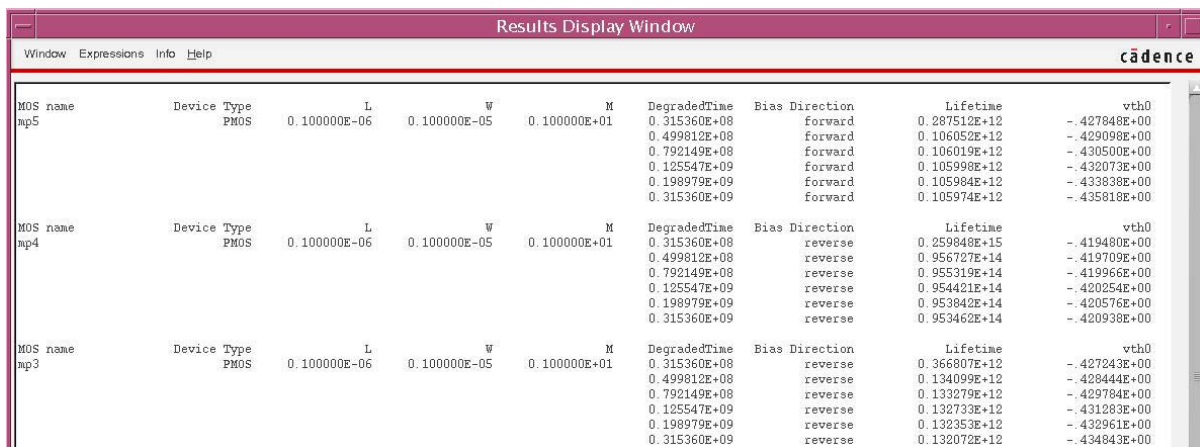


Figure 145 Radeg data plotted in a waveform

- Click **Print** to open a Results Display Window which reports the radeg file information in a table format.

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Other Features Supported in the HSPICE Reliability Analysis Window



The screenshot shows the 'Results Display Window' with a menu bar (Window, Expressions, Info, Help) and the Cadence logo. It displays three tables of MOS device reliability data for devices mp5, mp4, and mp3. Each table has columns for MOS name, Device Type, L, W, M, DegradedTime, Bias Direction, Lifetime, and vth0.

MOS name	Device Type	L	W	M	DegradedTime	Bias Direction	Lifetime	vth0
mp5	PMOS	0.100000E-06	0.100000E-05	0.100000E+01	0.315360E+08	forward	0.287510E+12	-.427848E+00
					0.499812E+08	forward	0.106052E+12	-.429098E+00
					0.792149E+08	forward	0.106019E+12	-.430500E+00
					0.125547E+09	forward	0.105998E+12	-.432073E+00
					0.198979E+09	forward	0.105984E+12	-.433838E+00
					0.315360E+09	forward	0.105974E+12	-.435818E+00
mp4	PMOS	0.100000E-06	0.100000E-05	0.100000E+01	0.315360E+08	reverse	0.259848E+15	-.419480E+00
					0.499812E+08	reverse	0.956727E+14	-.419709E+00
					0.792149E+08	reverse	0.955319E+14	-.419966E+00
					0.125547E+09	reverse	0.954421E+14	-.420254E+00
					0.198979E+09	reverse	0.953842E+14	-.420576E+00
					0.315360E+09	reverse	0.953462E+14	-.420938E+00
mp3	PMOS	0.100000E-06	0.100000E-05	0.100000E+01	0.315360E+08	reverse	0.366807E+12	-.427243E+00
					0.499812E+08	reverse	0.134099E+12	-.428444E+00
					0.792149E+08	reverse	0.133279E+12	-.429784E+00
					0.125547E+09	reverse	0.132733E+12	-.431283E+00
					0.198979E+09	reverse	0.132353E+12	-.432961E+00
					0.315360E+09	reverse	0.132072E+12	-.434843E+00

Figure 146 Radeg file information presented in tabular form

Other Features Supported in the HSPICE Reliability Analysis Window

Use the HSPICE Reliability Analysis window to save and load all the user operations for simulation as a standard ADE simulation, and create an OCEAN script based on all operations performed in an HSPICE Reliability Analysis session.

1. To **Save/Load States**, under the **File** menu, select either the **Save Setup** or **Load Setup**. (Alternatively, select the **Save State** or **Load State** option in the Environment Console to perform the same operation.)
2. Create an OCEAN script based on your HSPICE Reliability Analysis session by either: Selecting **File >XXX** and saving all the operations as a single OCEAN script to do a post batch run simulation with Reliability analysis, or use the **Help** menu to access an html user guide for the HSPICE Reliability feature or the reference guide for OCEAN API functions.

Chapter 13: HSPICE Reliability Analysis (MOSRA)

Other Features Supported in the HSPICE Reliability Analysis Window

HSPICE Violation Check (.BIASCHK)

Describes the HSPICE simulation violation (bias check) capability in the HSPICE integration to the Cadence® Virtuoso® Analog Design Environment.

The HSPICE Violation Check (.BIASCHK) is useful to ensure that certain conditions do not occur to violate design rules (for example, forward-biased diode). This chapter describes how to set up HSPICE .BIASCHK commands in the HSPICE integration to the ADE environment, run an HSPICE BIASCHK analysis simulation, highlight the corresponding device (that violates the check) in the schematic canvas, and show and print .BIASCHK detailed outputs.

For a detailed description of this command see [.BIASCHK](#) in the *HSPICE Reference Manual: Commands and Control Options*.

The following sections discuss these topics:

- [Invoking and Setting up Violation Check](#)
- [Netlisting and Running .BIASCHK Simulations](#)
- [Displaying/Printing Violation Check Results](#)

Invoking and Setting up Violation Check

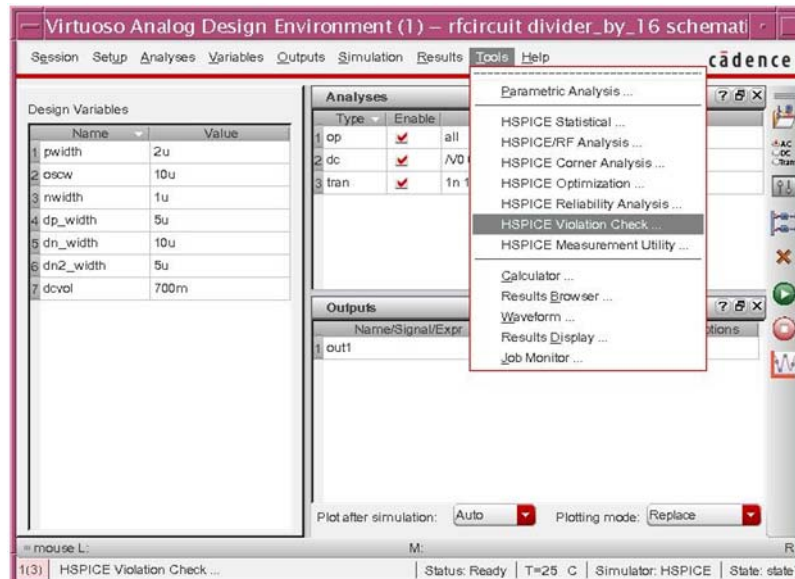
The HSPICE Violation Check form is used to:

- Set up HSPICE .BIASCHK commands before a simulation run
- Verify simulation violation check results
- Call up a listing of the violation details

To invoke the HSPICE Violation Check form, on the Environment console select **Tools > HSPICE Violation Check** ([Figure 147](#)).

Chapter 14: HSPICE Violation Check (.BIASCHK)

Invoking and Setting up Violation Check



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 147 HSPICE Violation Check menu item on the Tools menu

The HSPICE Violation Check form opens (Figure 148). This interface supports all HSPICE .BIASCHK command usages.

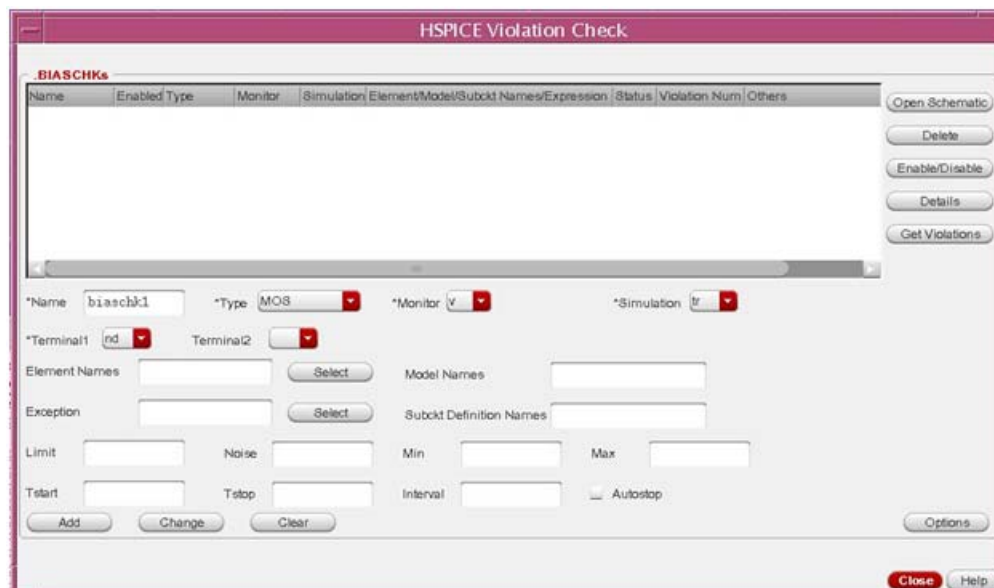


Figure 148 HSPICE Violation Check form

The form enables the monitoring of general elements, model, subckt, region, length and width, and expression; it can also monitor associated voltage, current, w and l. The form changes according to the **Type** (of element under study), and the specified parameter selected to be monitored (**Monitor**).

These topics are discussed in the following sections:

- [GUI Controls](#)
- [Example Monitoring Setups](#)
- [Setting Up .BIASCHK Commands and Options](#)

GUI Controls

For a detailed description of the use of the .BIASCHK command and use models see [.BIASCHK](#) in the *HSPICE Reference Manual: Commands and Control Options*.

.BIASCHKs Report Field

The **.BIASCHKs** report field is a table that lists all the BIASCHK commands that have been set up (added) and their summary information. It also provides selections for specific BIASCHK run modes and handling of outputs.

Control Buttons

The control buttons on the right side of the form are:

- **Open Schematic:** Opens design in schematic window in order to highlight violation devices.
- **Delete:** Deletes the selected .BIASCHK command in the **.BIASCHKs** report field from a simulation.
- **Enable/Disable:** Enables for simulation or disables a selected .BIASCHK command in the **.BIASCHKs** report field.
- **Details:** Opens an HSPICE Violation Details form following a simulation.
- **Get Violations:** Opens parsers-noted violations file after an HSPICE .BIASCHK simulation.

Selection Boxes and Fields

The following controls are displayed for setting up .BIASCHK commands. The HSPICE Violation Check form is dynamic in that the controls that are displayed

Chapter 14: HSPICE Violation Check (.BIASCHK)

Invoking and Setting up Violation Check

depend on the type of monitoring that is selected. See [Example Monitoring Setups on page 310](#).

- **Name** field: .BIASCHK command name; if you do not specify a name, the name **biaschk***n*, where *n* is an incremental number, is assigned. For example, if you do not specify a name, the name, “biaschk1” is assigned to the .BIASCHK command.
- **Type** selection box: Identifies what is to be monitored in a .BIASCHK simulation. Available circuit or design and model element types include DIODE, BIPOLAR, BJT, JFET, MOS, NMOS, PMOS, Capacitor, Resistor, SUBCKT, Region, or Expression.
- **Monitor** selection box: Type of value you want to monitor. You can specify **v** (voltage), **i** (current), **w**, and **I** (device size) for the element type. The **Monitor** selection box is removed for **Expression** and **Region** bias checks. For a **Region** bias check, check boxes are added for transitions to and from a region including **cutoff**, **linear**, and **saturation**.
- **Simulation** selection box: Specify **op**, **dc**, **tr** (transient), or **all** (op, dc, and tr). The tr option is the default simulation type.
- **Terminal1**, **Terminal2** or **Pin1**, **Pin2** selection boxes: Input bias check element terminals. When monitoring a **SUBCKT**, the terminal names are those pins defined by the subcircuit definition of *mname*. You can use the **Select** button to import directly from a schematic. The **Terminal** and **Pin** selection boxes only are visible for element and model or subcircuit-type bias checks.
- **Expression** field: Input an expression for an expression-type .BIASCHK command; you can input an expression according to .BIASCHK usage directly.
- **Element Names** field: Input element names (except expression-type) for a bias check. Element names can also be selected from the schematic using the **Select** button. Multiple names can be input by using delimiter blank spaces or commas “,”.
- **Model Names** field: Input model names for all except expression-type bias check. The element model can also include wildcard characters “*” or “?”. It can also be input multiple names by using delimiter space or “,”.
- **Exception** field: Input element or instance names that you do not want to have bias checked. These can also be selected from a schematic using the **Select** button. Multiple names can be input using delimiter spaces or commas “,”.

- **Subckt Instance Names** field: When **Type** indicates the bias check element is a **SUBCKT**, values for **Pin1, 2** can be input using the **Select** button to import from the schematic design. When the instance names are an input directory, use the backward slash "/" as delimiters.
- **Subckt Definition Names** field: Input subckt names (except for expression-type bias checks). These names can include wildcard characters "*" or "?". Multiple definition names can be input by using delimiter spaces or commas " , " .
- **Limit** field: Bias check limit that you define. Reports an error if the bias voltage (between appointed terminals of appointed elements and models) is larger than the limit.
- **Noise** field: Bias check noise that you define. The default is 0.1v. Noise-filter some of the results (the local maximum bias voltage that is larger than the limit). The next local max replaces the local max if all of the following conditions are satisfied: `local_max-local_min noise. next local_max-local_min noise`. This local max is smaller than the next local max. For a parasitic diode, HSPICE ignores the smaller local max biased voltage and does not output this voltage. To disable this feature, set the noise detection level to 0.
- **Min** field: Input minimum value.
- **Max** field: Input maximum value.
- **Tstart, Tstop** - used to define duration for bias check. Used only for transient analysis.
- **Interval** field: Active when `.OPTION BIASINTERVAL` is set to a non-zero value. This argument prevents reporting intervals that are less than or equal to the time specified.
- **Autostop** check box: When checked, supports an autostop for a `.BIASCHK` simulation so that it can report error messages and stop the simulation immediately.

Chapter 14: HSPICE Violation Check (.BIASCHK)

Invoking and Setting up Violation Check

Utility Buttons

The following buttons enable you to prepare to simulate according to your setup parameters.

- **Add:** Click this button when you have completed the input fields and selection boxes to create a new bias check. If all parameters are successful, it will be displayed in the **BIASCHKs** report table.
- **Change:** Click this button to edit any selected bias check in the **BIASCHKs** table.
- **Clear:** Click to clear all of the input fields to create a new bias check.
- **Options:** Click to open the HSPICE Options form ([Figure 149 on page 309](#)) and locate the “BIASCHK” section of bias check options for you to automatically add options to the violation check.

These options include:

- BIASFILE
- BIASINTERVAL
- BIASNODE
- BIASPARALLEL
- BIAWARN

For a description of these options see [HSPICE and RF Netlist Simulation Control Options](#) in the *HSPICE Reference manual: Commands and Control Options*. See also: [Setting Up .BIASCHK Commands and Options on page 313](#).

Chapter 14: HSPICE Violation Check (.BIASCHK) Invoking and Setting up Violation Check

The image shows the HSPICE Options dialog box. The 'Options Filter' is set to 'Output Control'. The 'Reset All To:' buttons are 'Cdsenv' and 'HSPICE'. The 'Log file formatting' section includes 'INGOLD' (Engineering (0)), 'LIS_NEW' (checked), 'DCCAP' (unchecked), 'NOELCK' (unchecked), 'NOTOP' (unchecked), 'NUMDGT' (4), 'OPTLST' (0), 'OPTS' (unchecked), 'WARNLIMIT' (1), and 'WARN_SEP' (unchecked). The 'Waveform files' section includes 'FFT_ACCURATE' (unchecked), 'FFTOUT' (unchecked), and 'INTERP' (unchecked). The 'Meas Options' section includes 'MEASDGT' (4), 'MEASFAIL' (checked), 'MEASFILE' (unchecked), 'MEASOUT' (checked), and 'PUTMEAS' (checked). The 'BIASCHK Options' section, which is circled in red, includes 'BIASFILE' (biasout1), 'BIASINTERVAL' (3 of All Violation Regions (3)), 'BIASNODE' (Port Names (0)), 'BIASPARALLEL' (unchecked), and 'BIAWARN' (unchecked). At the bottom are 'OK', 'Cancel', 'Apply', and 'Help' buttons.

Figure 149 .BIASCHK command control options

Example Monitoring Setups

The following shows the HSPICE Violation Check form for various sample usages:

- **Element and Model:** When your violation check setup is to monitor an element and model, the form is configured as shown in [Figure 150](#). In this type of .BIASCHK command setup, the **Monitor** input selector choices and **Terminal1**, **Terminal2** choices are changed according to the selected **Type** value:

Name	Enabled	Type	Monitor	Simulation	Element/Model/Subckt Names/Expression	Status	Violation Num	Others
biaschk1	Yes	Region	linear, sat	op	name="x".m"	FAIL	104	autostop=No
biaschk2	Yes	SUBCKT	v	tr	sktinsts=/I0,/I1,/I2	FAIL	116	pin1=out1,limit=0.2,aut
biaschk4	Yes	NMOS	i	tr	name=/I5/M0.mname=cmospn	FAIL	3566	terminal1=nd,terminal2
biaschk3	Yes	MOS	i	tr	name=/I5/M0.mname=cmospn	FAIL	39	terminal1=nd,limit=5u,
biaschk6	Yes	Expression	op		expression=(subx15.out11(x15.m0)	FAIL	1	min=0,max=0.000003n
biaschk7	Yes	Expression	all		expression=V(out31)V(out21)	FAIL	197	min=0.1,max=1.1stop=
biaschk8	Yes	CAP	i	all		FAIL	2	terminal1=n1,limit=1fa
biaschk9	Yes	PMOS	w	all	name="x".m".mname=cmosp	FAIL	5236	limit=1u,max=1u,autos
biaschk5	Yes	RESISTOR	v	dc	name=/I2/R0,/I0/R0,/I1/R0	FAIL	213	terminal1=n1,limit=1px
biaschk10	Yes	DIODE	v	all		FAIL	54	terminal1=nn,terminal2

Configuration fields for biaschk3:

- *Name: biaschk3
- *Type: MOS
- *Monitor: i
- *Simulation: tr
- *Terminal1: nd
- Terminal2: (empty)
- Element Names: /I5/I0
- Model Names: cmospn
- Exception: (empty)
- Subckt Definition Names: (empty)
- Limit: Su
- Noise: (empty)
- Min: (empty)
- Max: (empty)
- Tstart: (empty)
- Tstop: (empty)
- Interval: (empty)
- Autostop: (empty)

Figure 150 Monitoring element and model

- **Subckt:** When your violation setup is to monitor subcircuit instances, the **Type** field value is **SUBCKT** and the UI is configured as shown in [Figure 151 on page 311](#). You need to select the **Subckt Instance Names** before the **Pin1**, **Pin2** values can be input.

Note: The **Pin1**, **Pin2** input fields are automatically loaded according to the selection of the subcircuit instance name.

When you input several subcircuit instances (subcircuits) you can define more than one subcircuit name in the **Subcircuit Instance Names** field with comma (,) delimiters. The first instance name is used to load **Pin1**, **Pin2** selection box choices; the other subcircuit instance names should have the same subcircuit pin definitions as the first one.

Chapter 14: HSPICE Violation Check (.BIASCHK)

Invoking and Setting up Violation Check

The dialog box displays a table of violation checks and configuration options for the Subcircuit instance monitor.

Name	Enabled	Type	Monitor	Simulation	Element/Model/Subckt Names/Expression	Status	Violation Num	Others
biaschk1	Yes	Region	linear, sat	op	name="x".m*	FAIL	104	autostop=No
biaschk2	Yes	SUBCKT	v	tr	sktinsts=/I0./I1./I2	FAIL	118	pin1=out1,limit=0.2,aut
biaschk4	Yes	NMOS	i	tr	name=/I5/M0,mname=cmosn	FAIL	3568	terminal1=nd,terminal2
biaschk3	Yes	MOS	i	tr	name=/I5/M0,mname=cmosn	FAIL	39	terminal1=nd,limit=5u
biaschk6	Yes	Expression	op		expression=(sub(x5,out1)(x5,m0)	FAIL	1	min=0,max=0.000003n
biaschk7	Yes	Expression	all		expression=V(out31)/V(out21)	FAIL	197	min=0.1,max=1,tstop=
biaschk8	Yes	CAP	i	all		FAIL	2	terminal1=n1,limit=1fa
biaschk9	Yes	PMOS	w	all	name="x".m*,mname=cmosp	FAIL	5236	limit=1u,max=1u,auto
biaschk5	Yes	RESISTOR	v	dc	name=/I2/R0./I0/R0./I1/R0	FAIL	213	terminal1=n1,limit=1p
biaschk10	Yes	DIODE	v	all		FAIL	54	terminal1=nn,terminal2

Configuration fields for biaschk2:

- *Name: biaschk2
- *Type: SUBCKT
- *Monitor: v
- *Simulation: tr
- *Subckt Instance Names: /I0./I1./I2
- *Pin1: out1
- Limit: 0.2
- Noise:
- Min:
- Max:
- Tstart:
- Tstop:
- Interval:
- Autostop:

Figure 151 Subcircuit instance monitor

- Region:** When your violation setup is to monitor a region of a circuit, the UI is configured as shown in Figure 152. Select **Region** as the **Type** field value. The **Monitor** selection box is replaced by **cutoff**, **linear**, and **saturation** check boxes which can be user-selected. The **Terminal1**, **Terminal 2** input fields are removed from the form.

The dialog box displays a table of violation checks and configuration options for the Region monitor.

Name	Enabled	Type	Monitor	Simulation	Element/Model/Subckt Names/Expression	Status	Violation Num	Others
biaschk1	Yes	Region	linear, sat	op	name="x".m*	FAIL	104	autostop=No
biaschk2	Yes	SUBCKT	v	tr	sktinsts=/I0./I1./I2	FAIL	118	pin1=out1,limit=0.2,aut
biaschk4	Yes	NMOS	i	tr	name=/I5/M0,mname=cmosn	FAIL	3568	terminal1=nd,terminal2
biaschk3	Yes	MOS	i	tr	name=/I5/M0,mname=cmosn	FAIL	39	terminal1=nd,limit=5u
biaschk6	Yes	Expression	op		expression=(sub(x5,out1)(x5,m0)	FAIL	1	min=0,max=0.000003n
biaschk7	Yes	Expression	all		expression=V(out31)/V(out21)	FAIL	197	min=0.1,max=1,tstop=
biaschk8	Yes	CAP	i	all		FAIL	2	terminal1=n1,limit=1fa
biaschk9	Yes	PMOS	w	all	name="x".m*,mname=cmosp	FAIL	5236	limit=1u,max=1u,auto
biaschk5	Yes	RESISTOR	v	dc	name=/I2/R0./I0/R0./I1/R0	FAIL	213	terminal1=n1,limit=1p
biaschk10	Yes	DIODE	v	all		FAIL	54	terminal1=nn,terminal2

Configuration fields for biaschk1:

- *Name: biaschk1
- *Type: Region
- ☐ cutoff ☒ linear ☒ saturation
- *Simulation: op
- Element Names: x*.m*
- Model Names:
- Exception:
- Subckt Definition Names:
- Tstart:
- Tstop:
- Interval:
- Autostop:

Figure 152 Monitoring a Region

Chapter 14: HSPICE Violation Check (.BIASCHK)

Invoking and Setting up Violation Check

- **Length and Width:** When your violation setup is to monitor a length or width of an element, the UI is configured as shown in [Figure 153](#) for a PMOS circuit. Select the transistor circuit **Type** field value. Select **l** or **w** from the **Monitor** selection box. The **Terminal1**, **Terminal 2** input fields are removed from the form.

The screenshot shows the 'HSPICE Violation Check' dialog box. At the top, there is a table titled '.BIASCHKs' with columns: Name, Enabled, Type, Monitor, Simulation, Element/Model/Subckt Names/Expression, Status, Violation Num, and Others. The table lists several violations, including biaschk1 through biaschk10. Below the table, there are input fields for configuring a specific violation. The 'Name' field is set to 'biaschk9'. The 'Type' dropdown is set to 'PMOS'. The 'Monitor' dropdown is set to 'w'. The 'Simulation' dropdown is set to 'all'. Below these, there are fields for 'Element Names' (set to 'x*.m*'), 'Model Names' (set to 'cmosp'), 'Exception' (empty), 'Subckt Definition Names' (empty), 'Limit' (set to '1u'), 'Noise' (empty), 'Min' (empty), 'Max' (set to '1u'), 'Tstart' (empty), 'Tstop' (empty), 'Interval' (empty), and 'Autostop' (checked). At the bottom, there are buttons for 'Add', 'Change', 'Clear', 'Options', 'Close', and 'Help'.

Name	Enabled	Type	Monitor	Simulation	Element/Model/Subckt Names/Expression	Status	Violation Num	Others
biaschk1	Yes	Region	linear, sat...	op	name="x".m*	FAIL	104	autostop=No
biaschk2	Yes	SUBCKT	v	tr	skinsts=/0,/1,/2	FAIL	118	pin1=out1.limit=0.2,aut
biaschk4	Yes	NMOS	i	tr	name=/I5/M0,mname=cmosp	FAIL	3966	terminal1=nd,terminal2
biaschk3	Yes	MOS	i	tr	name=/I5/M0,mname=cmosp	FAIL	39	terminal1=nd.limit=5u
biaschk6	Yes	Expression	op	all	expression=Isub(xI5,out1(xI5,m0)	FAIL	1	min=0,max=0.000003n
biaschk7	Yes	Expression	all	all	expression=V(out31)V(out21)	FAIL	197	min=0.1,max=1,tstop=
biaschk8	Yes	CAP	i	all		FAIL	2	terminal1=n1.limit=1fa
biaschk9	Yes	PMOS	w	all	name="x".m*.mname=cmosp	FAIL	5236	limit=1u,max=1u,autob
biaschk5	Yes	RESISTOR	v	dc	name=/I2/R0,/I0/R0,/I1/R0	FAIL	213	terminal1=n1.limit=1ps
biaschk10	Yes	DIODE	v	all		FAIL	54	terminal1=nn,terminal2

*Name: biaschk9 *Type: PMOS *Monitor: w *Simulation: all

Element Names: x*.m* Model Names: cmosp

Exception: Subckt Definition Names:

Limit: 1u Noise: Min: Max: 1u

Tstart: Tstop: Interval: Autostop: [x]

Add Change Clear Options Close Help

Figure 153 Width monitoring setup

- **Expression:** When your violation setup is to monitor an expression in your design, the UI is configured as shown in [Figure 154 on page 313](#) (for a PMOS circuit). Select the transistor circuit **Type** field value. Select **l** or **w** from the **Monitor** selection box. The **Terminal1**, **Terminal 2** input fields are removed from the form.

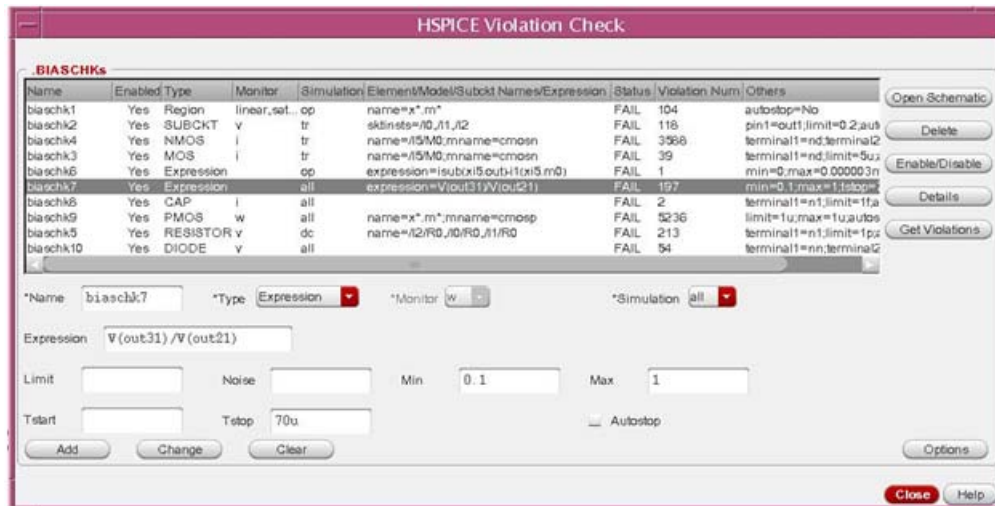


Figure 154 Expression monitoring setup

Setting Up .BIASCHK Commands and Options

This section describes procedures to set up HSPICE .BIASCHK commands and options.

