cadence®

Virtuoso ADE Assembler User Guide

Product Version IC23.1
November 2023

© 2023 Cadence Design Systems, Inc.

Portions © Regents of the University of California, Sun Microsystems, Inc., Scriptics Corporation. Used by permission.
Printed in the United States of America.

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Open SystemC, Open SystemC Initiative, OSCI, SystemC, and SystemC Initiative are trademarks or registered trademarks of Open SystemC Initiative, Inc. in the United States and other countries and are used with permission. Analog Design Environment XL contains technology licensed from, and copyrighted by: Apache Software Foundation, 1901 Munsey Drive Forest Hill, MD 21050, USA © 2000-2007, Apache Software Foundation.

Trademarks: Trademarks and service marks of Cadence Design Systems, Inc. contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 800.862.4522. All other trademarks are the property of their respective holders.

Restricted Permission: This publication is protected by copyright law and international treaties and contains trade secrets and proprietary information owned by Cadence. Unauthorized reproduction or distribution of this publication, or any portion of it, may result in civil and criminal penalties. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. Unless otherwise agreed to by Cadence in writing, this statement grants Cadence customers permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used only in accordance with a written agreement between Cadence and its customer.
2. The publication may not be modified in any way.
3. Any authorized copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement.
4. The information contained in this document cannot be used in the development of like products or software, whether for internal or external use, and shall not be used for the benefit of any other party, whether or not for consideration.

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

Cadence is committed to using respectful language in our code and communications. We are also active in the removal and replacement of inappropriate language from existing content. This product documentation may however contain material that is no longer considered appropriate but still reflects long-standing industry terminology. Such content will be addressed at a time when the related software can be updated without end-user impact.

Restricted Rights: Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor

Contents

1
Getting Started with Virtuoso ADE Assembler 33
Introducing Virtuoso® ADE Assembler 33
Licensing Requirements 36
Getting Started in the Virtuoso ADE Assembler Environment 37
Launching ADE Assembler 38
Using the Launch ADE Assembler Form 40
Creating a New Setup 41
Opening an Existing Setup 43
ADE Assembler Environment at Startup 45
ADE Assembler Environment User Interface 49
The Welcome to ADE Assembler Page 50
Menu Bar 51
Context-Sensitive Menus 59
Toolbars 66
Assistant Panes 72
Outputs 80
Workspaces 82
Specifying the Run Mode 83
Simulators Supported for Run Modes 83
Specifying Options for Saving Simulation Results 85
Specifying Results Database Location 88
How does ADE Assembler save the results in the results database? 89

2
Specifying Tests and Analyses 93
Working with Tests 94
Adding a Test 95
Renaming a Test 96
Copying a Test 97
Removing a Test 98

© 2023 Cadence Design Systems, Inc.

Adding Notes to a Test . . . 98
Changing the Order of Tests . . . 99
Viewing Information about a Test . . . 100
Adding, Changing, and Removing Analyses . . . 102
Adding an Analysis . . . 102
Changing an Analysis . . . 104
Changing the Order of Analyses . . . 105
Removing an Analysis . . . 105
Specifying a Design for a Test . . . 106
Opening a Design Schematic . . . 108
Choosing the Target Simulator . . . 109
Loading State Information . . . 112
Specifying Model Libraries . . . 115
Setting the Simulation Temperature . . . 117
Specifying Analog Stimuli . . . 119
Specifying a Custom Library of Sources . . . 121
Specifying Simulation Files . . . 122
Specifying Simulation Environment Options . . . 128
Environment Options for the Spectre Circuit Simulator . . . 129
Environment Options for the UltraSim Circuit Simulator . . . 130
Environment Options for the AMS Designer Simulator . . . 131
Environment Options for the hspiceD Circuit Simulator . . . 132
Common Environment Options for Simulators . . . 132
Specifying Simulator Options . . . 136
Specifying Simulator Options for Spectre and UltraSimCircuit Simulators . . . 136
Specifying Options for AMS Circuit Simulator . . . 137
Specifying Options for hspiceD Circuit Simulator . . . 138
Creating and Viewing the Netlist . . . 139
Editing Multiple Tests in ADE Assembler . . . 140
Multi-Test Editor User Interface . . . 141
Comparing and Modifying Test Setup . . . 143
Copying Values from Reference Test to the Other Tests . . . 145
Keeping the Active Setup and Multi-Test Editor in Sync . . . 148
Modifying a Test in ADE Explorer . . . 150
Working with OCEAN-Based Tests . . . 154

© 2023 Cadence Design Systems, Inc.

3
Working with Design Variables and Instance Parameters . . 155
Working with Design Variables 156
Adding a Design Variable 157
Changing the Value of a Design Variable 160
Using Expression Builder for Design Variables and Global Variables 165
Hiding Overridden Design Variables for a Test 167
Deleting a Design Variable 167
Changing the Order of Design Variables 169
Enabling or Disabling Design Variables 170
Copying Variable Values to Other Tests 171
Saving Design Variables 174
Toggling the View on the Variables tab of the Variables and Parameters Assistant Pane 174
Finding Design Variables in Schematic 175
Copying Design Variable Values to the Schematic 177
Copying Design Variable Values from the Schematic 178
Importing Design Variables from an ADE State 179
Displaying Design Variables on the Schematic 179
Defining Variables in a File 180
Adding Variable Tags 182
Assigning Tags to Corners 187
Filtering Variables and Corners 190
Working with Device Instance Parameters 194
Selecting Instance Properties to be Parameterized for Simulations 196
Changing the Value of a Device Instance Parameter for Simulation 202
Disabling a Changed Device Instance Parameter Value for a Simulation 204
Deleting a Parameter 204
Sorting Parameters by Properties and Objects 205
Filtering Device Instance Parameters 206
Creating Custom Device Filters 208
Changing the Order of Device Parameters 210
Creating Parameter Ranges 212
Working with Parametric Sets 216
Creating Parametric Sets in ADE Assembler 216

© 2023 Cadence Design Systems, Inc.

Using Parametric Sets Created in ADE Explorer . . . 218
Creating Parametric Sets . . . 219
Adding a Variable to a Parametric Set . . . 223
Including a Dependent Variables in a Parametric Set . . . 223
Removing a Variable from a Parametric Set . . . 225
Ungrouping Parametric Sets . . . 225
Enabling or Disabling Parametric Sets . . . 225
Enabling and Disabling Parameters . . . 226
Adding Notes for Parameters . . . 227
Adding or Changing a Variable or Parameter Specification . . . 228
Specifying an Inclusion List of Values . . . 229
Specifying an Exclusion List of Values . . . 230
Specifying a Range of Values . . . 232
Specifying Center and Span . . . 234
Specifying Center and Span as a Percentage . . . 236
Deleting a Parameter Specification . . . 238
Creating Matched Device Parameters . . . 238
Creating Ratio-Matched Device Parameters . . . 240
Disabling Callbacks on Swept Device Parameters . . . 244
Scope of Parameters . . . 244
Inheriting from the Same Instance: iPar() . . . 244
Passed Parameter Value of One Level Higher: pPar() . . . 245
Passed Parameters from Any Higher Level: atPar() . . . 246
Inheriting from the Instance Being Netlisted: dotPar() . . . 246
Table of Functions . . . 247

4
Working with Global Variables . . . 249

Creating a Global Variable . . . 250
Loading the Default Set of Global Variables . . . 252
Parameterizing Cellviews Defined in the Config View . . . 254
Creating Config Sweep Variables . . . 254
Importing Config Sweeps from the Parasitic Setup . . . 256
Using Config Sweep Variables in the Corners Setup Form . . . 257
Viewing Config Sweep Variables in Run Preview . . . 259

© 2023 Cadence Design Systems, Inc.

Viewing Config Sweeps in the Simulation Results . . . 260
Importing Sweep Variables as Global Variables . . . 261
Saving Parameters and Global Variables to a Saved State . . . 261
Importing Global Variables from a Saved State . . . 262
Exporting Global Variables to a CSV File . . . 266
Importing Global Variables from a CSV File . . . 267
Enabling and Disabling Global Variables for All Tests . . . 267
Disabling Global Variables for Specific Tests . . . 269
Disabling a Global Variable for a Test using the Variables and Parameters Assistant . 269
Disabling a Global Variable for a Test using the Data View Assistant . . . 272
Updating Global Variable Values with Design Variable Values . . . 274
Updating Design Variable Values with Global Variable Values . . . 274
Using calcVal for Calibration . . . 275
Using calcVal in Design Variables . . . 275
Using calcVal in Design Variables with Corners in Setup . . . 276

5
Working with Constraints . . . 279
Adding, Modifying, and Deleting Constraints . . . 280
Working with Parameters Created for Matched Parameter Constraints . . . 281
Importing Matched Parameter Constraints to ADE Assembler . . . 281
Resolving Mismatch Between Matched Parameter Constraints and their Corresponding Parameters . . . 283

6
Working with Corners . . . 285
The Corners Setup Form . . . 286
Creating Corners . . . 288
Disabling or Enabling Corners . . . 290
Filtering Corners . . . 292
Saving Filters . . . 299
Deleting Saved Filters . . . 299
Running Simulation with Corners . . . 300
Plotting Across Corners . . . 302

© 2023 Cadence Design Systems, Inc.

7
Working with Model Files and CDF 303
Associating a Model or Subcircuit Name with an Instance 304
Editing Component CDF 305
Adding a Model Name Parameter to a Component's CDF 307
Making the Model Name Parameter Editable 308
Using Component CDF to Specify Simulation Information 309
Creating a Stopping Cellview 310
Varying the Model File and Section during Simulation 311
Editing a Model File 314
Disabling a Model File 314

8
Working with DSPF Files 315
Including DSPF Files in the Setup 316
Specifying DSPF Files in the Simulation Files Setup Form 316
Sweeping DSPF Files in the Corners Setup Form 321
Sweeping DSPF Files in the Simulation Files Setup Form Using the VAR Function 324
Sweeping DSPF Files in the Corners Setup Form Using the VAR Function 326
Binding a DSPF View to a Design 330
Rules for Adding Prefixes to Instance Names 332
Customizing Netlist Creation Using the Options in .simrc 338
Customizing Netlist Creation Using Environment Variable 344

9
Netlisting 345
Creating a Netlist 346
Displaying a Netlist 346
Expanding Hierarchy to Netlist a Design 347

10
Selecting Data to Save and Plot 351
Opening the Outputs Setup Tab 353

© 2023 Cadence Design Systems, Inc.

Selecting Output Signals on the Schematic . . . 355
Specifying the Type of Data for Signal Outputs . . . 357
Selecting Outputs to Save . . . 360
Selecting Outputs to Plot . . . 360
Selecting Nodes, Nets, and Terminals . . . 361
Selecting a Digital Bus or a Bundle of Buses . . . 363
Specifying Whether a Result Will Be Saved or Plotted . . . 366
Adding or Modifying an Output Expression . . . 367
Creating calcVal Expressions Using calcVal Wizard . . . 372
calcVal Wizard Form . . . 378
Creating Dependent Expressions . . . 381
Creating a Combinatorial Expression . . . 385
Creating Expressions to be Measured Across Dimensions . . . 389
How is expression evaluation different when measured across dimensions? . . . 389
Measurement Across Corners . . . 390
Measurement Across Sweeps or Measurement Across All . . . 399
Creating Expressions to be Measured Across Any Dimension . . . 403
Viewing Results of Outputs Measured Across Any Dimension . . . 406
Viewing and Editing Output Signals or Expressions in Calculator . . . 408
Using MATLAB Expressions in Outputs . . . 412
Adding Outputs of Type MATLAB Expression in ADE Assembler . . . 413
Using an Interactive MATLAB Session Opened from ADE Assembler . . . 414
Evaluating MATLAB Expressions Across All Dimensions . . . 418
Evaluating MATLAB Expressions for a Data Point . . . 419
Examples of MATLAB Expressions . . . 421
Debugging MATLAB Expressions in the MATLAB Window . . . 427
Loading an OCEAN or a MATLAB Measurement . . . 430
Editing an OCEAN or a MATLAB Script File . . . 431
Writing a MATLAB Script . . . 431
Saving Operating Point Parameters . . . 434
Viewing Tabular Report for Operating Point Parameters . . . 438
Copying Outputs . . . 442
Copying Outputs Within and Across Tests . . . 442
Copying the Contents of an Output . . . 443
Adding User-Defined Columns in the Outputs Setup Tab . . . 444
Adding a User-Defined Column . . . 444

© 2023 Cadence Design Systems, Inc.

Renaming a User-Defined Column . . . 446
Deleting a User-Defined Column . . . 447
Hiding a User-Defined Column . . . 447
Exporting Outputs to a CSV File . . . 448
Importing Outputs from a CSV File . . . 450
Configuring How Outputs Appear on the Outputs Setup Tab . . . 452
Changing the Order of Outputs . . . 452
Sorting the Outputs . . . 453
Hiding and Showing Outputs . . . 453
Hiding and Showing Output Details . . . 454
Changing the Order of Columns . . . 454
Hiding and Showing Columns . . . 455
Filtering and Searching in Outputs Setup Pane . . . 455
Selecting Outputs to Save or Plot . . . 459
Removing Outputs . . . 459
Saving All Voltages or Currents . . . 461
Save Options form for Spectre Simulations . . . 462
Keep Options form for UltraSim Simulations . . . 463
Save Options form for AMS Simulations . . . 464
Save Options form for HspiceD Simulations . . . 465

11
Monte Carlo Analysis . . . 467

Preparing Setup for Monte Carlo Analysis . . . 469
Preparing Setup in the Data View Assistant of ADE Assembler . . . 469
Setting Run Options in the Monte Carlo Form . . . 470
Running a Monte Carlo Simulation and Viewing Results . . . 475
Running Incremental Monte Carlo Analyses . . . 475
Running Multi-Technology Simulations for Monte Carlo Analysis . . . 476
Viewing Results . . . 477
Viewing Statistical Parameters for Monte Carlo Samples . . . 479
Viewing Sensitivity Results . . . 479
Debugging in Debug Environment . . . 479
Varying Design Variables with Statistical Distribution . . . 480
Post-Processing Monte Carlo Results . . . 483

© 2023 Cadence Design Systems, Inc.

Managing Monte Carlo Results in the Yield View . . . 484
Viewing Data for a Specific Confidence Interval . . . 484
Creating Statistical Corners . . . 485
Displaying Statistical Parameters of Statistical Corners . . . 493
Filtering Out Data from the Yield View . . . 494
Generating Plots, Tables, and Reports . . . 495
Saving Monte Carlo Results in CSV Format . . . 496
Introduction to Mismatch ID . . . 498
The Configuration File for Mismatch ID . . . 498
Examples of Mismatch ID Configuration . . . 499
Specifying a Mismatch ID . . . 506
The Mismatch ID Form . . . 509
Exporting a Mismatch ID Configuration to a YAML File . . . 510
Importing a Mismatch ID Configuration from a YAML File . . . 511
Advanced Monte Carlo Methods . . . 512

12
Working with the Simulation Setup . . . 513
Directory Structure of a Maestro Cellview . . . 514
Saving the Simulation Setup . . . 518
Copying a maestro Cellview . . . 519
Copying only the Simulation Setup in a maestro View . . . 520
Copying the Simulation Setup and the Results Database in a maestro Cellview . . . 521
Copying Everything Contained in a maestro Cellview . . . 521
How is the Results Database Copied for a maestro Cellview? . . . 522
How are Simulation Results Copied for a maestro Cellview? . . . 522
Deleting a maestro View . . . 523
Working with Read-Only or Editable maestro Views . . . 524
Opening maestro Views in Read-Only Mode . . . 524
Opening maestro Views in Edit mode . . . 525
Viewing the DM Status of a Cellview . . . 526
Opening maestro Views for which You Do Not Have Write Permissions . . . 526
Running Simulations from Read-Only maestro Views . . . 527
Saving Setup Changes in Read-Only maestro Views . . . 530
Importing the Simulation Setup . . . 532

© 2023 Cadence Design Systems, Inc.

Working with Setup States . . . 536
Creating or Updating a Setup State . . . 538
Loading a Setup State . . . 540
Renaming a Setup State . . . 542
Deleting a Setup State . . . 542
How the Simulation Setup is Updated When You Load Setup States . . . 544
Running a Simulation Using a Setup State . . . 547
Creating a Plan Using Setup States . . . 548

13
Performing Circuit and Device Checks . . . 555

An Introduction to Checks and Asserts . . . 556
Checks . . . 556
Asserts . . . 556
Dependency on Simulators . . . 557
Setting up Checks and Asserts Analysis . . . 558
Adding Checks and Asserts . . . 560
Adding Checks using the Checks/Asserts Assistant . . . 561
Copying Checks or Asserts . . . 569
Importing Asserts from Text Files Included in the Setup . . . 571
Defining Global Device Checks . . . 573
Viewing Netlists for Checks and Asserts . . . 577
Filtering Violations . . . 580
Running Simulations and Viewing Violations . . . 583
Viewing Itemized Violation Details . . . 588
Waiving Violations . . . 594
View Cumulative Violation Details . . . 598
View Summary Violation Details . . . 599
View Instances with Violations on Schematic . . . 600
Creating Custom Violation Filter . . . 601
Viewing Violations in ViVA XL . . . 606
View Violations in Browser . . . 608
Viewing Only Device Checks . . . 610

© 2023 Cadence Design Systems, Inc.

14
Running Fault Simulations 615

Prerequisites 617
Preparing the Setup for Fault Simulation 617
Setting up Faults 619
Specifying Global Preferences for Fault Setup 623
Injecting Faults in a Design 625
Using Fault Rules 625
Fault Rules Form 631
Creating Individual Faults 636
Creating Custom Faults 641
Using Fault Files 645
Importing Fault Rules from Other Cellviews 645
Setting Up Fault Rules for Layout-based Faults 646
Creating Fault Groups 647
Creating a Fault Group by Copying An Existing Group 648
Adding Faults to Fault Groups 649
Importing Faults from a File into a Group 651
Deleting Fault Rules and Fault Groups 652
Specifying Fault Sampling Options and Auto Stop Criteria 653
Previewing Faults in a Fault Group 658
Clearing Fault Cache 663
Modifying Fault Groups 664
Setting up Run Options for a Fault Simulation 665
Run Options for Direct Fault Analysis 666
Run Options for Transient Fault Analysis 668
Running a Fault Simulation 669
Viewing Results of Fault Simulation 670
Viewing Summary of Fault Simulation Results 673
Exploring Instance that Contain Faults 674
Customizing the Display of Faults in the Fault View 675
Merging Results from Multiple Fault Simulations 682
Checking Fault Detection Rate 687
Automatic Fault Dropping Using a Run Plan 691
Creating a Run Plan for Fault Dropping 691

© 2023 Cadence Design Systems, Inc.

Running Simulations Using Selected Faults from Results . . . 696
Debugging Results and Troubleshooting Errors . . . 697
Debugging Results . . . 698
Running Functional Safety Checks . . . 699

15
Working with the Setup Library Assistant . . . 705
Opening the Setup Library Assistant . . . 708
Setup Library Assistant Graphical User Interface . . . 709
Creating a New Cellview or Loading an Existing Cellview . . . 712
Editing the Setup . . . 714
Adding New Sweep Setups . . . 714
Adding New Corner Setups . . . 716
Adding New Simulation Setups . . . 716
Adding New Reliability Setups . . . 719
Reusing the Setup from Data View and Setup Library Assistants . . . 722
Synchronizing ADE Assembler Setup with Changes in Setup Library . . . 726
Clearing References to Setups in the Setup Library . . . 728
Creating Verification Spaces . . . 729
Managing Documents . . . 731

16
Performing Reliability Analysis . . . 733
Simulator Modes for Reliability Analysis . . . 734
Specifying the Reliability Analysis Setup . . . 734
Variables and Temperature in the Reliability Setup . . . 739
Specifying Simulator Mode for Reliability Analysis . . . 741
Specifying Aging Model Files . . . 742
Reusing the Stress File . . . 745
Reliability Options Form . . . 748
Basic . . . 749
Modeling . . . 759
Degradation . . . 771
Output . . . 777
Creating Outputs for Reliability Analysis . . . 788

© 2023 Cadence Design Systems, Inc.

Masking Devices . . . 789
Copying Reliability Setup . . . 790
Loading Reliability Settings . . . 790
Setting Up Reliability Analysis for Scenarios with Varying Corner Conditions . . . 791
Enabling or Disabling a Reliability Scenario . . . 793
Exporting a Reliability Scenario Setup . . . 795
Contents of the Exported CSV File . . . 796
Importing a Reliability Scenario Setup from the CSV File . . . 799
Running the Reliability Simulation . . . 801
Important Points to Note . . . 802
Difference Between Reliability Mode And Aging Scenarios . . . 804
Working with Reliability Data . . . 805
Viewing Simulation Results . . . 805
Viewing Aged Netlist . . . 816
Viewing Output Logs for Reliability Simulations . . . 817
Plotting Results . . . 817
Plotting Results Across Corners . . . 818
Printing or Plotting Stress Results for Gradual Aging Run . . . 819
Annotating Simulation Results to Schematic View . . . 822
Performing Reliability Analysis for Monte Carlo Run Mode . . . 824
Viewing Reliability Report . . . 827
Annotating Degradation Parameters on the Schematic . . . 830
Customizing the Display of the Reliability Report . . . 831
Creating a Data Filter to Hide Selected Columns . . . 832
Applying Filters to Columns . . . 834
Creating a Custom Reliability Data Filter . . . 838
Custom Filter Syntax . . . 842
Query Operators for Custom Filters . . . 842
Customize Reliability Data Filters Form . . . 844

17
Running Simulations . . . 845

The Simulation Flow in ADE Assembler . . . 847
Using Simulation Manager with LSCS . . . 848
The Simulation Manager Infrastructure . . . 848

© 2023 Cadence Design Systems, Inc.

Running Simulations in a Multi-User Environment for Simulation Manager 850
Running Simulations in a Single-User Environment for Simulation Manager 852
Enabling Simulation Manager 854
License Requirement 854
Managing Simulations with Simulation Manager 855
Debugging Simulation Manager 859
Debugging Points with Simulation Manager 861
Setting Up Job Policies 862
Setting Up a Job Policy for a Test 864
Setting Up the Default Job Policy 865
Specifying a Job Policy Name 867
Optimizing a Single Point Run 867
Specifying a Distribution Method 868
Specifying Max Jobs 869
Specifying a Command for DRMS 870
Specifying an LBS Job Policy 872
Specifying Job Timeouts for ICRP 877
Specifying Error Reporting Options 879
Saving or Deleting a Job Policy 879
Running Simulations with LSCS Job Control Mode 882
How is LSCS Different from ICRP? 882
LSCS Architecture 883
Expression Evaluation Process 884
Benefits of LSCS Job Control Mode 884
Configuring the Job Policy Setup for LSCS 886
Specifying Job Timeouts for LSCS 889
Setting up the Netlisting Job Policy for LSCS 891
Setting up the Resource Estimation Options for LSCS 894
Starting a Simulation 902
Reviewing Simulation and Netlisting Progress 902
Automatically Suspend Simulations When the Disk Space is Low 905
Restarting the Jobs 908
Viewing Log Files 909
Viewing Log Files for a Data Point 909
Viewing a Consolidated Log for a History 912
Saving Logs in a Compressed File 913

© 2023 Cadence Design Systems, Inc.

Using Job Logs for Debugging . . . 916
Log Files for LSCS Job Control Mode . . . 918
Debug Utility Form . . . 919
Running Simulations With ICRP Job Control Mode . . . 922
Specifying Multiple Run Options . . . 922
Setting Up Resource Options . . . 923
Setting Up Run Options . . . 927
Reporting Identical Histories Before Running a Simulation . . . 932
Running Pre-run Scripts before Simulation Runs . . . 936
Adding a Pre-Run Script . . . 937
Modifying a Pre-Run Script . . . 940
Using the Run Preview Table . . . 942
Refreshing the Setup Summary and Points Table . . . 944
Enabling or Disabling Run Points in Points Table . . . 947
Using the Information Displayed on the Run Preview Tab . . . 949
Modifying the Format of the Run Preview Tab . . . 951
Using Convergence Aids . . . 952
Disabling and Enabling All Point Sweeps . . . 953
Disabling and Enabling All Corners . . . 953
Starting a Simulation . . . 954
AMS Simulations in ADE Assembler . . . 957
Simulation Snapshots in ADE Assembler . . . 957
Saving Simulation Snapshots . . . 957
Restarting a Simulation from a Saved Snapshot . . . 958
Simulation Runtime Improvements . . . 959
Specifying a Name for Simulation History . . . 965
Checking Run Status on the Progress Bar . . . 967
Suspending and Resuming Netlisting or Simulation Runs . . . 969
Suspending Simulations Automatically When the Disk Space is Low . . . 969
Suspending Simulations Manually . . . 971
Suspending Netlist Service Jobs Manually (LSCS Only) . . . 974
Stopping a Simulation . . . 976
Canceling Simulations for Selected Tests or Corners . . . 977
Canceling Simulations for Selected Corners . . . 977
Canceling Simulations for Selected Tests . . . 980
Stopping Jobs and Resubmitting Simulations . . . 980

© 2023 Cadence Design Systems, Inc.

Troubleshooting Jobs that Stay Pending until Timeout . . . 983
Ignoring Design Changes During Run . . . 985
Continuing the In-Process Simulations After ADE Assembler GUI Exits . . . 987
Enabling Continuation of the In-Process Simulations . . . 988
Supported Scenarios . . . 988
Limitations . . . 994
Viewing Job Status . . . 994
Job List . . . 995
Job Summary . . . 997
Viewing the Netlist . . . 999
Viewing the ADE Assembler Logs After Running Simulations . . . 1000
Viewing the Job Log . . . 1005
Viewing Job Log for an ICRP . . . 1005
Viewing Job Log for a Result . . . 1005
Viewing the Simulation Output Log File . . . 1008
Simulation Errors . . . 1009
Running an Incremental Simulation . . . 1011
Running an Incremental Simulation to Reuse Results . . . 1011
Running an Incremental Monte Carlo Simulation . . . 1017
Viewing Differences Between the Active Setup and the Reference History Setup 1019
Reusing a Netlist from Another History . . . 1020
Rerunning Simulation after Modifying the Netlist . . . 1022
Simulating Only Unfinished or Erroneous Points . . . 1024
Rerunning All Unfinished or Erroneous Points . . . 1024
Rerunning Specific Unfinished or Erroneous Points . . . 1026
Running Erroneous or Incomplete Points By Referencing the Simulation Results of a Previous Run . . . 1027
Setting Up Debug Options . . . 1028
Debugging Points . . . 1031
Creating and Running an OCEAN Script . . . 1042
Creating an OCEAN Script . . . 1042
Modifying an OCEAN Script in ADE Assembler . . . 1043
Running an OCEAN Script . . . 1045
Running Parallel OCEAN XL Simulation Runs for a maestro View . . . 1046
Viewing Results of Simulations Run using OCEAN Scripts . . . 1048

© 2023 Cadence Design Systems, Inc.

18
Debugging with ADE Diagnostics 1049
Using the Debug Utility Form 1049
ADE Diagnostics Report 1053
Check Results 1053
Details 1054
Point Mapping 1054
Checks Performed by ADE Diagnostics 1056
Generating the ADE Diagnostics Report 1056

19
Viewing, Printing, and Plotting Results 1059
Viewing Data on the Results Tab 1060
Working with Tabs for Simulation Checkpoints 1060
Results Views 1062
Capacitance 1076
Customizing the Detail - Transpose View 1082
Specifying the Default Formatting Options 1084
Configuring the Results Tab 1086
Customizing the Display of Variables and Parameters for Design Points 1093
Display All Fixed and Swept Variables and Parameters 1094
Show or Hide Design Variables Fixed for Some Tests 1094
Viewing the Run Statistics 1097
Freezing Columns in the Detail and Optimization View 1100
Sorting Results in the Detail and Detail-Transpose Views 1101
Filtering Results on the Results Tab 1102
Saving Commonly Used Filters 1104
Plotting Results 1106
Plotting Across All Corners and Sweeps 1108
Distributed Plot Commands 1116
Using Direct Plot 1118
Performing Quick Plotting 1122
Selecting the Plot Mode 1125
Setting Default Plotting Options for All Tests 1126

© 2023 Cadence Design Systems, Inc.

Setting Plotting Options for Specific Tests . . . 1133
Refreshing Graphs . . . 1136
Printing Results . . . 1163
Printing Noise Summary Information . . . 1168
Printing DC Mismatch Summary . . . 1177
Printing Pole Zero Summary . . . 1179
Backannotating Data from Results to Schematic . . . 1181
Viewing Waveform Info-Balloons on Schematic . . . 1182
Re-Evaluation of Expressions and Specifications . . . 1184
Re-evaluating Results Using FRE Mode . . . 1184
Distributed Re-Evaluation (DRE) Mode . . . 1186
Enabling Re-Evaluation for Unsuccessful Simulations . . . 1188
Re-Evaluating Results for an Older History . . . 1190
Viewing the Spec Summary . . . 1192
Comparing Results Using the Spec Comparison Form . . . 1196
Opening the Spec Comparison Form . . . 1197
Comparing Results for History Items . . . 1200
Comparing Results for Specific Tests . . . 1204
Comparing Results for Specific Design Points . . . 1206
Comparing the Detailed Results for Output Expressions . . . 1211
Hiding and Showing the Comparison Data for Tests . . . 1214
Hiding and Showing an Output Expression in the Spec Comparison . . . 1214
Updating the Spec Comparison with the Latest Results . . . 1214
Sorting Data in the Spec Comparison Form . . . 1214
Saving a Spec Comparison . . . 1215
Opening a Spec Comparison . . . 1215
Deleting a Spec Comparison . . . 1215
Exporting a Spec Comparison to a HTML or CSV File . . . 1215
Managing Results . . . 1217
Annotating Simulation Results . . . 1217
Saving Results . . . 1223
Exporting Results . . . 1224
Working in the Results Display Window . . . 1229
Using SKILL to Display Tabular Data . . . 1238
Showing Variable and Parameter Values for Design Points on the Results Tab . . . 1239
Showing Test Variables that are Fixed for Selected Tests . . . 1239

© 2023 Cadence Design Systems, Inc.

Showing Test Variables that are Fixed for All Tests 1240
Showing Variable Parameters for Design Points in the Variable Display Assistant . . 1241
Searching for Conditional Results 1243
Filtering Out Components by Model Name 1245
Sorting Components by Parameter Value 1247

20
Working with History Checkpoints 1249

Specifying How Many Histories to Save 1250
Overwriting a History Item during Subsequent Simulation Runs 1252
Viewing Histories 1256
Adding Notes to a History 1257
Deleting Notes from a History Item 1258
Renaming Histories 1259
Restoring a History Checkpoint 1260
Restoring an Entire Checkpoint 1260
Restoring Part of a Checkpoint 1260
Viewing Results from a Particular History Checkpoint 1262
Saving Results from a Particular Checkpoint 1262
Viewing Results for a Particular Checkpoint in the Results Browser Window 1263
Viewing the Run Log for a Particular Checkpoint 1263
Opening a Terminal Window in the Results Directory for a Particular History Item . . . 1263
Deleting Simulation Data for a History Item 1264
Deleting a History Checkpoint 1264
Locking and Unlocking a History Item 1265
Importing Histories from Another Cellview 1266
Merging Histories 1268

21
Working with Run Plan 1279

The Run Plan Assistant 1281
Creating a New Run 1283
Creating a Run by Using the Active Setup 1283
Creating a Run with a New Setup 1286
Creating a Run by Copying Another Run 1289

© 2023 Cadence Design Systems, Inc.

Modifying a Run in the Run Plan . . . 1292
Changing the Run Mode . . . 1293
Modifying the Simulation Options . . . 1293
Modifying the Values of Global Variables and Parameters . . . 1294
Modifying Corners . . . 1295
Modifying the Reliability Setup . . . 1297
Copying Settings from One Run to Another in a Run Plan . . . 1300
Changing the Display Order of Runs in a Run Plan . . . 1301
Enabling or Disabling Runs in a Run Plan . . . 1302
Rules to Use Local or Global Variables for Run Plan Simulations . . . 1303
Using Pre- and Post-Run Scripts for Optimization Runs . . . 1307
Executing the Run Plan . . . 1311
Executing all the Runs Together . . . 1311
Executing a Selected Run . . . 1312
Executing a Run Plan Based on Conditions . . . 1312
Executing Runs Using a Script . . . 1319
Viewing Results of Run Plan Simulations . . . 1321
Rerun Simulations for Erroneous or Unfinished Points in a Run Plan . . . 1323
Using Results of One Run in Another . . . 1324
Using calcVal in Run Plan . . . 1324

22
Working with Specifications . . . 1329

Working with Specifications . . . 1330
Defining a Specification . . . 1331
Using Variables in Specifications . . . 1332
Overriding the Measurement Specification for a Corner . . . 1334
Removing a Corner Specification Override . . . 1339
Disabling and Enabling Corner Specifications . . . 1339
Comparing Waveforms Using Wave Specifications . . . 1342
Performing Waveform Comparison . . . 1345
Specifying Global Options for Waveform Comparison . . . 1348
Specifying Local Options for Waveform Comparison . . . 1359
Analyzing Waveform Comparison Results . . . 1363
Analyzing Waveform Comparison Results for Digital Signals . . . 1372

© 2023 Cadence Design Systems, Inc.

Saving Waveform Comparison Results in a Datasheet 1374
Waveform Compare Algorithms 1375
Defining Operating Region Specifications 1378
Copying a Specification 1378
Viewing Specification Results in the Results Tab 1403
Viewing Operating Region Violations 1408
Working with the Specification Summary 1415
Viewing the Spec Summary 1416
Saving a Spec Summary 1419
Opening a Spec Summary 1420
Deleting a Spec Comparison 1420
Adding a Specification to the Spec Summary 1420
Deleting a Specification from the Spec Summary 1421
Changing the History Item from which Results are Displayed 1421
Updating the Spec Summary with the Latest Results 1422
Recreating the Spec Summary from the Results in the Active Results Tab 1423
Viewing the Detailed Results for Specifications 1423
Plotting the Results for Specifications 1425
Exporting a Spec Summary to a HTML or CSV File 1425
Sorting Data in the Spec Summary Form 1426
Hiding and Showing Columns in the Spec Summary Form 1426

23
Working with Documents and Datasheets 1427
Adding Documents 1427
Opening Documents 1428
Removing Documents 1428
Saving Documents 1429
Working with Datasheets 1430
Creating a Datasheet for a History 1430
Displaying Customized Waveform Images in the Datasheet 1442
Opening a Datasheet 1443
Customizing the Datasheet Format and Structure 1443

© 2023 Cadence Design Systems, Inc.

24
Circuit Optimization 1445

Parameterizing the Design 1446
Matching Devices and Device Properties 1446
Defining Values for Optimization 1450
Setting Up Specifications 1452
General Specifications 1452
Device Area Constraint Specifications 1456
Specifying How Much Optimization Data to Save 1461
Data Points—Definition 1463
Design Points—Definition 1463
Running Optimization 1466
Algorithms of Local Optimization 1467
Running a Local Optimization 1469
Running a Global Optimization 1472
Viewing the Variable Data from Optimization Results 1474
Sizing Over Corners 1476
Performing Manual Tuning 1482

25
Sensitivity Analysis 1489

Generating Sensitivity Data 1490
Support for Multi-Technology Simulations in Sensitivity Analysis 1496
Specifying Variables, Parameters, or Model Files to be Varied for Sensitivity Analysis 1497
Selecting Instances and Parameters for Mismatch Variation 1504
Saving Sensitivity Data for DC Operating Point Parameters 1510
Viewing Sensitivity Analysis Results 1524
Reading Data in the Sensitivity Analysis window 1525
Viewing Sensitivity Data in Different Tabs 1526
Working with Sensitivity Data 1530
Changing the Type and Format of Data To View 1530
Changing the Data View Type 1544
Filtering Data Based on Correlation Coefficient 1545
Filtering Data Based on Columns 1545

© 2023 Cadence Design Systems, Inc.

Filtering Data Based on Search Criteria . . . 1547
Resetting Filters . . . 1548
Sorting Data . . . 1548
Hiding Corner Data . . . 1549
Hiding Specification Data . . . 1549
Hiding or Showing the Detailed Results for Corners . . . 1549
Hiding or Showing the Detailed Results for Specifications . . . 1550
Hiding or Showing Statistical Parameters . . . 1550
Showing All Hidden Data . . . 1550
Highlighting Associated Devices in the Schematic . . . 1550
Viewing Results in Multiple Sensitivity Analysis Windows . . . 1551
Changing Number Format . . . 1551
Plotting Sensitivity Analysis Results . . . 1553
Setting the Plotting Mode . . . 1557
Displaying the ADE XL and Sensitivity Analysis Results in the Same Graph . . . 1565
Plotting Average Sensitivity Analysis Results . . . 1567
Displaying Spec Markers in Sensitivity Analysis Plots . . . 1568
Saving the Sensitivity Data . . . 1570

26
Worst Case Corners . . . 1571
Prerequisites for Setting Up the Worst Case Analysis . . . 1571
Specifying the Worst Case Corner Analysis Setup . . . 1572
Running Worst Case Corners . . . 1583
Validating Worst Case Corners . . . 1584
Creating Worst Case Corners Automatically . . . 1588
Viewing Worst Case Corners . . . 1590

27
Multi-Technology Designs and Locally Scoped Models and Options . . . 1597
Enabling Locally Scoped Models and Options . . . 1599
Enabling Local Scoping for a Cell . . . 1601
Enabling Local Scoping for an Instance . . . 1605

© 2023 Cadence Design Systems, Inc.

Disabling Local Scoping 1610
Limiting the View by Library or Instance 1611
Specifying Test and Block Information when Adding Model Files to Corners 1612
Detecting Context Symbol/Name Collisions 1613
Multi-Technology Simulations Using DSPF File 1615

A
Migrating ADE L or ADE XL Cellviews to ADE Assembler 1617
Migrating an ADE L State to ADE Assembler 1618
Loading an ADE L State for a Test in ADE Assembler 1618
Migrating an ADE L State to ADE Assembler Using SKILL 1620
Migrating an ADE XL Setup to ADE Assembler 1621
Opening an ADE XL Cellview in ADE Assembler 1621
Importing an ADE XL Setup in an Existing Cellview in ADE Assembler 1622
Loading an ADE XL Setup Using SKILL 1624
Reusing Menu Customization from ADE XL 1625
Importing Menu Customization From a Trigger in .cdsinit 1625
Migrating an Ocean script from ADE XL to ADE Assembler 1626
Job Policy Settings in Migrated Cellviews 1627

B
ADE Assembler Environment Variables 1629
maestro.assembler 1631
defaultOutputColumns 1632
outputSetupFilteringNumOutputThreshold 1633
maestro.caa 1634
enableDochecklimitFromDataView 1635
exportBatch 1636
refreshViolationPerCorner 1637
maestro.calcVal 1638
debug 1639
maestro.commonTestSetup 1641
defaultComponents 1642
maestro.debug 1643
beanstalk 1644

© 2023 Cadence Design Systems, Inc.

mainVirtuoso . . . 1646
netlisting . . . 1648
simAndEval . . . 1650
maestro.distribute . . . 1652
. . . 1652
jobCleanupTimeoutBeforeKill . . . 1653
commandJobTerminateTimeoutBeforeKill . . . 1654
continueICRPNumRetriesToWaitForGUIRecoveredResDB . . . 1655
inferCommandStatusFromProxy . . . 1656
maxNetlistingJobsForLocal . . . 1657
showErrorForNonExistingJobPolicy . . . 1658
maestro.faultsimulation . . . 1659
defaultConfidenceLevel . . . 1660
defaultBridgeFaultRes . . . 1661
defaultOpenFaultRes . . . 1662
defaultStuckAtFaultRes . . . 1663
dropUndetectedSimErrorsInFaultDroppingFlow . . . 1664
handleErrorsAs . . . 1665
handleSimulationErrorsAs . . . 1666
suppressAddToFaultGroupsPrompt . . . 1667
warnWhenSimsExceed . . . 1668
maestro.gui . . . 1669
alwaysEnableSaveButton . . . 1670
autoExpandSetupStatesSubTree . . . 1671
copytoRunPlanFromActiveSetup . . . 1672
displayAgingModelFiles . . . 1673
defaultAgingPlacementType . . . 1674
defaultAgingModelFiles . . . 1675
displayFixedTestVariables . . . 1676
displayImplicitSignals . . . 1677
displayJobSummaryBar . . . 1678
defaultReliabilityMode . . . 1679
detailTransposeViewShowTest . . . 1680
defaultReliabilityMode . . . 1682
enableManualRefresh . . . 1683
enableWCCfromRunPlanInActiveSetup . . . 1684

© 2023 Cadence Design Systems, Inc.

envVarCategoriesForDump . . . 1685
ignoreLockHistoryForReliability . . . 1686
manualResultsViewRefresh . . . 1687
openExplorerOnTestDoubleClicked . . . 1689
removeDuplicateCornersInResultMerge . . . 1690
showCapacitanceView . . . 1691
showOverriddenValueForSpec . . . 1692
showHistoryPrefixToolbar . . . 1693
showSchematicCrossSelectionHyperlinkInSimLog . . . 1694
maestro.logging . . . 1695
databaseDir . . . 1696
level . . . 1697
maxNumberSimulatorNoteWarn . . . 1698
maestro.lscs . . . 1699
defaultNetlistingJobPolicy . . . 1700
defaultSimulationJobPolicy . . . 1701
jobHealthCheckInterval . . . 1702
preferNetlistingSameTest . . . 1703
runBeforeSimulation . . . 1704
useLSCSFor3rdPartySimulators . . . 1705
maestro.plotting . . . 1706
reevaluateBeforePlotAll . . . 1707
maestro.monte . . . 1708
reliabilitymode . . . 1709
maxPointsPerJob . . . 1710
maestro.results . . . 1711
continueRunOnRdbFatalError . . . 1712
defaultReliabilityFilterDirectory . . . 1713
defaultRunStatisticItems . . . 1714
enableRunStatisticInfoDisplay . . . 1716
forceRdbStatusEnable . . . 1717
forceRdbStatusIgnoreList . . . 1718
forceRdbStatusTo . . . 1719
keepTmpRdbForReevaluation . . . 1720
postLayoutToSchematicInRelReport . . . 1721
RDBCreatePolicy . . . 1722

© 2023 Cadence Design Systems, Inc.

RDBTmpDirDiskQuotaLowerLimitInMegabytes . . . 1723
RDBNFSDiskQuotaLowerLimitInMegabytes . . . 1724
RDBTmpTouchInterval . . . 1725
RDBSynchronous . . . 1726
saveEvaluationErrorResults . . . 1728
saveFingersForOpParams . . . 1729
showDeviceCheckCumulativeEndTime . . . 1730
sumDeviceFingerCurrents . . . 1731
waiveViolationsByCorner . . . 1732
maestro.script . . . 1733
accessMode . . . 1734
exitAfterFinished . . . 1735
writeCallbackForSingleRun . . . 1736
maestro.setupdb . . . 1737
allowLocalCornersInRunPlan . . . 1738
autoDeleteMalformedLockedSDB . . . 1739
syncWithSLA . . . 1740
maestro.simulation . . . 1741
allowVariableCheckingDuringConfigSweep . . . 1742
behaviorOnPcellParamChangedByCallback . . . 1743
checkDesignInIdenticalHistory . . . 1744
checkSimFilesInReportIdenticalHistory . . . 1745
dspfNamePrefixSimInfo . . . 1746
exprEvalMaxTries . . . 1747
endNetlistingImmediatelyOnStop . . . 1748
groupRunA . . . 1749
ignoreAnalysisCheckInResourceEstimation . . . 1750
ignoreRunModeCheckInResourceEstimation . . . 1751
interactiveA . . . 1752
reportChangesAfterComparingSetupWithHistories . . . 1753
reportIdenticalHistoryIgnoreElementsList . . . 1754
retainNetlistLinkForRerun . . . 1755
reuseNetlistForRerun . . . 1756
runModesForIdenticalHistoryCheck . . . 1757
simulationServiceJobPolicy . . . 1758
tempDirectory . . . 1759

© 2023 Cadence Design Systems, Inc.

useReportIdenticalHistory . . . 1760
useSelfCDFInPostLayoutNetlist . . . 1761
waitToAcquireROLock . . . 1763
warnWhenWCCSimsExceed . . . 1764
maestro.sla . . . 1765
autoUpdateTagForReference . . . 1766
overwriteEntireCorner . . . 1767
saveCompositeSetups . . . 1768
maestro.test . . . 1769
autoPromoteVarsToGlobal . . . 1770
maestro.waveSpec . . . 1771
abstol . . . 1772
biasShift . . . 1773
disableChecks . . . 1774
excludeTimes . . . 1775
reltol . . . 1776
timeTolerance . . . 1777
timeDelay . . . 1778
compareAlgorithm . . . 1779
maestro.mts . . . 1780
MTSremoveOldConfigSetupAfterMigration . . . 1781
adexl.algorithm . . . 1782
linearModelR2 . . . 1783
quadraticModelR2 . . . 1784
adexl.setupdb . . . 1785
adexl.test . . . 1785
adexl.simulation . . . 1785
matlabExprReevalSearchUpdateLevel . . . 1787
matlabStartTimeout . . . 1788
matlabEnableIdleTimeout . . . 1789
matlabIdleTimeout . . . 1790
checkInstanceBindings . . . 1791
adexl.distribute . . . 1792
adexl.monte . . . 1792
adexl.historyNamePrefix . . . 1793
runPlan . . . 1794

© 2023 Cadence Design Systems, Inc.

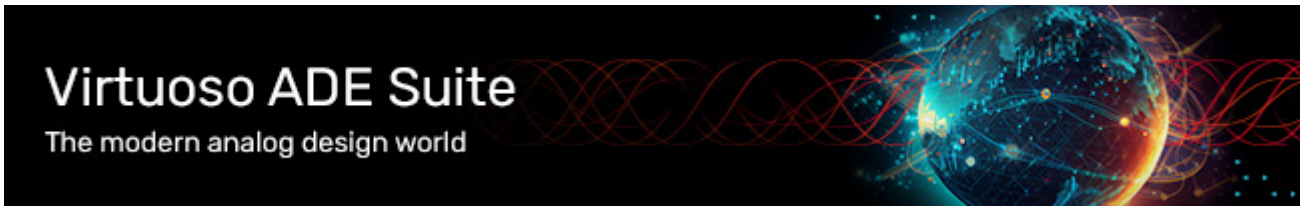
adexl.algorithm . . . 1795
linearModelR2 . . . 1796
quadraticModelR2 . . . 1797
adexl.setupdb . . . 1798
adexl.test . . . 1798
adexl.simulation . . . 1798
matlabExprReevalSearchUpdateLevel . . . 1800
matlabStartTimeout . . . 1801
matlabIdleTimeout . . . 1802
checkInstanceBindings . . . 1803
adexl.distribute . . . 1804
adexl.monte . . . 1804
adexl.historyNamePrefix . . . 1805
runPlan . . . 1806
adexl.icrpStartup . . . 1807
adexl.results . . . 1807
adexl.gui . . . 1807
adexl.cpupdtr . . . 1809
adexl.datasheet . . . 1809
asimenv.startup . . . 1810
adexl.oceanxl . . . 1810
adexl.plotting . . . 1810
asimenv.plotting . . . 1810
asimenv . . . 1811

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

1

Getting Started with Virtuoso ADE Assembler



Introducing Virtuoso® ADE Assembler

Virtuoso® ADE Assembler is an interactive, multi-test environment that is designed to pull together all the parts of the design to begin centering the design for manufacturing. After testing and tuning the performance of individual blocks in Virtuoso ADE Explorer, you can port the tests for different blocks into Virtuoso ADE Assembler to test them together under a multitude of conditions and to check how they interact with each other in the design.

In Virtuoso ADE Assembler, you can consolidate tests for multiple blocks of a design, including the variables, device parameters, corners, checks and assertions, along with their specifications. Here, you can apply an extensive set of analyses to push the blocks and the overall design to its limits.

The Virtuoso ADE Assembler environment allows you to do the following:

- Create a new test in Virtuoso ADE Assembler or bring in an existing test that is already defined and verified by using Virtuoso ADE Explorer, Virtuoso Analog Design Environment L, or Virtuoso Analog Design Environment XL.
- Define and place design variables (including equations) on the schematic elements

  The simulator evaluates the variables and equations, automatically passing the modified values down through the schematic hierarchy. You can archive important aspects of your

© 2023 Cadence Design Systems, Inc.

design with your schematic, which can contain both design equations and circuit topology details.

- Annotate the schematic with DC voltages and transistor operating point information
- Specify device constraints and commands to control the simulation environment
- Move tests to and from Virtuoso ADE Explorer for detailed debugging.
- The environment supports the following Cadence circuit simulators:
  - Accelerated Parallel Simulator
  - Spectre
  - SpectreRF
  - UltraSim
  - AMS Designer

  Alternatively, you can integrate your own third-party simulator using the Open Analog Simulator Interface Socket (OASIS) Direct. See your local Cadence representative for more information and to request a copy of the *OASIS Direct Integrator's Guide*.
- Set up and run various analyses:
  - Parametric sweeps
  - Corners analysis
  - Monte Carlo statistical analysis
  - Circuit and device checks
  - Worst case corners analysis
  - Local optimization
  - Global optimization
  - Manual tuning
- View waveforms and perform post-processing tasks on your simulation results using Virtuoso Visualization and Analysis XL
- Generate datasheets
- View and manipulate your active setup as well as histories or checkpoints from previous runs
- Backannotate parameter values to schematic

© 2023 Cadence Design Systems, Inc.

- Create run plans. In a run plan, you can create multiple variations of the active setup and run them in a single click. This feature enables the support for regression using SKILL functions in Virtuoso ADE Assembler.
- Integrate third-party calculators and waveform viewers into your Virtuoso ADE Assembler environment
- Customize your user interface (see the *Virtuoso Design Environment User Guide*)

Video

You can view a video demonstration of this feature at Introduction to ADE Assembler. Also read a blog related to this at Getting Started with the New ADE Product Suite.

© 2023 Cadence Design Systems, Inc.

Licensing Requirements

Virtuoso ADE Assembler requires the 95260, Virtuoso ADE Assembler license.

Additionally, you can use the 95290, Virtuoso Simulation Expansion Option license to enhance your existing setup to run more simulation jobs in parallel. This license enables simulation of additional 400 jobs in the existing setup. For every increment of 400 jobs, a separate instance of this license can be checked out, which is released after all the jobs and simulations are completed. These increments of 400 jobs are allocated per GUI process or simulation service, which means that if multiple GUI processes are launched, the jobs started by these GUI processes are not accumulated with one process but distributed evenly.

This license is especially helpful in cases when many jobs cannot be run in parallel due to technical limitations, such as NFS, machine load, or more.

- Without this license, you can run a maximum of 399 jobs in parallel using one GUI process.
- With one license, you can run 799 (399 + 400) jobs in parallel using one GUI process.
- Similarly, with two licenses, you can run 1199 (399+400+400) jobs in parallel using one GUI process.

In case of multiple GUI processes that individually run less than 400 jobs, no license is checked out and all simulations continue to run in parallel. The license is checked out only when the jobs from a GUI process reach the count of 400.

You would require additional licenses to run:

- Resimulations with layout-dependent effects (LDEs) as described in Chapter 13 of this user guide. For more details, refer to License Requirements for Running Simulations with LDE.
- Resimulations as part of the electrically aware design flow described in the Virtuoso® Electrically Aware Flow Guide
- Resimulations as part of the voltage dependent rules flow described in the Virtuoso® Voltage Dependent Rules Flow Guide

Note: If 95600 (`Virtuoso_Layout_Suite_EAD`) is already checked out as part of the Electrically Aware Design flow, 95510 is not required.

For information on licensing, see *Virtuoso Software Licensing and Configuration Guide*.

© 2023 Cadence Design Systems, Inc.

Getting Started in the Virtuoso ADE Assembler Environment

You can launch Virtuoso ADE Assembler from the Virtuoso Design Environment Command Interpreter Window (CIW) or from your schematic editing window.

The Virtuoso ADE Assembler environment consists of a set of menus, toolbars and assistant panes that make up your workspace. You can load a Cadence workspace or create and load a custom workspace. You can specify what workspace to load for a given cellview. For more information about workspaces, see Getting Started with Workspaces in the *Virtuoso Design Environment User Guide*.

See the following topics for more information about the Virtuoso ADE Assembler environment:

- Launching ADE Assembler on page 38
- ADE Assembler Environment at Startup on page 45
- ADE Assembler Environment User Interface on page 49
- Specifying the Run Mode on page 83
- Specifying Options for Saving Simulation Results on page 85
- Specifying Results Database Location on page 88

One you have created a test setup cellview, you can open it in the usual ways. See "Working with Cellviews" in the *Virtuoso Design Environment User Guide* for more information.

© 2023 Cadence Design Systems, Inc.

Launching ADE Assembler

To start the ADE Assembler environment, start the Virtuoso Design Environment and do one of the following:

To open the environment from the CIW and create a new cellview

1. In the CIW, choose *File – New – Cellview*.

   The New File form appears.

2. In the *Library* field, select a library.
3. In the *Cell* field, type a new or existing cell name.
4. In the *View* field, type a name for your cellview. The default view name is `maestro`.
5. In the *Type* field, select `maestro`.
6. Ensure that `ADE Assembler` is selected in the *Open with* drop-down list.
7. Click *OK*.

   The ADE Assembler environment is displayed.

© 2023 Cadence Design Systems, Inc.

To open an existing maestro cellview from the CIW

1. In the CIW, choose *File – Open*.

   The Open File form appears.
2. Select a maestro cellview.
3. Click *OK*.

To open the environment from the schematic and create a new maestro cellview

1. In the schematic editing window, choose *Launch – ADE Assembler*.

   The Launch ADE Assembler form appears.
2. Perform the procedure described in Creating a New Setup.

Note: If you have descended into a design hierarchy, the environment returns you to the top level of your design when you choose *Launch – ADE Assembler*.

To open an existing maestro cellview from the schematic

1. In the schematic editing window, choose *Launch – ADE Assembler*.

   The Launch ADE Assembler form appears.
2. Perform the procedure described in Opening an Existing Setup.

Note: If you have descended into a design hierarchy, the environment returns you to the top level of your design when you choose *Launch – ADE Assembler*.

To open the environment from the CIW without first selecting a design

➤ In the CIW, choose *Tools – ADE Assembler*.

The Launch ADE Assembler form appears. From this form, you can create a new test setup or open an existing test setup. See Using the Launch ADE Assembler Form for more information.

© 2023 Cadence Design Systems, Inc.

Using the Launch ADE Assembler Form

The Launch ADE Assembler form appears when you do one of the following:

- Choose *Tools – ADE Assembler* in your Command Interpreter Window (CIW).
- Choose *Launch – ADE Assembler* in your schematic editing window.

From this form, you can create a new test setup or open an existing test setup. See the following sections for information about getting started with any of these tasks:

- Creating a New Setup
- Opening an Existing Setup

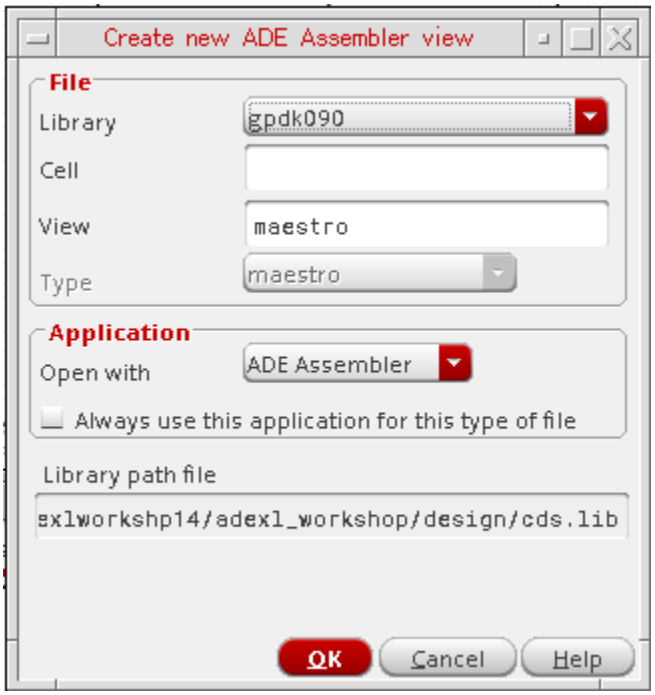
© 2023 Cadence Design Systems, Inc.

Creating a New Setup

To create a new setup, do the following:

1. On the Launch ADE Assembler form, select *Create New View*.
2. Click *OK*.

   The Create new ADE Assembler View form appears.



3. In the *Library Name* drop-down list, select a library.
4. In the *Cell Name* field, type a cell name for your testbench.
5. In the *View Name* field, type a view name for your test setup.

   The default ADE Assembler view name and view type is `maestro`. For more information about the maestro view, see Directory Structure of a Maestro Cellview.
6. Verify that *ADE Assembler* appears in the *Open with* drop-down list.
7. (Optional) Select the *Always use this application for this view type* check box if you want the program to use ADE Assembler when opening a view that is the same as what you specified in the *View Name* field.

© 2023 Cadence Design Systems, Inc.

8. In the *Open in* field, do one of the following:

Select	To
new tab	Open the cellview in a new tab in the schematic editing window. **Note:** If a schematic editing window is not open, ADE Assembler is opened as a main application in a new window.
current tab	Open the cellview in the current tab in the schematic editing window. **Note:** If a schematic editing window is not open, ADE Assembler is opened as a main application in a new window.
new window	Open the cellview in a new window. ADE Assembler is opened as a main application in a new window.

9. Click *OK*.

The ADE Assembler environment appears. For more information, see ADE Assembler Environment at Startup on page 45 and ADE Assembler Environment User Interface on page 49.

You can begin specifying your tests and analyses. See Chapter 2, “Specifying Tests and Analyses” for more information.

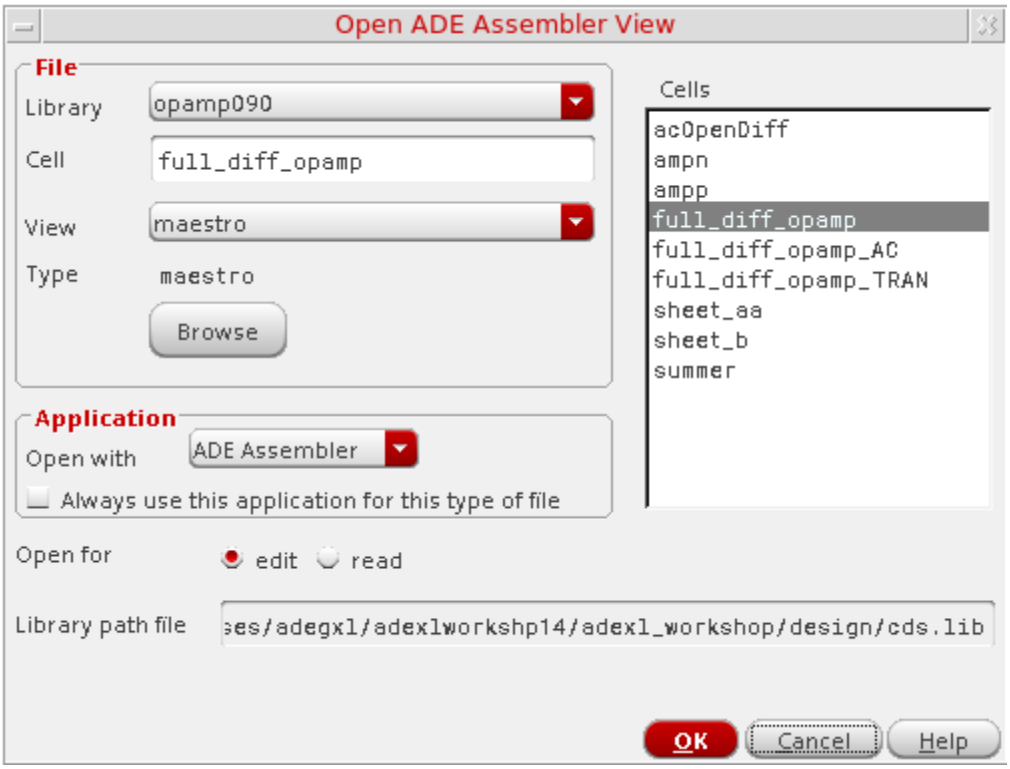
© 2023 Cadence Design Systems, Inc.