Setting up a .BIASCHK Command

The HSPICE .BIASCHK commands setup steps are as follows:

1. Input a unique name for a .BIASCHK command in **Name** field. The setup form provides a default name for new .BIASCHK command when it displays the empty HSPICE Violation Check form or after you successfully add a .BIASCHK command (Step 5).
2. Select a type value from the **Type** selection box according to the violation checking you would like to do.
3. Select the **Monitor** type (*l*, *w*, *v*, *i*, etc.) or **cutoff**, **linear**, and **saturation** for the **Region** type, to configure the HSPICE Violation Check form.
4. Input other values according to HSPICE .BIASCHK command syntax (See [.BIASCHK](#) in the *HSPICE Reference Manual: Commands and Control Options*)
5. Click **Options** to open the HSPICE Options dialog ([Figure 149 on page 309](#)) to invoke options depending on the type of .BIASCHK monitoring you want to initiate.

6. See [Options](#) in Chapter 6 of this manual for full discussion of HSPICE simulation option usage.
7. Click the **Add** button to complete each .BIASCHK command setup. After successfully adding the setup, the new .BIASCHK command is shown in the **.BIASCHKs** report field.
8. To edit an added .BIASCHK command setup, highlight the line in the **BIASCHKs** report table, and click **Change**.
9. Click **Clear** to remove all filled-in fields on the setup above to create a new .BIASCHK command.
10. Use the **Delete** to remove a highlighted existing .BIASCHK command from the HSPICE-ADE environment.
11. Use the **Enable/Disable** button to enable/disable a highlighted .BIASCHK command. Disabling comments out the command from the netlist.

Netlisting and Running .BIASCHK Simulations

After completing HSPICE .BIASCHK commands setup, select **Simulation > Netlist and Run** on the Environment console to netlist and run a simulation of all commands.

When the simulation run has finished, all the .biaschk setup statements are generated in a file named *biaschks* in the path:

```
./simulation/design/HSPICE/schematic/netlist/biaschks)
```

Displaying/Printing Violation Check Results

After successfully running an HSPICE violation check simulation, HSPICE-ADE automatically reads the HSPICE .BIASCHK command outputs and provides summary information, which you can also print to a file.

HSPICE-ADE automatically displays each .BIASCHK command violation check summary in HSPICE Violation Check form's **.BIASCHKs** report field using columns **Status** and **Violation num** ([Figure 155 on page 315](#)).

You can also print an HSPICE Violation Check summary by selecting **Results > Print > HSPICE Violations Summary** in the Environment console. For example,

Chapter 14: HSPICE Violation Check (.BIASCHK)

Displaying/Printing Violation Check Results

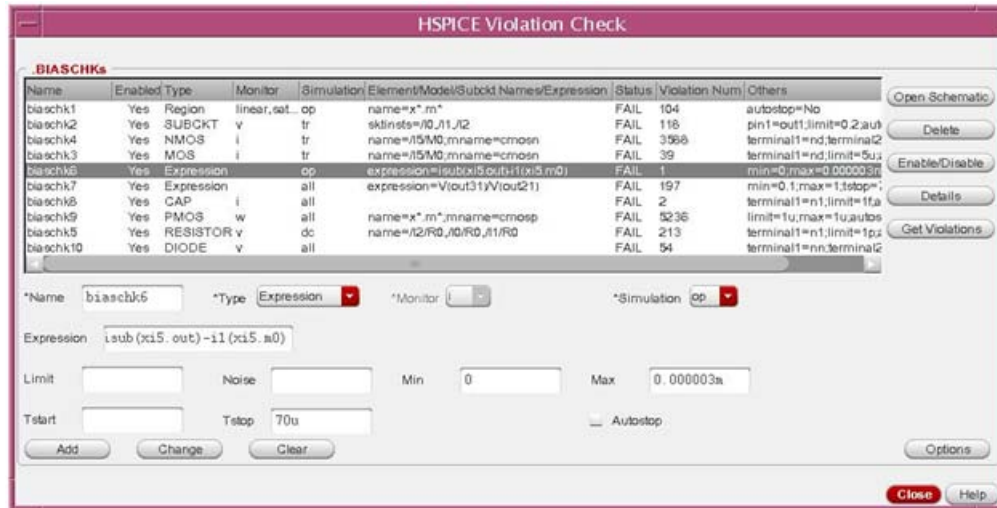


Figure 155 Status and Violation Number columns with biaschk6 highlighted

Name	Type	Monitor	Simulation	Element/Model/Subckt Names/Expression	Terminal/Node/Expression	Others	Status	Violation Number
biaschk1	Region	op	op	name="x".m"	no, op	limit=0.05	FAIL	104
biaschk10	DIODE	v	all		out1	limit=0.2	FAIL	54
biaschk2	SUBCKT	v	tr	sktinsts=/J3,/J1,/J2	nd	limit=5u	FAIL	118
biaschk3	MOS	i	tr	name=/J5/M0,mname=cmosn	nd, ng	limit=0, min=1a, tstart=0, tstop=20u	FAIL	39
biaschk4	NMOS	i	tr	name=/J5/M0,mname=cmosn	nl	limit=0	FAIL	356
biaschk5	RESISTOR	v	dc	name=/J2/R0,/J0/R0,/J1/R0	nl	limit=1p	FAIL	213
biaschk6	Expression	op	op	expression=isub(x15.out)-i1(x15.m0)	isub(x15.out)-i1(x15.m0)	min=0, max=0.000003a, tstop=70u	FAIL	1
biaschk7	Expression	all	all	expression=V(out31)/V(out21)	V(out31)/V(out21)	min=0.1, max=1, tstop=70u	FAIL	197
biaschk8	CAP	i	all			limit=1f	FAIL	2
biaschk9	PMOS	w	all	name="x".m", mname=cmosp		limit=1u, max=1u	FAIL	5236

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 156 Results Display Window with violations summary

These topics are discussed in the following sections:

- [HSPICE Violation Details Form](#)

HSPICE Violation Details Form

To view results with greater granularity, you can launch and print the HSPICE Violation Details form. Select the command of your choice in the **BIASCHKs** report field and click the **Details** button on the right side of the report field to display the detailed report in a dynamic window that can also access the design schematic to identify elements with violations ([Figure 157](#)).

Chapter 14: HSPICE Violation Check (.BIASCHK)

Displaying/Printing Violation Check Results

HSPICE Violation Details

.BIASCHKs Details

BIASCHK Name: **biaschk1**

Type: Region Monitor: linear,saturation Simulation: op

Element-name: x*.m* autostop: No

element-name	type	range	method	time	operation	model-name	subckt-name	Analysis
/6/M1	mos	linear	range	0.	leave	cmosp	inverter	op
/6/M0	mos	linear	range	0.	leave	cmosp	inverter	op
/15/M1	mos	linear	range	0.	leave	cmosp	inverter	op
/15/M0	mos	linear	range	0.	leave	cmosp	inverter	op
/10/M1	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M2	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M9	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M10	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M3	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M4	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M5	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M6	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M14	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M13	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M12	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M11	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/10/M7	mos	linear	range	0.	enter	cmosp	divider_by_2	op
/10/M8	mos	linear	range	0.	enter	cmosp	divider_by_2	op
/10/M15	mos	linear	range	0.	enter	cmosp	divider_by_2	op
/10/M16	mos	linear	range	0.	enter	cmosp	divider_by_2	op
/11/M1	mos	linear	range	0.	enter	cmosp	divider_by_2	op
/11/M2	mos	linear	range	0.	enter	cmosp	divider_by_2	op
/11/M9	mos	linear	range	0.	enter	cmosp	divider_by_2	op
/11/M10	mos	linear	range	0.	enter	cmosp	divider_by_2	op
/11/M3	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/11/M4	mos	linear	range	0.	leave	cmosp	divider_by_2	op
/11/M5	mos	linear	range	0.	leave	cmosp	divider_by_2	op

Color: **Red** Highlight Clear Print

Close Help

Figure 157 HSPICE Violation Details form

The HSPICE Violation Details window shows detailed outputs of the selected .BIASCHK command with in table format. This HSPICE Violation Details form shows the “biaschk1” violation details.

The form features the following UI controls. A list of steps follows.

- **.BIASCHKs Details** table: Displays .BIASCHK command violation detail outputs according to the **BIASCHK Name** selection box value. The columns in this table change according to the selected bias check type and each column in the table can be sorted.
- **Color**: Select a highlight color, from the choices of **Yellow**, **Red**, **Blue**, **Orange**, and Green. The selected color will be used to identify the device or element in violation of design rules. In the example below, the selected color is red.

- **Highlight** button: Links the selected device in the **.BIASCHKs Details** table to the schematic. (The schematic window must be open.) The highlighting color is the one selected in the **Color** selection box. You can apply different colors to various violations in the schematic by changing the **.BIASCHK** command under study in the **.BIASCHKs Details** selection box.
- **Clear** button: Removes all highlight violation devices from the schematic.
- **Print** button: Prints the selected **.BIASCHK** command violation detail outputs to the Results Display Window using the format in the **.BIASCHK Details** table.

These topics are discussed in the following sections:

- [Using the HSPICE Violation Details with a Schematic](#)
- [Printing .BIASCHKs Violation Details](#)

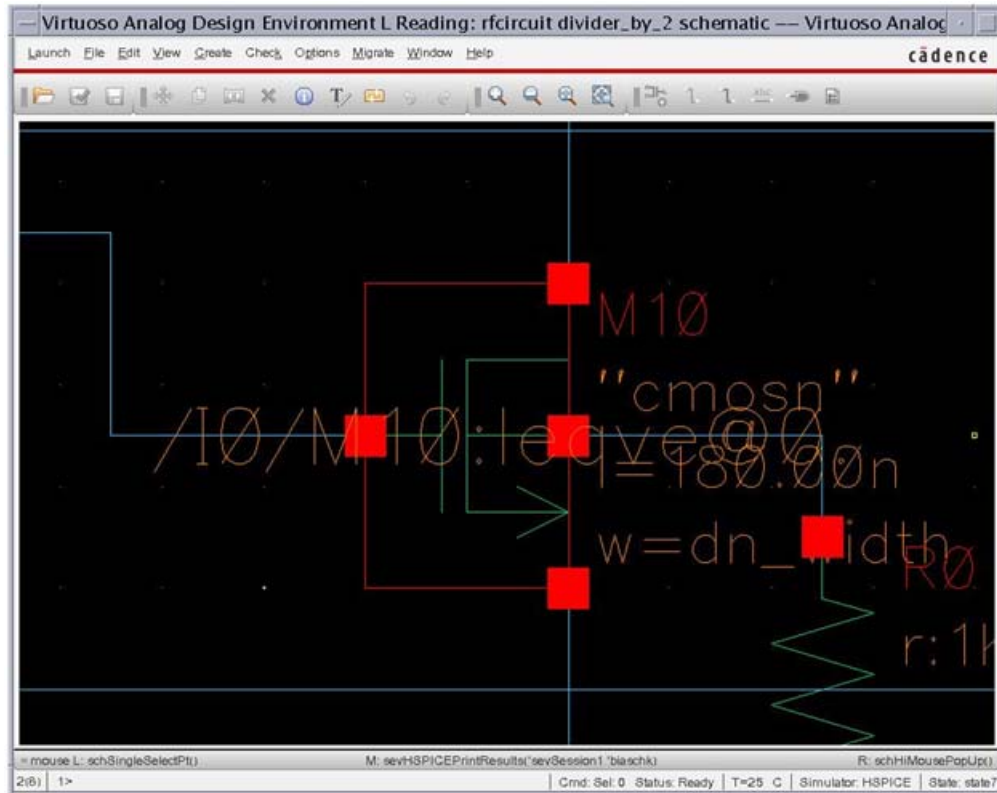
Using the HSPICE Violation Details with a Schematic

Follow these steps to use the Violation Details controls with a schematic design.

1. On the HSPICE Violation Check form, click the **Open Schematic** button to open the design in the schematic editor.
Note: If you select a device to highlight before opening the schematic, ADE reports the following warning message:
`*WARNING* The schematic window doesn't open,
can't execute highlighting
You cannot highlight an expression; ADE reports the following
message: *Error* Can't get element to highlight`
2. Select the violation device in HSPICE Violation details form **.BIASCHKs Details** report field.
3. Select a highlight color in the **Color** selection box.
4. Click **Highlight** to pop-up the schematic and hierarchically display the violation region or element of interest. [Figure 158 on page 318](#) shows violation device (in red overlaying the circuit) in the schematic window and probe of violation information on the device.

Chapter 14: HSPICE Violation Check (.BIASCHK)

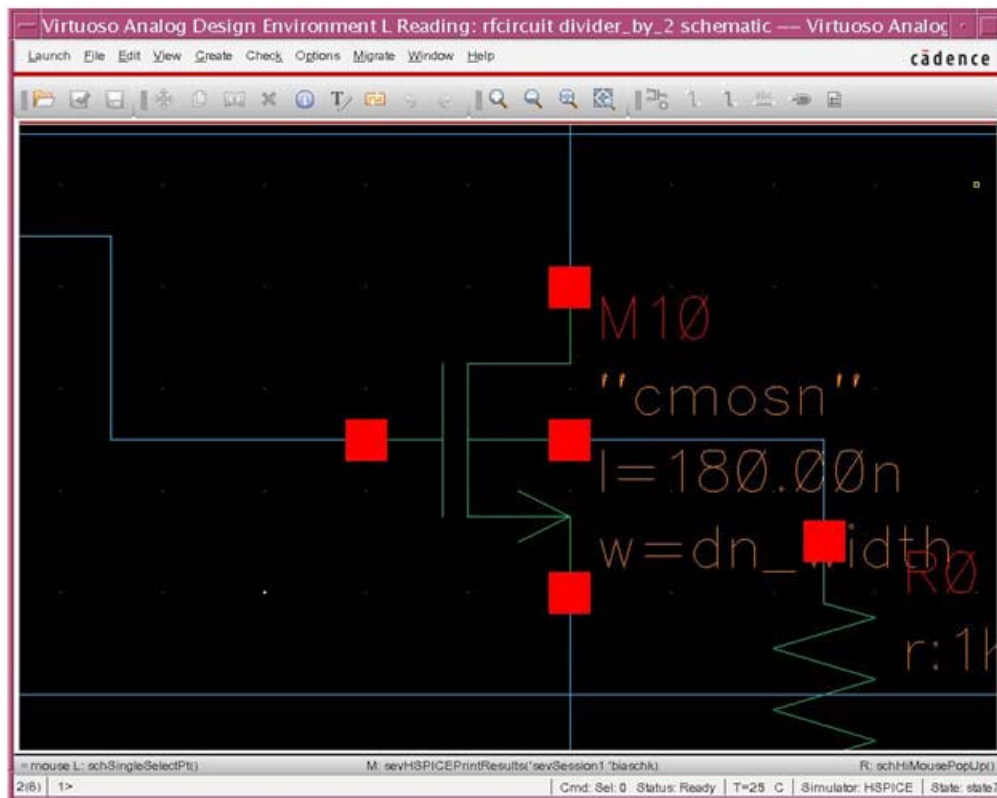
Displaying/Printing Violation Check Results



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 158 Element /I0/M/10highlighted violation (red outlined rectangle)

5. Click the **Clear** button on the HSPICE Violation Details form to remove the violation from the schematic.



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

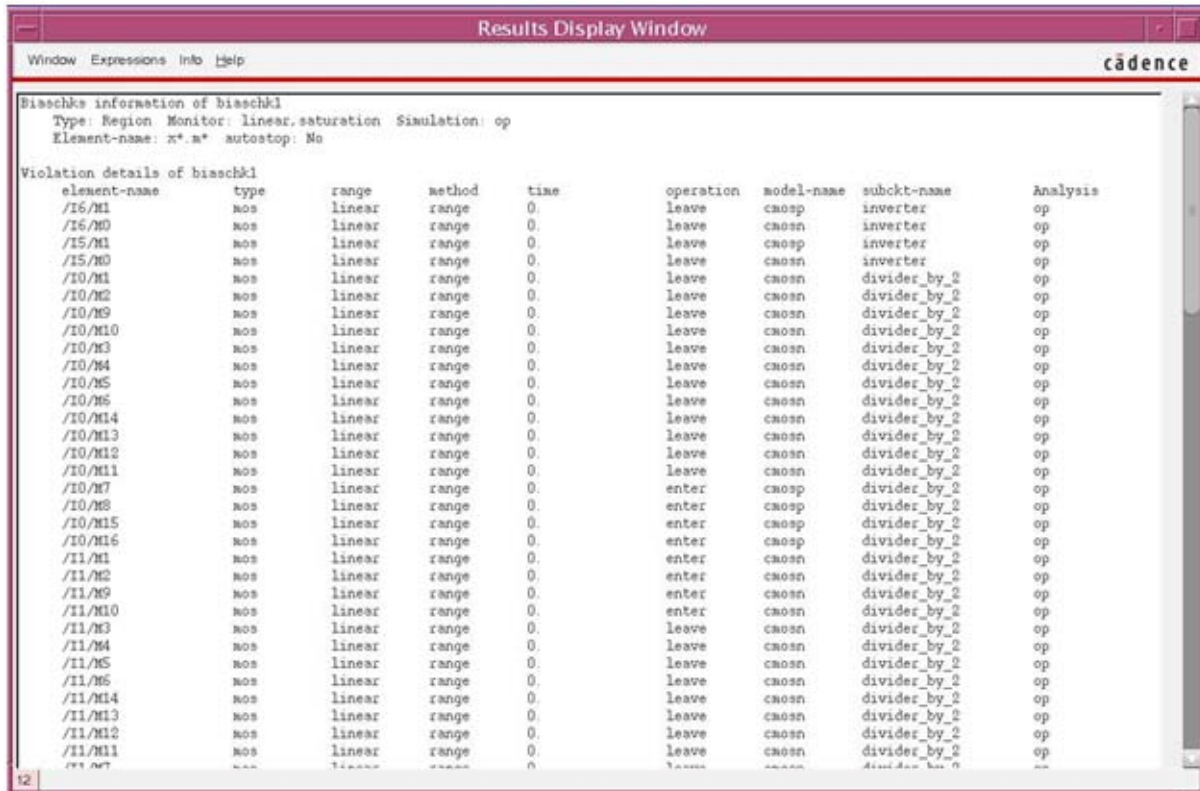
Figure 159 Violation cleared from schematic

Printing .BIASCHKs Violation Details

To print a report of the BIASCHK Details select **BIASCHK Name** in the HSPICE Violation Details form and click the **Print** button to print the .BIASCHK command violation details using the same tabular format. [Figure 160 on page 320](#) shows the printout window for “biaschk1”:

Chapter 14: HSPICE Violation Check (.BIASCHK)

Displaying/Printing Violation Check Results



Biaschk1 information of biaschk1
 Type: Region Monitor: linear.saturation Simulation: op
 Element-name: x*.m* autostop: No

Violation details of biaschk1

element-name	type	range	method	time	operation	model-name	subckt-name	Analysis
/16/M1	mos	linear	range	0	leave	cnosp	inverter	op
/16/M0	mos	linear	range	0	leave	cnosp	inverter	op
/15/M1	mos	linear	range	0	leave	cnosp	inverter	op
/15/M0	mos	linear	range	0	leave	cnosp	inverter	op
/10/M1	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M2	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M9	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M10	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M3	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M4	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M5	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M6	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M14	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M13	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M12	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M11	mos	linear	range	0	leave	cnosp	divider_by_2	op
/10/M7	mos	linear	range	0	enter	cnosp	divider_by_2	op
/10/M8	mos	linear	range	0	enter	cnosp	divider_by_2	op
/10/M15	mos	linear	range	0	enter	cnosp	divider_by_2	op
/10/M16	mos	linear	range	0	enter	cnosp	divider_by_2	op
/11/M1	mos	linear	range	0	enter	cnosp	divider_by_2	op
/11/M2	mos	linear	range	0	enter	cnosp	divider_by_2	op
/11/M9	mos	linear	range	0	enter	cnosp	divider_by_2	op
/11/M10	mos	linear	range	0	enter	cnosp	divider_by_2	op
/11/M3	mos	linear	range	0	leave	cnosp	divider_by_2	op
/11/M4	mos	linear	range	0	leave	cnosp	divider_by_2	op
/11/M5	mos	linear	range	0	leave	cnosp	divider_by_2	op
/11/M6	mos	linear	range	0	leave	cnosp	divider_by_2	op
/11/M14	mos	linear	range	0	leave	cnosp	divider_by_2	op
/11/M13	mos	linear	range	0	leave	cnosp	divider_by_2	op
/11/M12	mos	linear	range	0	leave	cnosp	divider_by_2	op
/11/M11	mos	linear	range	0	leave	cnosp	divider_by_2	op
/11/M7	mos	linear	range	0	leave	cnosp	divider_by_2	op

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 160 Results Display Window using the HSPICE Violation Details format

Distributed Mode—Monte Carlo/Corner Analyses

Describes how to set up distributed jobs for these analyses in the HSPICE integration to the Cadence™ Virtuoso® Analog Design Environment.

This chapter describes how to generate the netlist files and scripts to submit jobs on a distributed environment and provides a method to integrate a user-distributed environment into the HSPICE integration to ADE and enable submission of distributed jobs.

The following topics are discussed in these sections:

- [HSPICE Distributed Jobs Mode](#)
- [HSPICE Monte Carlo Distributed Simulation](#)
- [HSPICE Corner Analysis Distributed Simulation](#)

HSPICE Distributed Jobs Mode

This section describes how to generate the netlist files and scripts to submit jobs across a distributed environment and provides a method to integrate a user-distributed environment in HSPICE-ADE to enable submission of distributed jobs.

On the Environment Console, select **Setup > HSPICE Distributed Mode** to open the HSPICE Distributed Setup form ([Figure 161](#)).

Chapter 15: Distributed Mode—Monte Carlo/Corner Analyses

HSPICE Distributed Jobs Mode

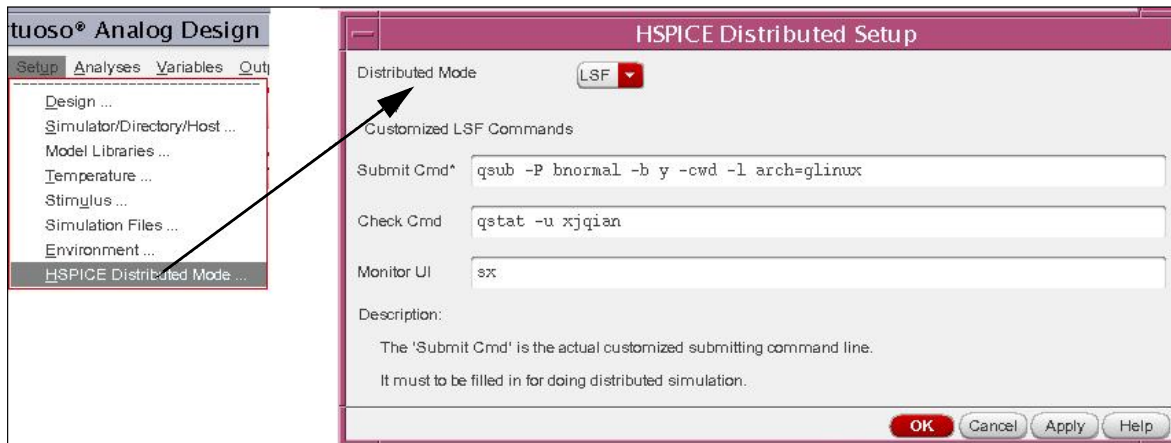


Figure 161 HSPICE Distributed Setup form

The form opens by default to the Load Sharing Facility (**LSF**) **Distributed Mode**.

When you click the following input fields, the **Description** text changes to user-customized description text.

- **Submit Cmd** (required) allows users to fill in customized submitting command to perform distributed simulation.
- **Check Cmd** (optional) allows users to fill in the actual command on checking parallel job status.
- **Monitor UI** (optional) invokes a user's customized UI application.
- Click **OK** to complete the setup.

After you complete HSPICE Distributed Setup form, the **Enable Distributed Jobs** feature becomes available on the Monte Carlo Setup form. (See [HSPICE Monte Carlo Distributed Simulation](#) in this chapter for usage model after setting up HSPICE distributed processing.)

Manually Setting Distributed Mode Environment Variables

Alternatively, you can use the following method to add HSPICE DP information by setting HSPICE Distributed Mode partition variables. Add these HSPICE environment variables to the *HSPICE.cdsenv*:

HSPICE.DM	mode	string	"LSF"	nil
HSPICE.DM	custSubCmd	string	" "	nil
HSPICE.DM	custChkCmd	string	" "	nil
HSPICE.DM	custMoniCmd	string	" "	nil

These can be set locally through `envSetVal()` after loading the HSPICE-ADE interface. For example:

```
envSetVal("HSPICE.DM" "custSubCmd" 'string "bsub -P bnormal -M  
`whoami`@synopsys.com -m e -b y -cwd -l  
arch=glinux,os_version=\"WS4.0\",emt64=1")
```

HSPICE Monte Carlo Distributed Simulation

After you complete the setup and **OK** the HSPICE Distributed Setup form, the **Enable Distributed Jobs** feature becomes available on the Monte Carlo Setup form. Select **Tools > HSPICE Statistical Analysis** in the Environment Console window to open HSPICE Monte Carlo Analysis window.

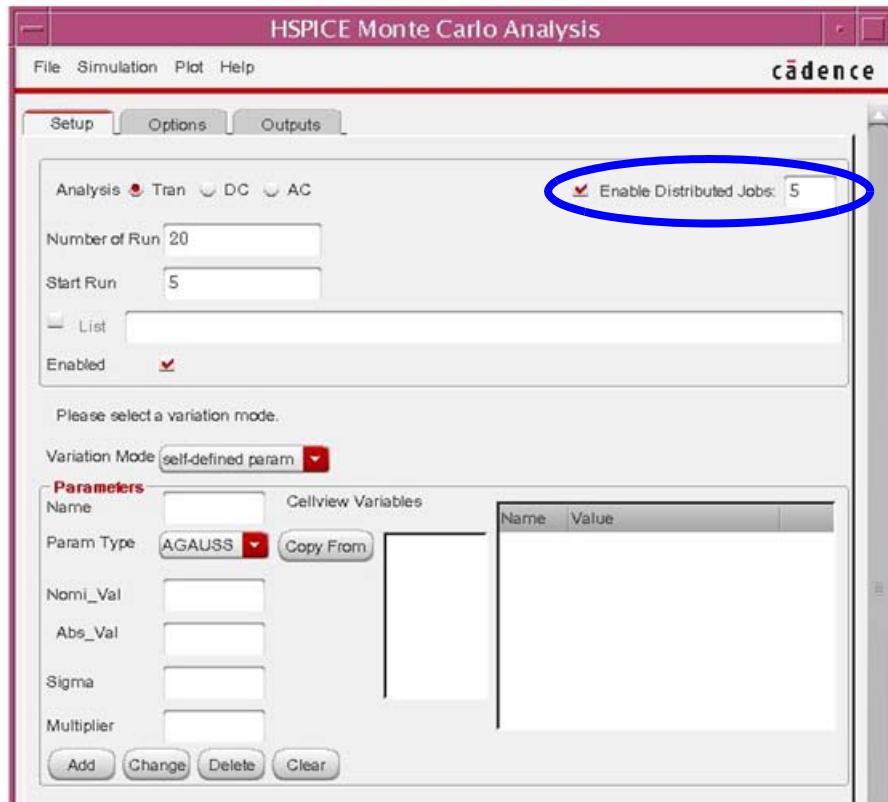


Figure 162 Setup tab on the HSPICE Monte Carlo Analysis main window

Chapter 15: Distributed Mode—Monte Carlo/Corner Analyses

HSPICE Monte Carlo Distributed Simulation

Run parallel jobs as follows:

1. Enter the number of parallel jobs you want to submit to do a distributed simulation.
2. Click the **Netlist** menu or the related button in the **Outputs** tab to do netlisting and display the HSPICE DP netlist file (Figure 163 on page 324) generated for parallel Monte Carlo simulations (which are ready to be run through submitting jobs).



```

* Generated for: HSPICE
* Generated on: Aug 10 10:06:01 2010
* Design library name: Monte
* Design cell name: montetest
* Design view name: schematic
.global vdd!
.param
+ deltaL=AGAUSS(0,0.1,1,2)
+ deltaW=AGAUSS(0,0.2,1,1)
+ vdd=3.3
.temp 25
.option ARTIST=2 PSF=2
.option MTHRESH=0 LIS_NEW=1
.lib '...../level49.lib' nmodel
.include '...../meas.file'

* Library name: Monte
* Cell name: montetest
* View name: schematic
m1 out in 0 0 nch w='(3+deltaW)*1e-6' l='(2+deltaL)*1e-6'
m2 out in vdd! vdd! pch w='(5+deltaW)*1e-6' l='(2+deltaL)*1e-6'
v1 vdd! 0 dc=3.3
v0 in 0 dc=0 PULSE ( 0 'vdd' 1e-6 1e-9 1e-9 1e-6 2.5e-6 ) AC=1
.option hier_delim=1
.probe ac v(out)
+ i1(m1)
.probe dc v(out)
+ i1(m1)
.probe tran v(out)
+ i1(m1)
< ----- monte carlo netlist for job = 1 -----
< .tran 10e-9 30e-6 start=0.0
< + SWEEP MONTE=list (5 6 7 8)
< .dc v1 2.5 3.5 100e-3
< + SWEEP MONTE=list (1 2 3)
< .ac DEC 10.0 1.0 100e6
< + SWEEP MONTE=list (1)
> ----- monte carlo netlist for job = 2 -----
> .tran 10e-9 30e-6 start=0.0
> + SWEEP MONTE=list (9 10 11 12)
> .dc v1 2.5 3.5 100e-3
> + SWEEP MONTE=list (4 5 6)
> .ac DEC 10.0 1.0 100e6

```

© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 163 Netlist file for each Monte Carlo job

The HSPICE netlist file window displays the difference for each HSPICE Distributed Job Netlist file.

- Click the **Run** menu or the **Run** button in the **Outputs** tab to submit the parallel jobs and pop-up the HSPICE Distributed Task form (Figure 164) to submit jobs.

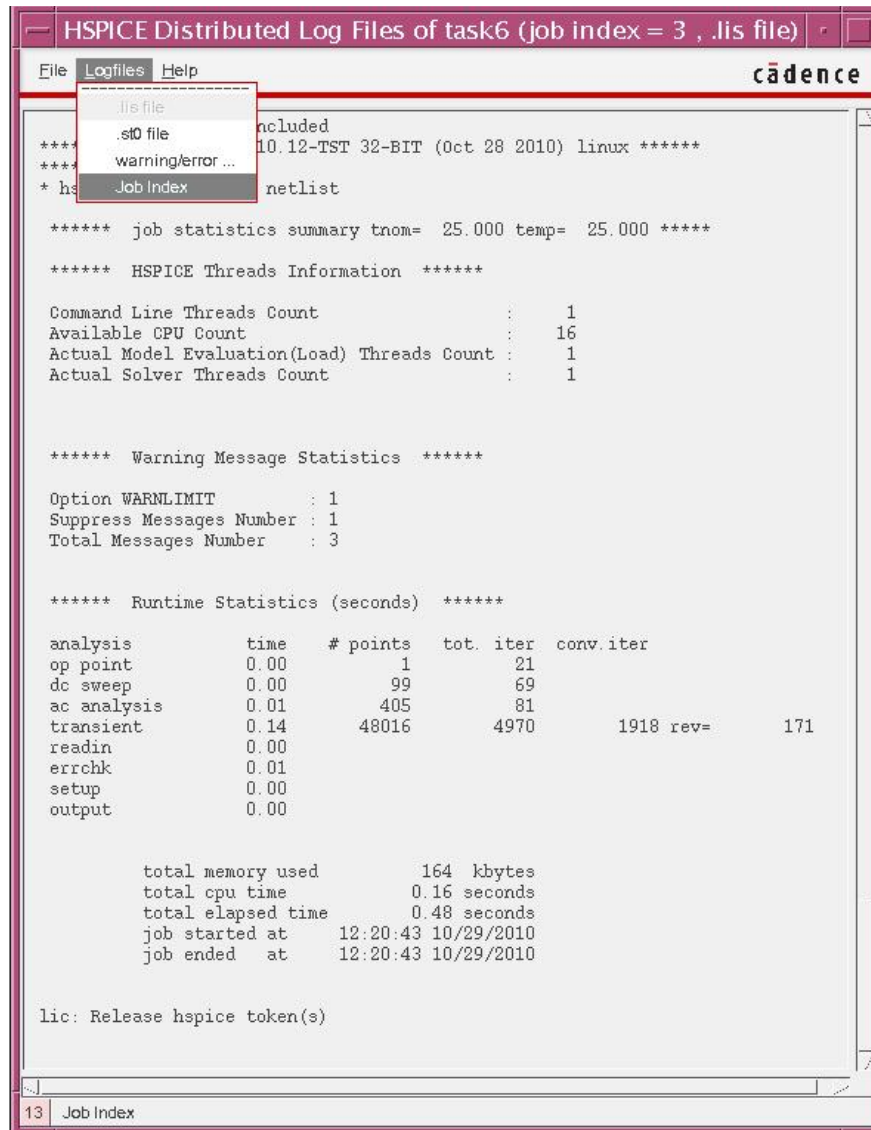
Name	Module	Type	Submit Time	Finished	Status	Job num - Run/Pass/Fail	Results Di
task1	MonteCarlo	LSF	Oct 29 11:59:28	YES	SUCC	4 - 0 / 4 / 0	/remote/hs
task2	MonteCarlo	LSF	Oct 29 12:11:05	YES	SUCC	5 - 0 / 5 / 0	/remote/hs
task6	MonteCarlo	LSF	Oct 29 12:20:22	YES	SUCC	5 - 0 / 5 / 0	/remote/hs
task3	MonteCarlo	LSF	Oct 29 13:00:14	YES	SUCC	4 - 0 / 4 / 0	/remote/hs
task4	MonteCarlo	LSF	Oct 29 13:43:27	NO		5 - 5 / 0 / 0	/remote/hs
task5	MonteCarlo	LSF	Oct 29 13:47:25	NO		5 - 5 / 0 / 0	/remote/hs

Figure 164 HSPICE Distributed Tasks form

- Use the HSPICE Distributed Tasks form to submit HSPICE Monte Carlo distributed task jobs; it also can show the existing task status.
 - The **Submit** button is used to submit the input or any of the selected tasks listed under the Task Name column.
 - The **Enable to overwrite existing task results** check box is used to run the submitted task in the same existing task results directory.
 - The **Check Task** button is used to check the task status of selected task in the HSPICE DP Tasks report field.
 - The **Task Log** button is used to check the jobs status of selected task in the **HSPICE Distributed Tasks** report field.
 - The **Output Log** button is used to show each job's output log window, (Figure 165 on page 326).
 - The **Custom UI** button is used to invoke a user-defined "Monitor UI" (any defined custom distributing environment).
- Click the **Output Log** button to display the HSPICE DP job Simulation log.

Chapter 15: Distributed Mode—Monte Carlo/Corner Analyses

HSPICE Monte Carlo Distributed Simulation



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 165 HSPICE DP Log Files of task# window showing Logfiles Dropdown with Job Index selected

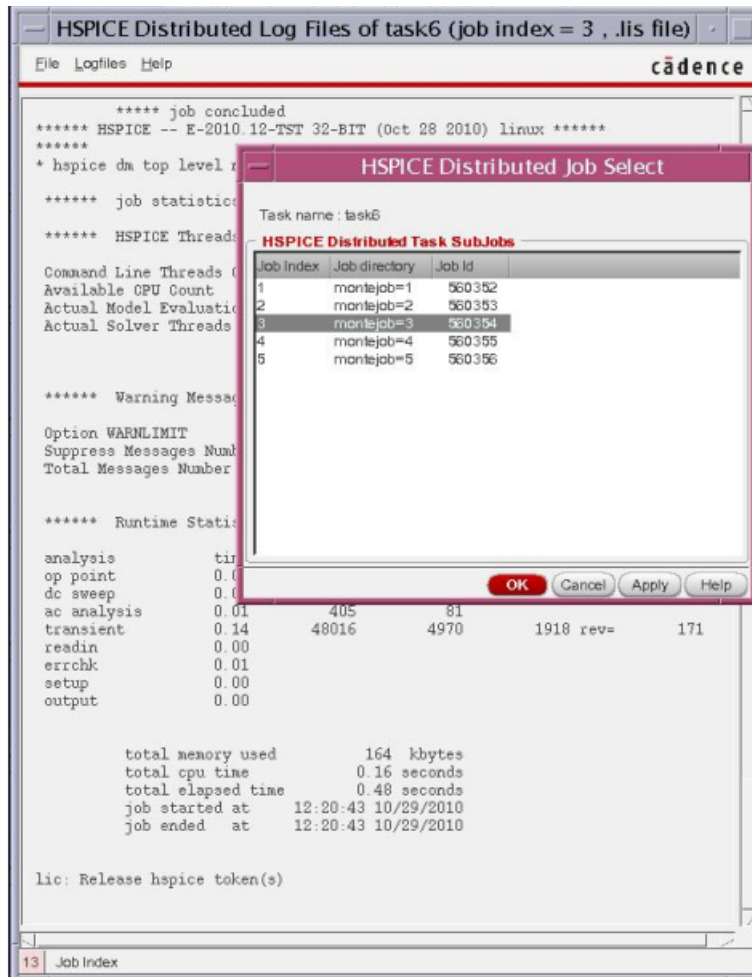
Distributed Log File Notes:

The Log Files of task# window

1. Provides a new task name when displayed, by default
2. Automatically loads existing task information when first creating and displaying

3. Automatically invokes the **Check Task** command for all running tasks when displayed

Figure 165 shows an HSPICE Distributed Log Files of task6 (job index = 3, .lis file). You can use these windows to view all jobs' simulation output files including: .lis, .st0, and warnerr files. Use the HSPICE Distributed Job Select form Figure 166 to switch between different jobs.

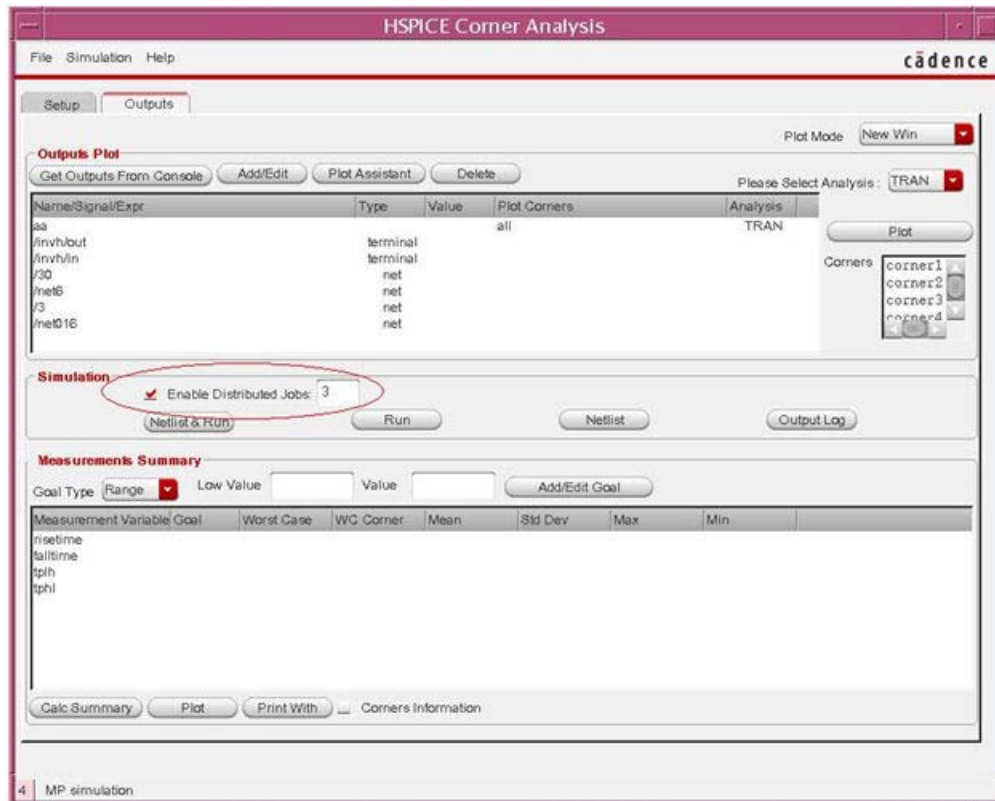


© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 166 The HSPICE Distributed Log Files window overlaid with the HSPICE Distributed Job Select form

HSPICE Corner Analysis Distributed Simulation

After completing HSPICE Distributed Setup form, the **Enable Distributed Jobs** feature becomes enabled on the HSPICE Corner Analysis **Outputs** tab. **Select Tools > HSPICE Corner Analysis** in the Environment console window to open HSPICE Corner Analysis window. [Figure 167](#) shows HSPICE Corner Analysis window **Outputs** form tab with **Enable Distributed Jobs** check box selected:



© 2007, Cadence Design Systems, Inc. All rights reserved worldwide. Printed with permission.

Figure 167 Enable Distributed Jobs check box available

To use the distributed processing capability:

1. Enter the number of jobs you want to submit for parallel simulation.

2. Click the **Netlist** menu or the related button in the **Outputs** tab to do netlisting and display the HSPICE DP netlist file (Fig. 9) generated for parallel HSPICE Corner Analysis simulations (which are ready to be run through submitting jobs).

Chapter 15: Distributed Mode—Monte Carlo/Corner Analyses
HSPICE Corner Analysis Distributed Simulation

Adding a CustomExplorer™ Menu to the HSPICE Integration

This Appendix discusses how to modify the HSPICE.menus file to add the Synopsys CustomExplorer wave view tool to the Cadence™ Virtuoso® Analog Design Environment console menu bar.

The following procedure and sample files are presented:

- [Updating the HSPICE.menus File](#)
- [Sample HSPICE.menus Files for Versions 51 and 61](#)

Updating the HSPICE.menus File

In order to add a Synopsys Spice-Explorer (SX) link to the HSPICE/ADE Integration, the *HSPICE.menus* file must be updated.

Note: Set up the SX-CDS-Link first and then modify the menu files. Read the {sx_install_path} /packages/SX-CDS-Link/sxcds.pdf documentation for instructions on how to set up WaveView for ADE. Essentially, you need to put the following in your .cdsinit file:

```
load("/path_to/sx_link.ile" "sandwork")
load("/path_to/sx_menu.il")
load("/path_to/sx_user.il")
```

For the SX-ADE menu to show up in other simulator integrations you should copy the *simui.menus* file into the local directory, so that you include both the *HSPICE.menus* and the *simui.menus* files.

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Updating the HSPICE.menu File

Use the following procedure to update the *HSPICE.menu* file to enable the display of the SX-ADE menu.

1. Locate and edit the file:
`$CDSHOME/tools/dfII/local/menus/HSPICE.menu`
2. On line 160 (containing "&Help"), add the following line:
`"S&X-ADE"`
3. On line 171 (containing ")), add the same line:
`"S&X-ADE"`
4. On line 464 (containing "("&Help", add the following lines:

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Updating the HSPICE.menus File

```

("S&X-ADE"
~(
  ("Setup Environment"          sxAASetup())
  ("Display Probes" ?disable (sevNoResults ',session) ?subMenu
    (
      ("Transient" sxAADisplay(?session ',session ?type 1))
      ("DC"        sxAADisplay(?session ',session ?type 2))
      ("AC"        sxAADisplay(?session ',session ?type 3))
    )
  )
  ("Multi-Net Probe" ?disable (sevNoResults ',session)
?subMenu
    (
      ("Transient" sxAADirect(?session ',session ?type 1))
      ("DC"        sxAADirect(?session ',session ?type 2))
      ("AC"        sxAADirect(?session ',session ?type 3))
    )
  )
  ("Instant Probe" ?disable (sevNoResults ',session)
?subMenu
    (
      ("Transient" sxAAQuick(?session ',session ?type 1))
      ("DC"        sxAAQuick(?session ',session ?type 2))
      ("AC"        sxAAQuick(?session ',session ?type 3))
    )
  )
  ("Noise Analysis" ?disable (sevNoResults ',session)
?subMenu
    (
      ("Equivalent Output Noise" sxAANoise(?session
',session ?type 1))
      ("Squared Output Noise"    sxAANoise(?session
',session ?type 2))
      ("Equivalent Input Noise"  sxAANoise(?session
',session ?type 3))
      ("Squared Input Noise"     sxAANoise(?session
',session ?type 4))
      ("Noise Figure"           sxAANoise(?session
',session ?type 5))
    )
  )
  ("Load Additional Files" ?subMenu
    (
      ("Transient" sxAddAAFile(?session ',session ?type 1))
      ("DC"        sxAddAAFile(?session ',session ?type 2))
      ("AC"        sxAddAAFile(?session ',session ?type 3))
    )
  )
  ("Output List" ?disable (sevNoResults ',session)

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
?subMenu
(
  ("Transient"  sxAAOutputList(?session ',session
?type 1))
  ("DC"        sxAAOutputList(?session ',session ?type 2))
  ("AC"        sxAAOutputList(?session ',session ?type 3))
)
)
("Clear Probes"  sxAAClearProbe(?session ',session))
("New SX Window" sxAAStart(?session ',session))
("Clear Highlight"  sxCleanHilight())
))
```

5. Save and restart the session. The Environment Console should now display a menu in HSPICE-ADE called “SX-ADE”.

Sample HSPICE.menus Files for Versions 51 and 61

The following two sample *HSPICE.menus* files show the results updating the for both versions 51 and 61 of the Cadence tool.

The sample file menus are as follows:

- [Version 51](#)
- [Version 61](#)

Version 51

```
/*
 *
 *                               Copyright (C) 1995
 *                               Cadence Design Systems, ICDG - Analog Division
 *                               All Rights Reserved.
 *
 *
 *
 *
 * $Source: /cvsrep/group/artist/src/simui/simui.menus,v $
 *
 * $Author: amit $
 * $Revision: 1.3.2.5 $
 * $Date: 2005/04/04 08:51:13 $
 * $State: Exp $
 *
 */
/
*****
*****
 *
 * DESCRIPTION
 * -----
 *
 * This file contains the menu definitions for the Analog Artist
 * Simulation Environment. Menus appear on the banner of the
Main Window,
 * on the banner of the Schematic Window, when in Analog Artist
mode,
 * on the banner of the "Command Type-In" window, and on the side of
 * the Main Window as a row of icons.
 *
 * Banner menus are defined in two steps. First, for each window
banner,
 * the list of pulldowns is defined using commands
 * sevSetMainWindowPulldownMenus, sevSetSchematicPulldownMenus,
and
 * sevSetTypeInWindowPulldownMenus. Second, lists of menu items
for all
 * pulldowns are defined using the command sevSetMenuItemLists.
The icon
 * menu only needs its list of items defined using
sevSetMenuItemLists
 * ("Fixedmenu").
 *
 * Menu definitions can be in one of two forms, the "simple form" or
 * the "complete form". Even though the default menu definitions
given
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
* later in this file follow the complete form, the simple form is
* adequate for most user menu customizations.
*
* Simple Custom Menu Form
* -----
*
* The list of pulldowns is given simply as the list of the names
(strings)
* of the pulldowns. For example,
*
* (sevSetMainWindowPulldownMenus '(
*   "Setup"
*   "Simulation"
*   "Results"
* ))
*
* The list of menu items is given as a list of item-lists, with each
* item-list composed of two strings, the name (text) of the
item and
* its callback. In the following example the given callbacks
may not
* be real functions. Also, the first line needs to be there as
given.
*
* (sevSetMenuItemLists (lambda (session name) (case name
*
*   ( "Setup" '(
*     ( "Design " "setDesignCB() " )
*     ( "Analyses" "setAnalysesCB() " )
*     ( "Variables" "setVariablesCB() " )
*     ( "Environment" "SetEnvironmentCB() " )
*   ))
*
*   ( "Simulation" '(
*     ( "Run      " "runSimulationCB() " )
*     ( "Stop    " "stopSimulationCB() " )
*   ))
*
*   ( "Results" '(
*     ( "Plot    " "plotResultsCB() " )
*     ( "Print   " "printResultsCB() " )
*     ( "Save    " "saveResultsCB() " )
*   ))
* ))
*
* In order to specify a slider (second-level) menu, the second
string
* of the menu item can be replaced by a list of name-callback
string
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

* pairs. For example, the last pulldown from the previous example
* could take the following form:
*
*      ( "Results" (
*      ( "Plot      "plotResultsCB() )
*      ( "Print"
*      (
*      ( "Tran" "printTranResultsCB() )
*      ( "AC      "printACResultsCB() )
*      ( "DC      "printDCResultsCB() )
*      ))
*      ( "Save      "saveResultsCB() )
*      ))
*
*
* Complete Custom Menu Form
* -----
*
* The complete custom menu form allows users to specify the
following:
* For each pulldown, an optional menu-items pointer name when
different
* from the pulldown's name (text). For each menu item, an optional
* disabling (greying-out) condition, an optional flag on whether
the
* item brings up a form and an optional submenu (slider) argument.
* Also, callbacks and disabling conditions can be specified either
* as strings or as (function-call) lists. For examples, look at the
* actual default menu definitions below.
*
*
* Multi-Level Customization
* -----
*
* In addition to using the command sevSetMenuItemLists which
(re)defines
* all menu item lists, users may use the command
sevAddMenuItemLists
* (same format as sevSetMenuItemLists) to add to the existing
item-list
* definitions. In this way, users may add private pulldowns on
top of site
* customizations, without having to repeat the site definitions.
*
*
* Direct-Plot Menu Customization
* -----
*
* The special command sevDirectPlotMenu is provided for easier

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
* setting of the entries of the "Direct Plot" (slider) menu. It
* must be called inside sevSetMenuItemLists or
sevAddMenuItemLists.
* The entries are specified as lists of three fields: a string
* indicating the name of the menu item, a symbol indicating the
* callback function and a symbol indicating the type of results
* required. The type can be a list of symbols, when more than one
* types are required. An optional fourth argument t indicates that
* a form will be brought up by the callback. As a note, the callback
* functions take two arguments, the environment session and the
* schematic window.
* As an example,
*
* (sevAddMenuItemLists (lambda (session name) (case name
*   ("Direct Plot"
*    (sevDirectPlotMenu session '(
*      ("Transient"transientDirectPlottran)
*      ("Transient-DC"transientMinusDCDirectPlot (tran
dc_op))
*      ("AC Magnitude"ACMagnitudeDirectPlotac)
*      ("AC Phase"ACPhaseDirectPlotac)
*      ("FormBased"FormBasedDirectPlotxyzt)
*    )))
*   )))
*
*****
*****/
(sevSetMainWindowPulldownMenus
~(
  "Session"
  "Setup"
  "Analyses"
  "Variables"
  "Outputs"
  "Simulation"
  "Results"
  "Tools"
  "SX-ADE"
  "Help"
))
(sevSetSchematicPulldownMenus
~(
  ("- Analog Environment"?items "Session_Schematic")
  ("Setup      "?items "Setup_Schematic")
  "Simulation"
  "Results"
  ("Sim-Tools"?items "Tools")
```


Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

    "SX-ADE"
  ))
  (sevSetTypeInWindowPulldownMenus
    ` (
      ("Window" ?items "Session_Encap")
    )
  )
  (sevSetMenuItemLists
    (lambda
      (session name)
      (case
        name
        ("Session_Common"
          ` (
            ("Save State" ?callback (sevSaveState ',session) ?form t
?disable (sevNoEnvironment ',session))
            ("Load State" ?callback (sevLoadState ',session) ?form t
?disable (sevNoEnvironment ',session))
            ("Save Script" ?callback (snps_sevSaveOceanScript ',session)
?form t ?disable (sevNoEnvironment ',session))
            ("Options" ?callback (sevEditOptions ',session) ?form t)
          )
          ("Session"
            (cons
              ` ("Schematic Window" ?callback (sevOpenSchematic ',session)
?form t)
              (append
                (sevMenuItems session "Session_Common")
                ` (
                  ("What's New" ?callback (HSPICEDisplayNew) ?form t
?disable nil)
                  ("Reset" ?callback (sevReset ',session))
                  ("Quit" ?callback (sevQuit ',session))
                )
              )
            )
            ("Session_Schematic"
              (cons
                ` ("Simulation Window" ?callback (sevCreateMainWindow
',session) ?form t)
                (append
                  (sevMenuItems session "Session_Common")
                  ` (
                    ("Quit" ?callback (sevQuit ',session))
                  )
                )
              )
            )
            ("Session_Encap"
              ` (
                ("Close" ?callback (hiCloseWindow (hiGetCurrentWindow)))
              )
            )
          ("Setup_Common"

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

        ~(
            ("Simulator/Directory/Host" ?callback (sevChooseSimulator
            ',session) ?form t)
            ("Model Libraries" ?callback (sevHSPICEEditModels ',session)
            ?form t ?disable (sevNoEnvironment ',session))
            ("Temperature" ?callback (sevChooseTemperature ',session)
            ?form t)
            ;("Model Path" ?callback (sevMpuTool ',session) ?form t
            ?disable (sevNoEnvironment ',session))
            ("Environment" ?callback
            (sevChooseEnvironmentOptions ',session) ?form t ?disable
            (sevNoEnvironment ',session))
            ("Simulation Files" ?callback
            (_HSPICESetupSimulationFiles ',session) ?form t ?disable
            (sevNoEnvironment ',session))
        ))
        ("Setup"
            (cons
                ~("Design" ?callback (sevChooseDesign ',session) ?form t)
                (sevMenuItems session "Setup_Common"))
        )
        ("Setup_Schematic"
            (append
                ~(
                    ("Analyses" ?callback (sevEditSelectedAnas ',session) ?form
                    t)
                    ("Variables" ?callback (sevEditSelectedVars ',session) ?form
                    t)
                    ("Outputs" ?callback (sevEditSelectedOuts ',session) ?form
                    t)
                    ("Select On Schematic"
                        ?subMenu
                        (
                            ("Outputs To Be Plotted" ?callback
                            (sevChangeOutsOnSchematic ',session 'plot))
                            ("Outputs To Be Saved" ?callback
                            (sevChangeOutsOnSchematic ',session 'save))
                        ))
                    ("Save Options" ?callback (sevSaveOptions ',session) ?form t)
                )
                (sevMenuItems session "Setup_Common")
            ))
        ("Analyses"
            ~(
                ("Choose" ?callback (sevEditSelectedAnas
                ',session) ?form t ?disable (sevNoEnvironment ',session))
                ("Delete" ?callback (sevDeleteSelectedAnas ',session)
                ?disable (sevNoAnaSelections ',session))
                ("Enable" ?callback (sevActivateSelectedAnas ',session t)