Opening an Existing Setup

To open an existing setup, do the following:

1. On the Launch ADE Assembler form, select *Open Existing View*.
2. Click *OK*.

   The Open ADE Assembler View form appears.



3. Use the *Library Name*, *Cell Name*, and *View Name* fields to select your setup.
4. Verify that `ADE Assembler` appears in the *Open with* drop-down list.
5. (Optional) Select the *Always use this application for this view type* check box if you want the program to use ADE Assembler when opening a view that is the same as what you specified in the *View Name* field.
6. (Optional) The *Open for* field indicates that the maestro view will be opened in edit mode by default. Select *read* to open the maestro view in read-only mode.

   The ADE Assembler title bar displays the text *Editing* if the ADE Assembler view is opened in edit mode. The ADE Assembler title bar displays the text *Reading* if the view is opened in read-only mode. For more information about working with ADE Assembler views in read-only mode, see Working with Read-Only or Editable maestro Views.

© 2023 Cadence Design Systems, Inc.

7. In the *Open in* field, do one of the following:

Select	To
new tab	Open the cellview in a new tab in the schematic editing window. **Note:** If a schematic editing window is not open, ADE Assembler is opened as a main application in a new window.
current tab	Open the cellview in the current tab in the schematic editing window. **Note:** If a schematic editing window is not open, ADE Assembler is opened as a main application in a new window.
new window	Open the cellview in a new window. ADE Assembler is opened as a main application in a new window.

8. Click *OK*.

   The ADE Assembler environment appears.

Important

The cellviews saved from both ADE Explorer or ADE Assembler are of type maestro, and can be opened in either one of these applications. Virtuoso remembers the application that was last used to save each maestro cellview and by default, opens the view in the same application next time. However, you can still open a maestro cellview in a specific application by using the `Open with` command.

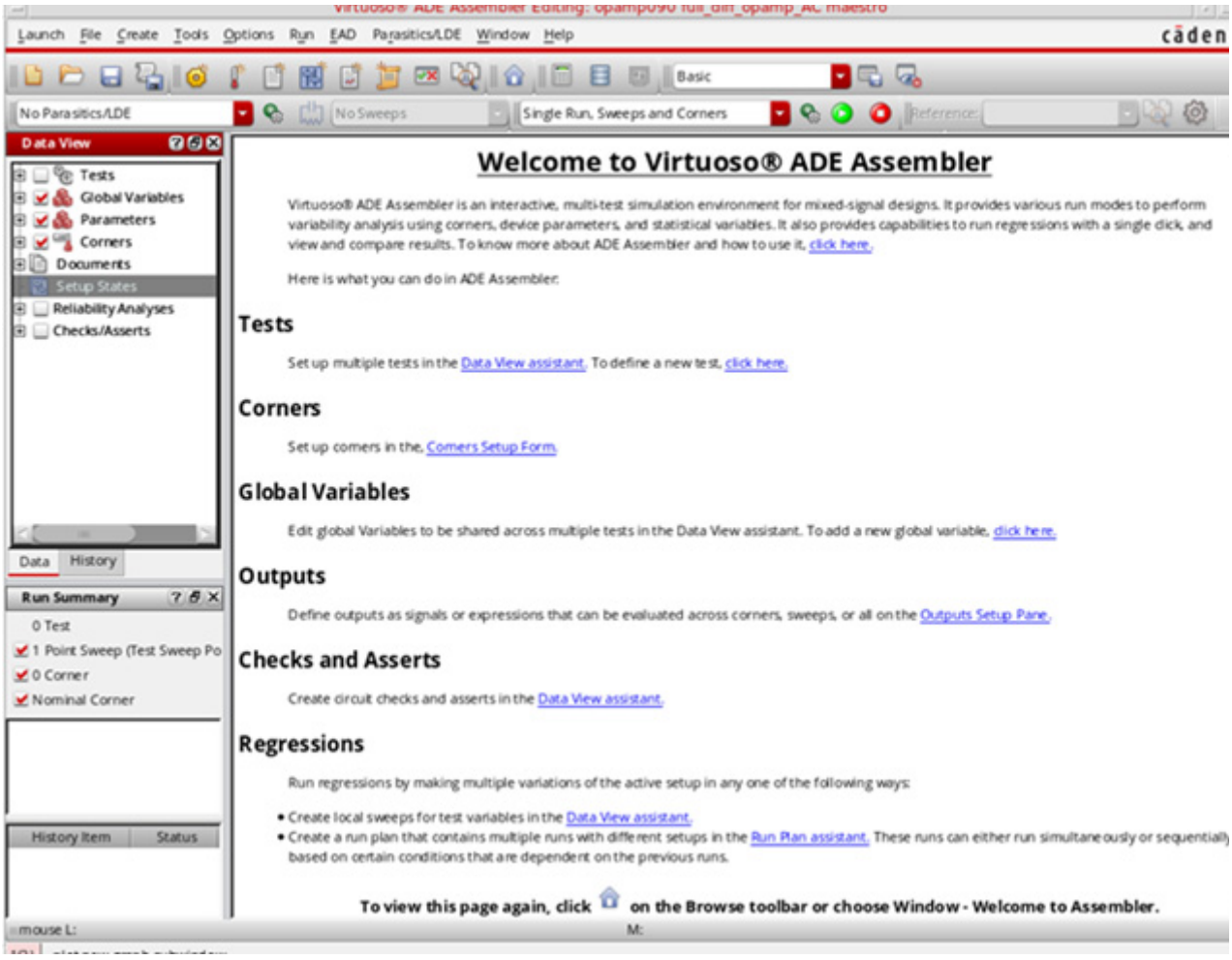
If you saved a maestro view using ADE Assembler, but now you want to open it in ADE Explorer, right-click the view name and choose *Open with*. The Open File form appears with the name of the selected library, cell, and view displayed in it. In the *Open with* drop-down list, select `ADE Explorer`. If the cellview contains multiple tests, Virtuoso displays the Choose a Test form that shows a list of tests available in that view. Since ADE Explorer can show only a single test, select the name of the test you want to open in ADE Explorer and click *OK* to open the view. For more details, refer to Modifying a Test in ADE Explorer.

© 2023 Cadence Design Systems, Inc.

ADE Assembler Environment at Startup

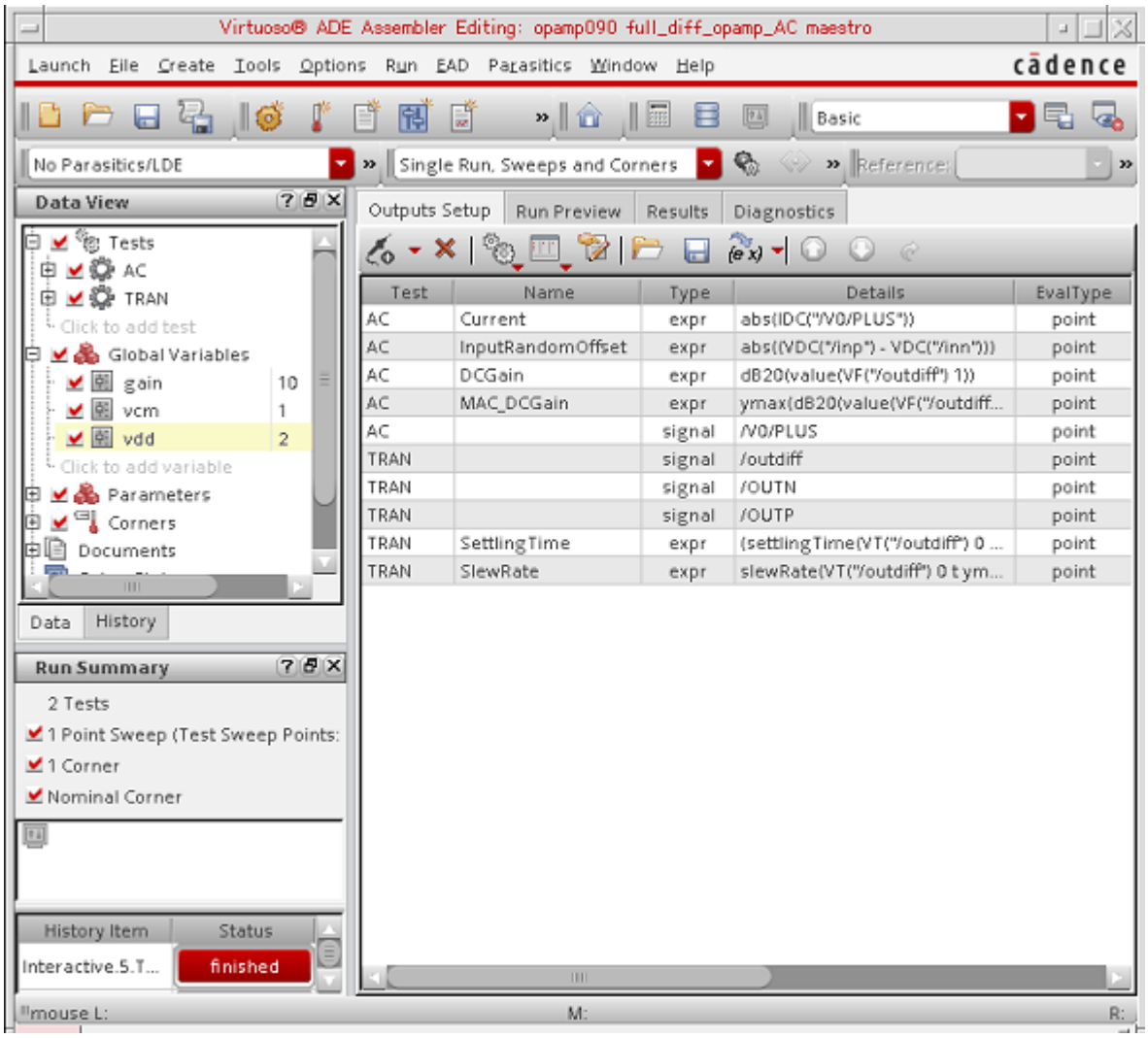
This section describes how the ADE Assembler environment appears on startup.

If you open a new cellview in ADE Assembler. the Welcome to Virtuoso ADE Assembler start page is displayed. For more information about the Welcome to ADE Assembler start page, see The Welcome to ADE Assembler Page on page 50.



© 2023 Cadence Design Systems, Inc.

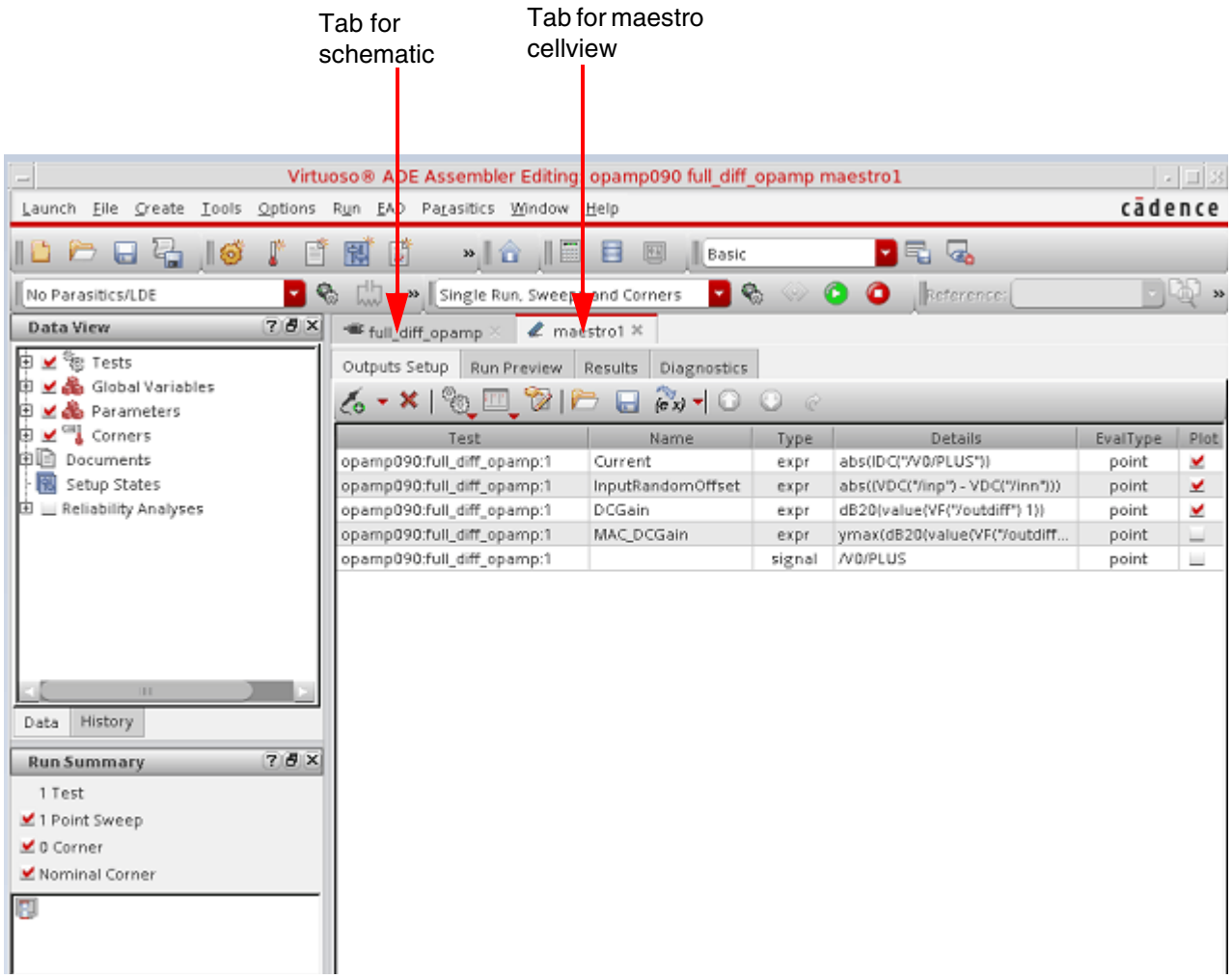
- The Outputs pane is displayed if you open an existing setup using the procedure described in Opening an Existing Setup.



- The ADE Assembler environment is displayed in a new tab if you choose *Launch – ADE Assembler* in your schematic editing window and created a new setup or opened an

© 2023 Cadence Design Systems, Inc.

existing setup using the procedures described in Creating a New Setup or Opening an Existing Setup.

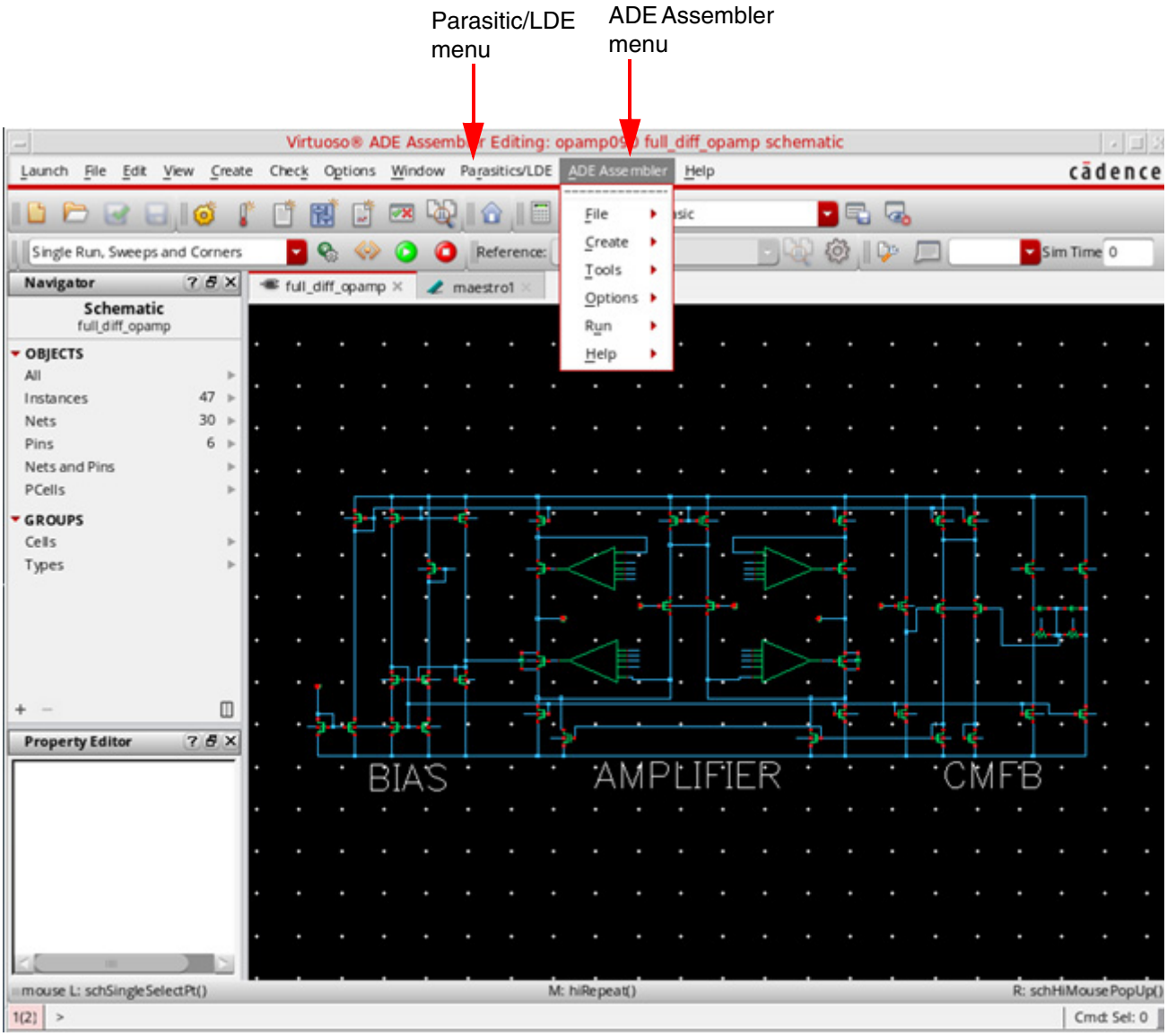


The tab for the schematic design displays the name of the cell for the schematic design. The tab for the ADE Assembler environment displays the name of the maestro cellview. For more information about working with tabs, see the *Virtuoso Design Environment User Guide*.

- In the tab for the schematic design, the *ADE Assembler* and *Parasitics* menus appear on the menu banner. You can access ADE Assembler related commands from the *ADE Assembler* menu. For more information, see Menu Bar. The *Parasitics* menu lets you

© 2023 Cadence Design Systems, Inc.

access Virtuoso Parasitic Aware Design to investigate the effects of parasitics on your circuits. For more information, see the *Virtuoso Parasitic Aware Design User Guide*.



For information about the user interface in the ADE Assembler environment, see ADE Assembler Environment User Interface.

© 2023 Cadence Design Systems, Inc.

ADE Assembler Environment User Interface

The Assembler user-interface provides various menus, toolbars, and assistants that can be used to configure a multi-test environment, manage simulation set up, run simulations, and view results.

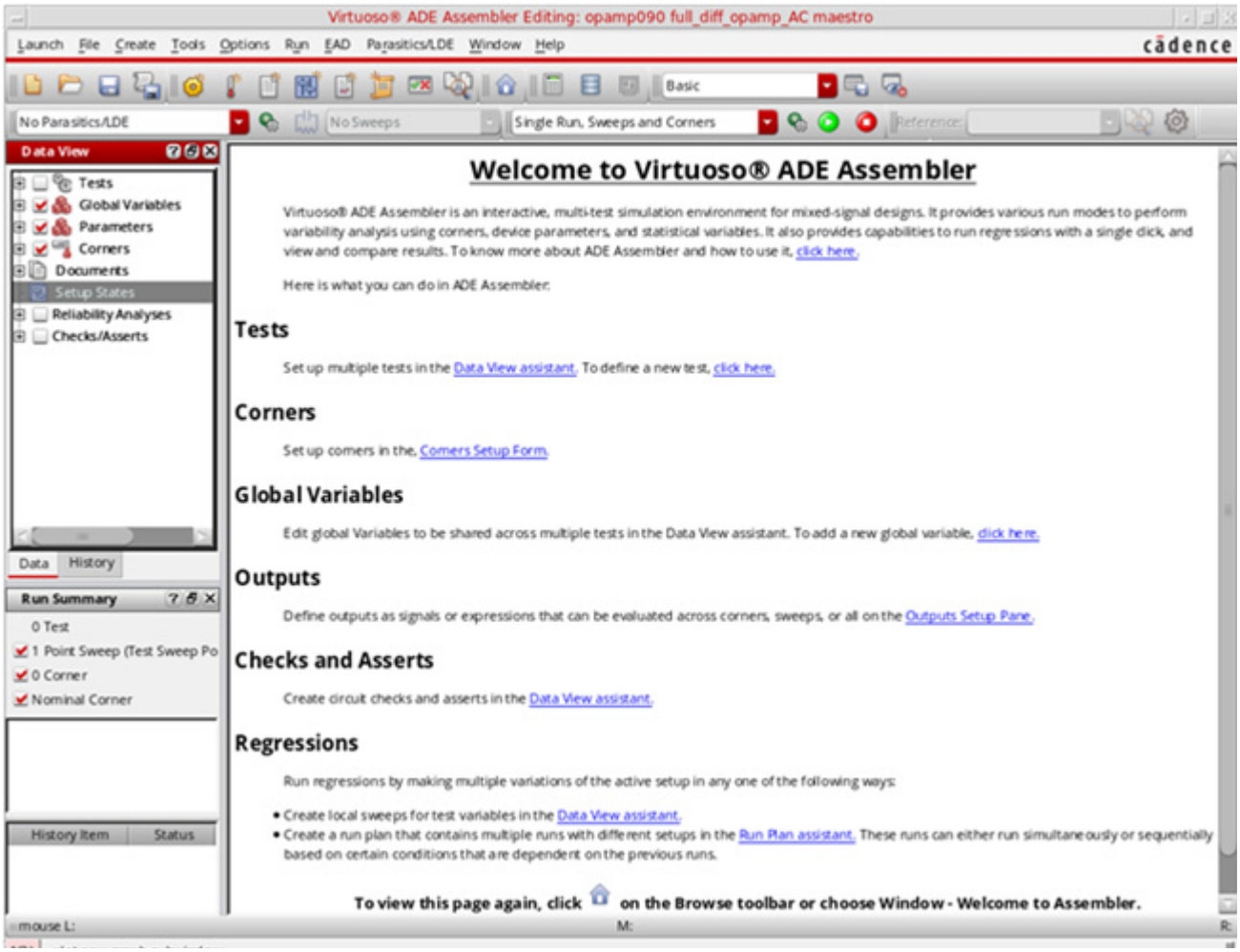
This section describes the following parts of the ADE Assembler environment user interface:

- The Welcome to ADE Assembler Page on page 50
- Menu Bar on page 51
- Context-Sensitive Menus on page 59
- Toolbars on page 66
- Assistant Panes on page 72
- Outputs on page 80
- Workspaces on page 82

© 2023 Cadence Design Systems, Inc.

The Welcome to ADE Assembler Page

The Welcome to ADE Assembler page provides an overview of ADE Assembler and guides you through the basic setup tasks in ADE Assembler. You can click a hypertext link in the start page to open the form corresponding to a setup task.



The welcome page is displayed if you do one of the following:

- Create a new setup using the procedure described in Creating a New Setup on page 41.
- Choose *Window – Welcome to ADE Assembler*.
- Choose ( ) on the Browse toolbar.

© 2023 Cadence Design Systems, Inc.

Menu Bar

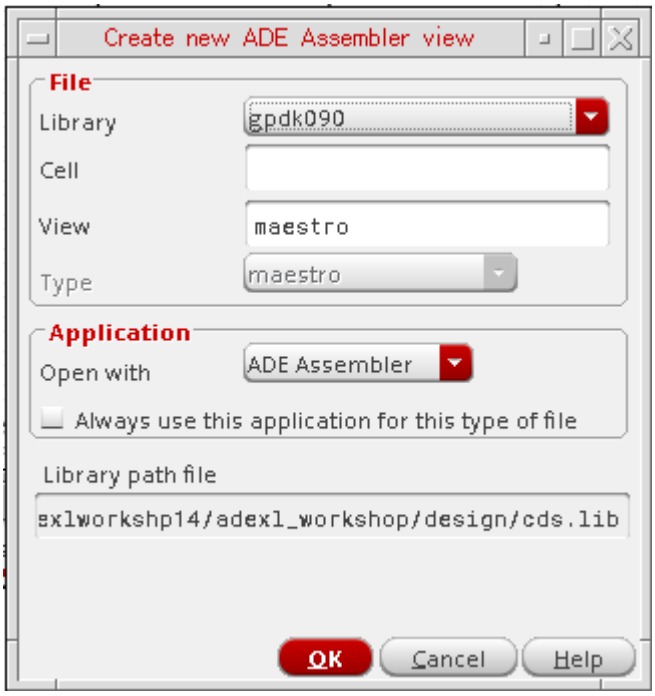
The menu bar in ADE Assembler has the following menus. You can also access these menus from the *ADE Assembler* menu that appears on the menu banner in the tab for the schematic design.

- File
- Create
- Tools
- Options
- Run
- EAD
- Parasitics/LDE
- Window
- Help

© 2023 Cadence Design Systems, Inc.

File

The options in the *File* menu are described below:

File Menu Options	Description
New	Opens the New File form  **Note:** This is the same form that appears when you launch ADE Assembler from the CIW and create a new cellview.
Open	Opens the Open File form
Save	Saves your maestro cellview.

© 2023 Cadence Design Systems, Inc.

File Menu Options	Description
Save As	Opens the Save a Copy form so you can save a copy of the current ADE Assembler setup database to a different location. **Note:** This action does not save history data, only the setup database.
Save Script	Opens the Save OCEAN Script form so you can save simulation setup and conditions to an OCEAN script file.
Save Setup State	Opens the Save Setup State form so you can save a copy of the current ADE Assembler setup database **Note:** This action does not save history data, only the setup database. For more information, see Creating or Updating a Setup State on page 538.
Load Setup State	Opens the Load Setup State form so you can load an existing setup state. For more information, see Loading a Setup State on page 540.
Remove Setup State	Opens the Remove Setup State form so you can delete an existing setup state. For more information, see Deleting a Setup State on page 542.
Import	Opens the Import Setup form so you can import settings from an existing ADE Assembler setup database. For more information, see Importing the Simulation Setup.
Import History	Opens the Import Histories form and lets you import histories from one cellview to another. By doing this, you can bring together histories from different cellviews that have the same simulation setup and were used to run different verifications.

© 2023 Cadence Design Systems, Inc.

File Menu Options	Description
Make Read Only/ Make Editable	Lets you switch between read-only and edit modes. For more details, see Working with Read-Only or Editable maestro Views.
Bookmarks	Allows you to bookmark design views and return to them during the current or future sessions. For information on bookmarks, see Using Bookmarks and Views in the *Virtuoso Design Environment User Guide*.
Close	Closes the ADE Assembler environment

Create

The options in the *Create* menu are described below:

Create Menu Options	Description
Test	Opens the ADE Explorer so that you can add a new test For more information, see Chapter 2, “Specifying Tests and Analyses.”
Corner	Opens the Corners Setup form so that you can setup corners for simulation For more information, see Chapter 6, “Working with Corners”
Document	Opens the Choose Documents to be Added form so that you can select the documents to be added to the maestro view. For more information, see Chapter 23, “Working with Documents and Datasheets.”
Setup State	Opens the Save Setup State form so you can save a copy of the current ADE Assembler setup database **Note:** This action does not save history data, only the setup database. For more information, see Creating or Updating a Setup State on page 538.

© 2023 Cadence Design Systems, Inc.

Create Menu Options	Description
Datasheet	Opens the Create Datasheet form so you can create a datasheet for the history item selected in the History tab of the Data View assistant pane. For more information, see Creating a Datasheet for a History on page 1430.
Spec Summary	Opens the Spec Summary form so you can view the specifications summary for the results of a simulation run. For more information, see Working with Specifications on page 1329.
Spec Comparison	Opens the Spec Comparison form so you can compare measured results of output expressions for: ■ Any two history items. ■ Any two tests in the same history item or in two different history items. ■ Any two design points in the same history item or in two different history items. For more information, see on page 1248.

Tools

The options in the *Tools* menu are described below:

Tools Menu Options	Description
Calculator	Opens the Calculator window
Results Browser	Opens the Results Browser window
Job Monitor	Opens the Job Monitor

© 2023 Cadence Design Systems, Inc.

Options

The options in the *Options* menu are described below:

Options Menu Options	Description
Job Setup	Opens the Job Policy Setup form
Run Options	Opens the Run Options form
Save	Opens the Save Options form
Outputs Formatting	Opens the Default Formatting Options form
Plotting/Printing	Opens the ADE Assembler Plotting/Printing Options form

Run

The options in the *Run* menu are described below:

Run Menu Options	Description
Single Run, Sweeps and Corners	Run simulations across multiple tests with sweeps and corners analyses.
Monte Carlo Sampling	Runs a simulation using the Monte Carlo method and sampling options. For more information, see Monte Carlo Analysis.
Global Optimization	Perform global optimization using a parallel simulated annealing algorithm For more information, see Running a Global Optimization
Local Optimization	Perform local optimization For more information, see Running a Local Optimization.
Improve Yield (VVO feature)	Performs an optimization run to improve the yield of your design. For more information, see the *Virtuoso Variation Option User Guide*.

© 2023 Cadence Design Systems, Inc.

Run Menu Options	Description
High Yield Estimation (VVO feature)	Uses worst case distance algorithm to estimate yield for circuits with high yields. Decreases the number of simulations required to estimate yield when compared with Monte Carlo simulation. For more information, see High Yield Estimation in the *Virtuoso Variation Option User Guide*.
Sensitivity Analysis	Performs sensitivity analysis. For more information, see Sensitivity Analysis.
Worst Case Corners	Uses sensitivity analysis to identify worst case corners for each design specification and adds those corners to the Corners Setup form. For more information, see Worst Case Corners.
Manual Tuning	Lets you tune your design by varying the values of parameters, running multiple simulations, and then comparing results. For more information, see Performing Manual Tuning
Run Plan	Runs the run plan. For more information, see Working with Run Plan
Size Over Corners	Optimizes test benches over a large number of corners. For more information, see Sizing Over Corners
Stop Simulation	Stops the simulation run.
Stop All Jobs	Stops all jobs you started during the current session regardless of their current state (started, getting configured, running). *Important* Use this option with extreme caution.

© 2023 Cadence Design Systems, Inc.

EAD

The EAD menu helps to enable the Electrically Aware Design flow in Virtuoso and configure settings to save the electrical data.. The options available in this menu are described below:

Options Menu Options	Description
Setup	Displays the EAD Setup form. For more information, see Preparing the EAD Setup for Simulation in *Virtuoso Electrically Aware Design Flow Guide*.
Signal Selection	Opens the design schematic in a new tab and displays the Parasitics & Electrical Setup assistant to enable selection of signals for the EAD flow. For more information, see Selecting Signals to Save Currents in *Virtuoso Electrically Aware Design Flow Guide*.

Note: To be able to save the electrical data while running simulations, you require Virtuoso_Layout_Suite_EAD (95600) or Virtuoso_Variation_Analysis_Op (95510) license.

Parasitics/LDE

The *Parasitics/LDE* menu lets you investigate the effects of parasitics and Layout Dependent Effects (LDEs) on your circuits. You can report on parasitics that exist in your design, show or hide them on your design, and create refined extracted cellviews.

For more information, see the following links:

- *Virtuoso Parasitic Aware Design User Guide*

Note: This feature can be used only with the Virtuoso_Variation_Option license.

Window

The options in the *Window* menu are described below:

Options Menu Options	Description
Welcome to ADE Assembler	Displays the Welcome to ADE Assembler start page. For more information, see The Welcome to ADE Assembler Page on page 50.

© 2023 Cadence Design Systems, Inc.

Options Menu Options	Description
Assistants	Displays or hides the selected assistant pane For more information about assistant panes, see the *Virtuoso Design Environment User Guide*.
Workspaces	Displays the selected workspace For more information about workspaces, see "Getting Started with Workspaces" in the *Virtuoso Design Environment User Guide*.
Toolbars	Displays or hides the selected toolbar You can also customize toolbars to regroup the commands as required. For more information about customizing toolbars, see the *Virtuoso Design Environment User Guide*.
Tabs	Allows you to work with tabs in the session window For more information about working with tabs, see the *Virtuoso Design Environment User Guide*.

Help

The options in the *Help* menu for ADE Assembler.

Context-Sensitive Menus

ADE Assembler provides context-sensitive menus for various UI elements to provide a quick access to the commands relevant to those element. For example, the context-sensitive menu for a test name listed in the Data View assistant provides all the commands that can be used to view or modify the contents for that test, or to modify its attributes.

Similarly, the context-sensitive menu for the Outputs Setup tab provides commands to add a new output or to modify the properties of an existing output.

Customizing the Context-Sensitive Menus

ADE Assembler provides you the flexibility to modify the context-sensitive menus to keep only the commands you use or to change their order to keep the most used commands on top.

Note: Currently, you can modify the context-sensitive menus for the following UI elements only:

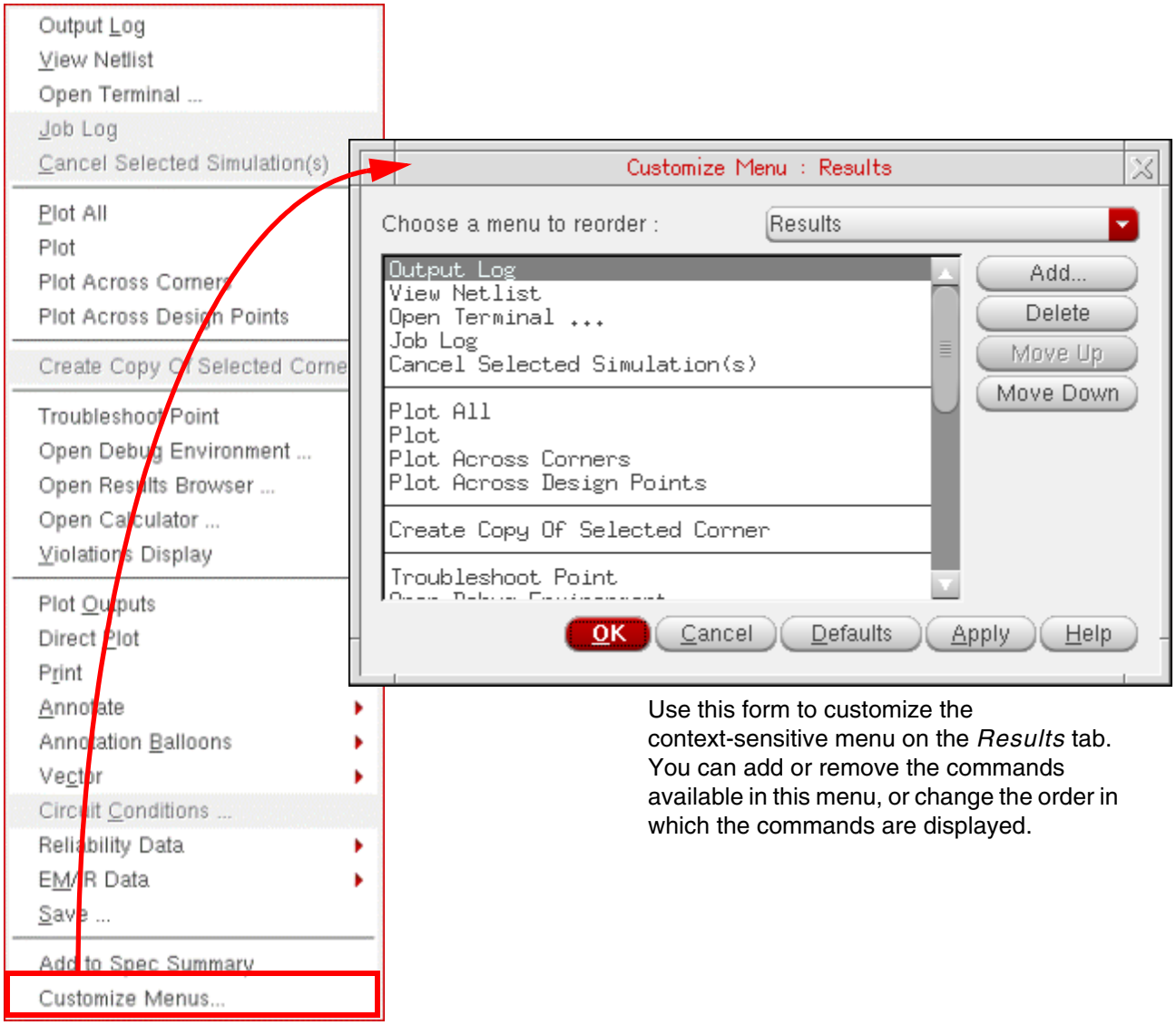
© 2023 Cadence Design Systems, Inc.

- A test in the Data View assistant
- The *Outputs Setup* tab
- The Detail results view
- The Detail - Transpose results view

To customize the context-sensitive menu for any of the elements mentioned above, you can use the *Customize Menus* command given at the bottom of that menu.

© 2023 Cadence Design Systems, Inc.

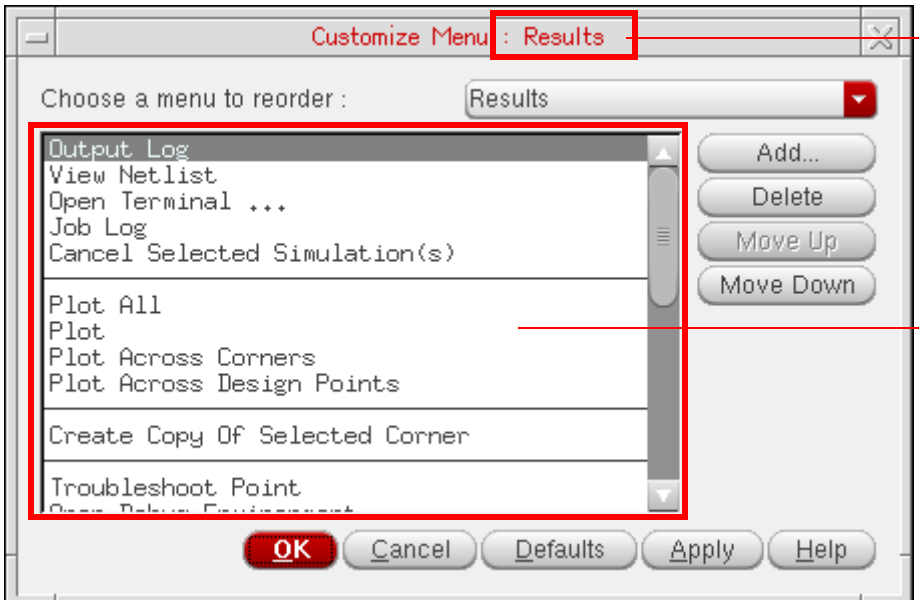
For example, to customize the context-sensitive menu for the *Results* tab, right-click anywhere on this pane and choose *Customize Menus*. The Customize Menu form is displayed, as shown below.



Use this form to customize the context-sensitive menu on the *Results* tab. You can add or remove the commands available in this menu, or change the order in which the commands are displayed.

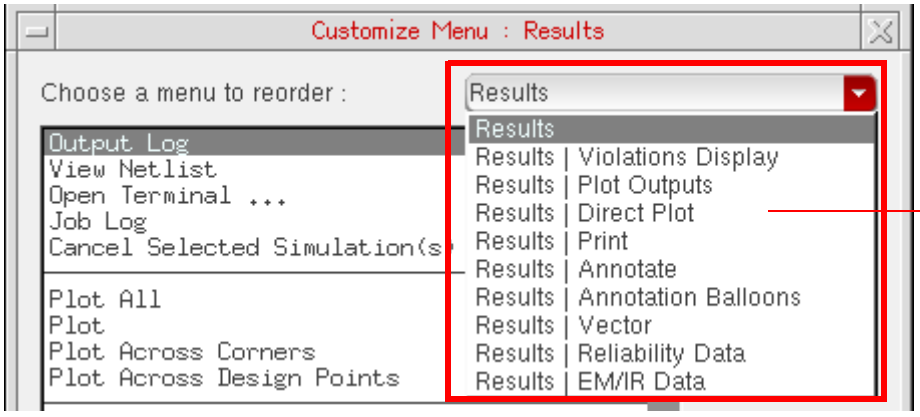
© 2023 Cadence Design Systems, Inc.

Observe the various components of the Customize Menu form.



You would notice the following points:

- The form name dynamically displays the name of the UI element for which you have chosen to customize the menu. In the example shown above, the form name includes *Results*, which indicates that the context-sensitive menu is for the *Results* tab.
- The form displays all the commands that are currently available in the menu being customized. The commands are ordered in the same way as they are available in the menu.
- A menu can contain various submenus. You can view and modify the commands available in the submenus by selecting the submenu name from the *Choose a menu to reorder* drop-down list. The contents of this list are also specific to the context-sensitive menu being customized.



© 2023 Cadence Design Systems, Inc.

Alternatively, you can double-click the submenu name in the list of commands on the Customize Menu form to view its commands.

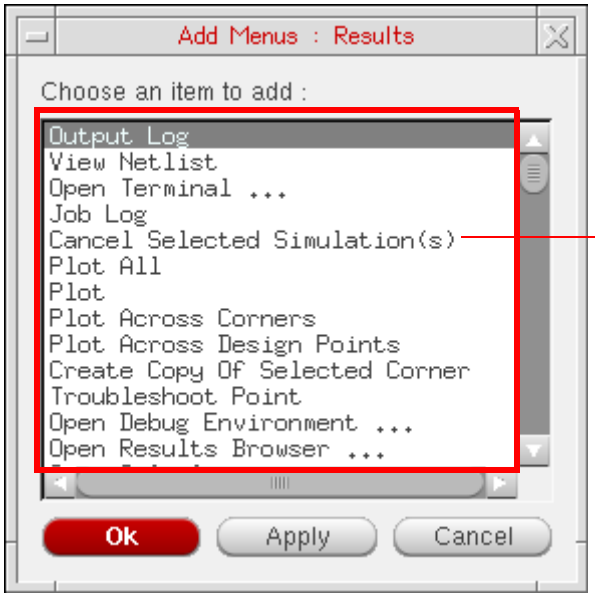
To customize a context-sensitive menu or its submenu, display the list of commands of that menu and do one or more of the following:

- Add a command
- Remove a command
- Change the order of commands

After customizing a context-sensitive menu, click *OK* on the Customize Menu form to save the changes.

Adding a Command

To add a command, click *Add* on the Customize Menu form. The Add Menus form is displayed.



The *Choose an item to add* list displays all the commands that can be added to the menu being customized. Select the name of the command you want to add and click *Apply*. If required, you can resize the form vertically to view the complete list of available commands.

Note: At the end of the list, a separator is also available. You can select this to add a separator in the menu.

© 2023 Cadence Design Systems, Inc.

Click *OK* to close the form. The added commands are now visible in the list of commands on the Customize Menu form.

At any time, you can click *Defaults* to ignore all the customization done and restore the default menu setup set in ADE Assembler.

If your custom menus do not appear as expected when added through the Customize Menu form, the customization file could be corrupted. In that case, delete the menu customization files that are saved in the `.cadence/RMBCustomization` directory and are named after the tab or context menu you have modified. For example, Data View - Tests, Results, or Outputs Setup. However, this will also delete any other menu customizations for that menu and the default settings will be used.

Important

ADE Assembler has a pre-defined set of commands that can be included in a context-sensitive menu for a UI element. These commands are available in the *Choose an item to add* list on the Add Menus form. In addition to these commands, if you want to include any custom command, you can add that to the relevant `.menus` file saved at the `<Virtuoso-installation-dir>/tools/dfII/etc/tools/menus/` path. After that, the command will be available at the bottom of the menu and can be rearranged as required.

Removing a Command

To remove a command from the menu, select the name of that command in the list of available commands. Click *Delete*. The command is removed from the list of commands visible in the menu. It is also possible to remove a separator from the menu.

The command once removed will still remain available in the Add Menus form and can be added back when required.

Changing the Order of Commands

To rearrange the commands in the menu, select a command in the list of available commands. Click *Move Up* or *Move Down*, as required, to move the command to some other place in the list.

Note: For each customized menu, the settings are saved in a file in the `.cadence/RMBcustomization/` directory. The name of the file is same as that of the corresponding pane. For example, the customized context-sensitive menu for a test in the Data View assistant is saved in a file named `Data View - Tests`. The path where to save

© 2023 Cadence Design Systems, Inc.

this setting file is decided by `setup.loc` file, which is an ASCII file that specifies the locations to be searched and the order in which they should be searched. For more information about the setup.loc file or how to edit the search order, see Cadence Setup Search File: setup.loc in the *Cadence Application Infrastructure User Guide*.

If you delete a settings file from the `.cadence/RMBcustomization/` directory, the corresponding context-sensitive menu takes the default settings.

Note: It is recommended not to customize a context-sensitive menu by directly editing the settings file because an incorrect XML tag or menu name can result into an incorrect menu.

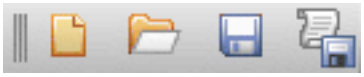
© 2023 Cadence Design Systems, Inc.

Toolbars

ADE Assembler has the following toolbars:

- File
- Create
- Browse
- Go
- Parasitic Mode
- Reference History
- Run
- Setup
- Tools
- Workspaces

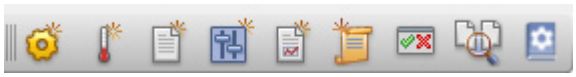
File



Icon	Name	Description
	New	Opens the New File form
	Open	Opens the Open File form
	Save	Saves your ADE Assembler cellview
	Save Script As	Opens the Save OCEAN Script form so you can save simulation setup and conditions to an OCEAN script file.

© 2023 Cadence Design Systems, Inc.

Create



Icon	Name	Description
	Create Test	Opens the ADE Explorer so you can add a new test For more information, see Chapter 2, “Specifying Tests and Analyses.”
	Create Corner	Opens the Corners Setup form so you can setup corners for simulation For more information, see Chapter 6, “Working with Corners”
	Create Document	Opens the Choose Documents to be Added form so you can select the documents to be added to the ADE Assembler view. For more information, see Chapter 23, “Working with Documents and Datasheets.”
	Create Setup State	Opens the Save Setup State form so you can save a copy of the current ADE Assembler setup database **Note:** This action does not save history data, only the setup database. For more information, see Creating or Updating a Setup State on page 538.
	Create Datasheet	Opens the Create Datasheet form so you can create a datasheet for the history item selected in the History tab of the Data View assistant pane For more information, see Creating a Datasheet for a History on page 1430.
	Create Spec Summary	Opens the Spec Summary form so you can view the specifications summary for the results of a simulation run. For more information, see Working with Specifications on page 1329.

© 2023 Cadence Design Systems, Inc.

Icon	Name	Description
	Open Setup Library Assistant	Opens the Setup Library assistant that you can use to work with setup libraries in ADE Assembler. For more information, see Working with the Setup Library Assistant.
	Create New Run	Opens the Run Plan assistant that provides the capability to create multiple variations of the setup within a single session. For more information, see Working with Run Plan.
	Create Spec Comparison	Opens the Spec Comparison form so you can compare measured results of output expressions for: ■ Any two history items ■ Any two tests in the same history item or in two different history items ■ Any two design points in the same history item or in two different history items For more information, see Comparing Results Using the Spec Comparison Form.

Browse

Icon	Name	Description
	Home	Displays the Welcome to ADE Assembler start page. For more information, see The Welcome to ADE Assembler Page on page 50.

Go

The Go toolbar allows you to do the following:

© 2023 Cadence Design Systems, Inc.

- Sequentially or non-sequentially navigate through a cellview hierarchy.
- Navigate between cells and views in various designs.

For more information about the Go toolbar, see the *Virtuoso Design Environment User Guide*.

Parasitic Mode

For more information about the Parasitics Mode toolbar, see the *Virtuoso Parasitic Estimation and Analysis User Guide*.

Reference History

Icon	Name	Description
	Reference drop-down list	Lets you select the reference history item for incremental simulation runs. For more information, see Running an Incremental Simulation to Reuse Results on page 1011.
	Show Differences	Opens the Comparison: Active Setup v/s Reference History form.
	Reference History Options	Opens the Reference History form.

© 2023 Cadence Design Systems, Inc.

Run

Icon	Name	Description
	Select a Run Mode	Specifies the run mode for simulation For more information, see Specifying the Run Mode.
	Simulation Options	Opens a form for specifying the simulation options for the run mode selected in the *Select a Run Mode* drop-down list.
	Run Simulation	Runs the simulation for the run mode selected in the *Select a Run Mode* drop-down list. You can observe variants of this command in the following situations: ■ : When the Manual Tuning run mode starts ■ : When the setup uses reference histories
	Stop Simulation	Stops the simulation run

Setup

Icon	Name	Description
	Load	Opens the Load Setup State form so you can load an existing setup state. For more information, see Loading a Setup State on page 540.

© 2023 Cadence Design Systems, Inc.

Icon	Name	Description
	Save Setup	Opens the Save Setup State form so you can save a copy of the current ADE Assembler setup database **Note:** This action does not save history data, only the setup database. For more information, see Loading a Setup State on page 540.

Tools

Icon	Name	Description
	Calculator	Opens the Calculator window
	Results Browser	Opens the Results Browser window
	Job Monitor	Opens the Job Monitor
	Remove .tmpADEDir	Deletes the `.tmpADEDir` directory.

Workspaces

The Workspaces toolbar allows you work with workspaces. For more information, see "Getting Started with Workspaces" in the *Virtuoso Design Environment User Guide*.

© 2023 Cadence Design Systems, Inc.

Assistant Panes

The following assistant panes are available in ADE Assembler:

- Variables and Parameters
- Run Summary
- Setup DB Viewer
- Data View
- Variable Display
- Fault Setup assistant
- Setup Library assistant
- Run Plan
- Parasitic Filters
- Parasitics and Electrical Setup
- Compare Parasitics Report

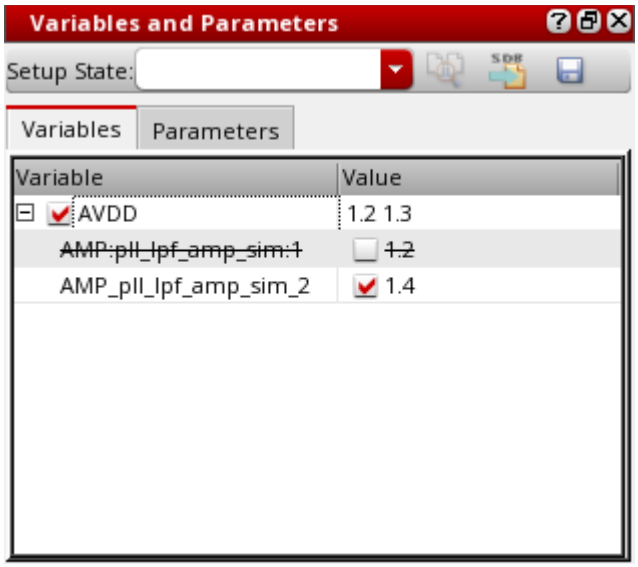
Variables and Parameters

The Variables and Parameters assistant pane allows you to view and set up design variables, global variables and parameters.

To display the Variables and Parameters assistant pane, do the following:

© 2023 Cadence Design Systems, Inc.

➤ Choose *Window – Assistant – Variables and Parameters*.



For information about using the Variables and Parameters assistant pane, see the following:

To use the Variables and Parameters pane to work with	See
Design variables, global variables and instance parameters	Chapter 4, “Working with Global Variables” Chapter 3, “Working with Design Variables and Instance Parameters”
Matched parameter constraints	Chapter 5, “Working with Constraints”
Parasitic parameters	*Virtuoso Parasitic Estimation and Analysis User Guide*. **Note:** The Parasitics tab is displayed only if your design has swept parasitic parameters.

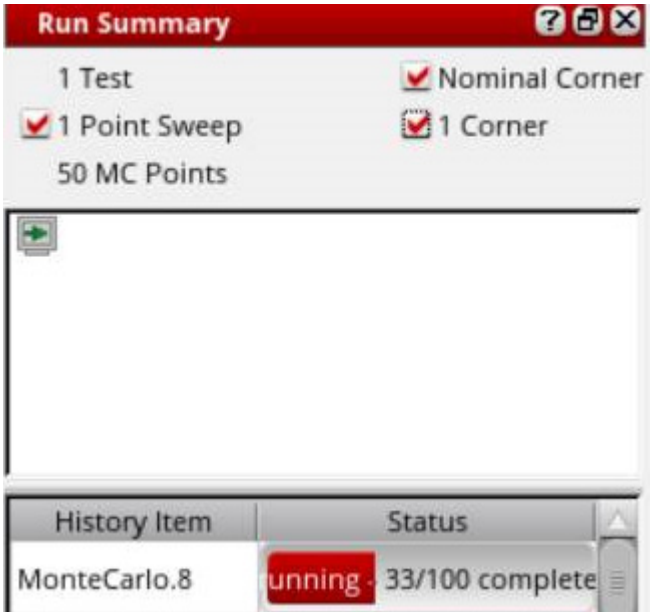
Run Summary

The Run Summary assistant pane displays a summary of the simulation setup and the status of simulation runs.

To display the Run Summary assistant pane, do the following:

© 2023 Cadence Design Systems, Inc.

➤ Choose *Window – Assistant – Run Summary*.



For information about using the Run Summary assistant pane, see the following:

- Disabling and Enabling All Point Sweeps
- Viewing Job Status on page 994

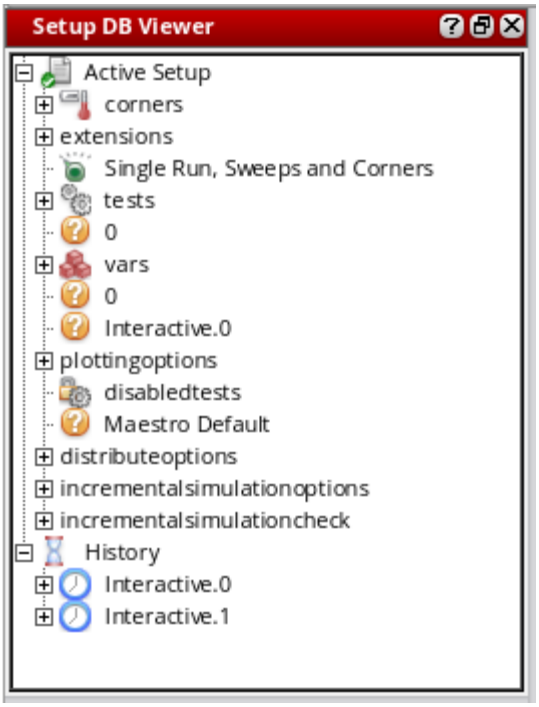
Setup DB Viewer

The Setup DB Viewer assistant pane provides a graphical view of the ADE Assembler setup database.

To display the Setup DB Viewer assistant pane, do the following:

© 2023 Cadence Design Systems, Inc.

➤ Choose *Window – Assistant – Setup DB Viewer*.



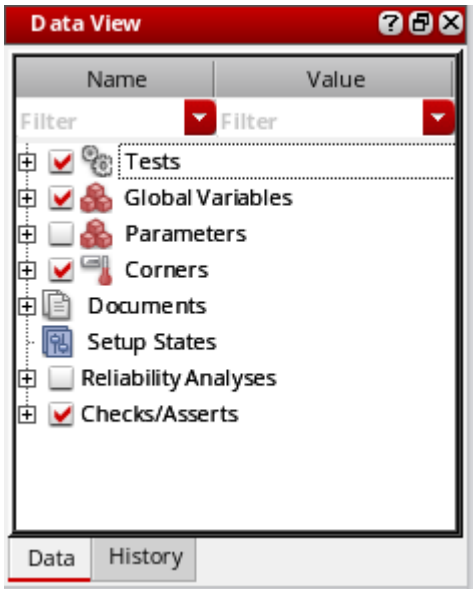
For information about using the Setup DB Viewer assistant pane, see Viewing Histories on page 1256.

Data View

The Data View assistant provides a single user interface to quickly view and set up commonly used setup data.

© 2023 Cadence Design Systems, Inc.

The *Data* tab on the Data View assistant pane helps you quickly view and set up tests, global variables, parameters, corners, device checks and asserts, and manage documents and setup states.



The data tab contains the following two columns:

- *Name*: It includes the names of the setup elements, such as variable, parameter, corner, and so on.
- *Value*: It includes the values of the elements defined in the setup.

These columns are separated through a divider bar that you can drag to change the width of the columns. The width of the divider bar can also be controlled by the `nameDisplayWidthInDataView` .cdsenv variable.

These columns provide search filters that you can use to filter the data based on the specified criteria. You can configure and use these filters in the same way as done for the Setup Assistant in ADE Explorer. For more details, refer to Filtering Data in the Setup Assistant in *Virtuoso ADE Explorer User Guide*.

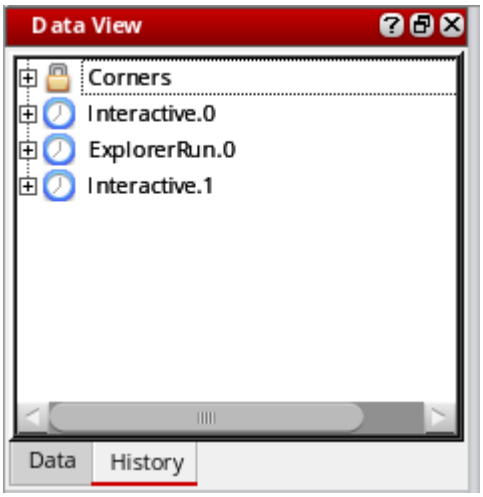
Related environment variables:

- `autoExpandSetupStatesSubTree`
- `showFilterInDataView`
- `nameDisplayWidthInDataView`

© 2023 Cadence Design Systems, Inc.

Note: Read a blog related to this at Virtuosity: Handy UI Enhancements in ADE Assembler & ADE Explorer.

The *History* tab on the Data View assistant pane allows you to work with history items.



For information about using the Data View assistant pane, see the following:

To use the Data View assistant pane to work with	See
Tests	Chapter 2, “Specifying Tests and Analyses”
Design variables, global variables and instance parameters	Chapter 4, “Working with Global Variables” Chapter 3, “Working with Design Variables and Instance Parameters”
Matched parameter constraints	Chapter 5, “Working with Constraints”
Corners	Chapter 6, “Working with Corners”
History Items	Chapter 20, “Working with History Checkpoints”
Documents	Chapter 23, “Working with Documents and Datasheets”
Setup States	Chapter 12, “Working with the Simulation Setup”

© 2023 Cadence Design Systems, Inc.

Fault Setup

The fault Setup assistant allows you to define a set of faults that you want to consider while running the fault simulation.