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration
Sample HSPICE.menus Files for Versions 51 and 61

```
?disable !(sevNonActiveSelectedAna ',session))
    ("Disable" ?callback (sevActivateSelectedAnas ',session
nil) ?disable !(sevActiveSelectedAna ',session))
    ))
    ("Variables"
    ~(
        ("Edit" ?callback (sevEditSelectedVars ',session) ?form t
?disable (sevNoEnvironment ',session))
        ("Delete" ?callback (sevDeleteSelectedVars ',session)
?disable (sevNoVarSelections ',session))
        ("Find" ?callback (sevFindSelectedVars ',session) ?disable
(sevNoVarSelections ',session))
        ("Copy From Cellview" ?callback
(sevCopyCellViewVariables ',session) ?disable (sevNoEnvironment
',session))
        ("Copy To Cellview" ?callback (sevCopyVariablesToCellView
',session) ?disable (sevNoEnvironment ',session))
    ))
    ("Outputs"
    ~(
        ("Setup" ?callback (sevEditSelectedOuts
',session) ?form t ?disable (sevNoEnvironment ',session))
        ("Delete" ?callback (sevDeleteSelectedOuts ',session)
?disable (sevNoOutSelections ',session))
        ("To Be Saved" ?disable (sevNoEnvironment ',session)
?subMenu
        (
            ("Select On Schematic" ?callback (sevChangeOutsOnSchematic
',session 'save) ?disable (sevNoEnvironment ',session))
            ("Add To" ?callback (sevSetPropertyForSelectedOuts
',session 'save t) ?disable (sevNoOutSelections ',session))
            ("Remove From" ?callback
(sevSetPropertyForSelectedOuts ',session 'save nil) ?disable
(sevNoOutSelections ',session))
        ))
        ; ("To Be Marched" ?disable or((sevNoEnvironment ',session)
; equal("spectre" sevSimulator(',session)))
        ;?subMenu
        ;(
            ;("Select On Schematic" ?callback (sevChangeOutsOnSchematic
',session 'march) ?disable (sevNoEnvironment ',session))
            ;("Add To" ?callback (sevSetPropertyForSelectedOuts
',session 'march t) ?disable (sevNoOutSelections ',session))
            ;("Remove From" ?callback (sevSetPropertyForSelectedOuts
',session 'march nil) ?disable (sevNoOutSelections ',session))
        ;))
        ("To Be Plotted" ?disable (sevNoEnvironment ',session)
?subMenu
        (
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
        ("Select On Schematic" ?callback
        (sevChangeOutsOnSchematic ',session 'plot) ?disable
        (sevNoEnvironment ',session))
        ("Add To" ?callback (sevSetPropertyForSelectedOuts
        ',session 'plot t) ?disable (sevNoOutSelections ',session))
        ("Remove From" ?callback
        (sevSetPropertyForSelectedOuts ',session 'plot nil) ?disable
        (sevNoOutSelections ',session))
        ))
        ("Save Options" ?callback (sevSaveOptions ',session) ?form
        t ?disable (sevNoEnvironment ',session))
        ))
        ("Simulation"
        ~(
        ("Netlist and Run" ?callback (sevNetlistAndRun ',session)
        ?disable (or (sevNoEnvironment ',session) (asiGetStatus
        (sevEnvironment ',session))))
        ("Run" ?callback (sevRunEngine ',session)
        ?disable (or (sevNoEnvironment ',session) (asiGetStatus
        (sevEnvironment ',session))))
        ("Stop" ?callback (sevStopEngine ',session) ?disable
        (sevNoEnvironment ',session))
        ; ("Continue" ?form t ?disable (or (sevNoEnvironment
        ',session) (null (sevIsContinuable ',session))))
        ; ("Reset" ?disable (sevNoEnvironment ',session))
        ("Options" ?disable (sevNoEnvironment ',session)
        ?subMenu
        (
        ("Commonly Used" ?callback (setSimulatorOptions ',session
        "Commonly Used") ?form t)
        ("Formatting & Output Control" ?callback (setSimulatorOptions
        ',session "Formatting & Output Control") ?form t
        )
        ("Input/Output Control" ?callback (setSimulatorOptions
        ',session "Input/Output Control") ?form t
        )
        ("Model Analysis" ?callback (setSimulatorOptions ',session
        "Model Analysis") ?form t
        )
        ("Analysis" ?form t
        ?subMenu
        (
        ("AC" ?callback (setSimulatorOptions ',session
        "Analysis: AC") ?form t
        )
        ("OP and DC" ?callback (setSimulatorOptions ',session
        "Analysis: OP & DC") ?form t
        )
        ("Transient" ?callback (setSimulatorOptions ',session
        "Analysis: Pseudo-Tran & Tran") ?form t
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration
Sample HSPICE.menus Files for Versions 51 and 61

```
    )
  )
)
("All" ?callback (setSimulatorOptions ',session "All") ?form t
)
)
)
  ("Netlist" ?disable (sevNoEnvironment ',session)
  ?subMenu
  (
    ("Create" ?callback (sevNetlistFile ',session 'create))
    ("Display" ?callback (sevNetlistFile ',session 'display)
?form t)
    ("Recreate" ?callback (sevNetlistFile ',session 'recreate))
  ))
  ;("Command Type-In" ?callback (sevOpenEncap ',session)
?form t ?disable (sevNoEnvironment ',session))
  ("Output Log" ?callback (sevViewSimulatorOutput ',session)
?form t ?disable (or (sevNoEnvironment ',session) (sevNoOutputLog
',session)))
  ("Convergence Aids" ?disable (sevNoEnvironment ',session)
?subMenu t)
  ))
  ("Convergence Aids"
  ~(
    ;("Store/Restore" ?callback (sevConvergence ',session
'storeRestore) ?form t)
    ;("Transient Store/Restore" ?callback (sevConvergence
',session 'transientStoreRestore) ?form t)
    ("Node Set" ?callback (sevConvergence ',session 'nodeSet)
?form t)
    ("Initial Condition" ?callback (sevConvergence ',session
'initialCondition) ?form t)
    ;("Force Node" ?callback (sevConvergence ',session
'forceNode) ?form t)
  ))
  ("Results"
  ~(
    ; ("Select Signals To Be Plotted" ?callback
(sevChangeOutsOnSchematic ',session 'plot) ?disable
(sevNoEnvironment ',session))
    ("Plot Outputs" ?disable (or (sevNoResults ',session)
(sevNoPlottableOutputs ',session)) ?subMenu t)
    ("Direct Plot" ?disable (sevNoResults ',session) ?subMenu t)
    ("Plotting Assistant ..." ?disable (or (sevNoEnvironment
',session) (sevNoDesign ',session)) ?callback
(snps_hspa_display_form ',session))
    ("Print" ?disable (sevNoResults ',session) ?subMenu t)
    ("Annotate" ?disable (sevNoEnvironment ',session) ?subMenu t)
    ;("Circuit Conditions" ?callback (sevCircuitCond ',session)
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
?form t ?disable (or (sevNoResults ',session 'dc_op) (sevNoDesign
',session)))
    ("Save" ?callback (sevSaveResults ',session) ?form t
?disable (sevNoEnvironment ',session))
    ("Select" ?callback (snps_sevSelectResults ',session) ?form
t)
    ("Delete" ?callback (snps_sevDeleteResults ',session) ?form
t)
    ("Printing/Plotting Options" ?callback
(sevEditPlottingOptions ',session) ?form t)
    ))
    ("Plot Outputs"
    `(
        ; ("All" ?callback (sevPlotAllOutputs ',session)
?disable (or (sevNoPlottableSignals ',session) (sevNoResults
',session)))
        ("Transient" ?callback (sevPlotSignals ',session 'tran)
?disable (or (sevNoPlottableSignals ',session)
(sevNoResults ',session 'tran)))
        ("AC" ?callback (sevPlotSignals ',session 'ac) ?disable
(or (sevNoPlottableSignals ',session) (sevNoResults ',session
'ac)))
        ("Noise" ?callback (sevPlotSignals ',session 'noise)
?disable (or (sevNoPlottableSignals ',session)
(sevNoResults ',session 'noise)))
        ("DC" ?callback (sevPlotSignals ',session 'dc) ?disable
(or (sevNoPlottableSignals ',session) (sevNoResults ',session
'dc)))
        ("Expressions" ?callback (sevEvaluateAndPlotExpressions
',session) ?disable (sevNoPlottableExpressions ',session))
    ))
    ("Direct Plot"
    (sevDirectPlotMenu session
    '(
        ("Transient Signal"asiiPlotTranSignalCBtran)
        ("Transient Minus DC"asiiPlotTranSignalNoDcCB (tran dc_op))
        ("Transient Sum"asiiPlotTranAddCBtran)
        ("Transient Difference"asiiPlotTranSubtractCBtran)
        ("AC Magnitude "asiiPlotACMagCBac)
        ("AC dB10      "asiiPlotACDb10CBac)
        ("AC dB20      "asiiPlotACDb20CBac)
        ("AC Phase     "asiiPlotACPhaseCBac)
        ("AC Magnitude & Phase"asiiPlotACMagAndPhaseCBac)
        ("AC Gain & Phase"asiiPlotBodeAnalysisCBac)
        ;("Equivalent Output Noise" asiiPlotEqOutputNoiseCB noise)
        ("Equivalent Output Noise"
_HSPICEPlotEquivalentOutputNoise noise)
        ;("Equivalent Input Noise"      asiiPlotEqInputNoiseCB  noise)
        ("Equivalent Input Noise"
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

_HSPICEPlotEquivalentInputNoise noise)
;("Squared Output Noise"   asiiPlotNoiseCB      noise)
("Squared Output Noise"
_HSPICEPlotSquaredOutputNoise      noise)
;("Squared Input Noise"   asiiPlotSquInputNoiseCB noise)
("Squared Input Noise"
_HSPICEPlotSquaredInputNoise      noise)
;("Noise Figure"          asiiPlotSpectreNoiseFigureCB noise)
("DC                      "asiiPlotDCCBdc)
))
)
("Print"
~(
    ("DC Node Voltages" ?callback (sevPrintResults ',session
'dcNodeVoltages) ?disable (sevNoResults ',session 'dc_op))
    ("DC Operating Points" ?callback (sevPrintResults ',session
'dcOpPoints) ?disable (sevNoResults ',session 'dc_op))
    ;("Model Parameters" ?callback (sevPrintResults ',session
'modelParameters) ?disable (sevNoResults ',session 'dc_op))
    ("Transient Node Voltages" ?callback (sevPrintResults
',session 'tranNodeVoltages) ?disable (sevNoResults ',session
'tran))
    ("Transient Operating Points" ?callback (sevPrintResults
',session 'tranOpPoints) ?disable (sevNoResults ',session 'tran))
    ("S-Parameter" ?disable (not (xor (sevNoResults ',session 'sp )
(sevNoResults ',session 'lin )))
        ?subMenu
        (
            ("1 port" ?callback (sevPrintResults
',session 'sparam1p) )
            ("2 port" ?callback (sevPrintResults ',session 'sparam2p) )
            ("3 port" ?callback (sevPrintResults ',session
'sparam3p) )
        )
    ("Noise Parameters" ?callback (sevPrintResults ',session
'noiseParameters) ?disable (sevNoResults ',session 'noise))
    ("Noise Summary" ?callback (sevHSPICEPrintResults ',session
'noiseSummary) ?form t ?disable (sevNoResults ',session 'noise))
    ("DCMatch" ?callback (sevHSPICEPrintResults ',session
'dcmatch) ?disable (sevHSPICENoResults ',session 'dcmatch))
    ("ACMatch" ?callback (sevHSPICEPrintResults ',session
'acmatch) ?disable (sevHSPICENoResults ',session 'acmatch))
))
("Annotate"
~(
    ("DC Node Voltages" ?callback (sevAnnotateResults ',session
'dcNodeVoltages) ?disable (sevNoResults ',session 'dc_op))
    ("DC Operating Points" ?callback (sevAnnotateResults

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
' ,session 'dcOpPoints) ?disable (sevNoResults ',session 'dc_op))
; ("Model Parameters" ?callback (sevAnnotateResults ',session
'modelParameters) ?disable (sevNoResults ',session 'dc_op))
    ("Transient Node Voltages" ?callback (sevAnnotateResults
',session 'tranNodeVoltages) ?form t ?disable (sevNoResults
',session 'tran))
        ("Transient Operating Points" ?callback
(snpsAnnotateTransientOperatingPoints ',session) ?disable
(sevNoResults ',session 'tran) )
            ("Net Names" ?callback (sevAnnotateResults ',session
'netNames))
                ("Component Parameters" ?callback (sevAnnotateResults
',session 'componentParameters))
                    ("Design Defaults" ?callback (sevAnnotateResults ',session
'defaults))
                        ))
        ("Tools"
        ~(
            ("Parametric Analysis" ?callback (sevParametricTool
',session) ?form t ?disable (sevNoEnvironment ',session))
                ("Corners" ?callback (sevCornersTool ',session) ?form t
?disable (sevNoEnvironment ',session))
                    ("Monte Carlo" ?callback (sevMonteCarloTool ',session)
?form t ?disable (sevNoEnvironment ',session))
                        ("Optimization" ?callback (sevOptimizationTool ',session)
?form t ?disable (sevNoEnvironment ',session))
                            ("separator")
                                ("Calculator" ?callback (sevOpenCalculator) ?form t)
                                    ("Results Browser" ?callback (sevOpenDRLBrowser) ?form t)
                                        ("Waveform" ?callback (sevOpenPlotWindow ',session) ?form t)
                                            ("Results Display" ?callback (sevOpenPrintWindow ',session)
?form t)
                                                ("Job Monitor" ?callback (sevOpenJobMonitor) ?form t)
                                                    ))
                ("FixedMenu"
                ~(
                    ("Choose Design" ?callback (sevChooseDesign ',session)
?form t ?itemIcon ,(sevIcon 'design))
                        ("Choose Analyses" ?callback (sevEditSelectedAnas ',session)
?form t ?itemIcon ,(sevIcon 'analyses))
                            ("Edit Variables" ?callback (sevEditSelectedVars ',session)
?form t ?itemIcon ,(sevIcon 'variables))
                                ("Setup Outputs" ?callback (sevEditSelectedOuts ',session)
?form t ?itemIcon ,(sevIcon 'outputs))
                                    ("Delete" ?callback (sevDeleteSelections ',session) ?itemIcon
,(sevIcon 'delete))
                                        ("Run Simulation" ?callback (sevRunEngine ',session)
?itemIcon ,(sevIcon 'run))
                                            ("Stop Simulation" ?callback (sevStopEngine ',session)
?itemIcon ,(sevIcon 'stop))
```


Sample HSPICE.menus Files for Versions 51 and 61

```
( "Plot Outputs" ?callback (sevPlotAllOutputs ',session)
?itemIcon , (sevIcon 'plot))
))
("Help"
~(
("Using Analog Design Environment" ?callback (hiHelp 'window
(hiGetHelp (hiGetCurrentWindow)) (hiGetCurrentWindow)))
("Contents" ?callback (ddsOnLineHelp "ArtistNews"))
("separator")
("Cadence Documentation" ?callback (ddsOnLineHelp 'main))
("separator")
("Known Problems and Solutions" ?callback (ddsOnLineHelp
"kpAnAsim" ))
("What's New in ADE" ?callback (ddsOnLineHelp "pnAnAsim"))
("What's New in Analog/Mixed Signal" ?callback (sevWhatsNew)
?form t ?disable nil)
("separator")
("About Analog Design Environment" ?callback (sevAboutTool
_artADENAME()))
))
)))
(sevAddMenuItemLists
(lambda
(session name)
(case
name
("SX-ADE"
~(
("Setup Environment"          sxAASetup())
("Display Probes" ?disable (sevNoResults ',session) ?submenu
(
("Transient"   sxAADisplay(?session ',session ?type 1))
("DC"         sxAADisplay(?session ',session ?type 2))
("AC"         sxAADisplay(?session ',session ?type 3))
)
)
("Multi-Net Probe" ?disable (sevNoResults ',session) ?submenu
(
("Transient"   sxAADirect(?session ',session ?type 1))
("DC"         sxAADirect(?session ',session ?type 2))
("AC"         sxAADirect(?session ',session ?type 3))
)
)
("Instant Probe" ?disable (sevNoResults ',session) ?submenu
(
("Transient"   sxAAQuick(?session ',session ?type 1))
("DC"         sxAAQuick(?session ',session ?type 2))
("AC"         sxAAQuick(?session ',session ?type 3))
)
)
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration
Sample HSPICE.menus Files for Versions 51 and 61

```
)
("Noise Analysis" ?disable (sevNoResults ',session) ?subMenu
(
("Equivalent Output Noise"  sxAANoise(?session ',session
?type 1))
("Squared Output Noise"    sxAANoise(?session ',session
?type 2))
("Equivalent Input Noise"  sxAANoise(?session ',session
?type 3))
("Squared Input Noise"    sxAANoise(?session ',session
?type 4))
("Noise Figure"           sxAANoise(?session ',session
?type 5))
)
)
("Load Additional Files"  ?subMenu
(
("Transient"  sxAddAAFile(?session ',session ?type 1))
("DC"        sxAddAAFile(?session ',session ?type 2))
("AC"        sxAddAAFile(?session ',session ?type 3))
)
)
("Output List"  ?disable (sevNoResults ',session) ?subMenu
(
("Transient"  sxAAOutputList(?session ',session ?type 1))
("DC"        sxAAOutputList(?session ',session ?type 2))
("AC"        sxAAOutputList(?session ',session ?type 3))
)
)
("Clear Probes"  sxAAClearProbe(?session ',session))
("New SX Window" sxAAStart(?session ',session))
("Clear Highlight"  sxCleanHilight())
))
)))
```

Version 61

```
/*
 *
 *                      Copyright (C) 1995
 *                      Cadence Design Systems, ICDG - Analog Division
 *                      All Rights Reserved.
 *
 *
 *
 * $Source: /cvsrep/group/artist/src/simui/simui.menus,v $
 *
 * $Author: meghraj $
 * $Revision: 1.16 $
 * $Date: 2006/05/10 13:30:12 $
 * $State: Exp $
 *
 */
/
*****
*****
 *
 * DESCRIPTION
 * -----
 *
 * This file contains the menu definitions for the Analog Artist
 * Simulation Environment. Menus appear on the banner of the
Main Window,
 * on the banner of the Schematic Window, when in Analog Artist
mode,
 * on the banner of the "Command Type-In" window, and on the side of
 * the Main Window as a row of icons.
 *
 * Banner menus are defined in two steps. First, for each window
banner,
 * the list of pulldowns is defined using commands
 * sevSetMainWindowPulldownMenus, sevSetSchematicPulldownMenus,
and
 * sevSetTypeInWindowPulldownMenus. Second, lists of menu items
for all
 * pulldowns are defined using the command sevSetMenuItemLists.
The icon
 * menu only needs its list of items defined using
sevSetMenuItemLists
 * ("Fixedmenu").
 *
 * Menu definitions can be in one of two forms, the "simple form" or
 * the "complete form". Even though the default menu definitions
given
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
* later in this file follow the complete form, the simple form is
* adequate for most user menu customizations.
*
* Simple Custom Menu Form
* -----
*
* The list of pulldowns is given simply as the list of the names
(strings)
* of the pulldowns. For example,
*
* (sevSetMainWindowPulldownMenus '(
*   "Setup"
*   "Simulation"
*   "Results"
* ))
*
* The list of menu items is given as a list of item-lists, with each
* item-list composed of two strings, the name (text) of the
item and
* its callback. In the following example the given callbacks
may not
* be real functions. Also, the first line needs to be there as
given.
*
* (sevSetMenuItemLists (lambda (session name) (case name
*
*   ( "Setup" '(
*     ( "Design " "setDesignCB() " )
*     ( "Analyses" "setAnalysesCB() " )
*     ( "Variables" "setVariablesCB() " )
*     ( "Environment" "SetEnvironmentCB() " )
*   ))
*
*   ( "Simulation" '(
*     ( "Run      " "runSimulationCB() " )
*     ( "Stop    " "stopSimulationCB() " )
*   ))
*
*   ( "Results" '(
*     ( "Plot    " "plotResultsCB() " )
*     ( "Print   " "printResultsCB() " )
*     ( "Save    " "saveResultsCB() " )
*   ))
* ))
*
* In order to specify a slider (second-level) menu, the second
string
* of the menu item can be replaced by a list of name-callback
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

string
* pairs. For example, the last pulldown from the previous example
* could take the following form:
*
*      ( "Results" (
*      ( "Plot      "plotResultsCB() ")
*      ( "Print"
*      (
*      ( "Tran" "printTranResultsCB() ")
*      ( "AC      " "printACResultsCB() ")
*      ( "DC      " "printDCResultsCB() ")
*      ))
*      ( "Save      " "saveResultsCB() ")
*      ))
*
*
*
* Complete Custom Menu Form
* -----
*
* The complete custom menu form allows users to specify the
following:
* For each pulldown, an optional menu-items pointer name when
different
* from the pulldown's name (text). For each menu item, an optional
* disabling (greying-out) condition, an optional flag on whether
the
* item brings up a form and an optional submenu (slider) argument.
* Also, callbacks and disabling conditions can be specified either
* as strings or as (function-call) lists. For examples, look at the
* actual default menu definitions below.
*
*
* Multi-Level Customization
* -----
*
* In addition to using the command sevSetMenuItemLists which
(re)defines
* all menu item lists, users may use the command
sevAddMenuItemLists
* (same format as sevSetMenuItemLists) to add to the existing
item-list
* definitions. In this way, users may add private pulldowns on
top of site
* customizations, without having to repeat the site definitions.
*
*
* Direct-Plot Menu Customization
* -----
*

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
* The special command sevDirectPlotMenu is provided for easier
* setting of the entries of the "Direct Plot" (slider) menu. It
* must be called inside sevSetMenuItemLists or
sevAddMenuItemLists.
* The entries are specified as lists of three fields: a string
* indicating the name of the menu item, a symbol indicating the
* callback function and a symbol indicating the type of results
* required. The type can be a list of symbols, when more than one
* types are required. An optional fourth argument t indicates that
* a form will be brought up by the callback. As a note, the callback
* functions take two arguments, the environment session and the
* schematic window.
* As an example,
*
* (sevAddMenuItemLists (lambda (session name) (case name
*   ("Direct Plot"
*    (sevDirectPlotMenu session '(
*      ("Transient"transientDirectPlottran)
*      ("Transient-DC"transientMinusDCDirectPlot (tran
dc_op) )
*      ("AC Magnitude"ACMagnitudeDirectPlotac)
*      ("AC Phase"ACPhaseDirectPlotac)
*      ("FormBased"FormBasedDirectPlotxyzt)
*    )))
*   )))
*
*****
*****/
(sevSetMainWindowPulldownMenus
  ` (
    "S&ession"
    "Set&up"
    "&Analyses"
    "&Variables"
    "&Outputs"
    "&Simulation"
    "&Results"
    "&Tools"
    "SX-ADE"
    "&Help"
  ) )
(sevSetSchematicPulldownMenus
  ` (
    ("&ADE L"?items "Session_Schematic")
    ("Set&up    "?items "Setup_Schematic")
    "&Simulation"
    "&Results"
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
( "Sim-&Tools"?items "&Tools")
" SX-ADE"
))
(sevSetTypeInWindowPulldownMenus
~(
  ("&Window "?items "Session_Encap")
))
(sevSetMenuItemLists
(lambda
  (session name)
  (case
    name
    ("Session_Common"
      ~(
        ("&Save State" ?callback (sevSaveState ',session) ?form t
?disable (sevNoEnvironment ',session))
        ("&Load State" ?callback (sevLoadState ',session) ?form t
?disable (sevNoEnvironment ',session))
        ("Save S&cript" ?callback
  (snps_sevSaveOceanScript ',session) ?form t ?disable
(sevNoEnvironment ',session))
        ("&Options" ?callback (sevEditOptions ',session) ?form t)
      ))
    ("S&ession"
      (cons
        ~("Schematic &Window" ?callback (sevOpenSchematic ',session)
?form t)
        (append
          (sevMenuItems session "Session_Common")
          ~(
            ("What's New" ?callback (HSPICEDisplayNew) ?form t
?disable nil)
            ("&Reset" ?callback (sevReset ',session))
            ("&Quit" ?callback (sevQuit ',session))
          ))
        ))
    ("Session_Schematic"
      (cons
        ~("Sim&ulation Window" ?callback
(sevCreateMainWindow ',session) ?form t)
        (append
          (sevMenuItems session "Session_Common")
          ~(
            ("&Quit" ?callback (sevQuit ',session))
          ))
        ))
    ("Session_Encap"
      ~(
        ("&Close" ?callback (hiCloseWindow (hiGetCurrentWindow)))
      ))
    )
  )
)
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

    ))
    ("Setup_Common"
      `(
        ("&Simulator/Directory/Host" ?callback (sevChooseSimulator
          ',session) ?form t)
        ("&Temperature" ?callback (sevChooseTemperature ',session)
          ?form t)
        ;("Model Path" ?callback (sevMpuTool ',session) ?form t
          ?disable (sevNoEnvironment ',session))
        ("Model Libraries" ?callback (sevEditModels ',session)
          ?form t ?disable (sevNoEnvironment ',session))
        ("Simulation Files" ?callback (_HSPICESetupSimulationFiles
          ',session) ?form t ?disable (sevNoEnvironment ',session))
        ("&Environment" ?callback (sevChooseEnvironmentOptions
          ',session) ?form t ?disable (sevNoEnvironment ',session))
        ;("Stim&ulus" ?disable (sevNoEnvironment ',session)
          ?subMenu
        ;(
          ;("Edit &Analog" ?callback (sevEditStimulus ',session
            'analog) ?form t)
          ;("Edit &Digital" ?callback (sevEditStimulus ',session
            'digital) ?form t ?disable (sevNonMixedSignal ',session))
          ;))
        ;("Simulation &Files" ?disable (sevNoEnvironment ',session)
          ?subMenu
        ;(
          ;("Edit &Include File" ?callback
            (sevEditSimulationFile ',session 'include) ?form t)
          ;("Edit &Model File" ?callback (sevEditSimulationFile
            ',session 'model) ?form t)
          ;))
        ))
    ("Set&up"
      (cons
        `("Design" ?callback (sevChooseDesign ',session) ?form t)
        (sevMenuItems session "Setup_Common"))
    )
    ("Setup_Schematic"
      (append
        `(
          ("Anal&yse" ?callback (sevEditSelectedAnas ',session) ?form
            t)
          ("&Variables" ?callback (sevEditSelectedVars ',session)
            ?form t)
          ("&Outputs" ?callback (sevEditSelectedOuts ',session) ?form
            t)
          ("Select On S&chematic"
            ?subMenu
            (

```


Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration
Sample HSPICE.menus Files for Versions 51 and 61

```
        ("Outputs To Be &Plotted" ?callback
(sevChangeOutsOnSchematic ',session 'plot))
        ("Outputs To Be Sa&ved" ?callback (sevChangeOutsOnSchematic
',session 'save))
        ;("Outputs To Be Marched" ?callback
(sevChangeOutsOnSchematic ',session 'march) ?disable
or((sevNoEnvironment ',session) equal("spectre"
sevSimulator(',session)))) ; Commented for PCR 882104
    ))
    ("Save &Options" ?callback (sevSaveOptions ',session) ?form
t)
    )
    (sevMenuItems session "Setup_Common")
    ))
    ("&Analyses"
    ~(
        ("&Choose" ?callback (sevEditSelectedAnas ',session) ?form
t ?disable (sevNoEnvironment ',session))
        ("&Delete" ?callback (sevDeleteSelectedAnas ',session)
?disable (sevNoAnaSelections ',session))
        ("&Enable" ?callback (sevActivateSelectedAnas ',session
t) ?disable !(sevNonActiveSelectedAna ',session))
        ("D&isable" ?callback (sevActivateSelectedAnas ',session
nil) ?disable !(sevActiveSelectedAna ',session))
    ))
    ("&Variables"
    ~(
        ("&Edit" ?callback (sevEditSelectedVars ',session) ?form
t ?disable (sevNoEnvironment ',session))
        ("&Delete" ?callback (sevDeleteSelectedVars ',session)
?disable (sevNoVarSelections ',session))
        ("&Find" ?callback (sevFindSelectedVars ',session) ?disable
(sevNoVarSelections ',session))
        ("Copy From &Cellview" ?callback (sevCopyCellViewVariables
',session) ?disable (sevNoEnvironment ',session))
        ("Copy &To Cellview" ?callback (sevCopyVariablesToCellView
',session) ?disable (sevNoEnvironment ',session))
    ))
    ("&Outputs"
    ~(
        ("&Setup" ?callback (sevEditSelectedOuts ',session) ?form
t ?disable (sevNoEnvironment ',session))
        ("&Delete" ?callback (sevDeleteSelectedOuts ',session)
?disable (sevNoOutSelections ',session))
        ("To Be Sa&ved" ?disable (sevNoEnvironment ',session)
?subMenu
        (
            ("&Select On Schematic" ?callback (sevChangeOutsOnSchematic
',session 'save) ?disable (sevNoEnvironment ',session))
            ("&Add To" ?callback (sevSetPropertyForSelectedOuts
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

',session 'save t) ?disable (sevNoOutSelections ',session))
    ("&Remove From" ?callback (sevSetPropertyForSelectedOuts
',session 'save nil) ?disable (sevNoOutSelections ',session))
    ))
; following part is commented for PCR 882104
;("To Be Marched" ?disable or((sevNoEnvironment ',session)
    ; equal("spectre" sevSimulator(',session)))
    ;?subMenu
    ;(
    ;("Select On Schematic" ?callback (sevChangeOutsOnSchematic
',session 'march) ?disable (sevNoEnvironment ',session))
    ;("Add To" ?callback (sevSetPropertyForSelectedOuts
',session 'march t) ?disable (sevNoOutSelections ',session))
    ;("Remove From" ?callback (sevSetPropertyForSelectedOuts
',session 'march nil) ?disable (sevNoOutSelections ',session))
    ;)) Commented for PCR 882104
    ("To Be &Plotted" ?disable (sevNoEnvironment ',session)
    ?subMenu
    (
    ("&Select On Schematic" ?callback (sevChangeOutsOnSchematic
',session 'plot) ?disable (sevNoEnvironment ',session))
    ("&Add To" ?callback (sevSetPropertyForSelectedOuts
',session 'plot t) ?disable (sevNoOutSelections ',session))
    ("&Remove From" ?callback (sevSetPropertyForSelectedOuts
',session 'plot nil) ?disable (sevNoOutSelections ',session))
    ))
    ("Save &Options" ?callback (sevSaveOptions ',session) ?form
t ?disable (sevNoEnvironment ',session))
    ))
    ("&Simulation"
    ~(
    ("Netlist &and Run" ?callback (sevNetlistAndRun ',session)
?disable (or (sevNoEnvironment ',session) (asiGetStatus
(sevEnvironment ',session)))) ;;PCR895431
    ("&Run" ?callback (sevRunEngine ',session) ?disable (or
(sevNoEnvironment ',session) (asiGetStatus (sevEnvironment
',session))))
    ("&Stop" ?callback (sevStopEngine ',session) ?disable
(sevNoEnvironment ',session))
    ; ("Continue" ?form t ?disable (or (sevNoEnvironment
',session) (null (sevIsContinuable ',session))))
    ; ("Reset" ?disable (sevNoEnvironment ',session))
    ("&Options" ?disable (sevNoEnvironment ',session)
    ?subMenu
    (
    ("Commonly Used" ?callback (setSimulatorOptions ',session
"Commonly Used") ?form t)
    ("Formatting && Output Control" ?callback (setSimulatorOptions
',session "Formatting & Output Control") ?form t)
    ("Input/Output Control" ?callback (setSimulatorOptions

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

',session "Input/Output Control") ?form t)
    ("Model Analysis" ?callback (setSimulatorOptions ',session
"Model Analysis") ?form t)
        ("Analysis" ?form t
            ?subMenu
                (
                    ("AC" ?callback (setSimulatorOptions ',session "Analysis:
AC") ?form t)
                        ("OP and DC" ?callback (setSimulatorOptions ',session
"Analysis: OP & DC") ?form t)
                            ("Transient" ?callback (setSimulatorOptions ',session
"Analysis: Pseudo-Tran & Tran")
                                ?form t
                                    )
                                )
                            )
                    )
                ("All" ?callback (setSimulatorOptions ',session "All") ?form t)
                    ;("&Analog" ?callback (sevSetEngineOptions ',session
'analog) ?form t)
                    ;("&Digital" ?callback (sevSetEngineOptions ',session
'digital) ?form t ?disable (sevNonMixedSignal ',session))
                    ;("&Mixed Signal" ?callback (sevSetEngineOptions ',session
'mixed) ?form t ?disable (sevNonMixedSignal ',session))
                    ))
                ("&Netlist" ?disable (sevNoEnvironment ',session)
                ?subMenu
                    (;;PCR895431
                    ("&Create" ?callback (sevNetlistFile ',session 'create))
                    ("&Display" ?callback (sevNetlistFile ',session 'display)
?form t)
                    ("&Recreate" ?callback (sevNetlistFile ',session 'recreate))
                    ))
                ;("Command &Type-In" ?callback (sevOpenEncap ',session)
?form t ?disable (sevNoEnvironment ',session))
                ("Output &Log" ?callback (sevViewSimulatorOutput ',session)
?form t ?disable (or (sevNoEnvironment ',session) (sevNoOutputLog
',session)))
                ("&Convergence Aids" ?disable (sevNoEnvironment ',session)
?subMenu t)
                ))
                ("&Convergence Aids"
                ^ (
                    ;("&Store/Restore" ?callback (sevConvergence ',session
'storeRestore) ?form t)
                    ;("&Transient Store/Restore" ?callback (sevConvergence
',session 'transientStoreRestore) ?form t)
                    ("&Node Set" ?callback (sevConvergence ',session 'nodeSet)
?form t)
                    ("&Initial Condition" ?callback (sevConvergence ',session
'initialCondition) ?form t)

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

        ;("&Force Node" ?callback (sevConvergence ',session
'forceNode) ?form t)
    ))
    ("&Results"
    \
        ; ("Select Signals To Be Plotted" ?callback
(sevChangeOutsOnSchematic ',session 'plot) ?disable
(sevNoEnvironment ',session))
        ("Plot &Outputs" ?disable (or (sevNoResults ',session)
(sevNoPlottableOutputs ',session)) ?subMenu t)
        ("Direct &Plot" ?disable (sevNoResults ',session) ?subMenu t)
        ("Plotting Assistant ..." ?disable (or (sevNoEnvironment
',session) (sevNoDesign ',session)) ?callback
(snps_hspa_display_form ',session))
        ("Pr&int" ?disable (sevNoResults ',session) ?subMenu t)
        ("&Annotate" ?disable (sevNoEnvironment ',session) ?subMenu
t)
        ; ("Circuit &Conditions" ?callback (sevCircuitCond ',session)
?form t ?disable (or (sevNoResults ',session 'dc_op) (sevNoDesign
',session)))
        ("&Save" ?callback (sevSaveResults ',session) ?form t
?disable (sevNoEnvironment ',session))
        ("S&elect" ?callback (snps_sevSelectResults ',session)
?form t)
        ("&Delete" ?callback (snps_sevDeleteResults ',session)
?form t)
        ("Prin&ting/Plotting Options" ?callback
(sevEditPlottingOptions ',session) ?form t)
    ))
    ("Plot &Outputs"
    \
        ; ("All" ?callback (sevPlotAllOutputs ',session) ?disable
(or (sevNoPlottableSignals ',session) (sevNoResults ',session)))
        ("&Transient" ?callback (sevPlotSignals ',session 'tran)
?disable (or (sevNoPlottableSignals ',session) (sevNoResults
',session 'tran)))
        ("&AC" ?callback (sevPlotSignals ',session 'ac) ?disable
(or (sevNoPlottableSignals ',session) (sevNoResults ',session
'ac)))
        ("&Noise" ?callback (sevPlotSignals ',session 'noise)
?disable (or (sevNoPlottableSignals ',session) (sevNoResults
',session 'noise)))
        ("&DC" ?callback (sevPlotSignals ',session 'dc) ?disable
(or (sevNoPlottableSignals ',session) (sevNoResults ',session
'dc)))
        ("&Expressions" ?callback (sevEvaluateAndPlotExpressions
',session) ?disable (sevNoPlottableExpressions ',session))
    ))
    ("Direct &Plot"
    (sevDirectPlotMenu session

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

    ' (
      ("&Transient Signal"asiiPlotTranSignalCBtran)
      ("Transient Min&us DC"asiiPlotTranSignalNoDcCB (tran
dc_op))
      ("Transient &Sum"asiiPlotTranAddCBtran)
      ("Transient Diff&erence"asiiPlotTranSubtractCBtran)
      ("AC &Magnitude"asiiPlotACMagCBac)
      ("AC dB&10      "asiiPlotACDb10CBac)
      ("AC dB&20      "asiiPlotACDb20CBac)
      ("AC &Phase     "asiiPlotACPhaseCBac)
      ("&AC Magnitude & Phase"asiiPlotACMagAndPhaseCBac)
      ("AC &Gain & Phase"asiiPlotBodeAnalysisCBac)
      ;("Equivalent &Output Noise"asiiPlotEqOutputNoiseCBnoise)
      ("Equivalent &Output Noise
"_HSPICEPlotEquivalentOutputNoisenoise)
      ;("Equivalent &Input Noise"asiiPlotEqInputNoiseCBnoise)
      ("Equivalent &Input Noise"
_HSPICEPlotEquivalentInputNoisenoise)
      ;("S&quared Output Noise"asiiPlotNoiseCBnoise)
      ("S&quared Output Noise"_HSPICEPlotSquaredOutputNoise
noise)
      ;("Squared Input &Noise"asiiPlotSquInputNoiseCBnoise)
      ("Squared Input &Noise"_HSPICEPlotSquaredInputNoisenoise)
      ;("Noise &Figure"asiiPlotNoiseFigureCB(ac noise))
      ("&DC              "asiiPlotDCCBdc)
    )
  )
  ("Pr&int"
  ~ (
    ("DC Node &Voltages" ?callback (sevPrintResults ',session
'dcNodeVoltages) ?disable (sevNoResults ',session 'dc_op))
    ("DC &Operating Points" ?callback (sevPrintResults ',session
'dcOpPoints) ?disable (sevNoResults ',session 'dc_op))
    ;("&Model Parameters" ?callback (sevPrintResults ',session
'modelParameters) ?disable (sevNoResults ',session 'dc_op))
    ("&Transient Node Voltages" ?callback (sevPrintResults
',session 'tranNodeVoltages) ?disable (sevNoResults ',session
'tran))
    ("Transient Operating &Points" ?callback (sevPrintResults
',session 'tranOpPoints) ?disable (sevNoResults ',session
'tran))
    ("&S-Parameter" ?disable (not (xor (sevNoResults ',session 'sp
) (sevNoResults ',session 'lin )))
      ?subMenu
      (
        ("&1 port" ?callback (sevPrintResults ',session
'sparam1p) )
        ("&2 port" ?callback (sevPrintResults ',session
'sparam2p) )
      )
    )
  )

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```

        ("%3 port" ?callback (sevPrintResults ',session
'sparam3p) )
    )
)
\ ("%Noise Parameters" ?callback (sevPrintResults ',session
'noiseParameters) ?disable (sevNoResults ',session 'noise))
    ("Noise S&ummary" ?callback (sevHSPICEPrintResults ',session
'noiseSummary) ?form t ?disable (sevNoResults ',session 'noise))
        ("%DCMatch" ?callback (sevHSPICEPrintResults ',session
'dcmatch) ?disable (sevHSPICENoResults ',session 'dcmatch))
        ("%ACMatch" ?callback (sevHSPICEPrintResults ',session
'acmatch) ?disable (sevHSPICENoResults ',session 'acmatch))
    ))
    ("%Annotate"
    \ (
        ("DC Node &Voltages" ?callback (sevAnnotateResults ',session
'dcNodeVoltages) ?disable (sevNoResults ',session 'dc_op))
        ("DC &Operating Points" ?callback (sevAnnotateResults
',session 'dcOpPoints) ?disable (sevNoResults ',session 'dc_op))
        ; ("%Model Parameters" ?callback (sevAnnotateResults ',session
'modelParameters) ?disable (sevNoResults ',session 'dc_op))
        ("%Transient Node Voltages" ?callback (sevAnnotateResults
',session 'tranNodeVoltages) ?form t ?disable (sevNoResults
',session 'tran))
        ("Transient Operating &Points" ?callback
(snpsAnnotateTransientOperatingPoints ',session) ?disable
(sevNoResults ',session 'tran) )
        ("Net &Names" ?callback (sevAnnotateResults ',session
'netNames))
        ("%Component Parameters" ?callback (sevAnnotateResults
',session 'componentParameters))
        ("Design &Defaults" ?callback (sevAnnotateResults ',session
'defaults))
    ))
    ("%Tools"
    nconc(
        \(("&Parametric Analysis" ?callback (sevParametricTool
',session) ?form t ?disable (sevNoEnvironment ',session))
        (if _axlIsAdeEDFMallowed() then ; 811559
        \(("Corners" ?callback (sevCornersTool ',session) ?form
t ?disable (sevNoEnvironment ',session))
        ("Monte Carlo" ?callback (sevMonteCarloTool ',session)
?form t ?disable (sevNoEnvironment ',session))
        ("Optimization" ?callback (sevOptimizationTool ',session)
?form t ?disable (sevNoEnvironment ',session))
        else
            nil
        )
    ' ("separator")
    ("%Calculator" ?callback (sevOpenCalculator) ?form t)

```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
        ("Results &Browser" ?callback (sevOpenDRLBrowser) ?form t)
        ("&Waveform" ?callback (sevOpenPlotWindow ',session) ?form t)
        ("Results &Display" ?callback (sevOpenPrintWindow ',session)
?form t)
        ("&Job Monitor" ?callback (sevOpenJobMonitor) ?form t)
        )
    ) ; end of nconc
) ; end of Tools
("FixedMenu"
~(
    ("Choose Design" ?callback (sevChooseDesign ',session)
?form t ?itemIcon ,(sevIcon 'design))
    ("Choose Analyses" ?callback (sevEditSelectedAnas ',session)
?form t ?itemIcon ,(sevIcon 'analyses))
    ("Edit Variables" ?callback (sevEditSelectedVars ',session)
?form t ?itemIcon ,(sevIcon 'variables))
    ("Setup Outputs" ?callback (sevEditSelectedOuts ',session)
?form t ?itemIcon ,(sevIcon 'outputs))
    ("Delete" ?callback (sevDeleteSelections ',session) ?itemIcon
,(sevIcon 'delete))
    ("Run Simulation" ?callback (sevRunEngine ',session)
?itemIcon ,(sevIcon 'run))
    ("Stop Simulation" ?callback (sevStopEngine ',session)
?itemIcon ,(sevIcon 'stop))
    ("Plot Outputs" ?callback (sevPlotAllOutputs ',session)
?itemIcon ,(sevIcon 'plot))
    ))
    ("&Help"
~(
        ("&Using Analog Design Environment" ?callback (hiHelp
'window (hiGetHelp (hiGetCurrentWindow)) (hiGetCurrentWindow)))
        ("Co&ntents" ?callback (ddsOnLineHelp "ArtistNews"))
        ("separator")
        ("Cadence &Documentation" ?callback (ddsOnLineHelp 'main))
        ("separator")
        ("&Known Problems and Solutions" ?callback (ddsOnLineHelp
"kpAnAsim" ))
        ("&What's New in ADE" ?callback (ddsOnLineHelp "pnAnAsim"))
        ("What's New in Analog/&Mixed Signal" ?callback (sevWhatsNew)
?form t ?disable nil)
        ("separator")
        ("&About Analog Design Environment" ?callback (sevAboutTool
_artADEName()))
    ))
))
(sevAddMenuItemLists
(lambda
(session name)
(case
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
name
("SX-ADE"
~(
  ("Setup Environment"      sxAASetup())
  ("Display Probes" ?disable (sevNoResults ',session) ?subMenu
  (
    ("Transient"  sxAADisplay(?session ',session ?type 1))
    ("DC"         sxAADisplay(?session ',session ?type 2))
    ("AC"         sxAADisplay(?session ',session ?type 3))
  )
)
  ("Multi-Net Probe" ?disable (sevNoResults ',session) ?subMenu
  (
    ("Transient"  sxAADirect(?session ',session ?type 1))
    ("DC"         sxAADirect(?session ',session ?type 2))
    ("AC"         sxAADirect(?session ',session ?type 3))
  )
)
  ("Instant Probe" ?disable (sevNoResults ',session) ?subMenu
  (
    ("Transient"  sxAAQuick(?session ',session ?type 1))
    ("DC"         sxAAQuick(?session ',session ?type 2))
    ("AC"         sxAAQuick(?session ',session ?type 3))
  )
)
  ("Noise Analysis" ?disable (sevNoResults ',session) ?subMenu
  (
    ("Equivalent Output Noise"  sxAANoise(?session ',session
?type 1))
    ("Squared Output Noise"     sxAANoise(?session ',session
?type 2))
    ("Equivalent Input Noise"   sxAANoise(?session ',session
?type 3))
    ("Squared Input Noise"      sxAANoise(?session ',session
?type 4))
    ("Noise Figure"            sxAANoise(?session ',session
?type 5))
  )
)
  ("Load Additional Files" ?subMenu
  (
    ("Transient"  sxAddAAFile(?session ',session ?type 1))
    ("DC"         sxAddAAFile(?session ',session ?type 2))
    ("AC"         sxAddAAFile(?session ',session ?type 3))
  )
)
  ("Output List" ?disable (sevNoResults ',session) ?subMenu
  (
    ("Transient"  sxAAOutputList(?session ',session ?type 1))
```


Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

```
( "DC"          sxAAOutputList(?session ',session ?type 2))
( "AC"          sxAAOutputList(?session ',session ?type 3))
)
)
("Clear Probes"  sxAAClearProbe(?session ',session))
("New SX Window" sxAASStart(?session ',session))
("Clear Highlight"  sxCleanHilight())
))
)))
```

Appendix A: Adding a CustomExplorer™ Menu to the HSPICE Integration

Sample HSPICE.menus Files for Versions 51 and 61

OCEAN API Functions for HSPICE Monte Carlo Analysis

Describes HSPICE-defined OCEAN functions provided to run Monte Carlo analyses.

Using the HSPICE-Provided OCEAN API for HSPICE Monte Carlo Runs

The Save OCEAN script option under the File menu of the Monte Carlo Analysis window saves all operations performed in the Monte Carlo Analysis window as an OCEAN script file to be executed under the Non-graphic mode. See [Chapter 8, Monte Carlo in the HSPICE Integration](#) for a description of the Monte Carlo Analysis form and its functionality. You can use these OCEAN API functions or the Virtuoso general OCEAN functions to write the OCEAN script directly to simulate in batch mode.

When you combine these provided HSPICE Monte Carlo-OCEAN functions with the native Cadence OCEAN functions, you can run a Monte Carlo analysis in batch mode without the GUI.

The following are the HSPICE-defined OCEAN functions provided to run Monte Carlo analyses.

1. **snpsMonte** function:

Syntax:

```
snpsMonte(s_analysis ?numRun x_numRun [startRun x_startRun])
=>t/nil
snpsMonte(s_analysis ?listRuns t_listRuns)
=>t/nil
```

Appendix B: OCEAN API Functions for HSPICE Monte Carlo Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Monte Carlo Runs

Description: Sets up run information for one Monte Carlo analysis. You can set up Monte Carlo information for DC, AC, or TRAN analysis.

Note: If startRun and numRun are needed, they must be input together; otherwise, you can only input listRuns.

Argument	Description
s_analysis	Type of analysis; valid values are: 'dc, 'ac, 'tran
?startRun	Number of Monte Carlo first run
?numRun	Monte Carlo number of runs
?listRuns	String of Monte Carlo run settings, such as: "1:5 10:15"

Value Returned:

t Monte Carlo setting succeeded.

nil One or more errors; note details in the printed report.

Example:

```
snpsMonte('tran ?startRun 1 ?numRun 30)
=>t
snpsMonte('tran ?listRuns "1:5 10:15")
=>t
```

2. snpsMonteAddParam function:

Syntax:

```
snpsMonteAddParam(S_paramName s_distrFunc t_nomiVal
+ t_absOrRelVal [t_sigma] [t_multipler])
=>t/nil
```

Description: Adds a parameter with distribution function for Monte Carlo analysis.

Note: 1. Value parameters should be input as a string with double quote.

Appendix B: OCEAN API Functions for HSPICE Monte Carlo Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Monte Carlo Runs

2. If the paramName is the same as an existing predefined parameter name, the parameter is redefined.

Argument	Description
S_paramName	Symbol or string of the parameter name.
s_distrFunc	Type of distribution function; valid functions are: 'AGAUSS, 'GAUSS, 'AUNIF, 'UNIF, and 'LIMIT
t_nomiVal	Nominal value string of the parameter.
t_absOrRelVal	Absolute or relative value string of the parameter.
t_multiplier	Multiplier value string of the parameter.

Value Returned:

t The parameter setting succeeded with distribution function.
nil One or more errors; note details the printed report.

Example:

Adding the parameter “deltaW” to the different distribution functions for Monte Carlo simulation runs:

```
snpsMonteAddParam('deltaW 'UNIF "0.0" "0.1" "1")
snpsMonteAddParam('deltaW 'AUNIF "0.1" "0.01" "2")
snpsMonteAddParam('deltaW 'GAUSE "0.0" "0.01" "0.1" "2")
snpsMonteAddParam('deltaW 'AGAUSE "0.0" "0.01" "0.1")
snpsMonteAddParam('deltaW 'LIMIT "1.0" "5.0")
```

3. **snpsMonteRun** function:

Syntax:

```
snpsMonteRun(@rest arglist)
```

Description: Wrap function of the OCEAN function “run()”, which does a check of user-defined Monte Carlo settings.

Appendix B: OCEAN API Functions for HSPICE Monte Carlo Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Monte Carlo Runs

Note: Before using the function to run Monte Carlo simulation, the `snpsMonte()` function must be used for setting the distribution parameters for simulation.

Argument	Description
Same as <code>run()</code>	Refer to the OCEAN documentation for the <code>run()</code> function to use this function.

Value Returned
Same as `run()`

Examples:

Only run a tran analysis Monte Carlo simulation.

```
snpsMonteRun(tran)
```

Run all analyses settings in a Monte Carlo simulation.

```
snpsMonteRun()
```

4. **snpsMonteGetMeas** function:

Syntax

```
snpsMonteGetMeas([analysisList])  
=>l_measResults/nil
```

Description: Load statistical simulation results and measurement results after Monte Carlo simulation which is used for the `snpsMontePlotMeas` to do plotting.

- Note:**
1. The `snpsMonteRun()` must be called before using this function.
 2. The HSPICE Measure file must be included by using the “definitionFile” OCEAN function before calling this function.
 3. If you want to plot measurement results, this function must be called before the plot function.

Argument	Description
analysisList	Use this parameter to specify which kind of analysis results to be loaded.

Appendix B: OCEAN API Functions for HSPICE Monte Carlo Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Monte Carlo Runs

Value Returned:

`L_measResults` The variables statistical value list.
`nil` One or more errors; note details the printed report.

Examples:

Loads TRAN, DC analysis Monte Carlo simulation measurement results and returns variables and statistical value list.

```
snpsMonteGetMeas('tran' 'dc')
```

Loads all Monte Carlo simulation measurement results and the statistical values list.

```
snpsMonteGetMeas()
```

5. **snpsMontePlotMeas** function:

Syntax:

```
snpsMontePlotMeas(s_analysis ?yAxis  
+ t_yAxis ?xAxis t_xAxis ?plotAs s_plotAs  
[?doPlot g_doPlot] [?plotMode s_plotMode])  
=>o_waveform/nil
```

Note: Used to create or plot scatter or curve waveform.

```
snpsMontePlotMeas(s_analysis ?xAxis  
+ t_xAxis ?bins x_bins ?plotAs s_plotAs  
+ [?doPlot g_doPlot] [?plotMode t_plotMode])  
=>o_waveform/nil
```

Note: Used to create or plot histogram, that is: ?plotAs 'histo

Description: Create or plot “snpsMonteGetMeas” loaded measurement results. This function can generate curve, scatter, and histogram plots.

Note: The snpsMonteGetMeas function must be called to load measurement results before this function is called.

Argument	Description
s_analysis	Symbol name of analysis, valid values are: 'dc' 'ac' 'tran'.
?yAxis	Input variation parameter or measured variable name to plot on the Y Axis

Appendix B: OCEAN API Functions for HSPICE Monte Carlo Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Monte Carlo Runs

Argument	Description
?xAxis	Input variation parameter or measured variable name as X Axis

Note: 1.To generate a histogram, this parameter must be input.
2. yAxis and xAxis must the variable in the measure results file, if it is incorrectly specified, the available variable name list is printed.

Plot Argument	Description
?bins	Bin number of the histogram; when plotting a histogram, this parameter must be input.
?plotAs	Waveform type; valid values are: <ul style="list-style-type: none">▪ 'scatter▪ 'histo▪ 'curve
?doPlot	Specify whether plotting the measure results; default value is t
?plotMode	Specify the window mode of plotting window; valid values are: <ul style="list-style-type: none">▪ Replace: Replace existing window's waveform▪ NewSubwin: New subwindow in the existing waveform window and append new created waveform▪ Append (default): Add a new waveform to the existing waveform window▪ NewWin: Create a new waveform window in the existing waveform window and append new created waveform

Value Returned:

```
o_waveform  New created waveform.  
nil         One or more errors; note details the printed report.
```

Examples:

This example uses the “trise” variable to create and plot a histogram waveform.

```
wave=snpsMontePlotMeas('tran ?xAxis "trise" ?plotAs 'histo  
+ ?bins 5 ?doPlot nil)  
plot wave
```


Appendix B: OCEAN API Functions for HSPICE Monte Carlo Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Monte Carlo Runs

This example uses “trise” and “m1@deltaw” variables to create a scatter plot and plot it in a newly created waveform window.

```
snpsMontePlotMeas('tran ?yAxis "trise" ?xAxis "m1@deltaw"  
+ ?plotAs 'scatter ?plotMode "newWin")
```

Appendix B: OCEAN API Functions for HSPICE Monte Carlo Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Monte Carlo Runs

OCEAN API Functions for HSPICE RF Analysis

Describes HSPICE-defined OCEAN functions provided to run HSPICE RF analyses.

Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis

The Save OCEAN script option under the File menu of the HSPICE RF Analysis form saves all operations performed in the HSPICE RF Analysis window as an OCEAN script file to be executed under the Non-graphic mode. See [Chapter 10, HSPICE RF Analysis](#) for a description of the HSPICE RF Analysis form and its functionality. You can use these OCEAN API functions or the Virtuoso general OCEAN functions to write the OCEAN script directly to simulate in batch mode.

When you combine these provided HSPICE Monte Carlo-OCEAN functions with the native Cadence OCEAN functions, you can run a RF analysis in batch mode without the GUI.

The following are the HSPICE-defined OCEAN functions provided to run RF analyses.

1. snpsAnalysis

snpsAnalysis function:

Syntax:

```
snpsAnalysis(s_analysis ?parameter t_value [?parameter t_value ...
] )
=>t/nil
```

Appendix C: OCEAN API Functions for HSPICE RF Analysis

Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis

Description: Sets up run information for an HSPICE RF analysis. You can set up HSPICE RF information for “HB”, “HBAC”, “HBOSC”, “HBPHASE”, “SN”, “SNAC”, “SNOSC”, “SNPHASE” and “PHASENOISE” analyses.

Note: The parameter and t_value should be input together. Once the “parameter” is given, its “t_value” should also be provided; the value parameters should be input as strings with double quotes.

Arguments for HB	Description
"fundTones"	Fundamental frequencies (required).
"harmNums"	Number of harmonics for each tone (required).
"interMaxOrder"	Intermodulation max order.
"ssTone"	Small signal tone number.
"sweepStart"	Start of the sweep.
"sweepType"	Type of the sweep, valid values are “LIN”, “DEC”, “OCT” and “POI”
"sweepStop"	Stop of the sweep.
"sweepStep"	Step of the sweep.
"sweepVariable"	Sweep variable name. If it is temperature sweep, its value is "Temperature".
"sweepListPoints"	Sweep point list, applied only when sweep type is "POI".

Arguments for HBAC and SNAC	Description
"sweepStart"	Start of the sweep.
"sweepType"	Type of the sweep, valid values are “LIN”, “DEC”, “OCT”, and “POI”.
"sweepStop"	Stop of the sweep.
"sweepStep"	Step of the sweep.
"sweepListPoints"	Sweep point list, applied only when sweep type is "POI".

Appendix C: OCEAN API Functions for HSPICE RF Analysis
Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis

Arguments for HBOSC	Description
"fundTones"	Approximate oscillator fundamental frequencies (required).
"harmNums"	Number of harmonics for each tone (required).
"interMaxOrder"	Intermodulation max order.
"probeNodeN1"	The positive node (required).
"probeNodeN2"	The negative node (required).
"probeNodeVP"	Set probeNode parameter with a separate current source (required).
"ssTone"	Small signal tone number.
"stpsNum"	Number of search frequency points.
"stpsMin"	Min search frequency.
"stpsMax"	Max search frequency.
"stability"	Valid values are: "0", "1", "2", "-2", "-1". Refer to the <i>HSPICE User Guide: RF Analysis</i> for their meanings.
"subharms"	Subharmonics in the analysis spectrum.
"sweepStart"	Start of the sweep.
"sweepType"	Type of the sweep, valid values are "LIN", "DEC", "OCT", and "POI".
"sweepStop"	Stop of the sweep.
"sweepStep"	Step of the sweep.
"sweepVariable"	Sweep variable name. If it is temperature sweep, its value is "Temperature".
"sweepListPoints"	Sweep point list, applied only when sweep type is "POI".

Appendix C: OCEAN API Functions for HSPICE RF Analysis
Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis

Arguments for HBNOISE and SNNOISE	Description
"fundTones"	Fundamental frequencies (required).
"harmNums"	Number of harmonics for each tone (required).
"interMaxOrder"	Intermodulation max order.
"ssTone"	Small signal tone number.
"sweepStart"	Start of the sweep.
"sweepType"	Type of the sweep, valid values are "LIN", "DEC", "OCT" and "POI"
"sweepStop"	Stop of the sweep.
"sweepStep"	Step of the sweep.
"sweepVariable"	Sweep variable name. If it is temperature sweep, its value is "Temperature".
"sweepListPoints"	Sweep point list, applied only when sweep type is "POI".

Arguments for SN	Description
"fundTones"	Fundamental frequencies (required).
"harmNums"	Number of harmonics for each tone (required).
"interMaxOrder"	Intermodulation max order.
"ssTone"	Small signal tone number.
"sweepStart"	Start of the sweep.
"sweepType"	Type of the sweep, valid values are "LIN", "DEC", "OCT" and "POI"
"sweepStop"	Stop of the sweep.
"sweepStep"	Step of the sweep.

Appendix C: OCEAN API Functions for HSPICE RF Analysis
Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis

Arguments for SN	Description
"sweepVariable"	Sweep variable name. If it is temperature sweep, its value is "Temperature".
"sweepListPoints"	Sweep point list, applied only when sweep type is "POI".

Arguments for SNOSC	Description
"fundTones"	Fundamental frequencies (required).
"harmNums"	Number of harmonics for each tone (required).
"interMaxOrder"	Intermodulation max order.
"ssTone"	Small signal tone number.
"sweepStart"	Start of the sweep.
"sweepType"	Type of the sweep, valid values are "LIN", "DEC", "OCT" and "POI"
"sweepStop"	Stop of the sweep.
"sweepStep"	Step of the sweep.
"sweepVariable"	Sweep variable name. If it is temperature sweep, its value is "Temperature".
"sweepListPoints"	Sweep point list, applied only when sweep type is "POI".

Arguments for PHASENOISE	Description
"type"	Valid values are "net", "dnet" and "instance", where the "dnet" means the "differential net"
"sweepStart"	Start of the sweep.
"sweepType"	Type of the sweep, valid values are "LIN", "DEC", "OCT" and "POI"
"sweepStop"	Stop of the sweep.

Appendix C: OCEAN API Functions for HSPICE RF Analysis

Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis

Arguments for PHASENOISE	Description
"sweepStep"	Step of the sweep.
"sweepVariable"	Sweep variable name. If it is temperature sweep, its value is "Temperature".
"sweepListPoints"	Sweep point list, applied only when sweep type is "POI".
"listFreq"	Frequencies list at which the element noise value is printed. Valid types are "none", "all" and "specified". For "specified" type, user can set up the frequencies with "," between them.
"listCount"	The number of noise elements to be printed into *.lis file
"listFloor"	Min noise value that can be printed.
"listSource"	Valid values are "ON" and "OFF". "ON" means print the contribution from each noise source and "OFF" disables this operation
"probeNodeN1"	The positive node, required.
"probeNodeN2"	The negative node, required if "type" is "dnet"
"inSrc"	Input source
"spurious"	Selects phase noise spur analysis. Valid values are "0" and "1". "0" = No spurious analysis (default). "1" = Activates additional SPUR source analysis
"method"	Valid values are "0", "1", and "2". <ul style="list-style-type: none"> ▪ METHOD=0 (default) selects the Nonlinear Perturbation (NLP) ▪ METHOD=1 selects the Periodic AC (PAC) ▪ METHOD=2 selects the Broadband Phase Noise (BPN)
"carrierIndex"	The harmonic index of the carrier at which HSPICERF computes the phase noise
t_value	The value of the parameter
Value Returned:	
t	RF setting succeeded.
nil	One or more errors; note details in the printed report.

HB Examples

Example 1(for HB): Sets up the "HB" analysis with fundamental tone of 2G and harmonic of 13.