For more details, refer to Working with the Setup Library Assistant.

Setup Library

The Setup Library assistant helps you define a project-specific (master) setup for the design blocks in ADE Assembler and then reuse the specified setup to create or modify the current simulation setup in the Data View assistant. The setup in this assistant is generic and can be applied on any design.

For more details, refer to Working with the Setup Library Assistant.

Run Plan

The Run Plan assistant allows you to create multiple variations of the setup within a single session, where each run has its own setup details that override the settings in the active setup. Simulations can be run for all the runs in a session together with a single click and their results are generated simultaneously.

For more details, refer to Working with Run Plan.

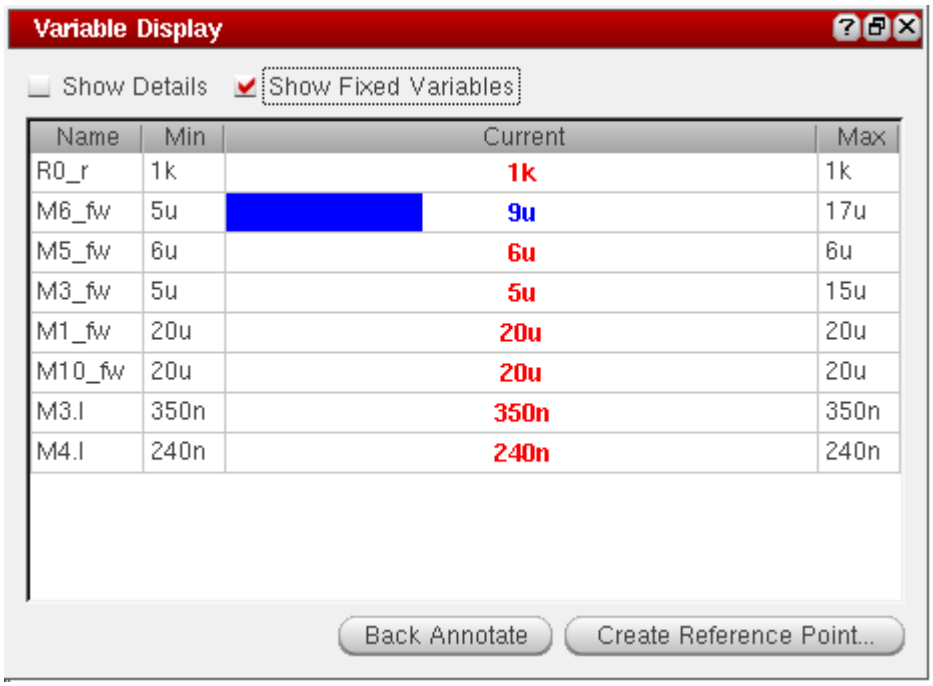
Variable Display

The Variable Display assistant pane displays the current, minimum and maximum value set for each variable and parameter for the selected design point.

To display the Variable Display assistant pane, do the following:

© 2023 Cadence Design Systems, Inc.

➤ Choose *Window – Assistant – Variable Display*.



For more information about the Variable Display assistant pane, see Showing Variable and Parameter Values for Design Points on the Results Tab on page 1239.

Parasitic Filters

For more information about the Parasitic Filters assistant pane, see the *Virtuoso Parasitic Estimation and Analysis User Guide*.

To display the Parasitic Estimates assistant pane, do the following:

➤ Choose *Window – Assistant – Parasitic Filters*.

Parasitics and Electrical Setup

For more information about this assistant pane, see the *Virtuoso Parasitic Aware Design User Guide* and *Virtuoso Electrically Aware Design Flow Guide*.

Compare Parasitics Report

For more information about the Parasitics Report assistant pane, see the *Virtuoso Parasitic Estimation and Analysis User Guide*.

© 2023 Cadence Design Systems, Inc.

To display the Compare Parasitics Report assistant pane, do the following:

➤ Choose *Window – Assistant – Parasitics Report*.

Outputs

The Outputs pane allows you to setup simulation outputs and view results and diagnostics information. This pane contains the following four tabs:

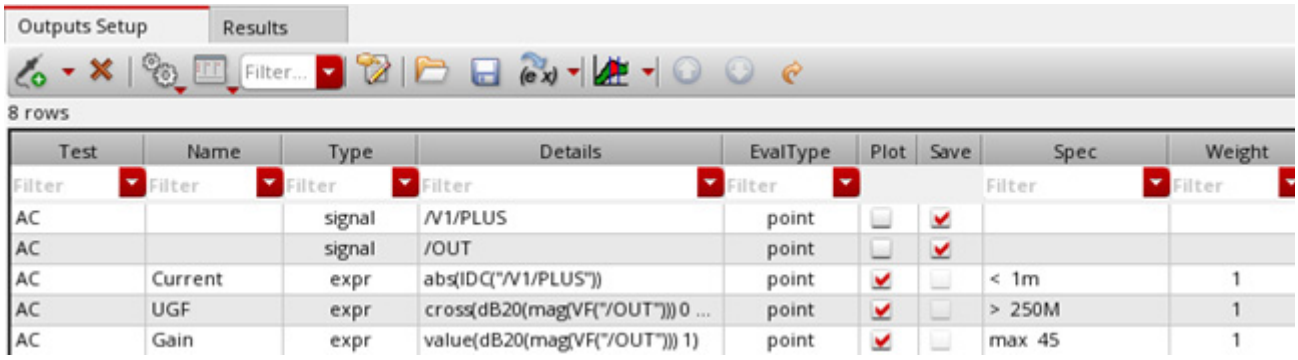
- Outputs Setup
- Run Preview
- Results

Tip

You can use the `Ctrl + Page Up` or `Ctrl + Page Down` keys to switch between the tabs on the Outputs pane.

Outputs Setup

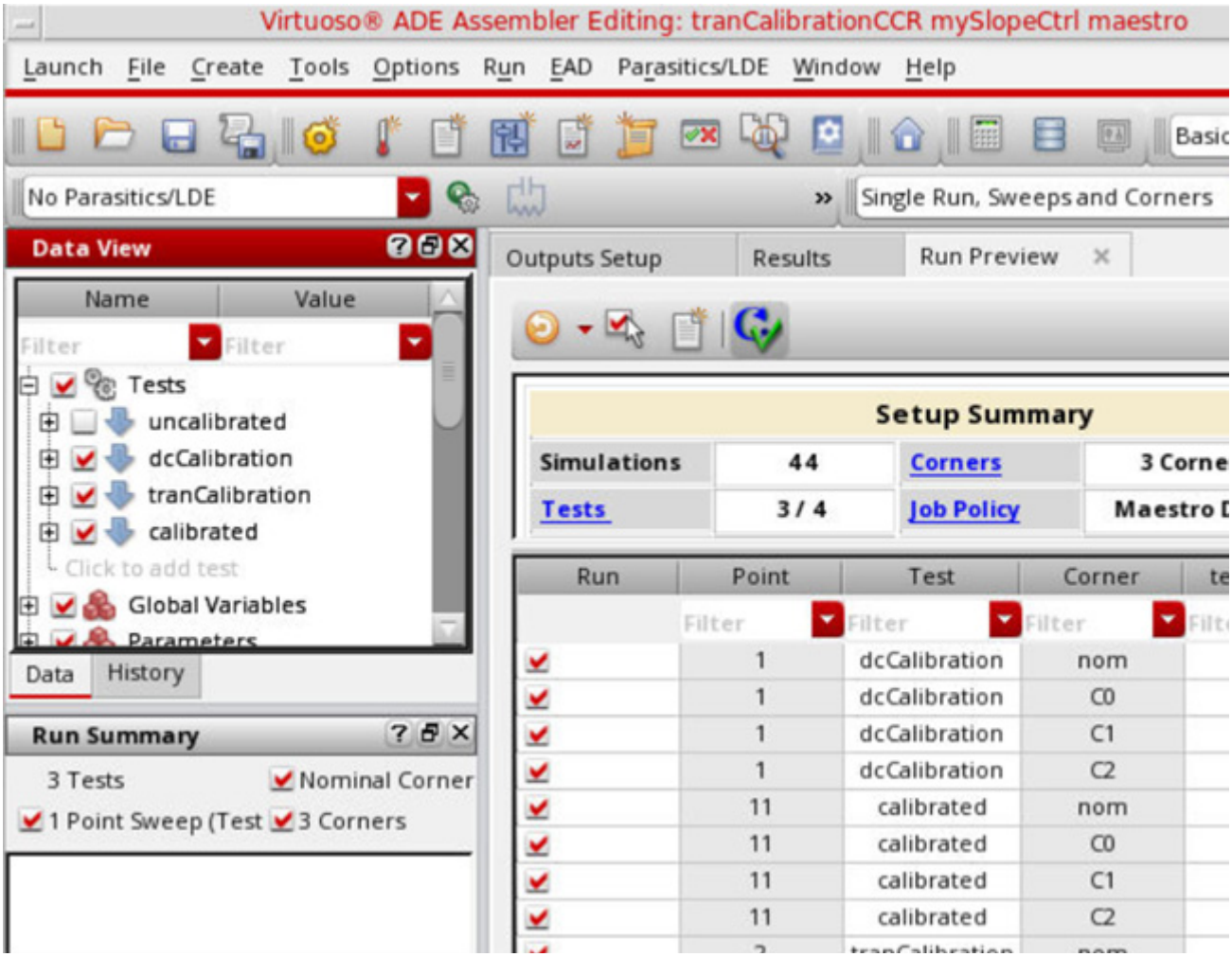
The Outputs Setup tab of the Outputs pane allows you to specify nets, terminals, and measurements you want to save and plot. For more information about using the Outputs Setup tab, see Chapter 10, “Selecting Data to Save and Plot.”



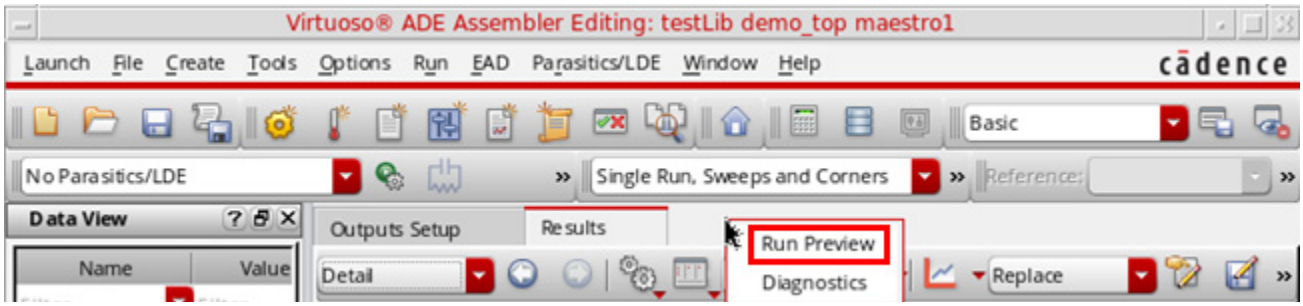
© 2023 Cadence Design Systems, Inc.

Run Preview

The Run Preview tab provides a summarized view of all the settings applied in the current ADE Assembler view.



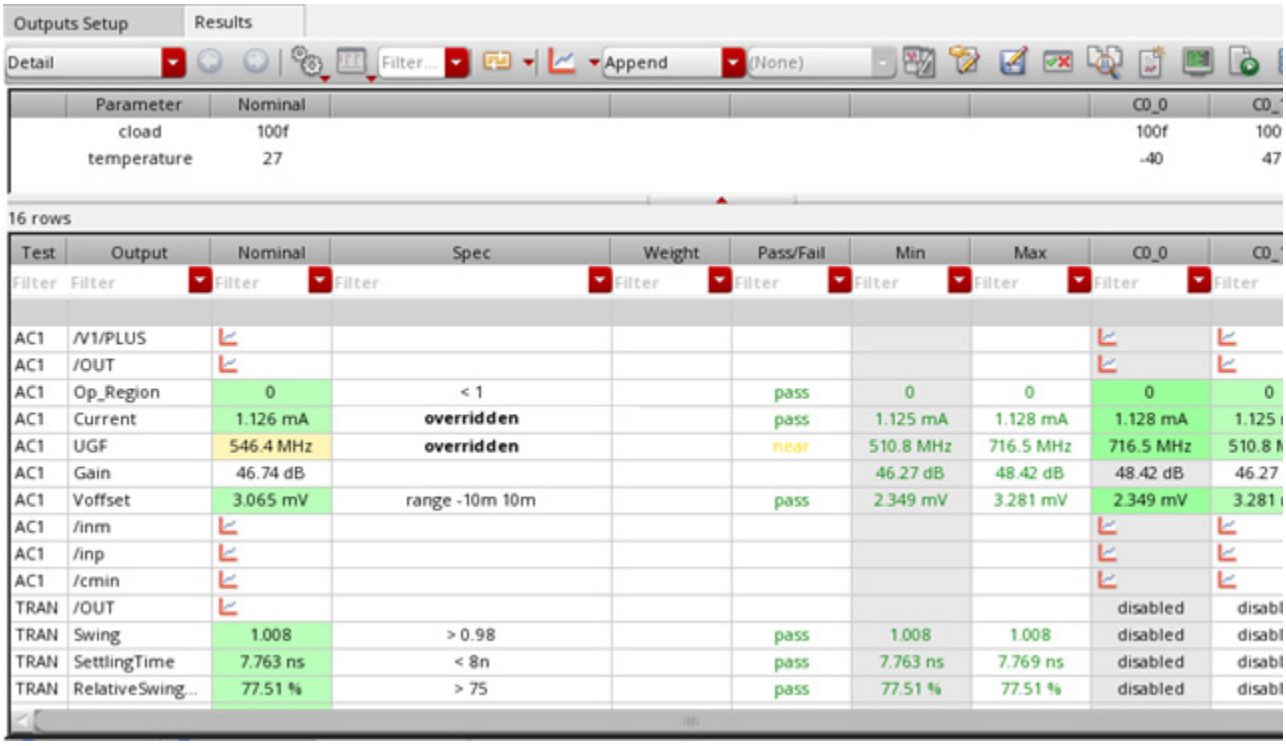
By default, this tab is not visible. To view it, right-click next to the *Results* tab title and choose *Run Preview*.



© 2023 Cadence Design Systems, Inc.

Results

The Results tab of the Outputs pane allows you to view simulation results. For more information about using the Results tab, see Chapter 19, “Viewing, Printing, and Plotting Results.”



Workspaces

The following workspaces are available in ADE Assembler:

- Basic Workspace
- Outputs Workspace

For more information about workspaces, see “Getting Started with Workspaces” in the *Virtuoso Design Environment User Guide*.

Basic Workspace

The following panes appear in the Basic workspace:

© 2023 Cadence Design Systems, Inc.

- Data View
- Run Summary
- Outputs

Outputs Workspace

The Outputs workspace displays only the Outputs pane.

Specifying the Run Mode

A run mode specifies the type of simulation you want to run in ADE Assembler.

You can choose a run mode to be used for simulation from the Run menu. Alternatively, you can select a run mode in the *Select a Run Mode* drop-down list.

Simulators Supported for Run Modes

The following simulators are supported for the Single Run, Sweeps and Corners, Global Optimization, Local Optimization, Sensitivity Analysis, Feasibility Analysis and Size Over Corners run modes:

- *spectre* (Spectre circuit simulator)
- *aps* (Spectre Accelerated Parallel simulator)
- *UltraSim* (Virtuoso UltraSim simulator)
- *hspiceD* (HSPICE simulator from Synopsys, Inc)
- *ams* (Spectre AMS Designer Simulator)

The following simulators are supported for the Monte Carlo Sampling run mode:

- *spectre* (Spectre Circuit Simulator)
- *aps* (Spectre Accelerated Parallel simulator)
- *hspiceD* (HSPICE simulator from Synopsys, Inc)
- *ams* (Spectre AMS Designer simulator with Spectre or APS as the solver.

  Note the following:

© 2023 Cadence Design Systems, Inc.

- Ensure that you are using the Cadence IUS 9.2 or later version of Spectre AMS Designer.
- You cannot use Spectre AMS Designer Simulator with UltraSim as the solver to run Monte Carlo analysis.
- Running Monte Carlo simulation in interactive mode (using SimVision) is not supported with the AMS Designer simulator.

Note: If you have specified device checks or operating region specifications for a test, ensure that only the Spectre Circuit Simulator, Spectre Accelerated Parallel simulator, or Spectre AMS Designer Simulator (with Spectre or APS as the solver) is selected for that test. For more information about selecting the simulator for a test, see Choosing the Target Simulator on page 109.

© 2023 Cadence Design Systems, Inc.

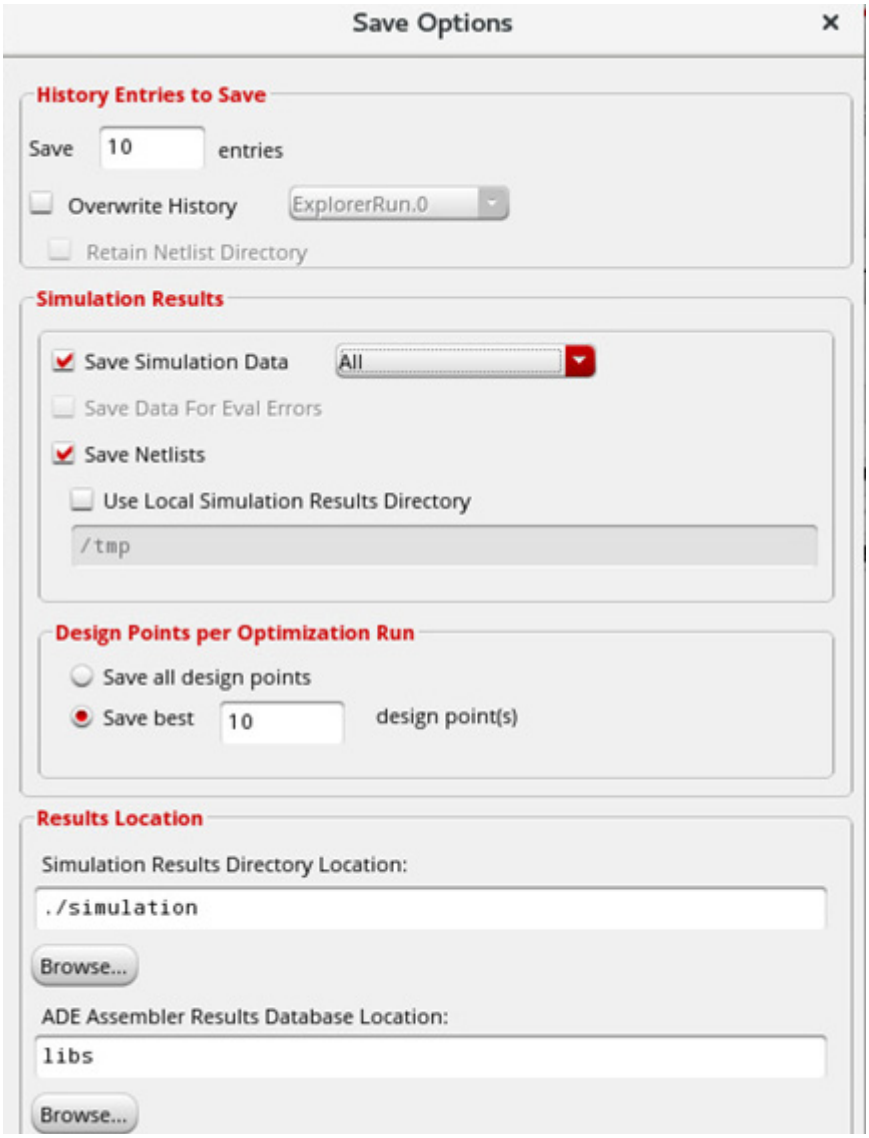
Specifying Options for Saving Simulation Results

By default, the simulation results information is saved to *libraryName*/*cellName*/maestro/results/data/*history_item* in the location specified using the asimenv.startup projectDir environment variable. The default setting for this environment variable is $HOME/simulation.

To specify the options for saving simulation results and a different location where you want the program to save the simulation results, do the following:

1. In the ADE Assembler session window, choose *Options – Save*.

   The Save Options form appears.



© 2023 Cadence Design Systems, Inc.

2. In the *Simulation Results* group box, specify what all simulation data must be saved.

   By default, *Save Simulation Data* and *Save Netlist* options are selected. The default selection saves the simulation results and the netlist data are saved for all the design points.

   - Select one of the following options from the *Save Simulation Data* drop-down list:
     - *All*: saves all the simulation results data
     - *Quick Plot Data Only*: saves only the quick plot data
   - To save only the simulation data for points with evaluation errors, select *Save Data for Eval Errors*. Alternatively, set the environment variable `saveEvaluationErrorResults` to `t`.

     This check box can be used only when either of the following conditions is met.

     - *Save Simulation Data* is deselected manually.
     - The specified run mode is Monte Carlo Sampling.

       In this case, *Save Simulation Data* is deselected automatically. Monte Carlo mode lets you use the *Save Waveforms (Simulation Data)* check box in the Monte Carlo form to control the simulation data to be saved.
   - To save only the simulation results, but not the netlist for each design point, deselect *Save Netlists* and select only the *Save Simulation Data* option. When this option is selected, the point-specific netlist data is not saved and the netlist directory for every point is replaced with a symbolic link to the reference netlist directory. This combination of save options is useful in saving space when the netlist for each design point is large and space-consuming.

3. By default, the results for distributed simulation runs are saved in the location specified in the *Simulation Results Directory Location* field. To save the results for distributed simulation jobs run on a remote system in a local directory on that system, select the *Use Local Simulation Results Directory* check box and specify the path to the local directory.

   Note the following:

   - You must ensure that the specified local directory path exists on all the remote systems on which a distributed simulation is run.
   - If this option is set, you can view the simulation results in the Results tab. However, you cannot plot the results, re-evaluate expressions, or annotate simulation results to the schematic.

© 2023 Cadence Design Systems, Inc.

- The results saved on remote systems will not be deleted if you delete the corresponding history items. You must manually delete the directories containing the results on remote systems.

Tip

Setting this option improves distributed simulation performance because the results are saved on the remote systems on which the jobs are run instead of saving them over the network.

4. In the *Simulation Results Directory Location* field, specify the directory path where you want the program to write your simulation results information.

   If your design library is set up as read only, you can use this field to specify a writable location.

5. Click *OK*.

   The program writes simulation results information to *libraryName*/*cellName*/maestro/results/data/<*history_item*> in the specified directory.

   If you do not specify a simulation results location, but specify a results database location (see Specifying Results Database Location on page 88), the program writes this information to *libraryName*/*cellName*/maestro/results/data/<*history_item*> in the results database location.

Note: You can also specify the simulation results location using the `adexl.results saveResDir` environment variable.

See also

- Directory Structure of a Maestro Cellview
- Specifying Results Database Location on page 88
- Specifying How Many Histories to Save on page 1250
- Overwriting a History Item during Subsequent Simulation Runs on page 1252
- Working with Read-Only or Editable maestro Views on page 524

© 2023 Cadence Design Systems, Inc.

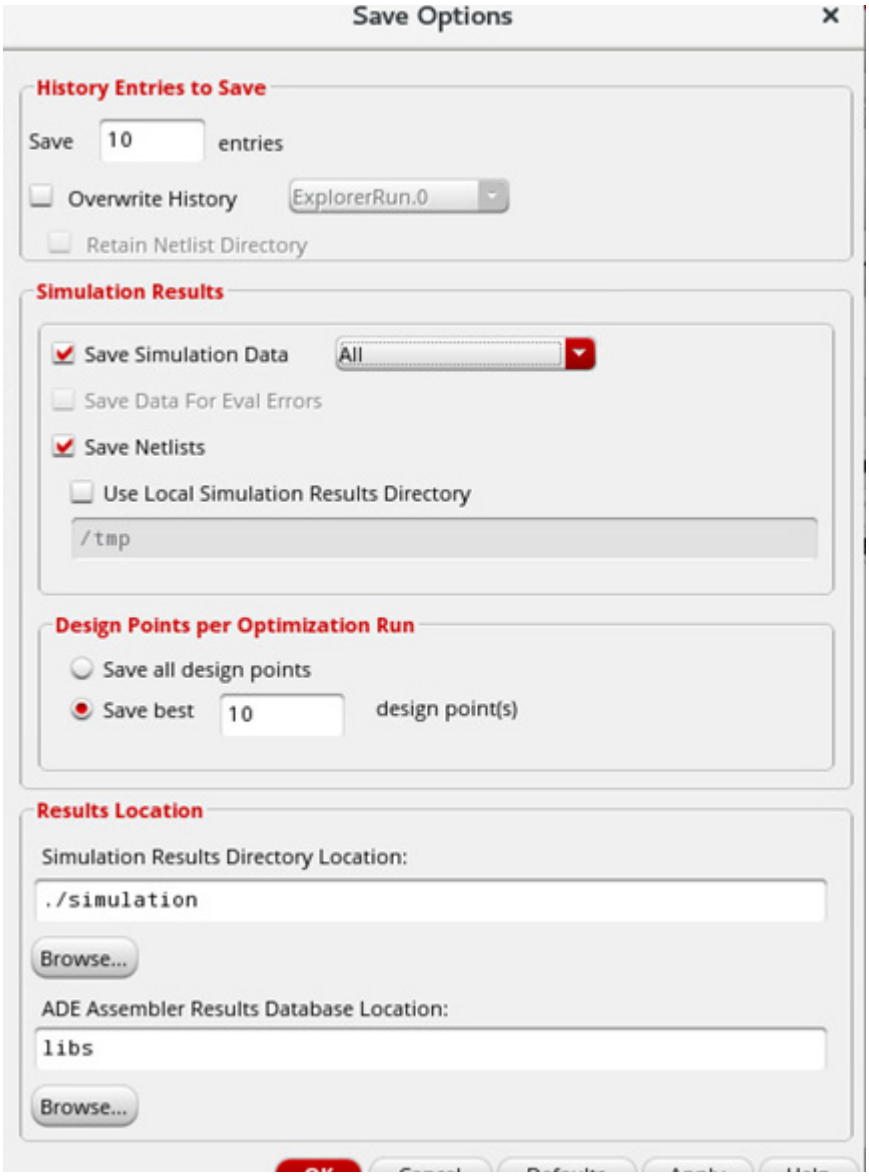
Specifying Results Database Location

By default, the program writes the results database and run log files to *libraryName*/*cellName*/*viewName*/results/data in the ADE Assembler view. For more information about the maestro view, see Directory Structure of a Maestro Cellview.

To specify a different location where you want the program to save the results database and run log files, do the following:

1. In the ADE Assembler session window, choose *Options – Save*.

   The Save Options form appears.



© 2023 Cadence Design Systems, Inc.

2. In the *ADE Assembler Results Database Location* field, specify the directory path where you want ADE Assembler to write your results database information and run log files.

   If your design library is set up as read-only, you can use this field to specify a writable location.

3. Click *OK*.

   The program writes results database information and run log files to *libraryName*/*cellName*/*viewName*/results/data in the specified directory.

   If you do not specify a results database location, and you open the maestro view in read-only mode or you do not have write permissions in the maestro view, the program writes this information to *libraryName*/*cellName*/maestro/results/data in the location specified using the asimenv.startup projectDir environment variable. The default setting for this environment variable is $HOME/simulation. For more information, see Working with Read-Only or Editable maestro Views on page 524.

Note: You can also specify a different location for the results database and run log files using the adexl.results saveDir environment variable.

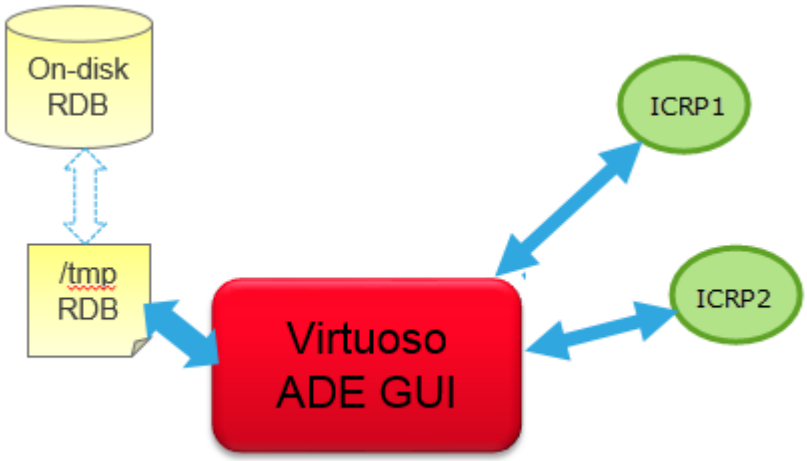
How does ADE Assembler save the results in the results database?

Earlier, ADE Assembler and the IC remote processes (ICRP) used to write to the .rdb file on the file system, which is usually an NFS disk. However, the NFS file locks, which are used to ensure database correctness, are slow. This used to impact the performance resulting in a non-responsive GUI during simulation and a lengthy simulation run time. In addition, simultaneous attempts to access the .rdb file by the ADE Assembler user interface and the ICRP increase the possibility of database corruption due to the unreliability of NFS file locking mechanism.

To avoid this, ADE Assembler now uses a local directory, named as /tmp, to temporarily save the results database. This avoids the NFS file lock issues and centralizes access to the .rdb file because it can be accessed only through the ADE Assembler UI. In addition, it uses a separate backup thread to copy the results data from the /tmp directory to the permanent results database, the location specified by the *ADE Assembler Results Database Location* field. ADE Assembler saves a .rdb file in the

© 2023 Cadence Design Systems, Inc.

libraryName/*cellName*/*viewName*/results/data directory under the specified location.



Important Points to Note

- ADE Assembler saves the .rdb file to the `/tmp` directory on the same computer where the application is running.
- You can choose to save the results database directly in the `.rdb` file on the NFS disk instead of `/tmp` directory by setting the RDBCreatePolicy environment variable to `"NFSDisk"`. By default, this variable is set to `"TmpDir"`.
- If the continueICRPRunOnAbruptGUIExit environment variable is set to `t`, ADE Assembler needs to continue the simulation run even if Virtuoso closes abruptly. In this case, it will wait for the successful copy of the data from `/tmp` to `/RDB`. While this data is being copied, Assembler checks for its successful completion status for a specific number of times specified by the `continueICRPNumRetriesToWaitForGUIRecoveredResDB` environment variable, which is by default set to `20`. If the success status is received, ADE Assembler continues the simulation run. If the success status is not received in the given number of checks, the simulation is not completed.

  If the simulation data is large and the expected time for data copy is large, you can set the `continueICRPNumRetriesToWaitForGUIRecoveredResDB` variable to a higher value.
- Only the ADE Assembler user interface can access `/tmp`. Therefore, it is not required to be locked and released for every read or write operation, and database operations are also fast.

© 2023 Cadence Design Systems, Inc.

- The thread to backup the results from the `/tmp` directory to the NFS disk runs at regular intervals of 50 seconds. This thread copies the data in small blocks. You can also specify the number of pages in each block to be copied by setting the RDBTmpDirDiskQuotaLowerLimitInMegabytes environment variables.
- It is recommended not to keep RDBTmpDirDiskQuotaLowerLimitInMegabytes large to ensure that the temporary database is accessible to ADE Assembler.
- If Virtuoso is closed abruptly due to some issue and all the data has not been copied to the results database, all the settings for block transfer and frequency of backup are ignored, and all the unsaved data is copied from the `/tmp` directory to the results database.
- The availability of disk space for both `/tmp` directory and the `.rdb` file is checked periodically. If the available disk space becomes less than the lower limit, appropriate warning messages are flagged. If these disks do not have sufficient space, the simulation is automatically suspended, the simulation is automatically suspended. You can free up the space and resume the simulation.

  Related environment variables:

  - RDBTmpDirDiskQuotaLowerLimitInMegabytes
  - RDBNFSDiskQuotaLowerLimitInMegabytes
- When you close the ADE Assembler window, it runs a check to confirm that all the data from /tmp directory has been successfully copied to the .rdb results database. If some data remains to be copied, ADE Assembler waits before closing the window, and shows a message in the CIW.

Related topics

- Directory Structure of a Maestro Cellview
- Specifying Options for Saving Simulation Results on page 85
- Specifying How Many Histories to Save on page 1250
- Overwriting a History Item during Subsequent Simulation Runs on page 1252
- Working with Read-Only or Editable maestro Views on page 524

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

2

Specifying Tests and Analyses

You can perform the following tasks from the Data tab in the Data View assistant pane in the ADE Assembler environment:

- Working with Tests on page 94
- Viewing Information about a Test on page 100
- Adding, Changing, and Removing Analyses on page 102
- Specifying a Design for a Test on page 106
- Opening a Design Schematic on page 108
- Choosing the Target Simulator on page 109
- Loading State Information on page 112
- Specifying Model Libraries on page 115
- Setting the Simulation Temperature on page 117
- Specifying Analog Stimuli on page 119
- Specifying a Custom Library of Sources on page 121
- Specifying Simulation Files on page 122
- Specifying Simulation Environment Options on page 128
- Specifying Simulator Options on page 136
- Creating and Viewing the Netlist on page 139
- Editing Multiple Tests in ADE Assembler on page 140
- Modifying a Test in ADE Explorer on page 150
- Working with OCEAN-Based Tests on page 154

© 2023 Cadence Design Systems, Inc.

Working with Tests

See the following topics:

- Adding a Test on page 95
- Renaming a Test on page 96
- Copying a Test on page 97
- Removing a Test on page 98
- Adding Notes to a Test on page 98
- Changing the Order of Tests on page 99

© 2023 Cadence Design Systems, Inc.

Adding a Test

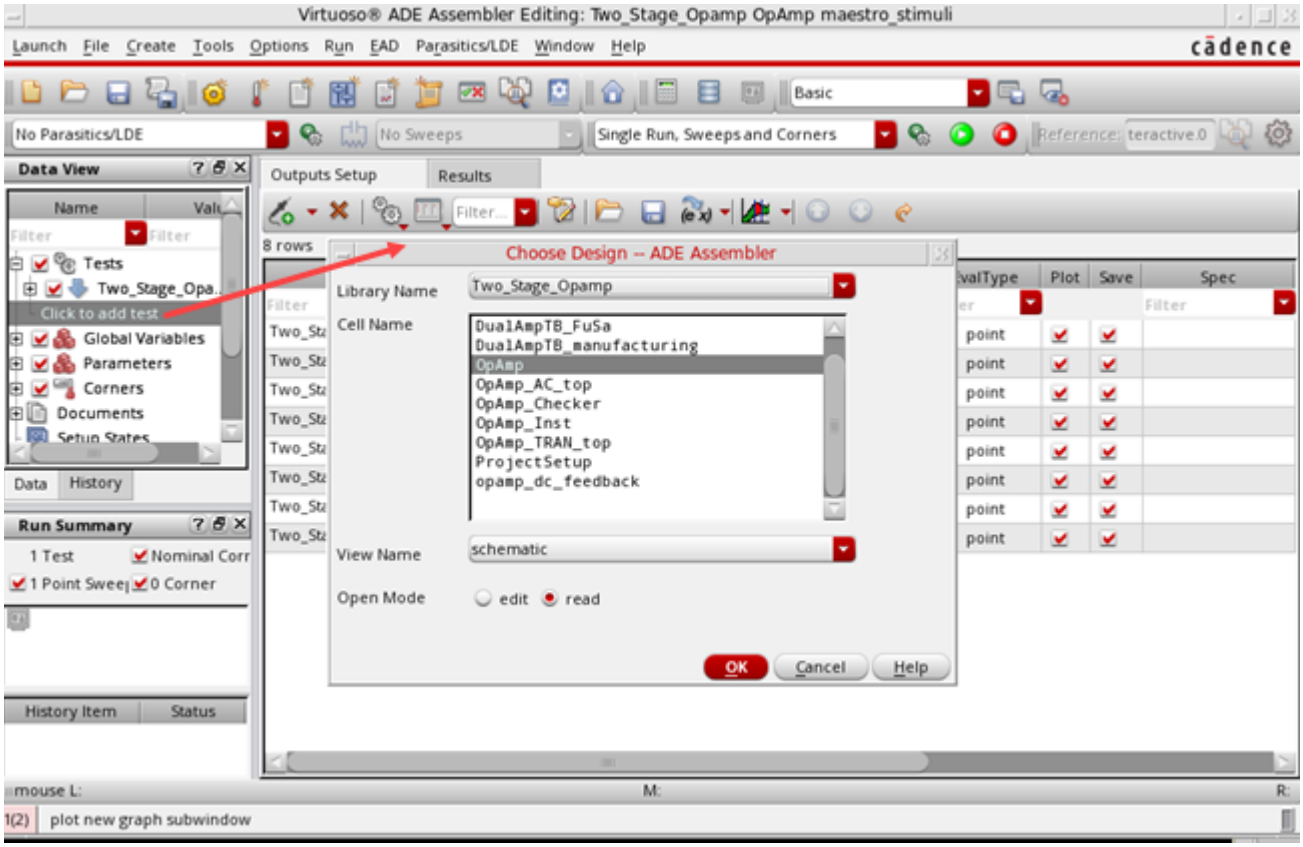
To add a new test, do the following:

1. On the Data tab in the Data View assistant pane, right-click the test and choose *Add Test*.

Tip

Alternatively, you can click the *Create Test* ( ) icon on the toolbar or click where it says *Click to add test* on the Data View assistant pane.

The Choosing Design -- ADE Assembler form appears.



2. In the *Library Name*, *Cell Name*, and *View Name* fields, select a design.

   See Specifying a Design for a Test for more information.

3. Click *OK*.

   Observe the following:

© 2023 Cadence Design Systems, Inc.

- A new test is created and opened in Virtuoso ADE Explorer, which is more suitable to work on a single test. You can set up the test details and switch to Virtuoso ADE Assembler. The newly created test is added to the *Tests* tree in the Data View assistant pane.
- The name of the test is `libName_cellName_sequenceNumber` where `sequenceNumber` starts at `1` and is incremented by one for every test you add that has the same base name. This helps in assigning a unique name to each test name. For example, the first test created in the `maestro_sample` cell in the `demoLib` library is named `demoLib_maestro_sample_1`, the second test is named `demoLib_maestro_sample_2`, and so on.
- The program performs a copy-from-cellview operation so that design variables appear on the Variables and Parameters assistant pane when you first add a test.

You can now edit the test details in ADE Explorer. See *Virtuoso ADE Explorer User Guide* for more details. After editing the details, click the up arrow to the left of test name in the Setup assistant to bring back the test in the ADE Assembler window.

Also see: Modifying a Test in ADE Explorer

Renaming a Test

To rename a test, do the following:

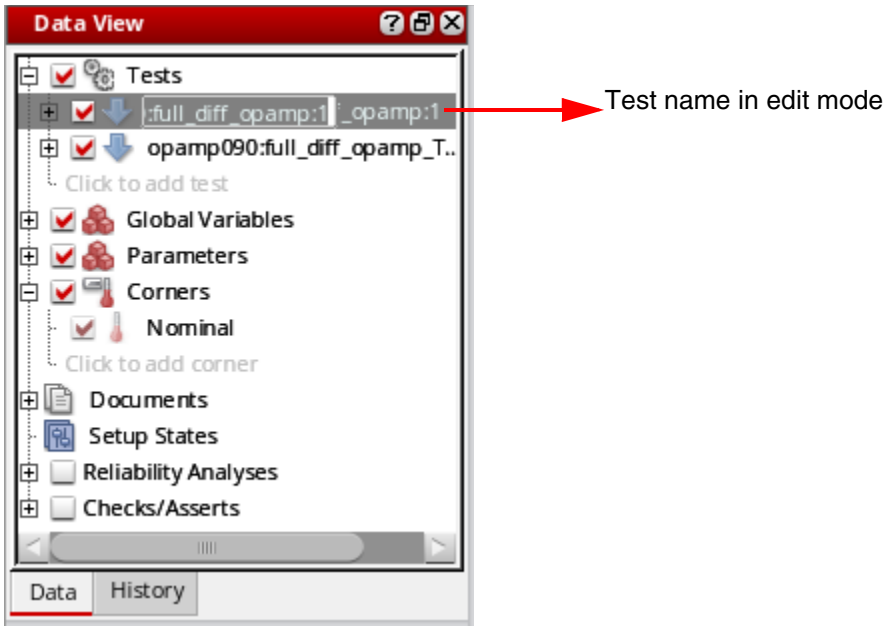
1. Click once to select the test name you want to change.

   Click again to make the name editable.

   Note: If you double-click a test name in quick succession, it is opened in ADE Explorer for editing. To disable opening of the test in ADE Explorer after its name is double-clicked, set the openExplorerOnTestDoubleClicked environment variable to `nil`.

© 2023 Cadence Design Systems, Inc.

The entire test name is highlighted.



2. Either type a completely new name for the test, or click to place the edit cursor and edit the name.
3. Press *Return*.

   The new test name appears on the Data View assistant pane and the Outputs pane.

Important

If you use colon in a test name, the tool automatically changes that to an underscore.

Copying a Test

To reuse the settings of an existing test, you can create a copy of the test. This allows you to quickly setup new tests.

To copy a test, do the following:

➤ On the Data View assistant pane, right-click the test you want to copy and choose *Create Test Copy*.

The new test name appears on the Data View assistant pane.

The name of the new test has the suffix `_sequenceNumber` where `sequenceNumber` starts at `1` and increments by one for every copy of the

© 2023 Cadence Design Systems, Inc.

original test (so that the test name of each copy is unique). For example, if the original test has the name `demoLib_ade_sample_1`, the first copy of the test is named `demoLib_ade_sample_1_1`, the second copy of the original test is named `demoLib_ade_sample_1_2` and so on.

Removing a Test

To remove a test, do the following:

➤ On the Data View assistant pane, right-click the test you want to remove and choose *Delete*.

The program deletes the selected test. You cannot undo this action.

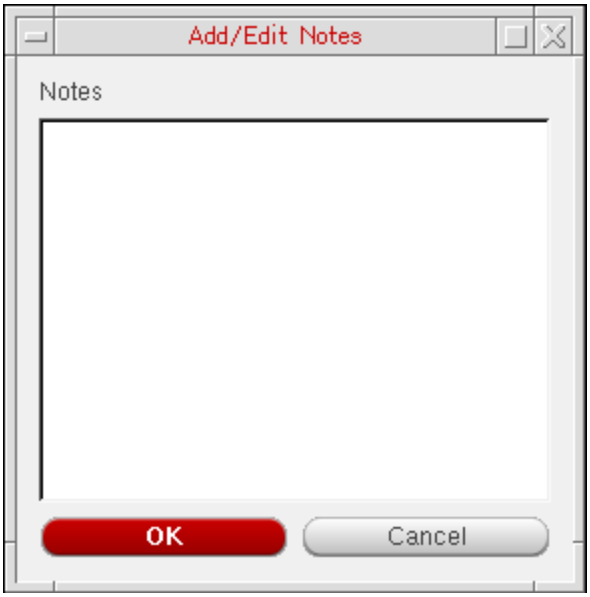
Note: If you delete the design attached to a test from the Library Manager, it shows a warning message to seek confirmation for removal.

Adding Notes to a Test

To add notes to a test, do the following:

1. On the Data View assistant pane, right-click the test for which you want to add notes and choose *Notes*.

   The Add/Edit Notes form is displayed.



2. In the *Notes* field, add notes for the test.

© 2023 Cadence Design Systems, Inc.

Note: By default, the notes field can accept only 512 characters. You can change this limit by setting the maxNotesLength environment variable.

3. Click *OK*.

   The notes are added to the test details in the ADE Assembler setup database. Similarly, you can add notes for global variables, parameters, corners, Reliability Analysis setup, or history items.

Important Points to Remember

- Notes added to a test are visible in the tooltip information, which is displayed when you hover the pointer over a test name on the Data View assistant pane. If the notes are long, only the first 10 rows are displayed in the tooltip by default. You can change this limit by setting the maxNotesRowsDisplay environment variable.
- To remove notes added to a test, right-click the test name and choose *Delete Notes*.

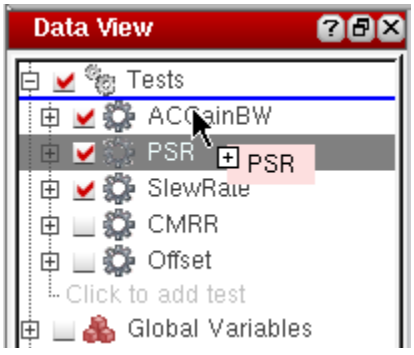
Changing the Order of Tests

When there are multiple tests defined in the ADE Assembler setup, you can change the order in which the tests are listed in the *Data View* assistant. The order of tests in this assistant also defines the order in which simulations are run.

To change the order of a test, do the following in the *Data* tab of the *Data View* assistant:

1. Expand *Tests* to view all the tests.
2. Click a test name and drag it.

   A blue indicator line appears at the location where you have moved it. The tooltip also shows the name of the test being moved.

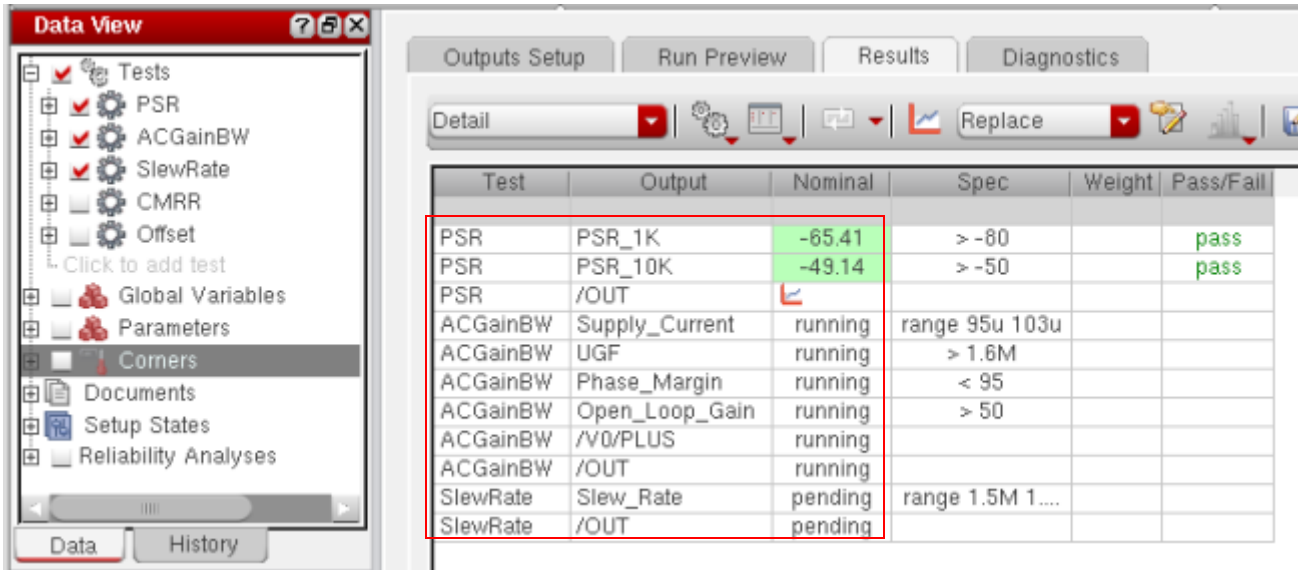


3. Drag the indicator line to the desired location and drop the test.

© 2023 Cadence Design Systems, Inc.

Note: You can select more than one test and move them together.

The simulations are run in the order in which the tests are listed in the *Data View* assistant. The order in which the outputs appear on the *Outputs Setup* tab and the *Results* tab is also changed to match this, as shown below.



Viewing Information about a Test

You can view information about a test in ADE Assembler such as the test name, the design associated with the test (lib/cell/view), the target simulator, and the ADE state.

To view information about a test, do the following:

➤ On the Data View assistant pane or on the Variables tab of the Variables and Parameters assistant pane, hover the mouse cursor over a test name.

A text box containing test information appears.

```
    Name: Trans_12u_Gain_Vio_CMRR_SR
    Tool: ADE
  Design: demoLib/adc_cascode_opamp_sim/config_Gain_Vio_CMRR_SR

Setup Arguments
                          path: $AXL_SETUPDB_DIR
                           sim: spectre
                         state: Trans_12u_Gain_Vio_CMRR_SR_Trans_12u_Gain_Vio_CMRR_SR_Interactive.0
```

If you have added notes for the test, they are appended to the test information.

© 2023 Cadence Design Systems, Inc.

After a while, or if you move the mouse cursor, the text box disappears.

© 2023 Cadence Design Systems, Inc.

Adding, Changing, and Removing Analyses

See the following topics:

- Adding an Analysis on page 102
- Changing an Analysis on page 104
- Changing the Order of Analyses on page 105
- Removing an Analysis on page 105

Adding an Analysis

To add an analysis to your test, do the following:

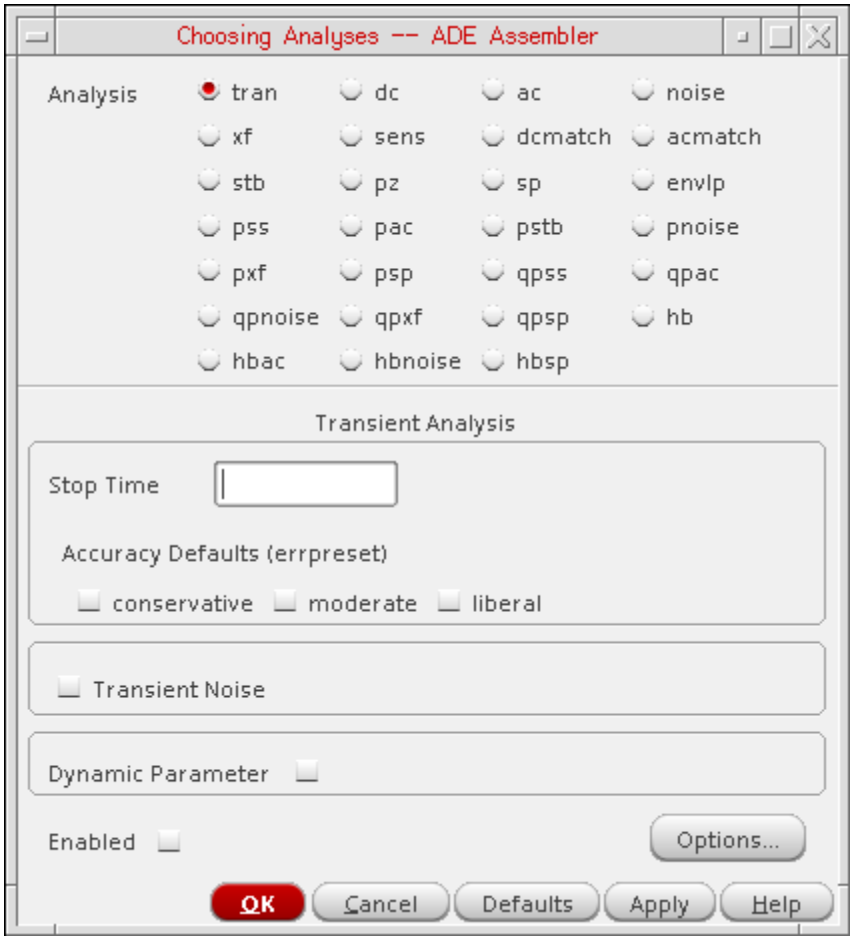
1. On the Data View assistant pane, right-click the test and choose *Add Analysis*.

Tip

Alternatively, you can click where it says *Click to add analysis* in an expanded test tree on the Data View assistant pane.

© 2023 Cadence Design Systems, Inc.

The Choosing Analyses form appears.



The items that appear on this form depend on the simulator you selected (see Choosing the Target Simulator).

2. In the *Analysis* section, select a radio button for the analysis type you want to define.

   The setup parameters corresponding to the analysis you selected appear on the form.

3. Set the analysis parameters you want (see the user guide of the simulator for more information about the options).

   After you specify one analysis parameter, the *Enabled* check box is selected.

4. Click *Options* to view more options for the analysis. Review and if required, modify the options and close the form.

5. Click *OK*.

© 2023 Cadence Design Systems, Inc.

The analysis you specified appears under the test name on the Data View assistant pane.

Related Topics

- Setting Up Analysis in *Virtuoso ADE Explorer User Guide*
- Setting up Transient Analysis for Electrothermal Analysis in Legato Reliability Solution

Changing an Analysis

To change an analysis setup, do the following:

1. On the Data View assistant pane, double-click the analysis you want to change.

   Note: You might need to expand the tree by clicking the plus sign (*+*) to the left of the test that contains the analysis you want to change.

   The Choosing Analyses form appears.

   The items that appear on this form depend on the simulator you selected (see "Choosing the Target Simulator" on page 109).

Tip

Alternatively, you can change an analysis by right-clicking its name and choosing *Edit* from the context-sensitive menu that appears.

2. (Optional) In the *Analysis* section, select a radio button to change the analysis type you want to define.

   The setup parameters corresponding to the analysis you selected appear on the form.

3. (Optional) Change the analysis parameters you want (see your simulator's user guide for more information).

4. (Optional) You can deselect the *Enabled* check box to turn off the analysis without removing the setup information.

Tip

Alternatively, you can turn off an analysis by deselecting the check box next to the analysis name in the tree on the Data View assistant pane.

5. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Changing the Order of Analyses

When you add an analysis, it is added at the end of the list of analyses displayed under the test in the Data View assistant. During simulation, the analyses are run in the same order as they appear in this list. If required, you can change the order in which the analyses are displayed and run.

To change the order of an analysis, click it in the Data View assistant, drag it to the desired position in the list, and drop.

To change the order of multiple analyses, press the *Ctrl* key, click the analyses to select them, drag them to the desired position, and drop.

Note: For some simulators the analyses need to be run in a pre-defined sequence. For example, the pss analysis should run before pac. Therefore, every time you change the position of an analysis, the tool checks whether all the analyses are placed in the correct order as required by the selected simulator. If you place an analysis at an incorrect position, it is moved back to its original position and an appropriate message is displayed.

Removing an Analysis

There is no undo for this action.

To remove an analysis, do the following on the Data View assistant pane:

1. To the left of the test containing the analysis you want to remove, click *+* to expand the test (if it is not already expanded).
2. Right-click the analysis you want to remove and choose *Delete*.

   The program removes the selected analysis.

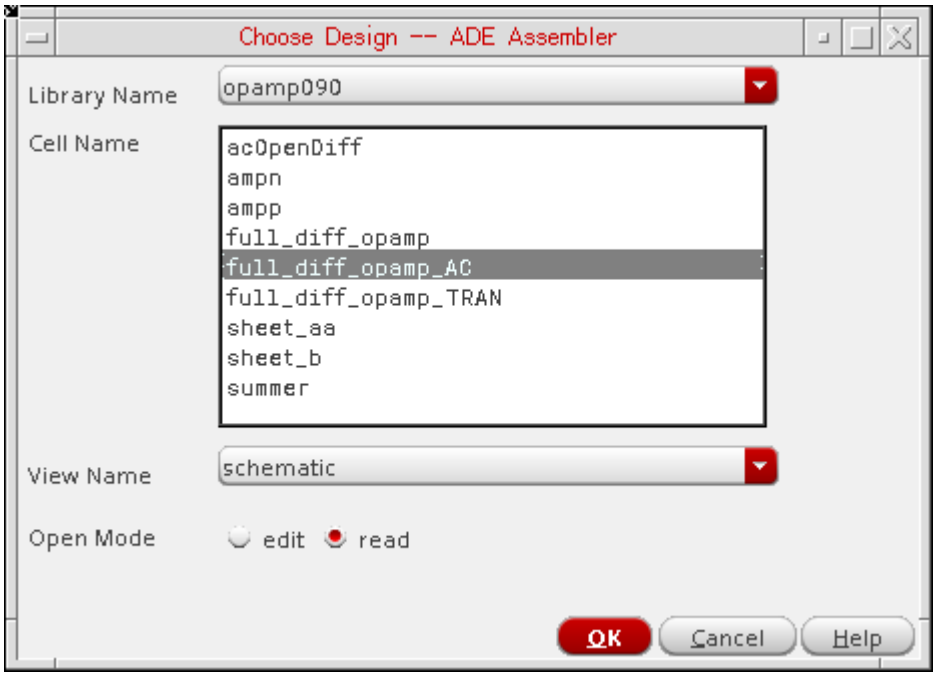
© 2023 Cadence Design Systems, Inc.

Specifying a Design for a Test

To specify a design for a test, do the following:

1. On the Data View assistant pane, right-click the test or analysis name and choose *Design*.

   The Choosing Design form appears.



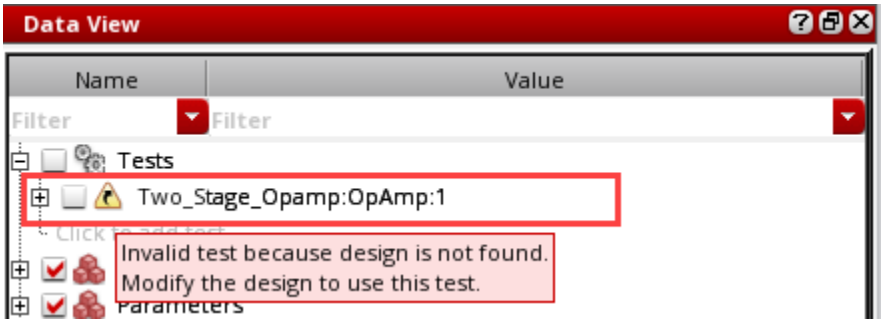
2. In the *Library Name* drop-down list, select a library name.
3. In the *Cell Name* list box, select a cell name.
4. In the *View Name* drop-down list, select a view name.
5. For the *Open Mode*, select one of the following options:
   - *edit* opens the design in edit mode
   - *read* opens the design in read-only mode
6. Click *OK*.

Note: You can modify the design for multiple tests together by selecting the test names in the Data View assistant and choosing the *Design* command from the context-sensitive menu.

A reference to the chosen design is saved with the test. Later in the flow, if you modify, rename, or move the design and it is not found while opening a maestro cellview, the test

© 2023 Cadence Design Systems, Inc.

using that design is marked as invalid, and it is disabled. The test icon indicates an issue and the tooltip shows an error, as shown below.



To resolve this issue, right-click on the disabled test, and choose an appropriate command to resolve the issue. Note that in this case, you can take one of the two actions:

- Change the design of the invalid test
- Delete the test and add a new one

If you use the *Change Design* command and assign a valid schematic, the test is considered valid and the icon is also changed.

If you run a simulation without resolving the invalid test issue, an error message is displayed prompting you to either continue the simulation for other valid tests in the setup or to cancel the simulation run to resolve the issue before continuing further.