```
snpsAnalysis('HB ?fundTones "2G" ?harmNums "13" )
```

Example 2: Sets up the "HB" analysis with fundamental tone of 1G and harmonic of 3. The sweep parameter is "ddd". The swept value is 1 and 2.

```
snpsAnalysis('HB ?fundTones "1G" ?harmNums "3" ?  
sweepVariable "ddd" ?sweepType "POI" ?sweepListPoints "1, 2" )
```

Example 3: Sets up the "HB" analysis with fundamental tone of 1G, 1.1G, 1.2G, 1.3G, 1.4G, and their harmonics of 5, 6, 7, 8, 9, respectively.

```
" snpsAnalysis('hb ? fundTones "1G,1.1G,1.2G,1.3G,1.4G" ?  
harmNums "5,6,7,8,9")
```

Example 4: Sets up the "HB" analysis with fundamental tone of 1G, 1.1G, 1.2G, 1.3G, 1.4G, and their harmonics of 5, 6, 7, 8, 9, respectively. The maximum inter-modulation order is 12.

```
snpsAnalysis('hb ?fundTones "1G,1.1G,1.2G,1.3G,1.4G" ?  
harmNums "5,6,7,8,9" ?intermMaxOrder "12")
```

Example 5: Sets up the "HB" analysis with fundamental tone of 1G, 1.1G, 1.2G, 1.3G, 1.4G, and their harmonics of 5, 6, 7, 8, 9, respectively. The maximum inter-modulation order is 12. The small signal tone is 5.

```
snpsAnalysis('hb ?fundTones  
"1G,1.1G,1.2G,1.3G,1.4G" ?harmNums "5,6,7,8,9" ?  
intermMaxOrder "12" ?ssTone "5" )
```

Example 6: Sets up the "HB" analysis with fundamental tone of 1G, 1.1G, 1.2G, 1.3G, 1.4G, and their harmonic of 5, 6, 7, 8, 9, respectively. The maximum inter-modulation order is 12. The small signal tone is 5. The temperature sweep is from 200k to 350k with the step of 10 and the sweep type is "LIN".

```
snpsAnalysis('hb ?fundTones "1G,1.1G,1.2G,1.3G,1.4G" ?  
harmNums "5,6,7,8,9" ?intermMaxOrder "12" ?  
ssTone "5" ?sweepVariable "Temperature" ?  
sweepType "LIN" ?sweepStart "200k" ?  
sweepStop "350k" ?sweepStep "10")
```

Example 7: This setup will result in an error message generated noting that "fund" is an unknown parameter.

```
snpsAnalysis('HB ?fund "2G" ?harmNums "13" )
```

Value Returned
Same as run()

HBNOISE Examples

Example 1: Sets up the "HBNOISE" analysis with reference net of "/OUT". The input source is "/V12". The sweep is from 1K to 1M with 10 steps of "LIN" type.

```
snpsAnalysis('HBNOISE ? nodeType "net" ?  
probeNodeN1 "/OUT" ?inSrc "/V1" ?sweepType "LIN" ?  
sweepStart "1k" ?sweepStop "1M" ?sweepStep "10")
```

Example 2: Sets up the "HBNOISE" analysis with reference net of "/OUT1". The input source is "/V12". The sweep is from 1K to 1M with 10 steps of "LIN" type. The output frequency is small signal harmonic 3, 4, 5, plus the fundamental frequency.

```
snpsAnalysis('HBNOISE ? nodeType "net" ?probeNodeN1 "/OUT1"  
?inSrc "/V1" ?sweepType "DEC" ?sweepStart "1k" ?sweepStop "1M"  
?sweepStep "10" ?outputFreqIdx "3 4 5 +1")
```

Example 3: Sets up the "HBNOISE" analysis with reference net of "/OUT1". The input source is "/V12". The sweep is from 1K to 1M with 10 steps of "LIN" type. The output frequency is small signal harmonic 3, 4, 5, plus the fundamental frequency.

```
snpsAnalysis('HBNOISE ? nodeType "net" ?  
probeNodeN1 "/OUT" ?inSrc "/V1" ?sweepType "LIN" ?  
sweepStart "1k" ?sweepStop "1M" ?  
sweepStep "10" ?outputFreqIdx "3 4 5 +1" ?listFreq "2K, 4k")
```

Example 4: Sets up the "HBNOISE" analysis with reference net of "/OUT". The input source is "/V1". The sweep is from 1K to 1M with 10 steps of "LIN" type. The output frequency is small signal harmonic 3, 4, 5, plus the fundamental frequency. The "listfreq" for noise is 2K and 4K. The "listsource" mean not to print element source noise.

```
snpsAnalysis('HBNOISE ? nodeType "net" ?  
probeNodeN1 "/OUT" ?inSrc "/V1" ?sweepType "LIN" ?  
sweepStart "1k" ?sweepStop "1M" ?sweepStep "10" ?  
outputFreqIdx "3 4 5 +1" ?listFreq "2K, 4k" ?listSource "off" )
```

Example 5: Sets up the "HBNOISE" analysis with reference instance of "/V1". The input source is "/V2". The sweep is from 1K to 1M with 10 steps of "LIN"

type. The output frequency is small signal harmonic 2, 4, minus the fundamental frequency. The "listfreq" for noise is 3K and 5K.

```
snpsAnalysis('HBNOISE ? nodeType "instance" ?  
probeNodeN1 "/V1" ?inSrc "/V2" ?sweepType "LIN" ?  
sweepStart "1K" ?sweepStop "1M" ?sweepStep "10" ?  
outputFreqIdx "2 4 -1" ?listFreq "3k 5k" )
```

Example 6: This setup will issue an error message that "node" is an unknown parameter.

```
snpsAnalysis('HBNOISE ? node "net" ?probeNodeN1 "/OUT" ?  
inSrc "/V1" ?sweepType "LIN" ?sweepStart "1k" ?  
sweepStop "1M" ?sweepStep "10")
```

HBAC Examples

Example 1: Sets up "HBAC" analysis. The sweep is from 1k to 1M with 10 steps. The sweep type is "DEC".

```
snpsAnalysis('HBAC ?sweepType "DEC" ?sweepStart "1k" ?  
sweepStop "1M" ?sweepStep "10" )
```

Example 2: Sets up "HBAC" analysis. The sweep type is "POI" and the sweep is 1k and 1M with 2 points.

```
snpsAnalysis('HBAC ?sweepType "POI" ?sweepListPoints "1k 1M" )
```

HBOSC Examples

Example 1: Sets up the "HBOSC" analysis with reference frequency of 1G and harmonic of 4. The reference positive net is "/OUT" and negative one is "/OUTN". The separate current source is 1.

```
snpsAnalysis('HBOSC ?fundTones "1G" ?harmNums "4" ?probeNodeN1  
"/OUT" ?probeNodeN2
```

Example 2: Sets up the "HBOSC" analysis with reference frequency of 1G and harmonic of 4. The reference positive net is "/OUT1" and negative one is "/OUTN1". The separate current source is 1. The minimum search frequency point is 0.9G; the maximum search frequency point is 1.2G and the search count is 6.

```
snpsAnalysis('HBOSC ?fundTones "1G" ?harmNums "4" ?probeNodeN1  
"/OUT1" ?  
probeNodeN2 "/OUTN1" ?probeNodeVP "1" ?  
stpsNum "6" ?stpsMin "0.9G" ?stpsMax "1.2G")
```

Example 3: Sets up the "HBOSC" analysis with reference frequency of 1G and harmonic of 4. The reference positive net is "/OUT1" and negative one is

Appendix C: OCEAN API Functions for HSPICE RF Analysis

Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis

"/OUTN1". The separate current source is 1. The minimum search frequency point is 0.9G; the maximum search frequency point is 1.2G and the search count is 6. The stability is -1, which means the single point stability plus the estimation of the amplitude is performed. The subharmonic is 5.

```
snpsAnalysis('HBOSC ?fundTones "1G" ?harmNums "4" ?  
probeNodeN1 "/OUT1" ?probeNodeN2 "/OUTN1" ?probeNodeVP "1" ?  
stpsNum "6" ?stpsMin "0.9G" ?stpsMax "1.2G" ?  
stability "-1" ?subharms "8" )
```

Example 4: Sets up the "HBOSC" analysis with reference frequency of 1G and harmonic of 4. The inter-modulation index is 5. The reference positive net is "/OUT" and negative one is "/OUTN". The separate current source is 1. The stability is 0. The sweep variable "ddd" is from 1 to 10 with the step of 10 and the sweep type is "LIN".

```
snpsAnalysis('HBOSC ?fundTones "1G" ?harmNums "4" ?  
interMaxOrder "5" ?ssTone "7" ?probeNodeN1 "/OUT" ?probeNodeN2  
"/OUTN" ?probeNodeVP "1" ?stability "0" ?  
sweepVariable "ddd" ?sweepType "LIN" ?sweepStart "1" ?  
sweepStop "10" ?sweepStep "10" )
```

Example 5: This setup will issue an error message that vp is not set up.

```
snpsAnalysis('HBOSC ?fundTones "1G" ?harmNums "4" ?  
probeNodeN1 "/OUT" ?probeNodeN2 "/OUTN" )
```

SN Examples

Example 1: Sets up the "SN" analysis under time domain. Its time resolution is 1e-6 and period is 2u.

```
snpsAnalysis('SN ?timeReso "1e-6" ?basePeriod "2u")
```

Example 2: Sets up the "SN" analysis under frequency information. Its fundamental tone is 1G and harmonic is 6.

```
snpsAnalysis('sn ?fundTones "1G" ?harmNums "6" )
```

Example 3: Sets up the "SN" analysis under time domain. Its time resolution is 1e-6 and period is 2u. The transient time is 3u. The maximum transient initialization time is 6 clock cycles.

```
snpsAnalysis('SN ?timeReso "1e-6" ?  
basePeriod "2u" ?trinit "3u" ?maxTrinitCyCs "6" )
```

Example 4: Sets up the "SN" analysis under frequency information. Its fundamental tone is 1G and harmonic is 6. The transient time is 4u. The maximum transient initialization time is 16 clock cycles.

```
snpsAnalysis('sn ?fundTones "1G" ?  
harmNums "6" ?trinit "4u" ?maxTrinitCycs "16")
```

Example 5: Sets up the "SN" analysis under frequency information. Its fundamental tone is 1G and harmonic is 6. The transient time is 4u. The maximum transient initialization time is 16 clock cycles. The temperature sweep is from 200k to 350k with the step of 10 and the sweep type is "LIN".

```
snpsAnalysis('sn ?fundTones "1G" ?harmNums "6" ?  
trinit "4u" ?maxTrinitCycs "16" ?sweepVariable "Temperature" ?  
sweepType "LIN" ?sweepStart "200k" ?sweepStop "350k" ?  
sweepStep "10")
```

Example 6: This setup will issue an error message that "fundTones" is incorrectly included together with "timeReso".

```
snpsAnalysis('SN ?timeReso "1e-6" ?  
basePeriod "2u" ?fundTones )
```

SNNOISE Examples

Example 1: Sets up the "SNNOISE" analysis with reference net of "/OUT". The input source is "/V1". The sweep is from 1K to 1M with 10 steps of "Lin" type.

```
snpsAnalysis('SNNOISE ?nodeType "net" ?  
probeNodeN1 "/OUT" ?inSrc "/V1" ?sweepType "LIN" ?sweepStart "1k"  
?sweepStop "1M" ?  
sweepStep "10")
```

Example 2: Sets up the "SNNOISE" analysis with reference instance of "/V1". The input source is "/V2". The sweep is from 1K to 1M with 10 steps of "LIN" type.

```
snpsAnalysis('SNNOISE ?nodeType "instance" ?  
probeNodeN1 "/V1" ?inSrc "/V2" ?sweepType "DEC" ?  
sweepStart "1K" ?sweepStop "1M" ?sweepStep "10")
```

Example 3: Sets up the "SNNOISE" analysis with reference net of "/OUT". The input source is "/V1". The sweep is from 1K to 1M with 10 steps of "Lin" type. The output frequency is small-signal harmonic 3, 4, 5, plus the fundamental

frequency. The "listfreq" for noise is 2K and 4K. The "listsource" means not to print element source noise.

```
snpsAnalysis('SNNOISE ? nodeType "net" ?  
probeNodeN1 "/OUT" ?inSrc "/V1" ?sweepType "LIN" ?  
sweepStart "1k" ?sweepStop "1M" ?sweepStep "10" ?  
outputFreqIdx "3 4 5 +1" ?listFreq "2K, 4k" ?listSource "off"
```

Example 4: Sets up the "SNNOISE" analysis with reference instance of "/V1". The input source is "/V2". The sweep is from 1K to 1M with 10 steps of "LIN" type. The output frequency is small signal harmonic 2, 4 minus the fundamental frequency. The "listfreq" for noise is 3K and 5K.

```
snpsAnalysis('SNNOISE ?nodeType "instance" ?  
probeNodeN1 "/V1" ?inSrc "/V2" ?sweepType "LIN" ?  
sweepStart "1K" ?sweepStop "1M" ?sweepStep "10" ?  
outputFreqIdx "2 4 -1" ?listFreq "3k 5k" )
```

SNOSC Examples

Example 1: Sets up the "SNOSC" analysis under time domain. Its time resolution is 1e-6 and period is 2u.

```
snpsAnalysis('SNOSC ?timeReso "1e-6" ?basePeriod "2u" ?oscnode  
"/OUT")
```

Example 2: Sets up the "SNOSC" analysis under frequency domain. Its fundamental tone is 1G and harmonic is 6.

```
snpsAnalysis('SNOSC ?fundTones "1G" ?  
harmNums "6" ?oscnode "/OUT")
```

Example 3: Sets up the "SNOSC" analysis under time domain. Its time resolution is 1e-6 and period is 2u. The transient time is 3u. The maximum transient initialization time is 4 clock cycles.

```
snpsAnalysis('SNOSC ?timeReso "1e-6" ?basePeriod "2u" ?oscnode  
"/OUT" ?trinit "3u" ?maxTrinitCycs "4" )
```

Example 4: Sets up the "SNOSC" analysis under frequency domain. Its fundamental tone is 1G and harmonic is 6. The transient time is 3u. The maximum transient initialization time is 4 clock cycles.

```
snpsAnalysis('SNOSC ?fundTones "1G" ?harmNums "6" ?  
oscnode "/OUT" ?trinit "3u" ?maxTrinitCycs "4" )
```

Example 5: Sets up the "SNOSC" analysis under frequency domain. Its fundamental tone is 1G and harmonic is 6. The transient time is 3u. The

maximum transient initialization time is 4 clock cycles. The temperature sweep is from 200k to 350k with the step of 10 and the sweep type is "LIN".

```
snpsAnalysis('SNOSC ?fundTones "1G" ?harmNums "6" ?  
oscnodet "/OUT" ?trinit "3u" ?maxTrinitCycs "4" ?  
sweepVariable "Temperature" ?sweepType "LIN" ?  
sweepStart "200k" ?sweepStop "350k" ?sweepStep "10")
```

SNAC Examples

Example 1: Sets up "SNAC" analysis. The sweep is from 1k to 1M with 10 steps. The sweep type is "DEC"

```
snpsAnalysis('SNAC ?sweepType "DEC" ?sweepStart "1k" ?sweepStop  
"1M" ?sweepStep "10" )
```

Example 2: Sets up "SNAC" analysis. The sweep type is "POI" and the sweep is 1k and 1M with 2 points.

```
snpsAnalysis('SNAC ?sweepType "POI" ?sweepListPoints "1k 1M" )
```

PHASENOISE Examples

Example 1: Sets up the "PHASENOISE" analysis with the node of "/OUT" as phase noise calculation reference. The sweep is from 1k to 1M and with steps of 10 and type of "LIN". The method is 1 and carrier index is 2 for phase noise calculation. The listfreq is "all" and listcount is 5 for element phase noise frequencies number. The minimum dumped noise value is -200 dBc/Hz. The listsource is "ON" which means to dump all the element sources. The spurious is 1, which activates the additional "SPUR" analysis.

```
snpsAnalysis('PHASENOISE ?nodeType "net" ?  
probeNodeN1 "/OUT" ?sweepType "LIN" ?sweepStart "1k" ?  
sweepStop "1M" ?sweepStep "10" ?method "1" ?  
carrierIndex "2" ?listFreq "all" ?listCount "5" ?  
listFloor "-200" ?listSource "on" ?spurious "1" )
```

Example 2: Sets up the "PHASENOISE" analysis with the differential node of positive one, "/OUT" and negative one, "/OUTN" as phase noise calculation references. The sweep is from 1k to 1M and with steps of 10 and type of "OCT". The method is 2 and carrier index is 3 for phase noise calculation. The listfreq is "none". The spurious is 0, which de-activates the additional "SPUR" analysis.

```
snpsAnalysis('phasenoise ?nodeType "dnet" ?  
probeNodeN1 "/OUT" ?probeNodeN2 "/OUTN" ?  
sweepType "OCT" ?sweepStart "1K" ?sweepStop "1M" ?  
sweepStep "10" ?method "2" ?carrierIndex "3" ?  
listFreq "none" ?spurious "0" )
```

2 snpsProbe

snpsProbe function:

Syntax:

```
snpsProbe(s_outputType ?parameter t_value [?parameter t_value ...  
] )  
=>t/nil
```

Description

Sets up the information to save the signal data for HSPICERF analysis. It can save the result for current, voltage, power and hertz information, as well as the noise for plotting. It supports HB and SN-related large signal and small signal analyses. The function works similar to the "save()" function from ADE.

Note: parameter and t_value should be input together. Once the "parameter" is given, its "t_value" should also be provided; the value parameters should be input as a string with double quotes.

Arguments	Description
S_outputType"	The symbol of output type; possible values are as follows: <ul style="list-style-type: none">▪ 'i' = current▪ 'v' = voltage▪ 'p' = power▪ 'h' = hertz▪ 'PHASENOISE' = phase noise plotting▪ 'HBNOISE' = noise plotting when doing HB-related analysis▪ 'SNNOISE' = noise plotting when doing SN-related analysis

Appendix C: OCEAN API Functions for HSPICE RF Analysis

Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis

Arguments	Description
parameter	<p>The parameter name that is set up with the saved signal. According to s_outputType value, they are as follows:</p> <p>For 'i'</p> <ul style="list-style-type: none"> ▪ "signal": Signal or element name, required for non-hertz outputs, required ▪ "anatype": Analysis type. Valid values are as follows: hb, sn, snac, snfd, hbtran ▪ "nodeType": Node type. Valid values are: (1) "terminal" (2) "dterminal": differential terminal ▪ "indexTones": Tone index ▪ "plotEnable": Plotted or not ▪ "outputName": Plot name ▪ "outputExpr": Expression to be plotted <p>For 'v'</p> <ul style="list-style-type: none"> ▪ "signal": Signal or element name, required for non-hertz outputs, required ▪ "anatype": Analysis type. Valid values are as follows: hb, sn, snac, snfd, hbtran ▪ "nodeType": Node type. Valid values are: (1) "net" (2) "dnet": differential net ▪ "indexTones": Tone index ▪ "plotEnable": Plotted or not ▪ "outputName": Plot name ▪ "outputExpr": Expression to be plotted <p>For 'p'</p> <ul style="list-style-type: none"> ▪ "signal": Signal or element name, required for non-hertz outputs, required ▪ "anatype": Analysis type. Valid values are as follows: hb, sn, and snac ▪ "nodeType": Node type. Valid value is: instance ▪ "indexTones": Tone index ▪ "plotEnable": Plotted or not ▪ "outputName": Plot name ▪ "outputExpr": Expression to be plotted

Appendix C: OCEAN API Functions for HSPICE RF Analysis

Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis

Arguments	Description
	<p>For 'h (hertz)</p> <ul style="list-style-type: none"> ▪ "anatype": Analysis type. Valid values are hb, sn, snac, snfd ▪ "indexTones": Tone index ▪ "plotEnable": Plotted or not <p>For 'PHASENOISE</p> <ul style="list-style-type: none"> ▪ "noiseTerm" Values includes "JITTER", "la", "phnoise", "ltotal", and "onoise" ▪ "noiseType" ▪ Values includes "fedp", "infedp", "cyclo", "stationary", "flicker", "cycloflicker", and "default" ▪ "nodeType" ▪ Node type. Valid value is: "instance" ▪ "anatype": Analysis type. Valid values are hb, sn, or phasenoise ▪ "indexTones": Tone index <p>For 'SNNOISE and 'HBNOISE</p> <ul style="list-style-type: none"> ▪ "value": Includes "INOISE", "ONNOISE", "NF", "SSNF", and "DSNF" ▪ "unit": Includes "db", "dbm", and "M"
t_value	The value of the parameter

Value Returned:

t RF setting succeeded.

nil One or more errors; note details in the printed report.

Examples for 'i, 'v, 'p, 'hertz, 'SNNOISE, 'HBNOISE, 'PHASENOISE

Example 1: For 'i — Sets up the "/I0/CLK" current output of "HBTRAN" analysis type.

```
snpsProbe('i ?nodeType "terminal" ?
anatype "HBTRAN" ?signal "/V1/PLUS" )
```

Example 2: Sets up the "/ V1/PLUS " current output of "hb" analysis type.

```
snpsProbe('i ?nodeType "terminal" ?anatype "hb" ?signal "/V1/
PLUS" )
```

Example 3: For 'v—Sets up the "/OUT" voltage output of "HB" analysis type .

```
snpsProbe('v ?nodeType "net" ?anatype "hb" ?signal "/OUT" )
```

Example 4: Sets up the differential "/OUT,OUTN" voltage output of "SNFD" analysis.

```
snpsProbe('v ?nodeType "dnet" ?anatype "SNFD" ?signal "/OUT,OUTN"
)
```

Example 5: Sets up the differential "/OUT,OUTN" voltage output of "snfd" analysis, the same as "SNFD" analysis.

```
snpsProbe('v ?nodeType "dnet" ?anatype "snfd" ?
signal "/OUT,OUTN" )
```

Example 6: Sets up the differential "/OUT,OUTN" voltage output of "snac" analysis.

```
snpsProbe('v ?nodeType "dnet" ?anatype "snac" ?signal "/OUT,OUTN"
)
```

Example 7: For 'p— Sets up the "/V1" power output of "hb" analysis.

```
snpsProbe('p ?nodeType "instance" ?anatype "hb" ?signal "/V1" )
```

Example 8: Sets up the "/V1" power output of "snac" analysis.

```
snpsProbe('p ?nodeType "instance" ?anatype "snac" ?signal "/V1" )
```

Example 9: For 'h—Sets up the hertz output of "hb" analysis.

```
snpsProbe('hertz ?anatype "hb" )
```

Example 10: For 'SNNOISE/'HBNOISE—Sets up the "DSNF" output in unit of "M" for "HBNOISE" analysis.

```
snpsProbe('HBNOISE ?value "DSNF" ?unit "M" )
```

Example 11: Sets up the "INOISE" output in default unit for "SNNOISE" analysis.

```
snpsProbe('SNNOISE ?value "INOISE" ?unit "default" )
```

Example 12: Sets up the "INOISE" output in "db" unit for "SNNOISE" analysis.

```
snpsProbe('SNNOISE ?value "INOISE " ?unit "db" )
```

Example 13: Sets up the "NF" output in "dbm" unit for "HBNOISE" analysis

```
snpsProbe('HBNOISE ?value "NF" ?unit "dbm" )
```

Appendix C: OCEAN API Functions for HSPICE RF Analysis

Using the HSPICE-Provided OCEAN API for HSPICE RF Analysis

Example 14: For 'PHASENOISE—Sets up the "/V3" phase noise output for "hb" analysis.

```
snpsProbe('PHASENOISE  ?nodeType "instance" ?  
anatype "hb" ?signal "/V3" )
```

Example 15: Sets up the "JITTER" output in "PHASENOISE" analysis.

```
"          snpsProbe('PHASENOISE  ?noiseTerm "JITTER" )
```

Example 16: Sets up the "la_indfedp" output for "PHASENOISE" analysis.

```
snpsProbe('PHASENOISE  ?noiseTerm "la" ?noiseType "indfedp" )
```

Example 17: Sets up the "phnoise_ stationary" output for "PHASENOISE" analysis.

```
snpsProbe('PHASENOISE  ?noiseTerm "phnoise" ?noiseType  
"stationary" )
```

OCEAN API Functions for HSPICE Corner Analysis

Describes HSPICE-defined OCEAN functions provided to run HSPICE Corner analyses.

Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

The **Save OCEAN** script option under the File menu of the HSPICE Corner Analysis form saves all operations performed in the HSPICE Corner Analysis window as an OCEAN script file to be executed under the Non-graphic mode. See [Chapter 9, Corners Analysis](#) for a description of the HSPICE Corner Analysis form and its functionality. You can use these OCEAN API functions or the Virtuoso general OCEAN functions to write the OCEAN script directly to simulate in batch mode.

When you combine these HSPICE Corner OCEAN functions with the native Cadence OCEAN functions, you can run a Corner Analysis in batch mode without the GUI.

The following are the HSPICE-defined OCEAN functions provided to run Corner analyses.

1. **snpsCorAddModelFile** function:

Syntax:

```
snpsCorAddModelFile (t_modelFileName) => t/nil
```

Appendix D: OCEAN API Functions for HSPICE Corner Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

Description: Adds a model file to the corners environment.

Argument	Description
t_modelFileName	Model file name with path included.

Value Returned:

t The model file addition succeeded.
nil One or more errors; note details in the printed report.

Example:

```
snpsCorAddModelFile("/remote/user/bjt90nm.lib") => t
```

2. **snpsCorAddDesignVar** function:

Syntax:

```
snpsCorAddDesignVar (t_varName) => t/nil
```

Description: Adds a design variable to the corners environment.

Argument	Description
t_varName	Name of the design variable.

Value Returned:

t The design variable addition succeeded.
nil One or more errors; note details the printed report.

Example:

Adding "vdd" to the HSPICE Corner Analysis simulation run:

```
snpsCorAddDesignVar("vdd")
```

3. **snpsCorAddTemperature** function:

Syntax:

```
snpsCorAddTemperature() => t/nil
```

Description: Adds an environment variable with name "temperature" to the corners environment.

Argument	Description
none	

Value Returned:

t The "temperature" addition succeeded.
nil One or more errors; note details the printed report.

4. **snpsCorDeleteModelFile** function:

Syntax

```
snpsCorDeleteModelFile (t_modelFileName) => t/nil
```

Description: Deletes a model file in the corners environment.

Note: The `t_modelFileName` parameter should include its path.

Argument	Description
t_modelFileName	Model file name to be deleted.

Value Returned:

t The model file deletion succeeded.
nil One or more errors; note details the printed report.

Example:

```
snpsCorDeleteModelFile("/remote/user/bjt90nm.lib") => t
```

5. **snpsCorDeleteAllModelFiles** function:

Syntax:

```
snpsCorDeleteAllModelFiles () => t/nil
```

Description: Deletes all model files that have been added with `snpsCorAddModelFile`.

Argument	Description
none	

Appendix D: OCEAN API Functions for HSPICE Corner Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

Value Returned:

t The model files deletion succeeded.
nil One or more errors; note details the printed report.

6. **snpsCorDeleteDesignVar** function:

Syntax:

```
snpsCorDeleteDesignVar (t_varName) => t/nil
```

Description: Deletes a design variable from the corners environment.

Argument	Description
t_varName	Name of the design variable to be deleted.

Value Returned:

t The design variable deletion succeeded.
nil One or more errors; note details the printed report.

Example:

Deleting "vdd" from the HSPICE Corner Analysis simulation run:

```
snpsCorDeleteDesignVar ("vdd")
```

7. **snpsCorDeleteDesignVars** function:

Syntax:

```
snpsCorDeleteDesignVars () => t/nil
```

Description: Deletes all design variables to the corners environment.

Argument	Description
none	

Value Returned:

t The design variables deletion succeeded.
nil One or more errors; note details the printed report.

Example:

```
none
```

8. **snpsCorDeleteTemperature** function:

Syntax

```
snpsCorDeleteTemperature () => t/nil
```

Description: Deletes the environment variable for the temperature that had been added with `snpsCorAddTemperature`

Argument	Description
none	

Value Returned:

```
t      The "temperature" deletion succeeded.
nil    One or more errors; note details the printed report.
```

Example:

```
none
```

9. **snpsCorAddCorner** function:

Syntax:

```
snpsCorAddCorner (t_cornerName ?vars l_vars ?models l_models
?temp f_temp) => t/nil
```

Description: Adds a corner.

Note: The models parameter uses the model file name (not including the path); therefore, *do not* use the model file which has the same model file name, but a different path.

Arguments	Description
t_cornerName	Name of the corner
l_vars	Design variables
l_models	Model files
f_temp	Temperature

Value Returned:

```
t      The the corner addition succeeded.
nil    One or more errors; note details the printed report.
```

Appendix D: OCEAN API Functions for HSPICE Corner Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

Example: This example adds a corner named "fastslow" whose design variables values are: "vdd" "1.0", "channelEffect" "1.12"; models are: "bjt90nm.lib" which uses the "fast" model, and "nmos90nm.lib", which uses the "slow" model; temperature is 27.

```
snpsCorAddCorner("fastslow" ?vars ' ("vdd" "1.0")
("channelEffect" "1.12")) ?models ' ( ("bjt90nm.lib" "fast")
("nmos90nm.lib" "slow") ?temp 27)
```

10. **snpsCorCopyCorner** function:

Syntax:

```
snpsCorCopyCorner (t_fromCorner t_toCorner) => t/nil
```

Description: Copies one corner to another.

Argument	Description
t_fromCorner	Corner to be copied
t_toCorner	New corner that is to be copied from t_fromCorner

Value Returned:

```
t      The corner copy operation succeeded.
nil    One or more errors; note details the printed report.
```

Example:

Copies "cor1" to "cor2"; "cor2" has the same setting as "cor1":

```
snpsCorCopyCorner("cor1" "cor2")
```

11. **snpsCorSetDesignVarVal** function:

Syntax:

```
snpsCorSetDesignVarVal (t_cornerName t_varName t_value) => t/
nil
```

Description: Sets the design variable to the given value for the specified corner.

Arguments	Description
t_cornerName	Name of the corner
t_varName	Name of the design variable
t_value	Design variable value

Value Returned:

```
t      The corner's design variable value setting succeeded
nil    One or more errors; note details the printed report.
```

Example:

```
none
```

12. **snpsCorSetModelFileVarVal** function:

Syntax

```
snpsCorSetModelFileVarVal (t_cornerName t_modelFileName
t_value) => t/nil
```

Description: Sets the section name in the model file to be used for the specified corner.

Note: The model file parameter uses the model file name (not the include path); therefore, do not use the model file which has the same model file name, but different path.

Argument	Description
t_cornerName	Name of the corner.
t_modelFileName	Name of the model file.
t_value	Section name.

Value Returned:

```
t      Setting the corner's model file value succeeded.
nil    One or more errors; note details the printed report.
```

Appendix D: OCEAN API Functions for HSPICE Corner Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

Example: Setting the "fastfast" corner's "bjt90nm.lib" model file's model to "fast":

```
snpsCorSetModelFileVarVal("fastfast" "bjt90nm.lib" "fast")
```

13. **snpsCorSetTempVal** function:

Syntax:

```
snpsCorSetTempVal (t_cornerName t_value) => t/nil
```

Description: Sets the value of the temperature to be used for the specified corner.

Argument	Description
t_cornerName	Name of the corner.
t_value	Temperature value.

Value Returned:

```
t      The corner's "Temperature" value setting succeeded.
nil    One or more errors; note details the printed report.
```

Example:

```
none
```

14. **snpsCorDeleteCorner** function:

Syntax:

```
snpsCorDeleteCorner (t_cornerName) => t/nil
```

Description: Deletes the corner named *t_cornerName*.

Argument	Description
none	

Value Returned:

```
t      The corner deletion succeeded.
nil    One or more errors; note details the printed report.
```

Example:

```
none
```

15. **snpsCorDeleteAll Corners** function:

Syntax:

```
snpsCorDeleteAllCorners () => t/nil
```

Description: Deletes all corners that have been added with
`snpsCorAddCorner`.

Argument	Description
none	

Value Returned:

```
t      The corners deletion succeeded.
nil    One or more errors; note details the printed report.
```

Example:

```
none
```

16. **snpsCorAddMeas** function:

Syntax:

```
snpsCorAddMeas (t_measName) => t/nil
```

Description: Adds a new measurement variable.

Argument	Description
t_measName	Name of the measurement variable.

Value Returned:

```
t      The measurement variable addition succeeded.
nil    One or more errors; note details the printed report.
```

Example:

```
none
```

17. **snpsCorSetMeasSummarizeOn** function:

Syntax:

```
snpsCorSetMeasSummarizeOn (t_measName g_value) => t/nil
```

Appendix D: OCEAN API Functions for HSPICE Corner Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

Description: Enables or disables the measurement variable summary (which will parse all measurement results files and evaluate them according to the measurement variable "Goal") after Corner Analysis simulation.

Note: The measurement variable must have been added into the Corner Analysis environment using the `snpsAddMeas` function.

Argument	Description
t_measName	Name of the measurement.
g_value	Boolean variable: t = enable summarize, nil = disable summarize.

Value Returned:

t The measurement variable enabling/disabling succeeded.
nil One or more errors; note details the printed report.

Example: Enabling the "trise" summary.

```
snpsCorSetMeasSummarizeOn("trise" t)
```

18. `snpsCorSetMeasGoal` function:

Syntax:

```
snpsCorSetMeasGoal (t_measName ?goalType t_goalType ?lowValue  
t_lowValue ?value t_value) => t/nil
```

Description: Sets measurement variable summarize "Goal".

Note: The measurement variable must have been added into the Corner Analysis environment.

Argument	Description
t_measName	Name of the measurement

Appendix D: OCEAN API Functions for HSPICE Corner Analysis
Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

Argument	Description
t_goalType	The goalType can be any one of the following choices: "Range": goal should between t_lowValue and t_value "+/-": goal should between -abs(t_value) and abs(t_value) "<": goal should less than the t_value "<=": goal should not larger than the t_value "=": goal should equal the t_value ">": goal should larger than t_value ">=": goal should not be less than the t_value
t_lowValue	When the goalType is "Range", this parameter should to point to the lower value.
t_value	When the goalType is "Range", this parameter should to point to the higher value; for any other goalType, it means the goal value

Value Returned:

t Setting the measurement variable goal succeeded.
nil One or more errors; note details the printed report.

Example: Sets the "trise" summarize goal to "200p,500p":

```
snpsCorSetMeasGoal("trise" ?goalType "Range" ?lowValue "200p"  
?value "500p")
```

19. snpsCorCorLoadMeasFromFile function:

Syntax:

```
snpsCorLoadMeasFromFile (t_measFile) => t/nil
```

Description: Loads measurement variables from measure file and adds them to Corner Analysis environment.

Appendix D: OCEAN API Functions for HSPICE Corner Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

Note: This function also loads the measurement variables goal if the goal has been set in the measure file with the syntax "goal=value", and enables the measurement variables' summarization.

Argument	Description
t_measFile	The measurement file path and name.

Value Returned:

t The loading of measurement variables succeeded.
nil One or more errors; note details the printed report.

Example:

none

20. **snpsCorDeleteMeas** function:

Syntax

```
snpsCorDeleteMeas (t_measName) => t/nil
```

Description: Deletes measurement variable form the Corner Analysis environment.

Argument	Description
none	

Value Returned:

t Deletion of the measurement variable succeeded.
nil One or more errors; note details the printed report.

Example:

none

21. **snpsCorDeleteAllMeas** function:

Syntax:

```
snpsCorDeleteAllMeas () => t/nil
```

Description: Deletes all measurements that have been added with snpsCorAddMeas or snpsCorLoadMeasFromFile.

Note: The measurement variable must have been added into the Corner Analysis environment with the `snpsCorAddMeas` function.

Arguments	Description
none	

Value Returned:

```
t      Deletion of measurement variables succeeded.
nil    One or more errors; note details the printed report.
```

Example:

```
none
```

22. **snpsCorRun** function:

Syntax:

```
snpsCorRun(@rest arglist)
```

Description: Runs the Corners Analysis using HSPICE. This is a wrap function of the `ocean` function "`run()`", which performs the extra job of invoking Corner Analysis netlisting and post-processing after simulation.

Argument	Description
This arguments is the same as the OCEAN function <code>run()</code> .	

Values Returned:

```
Same as those returned by Ocean run()
```

Example: Run all analysis settings specified in the HSPICE Corner Analysis:

```
snpsCorRun()
```

23. **snpsCorPlotOutputs** function:

Syntax:

```
snpsCorPlotOutputs(t_output ?type s_type ?analysis t_analysis
?cornersSelect s_cornerSelect ?corners l_corners ?plotMode
t_plotMode)=> t/nil
```

Appendix D: OCEAN API Functions for HSPICE Corner Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

Description: Plots the selected corners or all corners-specific analysis output.

Argument	Description
t_output	output signal or expression
s_type	The output signal type ('net or 'terminal); if it's an expression, it should be 'expr
s_cornerSelect	Plots either all corners output waveforms or (select) selected corners output waveform; value should be 'all or 'select. If it's select, the l_corners should be provided.
l_corners	When the s_cornerSelect is selected, the l_corners parameter should be provided to plot corners.
t_plotMode	Specify the window mode of plotting window, it's valid value are: "Append" (Default): Adds new waveform to the existing waveform window "Replace": Replaces existing window's waveform "New Subwin": Opens a new subwindow in the existing waveform window and appends new waveform "New Win": Opens a new waveform window in the existing waveform window and appends new waveform

Value Returned:

t Specified plots succeeded.
nil One or more errors; note details the printed report.

Example: Plot the "cor1" and "cor3" analyses "/out" signal waveforms:

```
snpsCorPlotOutputs("/out" ?type 'net ?analysis "TRAN"
?cornerSelect 'select ?corners '("cor1" "cor3"))
```

24. snpsCorPrintMeas function:

Syntax:

```
snpsCorPrintMeas () => t/nil
```

Appendix D: OCEAN API Functions for HSPICE Corner Analysis
Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

Description: Prints out the measurement variables all corners results and summarize information using a table format.

Argument	Description
none	

Value Returned:

t The printing of the measurement variables' corners results and summarization succeeded.
nil One or more errors; note details the printed report.

Example:

none

25. **snpsCorPlotMeas** function:

Syntax:

```
snpsMontePlotMeas(t_measName ?plotMode t_plotMode) =>t/nil
```

Description: Plots the measurement variable values with different corners. The waveform's X-axis denotes the simulated corners.

Argument	Description
t_measName	The measurement variable name.
t_plotMode	Specify the window mode of plotting window; valid values are "Append" (Default): Adds new waveform to the existing waveform window "Replace": Replaces existing window's waveform "New Subwin": Opens a new subwindow in the existing waveform window and appends new waveform "New Win": Opens a new waveform window in the existing waveform window and appends new waveform

Appendix D: OCEAN API Functions for HSPICE Corner Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Corner Analysis

Value Returned:

t Waveform plot of the measure results succeeded.
nil One or more errors; note details the printed report.

Example:

none

OCEAN API Functions for HSPICE Optimization Analysis

Describes HSPICE-defined OCEAN functions provided to run HSPICE Optimization analyses.

Using the HSPICE-Provided OCEAN API for HSPICE Optimization Analysis

The **Save OCEAN** script option under the File menu of the HSPICE Optimization Analysis form saves all operations performed in the HSPICE Optimization Analysis window as an OCEAN script file to be executed under the Non-graphic mode. See [Chapter 11, HSPICE Optimization Analysis](#) for a description of the HSPICE Optimization Analysis form and its functionality. You can use these OCEAN API functions or the Virtuoso general OCEAN functions to write the OCEAN script directly to simulate in batch mode.

When you combine these HSPICE Optimization OCEAN functions with the native Cadence OCEAN functions, you can run a Optimization Analysis in batch mode without the GUI.

The following are the HSPICE-defined OCEAN functions provided to run Optimization analyses.

1. **snpsOptimizeModelFile** function:

Syntax:

```
snpsAnalysisModel(?parameter t_value [?parameter t_value... ])  
=> t/nil
```

Description: Sets up Optimize model information.

Appendix E: OCEAN API Functions for HSPICE Optimization Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Optimization Analysis

Note: `parameter` and `t_value` should be input together. Once the "parameter" is given, its "t_value" should also be provided; the value parameters should be input as string with double quotes.

Arguments	Description
<code>parameter</code>	<p>The parameter of the module. Corresponding to HSPICE Optimize analysis, the parameters include:</p> <ul style="list-style-type: none"> ▪ <code>name</code>: Name of the model (required) ▪ <code>method</code>: Optimization method; valid values are "BISECTION" and "PASSFAIL" ▪ <code>close</code>: Initial estimate of how close parameter initial value estimates are to the solution. ▪ <code>max</code>: Upper limit on close ▪ <code>cut</code>: How much to modify the "close" estimate ▪ <code>difsiz</code>: Increment change in a parameter value for gradient calculations ▪ <code>grad</code>: How much the gradient of the results function is less than to represent possible convergence ▪ <code>parmin</code>: Allows better control of incremental parameter changes during error calculations ▪ <code>relin</code>: Relative input parameter for convergence. ▪ <code>relout</code>: Relative tolerance to finish optimization ▪ <code>itropt</code>: Maximum number of iterations ▪ <code>dynacc</code>: Dynamic accuracy tolerance setting ▪ <code>sendif</code>: Point at which more accurate derivations are desired
<code>t_value</code>	Value of the parameter.

Value Returned:

`t` The Optimization analysis setting succeeded.
`nil` One or more errors; note details in the printed report.

Examples

Example1: Sets up the optimize model. Its name is "opt" with the parameter "relin" of "1e-5", "close" of "10" and "relout" of "1e-5"

```
"snpsOptimizeModel(?name "opt" ?relin "1e-5" ?close "10"
?itropt "40" ?relout "1e-5")
```

Appendix E: OCEAN API Functions for HSPICE Optimization Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Optimization Analysis

Example 2: Sets up the optimize model. Its name is "optmod" with the parameter "relin" of "1e-4". It is a "PASSFAIL" timing analysis.

```
"snpsOptimizeModel (?name "optmod" ?relin "1e-4" ?method  
"passfail")
```

Example 3: Sets up the optimize model. Its name is "optmod" with the parameter "relin" of "1e-4". It is a "BISECTION" timing analysis.

```
"snpsOptimizeModel (?name "optmod" ?relin "1e-4" ?method  
"bisection")
```

Example 4: Sets up the optimize model. Its name is "optmod" with the parameter "relin" of "1e-4". The "dynacc" is "1".

```
"snpsOptimizeModel (?name "optmod" ?relin "1e-4" ?dynacc "1")
```

Example 5: Sets up the optimize model. Its name is "optmod" with the parameter "relin" of "1e-4". It is a "BISECTION" timing analysis.

```
"snpsOptimizeModel (?name "optmod" ?relin "1e-4" ?method  
"bisection")
```

Example 6: Error reveals that the model name is missing.

```
"snpsOptimizeModel (?relin "1e-4" ?method "bisection")
```

Example 7: Error reveals that the valid value for "dynacc" is "0" and "1".

```
"snpsOptimizeModel (?name "optmod" ?relin "1e-4"  
?dynacc "-1")
```

2. **snpsOptimizeVariable** function:

Syntax:

```
snpsAnalysisVariable( ?parameter t_value [?parameter t_value  
... ] )  
=>t/nil
```

Description: Sets up a single Optimization parameter.

Appendix E: OCEAN API Functions for HSPICE Optimization Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Optimization Analysis

Note: Both parameter and t_value should be input together. Once the "parameter" is given, its "t_value" should also be provided; the value parameters should be input as a string with double quotes.

Argument	Description
parameter	The parameter of the module. Corresponding to HSPICE Optimize analysis, the parameters include: <ul style="list-style-type: none">▪ name: name of the parameter (required)▪ initial: Initial guess (required)▪ low: Lower limit (required)▪ upper : Upper limit (required)▪ delta: delta guess▪ optGroup: Optimize group name (required)
t_value	Value of the parameter

Value Returned:

t Setting of the optimization parameter succeeded.
nil One or more errors; note details the printed report.

Examples

Example 1: Sets up the optimization parameter named "wml" with the parameter "upper" of "100u", "low" of "20u" and "initial" of "60u". It belongs to the "opt1" group.

```
"snpsOptimizeVariable(?name "wml" ?upper "100u" ?low "20u"  
?initial "60u" ?optGroup "opt1")
```

Example 2: Sets up the optimization parameter named "wm2" with the parameter "upper" of "100u", "low" of "20u" and "initial" of "60u". It belongs to "opt2". The delta is "0.1"

```
"snpsOptimizeVariable (?name "wm2" ?upper "100u" ?low "20u"  
?initial "60u" ?optGroup "opt2" ?delta "0.1" )
```

Example 3: Error is reported that the "initial guess is missing.

```
snpsOptimizeVariable (?name "wml" ?upper "100u" ?low  
"20u"?optGroup "opt1")
```

3. **snpsOptimizeAnalysis** function:

Appendix E: OCEAN API Functions for HSPICE Optimization Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Optimization Analysis

Syntax:

```
snpsAnalysisAnalysis( ?parameter t_value [?parameter t_value
... ] )
=>t/nil
```

Description: Sets up information for an optimization analysis.

Note: Both parameter and t_value should be input together. Once the "parameter" is given, its "t_value" should also be provided; the value parameters should be input as a string with double quotes.

Argument	Description
parameter	<p>The parameter of the module; corresponding to HSPICE Optimize analysis, parameters include:</p> <p>nam : Name of the optimize analysis, valid values, "DC", "AC" and "TRAN" are supported (required)</p> <p>optimize: Optimize group name (required)</p> <p>results: Optimization parameter names (required)</p> <p>model Optimization model name (required)</p> <p>data: Optimization data name</p>
t_value	Value of the parameter.

Value Returned:

```
t      The "temperature" addition succeeded.
nil    One or more errors; note details the printed report.
```

Example: Sets up the optimization analysis. The analysis is transient with the "opt1" optimize group. It uses the measures of "area_min", "delayf", "delayr" and "tot_power" and the optimization model named "opt".

```
"snpsOptimizeAnalysis(?name "TRAN" ?optimize "opt1"
?results "area_min,delayf,delayr,tot_power" ?model "opt")
```

4. **snpsOptimizeDisplay** function:

Syntax

Appendix E: OCEAN API Functions for HSPICE Optimization Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Optimization Analysis

```
snpsOptimizeDisplay ( [session] )  
=> t/nil
```

Description: Displays Optimization results information.

Argument	Description
session	Name of the current session (optional)

Value Returned:

```
t      The Optimization display setting succeeded. A new terminal  
       window with optimization results is shown.  
nil    One or more errors; note details the printed report.
```

Example: Displays the optimization results in a terminal window.

```
snpsOptimizeDisplay()
```

OCEAN API Functions for the HSPICE Measure Utility

Describes HSPICE-defined OCEAN functions provided to run the HSPICE Measure utility.

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

The **Save OCEAN** script option under the File menu of the HSPICE Corner Analysis form saves all operations performed in the HSPICE Corner Analysis window as an OCEAN script file to be executed under the Non-graphic mode. See [Chapter 12, HSPICE Measurement Utility](#) for a description of the HSPICE Corner Analysis form and its functionality. The Measure Utility can be combined with several tools such as Corner, Optimization, and Monte Carlo. It can be also used independently.

You can use these OCEAN API functions or the Virtuoso general OCEAN functions to write the OCEAN script directly to simulate in batch mode. When you combine these HSPICE Corner OCEAN functions with the native Cadence OCEAN functions, you can run measurements in batch mode without the GUI.

The following are the HSPICE-defined OCEAN functions provided to run the Measure utility.

1. **snpsMeasureTrigTarg** function function:

Syntax:

```
snpsMeasureTrigTarg(s_name ?parameter t_value
[ ... ?parameter t_value ... ]
=>nil
```

Description: Sets up an HSPICE measurement.

Appendix F: OCEAN API Functions for the HSPICE Measure Utility

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Note: The “parameter” and “t_value” should be input together except when the parameter is “trig” or “targ”. For “targ”, *only* 'targ is sufficient without a value. For “trig”, *only* 'trig is sufficient without a value. Once the other “parameter” is given, its “t_value” should also be provided; the value parameters should be input as a string with double quotes.

Argument	Description
s_name	Name of the measure; required.
parameter	<p>The parameter of the measure. Corresponding to a HSPICE measure statement; parameters include:</p> <ol style="list-style-type: none">1. <i>goal</i> Goal in the measure; optional2. <i>minval</i> Minimum value in the measure; optional3. <i>weight</i> Weight in the measure; optional4. <i>analysis</i> Analysis in the measure. Valid values: "TRAN," "DC," AC"5. <i>precision</i> MEASDGT option. Valid values are [1, ... , 10]6. <i>func</i> Function name that specifies the general measure function Valid values are "RMS", "MIN", "AVG", "MAX", "INTEG", "PP", "DERIV" and "EM_AVG"7. <i>trig</i> and <i>targ</i> Trigger and Target keywords indicate the following parameters (except for the above) are all for trig or targ measures. Only 'trig and 'targ are valid; ?trig and ?targ are not supported8. <i>var</i> Signal or variable9. <i>type</i> Signal or variable type. Valid values "V", "I","P", "VP", "VM", "VR", "VI", "IM", "IP", "IR", "II", "VDB" and "IDB"10. <i>val</i> The value of a signal or variable11. <i>event</i> The measure event. Valid values are "RISE", "FALL" and "CROSS"12. <i>eventValue</i> The value corresponding to the "RISE", "FALL" and "CROSS" measure events; Integer value or "last"13. <i>td</i> Time delay

Argument	Description
14. <i>at</i>	Used only in special case

Value Returned:

t The model file addition succeeded.
nil One or more errors; note details in the printed report.

Examples

Example 1. Sets up the trig/targ measure. The analysis is transient. The measure name is `time1`. The `trig` and `targ` specifications are all `v(net06) val=0.1 RISE=2`.

```
snpsMeasureTrigTarg('time1 ?analysis "TRAN" 'trig ?type "V"
?var "/net6" ?val "0.1" ?event "RISE" ?eventValue "2" 'targ
?type "V" + ?var "/net6" ?val "0.1" ?event "RISE" ?eventValue
"2")
```

Example 2. Sets up the trig/targ measure. The analysis is transient. The measure name is `"time1"`. The `trig` specifications are `"v(net06) val=0.1 RISE=2"` and the `targ` is at special case of time 2.

```
snpsMeasureTrigTarg('time1 ?analysis "TRAN" 'trig ?type "V"
?var "/net6"
?val "0.1" ?event "RISE" ?eventValue "2" 'targ ?at "2" )
```

2. **snpsMeasureFunc** function:

Syntax:

```
snpsMeasureFunc('s_name ?parameter t_value [?parameter
t_value ...])
```

Description: Sets up measurement information for a general function.

Note: The “parameter” and “t_value” should be input together except when the parameter is “trig” or “targ”. For “targ”, *only* 'targ is sufficient without a value. For “trig”, *only* 'trig is sufficient without a value. Once the other “parameter” is given, its “t_value” should also be provided; the value parameters should be input as a string with double quotes.

Argument	Description
s_name	Name of the measure; required

Appendix F: OCEAN API Functions for the HSPICE Measure Utility

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Argument	Description
parameter	The parameter of the Find/When/At measure in the HSPICE Measure Utility. <ol style="list-style-type: none">1. <i>goal</i> Goal in the measure; optional2. <i>minval</i> Minimum value in the measure; optional3. <i>weight</i> Weight in the measure; optional4. <i>analysis</i> Analysis in the measure. Valid values: "TRAN," "DC," "AC"
parameter	<ol style="list-style-type: none">5. <i>precision</i> MEASDGT option. Valid values are [1, ... , 10]6. <i>func</i> Function name that specifies the general measure function Valid values are "RMS", "MIN", "AVG", "MAX", "INTEG", "PP", "DERIV" and "EM_AVG"7. <i>trig</i> and <i>targ</i> Trigger and Target keywords indicate the following parameters (except for the above) are all for trig or targ measures. Only 'trig and 'targ are valid; ?trig and ?targ are not supported8. <i>var</i> Signal or variable9. <i>type</i> Signal or variable type. Valid values "V", "I", "P", "VP", "VM", "VR", "VI", "IM", "IP", "IR", "II", "VDB" and "IDB"10. <i>val</i> The value of a signal or variable11. <i>event</i> The measure event. Valid values are "RISE", "FALL" and "CROSS"12. <i>eventValue</i> The value corresponding to the "RISE", "FALL" and "CROSS" measure events. Integer value or "last"13. <i>td</i> Time delay14. <i>from</i> Start value of signal or variable15. <i>to</i> End value of signal or variable16. <i>at</i> Used only in special cases
t_value	The value of the parameter

Appendix F: OCEAN API Functions for the HSPICE Measure Utility

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Value Returned:

t Parameter setting succeeded with distribution function.
nil One or more errors; note details in the printed report.

Example1:

Sets up the 'AVG measure. The analysis is transient. The measure name is "val2". The measurement is taken when V(/net6) is from 0 to 50m:

```
snpsMeasureFunc('val2 ?analysis "TRAN" ?func "AVG" ?var "/"  
net6" ?from "0" ?to "50m" )
```

Example 2:

Sets up the 'deriv measure. The analysis is transient. The measure name is "val2". The derivation takes place for V(/net6) at 50m.

```
snpsMeasureFunc('val2 ?analysis "TRAN" ?func "deriv" ?var "/"  
net6" ?type "V" 'deriv ?at "50m" )
```

Example 3:

Sets up the 'deriv measure. The analysis is transient. The measure name is "val2". The derivation takes place for V(/net6) when V(/net5) = 2 under and FALL = last.

```
snpsMeasureFunc('val2 ?analysis "TRAN" ?func "deriv" ?var "/"  
net6" ?type "V" 'deriv ?type "V" ?var "/net5" ?event "FALL"  
?eventValue "yes" ?td "3" ?val "2" )
```

Example 4:

Sets up the 'AVG power measurement. The analysis is transient. The measure name is "val2".