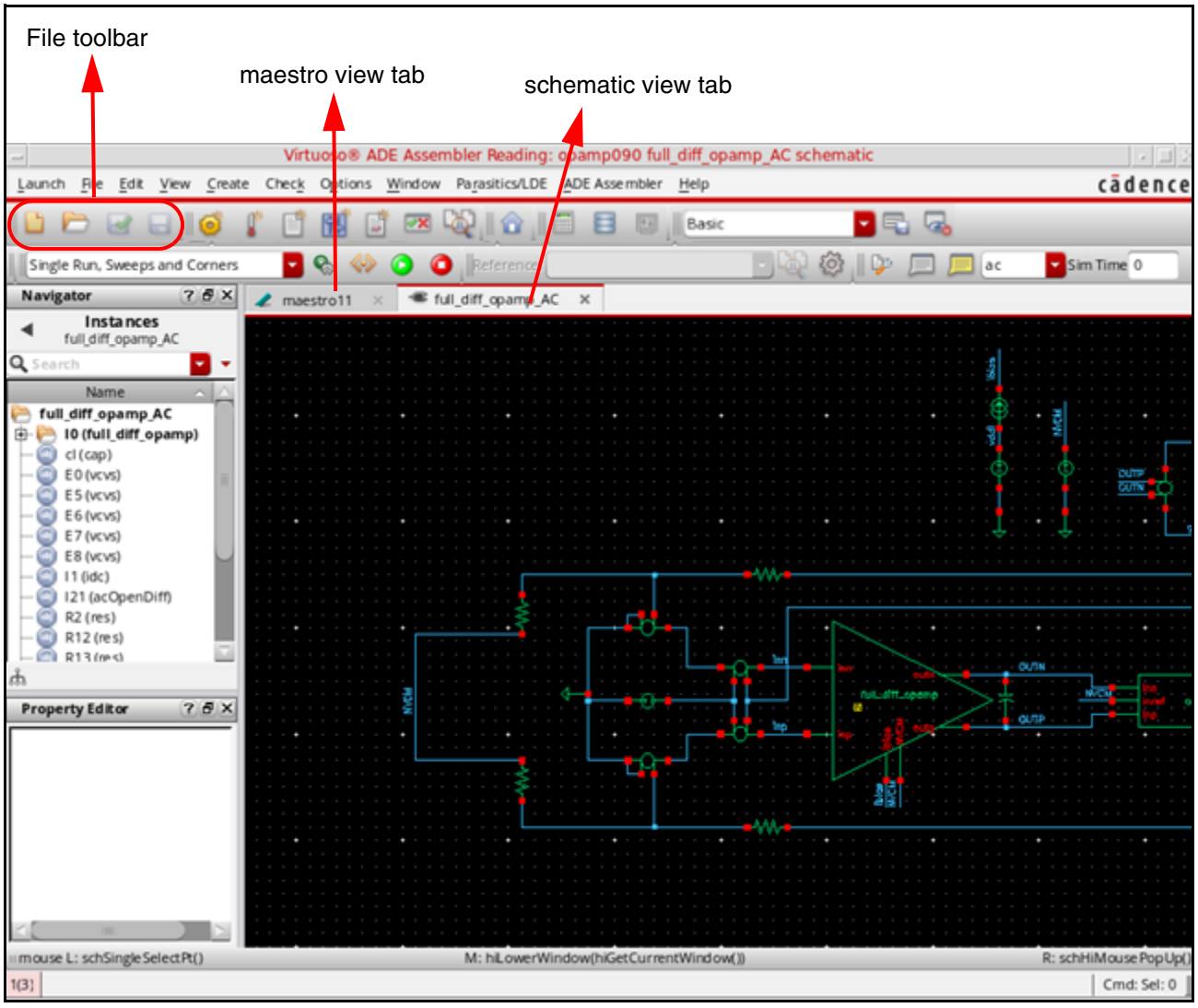
© 2023 Cadence Design Systems, Inc.

Opening a Design Schematic

To open the design associated with a test, do the following:

➤ On the Data View assistant pane, right-click the test and choose *Open Design in Tab*.

The design associated with the test appears on a new tab in your session window.



Tip

When the design is open, press `Ctrl+Tab` to switch between ADE Assembler and the schematic view.

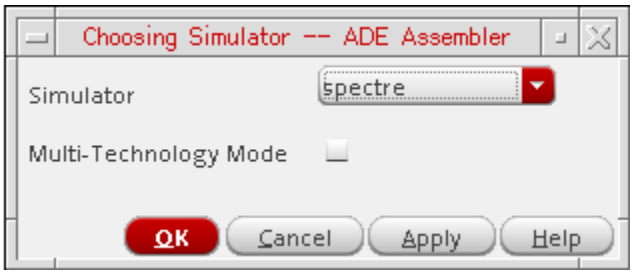
© 2023 Cadence Design Systems, Inc.

Choosing the Target Simulator

To choose the target simulator for a test, do the following:

1. On the Data View assistant pane, right-click the test or analysis name and choose *Simulator*.

   The Choosing Simulator form appears.

   Note: For information about *Multi-Technology Mode*, see Multi-Technology Designs and Locally Scoped Models and Options.

2. In the *Simulator* drop-down list, select a simulator.
3. Click *OK*.

To specify options for high performance simulation, do the following:

1. On the Data View assistant pane, , right-click the test or analysis name and choose *High-Performance Simulation*.

© 2023 Cadence Design Systems, Inc.

The High-Performance Simulation Options form appears.

By default, the *APS* mode is selected.

© 2023 Cadence Design Systems, Inc.

If you have the path to Spectre 19.10 set in your environment, you see an additional simulation performance mode, *Spectre X*, in this form.

For more details on the various performance modes and related options on this form, refer to Setting Spectre Options in the *Virtuoso ADE Explorer User Guide*.

For details on using the AMS simulator for running simulations in ADE Assembler, refer to Working with AMS Simulator in *Virtuoso ADE Explorer User Guide*.

For details on using the Ultrasim simulator for running simulations in ADE Assembler, refer to Working with Ultrasim Simulator in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

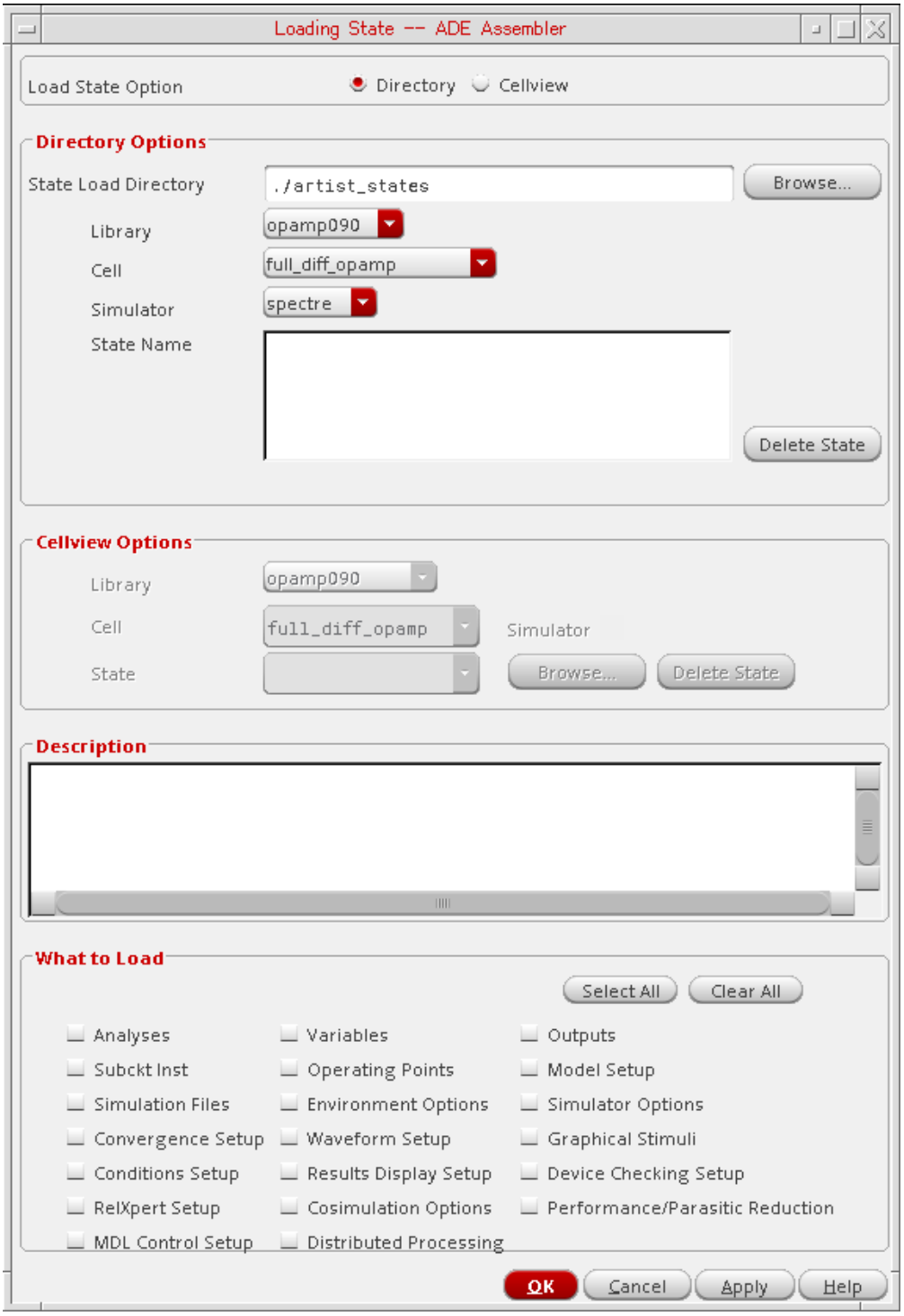
Loading State Information

To load state information to a test, do the following:

1. On the Data View assistant pane, right-click the test or analysis name and choose *Load State*.

© 2023 Cadence Design Systems, Inc.

The Loading State form appears.



In the *Load State Option* group box, the *Directory* radio button is selected by default.

© 2023 Cadence Design Systems, Inc.

2. In the *Directory Options* group box, select a state load directory by choosing information from the various drop-down lists or by clicking *Browse* to open a directory browser window and navigating to your state directory.

   State names appear in the *State Name* list box.

3. In the *State Name* list box, select the name of the state you want to load.

4. In the *What to Load* group box, select the check box for each type of information you want to load. For example, if you want to load analyses from your state file, select the *Analyses* check box; if you want to load design variables, select the *Variables* check box.

Tip

You can click *Select All* to select all check boxes or *Clear All* to clear them.

5. Click *OK*.

   The state information you selected appears on the appropriate assistant panes in your environment according to the following guidelines:

   - ADE Assembler ignores state information that is incompatible with the selected simulator.
   - The program overwrites state variables with matching name and value types.
   - An error message appears if the state variable name is the same but the value type is different.

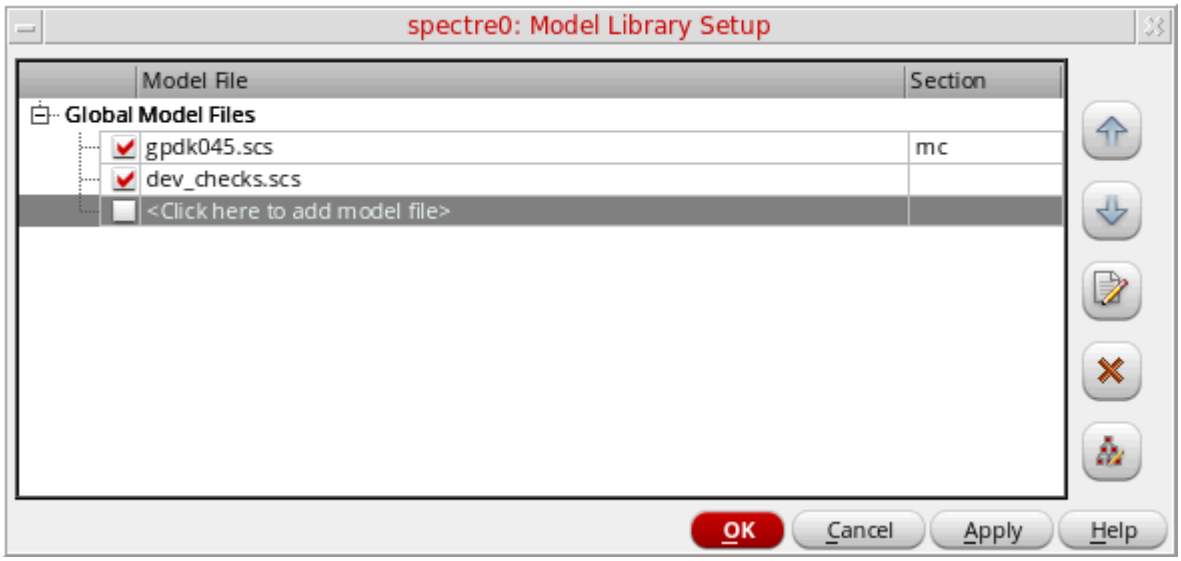
© 2023 Cadence Design Systems, Inc.

Specifying Model Libraries

To specify the model libraries to use when simulating a test, do the following:

1. On the Data View assistant pane, right-click the test or analysis name and choose *Model Libraries*.

   The Model Library Setup form appears.



2. In the *Model File* column, type the path and file name of the model file you want to use.

   Alternatively, you can click … to open the Choose Model File form where you can navigate to select a valid model file.

   See Chapter 7, “Working with Model Files and CDF” for more information about model files.

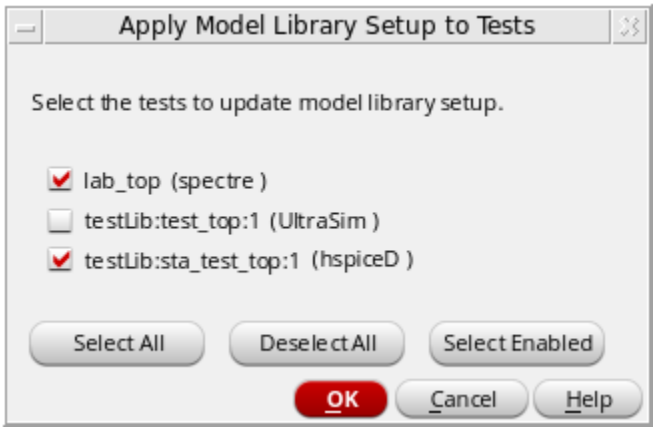
3. (Optional) In the *Section* column, select a section from the drop-down list.

   A model file can have zero or more sections. If a model file contains no sections, there is no drop-down list available. The section you select determines which model definition the simulator uses.

4. (Optional) Click the *Apply to the Selected Tests in the Setup* ( ) button to update model file(s) for multiple tests in a setup.

© 2023 Cadence Design Systems, Inc.

The Apply Model Library Setup to Tests form appears.



a. Select the tests to which you want to apply the model file(s).

You can also click the *Select Enabled* button to select the tests that are enabled in the Data View assistant.

b. Click *OK*.

Caution

When you apply the model library setup to other tests, their existing model library setup is overwritten. Therefore, it is recommended to verify their existing setup before using this feature.

For more information about sections in model files, see “Corners Modeling” in the *Direct Simulation Modeling User Guide*.

© 2023 Cadence Design Systems, Inc.

Setting the Simulation Temperature

You can set the simulation temperature for a test, in any of the following ways:

- By using the Setting Temperature form
- By using a test variable named `temperature`
- By using a global variable named `temperature`, a common value for multiple tests

Using the Setting Temperature Form

On the Data View assistant pane, right-click the test or analysis name and choose *Temperature* to open the Setting Temperature form. Specify a constant temperature value or an expression using `VAR`. For more details, refer to the Setting the Simulation Temperature section in *Virtuoso ADE Explorer User Guide*.

Note: If you parameterize temperature by using the VAR expression in the Setting Temperature form, the temperature field in the Corners Setup form is reset to default blank.

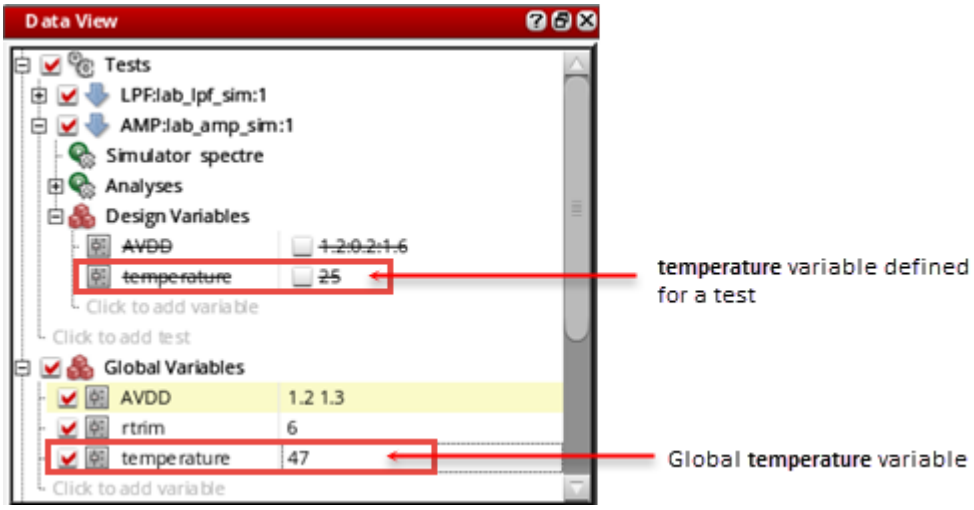
Using a Test Variable Named `temperature`

Create a **test variable** named `temperature`. Specify a single value or a sweep set to vary temperature. For more details, refer to the Setting the Simulation Temperature section in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

Using a Global Variable Named `temperature`

To specify a single value or a sweep set for temperature to be used by multiple tests, create a **global variable** named `temperature`, as shown below.



Important

To refer to the temperature value in an output expression or specification, use `VAR("temp")` instead of `VAR("temperature")`. Even if the temperature for a particular test is specified using the `temperature` design variable, you can access it as `VAR("temp")` in the results.

© 2023 Cadence Design Systems, Inc.

Specifying Analog Stimuli

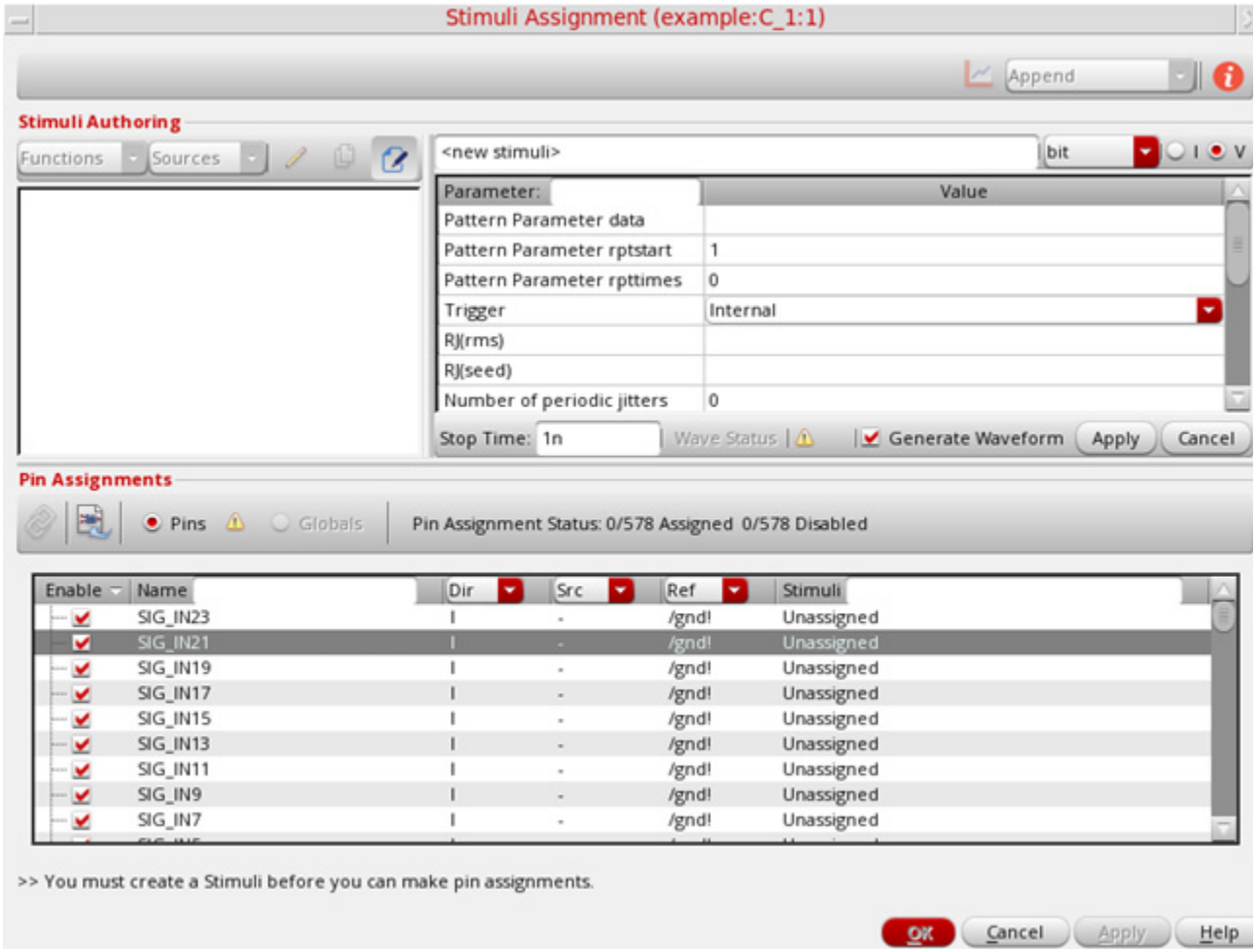
To specify analog stimuli (input stimulus and global sources) for a test, do the following:

Important

The top-level schematic of your design must have input pins, bidirectional pins, or global nets (such as `vdd!` for power stimulus).

1. On the Data View assistant pane, right-click the test or analysis name and choose *Stimuli*.

   The Stimuli Assignment form appears.



For details on how to use this form, refer to Setting up Stimuli in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

See also

- Specifying a Custom Library of Sources on page 121
- Specifying Simulation Files on page 122

© 2023 Cadence Design Systems, Inc.

Specifying a Custom Library of Sources

Unless you specify otherwise, all sources, whether used for stimulus or for a power supply, come from `analogLib`. To specify a custom library of sources, do the following:

1. In the CIW, choose *Tools – Library Manager*.

   The Library Manager form appears.

2. From the *Library* list box, select the library of the current design.
3. Choose *Edit – Properties*.

   The Library Property Editor form appears.

4. If the *refLibs* property does not already exist, create it as follows:

   a. On the Library Property Editor form, click *Add*.

      The Add Property form appears.

   b. In the *Name* field, type `refLibs`.

   c. In the *Type* drop-down list, select *string*.

      The *Value* and *Possible Choices* fields appear.

   d. Click *OK*.

      The *refLibs* property appears on the Library Property Editor form.

5. In the value field for *refLibs*, type a list of one or more libraries in the search order you want.
6. Click *OK*.
7. On the Library Manager form, choose *File – Exit*.

© 2023 Cadence Design Systems, Inc.

Specifying Simulation Files

Depending on your design and setup requirements, you need to use various types of files for simulations run in ADE Assembler.

The table below describes each file type and its purpose.

File Type	Description
Definition Files	Definition files contain function and parameter definitions that are not displayed in the *Design Variables* list in the Data View assistant. See the following sample file that contains function and parameter definitions for the Spectre circuit simulator. `<your_install_dir>/tools/dfII/samples/artist/models/spectre/definitions.scs` The parameters in this file are referenced by included models and are not referenced from any part of the design in the Cadence library.
Stimulus Files	Stimulus files can contain input and power supply stimuli, initializing nodes, and estimated parasitics in the netlist. You can look at the following example file that contains Spectre circuit simulator stimuli definitions for the opamp sample design in the aExample library. `<your_install_dir>/tools/dfII/samples/artist/models/spectre/opampStimuli.scs` In a stimulus file, you can type node names and component names using Open Simulation System (OSS) syntax `[#name]` and the system will substitute the corresponding node numbers when writing the netlist. You can use a backslash (`\`) to escape a square bracket. For information about OSS syntax, see the *Open Simulation System Reference*.
DSPF Parasitic Files	DSPF files contain the post-layout data, including fingered devices and net parasitics, of the design. For EMIR analysis, Spectre APS/XPS requires a complete testbench that contains the DSPF files (with the parasitic and instance sections describing the circuit to be analyzed), stimuli, and device models.

© 2023 Cadence Design Systems, Inc.

SPEF Parasitic Files	SPEF post-layout netlists are supported by the Spectre APS/XPS EMIR flow. An SPEF netlist contains a combination of a schematic netlist with the active devices and an SPEF netlist that includes the parasitic element information.
PSpice® Files	PSpice files contain netlists that include PCB components modeled using the PSpice format. When you use PSpice files, the top-level netlist and control statement need to be defined in Spectre or SPICE format. A subcircuit defined in PSpice netlist format is instantiated in the Spectre netlist. **Note:** For the AMS simulator, the *PSpice Files* field is supported only when you select AMS Unified Netlister with xrun from the Netlist and Run Options form. For information on AMS UNL, see Working with AMS Simulator.
Vector Files	Vector (VEC) files include a table of times and values at those times. A vector file can include more than one vector definition.
VCD Files	VCD (Verilog® Value Change Dump) files contain information about value changes for selected variables in the circuit design.
EVCD Files	EVCD (Extended Verilog® Value Change Dump) files contain variable changes in all states and provide strength information and port direction.

You can use the Simulation Files Setup form in ADE Assembler to specify the paths to these files.

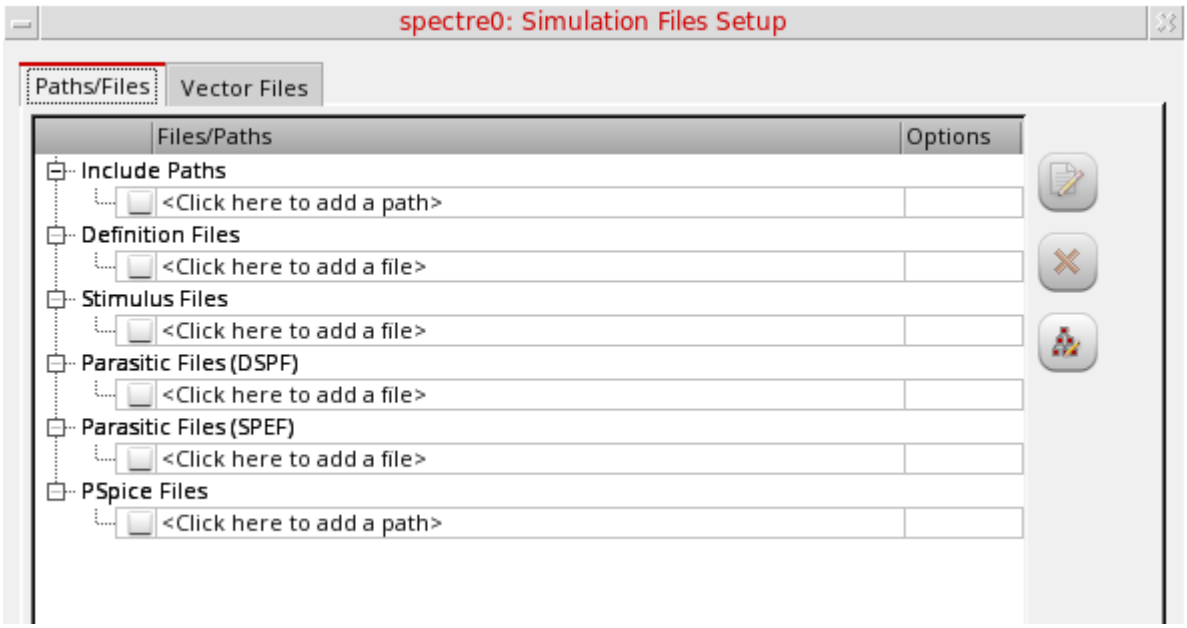
To specify simulation files for a test, do the following:

1. In the Data View assistant pane, right-click the test or analysis name, and choose *Simulation Files*.

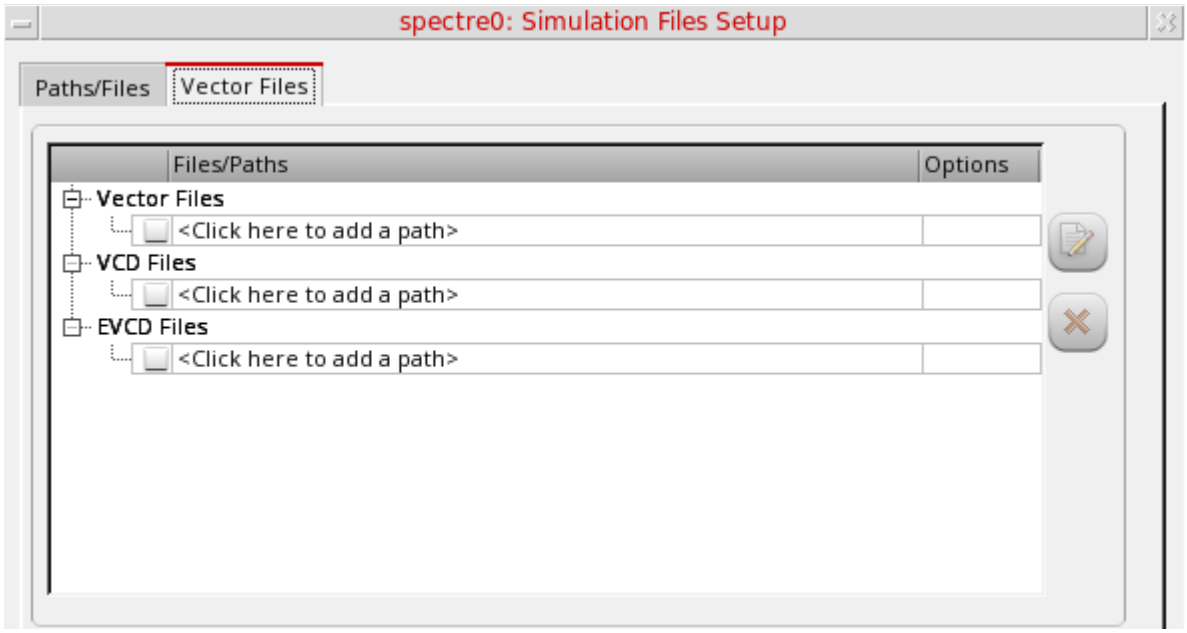
   The Simulation Files Setup form appears. This form contains two tabs:

© 2023 Cadence Design Systems, Inc.

- *Paths/Files*: On this tab, you can specify paths for all file types other than vector files.



- *Vector Files*: On this tab, you can specify digital vector files.

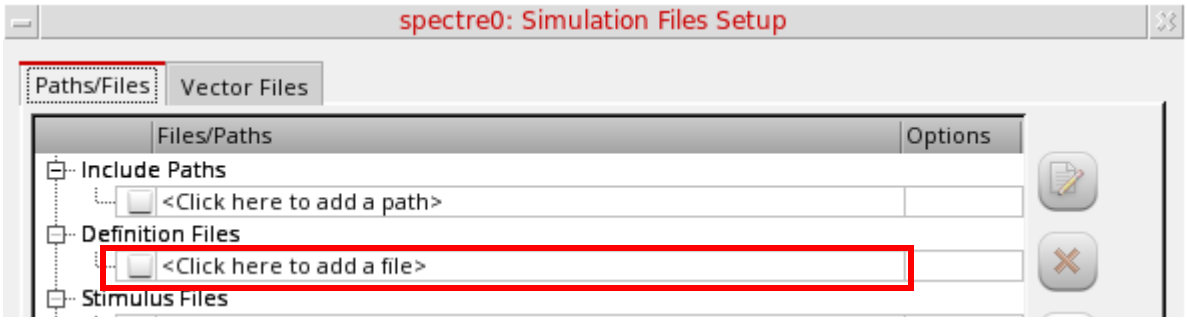


2. If the files to be used are located at multiple locations and you want ADE Assembler to look for files in all those paths, specify those paths in the *Include Paths* tree.

© 2023 Cadence Design Systems, Inc.

3. To specify the path to a file of a specific type, click where it says *<Click here to add a path>* or *<Click here to add a file>* in the tree of that file type.

   - For example, to specify the path to a definition file, click where it says *<Click here to add a file>* in the *Definition Files* tree.



4. Type the path and name of your file, or click the browse button to select one or more files or paths using the Choose Files/Paths form.

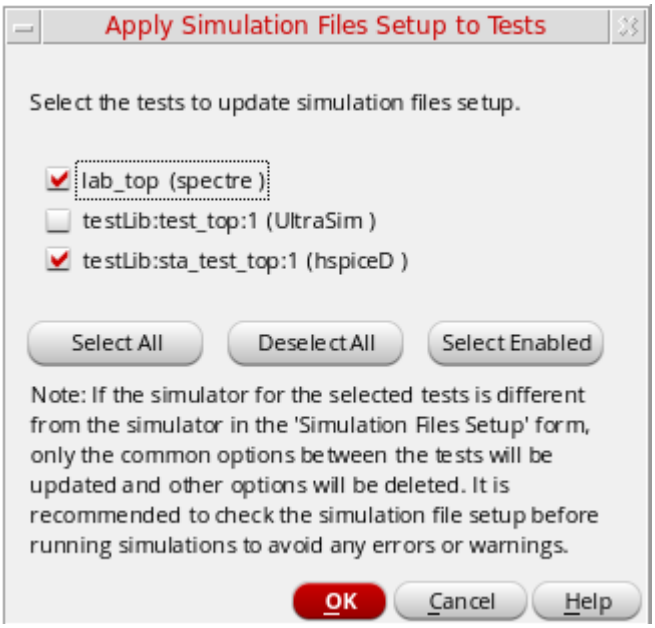
   Note: The simulator resolves a relative path by first looking in the `netlist` directory (relative to where you run the simulation). If the path starts with the `.` character, the simulator resolves this by looking in the `netlist` directory first, then in each of the directories specified in *Include Paths* in the order you type them. The `.` character does not mean the current directory.

   After you have specified a file or path, a new row is added to the tree. If you have multiple files, you can use the additional rows to specify the paths to those files.

5. (Optional) Click *Apply to the Selected Tests in the Setup* ( ) on the *Paths/Files* tab of this form to update simulation files for all the applicable tests in a setup.

© 2023 Cadence Design Systems, Inc.

The Apply Simulation Files Setup to Tests form appears.



a. Select the tests to which you want to apply the simulation files.

You can also click *Select Enabled* to automatically select the tests that are enabled in the Data View assistant.

b. Click *OK* to apply the selected simulation files.

Caution

When you apply the simulation files to other tests, their existing simulation files setup is overwritten. Therefore, it is recommended to verify their existing setup before using this feature.

6. Click *OK* to close the Simulation Files Setup form.

Important Points to Note

- For some of the file types, a few additional fields are displayed on the form.

  For example, when you select a vector file, the *Vector File Options* section is displayed on the *Vector Files* tab with the *hlCheck* field. To enable the check for H and L states for input signals, choose *1*.

  Selecting a VCD or EVCD file displays an additional field in which you can specify a VCD or EVCD file is.

© 2023 Cadence Design Systems, Inc.

- By default, the simulation files you add are enabled for simulation. You can use the check box next to a file name to enable or disable a simulation file for simulation.
- To edit a simulation file, select the file and click . The selected simulation file is opened in a text editor where you can make changes to the file content and save it.
- To remove a simulation file from the setup, select it and click .

© 2023 Cadence Design Systems, Inc.

Specifying Simulation Environment Options

To specify environment options for a test, do the following:

1. On the Data View assistant pane, right-click the test or analysis name and choose *Environment*.

   The Environment Options form appears.

   The fields that appear on the form depend on your target simulator as follows:

   - Environment Options for the Spectre Circuit Simulator on page 129
   - Environment Options for the UltraSim Circuit Simulator on page 130
   - Environment Options for the AMS Designer Simulator on page 131
   - Environment Options for the hspiceD Circuit Simulator on page 132

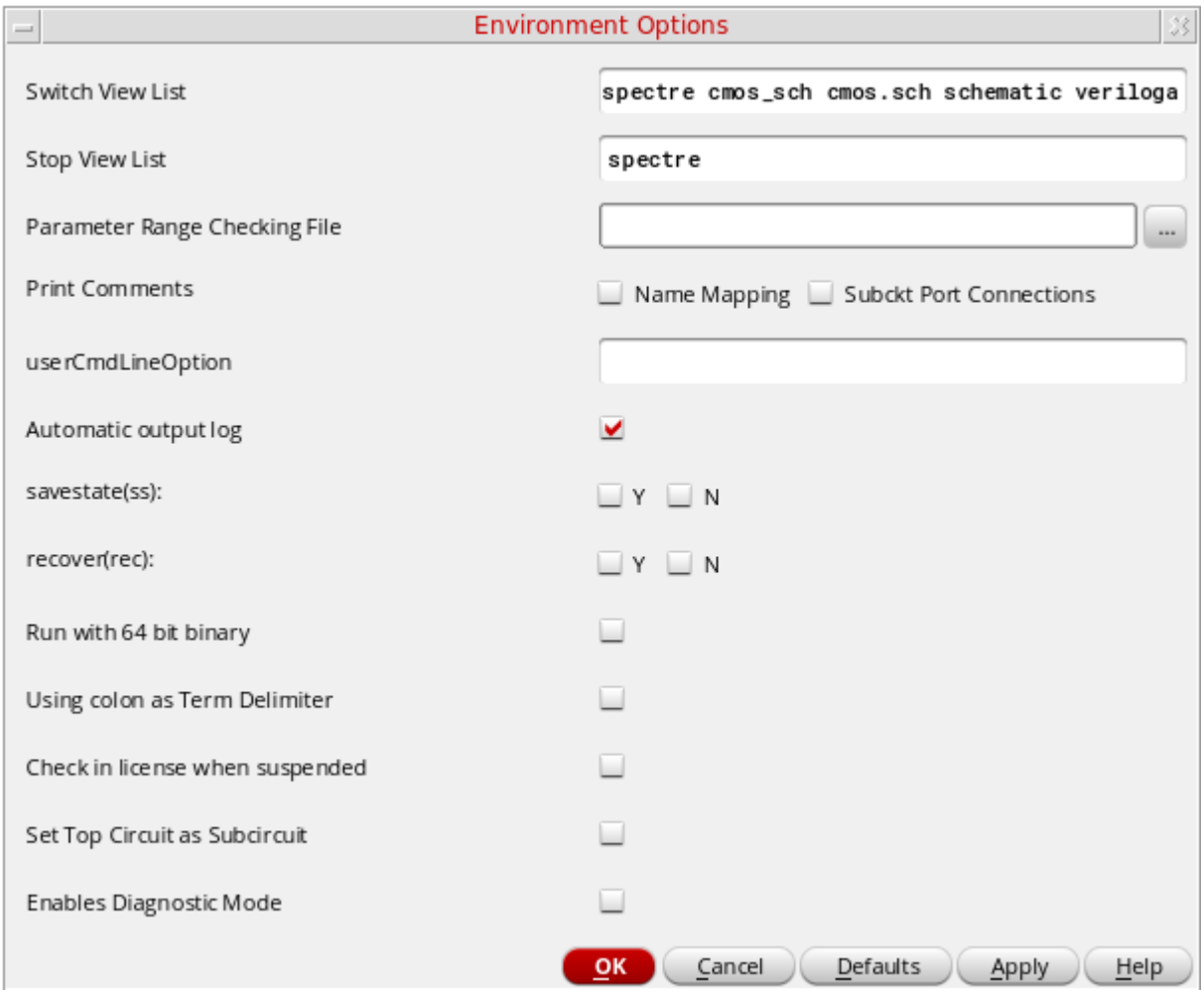
2. When you are done specifying options, click *OK*.

   The environment options you specified are applied to future simulations.

© 2023 Cadence Design Systems, Inc.

Environment Options for the Spectre Circuit Simulator

You can specify the following options for the Spectre circuit simulator:

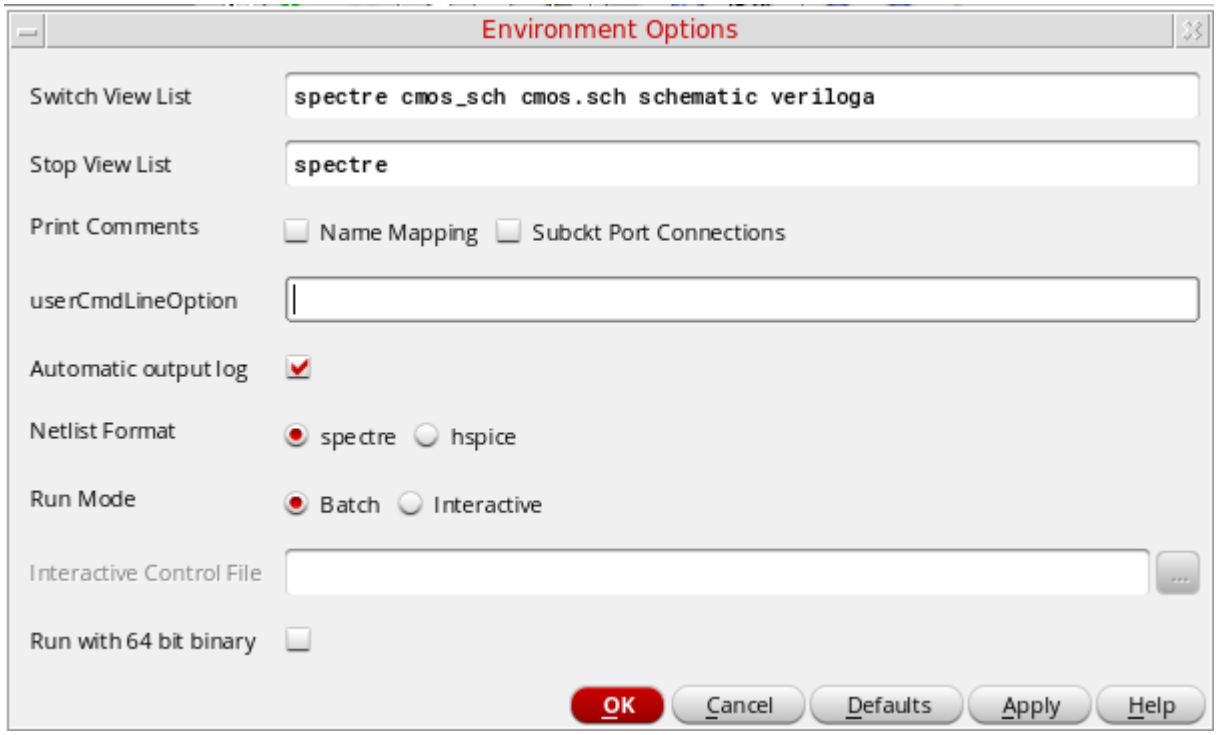


For more details about Spectre environment options on this form, refer to Setting up Simulator Environment Options in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

Environment Options for the UltraSim Circuit Simulator

You can specify the following options for the UltraSim circuit simulator:

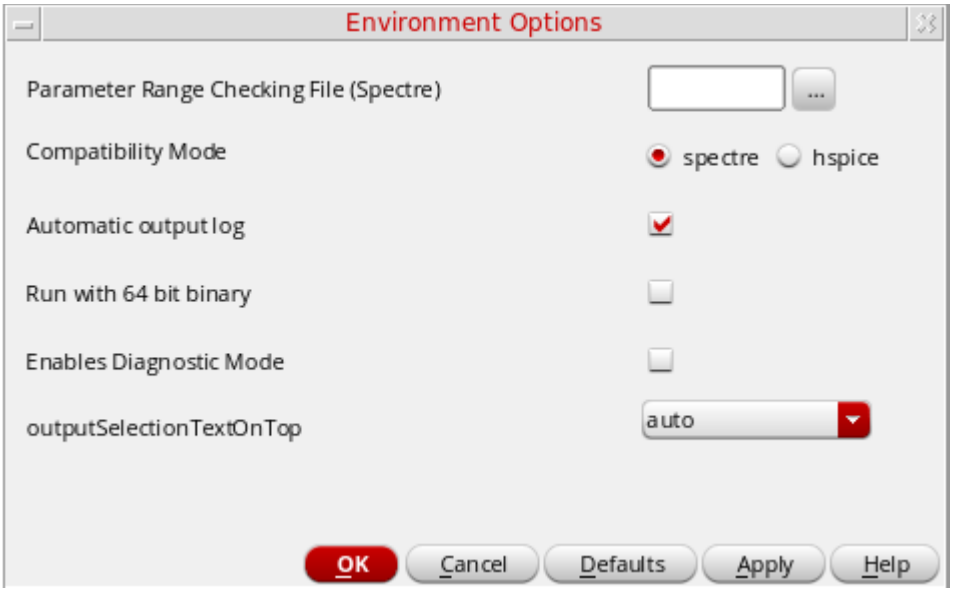


For more details about the Ultrasim simulator interface and the environment options on this form, refer to Working with UltraSim Simulator in the *ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

Environment Options for the AMS Designer Simulator

You can specify the following options for the AMS Designer simulator:



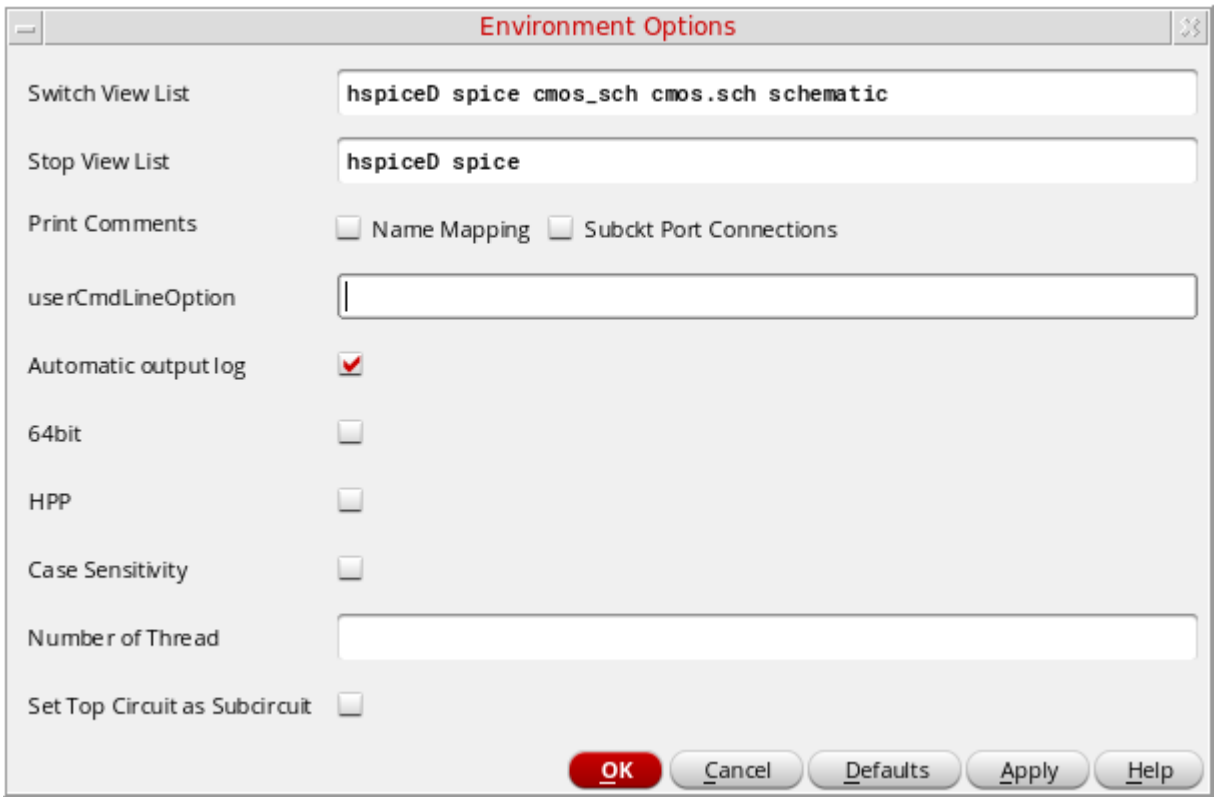
For more details about the AMS Designer simulator interface and the environment options on this form, refer to Setting Up Environment Options in the *ADE Explorer User Guide*.

Note: When the AMS simulator is running in interactive mode, you can run only single-point simulations. If the setup involves a multi-point simulation run, an error message is displayed in CIW when you click the Run button.

© 2023 Cadence Design Systems, Inc.

Environment Options for the hspiceD Circuit Simulator

You can specify the following options for the hspiceD circuit simulator:



For more details about the hspiceD simulator interface and the environment options on this form, refer to Working with HSPICE Simulator in the *ADE Explorer User Guide*.

Common Environment Options for Simulators

Switch View List

The *Switch View List* field appears on the Environment Options form for Spectre, UltraSim, and hspiceD circuit simulators. The switch view list tells the netlister the sequence to use when descending into different views of your design. The flowchart in the Netlisting chapter shows how the netlister selects views when expanding a design. The software searches through hierarchical views in the order you specify. See also "Expanding Hierarchy to Netlist a Design" on page 347.

➤ (Optional) In the *Switch View List* field, type a list of one or more view names to search when looking for design variables and when netlisting.

© 2023 Cadence Design Systems, Inc.

Stop View List

The *Stop View List* field appears on the Environment Options form for Spectre, UltraSim, and hspiceD circuit simulators.

➤ (Optional) In the *Stop View List* field, type a list of one or more view names (in no particular order) that you want the netlister to use as stopping views.

Print Comments

The *Print Comments* check box appears on the Environment Options form for Spectre, UltraSim, and hspiceD circuit simulators.

➤ (Optional) Select the *Print Comments* check box if you want the program to write extra comments regarding component location and name to your netlist.

Automatic Output Log

The *Automatic output log* check box appears on the Environment Options form for Spectre, UltraSim, and hspiceD circuit simulators.

➤ (Optional) Select the *Automatic output log* check box if you want the output log to appear automatically when the simulator generates messages.

Parameter Range Checking File

The *Parameter Range Checking File* field appears on the Environment Options form for Spectre and AMS circuit simulators.

➤ (Optional) In the *Parameter Range Checking File* field, type the name of the file containing the parameter range limits.

You do not need to type the full path for the file if it is in the directory specified in the *Include Path* field on the Simulation Files Setup form. The program interprets the . character in a UNIX path specification relative to the directory where you started the environment.

© 2023 Cadence Design Systems, Inc.

userCmdLineOption

The *userCmdLineOption* field appears on the Environment Options form for Spectre and UltraSim circuit simulators.

➤ (Optional) In the *userCmdLineOption* field, type any command-line options you want to send to the simulator.

savestate (ss)

The *savestate (ss)* check boxes appear on the Environment Options form for the Spectre simulators.

➤ Select one of the following *savestate (ss)* check boxes:

- *Y* indicates that you want to run spectre with the `+ss` option, which causes the simulator to save the states to a file when receiving interrupt signals like QUIT, TERM, INT, or HUP.
- *N* indicates that you want to run spectre with the `-ss` option, which disables the savestate option.

Start from Checkpoint File (rec)

The *Start from Checkpoint File (rec)* check boxes appear on the Environment Options form for Spectre circuit simulator.

➤ Select one of the following *Start from Checkpoint File (rec)* check boxes:

- *Y* indicates that you want to run spectre with the `+rec` option, which restarts a transient simulation based on conditions specified in a checkpoint file.
- *N* indicates that you want to run spectre with the `-rec` option, which does not restart a transient simulation, even if conditions for this have been specified in a checkpoint file.

Netlist Format

The *Netlist Format* radio buttons appear on the Environment Options form for the UltraSim circuit simulator.

➤ Select one of the following *Netlist Format* radio buttons:

- *spectre* specifies a Spectre format netlist

© 2023 Cadence Design Systems, Inc.

- *hspice* specifies an HSpice format netlist

Run Mode

The *Run Mode* radio buttons appear on the Environment Options form for the UltraSim circuit simulator.

➤ Select one of the following *Run Mode* radio buttons:

- *Batch*
- *Interactive*

  If you select *Interactive*, you must type a valid name in the *Interactive Control File* field.

© 2023 Cadence Design Systems, Inc.

Specifying Simulator Options

The options you can specify depend on your target simulator.

- Specifying Simulator Options for Spectre and UltraSimCircuit Simulators
- Specifying Options for AMS Circuit Simulator
- Specifying Options for hspiceD Circuit Simulator

Specifying Simulator Options for Spectre and UltraSimCircuit Simulators

To specify simulator options for Spectre, APS, or UltraSim circuit simulators, do the following:

1. On the Data View assistant pane, right-click the test and choose one of the following:

Menu Choice	Form
Options – Analog	The Simulator Options form appears

   The fields that appear on the form depend on your target simulator. Refer to your simulator's user guide for information about the various options you can select.

2. When you are done specifying options, click *OK*.

   For details of Spectre simulator options, refer to the Immediate Set Options (options) section in the Analysis Statements chapter of the Spectre Circuit Simulator Reference.

   Also see Environment Variables for Spectre Simulator Options Form in *Virtuoso ADE Explorer User Guide*.

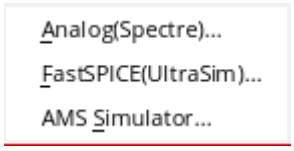
   For details of UltraSim simulator options, refer to Specifying UltraSim Options in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

Specifying Options for AMS Circuit Simulator

To specify options for AMS circuit simulator, do the following:

1. On the Data View assistant pane, right-click the test, choose *Options*, then choose one of the following:



   The appropriate form appears.

2. When you are done specifying options, click *OK*.

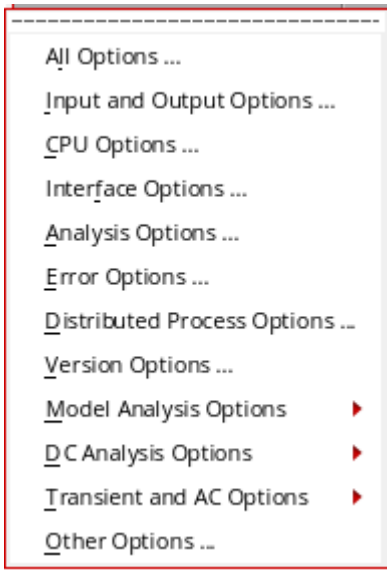
   For details on AMS simulator options, refer to Setting Up AMS Options in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

Specifying Options for hspiceD Circuit Simulator

To specify options for hspiceD circuit simulator, do the following:

1. On the Data View assistant pane, right-click the test, choose *Options*, then choose one of the following:



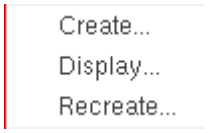
2. *Click All Options* to display the simulator options form.
3. When you are done specifying options, click *OK*.

   The simulator options you specified are applied to future simulations.

   For details on Hspice simulator options, refer to Working with Hspice Simulator in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

Creating and Viewing the Netlist



To create a netlist, do the following:

1. On the Data View assistant pane, right-click the test and choose *Netlist – Create*.

   ADE Assembler updates the netlist only for those cells that have changed since the previous netlist was created. Netlist is not created for the cells that are not checked and saved. Instead, their existing netlist is used. The netlist appears in a text browser window.

2. When you are done viewing the netlist, choose *File – Close Window*.

To recreate the netlist, do the following:

1. On the Data View assistant pane, right-click the test and choose *Netlist – Recreate*.

   ADE Assembler runs full hierarchical netlisting for all the schematics in the hierarchy. A message appears in your Command Interpreter Window indicating success or failure.

Important

If the `nlReNetlistAll` variable is set to `t` in the `.simrc` file, the command to create a netlist always recreates it.

To view the netlist, do the following:

1. On the Data View assistant pane, right-click the test and choose *Netlist – Display*.

   The netlist appears in a text browser window.

2. When you are done viewing the netlist, choose *File – Close Window*.

© 2023 Cadence Design Systems, Inc.

Editing Multiple Tests in ADE Assembler

Video

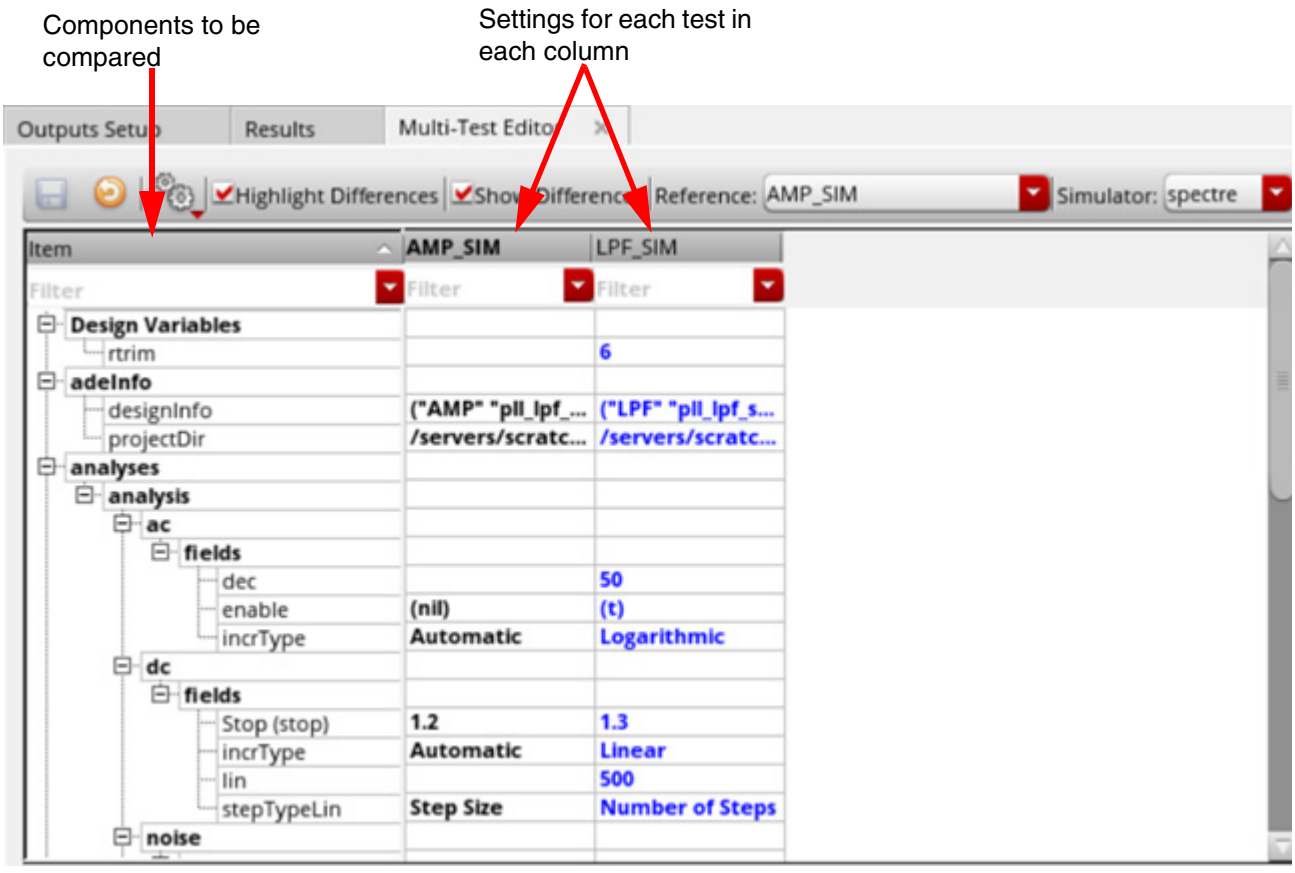
You can view a video demonstration of this feature at Using Multi-Test Editor in Virtuoso ADE Assembler.

After creating multiple tests in ADE Assembler, you can compare the settings of tests using Multi-Test Editor. Based on the comparison, you can copy the settings from one test to other tests.

To open the multi-test editor, do one of the following:

- Click *Tools – Multi-Test Editor*
- Right-click next to the *Results* tab and choose *Multi-Test Editor*

The Multi-Test Editor is displayed in a new tab, as shown below.



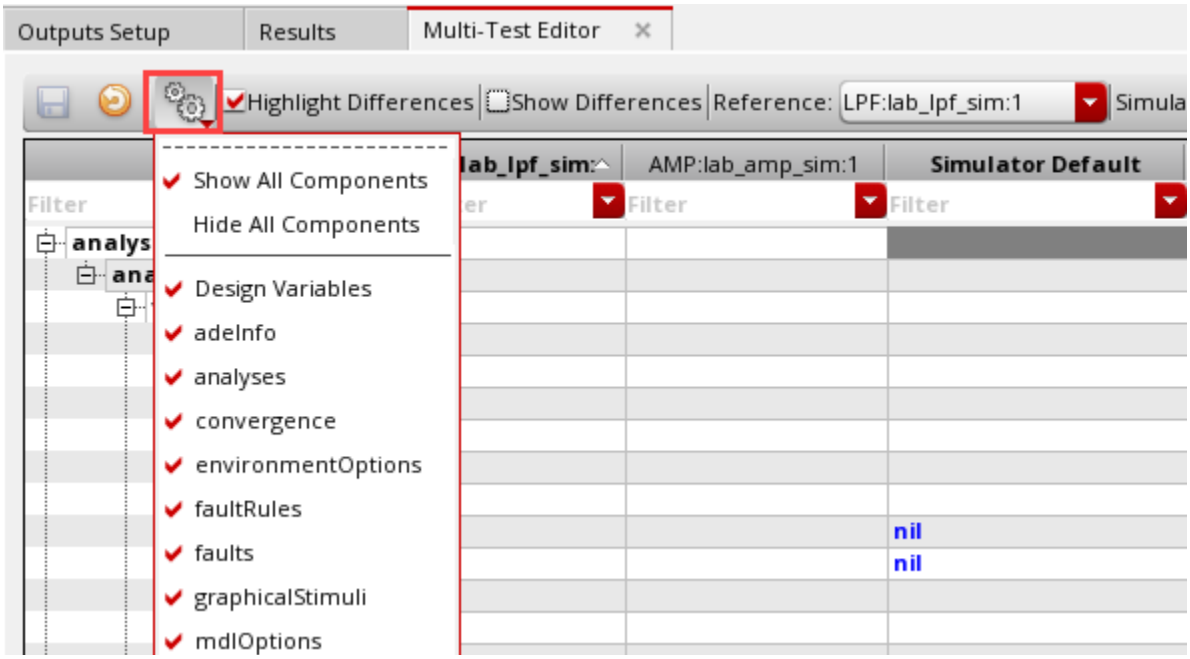
© 2023 Cadence Design Systems, Inc.

Multi-Test Editor User Interface

The Multi-Test Editor contains the following columns:

- ***Item***: Shows the names of items or settings to be compared. By default, this column lists all the components (or items) specified by the defaultComponents environment variable. You can edit this variable to modify the default items to be compared.

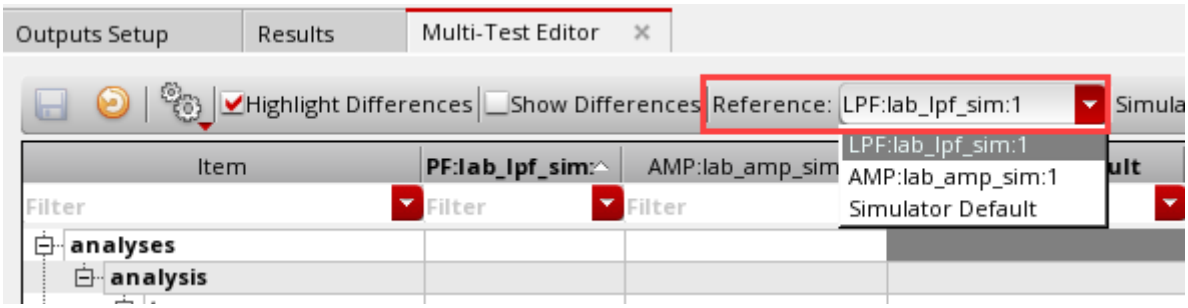
  While working in the multi-test editor, you can modify the items to be shown by using the *Select Options to Compare* drop-down list in the toolbar on the *Multi-Test Editor* tab.



- **A reference test** (the column displayed in bold to the right of *Item*): Shows the values set for the reference test, which is the test to which the settings of other tests are compared.

© 2023 Cadence Design Systems, Inc.

By default, the first column in the *Tests* tree in the *Data View* assistant is used as the reference test. You can change the reference test by selecting any other test name from the *Reference* drop-down list in the toolbar on the *Multi-Test Editor* tab.



Alternatively, you can drag a test and drop it on the current reference test to bring it to the left.

The *Reference* drop-down list shows only those columns that are visible in the table. Hidden columns are not listed.

- **One or more tests to be compared (columns for other tests placed to the right of the reference test)**: These are the columns to be compared with the reference test, or with other tests. By default, all enabled tests from the current maestro view and with simulator same as that of the reference test are shown in the multi-test editor. Their settings from the active setup are also loaded

  If you do not want to compare any specific test, you can hide the column by right-clicking on the header of the table and clearing the check box to the left of the test name. Whenever required, select the check box to unhide the test in this table.

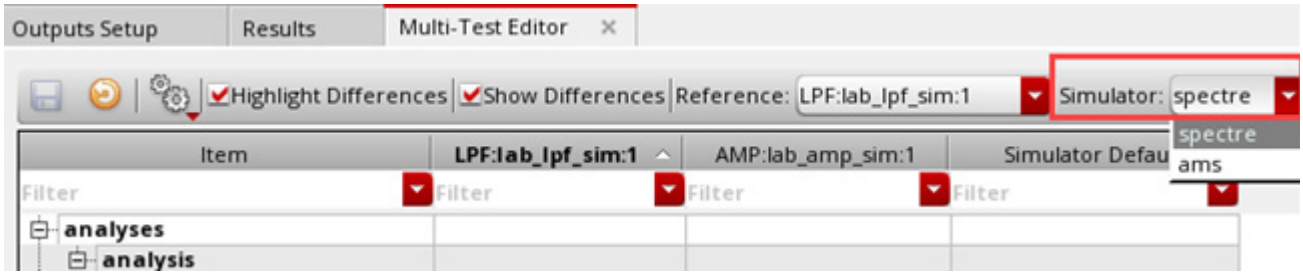
  Note: You cannot compare tests from other maestro views.

- ***Simulator Default***: Shows the default values set for the selected simulator. This column is hidden by default. To view it, right-click on the gray header of the Multi-Test Editor table and select *Simulator Default*.

  By default, the tests that use the simulator same as that of the reference test are shown in the table. If the maestro view contains tests that use a different simulator, choose another simulator name from the *Simulator* drop-down list on the toolbar of the

© 2023 Cadence Design Systems, Inc.

Multi-Test Editor tab to view the tests with use the selected simulator. Columns for other tests are hidden.



Important Points to Note

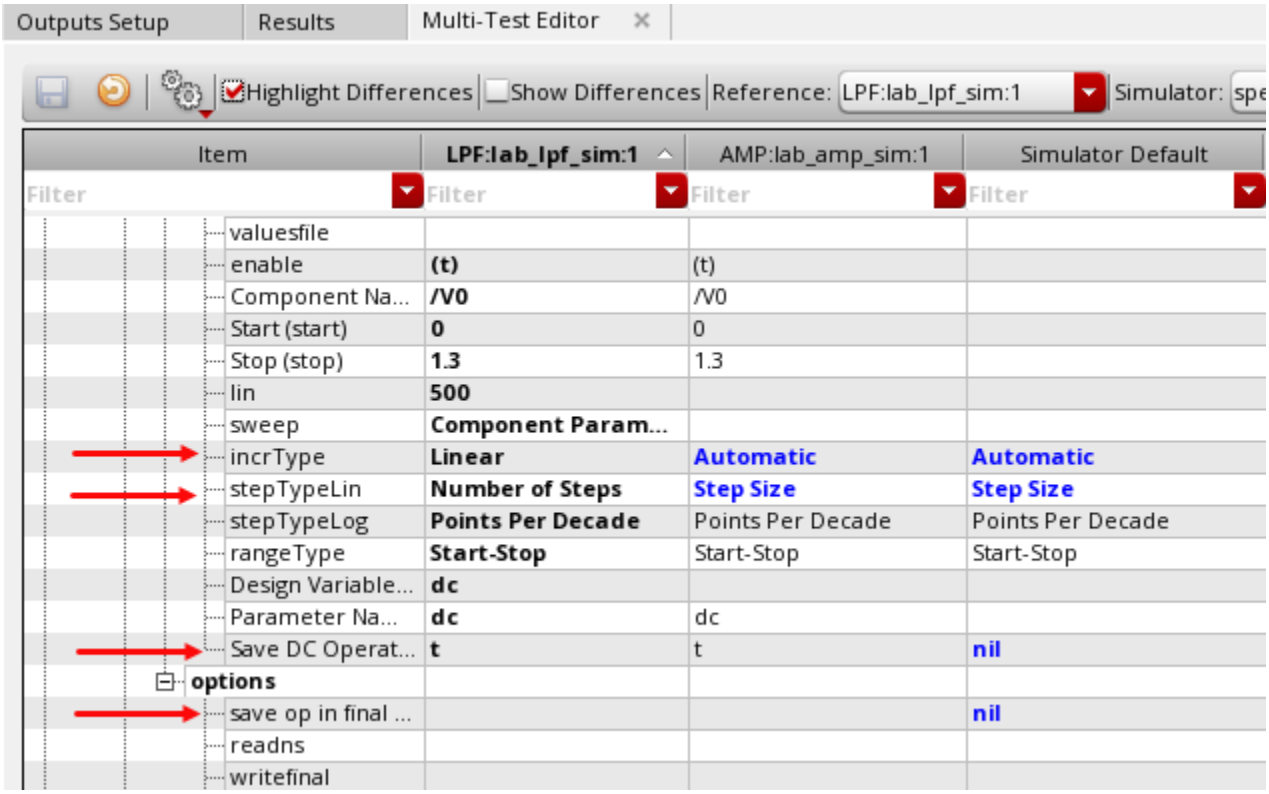
- Every time you open the *Multi-Test Editor* tab, the latest data from the active setup is displayed in this tab.
- The placement of the *Item* column is frozen. You can drag and drop other columns to change their location or use the horizontal scroll bar to view those.
- The first item, *adeInfo*, displays the design information, including the library, cell, view, simulator, and the path to the project directory. This item lets you to compare the design and simulator for each test.
- You can use the following commands in the context-menu of items displayed in the *Items* column to expand or collapse the rows for that item:
  - *Expand Deep*: Expands all the items currently displayed in the Multi-Test Editor.
  - *Collapse*: Collapses the selected item
  - *Expand All*: Expands the selected item. Displays the details of all sub items.
  - *Collapse All*: Collapses all the items currently displayed in the Multi-Test Editor.

Comparing and Modifying Test Setup

The rows in the Multi-Test Editor show the values set in each test for the listed items or components. While comparing the tests, you can make use of the following features of this editor:

© 2023 Cadence Design Systems, Inc.

- The *Highlight Differences* check box on the toolbar: Use this to highlight the rows with differences in values across the tests in blue, as shown below.



- The *Show Differences* check box on the toolbar: Use this to show only those rows in which the tests have different values set for the items. Other rows are hidden.
- The filters in each column: Specify filter criteria in the filter cells given on top of each column to show rows that contain the specified values or keywords. You can customize the filters by using the context-menu of the filter cells.

  Note: Filters expand the trees in the *Item* column to the level where the values meeting the filter criteria is met.

  A history of filters is saved for each column. Therefore, you can reuse a filter from the drop-down list in a filter cell.

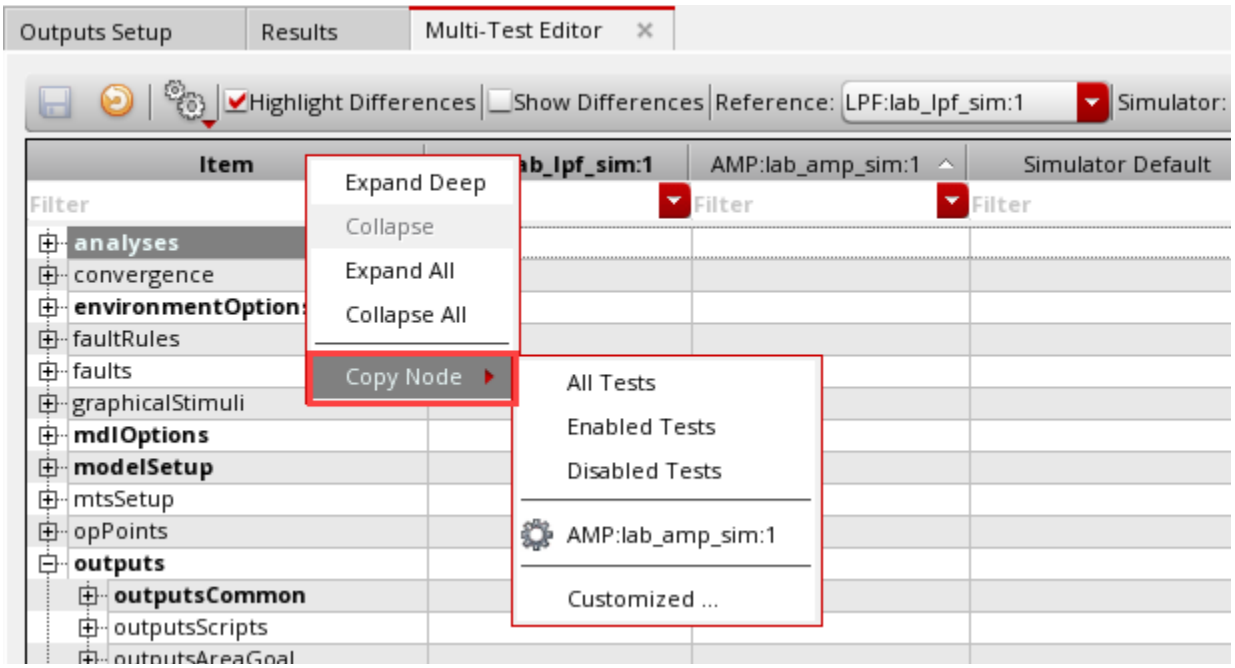
  To remove column filters, click . The table returns to the same state as it was before applying the filter.

© 2023 Cadence Design Systems, Inc.

Copying Values from Reference Test to the Other Tests

You can copy the values from the referenced test to one, more, or all compared tests, in the following ways:

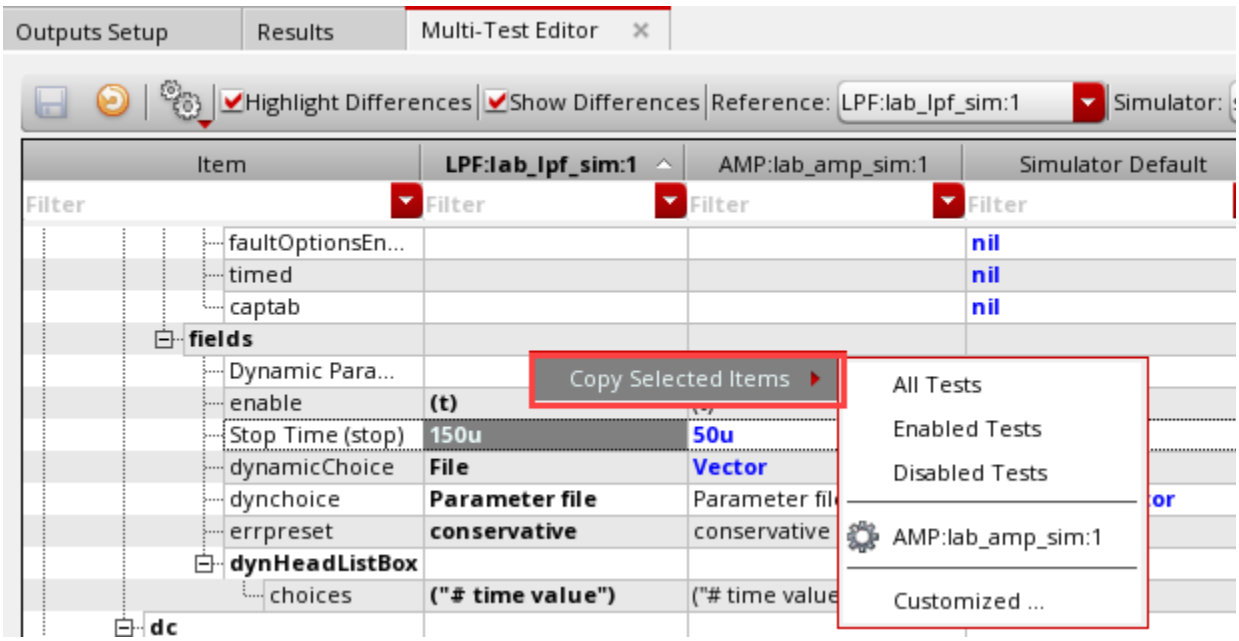
- To copy all settings from an item category, which is also referred to as a node, use the *Copy Node* command from the context-menu of that node. For example, in the setup shown below, to copy all analyses details from the reference test to test `AMP_lab_amp_sim:1`, right-click the ***analyses*** node and choose the test name from the context-sensitive menu.



Note: The Multi-Test Editor does not support modification in the adeInfo, convergence and output details. Therefore, the *Copy Node* command is not available for the *adeInfo*, *convergence* and *outputs* items.

© 2023 Cadence Design Systems, Inc.

- To copy only selected settings of a node, use the *Copy Selected Items* command from the context-menu of the cells.

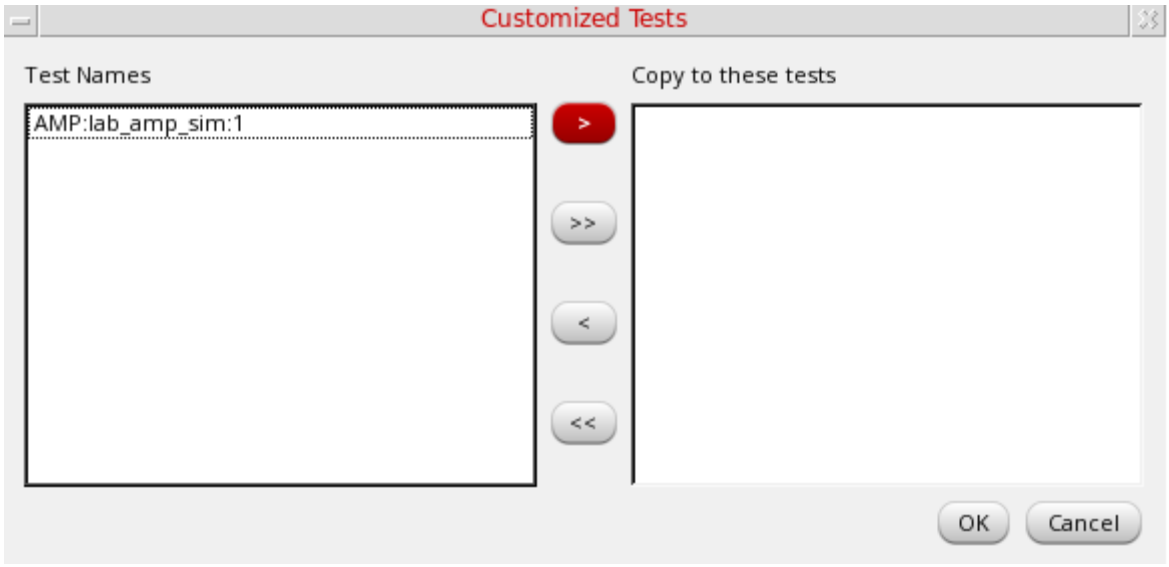


The *Copy Node* and *Copy Selected Items* commands provide the following options:

- *All Tests*: To copy values to all tests currently displayed in the multi-test editor.
- *Enabled Tests*: To copy values to all tests currently enabled in the active setup.
- *Disabled Tests*: To copy values to all tests currently disabled in the multi-test editor.
- Names of tests currently displayed: To copy values to a selected test

© 2023 Cadence Design Systems, Inc.

- *Customized*: To copy values to one or more selected test. This command opens the Customized Tests form using which you can select tests to which you need to copy the values.



Important

You cannot modify the values in the *Simulator Default* column.

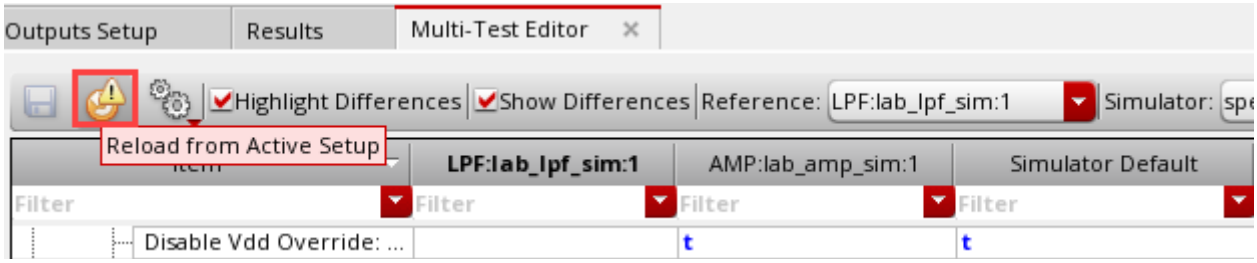
© 2023 Cadence Design Systems, Inc.

Keeping the Active Setup and Multi-Test Editor in Sync

The Multi-Test Editor allows two way synchronization:

- To reflect the changes in values from the active setup to the columns for tests

  After opening the Multi-Test Editor, if you make any change, such as a change in the test name, project directory or an analysis setting in the active setup, ADE Assembler shows a warning sign on the *Reload from Active Setup* command on the toolbar of the Multi-Test Editor.



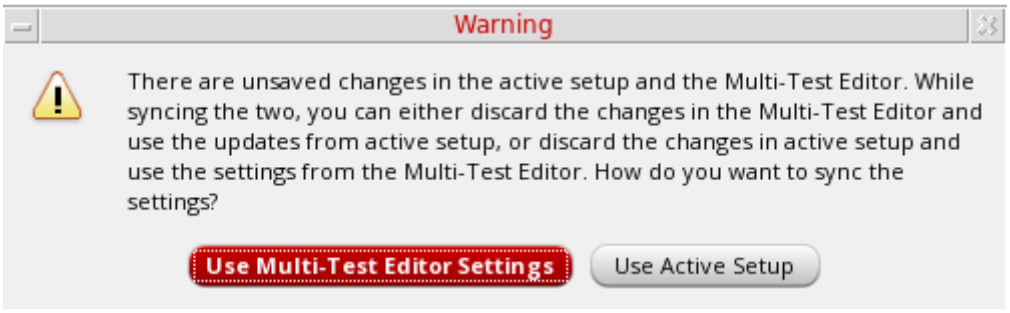
  Use this command to refresh Multi-Test Editor and to display the latest values from the active setup.

- To reflect the changes in values from the test columns in the Multi-Test Editor

  If you make any change in the Multi-Test Editor by copying values from the reference test to other tests, the *Save* command on toolbar of this tab becomes active. Use this command to reflect the changes from the Multi-Test Editor in the active setup.

  If you make changes in both the active setup as well as in the Multi-Test Editor, ADE Assembler shows a warning sign on the *Reload from Active Setup* command on the toolbar of the Multi-Test Editor.

  When you click *Reload from Active Setup*, ADE Assembler shows a warning message to inform you that changes have been done in both.



  The same warning message is displayed if you do not click *Reload from Active Setup*, but do any of the following:

© 2023 Cadence Design Systems, Inc.

- Close the Multi-Test Editor
- Switch to the *Results*, *Outputs Setup* or *Run Preview* tab
- Run a simulation
- Close the maestro cellview
- Exit Virtuoso

In the warning message, click an appropriate button to specify whether you want to revert to the active setup or you want to save the changes in Multi-Test Editor.

© 2023 Cadence Design Systems, Inc.

Modifying a Test in ADE Explorer

Video

You can view a video demonstration of this feature at Transitioning between ADE Assembler and Explorer.

Editing a test in ADE Explorer is helpful in the cases when you want to modify the setup and run simulations for a single test. In this application, you can perform various tasks, such as choosing a design, updating design variables, setting up analyses and output variables, running simulations, and performing real-time tuning.

- Steps to Modify a Test in ADE Explorer
- License Requirements for Transition
- Important Points to Note

Steps to Modify a Test in ADE Explorer

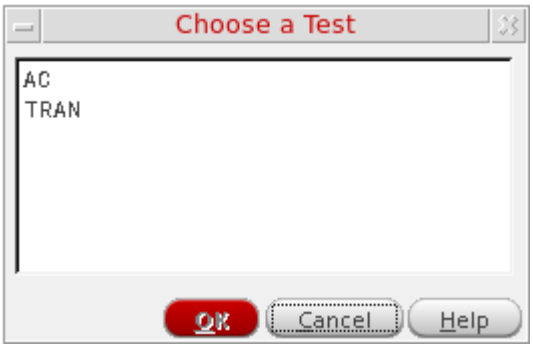
To open the ADE Explorer window for a test, do one of the following:

➤ Click the down arrow ( ) icon to the left of the test name in the *Tests* tree on the Data tab of the Data View assistant pane.

➥ Double-click a test name quickly (Double-clicking the test name slowly enables editing of the test name.) Also see: openExplorerOnTestDoubleClicked

➥ Right-click the test name and choose *Open Test in Explorer*

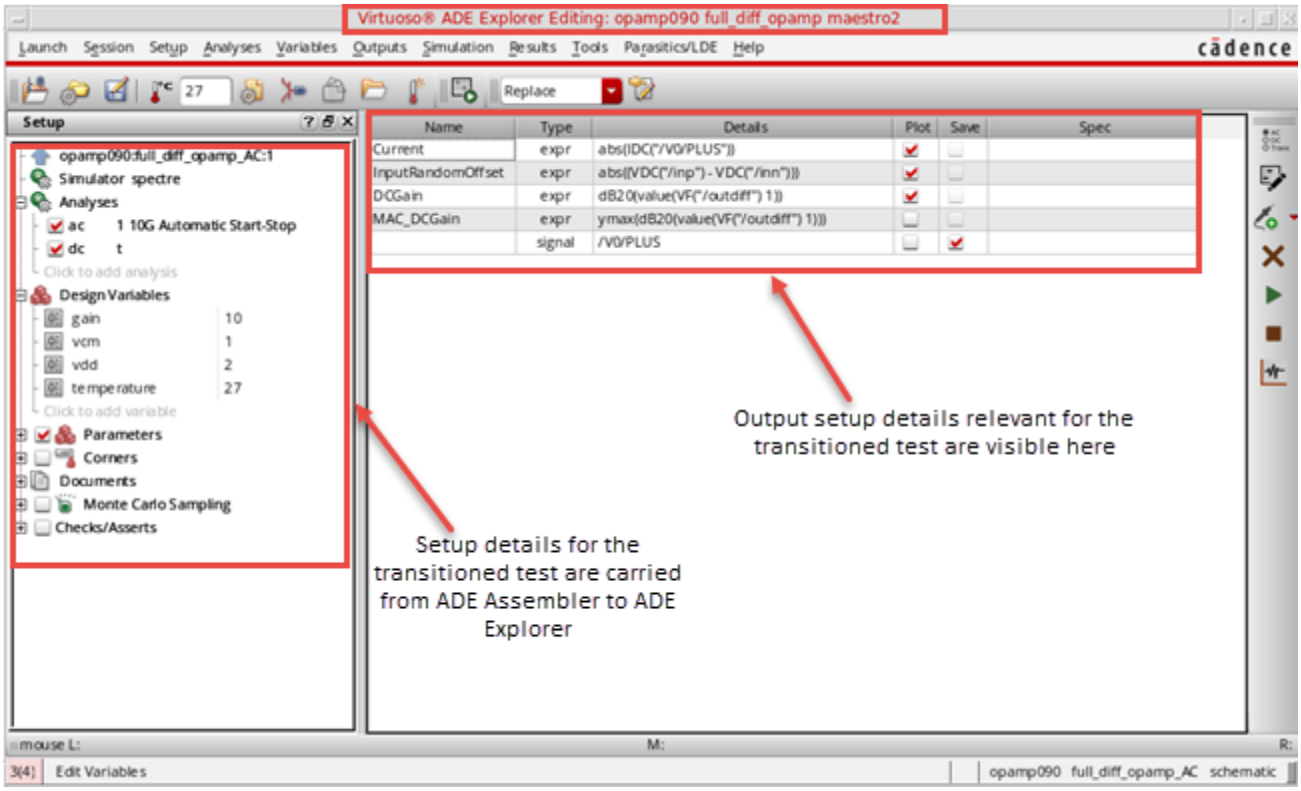
➥ Choose *Launch – ADE Explorer.* If there are multiple tests in ADE Assembler and you choose *Launch – ADE Explorer* to open a test, Virtuoso displays the Choose a Test form that shows a list of tests available in the view.



© 2023 Cadence Design Systems, Inc.

Select the name of the test you want to open in ADE Explorer and click *OK* to open the view. The OCEAN-based tests do not appear in the list of available tests on this form.

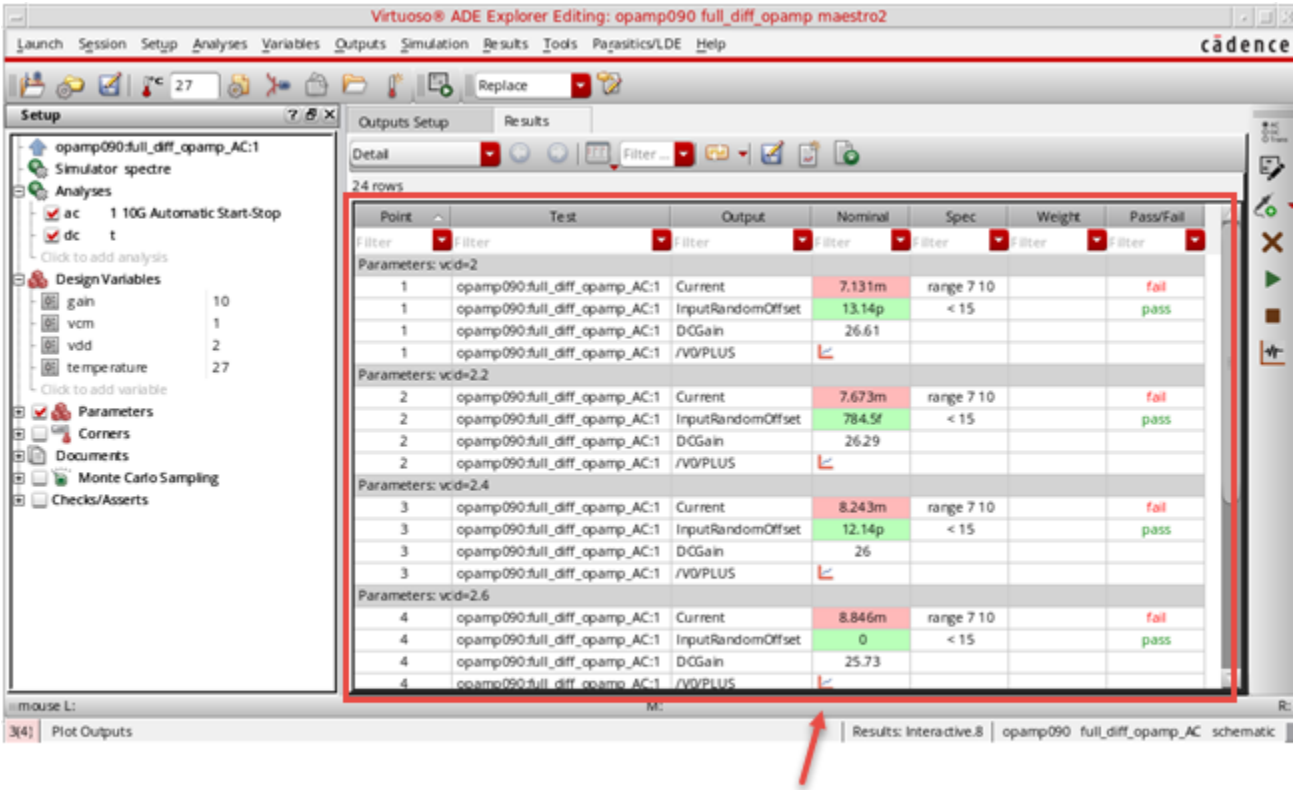
The test is opened in the ADE Explorer window, as shown below.



If simulation results are displayed in the *Results* tab of ADE Assembler, they are visible in the ADE Explorer window too. However, if no results are displayed in ADE Assembler, you will

© 2023 Cadence Design Systems, Inc.

see only the output setup in ADE Explorer. If required, you can load the results from a history that was earlier saved from ADE Explorer.



After the test is visible in the ADE Explorer window, you can edit it, run simulations and view output results and plots, or perform real time tuning as required. For details on the ADE Explorer environment, refer to the *Virtuoso ADE Explorer user guide*.

To move back to ADE Assembler, click the up arrow ( ) icon to the left of test name. Any changes done in the test setup using ADE Explorer are reflected in ADE Assembler.

License Requirements for Transition

As described in the steps above, using the arrow icons to the left of test names, you can transition from Assembler to Explorer, and then from Explorer to Assembler. Once you open Assembler, the Assembler license is checked out. After this, if you transition back to Explorer, the Assembler license that is already checked out is used. Any additional license for Explorer is not required.

© 2023 Cadence Design Systems, Inc.

Important Points to Note

- The setup in ADE Explorer displays only those options that are relevant for this test. For example, only those corners are listed under *Corners* that are enabled for the test displayed in ADE Explorer. Corners that were enabled only for other tests in ADE Assembler are not displayed in ADE Explorer.
- Global variables are not supported in ADE Explorer. Therefore, when a test is opened for editing in ADE Explorer, the variables shown in the Setup assistant and used for simulations are the local design variables of that test.

  However, if you set the `autoSyncGlobalVarsToTestVars` environment variable to `t`, the global variables are also taken to the ADE Explorer setup when you open a test for editing. If you change a design variable in ADE Explorer and then go to ADE Assembler, only the local value of design variable for that test is updated. While changing a variable value in ADE Explorer, the value of corresponding global variable is not updated; however, a yellow highlight appears on that global variable indicating that it has been overridden by the test-specific local value.
- If you add a new design variable for the test in ADE Explorer, it is shown in the list of design variables for that test in ADE Assembler. The new variable is not added to the list of global variables. You can use the *Create/Update Global* command in the pop-up menu of that variable to promote it to the global level. Alternatively, you can set the autoPromoteVarsToGlobal environment variable in `maestro.cdsenv` to `t` to automatically promote all the design variables as global variables.
- If a local design variable for the test contains sweep values in the Data View assistant of ADE Assembler, the sweep set is taken to ADE Explorer. Simulations run from ADE Explorer also use the swept values. If you make any changes in the sweep set, they are copied back to the test setup in ADE Assembler.
- Even if a test in the ADE Assembler setup uses the value specified for the corresponding global variable, only the values specified in the design variables are copied to ADE Explorer. Any changes made to its values are copied back to the design variable for that test.

  Consider an example of a test `test1` that has a design variable `CAP` and its value is set to `200u`, but it uses the value specified for the global variable `CAP` that is set to 250u. When you edit `test1` in ADE Explorer, the value for `CAP` will be `200u`, and not `250u`. Now, if you change this value to `220u` in ADE Explorer, the same will be copied back to ADE Assembler when you move the test there.
- New corners created for a test being edited in ADE Explorer will be enabled only for that test when the setup is copied back to ADE Assembler. These corners will not be enabled for other tests in ADE Assembler.

© 2023 Cadence Design Systems, Inc.

The new corners in ADE Explorer automatically get unique names. While adding the default sequence number suffix to a corner name, ADE Explorer ensures that the new corner name is not same as that of an existing corner in ADE Assembler.

- If a test uses any setup attribute that is specific to ADE Assembler, it is not reflected in ADE Explorer. For example, if the run mode in ADE Assembler is set to `Global Optimization`, which is not supported in ADE Explorer, the test setup in ADE Explorer will use the default run mode, `Single Run, Sweeps, and Corners`.

  Similarly, if the setup contains CONFIG variables or pre-run scripts, when you open it in ADE Explorer, a message is displayed prompting you to disable these attributes in ADE Assembler and then reopen the test in ADE Explorer to run simulations.

- If a test in ADE Assembler uses an overridden job policy, its job setup is also copied to ADE Explorer and it does not use the default job policy in ADE Explorer. If you make any change in the job setup, the changes are copied back to that test setup in ADE Assembler.

- When you transition from ADE Assembler to ADE Explorer, the *Checks/Asserts* check box is checked in the Setup assistant of ADE Explorer only if both of the following check boxes are selected in ADE Assembler:

  - The *Checks/Asserts* check box in the Data View assistant
  - The check box corresponding to the test sub tree under *Checks/Asserts*

- If you had run a simulation in ADE Explorer, that is also listed on the History tab of the Data View assistant In ADE Assembler. The history is named as `ExplorerRun.0`.

- You cannot open an OCEAN-based test in ADE Explorer. If you click the down arrow ( ) icon for an OCEAN-based test in the Data View assistant pane, it is opened in the OCEAN Test Editor window.

Working with OCEAN-Based Tests

In ADE Assembler, you can save the simulation setup in OCEAN scripts and run these scripts either from UNIX shell or from Virtuoso GUI.

For more details on creating and running OCEAN scripts, refer to Creating and Running an OCEAN Script.

© 2023 Cadence Design Systems, Inc.

3

Working with Design Variables and Instance Parameters

You can use design variables and Component Description Format (CDF) instance parameters to set component values for simulation. Design variable values apply to specific tests. The scope of a CDF parameter value depends on which Analog Expression Language (AEL) functions you use to refer to the parameter.

See the following topics for more information:

- Working with Design Variables
- Working with Device Instance Parameters

© 2023 Cadence Design Systems, Inc.

Working with Design Variables

You can add, change, and delete design variables on the Data View pane or the Variables tab of the Variables and Parameters pane in your ADE Assembler environment.

- Adding a Design Variable
- Changing the Value of a Design Variable
- Hiding Overridden Design Variables for a Test
- Deleting a Design Variable
- Changing the Order of Design Variables
- Enabling or Disabling Design Variables
-
- Saving Design Variables
- Toggling the View on the Variables tab of the Variables and Parameters Assistant Pane
- Finding Design Variables in Schematic
- Copying Design Variable Values to the Schematic
- Copying Design Variable Values from the Schematic
- Importing Design Variables from an ADE State
- Displaying Design Variables on the Schematic
- Defining Variables in a File
- Adding Variable Tags

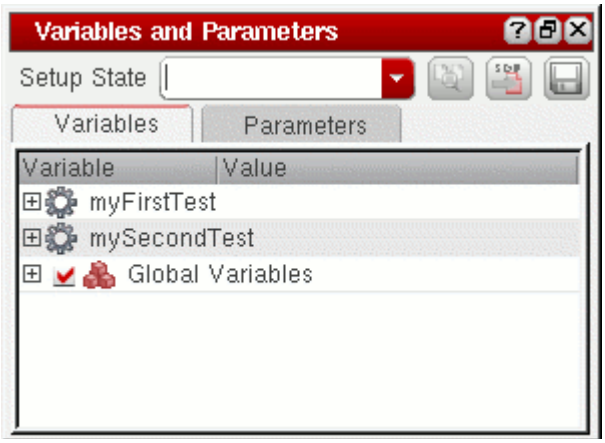
© 2023 Cadence Design Systems, Inc.

Adding a Design Variable

To add a design variable to a test, do the following:

1. Choose *Window – Assistant – Variables and Parameters*.

   The Variables and Parameters pane appears.



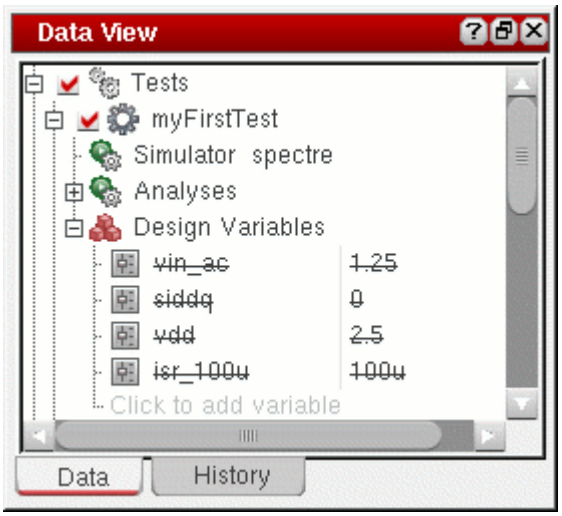
2. On the Variables tab of the Variables and Parameters pane, right-click the test and choose *Add Variable*.

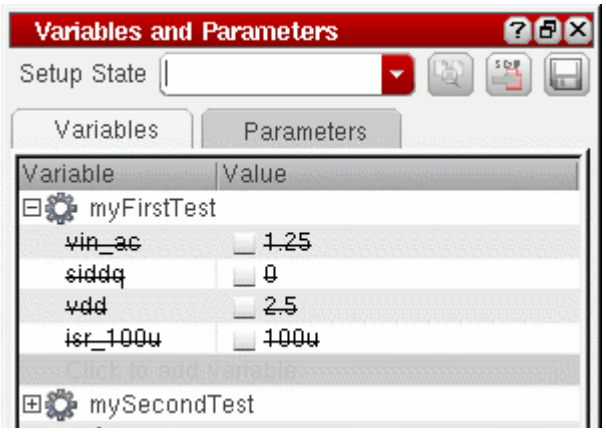
Tip

Alternatively, you can do one of the following:

© 2023 Cadence Design Systems, Inc.

- Click where it says *Click to add variable* in an expanded test tree on the Data View pane or on the Variables tab of the Variables and Parameters pane.



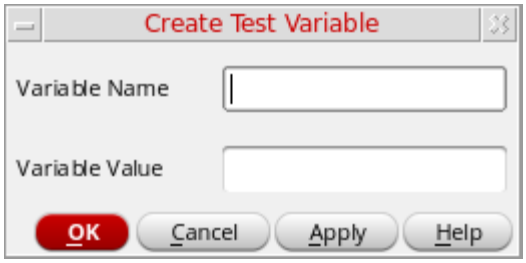


- In an expanded test tree on the Data View pane, right-click a design variable and choose *Add Variable*.

  Alternatively, click *Click to add variable*.

© 2023 Cadence Design Systems, Inc.

The Create Test Variable form is displayed.



3. In the *Variable Name* field, type a name for your design variable.

   The name must begin with a letter or the underscore character (_), and can contain only letters, numbers, or underscore characters.

4. In the *Variable Value* field, type a value or expression for your design variable.

   The expression can be an equation, a function, or another variable. The simulator evaluates the expression which must follow Analog Expression Language (AEL) syntax.

   Values from the results of one test can be used in the expressions for the local variables of another test. For this, use the calcVal function. For example, if an output of `test2` is `output1`, as shown below.

   ```
   calcVal("output1" "test1")
   ```

   In such cases, simulations for test1 are run before other tests so that the results are available for calculations.

5. Click *OK* to close the form.

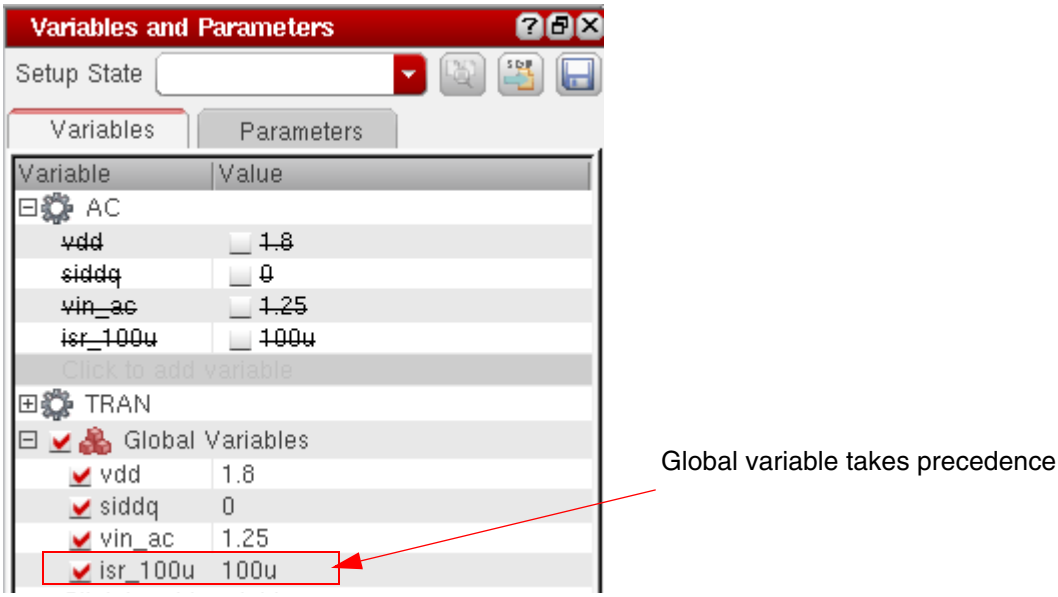
   Your design variable and its value appears in the expanded test tree on the Data View pane and on the Variables tab of the Variables and Parameters pane.

   Note: You can also define variables in a definitions file.

   By default, the design variables defined for your tests are not automatically promoted as global variables in the *Global Variables* tree on the Data View assistant pane and the Variables tab of the Variables and Parameters assistant pane. You can set the autoPromoteVarsToGlobal environment variable in `maestro.cdsenv` to `t` to automatically promote new design variables as global variables. The design variables are displayed with a ~~strikethrough~~ because the settings for a global variable override the settings of design variables that have the same name as the global variable. For example, in the following figure, the design variables specified for the test named `myFirstTest` are automatically added as global variables. The design variables are

© 2023 Cadence Design Systems, Inc.

displayed with a ~~strikethrough~~ because the global variables override the design variables.



Note: You can disable the default automatic copy of design variables to global variables by setting the autoPromoteVarsToGlobal environment variable to `nil`.

Video

You can view a video demonstration on how to use the `calcVal` function for a design variable at Using the calcVal SKILL Function in a Design Variable in the ADE Assembler.

Video

You can view a video demonstration on how to use the autoPromoteVarsToGlobal environment variable at Use the autoPromoteVarstoGlobal Environment Variable in the ADE Assembler.

Changing the Value of a Design Variable

By default, all the design variables use the value set for the corresponding global variables. However, if required, you can choose to use a different value for a design variable for a specific test.

If you want to change the value of a design variable that is overridden by a global variable, you must do one of the following:

© 2023 Cadence Design Systems, Inc.

- Disable the global variable for test in which you want to change the value of the design variable.

  For more information, see Disabling Global Variables for Specific Tests.

- Disable the global variable for all tests.

  For more information, see Enabling and Disabling Global Variables for All Tests.

To change a design variable, do the following:

1. In an expanded test tree on the Variables tab of the Variables and Parameters pane, double-click in the *Value* cell of the design variable you want to change.

Tip

Alternatively, in an expanded test tree on the Data View pane, double-click the value of the design variable you want to change.

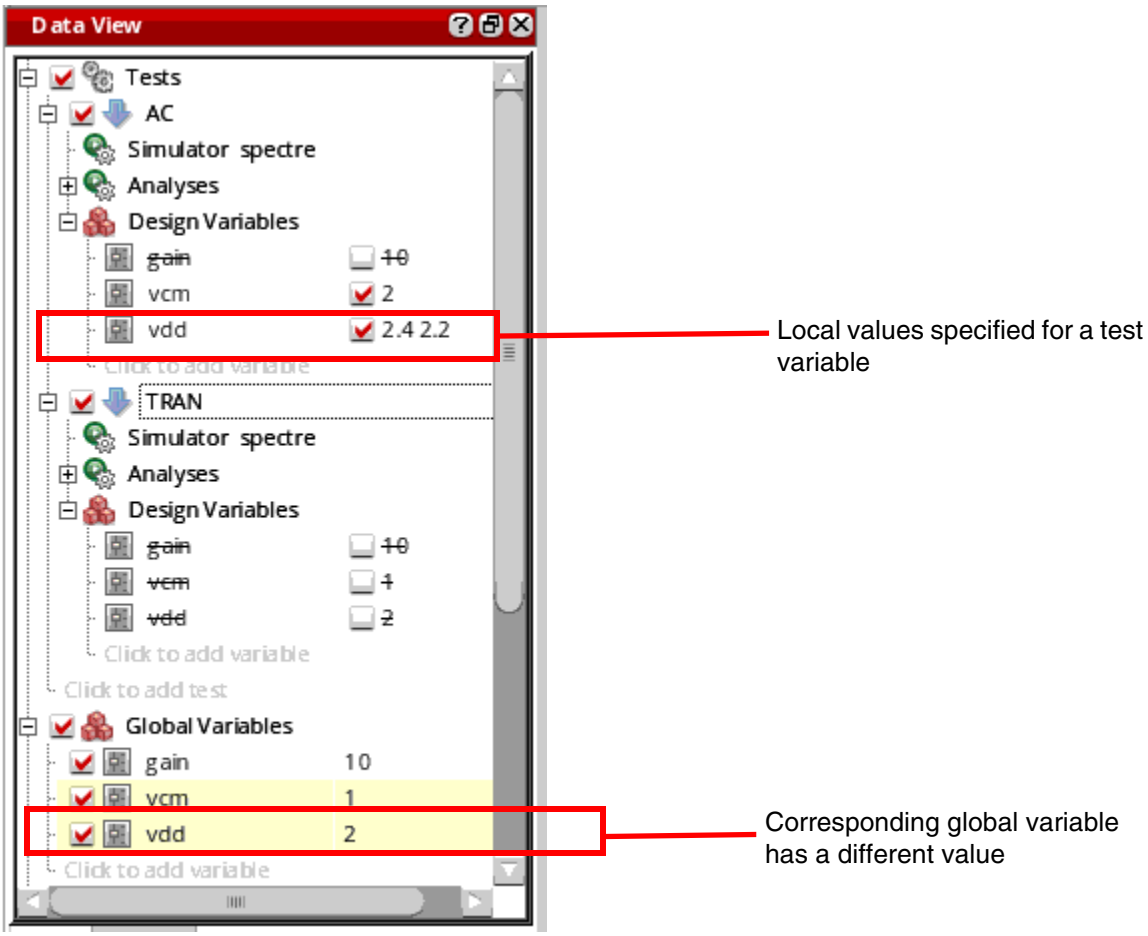
2. Type a new value or expression for the design variable.

   If required, you can specify multiple values, a range, or an inclusion or exclusion list for a design variable. For details, refer to Adding or Changing a Variable or Parameter Specification.

3. Press *Return* or click anywhere outside that table cell.

© 2023 Cadence Design Systems, Inc.

The new value appears in the *Value* cell for that design variable.



Alternatively, you can change the value of a design variable in the Editing Design Variables form as follows:

1. In an expanded test tree on the Data View pane or on the Variables tab of the Variables and Parameters pane, right-click the design variable and choose *Edit Variable*.

   The Editing Design Variables form appears.

2. In the *Design Variables* table, select the design variable you want to change.

   The row appears highlighted and the *Selected Variable* fields, *Name* and *Value (Expr)*, contain the name and value of the design variable you selected.

3. Click in the *Value (Expr)* field.

   The editing cursor appears where you clicked.

4. Type a new value or expression.

© 2023 Cadence Design Systems, Inc.

To specify a valid sweep value, ensure that:

- The value is not a blank string or does not contain a whitespace
- If contains a range, it has the start, step, and stop values separated by colons, commas, spaces, or a combination of these
- If specified as an expression, it is written in correct SKILL syntax and contains balanced parenthesis and quotes
- If contains a file path, it is enclosed in double quotes so that it is not evaluated

5. Click *Change*.

   The new value appears in the *Design Variables* table.

6. Click *OK*.

When a global variable is disabled for a test and the design variable uses sweep values, separate design points are created for that test on the Results tab. For more details on the results display, see Viewing Simulation Results When Design Variables Use Swept Values.

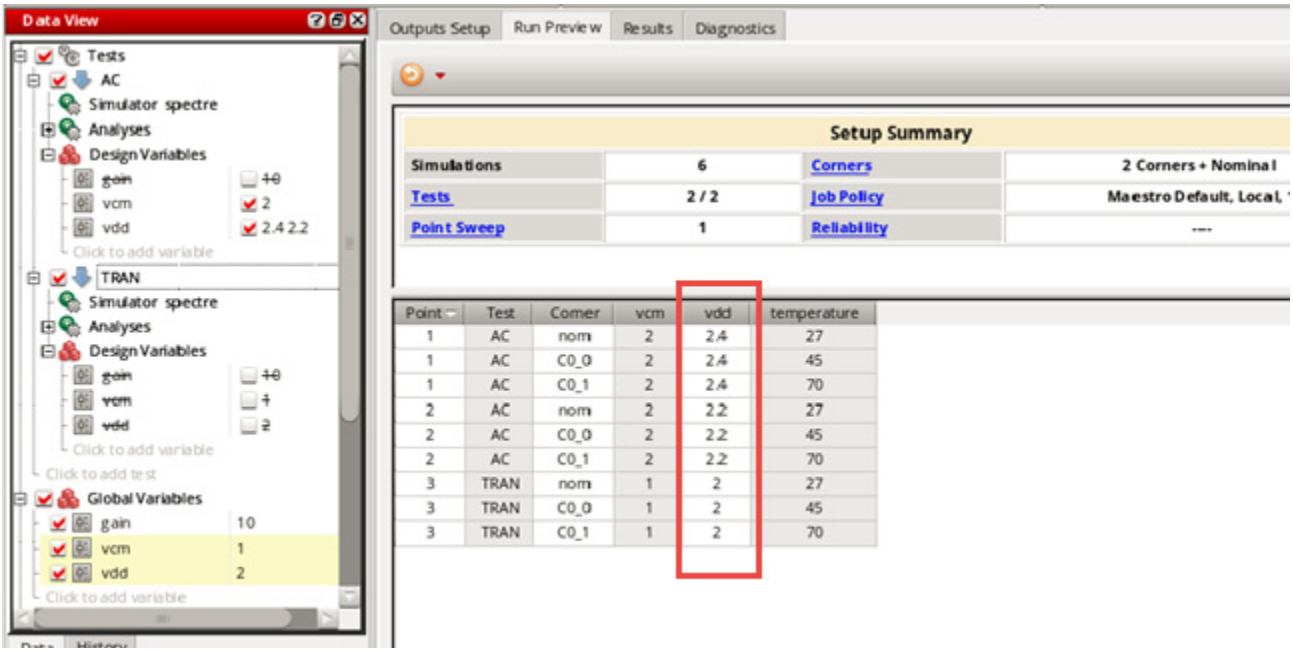
Video

You can view a video demonstration for this feature at Global and Design Variable Sweeps in the ADE Assembler.

© 2023 Cadence Design Systems, Inc.

Tip

Using the Run Preview tab you can verify the values of the swept variable being used for each design point, as shown below.



© 2023 Cadence Design Systems, Inc.

Using Expression Builder for Design Variables and Global Variables

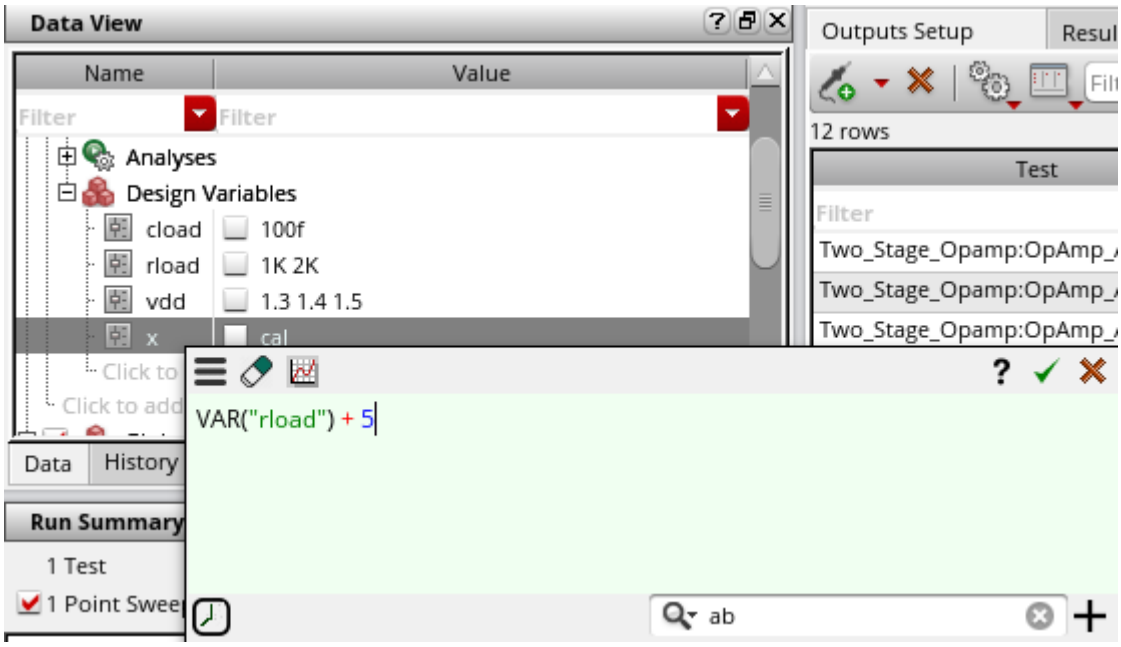
Design variables or global variables can contain static values or expressions that are evaluated to get values at the run time.

You can either directly type in expressions in the *Value* column for the variables or use the Expression Builder to use the expression building prompts.

1. Click in the *Value* column of a variable.
2. A text box is displayed to enter a value or expression. In addition, the *Open expression builder* ( ) command is also displayed to the right of that box.
3. Click *Open expression builder*.

   The expression builder is displayed.

   Build an expression in the space given on the expression builder, as shown below.



   If you use an existing function in the expression, the tool shows prompts to add argument values or to balance the braces.

   You can use `Alt + Enter` to enter line breaks while creating an expression in the expression builder. This improves readability, and makes it easy to edit long expressions.

   These line breaks in the expression are retained even when you close and reopen Expression Builder. Line breaks in the expression are retained also when outputs are

© 2023 Cadence Design Systems, Inc.

exported to the CSV file. When you import these outputs using the CSV file again, line breaks in the expression remain as they were earlier.

4. After the expression is complete, click ✓ on the expression builder.

   The expression builder is closed and expression is displayed in the *Value* column.

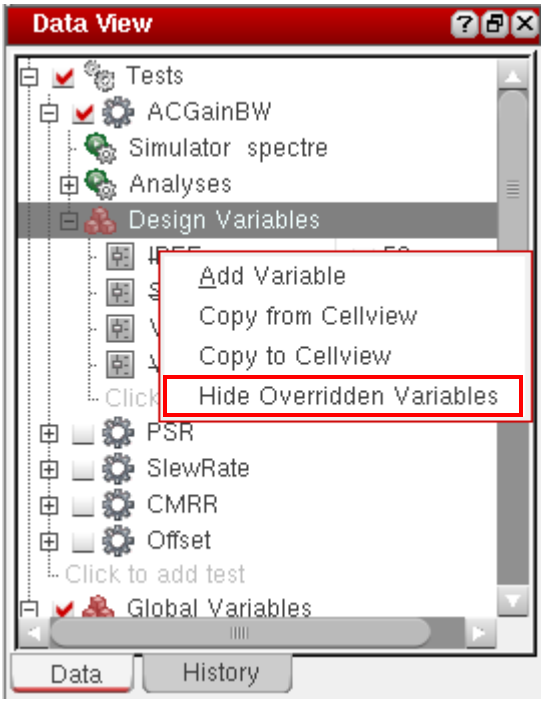
© 2023 Cadence Design Systems, Inc.

Hiding Overridden Design Variables for a Test

If the autoPromoteVarsToGlobal environment variable is set to `nil`, a design variable for a test is automatically promoted as global variables and the variable appears with a ~~strikethrough~~ in the list of variables for that test.