```
snpsMeasureFunc('val2 ?analysis "TRAN" ?func "AVG" ?var  
"power" )
```

3. **snpsMeasureFind** function:

Syntax:

```
snpsMeasureFind('s_name ?parameter t_value [?parameter
t_value ...])
```

Description: Sets up information for a Find/When/At combination measure.

Note: The “parameter” and “t_value” should be input together except when the parameter is “trig” or “targ”. For “targ”, *only* 'targ is sufficient without a value. For “trig”, *only* 'trig is sufficient without a value. Once the other “parameter” is given, its “t_value” should also be provided; the value parameters should be input as a string with double quotes.

Argument	Description
s_name	Name of the measure; required
parameter	The parameter of the Find/When/At measure in HSPICE Measure Utility. <ol style="list-style-type: none"> 1. <i>goal</i> Goal in the measure; optional 2. <i>minval</i> Minimum value in the measure; optional 3. <i>weight</i> Weight in the measure; optional 4. <i>analysis</i> Analysis in the measure. Valid values: "TRAN," "DC," AC" 5. <i>precision</i> MEASDGT option. Valid values are [1, ... , 10] 6. <i>find</i> Keyword that indicates the following parameters (except for the above) are all for find/when/at combination measure. Only 'find is valid; ?find is not supported. 7. <i>var</i> Signal or variable; required in When and Find/When combination. 8. <i>var1</i> The 2nd signal or variable, required in Find/At and Find/When combination. 9. <i>type</i> Signal or variable type. Valid values "V", "I","P", "VP", "VM", "VR", "VI", "IM", "IP", "IR", "II", "VDB" and "IDB" 10. <i>type1</i> Signal1 or variable1 type. Valid values are the same as <i>type</i>, see parameter 9.

Appendix F: OCEAN API Functions for the HSPICE Measure Utility

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Argument	Description
11. <i>val</i>	The value of a signal or variable
12. <i>event</i>	The measure event. Valid values are "RISE", "FALL" and "CROSS"
13. <i>eventValue</i>	The value corresponding to the "RISE", "FALL" and "CROSS" measure events. Integer value or "last"
14. <i>td</i>	Time delay
15. <i>at</i>	Used only in special cases
t_value	The value of the parameter

Value Returned:

t The parameter setting succeeded with distribution.
 nil One or more errors; note details the printed report.

Examples

Example 1:

Sets up the 'When measurement. The analysis is transient. The measure name is "time2". The measurement is taken when $V(/net6) = 0.98$

```
snpsMeasureFind('time2 ?analysis "TRAN" 'find ?type "V" ?var
"/net6" ?val "0.98" )
```

Example 2:

Sets up the 'Find/At measure. The analysis is transient. The measure name is "time2". The measure finds $V(/net6)$ at time of $1e-2$.

```
snpsMeasureFind('time2 ?analysis "TRAN" 'find
?type1 "V" ?var1 "/net6" ?at "1e-1" )
```

Example 3:

Sets up the 'Find/At measure. The analysis is transient. The measure name is "time2". The measure finds $V(/net5)$ when $V(/net6) = "0.98"$.

```
snpsMeasureFind('time2 ?analysis "TRAN" 'find ?type1 "V"
?var1 "/net5" ?type "V" ?var "0.98" )
```

Appendix F: OCEAN API Functions for the HSPICE Measure Utility

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Example 4:

This setup will report the error that ?var, ?var1 and ?at can not appear in the same measure statement.

```
snpsMeasureFind('time2 ?analysis "TRAN" 'find ?type1 "V"  
?var1 "/net5" ?type "V" ?var "0.98" ?at "1")
```

4. snpsmeasEqua function:

Syntax

```
snpsMeasureEqua('s_name ?parameter t_value [?parameter  
t_value ...])
```

Description: Sets up information for an equation (expression) measure.

Note: The “parameter” and “t_value” should be input together except when the parameter is “trig” or “targ”. For “targ”, *only* 'targ is sufficient without a value. For “trig”, *only* 'trig is sufficient without a value. Once the other “parameter” is given, its “t_value” should also be provided; the value parameters should be input as a string with double quotes.

Argument	Description
s_name	Name of the measure; required
parameter	The parameter of the equation measurement in the HSPICE Measure Utility. 1. <i>goal</i> Goal in the measure; optional 2. <i>minval</i> Minimum value in the measure; optional 3. <i>weight</i> Weight in the measure; optional 4. <i>analysis</i> Analysis in the measure. Valid values: "TRAN," "DC," AC" 5. <i>precision</i> MEASDGT option. Valid values are [1, ... , 10] 6. <i>param</i> Keyword preceding equation in expression form
t_value	The value of the parameter

Value Returned:

t The parameter setting succeeded with distribution.
nil One or more errors; note details the printed report.

Example: Sets ip the equation measure. The analysis is transient. The measure name is "val922". The equation is "val2+30".

```
snpsMeasureEqua('val922 ?analysis "TRAN" ?param "val2+30" )
```

5. snpsMeasureError function:

Syntax:

```
snpsMeasureError's_name ?parameter t_value [?parameter  
t_value ...])
```

Description: Sets up the information for an error measurement.

Note: The “parameter” and “t_value” should be input together except when the parameter is “trig” or “targ”. For “targ”, *only* 'targ is sufficient without a value. For “trig”, *only* 'trig is sufficient without a value. Once the other “parameter” is given, its “t_value” should also be provided; the value parameters should be input as a string with double quotes.

Argument	Description
s_name	Name of the measure; required
parameter	The parameter of the Find/When/At measure in HSPICE Measure Utility. <ol style="list-style-type: none"> 1. <i>goal</i> Goal in the measure; optional 2. <i>minval</i> Minimum value in the measure; optional 3. <i>weight</i> Weight in the measure; optional 4. <i>analysis</i> Analysis in the measure. Valid values: "TRAN," "DC," AC" 5. <i>precision</i> MEASDGT option. Valid values are [1, ... , 10] 6. <i>error</i> Error function name that species the error measure function. Valid values are "ERR", "ERR1", "ERR2" and "ERR3" 7. <i>error_type</i> Error calculation type. Valid values "IGNORE", "YMIN" and "YMAX"

Appendix F: OCEAN API Functions for the HSPICE Measure Utility

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Argument	Description
8. <i>value</i>	The value of <i>error_type</i> above.
9. <i>type</i>	Signal or variable type. Valid values "V", "I", "P", "VP", "VM", "VR", "VI", "IM", "IP", "IR", "II", "VDB" and "IDB"
10. <i>mea_var</i>	Name of any output variable or parameter
11. <i>cal_var</i>	Name of any simulated output variable or parameter
12. <i>from</i>	Start value of signal or variable
13. <i>to</i>	End value of signal or variable
t_value	The value of the parameter

Value Returned:

t The parameter setting succeeded.
nil One or more errors; note details the printed report.

Example: Sets up the Error measurement. The analysis is transient. The measure name is "e1". The error function is "ERR". The measured variable is "m1" and the simulated variable is "c1". The error calculation method is "IGNORE" and from 1 to 2. The goal is 1; minval is 2 and weight is 3.

```
snpsMeasureError('e1 ?analysis "TRAN" ?error "ERR"
+ ?mea_var "m1" ?cal_var "c1" ?from "1" ?to "2" ?goal "1"
?minval "2"
+ ?error-type "IGNORE" ?value "1e-15" )
```

6. **snpsMeasureSpec** function:

Syntax:

```
snpsMeasureSpec('s_name ?parameter t_value [?parameter
t_value ...])
```

Description:

Sets up Special measure information.

Appendix F: OCEAN API Functions for the HSPICE Measure Utility

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Note: The "parameter" and "t_value" should be input together. Once the "parameter" is given, its "t_value" should also be provided; the value parameters should be input as strings with double quotes.

Argument	Description
s_name	Name of the measure; required
parameter	<p>The parameter of the Special measure in HSPICE Measure Utility. NOTE: The "parameter" and "t_value" should be input together. Once the "parameter" is given, its "t_value" should also be provided; the value parameters should be input as strings with double quotes.</p> <ol style="list-style-type: none"> 1. <i>goal</i> Goal in the measure; optional 2. <i>minval</i> Minimum value in the measure; optional 3. <i>weight</i> Weight in the measure; optional 4. <i>analysis</i> Analysis in the measure. Valid values: "FFT" and "PHASENOISE" 5. <i>precision</i> MEASDGT option. Valid values are [1, ... , 10] 6. <i>spec</i> The keyword for special measure. Required for FFT analysis, Valid values are "THD", "SNR", "SNDR", "ENOB" and "SFDR". For PHASENOISE analysis, Valid values are "PERJITTER", "CTCJITTER", "RMSJITTER", "PHJITTER", "TRJITTER", "LTJITTER", "AM" and "PM" 7. <i>from</i> Start point of the range, optional. Only used with PHASENOISE analysis and when the spec function is any of "PERJITTER", "CTCJITTER", "AM" and "PM" to End point of the range, optional. Only used in PHASENOISE analysis and when spec function is either "PERJITTER", "CTCJITTER", "AM", or "PM". 8. <i>units</i> Unit specification in the PHASENOISE jitter measure, optional. Valid values are "sec", "rad", and "UI". Used for all functions except "AM" and "PM". 9. <i>ber</i> Bit error rate in the PHASENOISE measure, optional. Valid for all functions except "AM" and "PM". 10. <i>nbharm</i> Harmonic up to which to carry out the measure under FFT. Optional. Valid for functions "THD", "SNR", "SNDR", and "ENOB".

Appendix F: OCEAN API Functions for the HSPICE Measure Utility

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Argument	Description
	11. <i>binsiz</i> Index for the noise calculation; optional. Refer to <code>.MEASURE FFT</code> in the <i>HSPICE Reference Manual: Commands and Control Options</i> for more information. Valid for functions "SNR", "SNDR", and "ENOB"
	12. <i>maxfreq</i> Upper limit of frequency to carry out the measure; optional. Valid for functions "SFDR", "SNDR" and "ENOB".
	13. <i>minfreq</i> Lower frequency limit to carry out the measure. Optional. Valid for functions "SFDR", "SNDR", and "ENOB".
<code>t_value</code>	The value of the parameter

Value Returned:

`t` The parameter setting succeeded.
`nil` One or more errors; note details the printed report.

Example: Sets up the special measure. The analysis is phasenoise. The measure name is "p5". The range is from "1k" to "2M".

```
snpsMeasureSpec('p5 ?analysis "PHASENOISE" ?spec "PERJITTER"  
?from "1k" ?to "2M")
```

OCEAN API Functions for HSPICE MOSRA Analysis

Describes HSPICE-defined OCEAN functions provided to run the HSPICE MOSRA model reliability analysis.

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

The **Save OCEAN** script option under the File menu of the HSPICE Reliability Analysis form saves all operations performed in the HSPICE Reliability Analysis window as an OCEAN script file to be executed under the non-graphic mode. You can use these OCEAN API functions or the Virtuoso® general OCEAN functions to write the OCEAN script directly to simulate in batch mode.

When you combine these HSPICE Reliability Analysis OCEAN functions with the native Cadence™ OCEAN functions, you can run a Reliability Analysis in batch mode without the GUI.

The following are the HSPICE-defined OCEAN functions provided to run Reliability analyses.

See [Chapter 13, HSPICE Reliability Analysis \(MOSRA\)](#) for a description of the graphics mode of the HSPICE Integration to ADE.

The following are the HSPICE-defined OCEAN functions provided to run the Measure utility.

1. **snpsMosra** function:

Appendix G: OCEAN API Functions for HSPICE MOSRA Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Syntax:

```
snpsMosra(t_simMode t_relTotalTime
          [?mosraArgument1 t_mosraArgumentValue1] ...
          [?mosraArgumentN t_mosraArgumentValueN]
          ?temperature temperature ?designVariables
          l_designVariables) => t/nil
```

Description: Adds a MOSRA command to the HSPICE Reliability Analysis environment.

Argument	Description
t_simMode	0: Select pre-stress simulation only 1: Select post-stress simulation only 2: Select both pre- and post-stress simulation HSPICE reads in the *.radeg0 file for post-stress simulation (only) and uses it to update the input for reliability analysis; the transient output is in a *.tr1 waveform file. The *.radeg file and input netlist must be in the same directory. When SimMode =1, the netlist stimuli could be different from the SimMode=0 netlist that generated the *.radeg file. Required argument.
?mosraArgument1	MOSRA command argument. See the <i>HSPICE User Guide: Simulation and Analysis</i> for MOSRA arguments you can use.
t_mosraArgumentValue1	Value for the MOSRA command argument.
?mosraArgumentN	Any subsequent MOSRA command argument. See the <i>HSPICE User Guide: Simulation and Analysis</i> for MOSRA arguments you can use.
t_mosraArgumentValueN	Value for the MOSRA command argument.
t_relTotalTime	Final reliability test time to use in post-stress simulation phase. Required argument (input with a floating-point string).
t_temperature	Temperature, input as a floating-point string.

Argument	Description
<code>l_designVariables</code>	The design variables name and value pair list. For example: <code>'(("vdd" "0.1")("vcc" "0.2"))</code>

Value Returned:

- `t` The MOSRA Command addition succeeded.
- `nil` One or more errors; note details in the printed report.

Example

The two commands below add two MOSRA commands; the first MOSRA command simulates as alter 0, the second MOSRA command simulates as alter 1.

```
snpsMosra("0" "315360000" ?relStepTime "31536000"
+ ?degradationtime "31536000")=>t
snpsMosra("1" "315360000" ?relStepTime "31536000"
+ ?degradationtime "31536000" ?temperature "100"
+ ?designVariables '(("vdd" "1.8")))=>t
```

2. **snpsMosraModel** function:

Syntax:

```
snpsMosraModel(t_modelName ?hciParams l_hciParams
+ ?nbtiParam l_nbtiParam)=> t/nil
```

Description: Adds an HSPICE default MOSRA Model command.

Argument	Description
<code>t_modelName</code>	User-defined MOSFET reliability model name.
<code>l_hciParams</code>	Reliability model parameter for HCI. See the <i>HSPICE User Guide: Simulation and Analysis</i> for sections on HCI parameters.
<code>l_nbtiParam</code>	Reliability model parameter for NBTI. See the <i>HSPICE User Guide: Simulation and Analysis</i> for sections on NBTI parameters.

Value Returned:

- `t` Addition of the MOSRA default model succeeded.
- `nil` One or more errors; note details in the printed report.

Adding "pch_ra" MOSRA Model into the HSPICE Reliability Analysis simulation run:

```
snpsMosraModel("pch_ra" ?nbtiParam " ("tit0" "1e-5")
+ ("titce" "0") ("titfd" "0") ("titttd" "0") ("tn" "0.25")
+ ("tot0" "0") ("totdd" "0") ("totfd" "0") ("totttd" "0")
+ ("tk" "0.5")))
```

3. **snpsCustomMosraModel** function:

Syntax:

```
snpsCustomMosraModel(t_shareLibPath t_modelFile) => t/nil
```

Description: Adds Custom MOSRA shared library path and Model File into HSPICE Reliability Analysis simulation environment.

Argument	Description
t_shareLibPath	Name of the measure; required
t_modelFile	Custom MOSRA Model File to be included in the HSPICE Reliability Analysis simulation.

Note: If the two arguments above use a relative path, it is recognized as the relative path to the working directory.

Value Returned:

t The Custom MOSRA setup succeeded.

nil One or more errors; note details in the printed report.

Example

Sets up Custom MOSRA share library path and model file. The share library and Model file both use "relative path" to user's working directory.

```
snpsCustomMosraModel("./mosra_model_lib" "./
mosra_model.card"
```

4. **snpsMosraAppendModel** function

Syntax

```
snpsMosraAppendmodel(s_mosraModelType
t_modelName t_genModelName s_modelKeyword) => t/nil
```

Description: This command appends the parameter values from the source model card (SrcModel) to the destination model card (DestModel). All arguments are required.

Argument	Description
s_mosraModelType	MOSRA model type: either HSPICE default MOSRA Model (hspapdmdls) or custom MOSRA Model (cusapdmdls).
t_modelName	Source model name, e.g., the name of the MOSRA model.
t_genModelName	Destination model name, e.g., the original model in the model library.
s_modelKeyword	Model type for DestModel. For example, 'nmos, 'pmos.

Value Returned:

t The APPENDMODEL succeeded

nil One or more errors; note details in the printed report.

Example:

```
snpsMosraAppendmodel('cusapdmdls "nl_ra" "nch" 'NMOS)
=>t
```

5. **snpsMosraRun** function:

Syntax:

```
snpsMosraRun(s_mosraModelType @rest run_args)
```

Description: Runs reliability analysis using HSPICE as a wrap function of OCEAN function "run()", which does the extra job of invoking the reliability analysis netlisting and post-process after simulation.

Argument	Description
s_mosraModelType	MOSRA model type, either HSPICE default MOSRA Model ('hspmdl) or custom MOSRA Model ('cusmdl) to run simulation.
run_args	The same arguments as the OCEAN run().

Value Returned:

The same as for run().

Appendix G: OCEAN API Functions for HSPICE MOSRA Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Example

Runs all settings for a MOSRA analysis simulation using a Custom MOSRA Model setup.

```
snpsMosraRun('cusmdl')
```

6. **snpsMosraOutputsPlot** function:

Syntax:

```
snpsMosraOutputsPlot(t_output ?type s_type
+ ?wavFile t_wavFile ?degradationPeriod
+ t_degradationPeriod ?plotMode t_plotMode)=> t/nil
```

Description: Plots the selected waveform file and degradation time-specific output.

Argument	Description
t_output	Output signal or expression.
s_type	The output signal type ('net' or 'terminal'); if it is an expression, it is 'expr'.
t_wavFile	Outputs psf file; combines ".tr" file string with alter index and simMode. For example: ".tr0" means alter 0, pre-stress simulation psf output file plot; ".tr1@ra" means alter1, post-stress simulation psf output file plot.
t_degradationPeriod	Post-stress simulation degradation Time. Only used for post-stress output file plot. To plot all degradation time waveform, use "All".
t_plotMode	Specify the window mode of plotting window; valid values are: Append (default) — Add new waveform to the existing waveform window Replace — Replace existing window's waveform New Subwin — New a SubWindow in the existing waveform window and append new created waveform New Win— New waveform Window in the existing waveform window and append new created waveform

Value Returned:

- `t` The waveform file output plot succeeded.
- `nil` One or more errors; note details in the printed report.

Examples

Example 1: Plots the net `"/i1"` signal waveform of alter 0 pre-stress simulation output.

```
snpsMosraOutputsPlot("/i1" ?type 'net ?waveFile ".tr0")
```

Example 2: Plot the net `"/i1"` signal, degradation time `"1e+07"` waveform of alter 1 pre-stress simulation output.

```
snpsMosraOutputsPlot("/i1" ?type 'net ?waveFile ".tr1@ra"
+ "1e+07")
```

Example 3: Plot the net `"/i1"` signal, all degradation time waveforms of alter 1 pre-stress simulation output.

```
snpsMosraOutputsPlot("/i1" ?type 'net ?waveFile ".tr1@ra"
+ "All")
```

7. `snpsMosraMeasPlot` function:

Syntax:

```
snpsMosraMeasPlot(t_measResultsFile
  t_measVariable ?plotMode t_plotMode) =>t/nil
```

Description: Plots the selected waveform file and degradation time-specific output.

Argument	Description
<code>t_measResultsFile</code>	Measure results file that combines ".mt" string with alter index and simMode. For example: ".mt0" means alter 0, pre-stress simulation measure results file plot; ".mt1@ra" means alter1, post-stress simulation measure result file plot.
<code>t_measVariable</code>	The measurement variable name.

Appendix G: OCEAN API Functions for HSPICE MOSRA Analysis

Using the HSPICE-Provided OCEAN API for HSPICE Measure Utility

Argument	Description
t_plotMode	Specifies the window mode of plotting window; valid values are: Append (default) — Add new waveform to the existing waveform window Replace — Replace existing window's waveform New Subwin — New a SubWindow in the existing waveform window and append new created waveform New Win— New waveform Window in the existing waveform window and append new created waveform

Value Returned:

t The measurement variable's results waveform plotting succeeded.

nil One or more errors; note details in the printed report.

Example

Plot measure variable "pids_vgs" results of alter 0 post-stress simulation measure results file.

```
snpsMosraMeasPlot(".mt0@ra" "pids_vgs" "New Win")
```

8. **snpsMosraMeasPrint** function:

Syntax:

```
snpsMosraMeasPrint(t_measResultsFile) => t/nil
```

Description: Prints out the measurement variables results information in table format.

Argument	Description
t_measResultsFile	Measure results file that combines ".mt" string with alter index and simMode. For example: ".mt0" means alter 0, pre-stress simulation measure results file plot; ".mt1@ra" means alter1, post-stress simulation measure result file plot.

Value Returned:

t The measurement variables results printing succeeded.

`nil` One or more errors; note details in the printed report.

Example

Prints measure variables results of alter 1 post-stress simulation measure results output.

```
snpsMosraMeasPrint(".mt1@ra")
```

9. **snpsMosraRadegPlot** function:

```
snpsMosraRadegPlot(t_radegFile t_mosElemName  
+ t_mosElemParamName  
+ ?plotMode t_plotMode) =>t/nil
```

Description: Plots the degradation parameter values of the specific radeg file with degradation time. The waveform's X-axis is the degradation time.

Argument	Description
t_radegFile	Combined ".radeg" string with alter index. For example: "input.radeg0" means alter 0, radeg file plot; ".radeg1" means alter1, radeg file plot.
t_mosElemName	The plotted MOS element name in the <i>radeg</i> file.
t_mosElemParamName	The plotted MOS element parameter name.
t_plotMode	Specify the window mode of plotting window; valid values are: Append (default) — Add new waveform to the existing waveform window Replace — Replace existing window's waveform New Subwin — New a SubWindow in the existing waveform window and append new created waveform New Win— New waveform Window in the existing waveform window and append new created waveform

Value Returned:

`t` The radeg file MOS element parameter plotting succeeded.

`nil` One or more errors; note details in the printed report.

Example

Plot MOS element "m2" parameter "delvth0" of alter 1 radeg file.

```
snpsMosraRadegPlot("input.radeg1" "m2" "delvth0" "New Win")
```

10. **snpsMosraRadegPrint** function:

```
snpsMosraRadegPrint(t_radegFile) => t/nil
```

Description: Prints out the radeg file information in table format.

Argument	Description
t_radegFile	Combined ".radeg" string with alter index. For example: "input.radeg0" means alter 0, radeg file plot; ".radeg1" means alter1, radeg file.

Value Returned:

t The printing of the radeg file information succeeded.

nil One or more errors; note details in the printed report.

Example

Prints the radeg file "input.radeg2" after an alter 2 simulation.

```
snpsMosraRadegPrint("input.radeg2")
```


A

- AC analysis 119
- ACMatch analysis 131
- AGAUSS keyword 202
- Analog Options forms 165
- analogLib 78
- analyses
 - AC 119
 - ACMatch 131
 - DC 117
 - DCMatch 129
 - FFT 122, 124
 - Linear Network Parameter 127
 - Loop Stability 132
 - Noise 122
 - Operating Point (OP) 121
 - Pole/Zero 146
 - Transient 110
 - transient noise 113
- analyses forms setup, tutorial 24, 26
- analysis
 - Linear Network Parameter 124
- annotation, Results menu 189
- AUNIF keyword 202
- automatic output log 103

B

- .BIASCHK analysis 303

C

- calculator 153
- cdsenv file 94
- cdsinit SKILL file 94
- CIW window 93
- clock source, random jitter 92
- command line option 103
- command line option control 104
- component reference library 78

- components, added/converted in HSPICE
 - integration 80
- console 93
- console, environment 93
- convergence 169
 - problems, operating point Debug mode 122
- conversion messages 79
- conversion script 75
- corner analysis 213
- corner analysis distributed jobs 328
- customerHook function 162

D

- data outputs 151
- DC analysis 117
- DCMatch analysis 129
- DEBUG keyword 122
- default simulator, HSPICE 92
- definition files 102
- demo directory 17
- demostate, tutorial 40
- design variable 148
- design, configuring, tutorial 23
- DFT calculation 122, 124
- Direct Plot, Results menu 185
- distributed jobs
 - corner analysis 328
 - environment setup 321
 - Monte Carlo 323
- DM 321, 323, 328
- downloading tutorial demo directory 17

E

- editing design variables 148
- editing session options 97
- environment console 93, 94
- environment console setup 93
- Environment Options
 - command line options 104

Index

F

- multithreading 105
- output log files 104, 168
- print netlist comments 103
- stop view list 103
- subsckt, top level 105
- switch view list 103
- environment options
 - form 103
 - setup, tutorial 24
- error-warning file 104, 168

F

- FFT analysis 122, 124
- form sections commonly shared across functions in Plotting Assistant 178

G

- GAUSS
 - keyword 202

H

- HSPICE
 - components added/converted 80
 - What's New window 97
- HSPICE 64-bit simulation 104
- HSPICE components ,added or converted 80
- HSPICE integration
 - Verilog-A 106
- HSPICE measures 257
- HSPICE Plotting Assistant 174
- HSPICE Plotting Assitant, tutorial 33
- HSPICE, as default simulator 92
- HSPICE.menus file 331

I

- include path 102
- incremental netlisting, netlisting, incremental 167
- initial conditions 169
- initial conditions, setting 170
- installtion README 17

J

- jitter
 - random, with clock source 92

K

- keywords
 - DEBUG 122

L

- library
 - conversion messages 79
- library conversion 75
- library update utility 76
- licenses required 17
- LIMIT keyword 202
- LIN analyses 127
- LIN analysis 124
- .listing file *.lis 104, 168
- loading tutorial files 17
- log files 104, 168
- LSTB analysis 132

M

- Measurement Utility 257
- mismatch analyses 129, 131
- Model Library Setup form 99
- model setup, tutorial 23
- Monte Carlo analysis
 - configuring 200
 - invoking 200
 - multiprocessing 202
 - OCEAN scripting 210
 - results, measuring, plotting 205
 - session saving, loading 210
 - setting options 203
 - tutorial example 47
- Monte Carlo distributed jobs 323
- MOSFETmodels reliability 271
- MOSRA analysis 271
- multi-dimensional sweeps 110
- multi-port/two-port noise 127
- multiprocessing 202
- multithreading 105

N

- netlist and run 162
- netlist component information 167
- netlist file 161

- netlist file, tutorial 38
- netlisting script, customerHook function 162
- nets and terminals, saving/plotting 152
- node sets, selecting 170
- nodeset and initial conditions 169
- nodesets 169
- Noise analyses 122
- noise analysis, when part of a LIN analysis 128

O

- OCEAN script API
 - Monte Carlo 365
 - RF 373, 391, 407, 413, 425
- OCEAN scripts 96
- OP analysis 121
- optimization analysis 249
- options menu, HSPICE-specific controls 162
- Other options field 165
- output log files 104, 168
- output log, automatic 103
- outputs 151
- outputs menu, tutorial 32

P

- parametric analysis form 110
- PDKs, 3rd party 78
- PLL_Demo 17
- plot outputs, Results menu 184
- Plotting Assistant
 - common form section shared across functions 178
- plotting assistant form, tutorial 33
- plotting assistant usage, tutorial 40
- plotting expressions 153
- plotting preferences 196
- Pole/Zero analysis 146
- post-processing 173
- print comments (netlist) checkbox 103
- print, Results menu 188
- print/plot options 196
- printing/plotting results 173
- PSF output, 64-bit simulation 104
- PZ analysis 146

Q

- quick-start tutorial 17

R

- README, installation 17
- reliability analysis 271
- restoring states 95
- Results menu
 - Annotate 189
 - Direct Plot 185
 - Plot Outputs 184
 - Print 188
 - printing/plotting options 196
 - save, select, delete 194
- Results menu overview 173
- results, printing/plotting 173
- RF analyses
 - invoking 235
 - OCEAN script API 365, 373, 391, 407, 413, 425
 - OCEAN scripting 233, 247, 254
 - results analysis 246
 - running and post-processing 245
 - selecting and configuring 236
 - session, saving and loading 233, 247, 254
 - setting options 239
 - tutorial example 61
 - using the Outputs tab 243

S

- Save Options form 156
- save options window, tutorial 32
- saving or restoring a list of outputs 159
- saving, selecting, deleting results 194
- saving/plotting outputs 151
- Schematic Editor 93
- session menu 95
- setting outputs form 151
- setup for saving/plotting nets and terminals 152
- setup menu
 - design 98
 - Model Libraries 99
 - Simulation Files 102
 - Simulator/Directory/Host 99
- simulation files setup, tutorial 24
- simulation menu overview 161

Index

T

- simulation primitives, library updater 76
- simulation, running, tutorial 38, 40
- simulator input file 161
- software startup, tutorial 19
- spectrum analysis 124
- standard out *st0 file 104, 168
- states
 - outputs list 159
 - restoration 95
 - saving and loading 95
 - tutorial 36
- stop view list 103
- subckt, top level 105
- switch view list 103
- SX menu, adding to console 331

T

- table of outputs 153
- Tool Filter, HSPICE simulator 92
- top level, subckt 105
- transient analysis 110
- transient noise analysis 113
- tutorial 17
- tutorial demo directory 17

- two-port/multi-port noise 124

U

- UNIF keyword 202
- update script 78
- update utility, library 76
- updating HSPICE.menus file to add SX menu 331

V

- variables, design 148
- vector files 102
- vector files, tutorial 24
- Verilog-A 106
- Verilog-A support 106
- Verilog-A,example, tutorial 44
- version 5.1 93
- version 6.1 93
- violation check 303

W

- warning-error file 104, 168