You can choose to hide all the overridden variables for a test and to display only those variables for which the value specified for a test is used. For this, do any one of the following:

- In the Data View pane, expand the design tree, right-click *Design Variables* and choose *Hide Overridden Variables*.



- On the Variables tab of the Variables and Parameters assistant, right-click a test name and choose *Hide Overridden Variables*.

In both the Data View pane and the Variables and Parameters assistant, the overridden variables are hidden for the test.

Deleting a Design Variable

To delete a design variable, do the following:

© 2023 Cadence Design Systems, Inc.

➤ In an expanded test tree on the Data View pane or on the Variables tab of the Variables and Parameters pane, right-click the design variable you want to delete and choose *Delete Variable*.

Note: You can also delete design variables from the Editing Design Variables form.

© 2023 Cadence Design Systems, Inc.

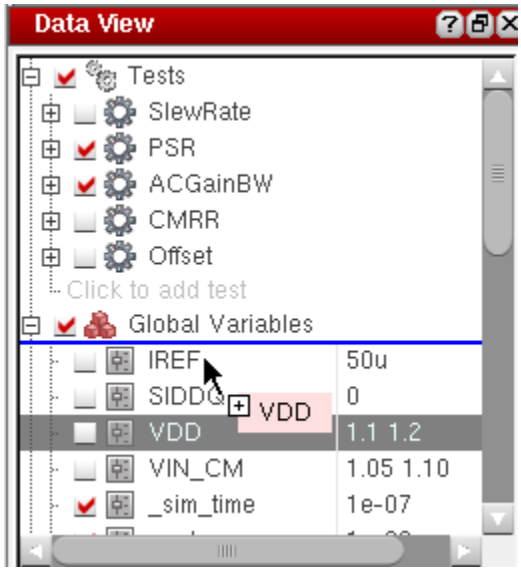
Changing the Order of Design Variables

You can change the order in which the design variables are listed in the *Data View* assistant. A change in the order of design variables in this assistant also changes the nesting of sweep points.

To change the order of design variables, do the following in the *Data* tab of the *Data View* assistant:

1. Expand *Global Variables* to view all the variables.
2. Click a variable name and drag it.

   A blue indicator line appears at the location where you have moved it. The tooltip also shows the name of the variable being moved.



3. Drag the indicator line to the desired location and drop the variable.

Note: You can hold down the *Ctrl* key to select multiple variables and then move them together.

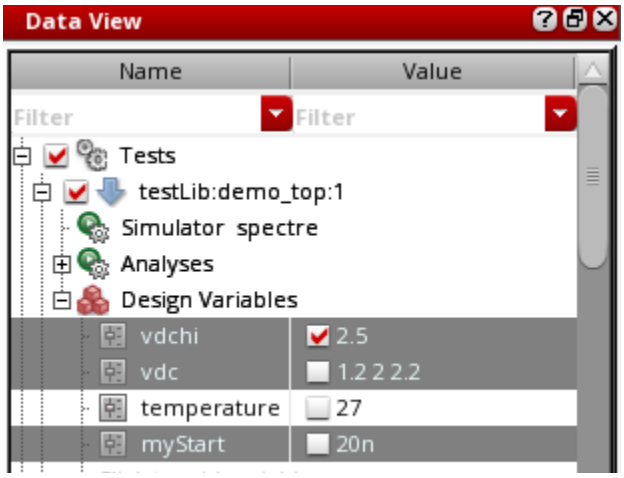
© 2023 Cadence Design Systems, Inc.

Enabling or Disabling Design Variables

To enable or disable the design variables:

1. In the Data View assistant, expand the *Design Variables* tree for a test.
2. Select one or more variables, right-click and choose *Enable/Disable* from the context-sensitive menu.
3. The current status of the selected variables is toggled.

For example, in the following scenario, the *Enable/Disable* command enables variables `vdc` and `myStart`, and disables `vdchi`.



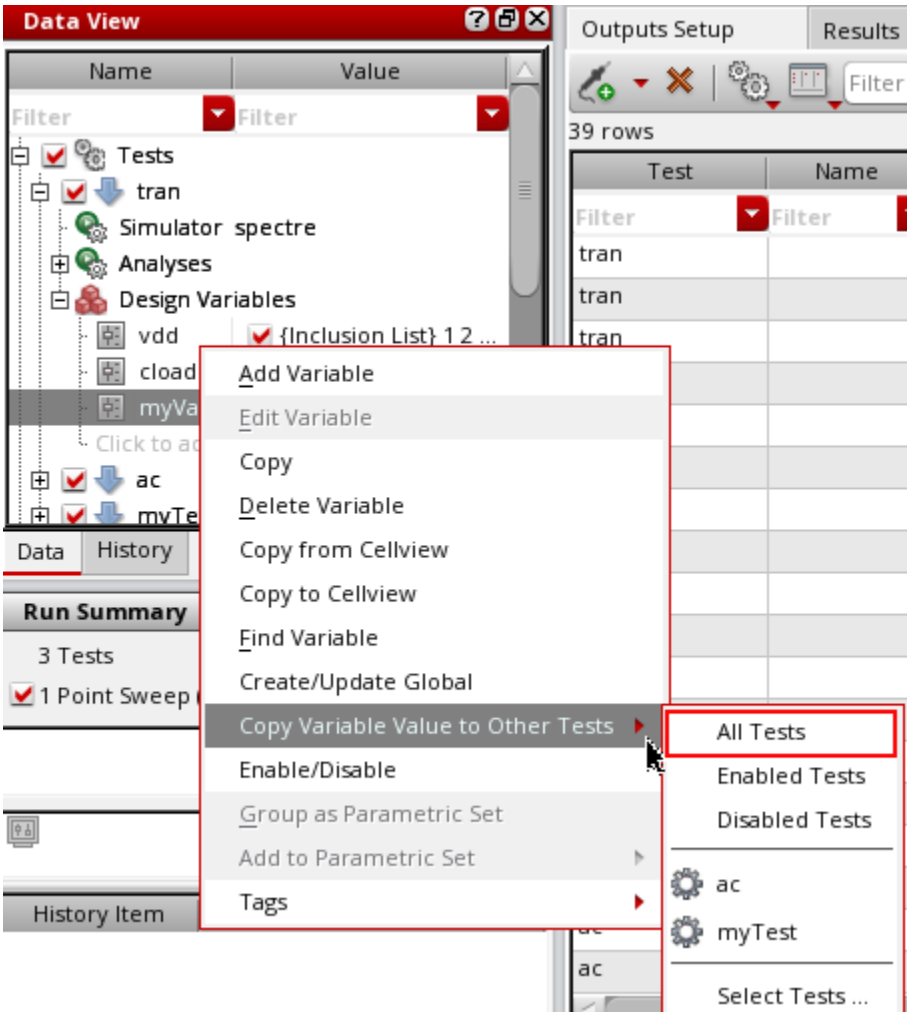
© 2023 Cadence Design Systems, Inc.

Copying Variable Values to Other Tests

If you have edited the values of variables in one test and want to use the same values in other tests, you can copy variable values to those tests. The *Copy Variable Value to Other Tests* command in the context menu of a design variable lets you copy values of variables to all or selected tests in the setup.

To copy the values of variables from one test to all other tests in the setup:

1. In the Data View assistant, select one or more variables in the source test from which you need to copy values.
2. Right-click one of the selected variables and choose *Copy Variable Value to Other Tests – All Tests*.



The values for the selected variables are copied from the source test to all other tests that have the same variable.

© 2023 Cadence Design Systems, Inc.

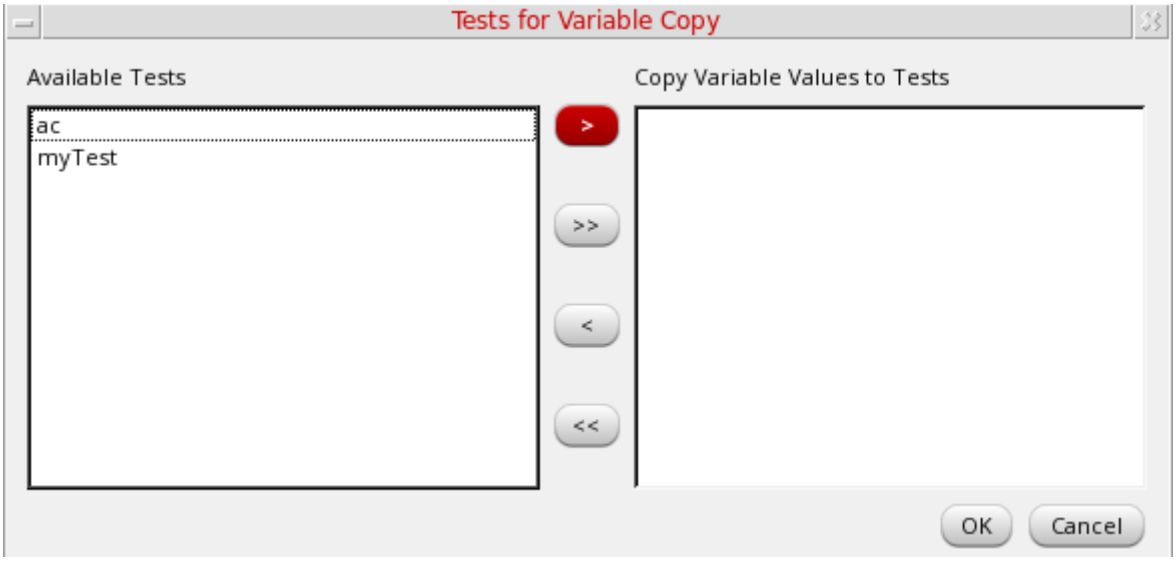
Similarly, you can copy values from one test to all enabled, all disabled, or a selected test by using the following commands in the same context menu of a design variable:

- *Copy Variable Value to Other Tests – All Enabled Tests*
- *Copy Variable Value to Other Tests – All Disabled Tests*
- *Copy Variable Value to Other Tests – `test-name`*

To copy the values of variables to multiple tests:

1. Select one or more variables in the source test from which you need to copy values.
2. Right-click one of the selected variables and choose *Copy Variable Value to Other Tests – Select Tests*.

   The Tests for Variable Copy form is displayed.



3. In the *Available Tests* list of this form, select the names of the tests to which you want to copy the variable values.
4. Click the right arrow to move the selected tests to the *Copy Variable Values to Tests* list.
5. Click *OK*.

   The values of the variables are copied to the selected tests.

Observe the following:

- ADE Assembler copies only the variable values. It does not copy the enabled or disabled status of the variable. To copy variables from one test to another, use Multi-Test Editor.

© 2023 Cadence Design Systems, Inc.

- The tool copies the variable values if the destination test has a variable with the same name. If no variable is found with the same name, the tool does not add a new variable to the test.

© 2023 Cadence Design Systems, Inc.

Saving Design Variables

Your design variables are saved with your ADE Assembler setup. To perform an explicit save, do the following:

➤ Choose *File – Save*.

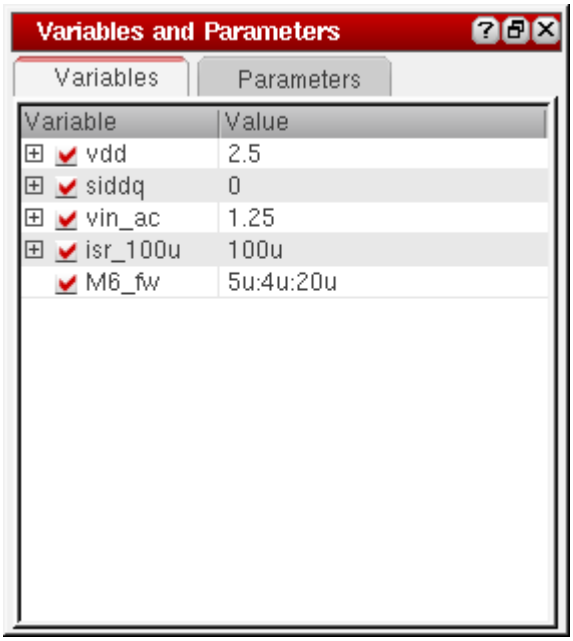
The program saves your setup information, including design variables. The program reloads these values the next time you open this design cellview.

Toggling the View on the Variables tab of the Variables and Parameters Assistant Pane

To toggle the view of variables on the Variables tab of the Variables and Parameters assistant pane, do the following:

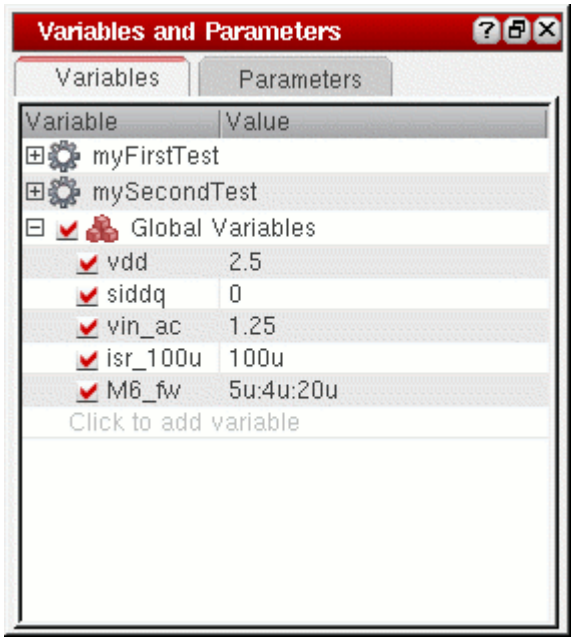
➤ Right-click any item and choose *Toggle View*.

If you were viewing the variable categories, the view changes to a tree of variables.



If you were viewing a tree of variables, the view changes to the variable categories view.

© 2023 Cadence Design Systems, Inc.



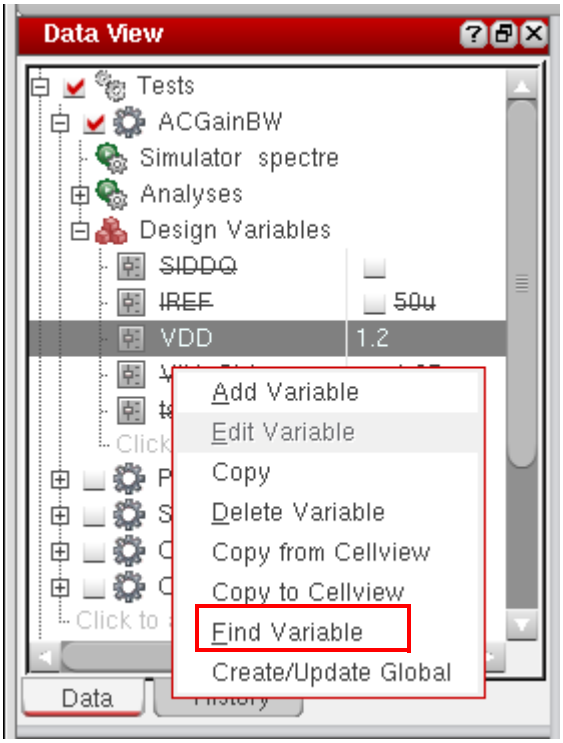
Finding Design Variables in Schematic

You can search for a design variable in the schematic to identify its location. To find a variable, do the following:

1. In the Data View pane, expand the test tree.
2. Expand the Design Variables tree.

© 2023 Cadence Design Systems, Inc.

Right-click a design variable and choose *Find Variable*.

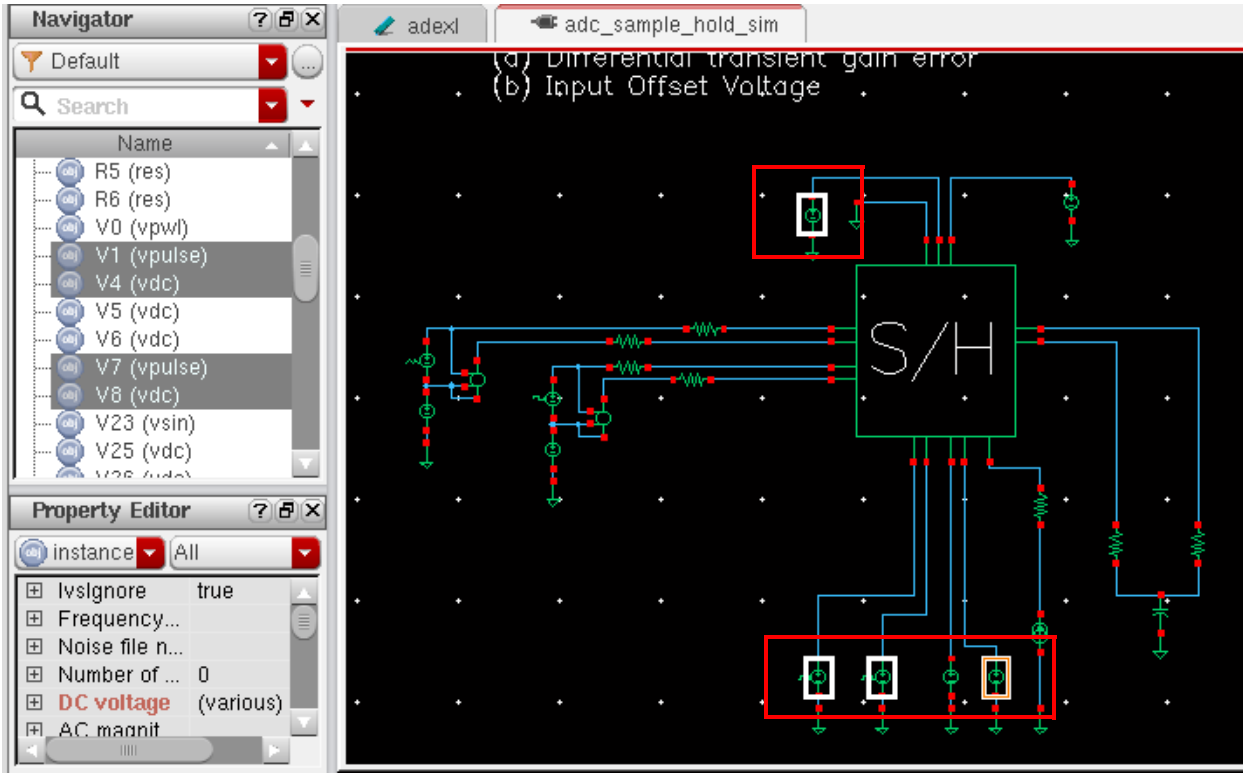


Alternatively, do the following on the Variables tab of the Variables and Parameters assistant:

1. Toggle the view to show the variables names. Expand a variable tree to show the test names.
2. Right-click the test name and choose *Find Variable*.

© 2023 Cadence Design Systems, Inc.

If the variable is found in the schematic, the design is opened in the design tab and all the instances are highlighted, as shown below.



In case of a hierarchical design, if the variable is not found in the top level of design, the tool traverses down the hierarchy and highlights the variable in the lower level, where found.

If the variable is not found in the schematic, a warning message, as shown below, is displayed in the CIW.

```
Variable '<var-name>' not found on the schematic.
```

Copying Design Variable Values to the Schematic

If you change design variables in your simulation setup and want to copy the values back to the cellview before you save the schematic, do the following:

➤ On the Variables tab of the Variables and Parameters pane, right-click the test and choose *Copy to Cellview*.

© 2023 Cadence Design Systems, Inc.

Tip

Alternatively, in an expanded test tree on the Data View pane, right-click a design variable and choose *Copy to Cellview*.

The program copies design variable values from the Variables tab of the Variables and Parameters pane to the cellview.

Alternatively, you can do the following:

1. On the Variables tab of the Variables and Parameters pane, right-click the test and choose *Edit Variable*.

Tip

Alternatively, in an expanded test tree on the Data View pane, right-click a design variable and choose Edit Variable.

The Editing Design Variables form appears.

2. Select one or more variables you want to copy to the current cellview.
3. In the *Cellview variables* section, click *Copy To*.

   The program copies the selected design variable values to the current cellview.
4. When you are done with the form, click *OK*.

Copying Design Variable Values from the Schematic

If you change design variables on your schematic and want to use these values in your next simulation, do the following:

➤ On the Variables tab of the Variables and Parameters pane, right-click the test and choose *Copy from Cellview*.

Tip

Alternatively, in an expanded test tree on the Data View pane, right-click a design variable and choose *Copy from Cellview*.

The program copies design variable values from the schematic to the Variables tab of the Variables and Parameters pane.

© 2023 Cadence Design Systems, Inc.

Alternatively, you can do the following:

1. On the Variables tab of the Variables and Parameters pane, right-click the test and choose *Edit Variable*.

Tip

Alternatively, in an expanded test tree on the Data View pane, right-click a design variable and choose *Edit Variable*.

The Editing Design Variables form appears.

2. Click *Copy From*.

The program copies design variable values from the schematic to the Variables tab of the Variables and Parameters pane.

3. When you are done with the form, click *OK* or *Cancel*.

Note: You can disable copying design variables to global variables by setting the `copyDesignVarsFromCellview` environment variable to `nil`.

Importing Design Variables from an ADE State

For details on importing the design variables and parameters by using the **Loading State** form, see "Loading State Information" on page 112.

Displaying Design Variables on the Schematic

To display the values of instance parameters that are design variables on the schematic, do the following:

1. In the CIW, choose Tools – CDF – Edit.

The Edit CDF form appears.

2. In the *Scope* group box, select the *Library* radio button.
3. In the *CDF Layer* group box, select one of the following CDF types (see *CDF Type* in the CDF Commands chapter in the *Component Description Format User Guide* for information about these choices):
   - *Effective*
   - *User*

© 2023 Cadence Design Systems, Inc.

- *Base*

4. In the *Library Name* field, select the library whose CDF you want to edit.
5. On the Edit CDF form, click the *Interpreted Labels* tab.

   You can use interpreted labels to display parameter values, evaluated parameter values, net connectivity information, backannotated simulation information, and more. For more information, see “Specifying Label Information in the *Component Description Format User Guide*.
6. On the Interpreted Labels tab, click the *Parameters(cdsParam)* tab.
7. For *paramEvaluate*, select the *full* check box.

   Design variables appear on your schematic.

Defining Variables in a File

You can define functions and global variables that are not design variables (such as model parameters and simulator parameters) in a definitions file. Cadence provides sample definition files for the Spectre circuit simulator in `your_install_dir/tools/dfII/samples/artist/models/spectre` (see `defaults.scs` and `definitions.scs`).

To set up a definitions file, do the following:

1. Create the file in the directory you specify in the *Include Paths* field on the Simulation Files Setup form.
2. In the *Definition Files* field on the Simulation Files Setup form, type the full UNIX path and name of the definitions file.

The simulator reads the definitions file when it starts.

Your definitions file can contain any of the following:

Simple passing parameters

```
real R(real l, real w) {
return (500*l/w);
}
```

Functions returning constant values

```
real PiRho() {
return 2500;
}
real Rpi(real l, real w) {
return PiRho()*l/w;
}
```


© 2023 Cadence Design Systems, Inc.

For more information about defining functions in a definitions file, see “User Defined Functions (functions)” in the *Spectre Circuit Simulator Reference*.

See also

- Example of a Polynomial Resistor as a Function of Temperature on page 181
- About Inherited Parameter Value Functions (iPar, pPar) on page 181

Example of a Polynomial Resistor as a Function of Temperature

You can define a polynomial resistor as a function of temperature as follows:

```
real rpoly(real value, real tdc) {
value*(1+.01*(tdc-25)+.002*(tdc-25)**2);
}
```

You can use this function to specify the value of a resistor as follows:

```
rpoly(1k,tempdc)
```

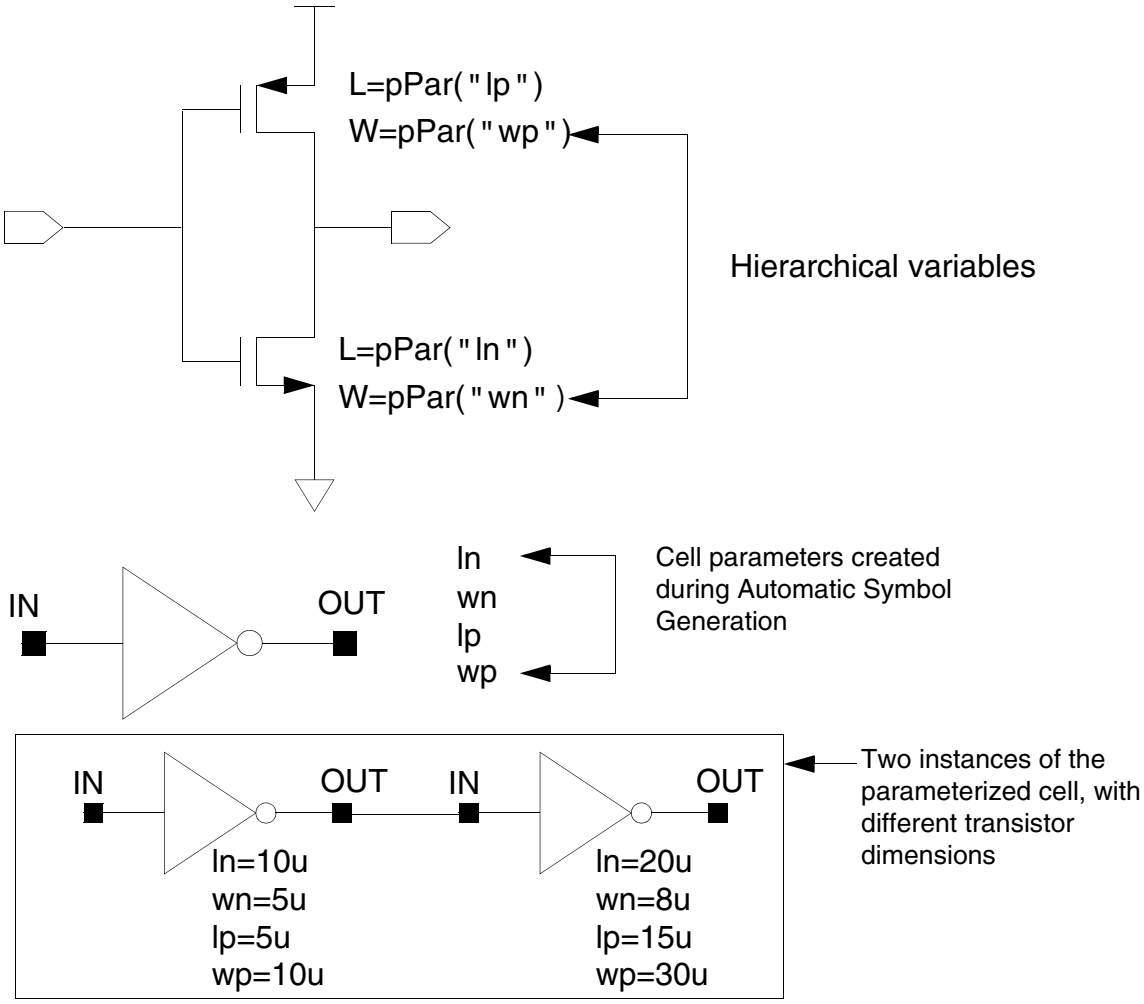
You can set resistor properties `tc1` and `tc2` so that the system automatically models resistor temperature effects, rather than defining your own functions.

About Inherited Parameter Value Functions (iPar, pPar)

You can use inherited parameter value functions iPar and pPar in conjunction with built-in functions or user-defined functions.

© 2023 Cadence Design Systems, Inc.

The following diagram shows a schematic example using pPar:



For more information about iPar and pPar parameters, see Passing Parameters in a Design in the Component Description Format User Guide.

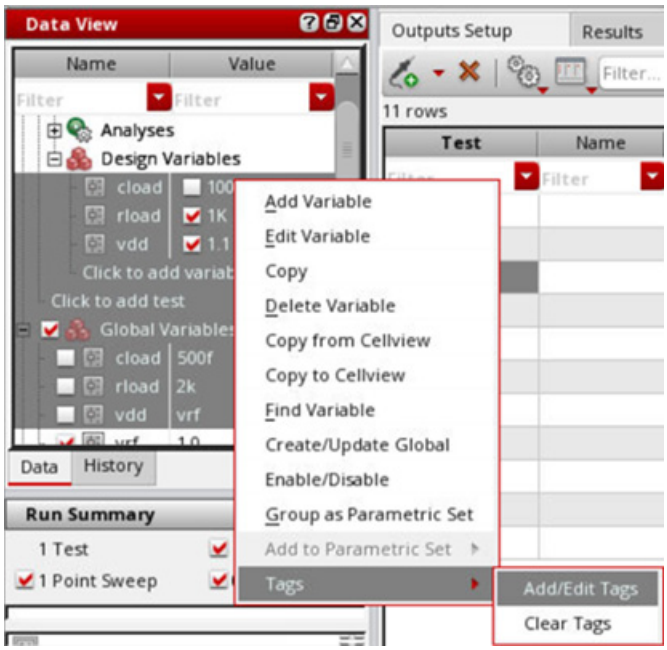
Adding Variable Tags

To filter or search through a long list of design and global variables, you can assign specific tags to them.

To do this, perform the following steps:

© 2023 Cadence Design Systems, Inc.

1. Right-click a variable or a group of variables in the Data View assistant and choose *Tags — Add/Edit Tags*. You can select both design and global variables at once.



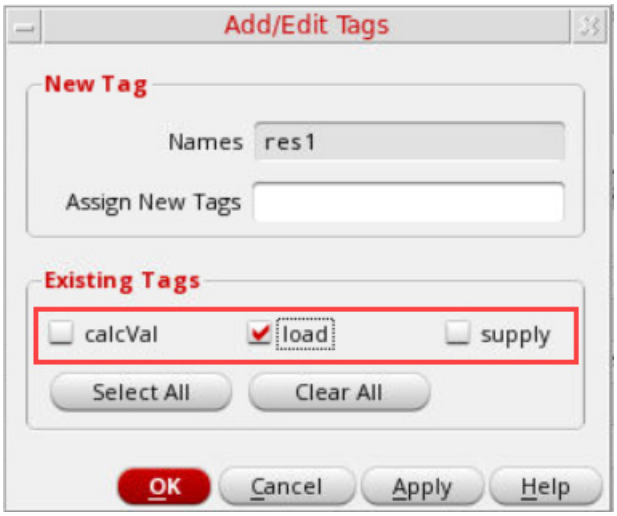
The Add/Edit Tags form is displayed.

2. In the *Assign New Tags* field, specify the tag you want to assign to the variable.

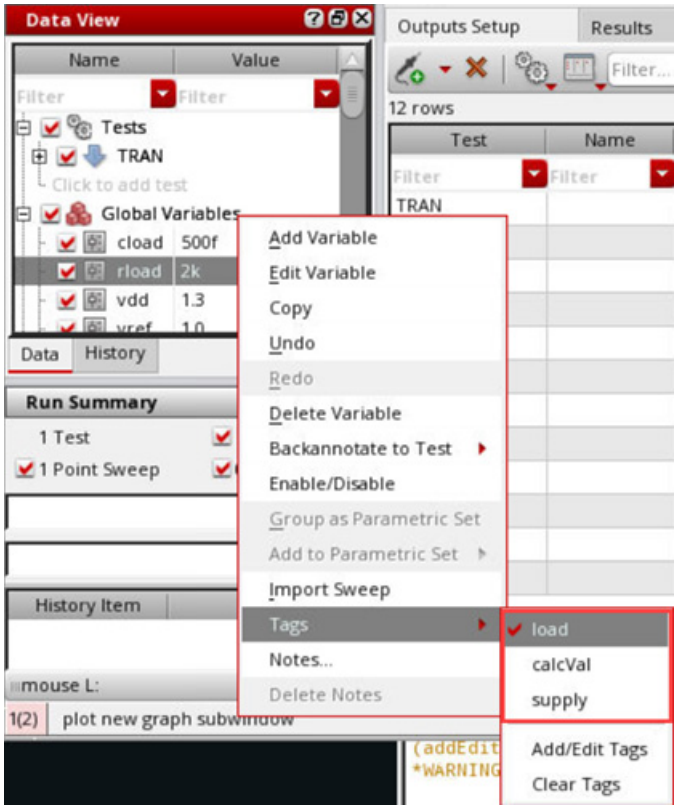
   This tag is added to the *Existing Tags* section of the form as well as in the *Tags* submenu of the context menu for global variables.

© 2023 Cadence Design Systems, Inc.

After adding tags for multiple variables you can choose to assign the existing tags to another variable by selecting the check boxes against them.



Alternatively, you can right-click a variable or a group of selected variables and select the tag you want to assign from the *Tags* submenu.



© 2023 Cadence Design Systems, Inc.

You can then hover over a variable to see the associated tag in the tooltip. This information can also be viewed in ADE Explorer.

A tag added to a design variable is also visible for the global variable with the same name, and vice-versa.

Note: On exporting the variables to a CSV file, the tags related to the variables are also exported. You can then import that CSV file to a different session and retain the tag assignments.

To clear the tags for specific variables, right-click the variable and choose *Clear Tags* from the *Tags* submenu.

Note: The *Clear Tags* command clears the tags associated with a variable but does not delete them.

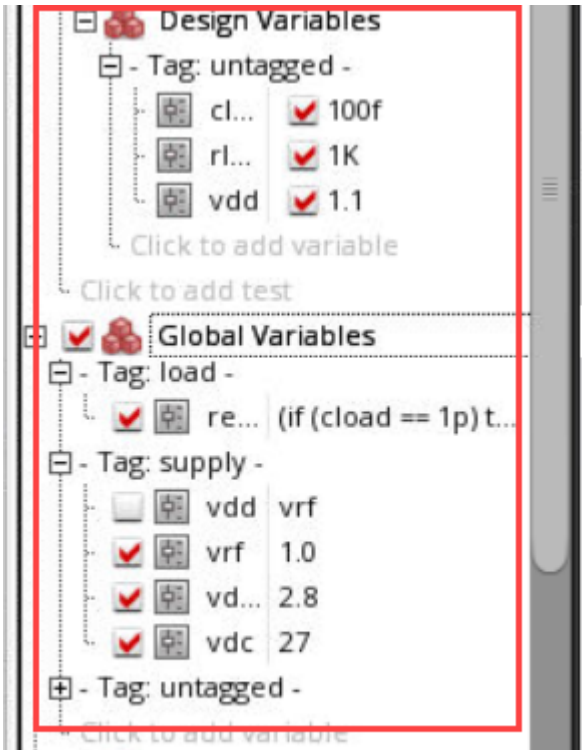
Grouping Variables using Tags

You can group global and design variables according to the tags assigned to them. To do this:

➡ Right-click the *Design Variables* or *Global Variables* section in the Data View assistant and select *Group By Tags*.

© 2023 Cadence Design Systems, Inc.

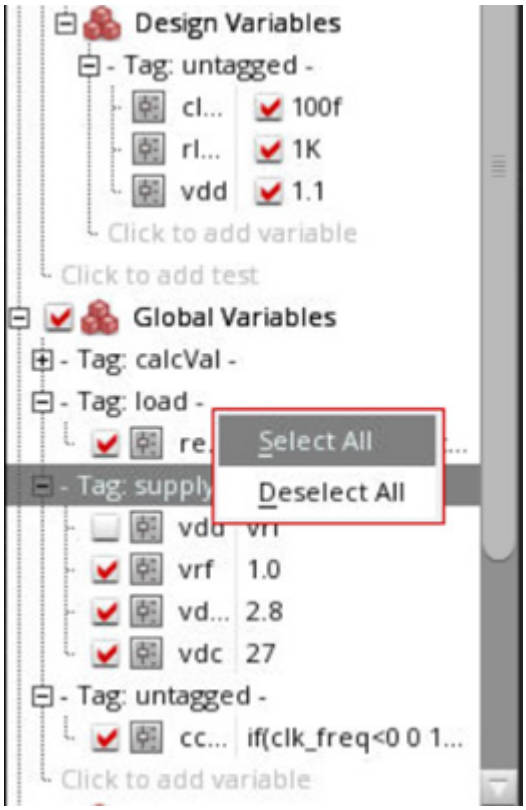
This groups the variables in both, the *Design Variables* and *Global Variables* sections according to the assigned tags and creates sub-trees named `Tag:tagname`.



Variables that do not have any assigned tags are displayed in a sub-tree named `Tag:untagged`.

© 2023 Cadence Design Systems, Inc.

You can also right-click a tag group name and select or deselect all variables.



To ungroup the tags, re-select the *Group By Tags* command from the *Design Variables* or *Global Variables* context menu.

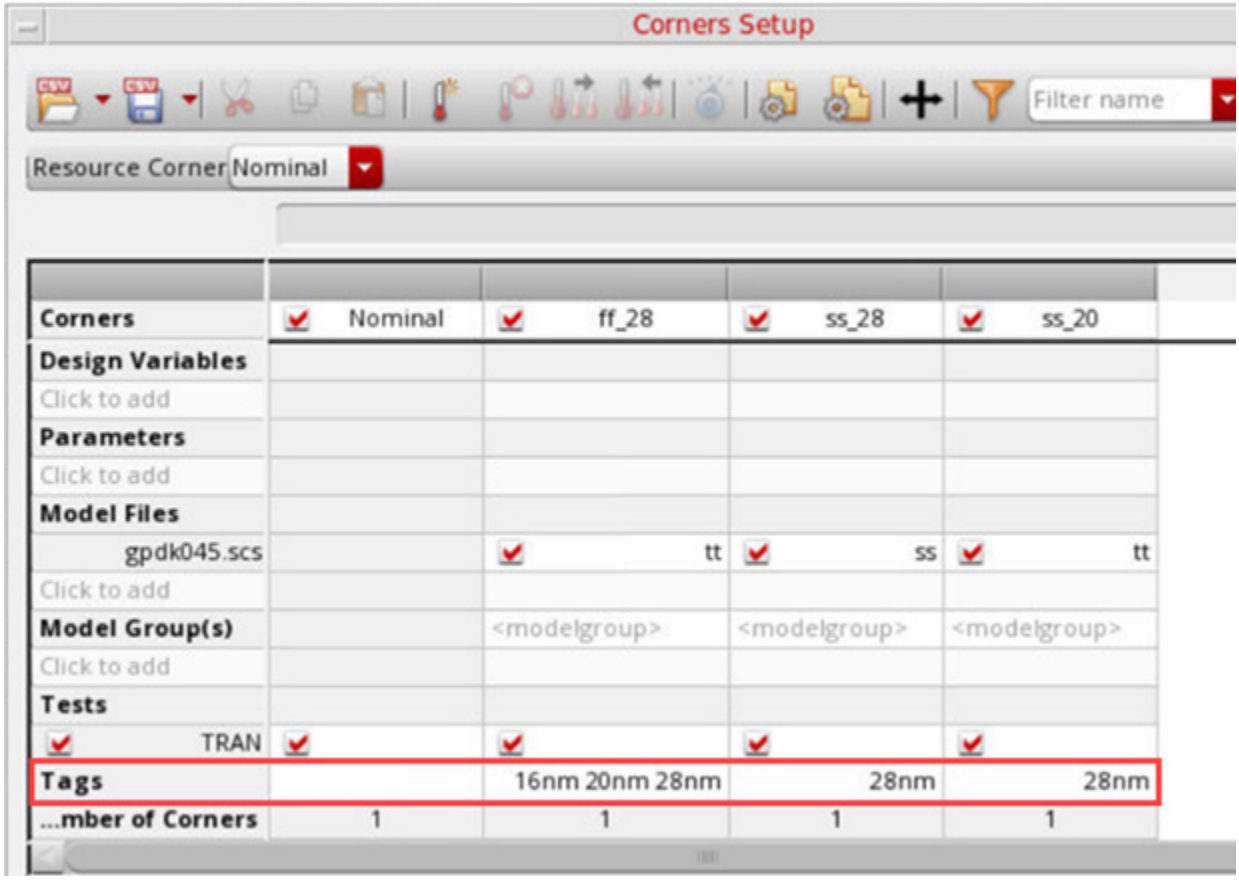
Assigning Tags to Corners

1. Open the Corners Setup form.
2. Specify a tag or a space-separated list of tags for each corner in the *Tags* row.

   The tag names must be unique and alphanumeric with no blank spaces. You can use _,-, +, # and @ special characters in the tags for corners. The *, =, <, >, ?, |, ', \\, \", ~, (,), /, ;, &, and $ special characters are not supported.

© 2023 Cadence Design Systems, Inc.

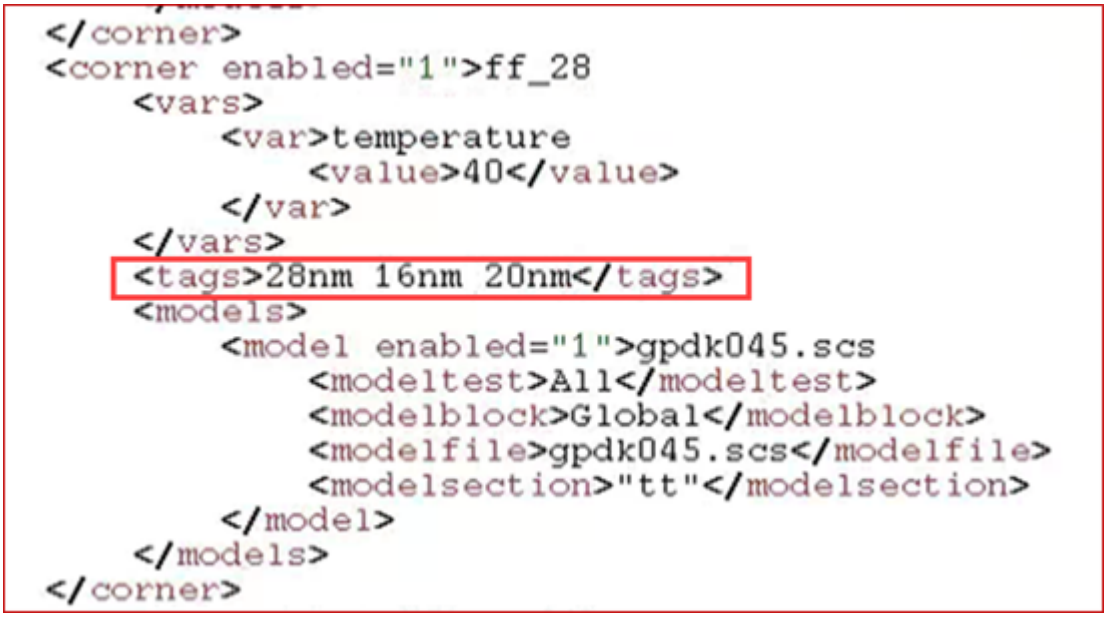
For example, let us consider the following setup.



Here, the tags `16nm`, `20nm`, and, `28nm` have been assigned to the corner `ff_28` while the tag `28nm` has been assigned to the corners `ss_28` and `ss_20`.

© 2023 Cadence Design Systems, Inc.

Alternatively, you can edit the corner definitions through a CSV or SDB file and add a `tags` row as shown in the figure.



On importing this CSV or SDB file through the Corners Setup form, the specified tags get added to the *Tags* row.

You can then hover over a corner to see the associated tag in the tooltip. This information can also be viewed in ADE Explorer.

Note: You can assign the existing corner tags to variables using the *Tags* submenu of the context menu for global or design variables.

To remove a tag, select it and press the `Delete` key.

Grouping Corners using Tags

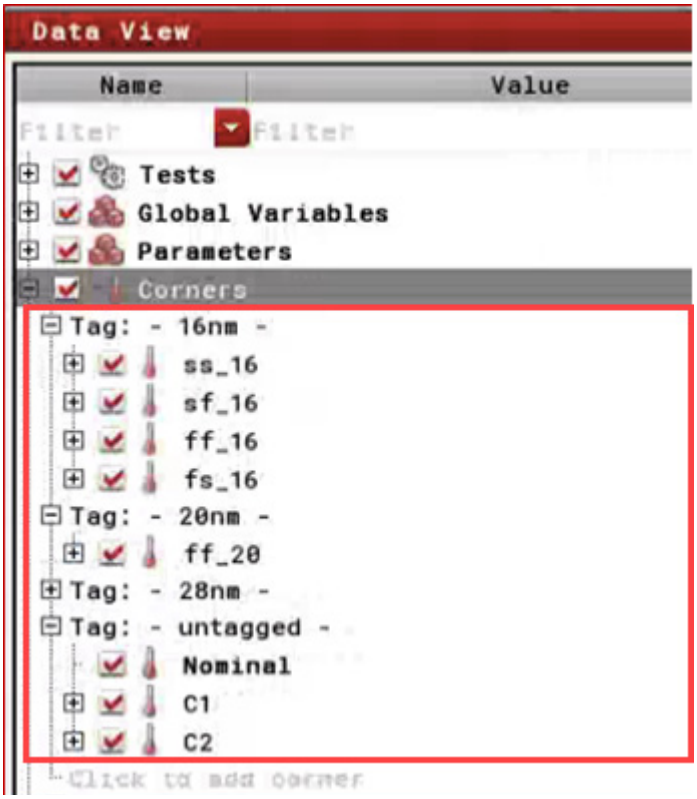
You can group corners according to the tags assigned to them. To do this:

➡ Right-click the *Corners* section in the Data View assistant and select *Group By Tags*.

This groups the corners according to the assigned tags and creates subtrees named `Tag:`*`tagname`*.

© 2023 Cadence Design Systems, Inc.

The corners with no tags specified to them are displayed under a subtree named `Tag:untagged`.



You can also set the variable `enableGroupByTagsForCorners` to use this option.

To ungroup the tags, re-select the *Group By Tags* command from the *Corners* context menu.

Related Topics

enableGroupByTagsForCorners

Filtering Variables and Corners

You can use tags, variable names, and corner names to sort or filter a long list of variables. There are two ways to do this.

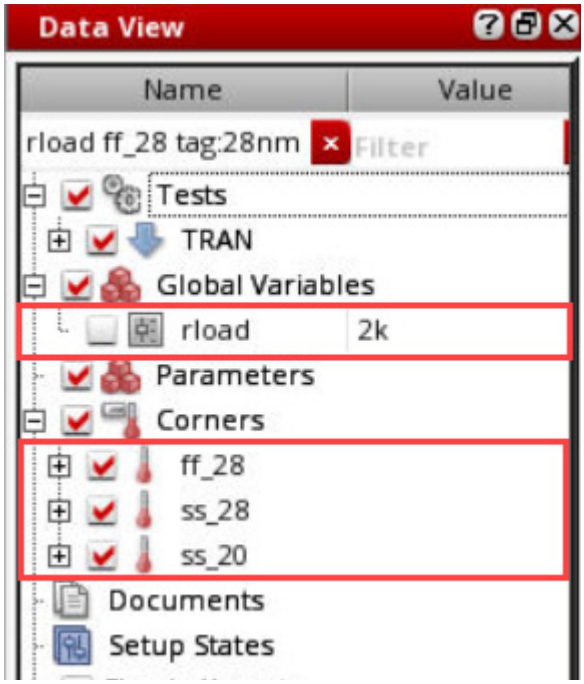
1. In the filter of the *Name* column of the Data View assistant, use the format *`variablename cornername tag:tagname`*.
2. Right-click the filter and set the *Find with* criterion to filter out the variables and corners with all, any, exactly, or none of the specified names and tags.

© 2023 Cadence Design Systems, Inc.

By default, the *All Of The Words* option is selected. However, it is recommended to select the *Any Of The Words* option.

For example, setting the filter to *Any Of The Words* and using the command `rload ff_28 tag:28nm` in the filter of the *Name* column will filter out:

- The variable with the name `rload`
- The corner with the name `ff_28`
- The variables and corners that have the tag `28nm` assigned to them



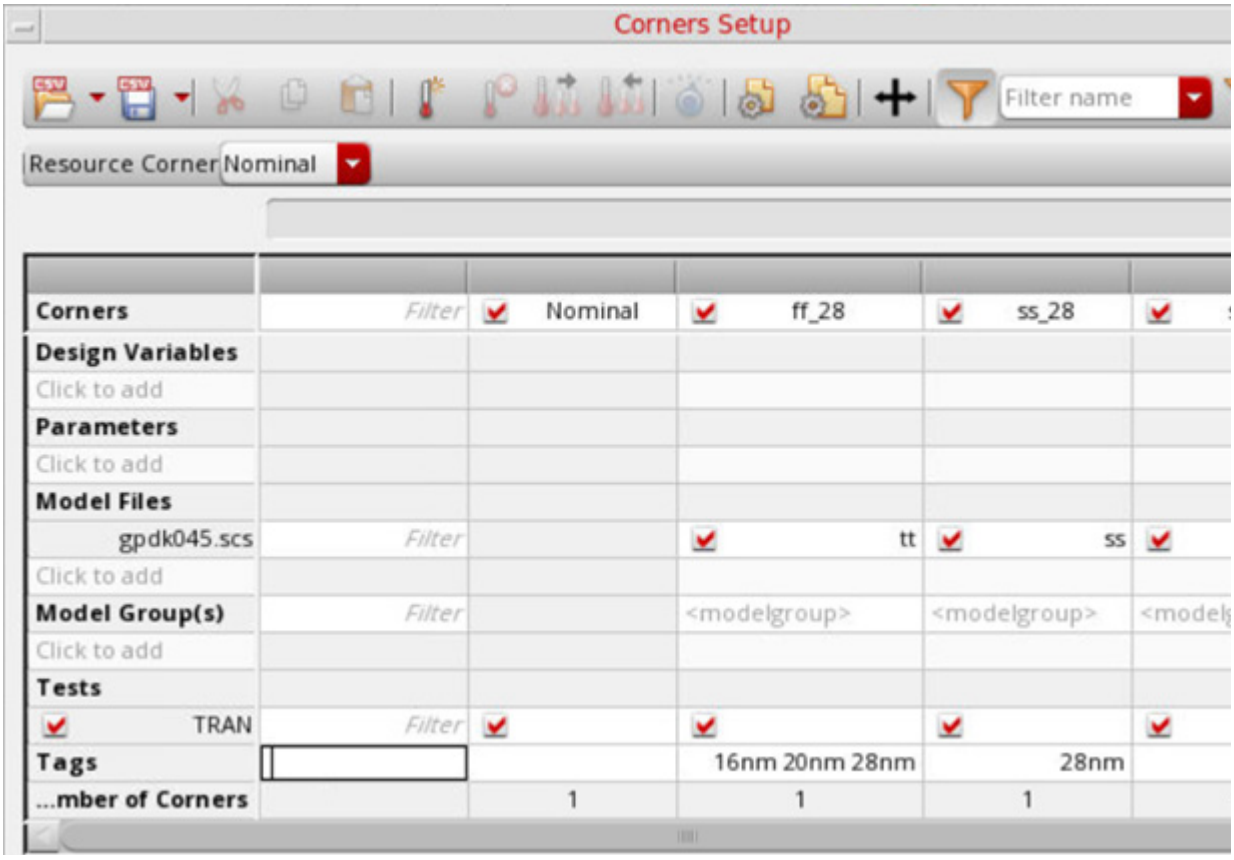
© 2023 Cadence Design Systems, Inc.

To filter variables using multiple tags, variable names, and, corner names, specify a space-separated list of variable and corner names followed by the format `tag:`*`tag1`* `tag:`*`tag2`*.

Alternatively, you can click the *Toggle visibility of the Filter column* In the Corners Setup form.

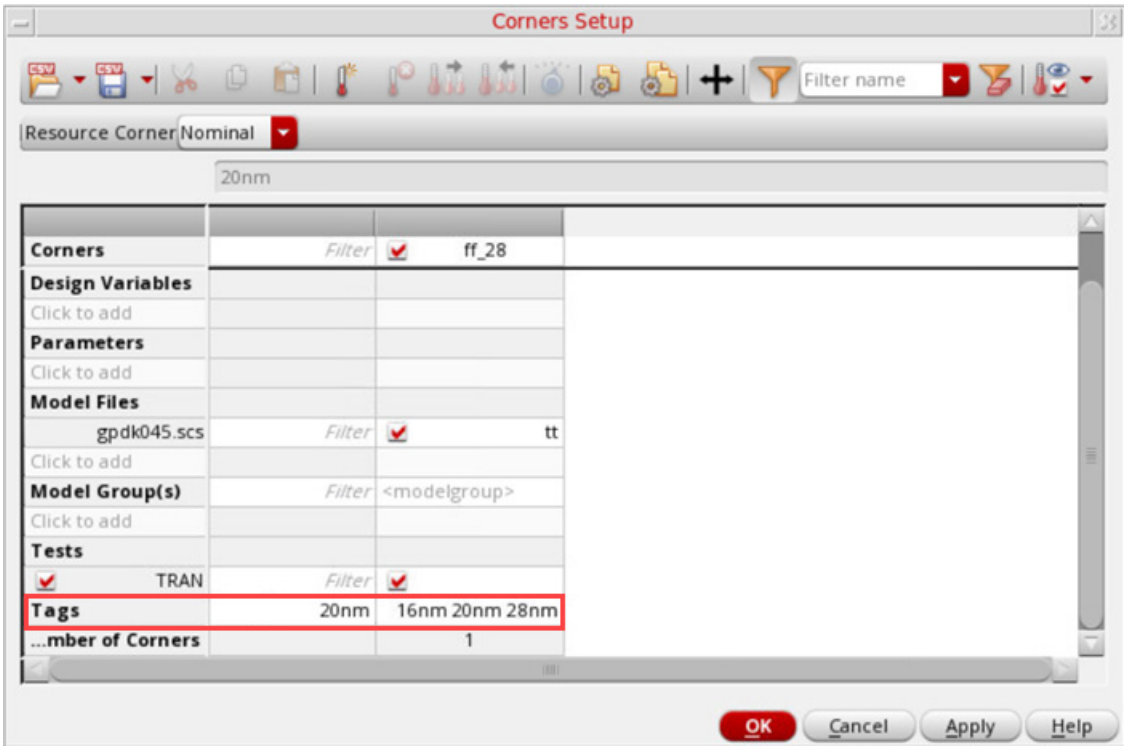
In the resulting *Filter* column, specify the tags you want to filter the corners by.

For example, let us consider the following setup.



© 2023 Cadence Design Systems, Inc.

Here, on specifying the tag `20nm` in the *Tags* row of the *Filter* column, the corner `ff_28` will get filter out.



© 2023 Cadence Design Systems, Inc.

Working with Device Instance Parameters

In ADE Assembler, you can

- Change the value of a device instance parameter for a simulation without affecting the value on the schematic
- Disable the changed value of a device instance parameter for a simulation
- Sweep a device instance parameter
- Disable callbacks on swept device parameters
- Create matched device parameters
- Create ratio-matched device parameters
- Backannotate changed device instance parameter values back to the schematic

See the following topics for more information:

- Selecting Instance Properties to be Parameterized for Simulations
- Changing the Value of a Device Instance Parameter for Simulation
- Disabling a Changed Device Instance Parameter Value for a Simulation
- Sorting Parameters by Properties and Objects
- Filtering Device Instance Parameters
- Creating Custom Device Filters
- Changing the Order of Device Parameters
- Deleting a Parameter
- Creating Parameter Ranges
- Working with Parametric Sets
- Working with Parametric Sets
- Enabling and Disabling Parameters
- Adding Notes for Parameters
- Adding or Changing a Variable or Parameter Specification
- Creating Matched Device Parameters

© 2023 Cadence Design Systems, Inc.

- Creating Ratio-Matched Device Parameters
- Disabling Callbacks on Swept Device Parameters

© 2023 Cadence Design Systems, Inc.

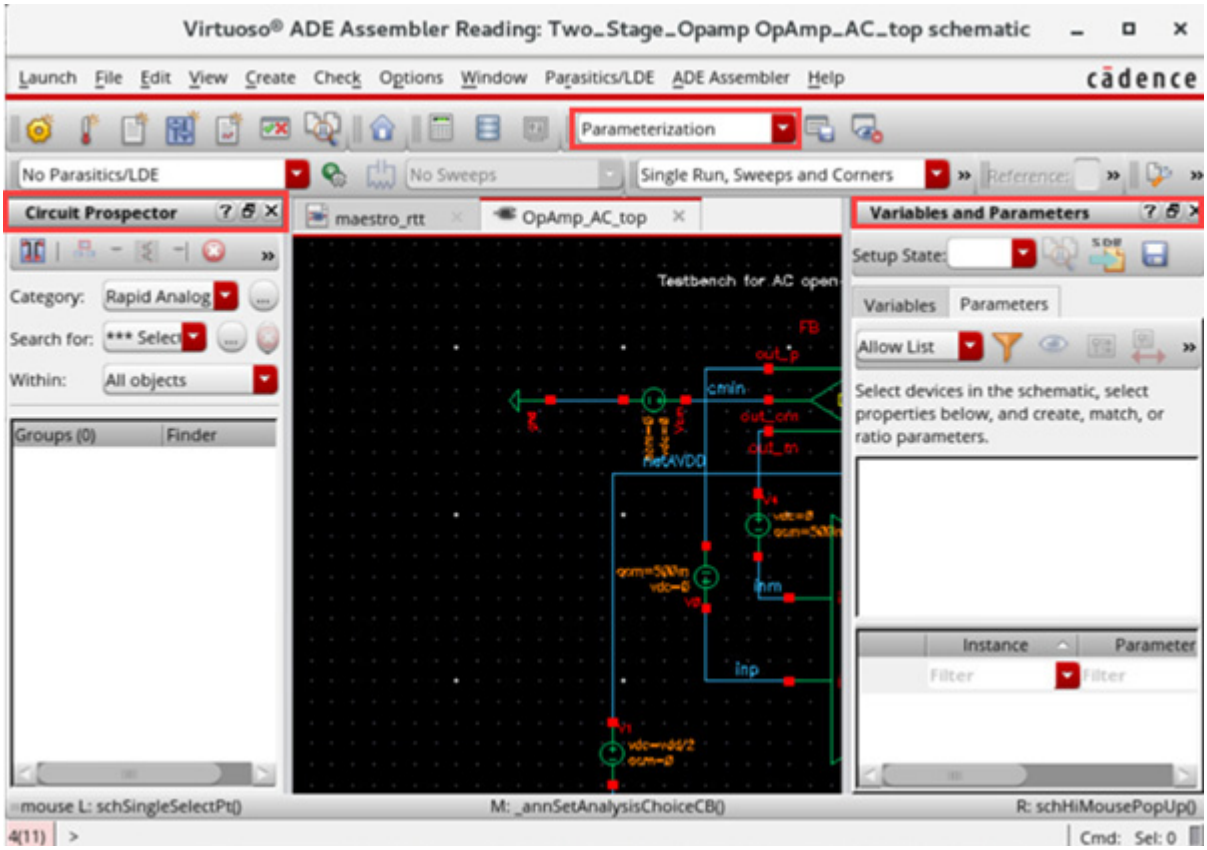
Selecting Instance Properties to be Parameterized for Simulations

To select instance parameters for the design associated with your test, do the following:

1. Click *Click to add parameter* in the Parameters tree on the Data View assistant.

   The schematic associated with the test is displayed in a new tab in the ADE Assembler window and the menu changes to that of Virtuoso Schematic Editor and the `Parameterization` workspace is set.

   In this workspace, the Circuit Prospector, and the Variables and Parameters assistant panes are displayed. The *Parameters* tab on the Variables and Parameters assistant pane is also opened.

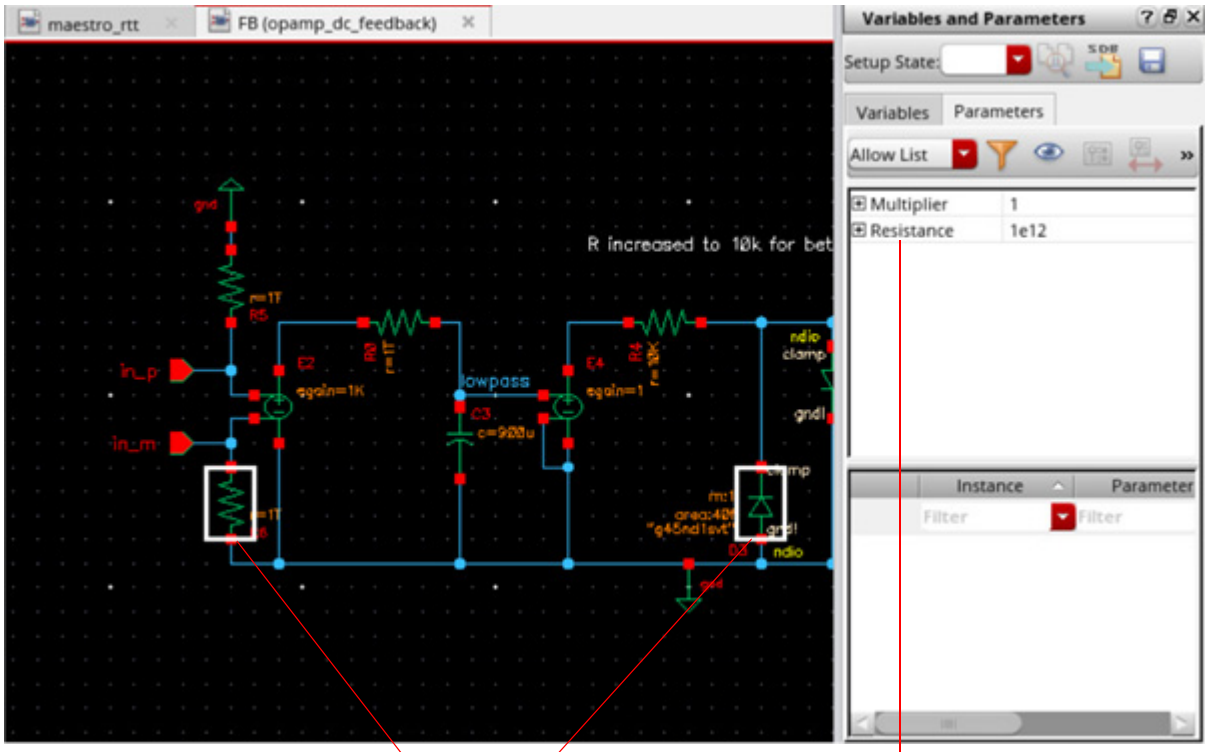


2. On the schematic, select one or more device instances.

   To select more than one instance, hold down the `Shift` key and click each instance one by one or by dragging a selection box around a set of instances. Alternatively, you can use the Navigator assistant pane to select the instances.

© 2023 Cadence Design Systems, Inc.

The parameter properties of the selected devices are listed in the upper half of the *Parameters* tab of the Variables and Parameters assistant pane.

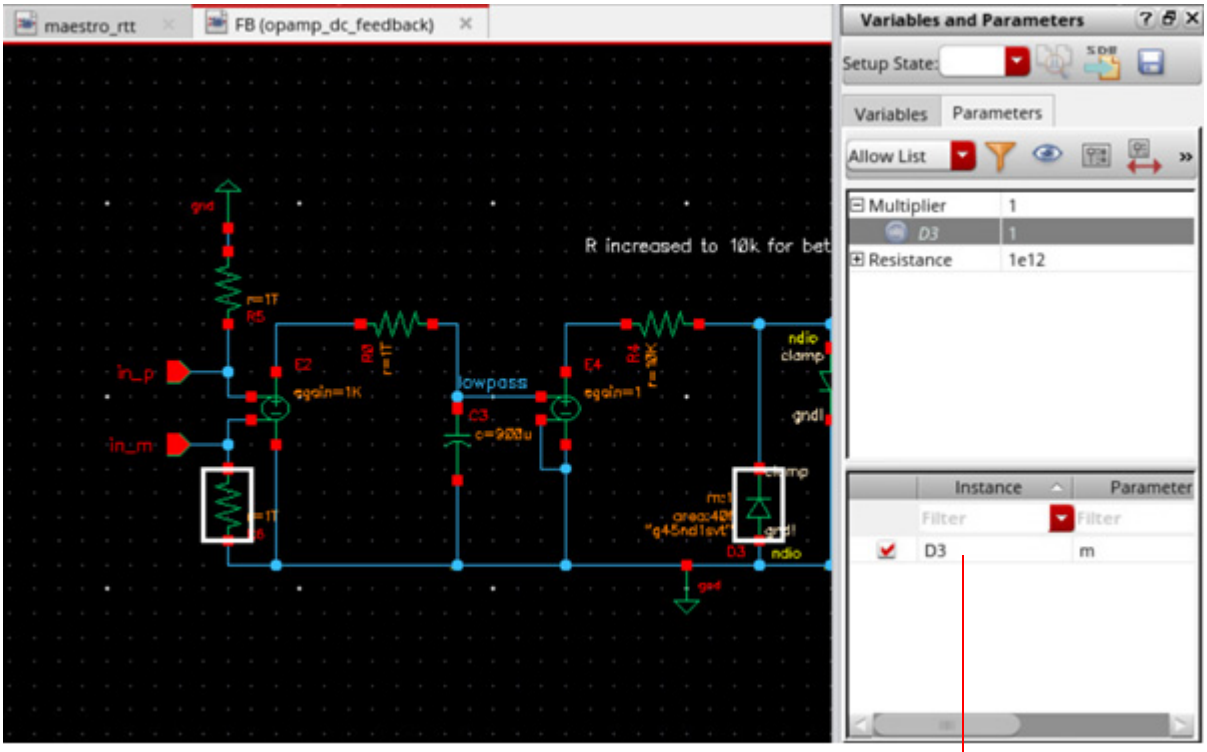


3. Expand a property tree to view the device names.
4. Right-click an instance name and choose *Create Parameter*.

   Alternatively, click *Create Parameter* on the toolbar.

© 2023 Cadence Design Systems, Inc.

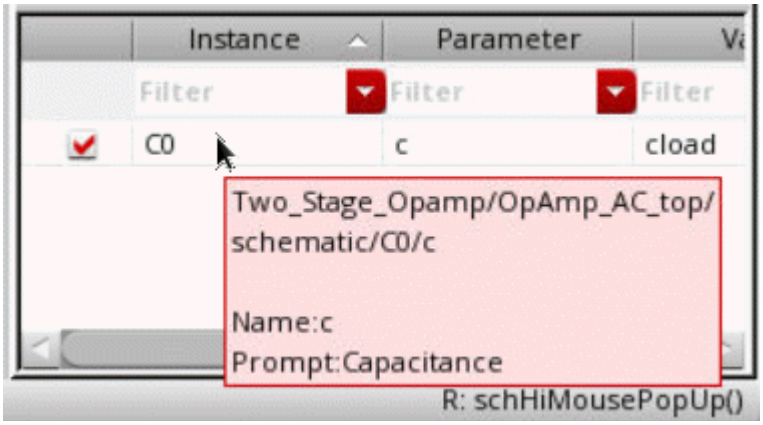
A parameter is created for the selected property of the selected device instance. The details of the added parameter appear in the lower half of the Variables and Parameters assistant pane.



The value to be used (shown in the *Value* column) is same as that of the value of the parameter in the schematic (also shown in the *Design Value* column). You can specify a different value in the *Value* column to be used while running simulation.

© 2023 Cadence Design Systems, Inc.

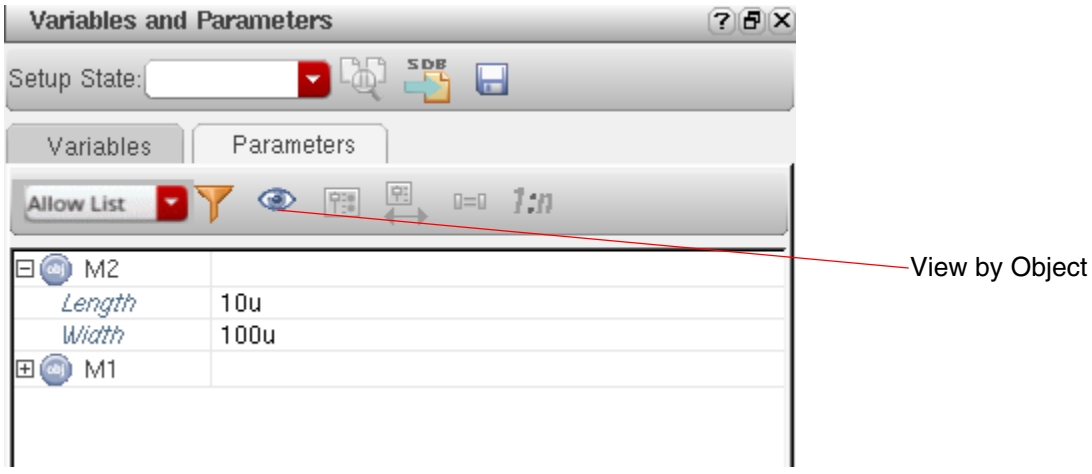
You can hover above an instance name to view the related test information along with the parameter name and prompt name for that instance.



For more details, refer to Changing the Value of a Device Instance Parameter for Simulation.

Additional Points to Note

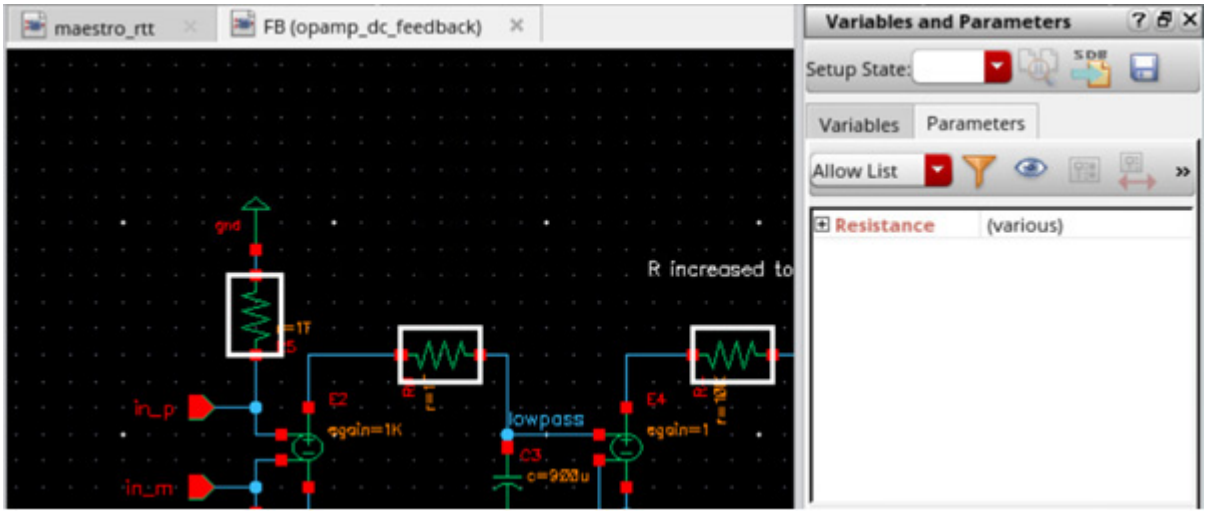
- It is possible to create parameters for the properties of parametrized cells (Pcells) in the design.
- The default view for the Parameters tab is `View by Property`. That is why when you select an instance in the schematic, by default, a list of property names is displayed. However, you can change the view to list the instance names by clicking the toggle on the toolbar of the Parameters tab, as shown below.



You can also set the default view by setting the defaultParametersViewBy environment variable.

© 2023 Cadence Design Systems, Inc.

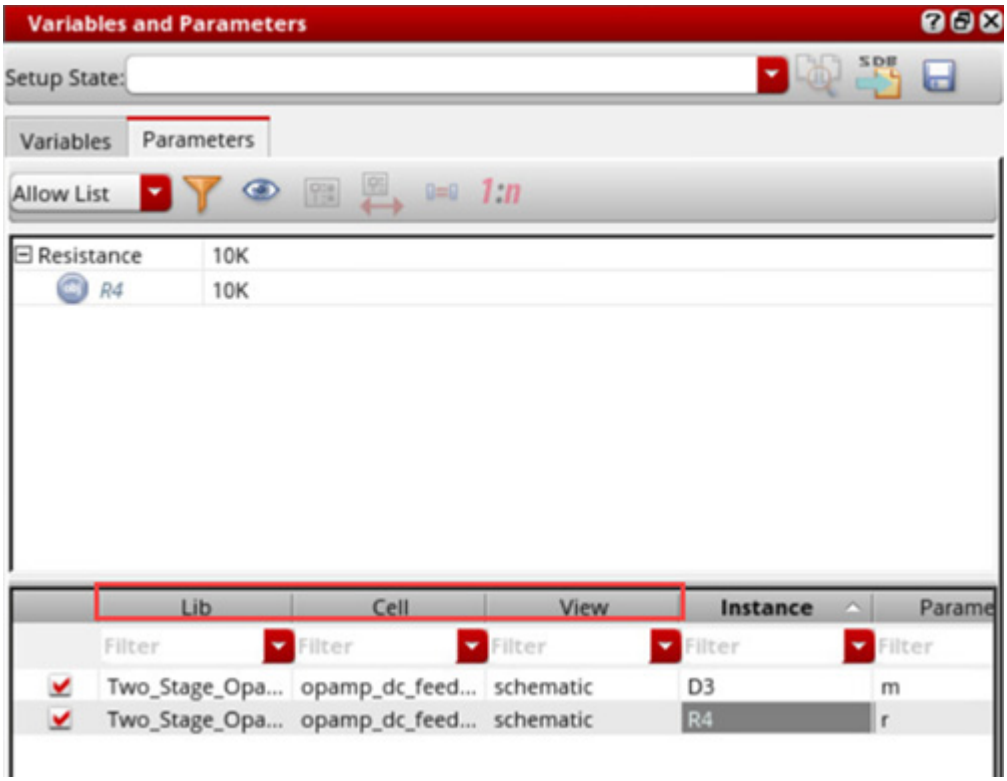
- If you have selected more than one instance in the schematic and each instance has a different design value for a property, the value is shown as `(various)`.



- When the view is set to `View by Property`, you can select multiple devices in the schematic and use the *Create Parameter* command to create parameters for all the selected devices simultaneously. For example, if you have selected two instances, `M1` and `M2`, on the schematic. In the Variables and Parameters assistant, right-click `Length`, and choose *Create parameter*. The `Length` parameter is created for both `M1` and `M3` simultaneously.
- You can also create matched or ratio-matched parameters by using the *Match Parameters* or *Ratio Matched Parameters* commands on the toolbar. For more details, refer to Creating Matched Device Parameters and Creating Ratio-Matched Device Parameters.
- If the devices have a large number of parameter properties, you can use filters to show only the relevant set of properties. For more details, see Filtering Device Instance Parameters.
- When you create parameters, the lower half of the Variables and Parameters assistant pane displays the instance name, parameter name, parameter value, and the value of property in the design. You can also show the names of library, cell, and view of the

© 2023 Cadence Design Systems, Inc.

instances. For this, right-click any column name in the lower half of the assistant and choose *Show Lib, Cell, View*. Three more columns are displayed, as shown below.



You can also show either one of these columns, as required, using the commands in the context-sensitive menu for this section of the assistant pane. If required, you can edit the names of library, cell, and view in this column. ADE Assembler validates the existence of the newly specified library, cell, and view. If found, the details are updated in the assistant pane. Otherwise, an error is displayed in the CIW.

- When you open a saved maestro view, if the cellview details of the instance parameters are different, the Variables and Parameters assistant shows all the columns to display the cellview details.

© 2023 Cadence Design Systems, Inc.

Changing the Value of a Device Instance Parameter for Simulation

To change the value of a device instance parameter for simulation, do the following:

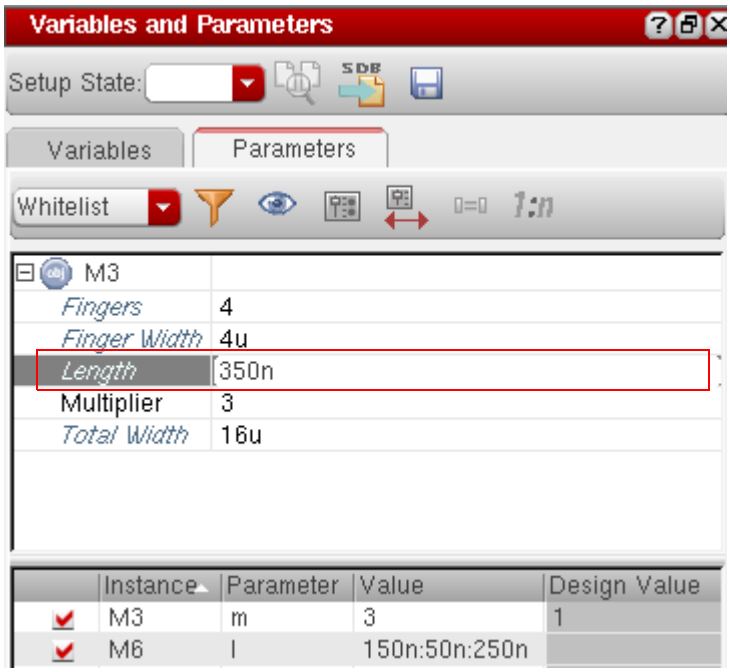
1. On the upper half of the Parameters tab on the Variables and Parameters pane, click the value of the device instance parameter you want to change and type a new value.

   Note: You might have to expand the tree by clicking the plus sign to the left of the instance name at the root of the tree.

Tip

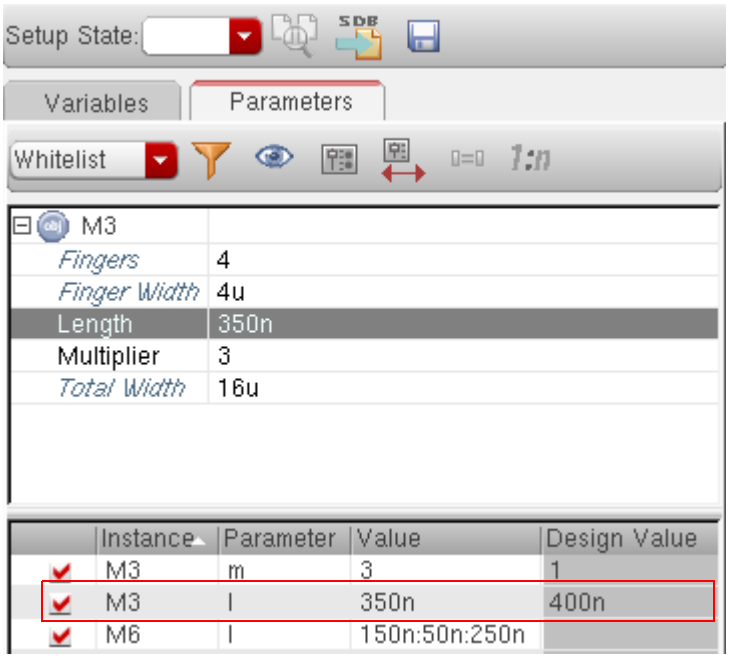
To view only the parameters in the Parameters tab, see Sorting Parameters by Properties and Objects.

For example, change the value of the *Length* parameter from `400n` to `350n`.



The instance parameter and its new value (`350n`) appear in the lower half of the Parameters tab on the Variables and Parameters assistant pane. The simulator uses this value when run simulations.

© 2023 Cadence Design Systems, Inc.



As shown for the `M6` instance in the above example, you can sweep the values of parameters.

To specify a valid sweep value, ensure that:

- The value is not a blank string or does not contain a whitespace
- If contains a range, it has the start, step, and stop values separated by colons, commas, spaces, or a combination of these
- If specified as an expression, it is written in correct SKILL syntax and contains balanced parenthesis and quotes
- If contains a file path, it is enclosed in double quotes so that it is not evaluated

Alternatively, you can do the following:

1. Right-click the instance parameter whose value you want to change and choose *Create Parameter*.

   The instance parameter and its value appear in the lower half of the Parameters tab on the Variables and Parameters assistant pane.

2. Double-click the value in the lower half of the Parameters tab and type a new one.

The parameter is also displayed in the *Parameters* tree in the Data View pane.

© 2023 Cadence Design Systems, Inc.

Important Points to Note

- You can add parameters only by using the *Parameters* tab of the Variables and Parameters assistant pane. Parameters cannot be directly added in the Data View pane.
- At any time, you can reset the original value, which was set in the design schematic, for a parameter. For this, in the lower half of the Parameters tab, right-click a parameter name and choose *Set to Design Value*.

Sweeping Pcell Parameters

You can sweep Pcell parameters in the same way as described above for other device parameters. The sweep values are used to create multiple design points for simulations. During netlist creation, ADE Explorer and ADE Assembler creates subcircuits for each swept Pcell parameter value.

This feature is helpful for advanced node designs that contain Pcells with parameters to adjust circuit topology changes, such as the number of stacked components represented by a device. You can run local or global optimization to identify the best value for the Pcell parameters to optimize your designs.

Disabling a Changed Device Instance Parameter Value for a Simulation

To disable a changed device instance parameter value for a simulation, do the following:

➤ In the lower half of the Parameters tab on the Variables and Parameters assistant pane, deselect the check box for that parameter.

Tip

Alternatively, in an expanded *Parameters* tree on the Data View pane, deselect the check box for that parameter.

The next time you click the Run Simulation button, the simulator uses the device instance parameter value from the schematic.

Deleting a Parameter

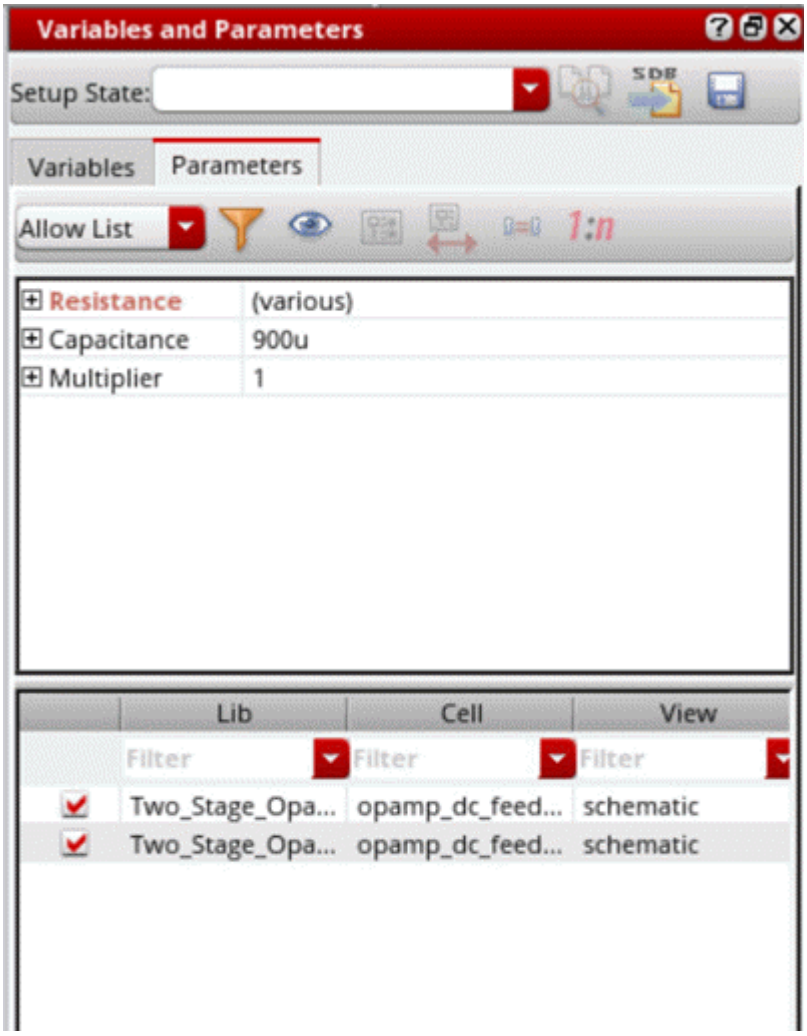
To delete a parameter, do the following:

➤ In an expanded *Parameters* tree on the Data View pane or on the lower part of the Parameters tab of the Variables and Parameters pane, right-click the parameter you want to delete and choose *Delete Parameter*.

© 2023 Cadence Design Systems, Inc.

Sorting Parameters by Properties and Objects

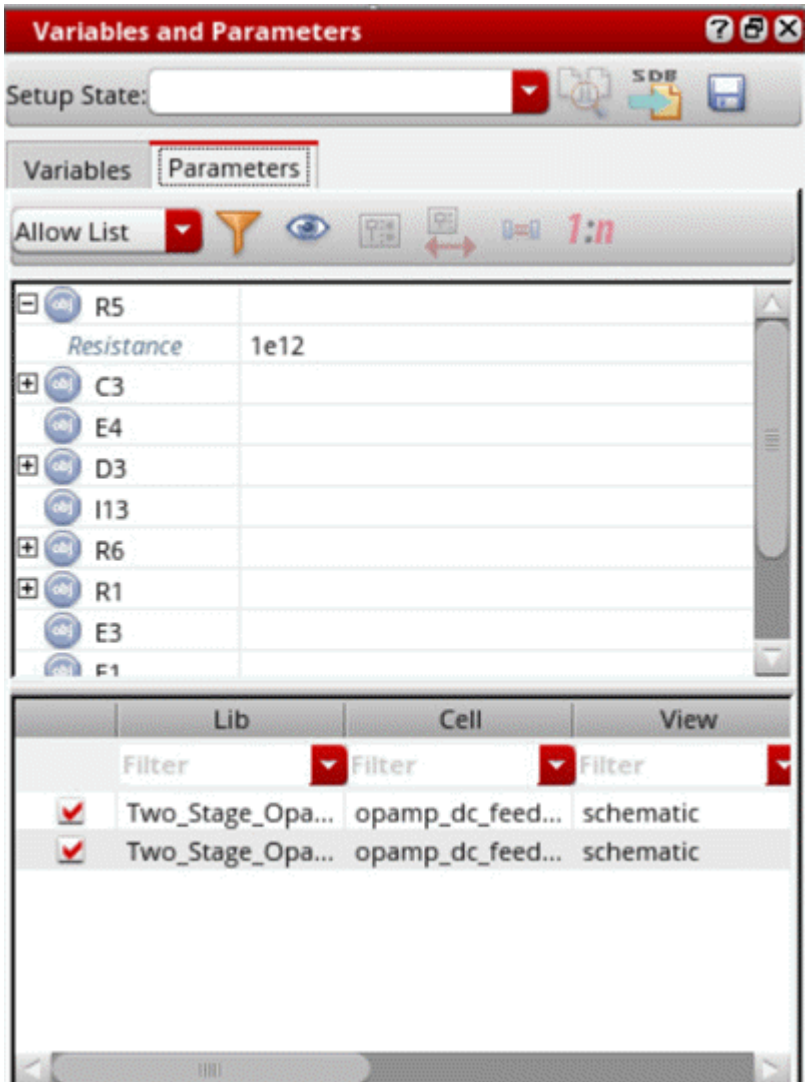
By default, in the upper part of the Parameters tab of the Variables and Parameters assistant pane, the parameters are sorted by property names, as shown below.



To sort the parameters by object names, do the following:

© 2023 Cadence Design Systems, Inc.

➡ Right-click the upper part of the Parameters tab and choose *View by Object*. Alternatively, click ( ) to toggle the view types. The view changes to a tree of objects.



Note: You can set the preferred view as the default view by using the defaultParametersViewBy environment variable.

Filtering Device Instance Parameters

To filter the device instance parameters that appear on the upper half of the Parameters tab on the Variables and Parameters assistant pane, do the following:

© 2023 Cadence Design Systems, Inc.

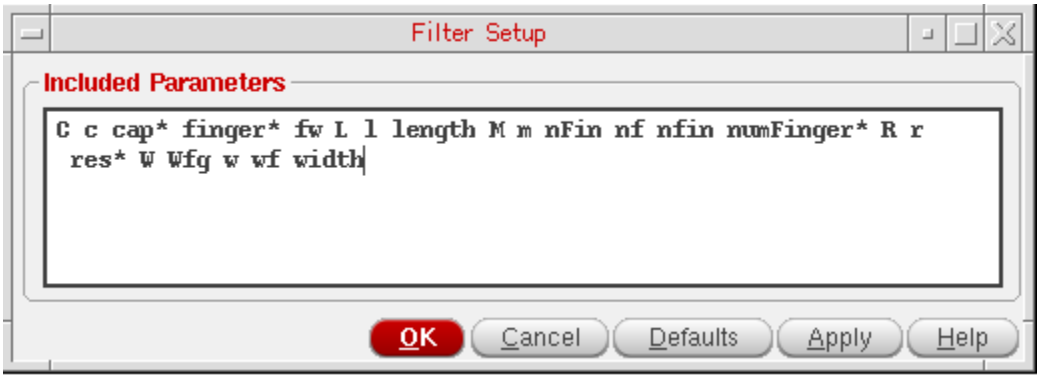
➤ In the *Filter* drop-down list, select one of the following:

`Allow List`	Displays parameters specified in the list that you can edit by using the Filter Setup form.
`Automatic`	Displays all editable CDF parameters with non-nil CDF values
`Editable`	Displays all editable parameters
`All`	Displays all the parameters

Only those device instance parameters that meet the filter specification appear in the upper half of the Parameters tab on the Variables and Parameters assistant pane.

By default, the allow list filter shows the parameters set by the defaultParametersAllowList environment variable. You can edit this list in either of the following ways:

- By editing the environment variable
- By using the Filter Setup form that is displayed when you click *Allow List Filter* ( ) on the toolbar.



You can use the `*` wildcard character to specify the parameter names in the Allow List filter. For example, `*inger* res*`.

The *Allow List Filter* command is enabled only when the *Filter* drop-down list is set to `Allow List`. For other filter types, this is`defaultParametersAllowList` not applicable.

Note: If you have defined any custom device filter using the `axlRegisterCustomDeviceFilter` function, the name of that filer also appears in the drop-down list. See Creating Custom Device Filters for more information. However, it is recommended to use the `Allow List` filter to customize the list of parameters to be displayed.

© 2023 Cadence Design Systems, Inc.

Creating Custom Device Filters

To create a custom device instance parameter filter, do the following:

1. Write a procedure to define your custom filter.

   For example, you want to filter out the `Total Width`, `Threshold` and `S/D Metal Width` parameters for all instances of the `pmos2v` cell, write the following procedure:

```
(procedure (myCustomFilter inst simulator)
    (let (paramList libName cellName instName)
    (setq instName inst->name)
    (setq cellName inst->cellName)
    (setq libName inst->libName)
    (setq paramList list())

    ;;Example filter for hiding Total Width, Threshold and S/D Metal Width
    ;;parameters for instances of cells whose names start with pmos2v
    (rexCompile "^pmos2v")
    (if (rexExecute cellName) then
         (setq paramList
              (list
                   (list "fw" (get inst "fw"))
                   (list "l" (get inst "l"))
                   (list "fingers" (get inst "fingers"))
                   (list "m" (get inst "m"))
              )))

    paramList
    )
)
```

   Note: You must specify the CDF parameter names and not the CDF prompt names in the procedure. For example, you must specify the CDF parameter name `fw` and not the CDF prompt name `Finger Width` in the procedure.

© 2023 Cadence Design Systems, Inc.

Tip

You can also write procedures to filter parameters of instances starting with specific names (for example, all instances starting with the name `PM`), or to filter parameters of instances of cells in a library (for example, filter parameters of all instances of the `pmos2v` cell in the `gpdk090` library).

2. Call `axlRegisterCustomDeviceFilter` as follows:

   ```
   axlRegisterCustomDeviceFilter "My Filter" 'myCustomFilter
   ```

   The function returns `t` if the registration is successful; otherwise, `nil`.

   My Filter appears in the *Filter* drop-down list.

To load your custom filters each time the program starts, you can declare and register them in your `.cdsinit` file.

When you select *My Filter* in the *Filter* drop-down list, the device parameters `Total Width`, `Threshold` and `S/D Metal Width` do not appear on the upper half of the Parameters tab on the Variables and Parameters assistant pane for the selected device instance(s).

Note: To set a custom filter as the default filter, use the defaultParametersAssistantFilter environment variable.

© 2023 Cadence Design Systems, Inc.

Changing the Order of Device Parameters

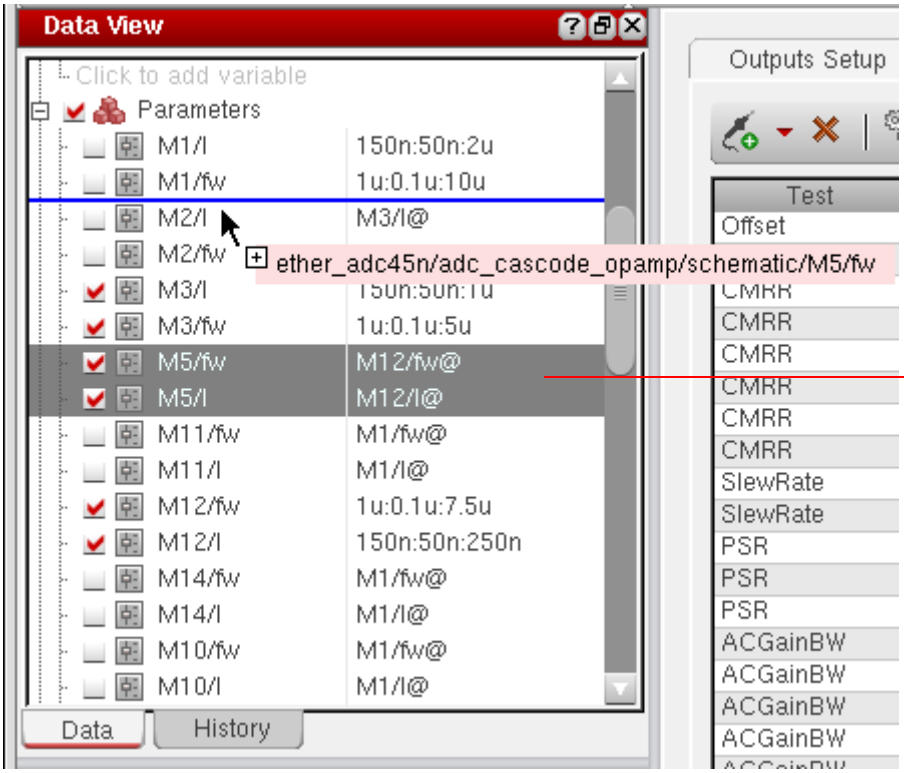
When there are multiple parameters in the *Parameters* list of the *Data View* assistant, you can change the order in which they are listed. A change in the order of device parameters in this assistant changes the nesting of sweep points as well.

To change the order of a parameter, do the following in the *Data* tab of the *Data View* assistant:

1. Expand *Parameters* to view all the parameters.
2. Click a parameter name and drag it.

   A blue indicator line appears at the location where you have moved it. The tooltip also shows the name of the parameter being moved.

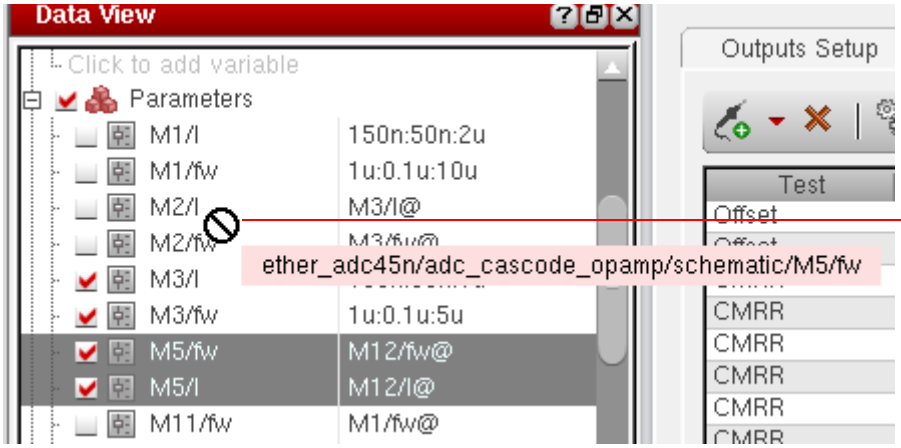
   Note: Since all the parameters for a device are placed together, you cannot move only a single parameter for a device to another location in the list. Therefore, while you are moving a device parameter, all the other parameters for the device are also highlighted and moved, as shown below.



All the parameters for the device M5 are moved together.

© 2023 Cadence Design Systems, Inc.

While you are dragging the parameters, you cannot drop the device parameters in the midst of a list of parameters for another device. The pointer changes from a blue line to a stop icon to indicate that the movement is not allowed.



The icon indicates that you cannot move and place the parameters for M5 in the midst of the list of parameters for M2.

3. Drag the indicator line to the desired location and drop the parameter.

You can also select the parameters for more than one device and move them together.

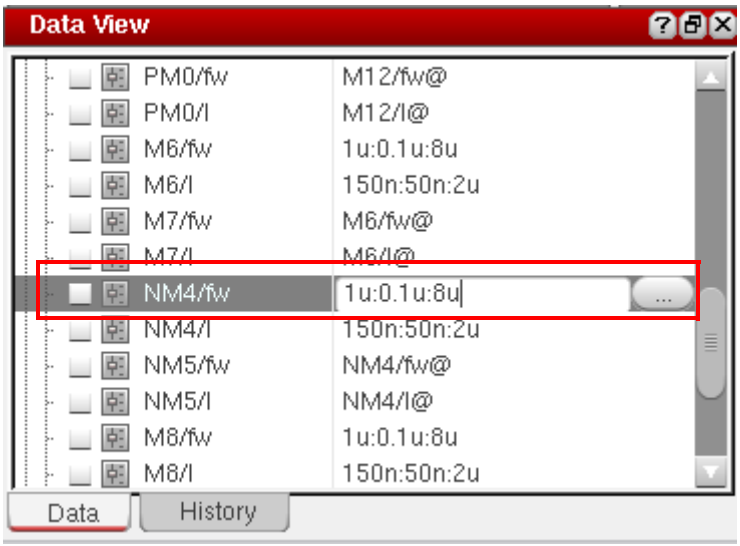
© 2023 Cadence Design Systems, Inc.

Creating Parameter Ranges

Video

You can view a video demonstration for this feature at Create Design Parameter Ranges in the ADE Assembler.

You can specify a parameter range for an individual parameter by editing the value field in the Data View pane or the Variables and Parameters assistant, as shown below.



However, if the parameter list is long, applying a parameter range for each parameter individually can take a long time. Instead, you can use the *Create Parameter Range* command in the context-sensitive menu of the Parameters list. This command applies a standard parameter range to a set of selected or all the parameters in the Parameters list.

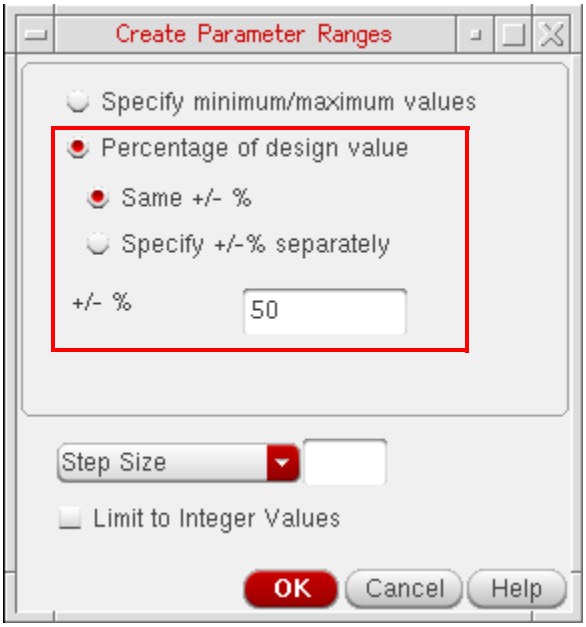
To create a parameter range for more than one or all the parameters together, perform the following steps in Data View pane or the lower part of the Variables and Parameters assistant:

1. Select one or more variables in the Parameters list for which you want to create parameter range. Press the *Shift* or *Ctrl* key and select the parameter names.
2. Right-click a parameter in the Parameters list and choose *Create Parameter Range*.

   Alternatively, click *Create Parameter Range* ( ) on the toolbar given on the Parameters tab of the Variables and Parameters assistant.

© 2023 Cadence Design Systems, Inc.

The Create Parameter Ranges form appears, as shown below.



3. In this form, you can specify parameter ranges in one of the following two ways:

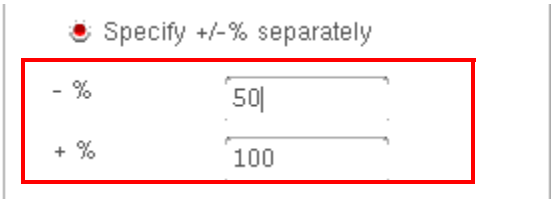
   - **Parameter range as percentage of the design value:** This is the default way. In this case, the *Percentage of design value* option is selected and the parameter range is relative to the value of variable. If the variable is assigned any value in the Data View pane, that value is taken as a reference. If there is no value assigned to this variable in the Data View pane, the value for this variable is taken from the design.

     By default, you can specify a common percentage value for both the upper and the lower limit in the *+/- %* field. The lower limit is the value of a parameter in the design minus the specified % of the same design value. Similarly, the upper limit is the value of a parameter in the design added to the specified % of the value of parameter in the design.

     Note: The +/- % field can contain only an integer value between `1` and `100`, including these two values.

© 2023 Cadence Design Systems, Inc.

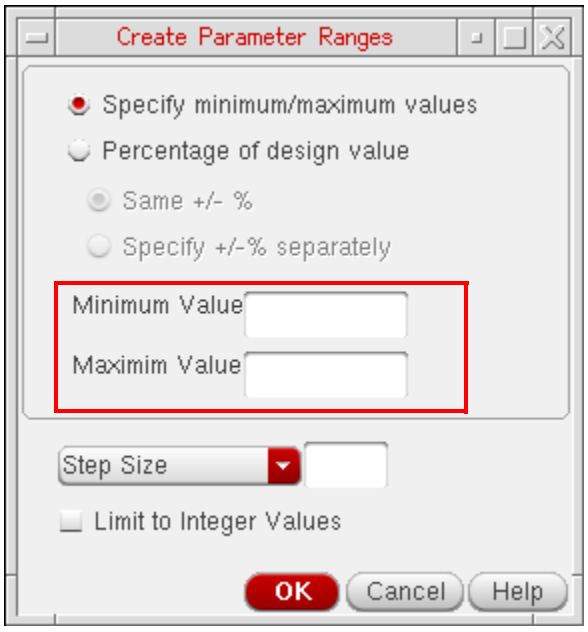
Alternatively, select the *Specify +/- % separately* option. When you select this option, the following fields appear in the form.



In these fields, you can specify separate % values for the upper and lower range limits.

Note: The -% field can contain an integer between `0` and `100`, including these two values, and the *+%* can be set to any non-negative integer.

- **Parameter range with a specific minimum and maximum value:** This option is used to specify specific minimum and maximum value for the parameter range. These values might not be relative to the value of a parameter in the design. To specify these values, select the *Specify minimum/maximum values* option. The *Minimum Value* and *Maximum Value* fields appear in the form.



Specify the minimum and maximum values of the parameter range in their respective fields.

© 2023 Cadence Design Systems, Inc.

Tip

To set the value of a parameter same as given in the design, right click the parameter name and choose *Set to Design Value*.

4. Next, you can specify how to create intermediate values within the range. For this, you have the following two options in the drop-down list displayed in the lower section of the form:

   - *Step Size*: Use this option to specify a step size between two steps. The tool automatically creates steps between the two outer limits of the range at a gap of the given step size. In this case, ranges of all the design parameters will have the same step size, but the number of step values will depend on the minimum and maximum values.
   - *Number of Step Values*: Use this option to specify the number of step values to be created between the two range limits. The tool automatically calculates the required step size and creates the intermediate values. In this case, the number of step values will remain same for all the design parameters, but the step size will vary.

5. Select the *Limit to Integer Values* option to specify that the step values should be only integer numbers. This would be required for parameters that cannot have decimal values, such as mfactor and number of fingers. When you select this option, parameter range is not created if the step values contain decimal values.

   Note: An alternate way to limit the creation of range for parameters defining m-mfactorfactor and number of fingers to only integer values is to add the names of such parameters in the `layoutXL.mFactorNames` and `layoutXL.lxFingeringNames` environment variables. ADE Assembler automatically sets only integer ranges for the parameters listed in these variables.

6. Click *OK* to close the form.

   ADE Assembler creates the parameter ranges for all the parameters and updates the Data View and the Variables and Parameters assistants.

   When the Variables and Parameters assistant is open in the schematic view and you select an instance to create a parameter, you can right-click the instance in the top area of the assistant and use the *Create Parameter Range* command to create a parameter as well as to apply a range of values for it.

© 2023 Cadence Design Systems, Inc.

Working with Parametric Sets

By default, ADE Assembler creates all possible sweep combinations by pairing each value of a variable or parameter with all values of the other variables or parameters. By using parametric sets, you can create a selected set of sweep combinations by picking values from the same ordinal position for all the variables or parameters in that parametric set. This reduces the number of design points, therefore, reducing the number of simulations.

You can create parametric sets when you need to run simulations for a specific set of sweep values for different parameters. This helps in saving the simulation run time.

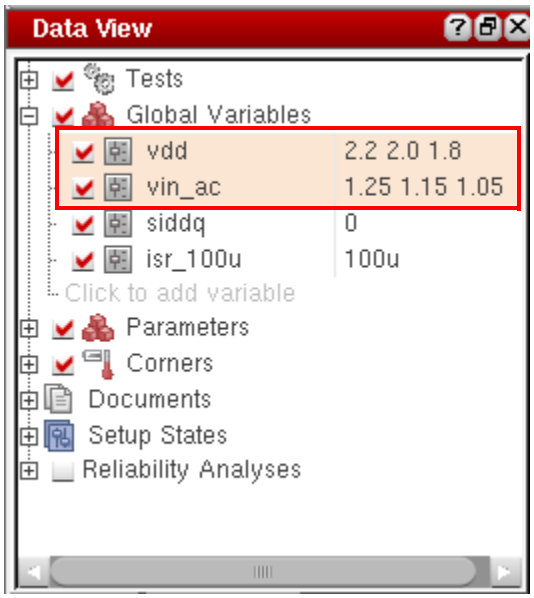
In ADE Assembler, you can create parametric sets for the design variables at the test or global level, and for device parameters.

Video

You can view a video demonstration for this feature at Creating Parameter Sets in the ADE Assembler.

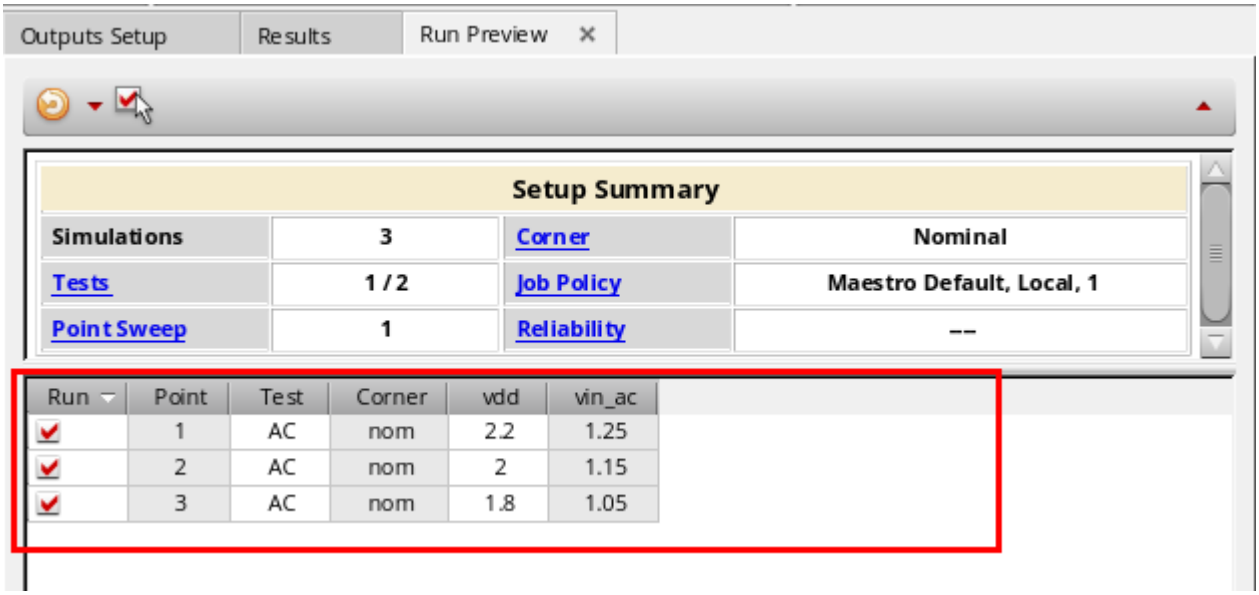
Creating Parametric Sets in ADE Assembler

Consider an example where two global variables, `vdd` and `vin_ac`, are grouped together to create a parametric set as shown below.



© 2023 Cadence Design Systems, Inc.

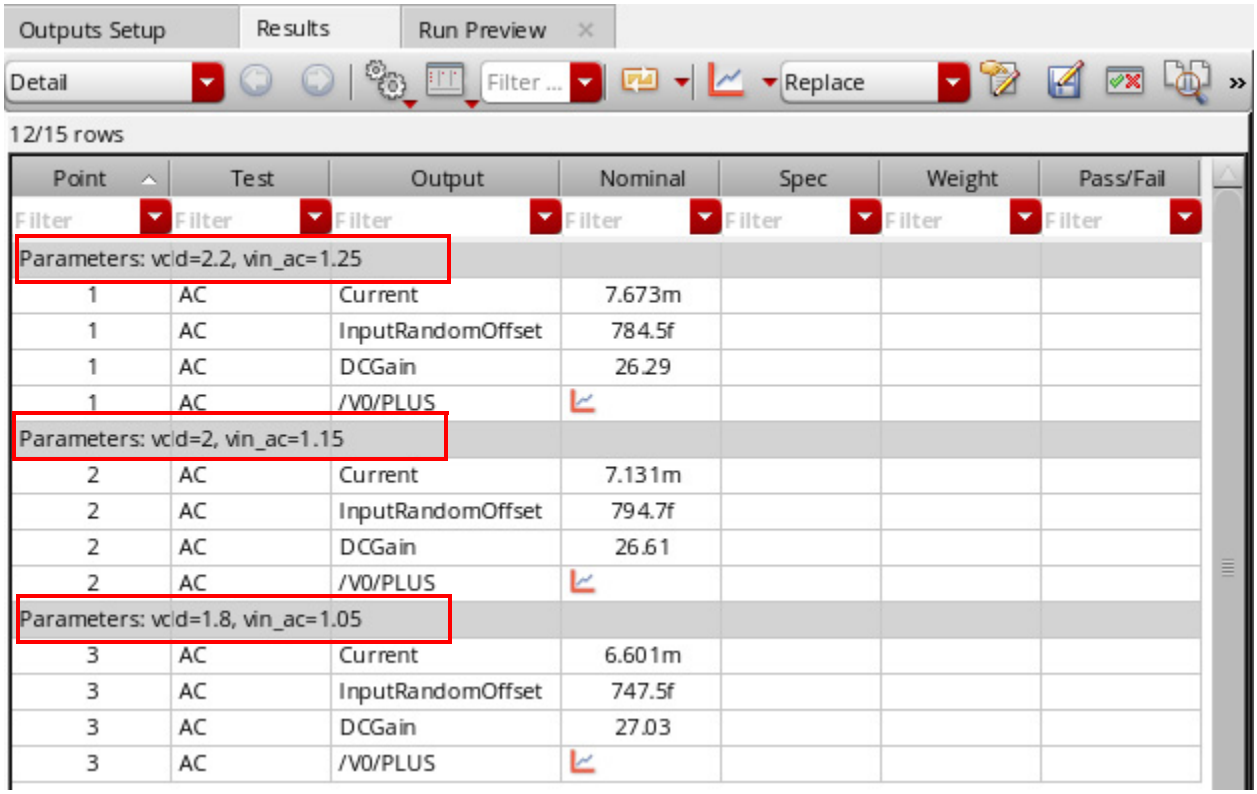
In this case, ADE Assembler creates three sweep value sets that you can view in the *Run Preview* tab, as shown below.



The sweep value sets are created by picking values from the same ordinal position for all the variables or parameters in a parametric set.

© 2023 Cadence Design Systems, Inc.

The following figure shows that while running simulations, ADE Assembler creates three design points.



Point	Test	Output	Nominal	Spec	Weight	Pass/Fail
Filter	Filter	Filter	Filter	Filter	Filter	Filter
Parameters: vdd=2.2, vin_ac=1.25						
1	AC	Current	7.673m			
1	AC	InputRandomOffset	784.5f			
1	AC	DCGain	26.29			
1	AC	/V0/PLUS				
Parameters: vdd=2, vin_ac=1.15						
2	AC	Current	7.131m			
2	AC	InputRandomOffset	794.7f			
2	AC	DCGain	26.61			
2	AC	/V0/PLUS				
Parameters: vdd=1.8, vin_ac=1.05						
3	AC	Current	6.601m			
3	AC	InputRandomOffset	747.5f			
3	AC	DCGain	27.03			
3	AC	/V0/PLUS				

As compared to this, if the parametric sets are not created, ADE Assembler creates nine design points, where each value of `vdd` is paired with all three values of `vin_ac`.

Note: If a parametric set is specified for global variables in ADE Assembler, all the variables that are the part of that parametric set must have same number of variable values. Otherwise, the simulation run fails with an error.

Using Parametric Sets Created in ADE Explorer

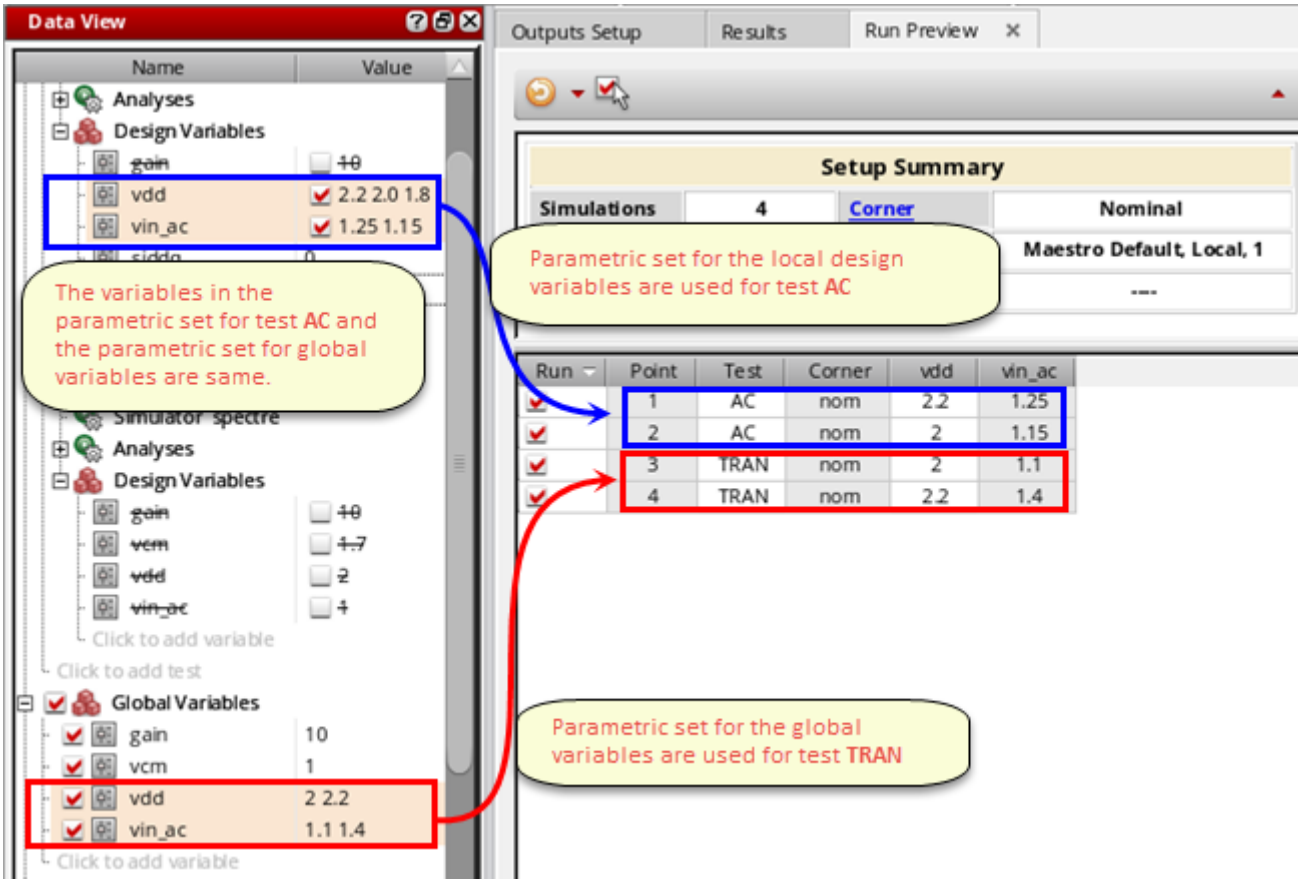
If you create parametric sets for the design variables of a test in ADE Explorer, they are reflected in the Data View assistant in ADE Assembler.

For details on how to create and manage parametric sets in ADE Explorer, refer to Working with Parametric Sets in the *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

Important

A parametric set created for test-level design variables is not propagated to the global level. However, in a scenario when the parametric set created for global variables is a duplicate of a test-level parametric set and the test variables are not overridden, the local parametric set takes precedence. This is shown in the example given below.



Creating Parametric Sets

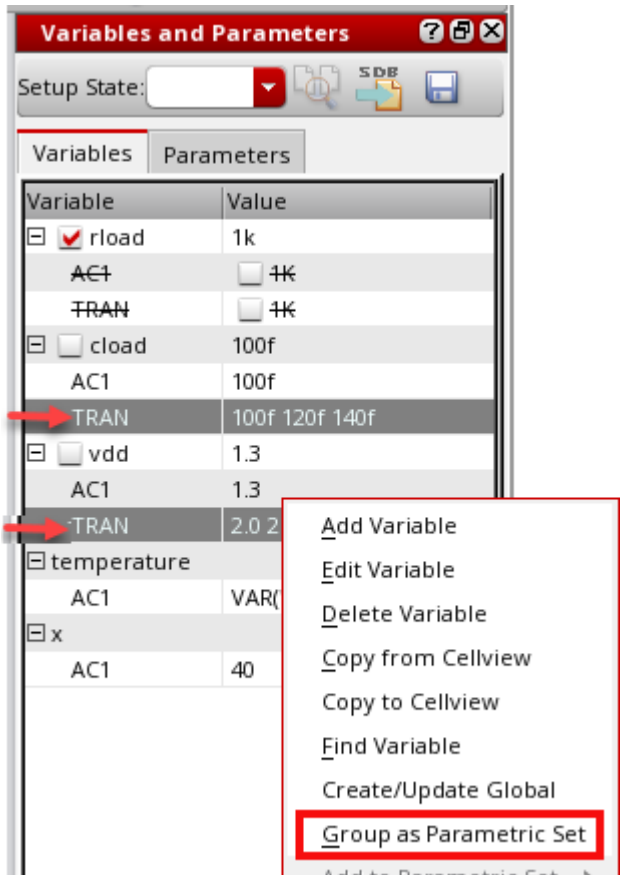
To create a parametric set, do the following in the Data View pane or the Variables and Parameters assistant:

1. Hold down the *Ctrl* key and select two or more variables or parameters that you want to group together.
2. Right-click and choose *Group as Parametric Set*.

   Note: If you are using the Variables and Parameters assistant, click the test name or

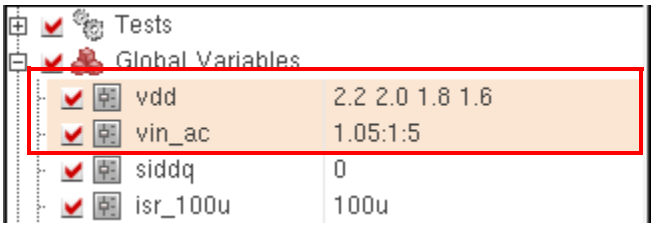
© 2023 Cadence Design Systems, Inc.

variable value in the rows under the variable name, as shown below.



3. If required, specify sweep values for the variables.

   Note: For different variables, you can specify sweep values in different format. For example, you can specify values for variables as shown below.

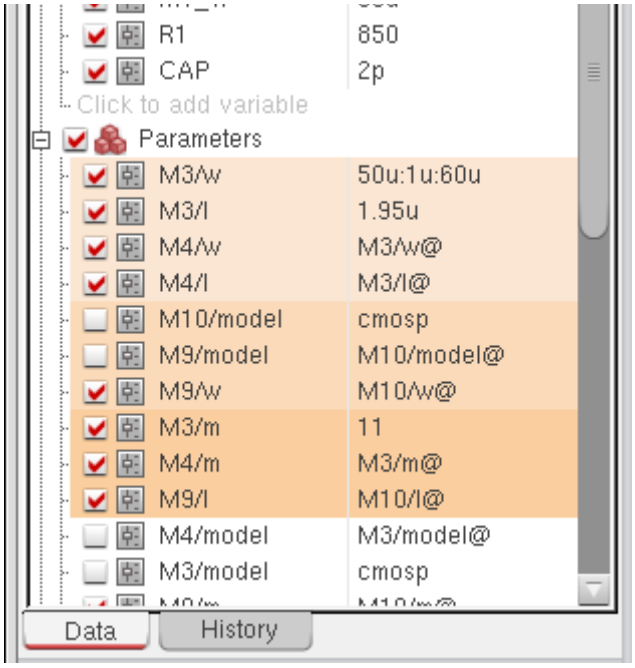


   However, it is important that the number of sweep points is same for all the variables or parameters in a parametric set. If the number of sweep values are not equal, ADE Assembler flags an error during the simulation run.

Note the following changes in the Data View pane and the Variables and Parameters assistant:

© 2023 Cadence Design Systems, Inc.

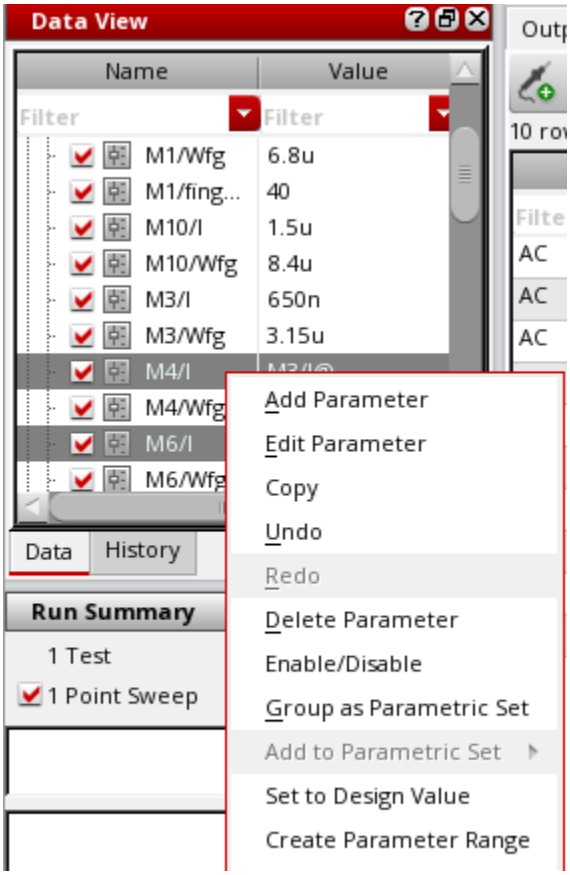
- The variables or parameters in a parametric set are bound as a group.
- When you create a parametric set, the list of variables is realigned to display all variables or parameters of a group together.
- Each parametric set is highlighted with a different color, as shown below.



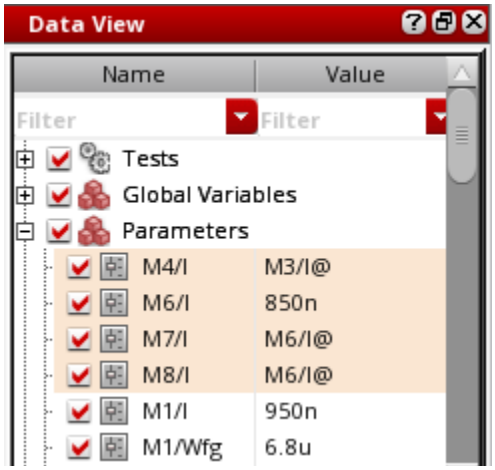
- When you group two more variables or device parameters together, the tool automatically identifies matched parameters and includes them in the parametric set.

© 2023 Cadence Design Systems, Inc.

Consider an example. From the *Parameters* list, select two variables, `M4/l` and `M6/l` and add them to a parametric set, as shown below.



A parametric set is formed and two more variables `M7/l` and `M8/l` are also added to that set. This is because `M4/l` and `M6/l` are matched with `M7/l` and `M8/l`, respectively.



© 2023 Cadence Design Systems, Inc.

- When you move the mouse over any variable in a parametric set, the tooltip shows that the variable is part of a parametric set.
- You cannot create a parametric set by including a mix of global variables and parameters.
- A variable or device parameter cannot be included in more than one parametric set.

Adding a Variable to a Parametric Set

To add a variable to an existing parametric set:

1. Right-click a variable and choose *Add to Parametric Set*.
2. From the list of existing parametric sets shown in the submenu, select the name of the parametric set to which you want to add the parameter.

The variable is added to the specified parametric set. In addition, all the variables are realigned to display all variables or parameters of a group together.

Including a Dependent Variables in a Parametric Set

Dependent variables that have the same number of values as that of the other variables are supported in the parametric sets. The following examples show how dependent variables can be used in parametric sets:

Example 1:

A: 1, 2

B: 7, 8

C: A, B

D: 4, 5

In this example, a parametric set that includes variables C and D, is a valid set. This simulation will create 8 design points, as listed below.

Design point 1: A1=1, B1= 7, {C1=1, D1=4}

Design point 2: A1=1, B1= 7, {C1=7, D1=5}

Design point 3: A1=1, B1= 8, {C1=1, D1=4}

Design point 4: A1=1, B1= 8, {C1=8, D1=5}

© 2023 Cadence Design Systems, Inc.

Design point 5: A1=2, B1= 7, {C1=2, D1=4}

Design point 6: A1=2, B1= 7, {C1=7, D1=5}

Design point 7: A1=2, B1= 8, {C1=2, D1=4}

Design point 8: A1=2, B1= 8, {C1=8, D1=5}

Example 2:

A: 1, 2

B: 7

C: A, B

D: 4, 5

In this example, a parametric set that includes variables C and D, is a valid set. This simulation will create 4 design points, as listed below.

Design point 1: A1=1, B1= 7, {C1=1, D1=4}

Design point 2: A1=1, B1= 7, {C1=7, D1=5}

Design point 3: A1=2, B1= 7, {C1=2, D1=4}

Design point 4: A1=2, B1= 7, {C1=7, D1=5}

Unsupported Dependent Variables

Dependent variables are not supported when:

- A dependent variable does not have the same number of values as that of the other variables

  For example:

  A: 1, 2

  B: A

  C: 3, 4

  In this example, A, B, and C cannot make a parametric set because B has only one value whereas A and C have two values.

© 2023 Cadence Design Systems, Inc.

- When there is cyclic dependency

  For example:

  A: 1, 2

  B: A, A

  C: 3, 4

  In this example, A, B, and C cannot make a parametric set because there is cyclic dependency between A and B.

Removing a Variable from a Parametric Set

To remove a variable from a parametric set:

➡ Right-click that variable and choose *Remove from Parametric Set*.

The variable is removed from the parametric set and moved to its original location in the list.

Ungrouping Parametric Sets

To ungroup the parametric set:

➡ Right-click any variable name in the group and choose *Ungroup Parametric Set*.

The group is dissolved and the variables are moved back to their original location in the Global Variables or Parameters list.

Enabling or Disabling Parametric Sets

You can disable or enable all parametric sets present in the Global Variables or Parameters list in the Data View pane.

To disable all parametric sets:

➡ In the Parameters list, right-click *Parameters* and choose *Disable All Parametric Sets*.

After the parametric sets are disabled, the variables are moved back to their original location in the list.

© 2023 Cadence Design Systems, Inc.

Note: If the parametric sets are disabled, you can enable all the sets together. To do this, right-click the list name, Global Variables or Parameters and choose *Enable All Parametric Sets*. All the parametric sets become visible again.

Enabling and Disabling Parameters

You can enable or disable parameters before starting your simulation. This allows you to specify the set of parameters you want to be used for a particular simulation.

To enable a parameter,

➤ In the *Parameters* tree on the Data View pane or the Parameters tab on the Variables and Parameters pane, select the check box to the left of the parameter.

Alternatively, right-click the parameter and choose *Enable/Disable* from the context-sensitive menu.

To disable a parameter,

➤ In the *Parameters* tree on the Data View pane or the Parameters tab on the Variables and Parameters pane, clear the check box to the left of the parameter.

Alternatively, right-click the parameter and choose *Enable/Disable* from the context-sensitive menu.

Note: When a parameter is disabled, its value given on the Data View pane is not used for simulation, but its definition is retained in the *Parameters* tree on the Data View pane and the Parameters tab on the Variables and Parameters pane. The value specified for the CDF parameter on the instance will be used for simulation.

Tip

To enable or disable more than one parameter at a time, select the parameters, right-click and choose *Enable/Disable* from the context-sensitive menu.

To toggle the enabling or disabling of one or more parameters,

➤ On the Parameters tab of the Variables and Parameters pane, select one or more parameters, right-click and choose *Toggle Enable/Disable*.

The enabled parameters are disabled and the disabled parameters are enabled.

To disable all parameters, do any of the following:

■ On the Data View pane, clear the check box to the left of *Parameters*.

© 2023 Cadence Design Systems, Inc.

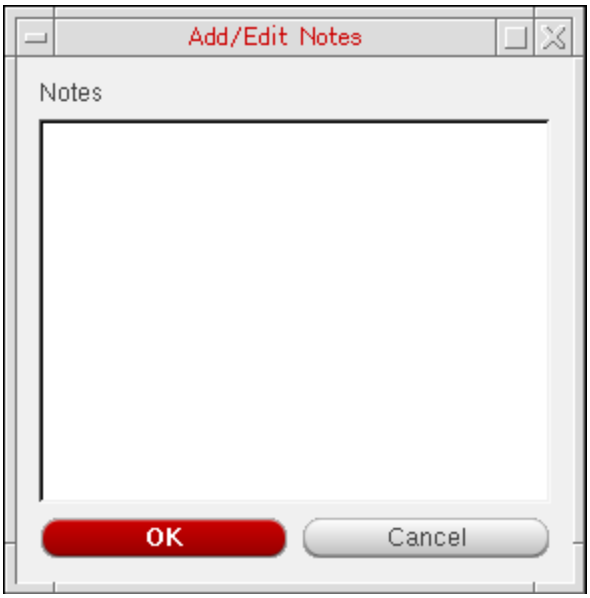
- Right-click the Parameters tab of the Variables and Parameters pane and choose *Enable All/Disable All*.

Adding Notes for Parameters

If you need to save some additional reference information about a device parameters, you can add notes for it. For this, do the following:

1. On the *Data* tab of the Data View assistant pane, expand the *Parameters* tree, right-click the parameter for which you want to add notes, and choose *Notes*.

   The Add/Edit Notes form is displayed.



2. In the *Notes* field, add notes for the device parameter.

   Note: By default, the notes field can accept only 512 characters. To change the default maximum characters limit, you can set the maxNotesLength environment variable.

3. Click *OK*.

The notes is added to the parameter. This is displayed in the tooltip for the parameter and is saved with the setup state.

For related information, see Adding Notes to a Test.

© 2023 Cadence Design Systems, Inc.

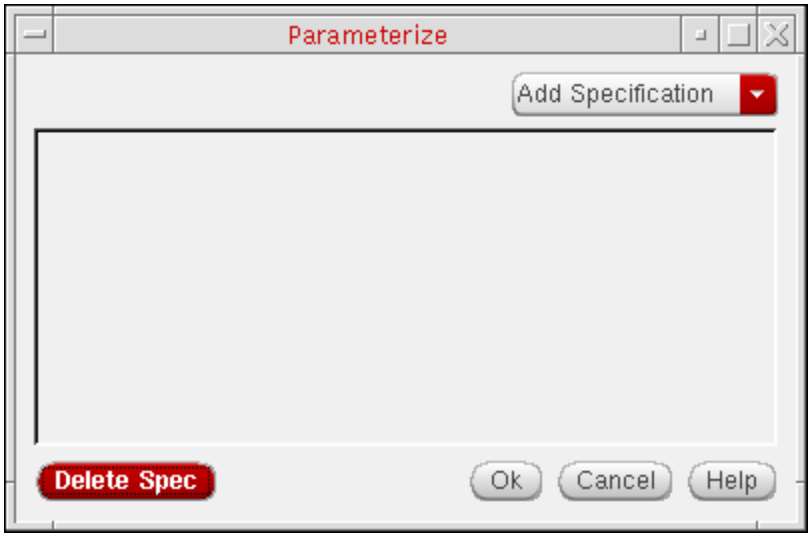
Adding or Changing a Variable or Parameter Specification

To add or change the specification for a variable or device parameter you want to vary (sweep), do the following:

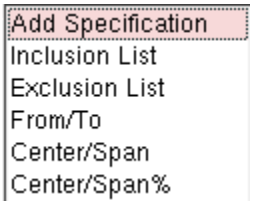
1. In the Global Variables tree on the Data View pane or the Variables tab of the Variables and Parameters pane, right-click the global variable for which you want to add or change the specification and choose *Edit Variable*.

   Alternatively, double-click the *Value* field next to the global variable, then click the browse button.

   The Parameterize form appears.



2. Click *Add Specification* and choose one of the following from the drop-down menu:



- *Inclusion List* of values — You can specify a set of values through which you want to sweep your design variable.
- *Exclusion List* of values — You can specify a set of values to exclude from your parametric sweep.

© 2023 Cadence Design Systems, Inc.

■ *From/To* range of values	You can specify a range of values through which you want to sweep your design variable and a method for stepping through that range.
■ *Center/Span* values	You can specify a center value and a span value for varying your design variable.
■ *Center/Span%* values	You can specify a center value and a percentage span value for varying your design variable.

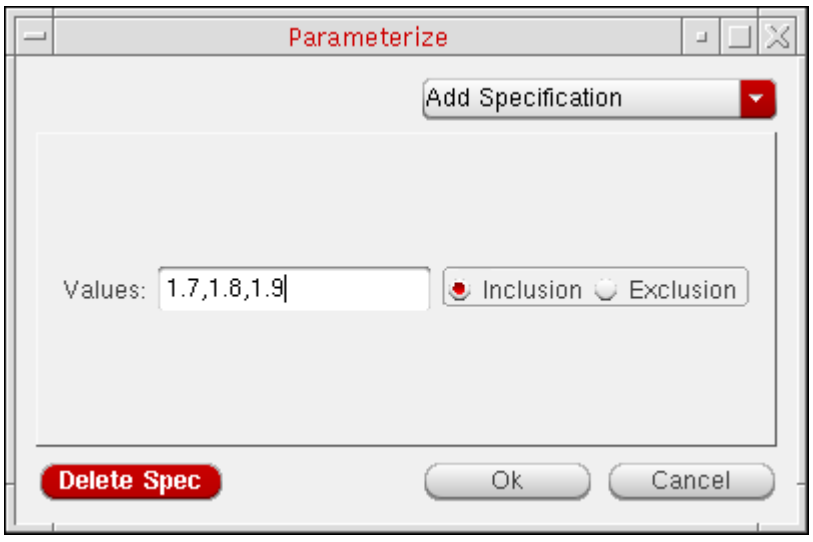
Specifying an Inclusion List of Values

To specify an inclusion list of values, do the following:

1. From the *Add Specification* drop-down menu, choose *Inclusion List*.

   A row appears on the Parameterize form for defining the specification type you selected. The *Inclusion* radio button is on.

2. In the *Values* field, type a list of values you want to include in your sweep.



   Note: You can separate the values in the list with either a comma or a space.

3. (Optional) Click *Add Specification* to add another parameter specification.

4. Click *OK*.

   The list of values appears in the *Value* column beginning and ending with the *{Inclusion List}* text string.

© 2023 Cadence Design Systems, Inc.

Note: You can change an inclusion list of values to an exclusion list (to exclude the list of values from the sweep) by selecting the *Exclusion* radio button in the row where your inclusion list is specified. See "Specifying an Exclusion List of Values" on page 230.

Specifying an Exclusion List of Values

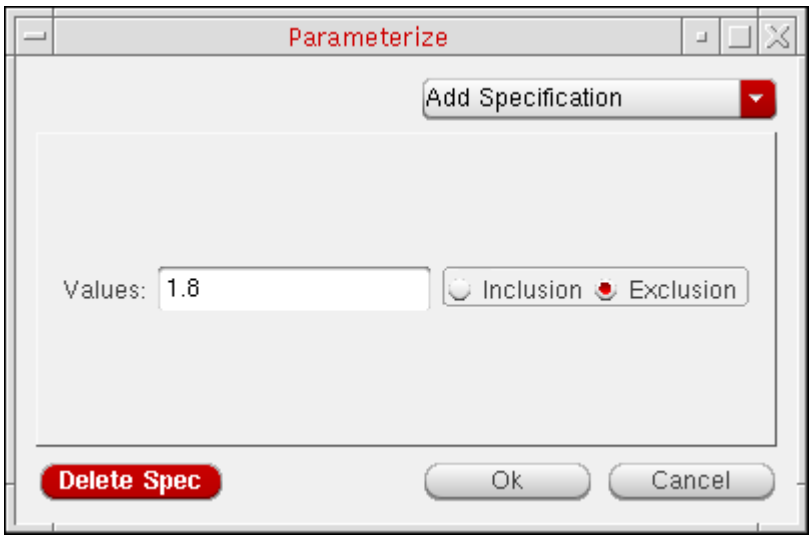
Tip

One application of the exclusion list is to define a *From/To* parameter specification (see "Specifying a Range of Values" on page 232) and exclude some values from the range by defining an exclusion list.

To specify an exclusion list of values, do the following:

1. From the *Add Specification* drop-down menu, choose *Exclusion List*.

   A row appears on the Parameterize form for defining the specification type you selected. The *Exclusion* radio button is on.



2. In the *Values* field, type a list of values you want to exclude from your sweep.

   Note: You can separate the values in the list with either a comma or a space.

3. (Optional) Click *Add Specification* to add another parameter specification.

4. Click *OK*.

   The list of values appears in the *Value* column beginning and ending with the *{Exclusion List}* text string.

© 2023 Cadence Design Systems, Inc.

Note: You can change an exclusion list of values to an inclusion list (to include the list of values in the sweep) by selecting the *Inclusion* radio button in the row where your exclusion list is specified. See "Specifying an Inclusion List of Values" on page 229.

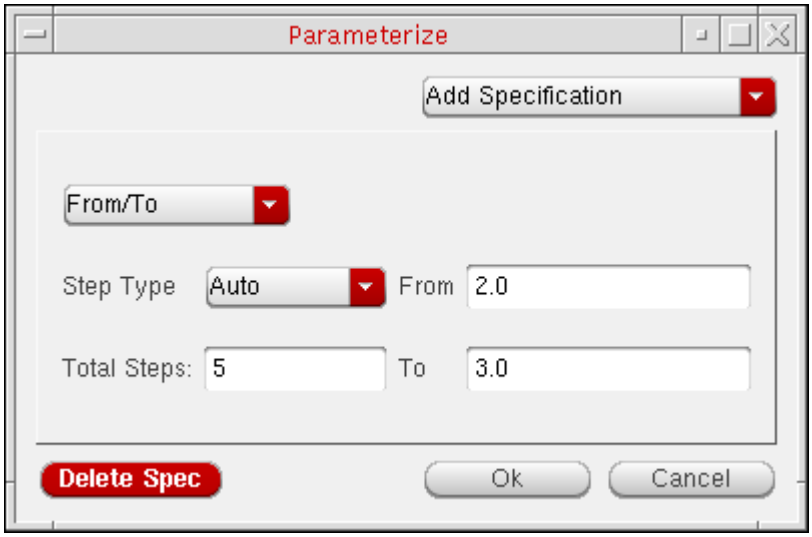
© 2023 Cadence Design Systems, Inc.

Specifying a Range of Values

To specify a range of values and how to sweep through those values, do the following:

1. From the *Add Specification* drop-down menu, choose *From/To*.

   A section appears on the Parameterize form for defining the specification type you selected.



2. In the *From* field, type the starting value for your range.
3. In the *To* field, type the ending value for your range.
4. From the *Step Type* drop-down menu, select how you want the simulator to step from one value to the next:

Linear	Linear steps from the *From* value to the *To* value
Decade	Decade steps from the *From* value to the *To* value
Octave	Octave steps from the *From* value to the *To* value
Logarithmic	Logarithmic steps from the *From* value to the *To* value
Times	Steps taken according to a specified multiplier from the *From* value to the *To* value

© 2023 Cadence Design Systems, Inc.

5. Depending on the *Step Type* you selected, type the remaining value for your parameter specification in the field that appears:

Step Type	Field that appears	Value
Auto	*Total Steps*	Total number of steps to take
Linear	*Step Size*	Size of steps to take when sweeping
Decade	*Steps/Decade*	How many steps to take per decade
Octave	*Steps/Octave*	How many steps to take per octave
Logarithmic	*Total Steps*	Total number of steps to take
Times	*Multiplier*	Multiplier for taking steps **Note:** Typing `10` for *Multiplier* is tantamount to selecting *Decade* and specifying `1` step per decade.

6. (Optional) Click *Add Specification* to add another parameter specification.
7. Click *OK*.

Note: You can change the specification type from *From/To* to *Center/Span* or *Center/Span%* by selecting a different item from the drop-down menu in the added section. See "Specifying Center and Span" on page 234 and "Specifying Center and Span as a Percentage" on page 236 for more information.

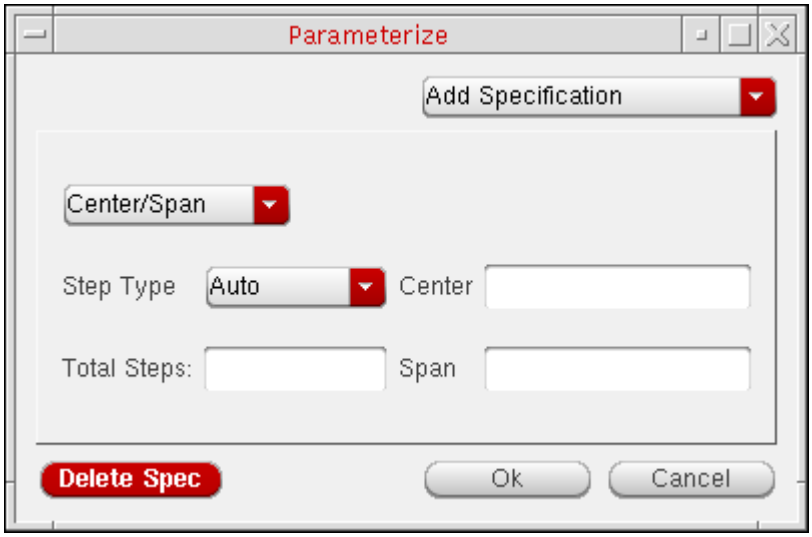
© 2023 Cadence Design Systems, Inc.

Specifying Center and Span

To specify a center value and a span value, do the following:

1. From the *Add Specification* drop-down menu, choose *Center/Span*.

   A section appears on the Parameterize form for defining the specification type you selected.



2. In the *Center* field, type the center value.
3. In the *Span* field, type a span value.

   The simulator can vary your parameter between *Center−Span* and *Center+Span* according to the *Step Type* you select (next).

4. From the *Step Type* drop-down menu, select how you want the simulator to vary the parameter value:

Linear	Linear steps from *Center−Span/2* to *Center+Span/2*
Decade	Decade steps from *Center−Span/2* to *Center+Span/2*
Octave	Octave steps from *Center−Span/2* to *Center+Span/2*
Logarithmic	Logarithmic steps from *Center−Span/2* to *Center+Span/2*
Times	Steps taken according to a specified multiplier from *Center−Span/2* to *Center+Span/2*

© 2023 Cadence Design Systems, Inc.

5. Depending on the *Step Type* you selected, type the remaining value for your parameter specification in the field that appears:

Step Type	Field that appears	Value
Auto	*Total Steps*	Total number of steps to take
Linear	*Step Size*	Size of steps to take when varying
Decade	*Steps/Decade*	How many steps to take per decade
Octave	*Steps/Octave*	How many steps to take per octave
Logarithmic	*Total Steps*	Total number of steps to take
Times	*Multiplier*	Multiplier for taking steps **Note:** Typing `10` for *Multiplier* is tantamount to selecting *Decade* and specifying `1` step per decade.

6. (Optional) Click *Add Specification* to add another parameter specification.
7. Click *OK*.

Note: You can change the specification type from *Center/Span* to *Center/Span%* or *From/To* by selecting a different item from the drop-down menu in the added section. See "Specifying Center and Span as a Percentage" on page 236 and "Specifying a Range of Values" on page 232 for more information.

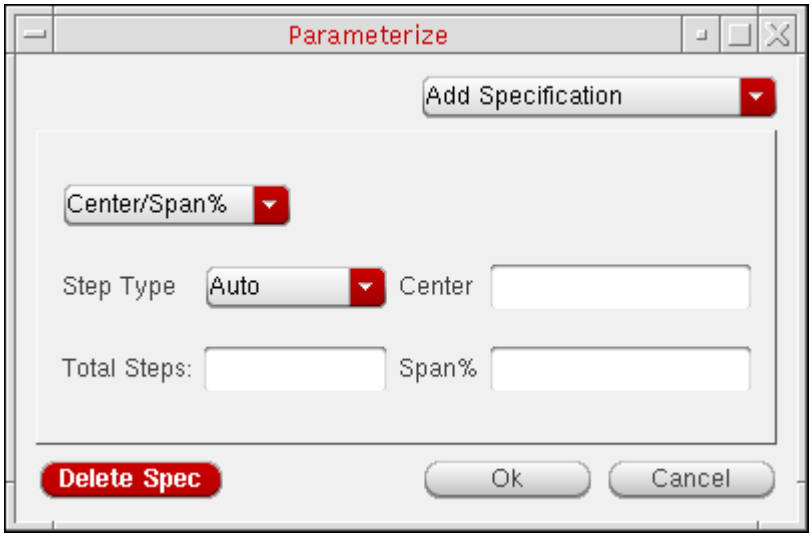
© 2023 Cadence Design Systems, Inc.

Specifying Center and Span as a Percentage

To specify a center value and a span percentage, do the following:

1. From the *Add Specification* drop-down menu, choose *Center/Span%*.

   A section appears on the Parameterize form for defining the specification type you selected.



2. In the *Center* field, type the center value.
3. In the *Span* field, type a span value.

   The simulator can vary your parameter between $Center - \left(\frac{(Span\% \times Center)}{100}\right)/2$ and $Center + \left(\frac{(Span\% \times Center)}{100}\right)/2$ according to the *Step Type* you select (next).

4. From the *Step Type* drop-down menu, select how you want the simulator to vary the parameter value:

   Linear — Linear steps from $Center - \left(\frac{(Span\% \times Center)}{100}\right)/2$ to $Center + \left(\frac{(Span\% \times Center)}{100}\right)/2$

   Decade — Decade steps from $Center - \left(\frac{(Span\% \times Center)}{100}\right)/2$ to $Center + \left(\frac{(Span\% \times Center)}{100}\right)/2$

© 2023 Cadence Design Systems, Inc.

Octave Octave steps from $Center - \left(\frac{(Span\% \times Center)}{100}\right)/2$ to $Center + \left(\frac{(Span\% \times Center)}{100}\right)/2$

Logarithmic Logarithmic steps from $Center - \left(\frac{(Span\% \times Center)}{100}\right)/2$ to $Center + \left(\frac{(Span\% \times Center)}{100}\right)/2$

Times Steps taken according to a specified multiplier from $Center - \left(\frac{(Span\% \times Center)}{100}\right)/2$ to $Center + \left(\frac{(Span\% \times Center)}{100}\right)/2$

5. Depending on the *Step Type* you selected, type the remaining value for your parameter specification in the field that appears:

Step Type	Field that appears	Value
Auto	*Total Steps*	Total number of steps to take
Linear	*Step Size*	Size of steps to take when varying
Decade	*Steps/Decade*	How many steps to take per decade
Octave	*Steps/Octave*	How many steps to take per octave
Logarithmic	*Total Steps*	Total number of steps to take
Times	*Multiplier*	Multiplier for taking steps **Note:** Typing `10` for *Multiplier* is tantamount to selecting *Decade* and specifying `1` step per decade.

6. (Optional) Click *Add Specification* to add another parameter specification.
7. Click *OK*.

Note: You can change the specification type from *Center/Span%* to *Center/Span* or *From/To* by selecting a different item from the drop-down menu in the added section. See "Specifying Center and Span" on page 234 and "Specifying a Range of Values" on page 232 for more information.

© 2023 Cadence Design Systems, Inc.

Deleting a Parameter Specification

To delete a parameter specification, do the following:

➤ On the Parameterize form, right-click in the row or section for the parameter specification you want to delete and choose *Delete Specification* from the context-sensitive menu. (*Delete Specification* is the only item on this context-sensitive menu.)

To delete the last parameter specification listed in the form, do the following:

1. On the Parameterize form, click *Delete Spec*.

   The last specification listed in the form is deleted.

2. Click *OK*.

Note: You can delete all but one specification in the form. If you want to delete the only remaining specification in the form, delete the values specified for the specification and click *OK*.

Creating Matched Device Parameters

You can create matched parameters for device matching such that one device's parameters track with another's. To specify two or more devices to match, do the following:

1. On your schematic, click to add the first device, then *Shift*+click to add one or more additional devices.

   As you click, each device and its parameters appear on the upper half of the Parameters tab of the Variables and Parameters assistant pane.

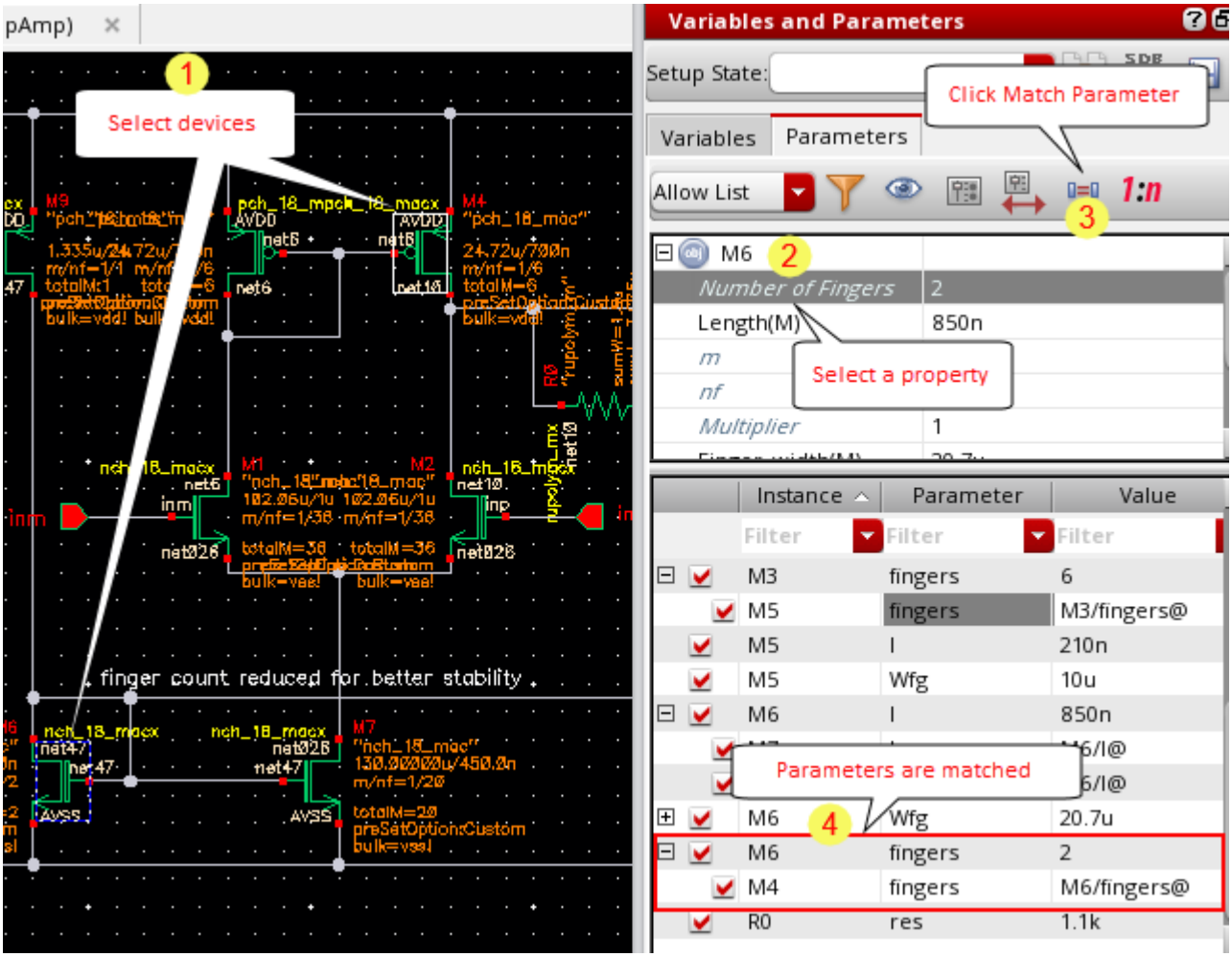
2. In the upper half of the Parameters tab of the Variables and Parameters assistant pane, select (highlight) the device you want to be the primary device against which all other devices are matched.

3. In the upper half of the Parameters tab, click a property that you want to match between the selected devices.

4. Click *Match Parameters* on the toolbar.

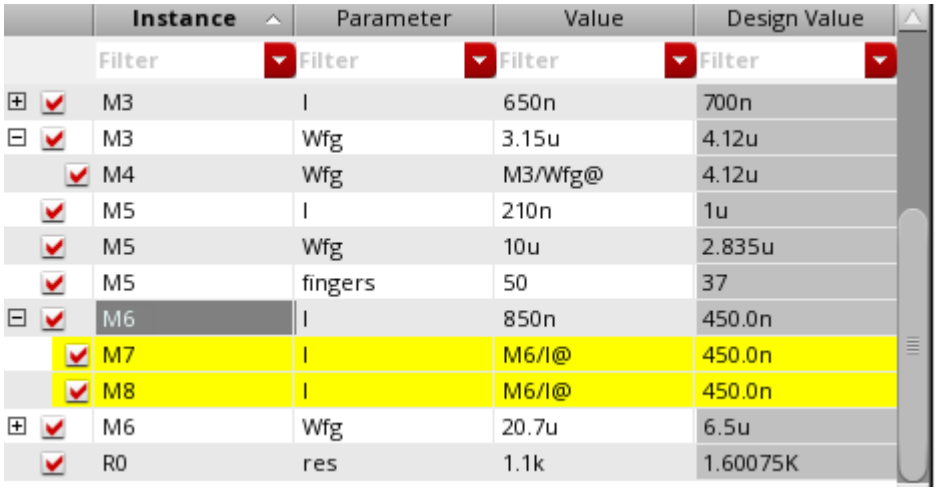
© 2023 Cadence Design Systems, Inc.

The instance name of the primary device for which matched parameters exist appears in the lower half of the Parameters tab of the Variables and Parameters assistant pane.



© 2023 Cadence Design Systems, Inc.

5. (Optional) To view the devices whose parameters are matched to an instance, select the instance and parameter combination and click the plus sign to the left of an instance name in the lower half of the *Parameters* tab.



Instance	Parameter	Value	Design Value
Filter	Filter	Filter	Filter
M3	l	650n	700n
M3	Wfg	3.15u	4.12u
M4	Wfg	M3/Wfg@	4.12u
M5	l	210n	1u
M5	Wfg	10u	2.835u
M5	fingers	50	37
M6	l	850n	450.0n
M7	l	M6/l@	450.0n
M8	l	M6/l@	450.0n
M6	Wfg	20.7u	6.5u
R0	res	1.1k	1.60075K

The value of a matched parameter appears in the *Value* column as follows to indicate the matching relationship:

`primaryInstName/deviceParam@`

For example, in the above figure, the value `M6/l@` for instance `M7` indicates that the `length` parameter of instance `M7` is matched to the `l` parameter of instance `M6` (the primary device).

Note: You can also match parameters across different levels of design hierarchy.

Tip

You can also right-click in the lower half of the Parameters tab and choose *Expand All* to view the devices whose parameters are matched to all primary devices. Right-click and choose *Collapse All* to collapse the expanded view.

For information about using matched device parameters in the context of optimization, see Matching Devices and Device Properties.

Creating Ratio-Matched Device Parameters

You can create ratio-matched parameters for device tracking such that one device's parameters track with the ratio of another's. To specify two or more devices to ratio-match, do the following:

© 2023 Cadence Design Systems, Inc.

1. On your schematic, click to add the first device. Next, *Shift*+click to add each additional device.

   As you click, each device and its parameters appears in the panel on the upper section of the Parameters tab of the Variables and Parameters assistant pane.

   Note: You can also create ratio-matched parameters for the devices in different hierarchy levels of the design. For more details, see .

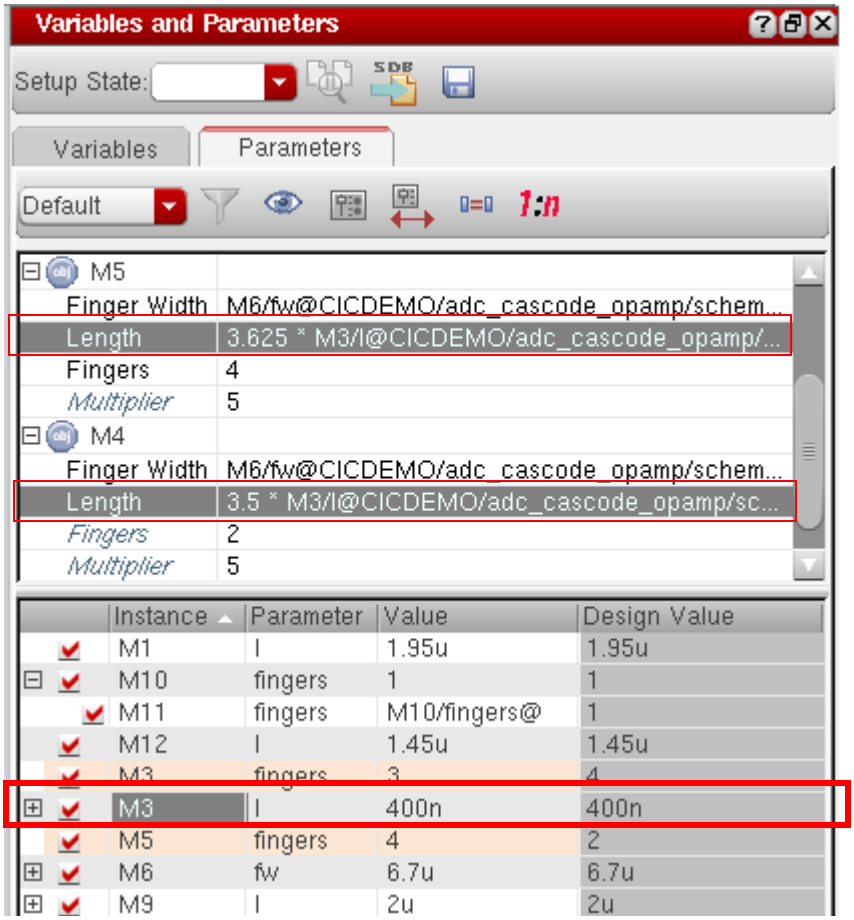
2. Select a device in the upper section on the Parameters tab of the Variables and Parameters assistant pane.

3. Click *Ratio Matched Parameters* .

   The ratio-matched parameters are created. A row is created in the lower section of the assistant pane and name of the primary device (that appeared on top in the upper section of this pane) and the parameter name is displayed in that.

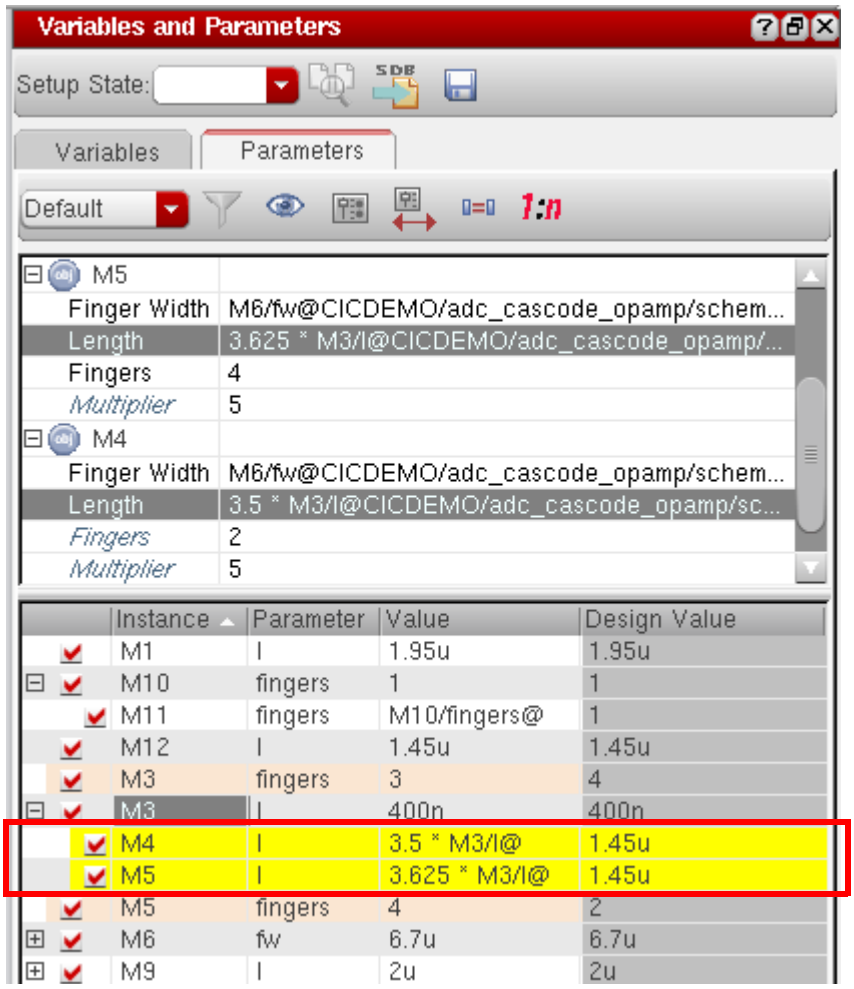
© 2023 Cadence Design Systems, Inc.

For example, if you selected three devices, `M3`, `M4`, and `M5`, and created a ratio-matched parameter for the parameter length, a new row displaying `M3` and `l` is created, as shown below.



© 2023 Cadence Design Systems, Inc.

4. (Optional) To view the devices whose parameters are ratio-matched to the primary device, click the plus (+) sign to the left of an instance name in the lower half of the Parameters tab.



The value of a ratio-matched parameter appears in the *Value* column as follows to indicate the matching relationship:

`ratio*primaryInstName/deviceParam@`

For example, in the above figure, the value 3.5*`M3/l@` for instance `M4` indicates that the `l` parameter of instance `M4` is ratio-matched to the `l` parameter of instance `M3` (the primary device).

Tip

You can also right-click in the lower half of the Parameters tab and choose *Expand All* to view the devices whose parameters are matched to all primary devices. Right-click and choose *Collapse All* to collapse the expanded view.

© 2023 Cadence Design Systems, Inc.

For information about using ratio-matched device parameters in the context of optimization, see Ratio-Matching Device Properties.

Disabling Callbacks on Swept Device Parameters

By default, the program executes callbacks on device parameters that you sweep.

To disable callbacks on swept device parameters, do the following:

➤ In your `.cdsinit` file, set the `axlExecuteCallbacks` variable to `nil`:

```
axlExecuteCallbacks=nil
```

For more information about callbacks in parameter expressions, see also

- Scope of Parameters
- Triggering callbacks in the Advice and Warnings appendix of the *Component Description Format User Guide*.

Scope of Parameters

You can use design variables and CDF parameters to set component values.

Note: Do not use callbacks on parameters whose values are expressions, particularly expressions that use pPar. The expressions might not be evaluated correctly and the system does not detect the errors. In general, try to avoid callbacks whenever possible.

Inheriting from the Same Instance: iPar()

When a parameter value must depend on the value of another parameter on the current instance, use the iPar function.

```
iPar( "CDF_parameter_name" )
```

The value of this expression is the value of this parameter on the current instance, or its value on the cell's effective CDF.

For example, suppose the parameter `AD` of a MOS transistor is a function of its channel width. You could define `AD` in the Schematic window using the *Edit – Properties* command as

```
iPar("w")*5u
```

The resulting value is the value of `w` on the instance times 5u.

© 2023 Cadence Design Systems, Inc.

The iPar expression is substituted with the value of the parameter, enclosed by parentheses, during netlisting. If no value is found, the system reports an error.

Passed Parameter Value of One Level Higher: pPar()

When a parameter expression must depend on the value of a passed parameter, use the pPar function.

```
pPar( "CDF_parameter_name" )
```

The value of this expression is the value of the passed parameter.

For example:

```
pPar("vss")
```

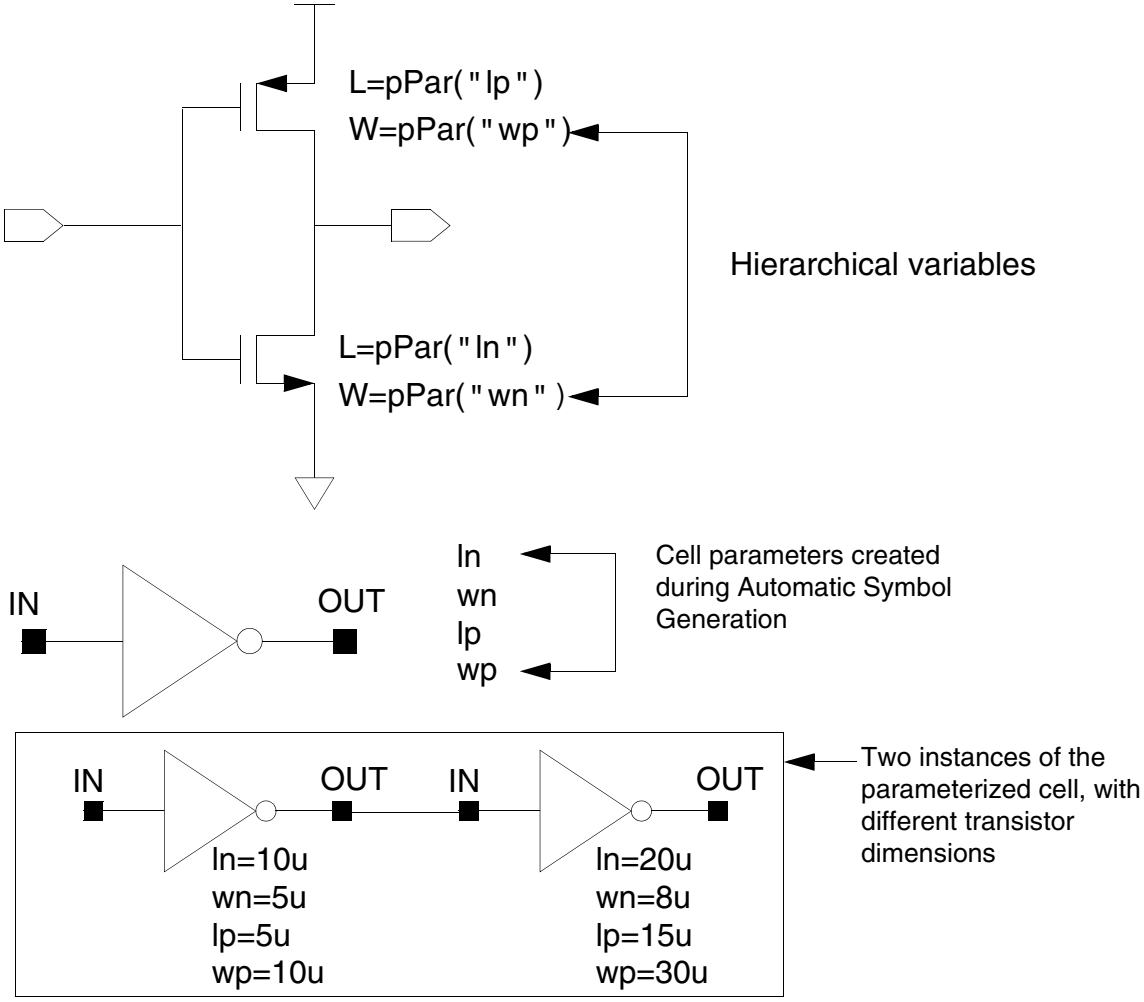
value of the “DC Voltage” parameter on the `v27` instance in the `aExamples opamp` schematic is specified for the `aExamples lowpass` schematic as `15`.

When you create new symbols using automatic symbol generation (the *Create Cellview – From Cellview* command in the Schematic window), the system creates component parameters for the parameters you defined with pPar. The following illustration gives an example of the automatic symbol generation process.

During netlisting, the pPar expression is substituted by the name of the parameter.

© 2023 Cadence Design Systems, Inc.

Using Ppar



Passed Parameters from Any Higher Level: atPar()

You should avoid using atPar. Use pPar instead.

Inheriting from the Instance Being Netlisted: dotPar()

You should avoid using dotPar. Use iPar instead.

© 2023 Cadence Design Systems, Inc.

Table of Functions

Parameters can be inherited by algebraic expressions that are used as component values. The Analog Expression Language (AEL) provides functions to control how parameters are inherited. The AEL inheritance functions are compatible with the corresponding NLP functions shown in the following table.

Functions		Meaning	Scope Rules
AEL	**NLP (OSS)**		
iPar	[~	Instance parameter	Search the instance carrying iPar, then the effective cell CDF
pPar	[+	Parent parameter	Search the parent instance, then the effective cell CDF of the parent instance

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

4

Working with Global Variables

Global variables are the variables defined globally in the ADE Assembler setup. Being global, these variables can be used by all the tests if you need to use a common value in all or multiple tests. Another benefit is that you can override the design variable values to be used in test simulations without modifying the value in the test itself. The simulator will use the value you specified and not the value from the schematic.

This chapter provides details about global variables and how to configure these in ADE Assembler.

For more information, see the following:

- Creating a Global Variable on page 250
- Loading the Default Set of Global Variables on page 252
- Parameterizing Cellviews Defined in the Config View on page 254
- Importing Sweep Variables as Global Variables on page 261
- Saving Parameters and Global Variables to a Saved State on page 261
- Importing Global Variables from a Saved State on page 262
- Exporting Global Variables to a CSV File on page 266
- Importing Global Variables from a CSV File on page 267
- Enabling and Disabling Global Variables for All Tests on page 267
- Disabling Global Variables for Specific Tests on page 269
- Using calcVal for Calibration on page 275

See also

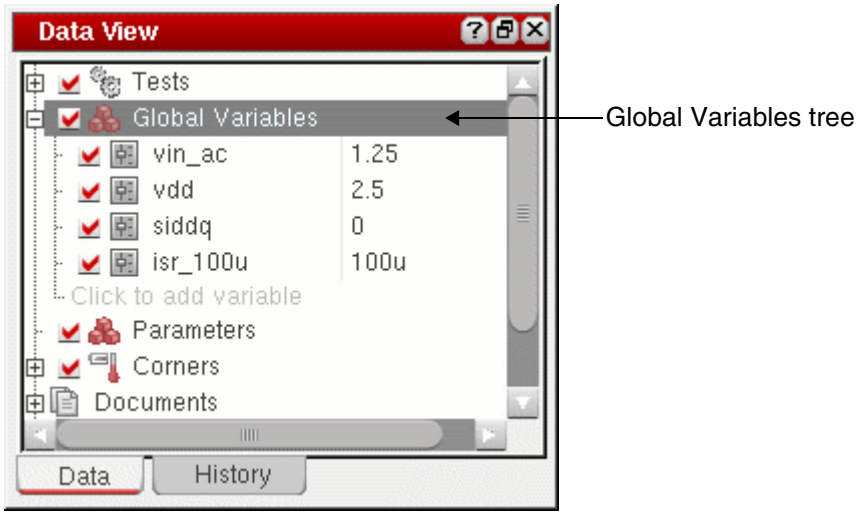
- Varying the Model File and Section during Simulation on page 311

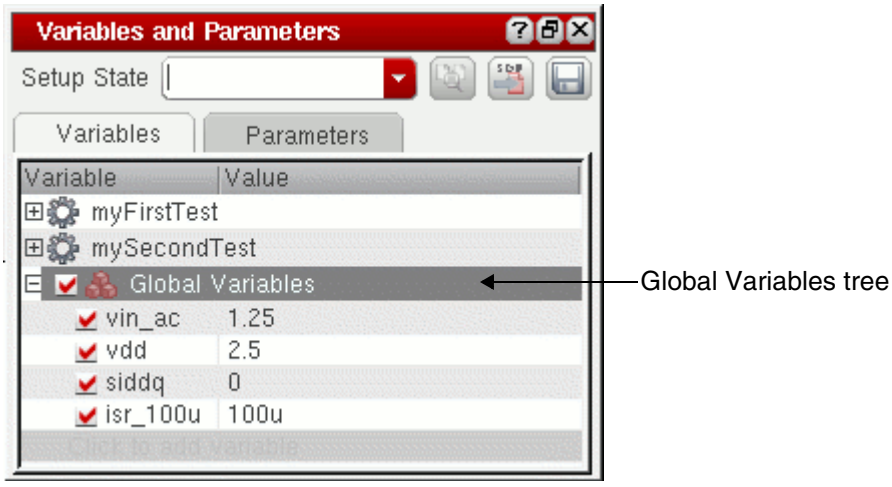
© 2023 Cadence Design Systems, Inc.

Creating a Global Variable

To create a new global variable, do the following:

1. On the Data View assistant pane or the Variables tab of the Variables and Parameters assistant pane, right-click *Global Variables* and choose *Add Global Variable*.





Tip

Alternatively, click where it says *Click to add variable* in an expanded *Global Variables* tree on the Data View pane or the Variables tab of the Variables and Parameters pane.

© 2023 Cadence Design Systems, Inc.

The Create Global Variable form appears.

2. In the *Variable Name* field, type a name for your global variable.

   The name must begin with a letter or number and can contain only letters and numbers.

3. In the *Variable Value* field, type the value for your global variable.

4. Click *OK*.

   Your global variable appears in the *Global Variables* tree on the Date View assistant pane and the Variables tab of the Variables and Parameters assistant pane. A global variable setting takes precedence over all other settings for variables that have the same name (such as a design variable with the same name).

5. To define sweep values for this global variable, do one of the following:

   - Double-click the *Value* field next to the global variable and modify the value.
   - Double-click the *Value* field next to the global variable, then click the browse button to define a parameter specification. For more information, see Adding or Changing a Variable or Parameter Specification.

     To specify a valid sweep value, ensure that:

     - The value is not a blank string or does not contain a whitespace
     - If contains a range, it has the start, step, and stop values separated by colons, commas, spaces, or a combination of these
     - If specified as an expression, it is written in correct SKILL syntax and contains balanced parenthesis and quotes
     - If contains a file path, it is enclosed in double quotes so that it is not evaluated

If you need to save some information about a global variable, you can add notes for it. For this, right-click the global variable on the Data View pane and click *Notes* to open the Add/Edit Notes form. Add notes in the *Notes* field and click *OK*. These notes are displayed in the tooltip for the variable and saved in the setup database.

© 2023 Cadence Design Systems, Inc.

For related information, see Adding Notes to a Test.

A variable is removed from the list of global variables only if it is not a part of any test in the maestro view. If any test contains the variable being deleted, the variable is removed and created again in the Global Variables list.

Note: Tags added to local variables are visible in the global variables and vice-versa.

Loading the Default Set of Global Variables

If you have a distinct set of default variables to be used for all the cellviews in each library or not associated with any library, you can define a SKILL function to load them into a new ADE Assembler setup.

When you open an ADE Assembler setup, the program loads the set of default variables associated with the same library as the setup (if it exists). After that, it loads the set of default variables not associated with any library (if it exists).

This pre-defined a set of variables appears in the *Global Variables* tree on the Date View assistant pane and the Variables tab of the Variables and Parameters assistant pane.

To create and load a set of global variables, do the following:

1. Create a file containing only one `axlSetDefaultVariables` SKILL function call for each set of global variables you want to define. (For example, `adeGlobals.il`.)
   - For global variables particular to a library, one call to the function for each library:

     ```
     axlSetDefaultVariables( '(global1 value1 global2 value2) "libName" )
     ```

     For example:

     ```
     axlSetDefaultVariables( '(_n_len 1u _sim_time 100n) "demoLib" )
     ```
   - For global variables not for a particular library, only one call to the function:

     ```
     axlSetDefaultVariables( '(global1 value1 global2 value2) )
     ```

     For example:

     ```
     axlSetDefaultVariables( '(_n_len 1u _sim_time 100n) )
     ```
2. You can load the file into the CIW using the `load` command.

   For example:

   ```
   load("adeGlobals.il")
   ```


© 2023 Cadence Design Systems, Inc.

Tip

You can put this load command in your `.cdsinit` file so that the program loads it every time you run the software.

If your file contains more than one call for a library or more than one call that does not name a library, the last call takes precedence such that the program loads only those global variables that you defined in each of the last of these calls.

© 2023 Cadence Design Systems, Inc.

Parameterizing Cellviews Defined in the Config View

Related blogs for this feature:

- Virtuosity: Sweeping Multiple Config Views
- Virtuosity: Sweeping Multiple DSPF Views in ADE

By default, the setup options defined in the ADE Assembler environment are applied to the cellview specified in the ADE Assembler tests. At times, you might need to run simulations for different views of a cell in the design using the same simulation setup. You can do this by sweeping the cellviews, including the parasitic views (`estimated` or `extracted` views), defined in the config view of that cell.

Prerequisites to sweep the config view of a cell:

- The design in the ADE Assembler test must be the config view
- More than one view should be present in the config view for the cell you want to sweep
- The Parasitic Mode on the Parasitic Mode toolbar is set to `No Parasitics/LDE`

To sweep the view for a cell, you need to define a global variable prefixed with `CONFIG`. The value of this global variable is a list of view names from the config view. These view names are swept during simulation. The following sections describe how to create config sweep variables or import config sweeps from the parasitic setup, and how to view these variables on the Run Preview tab or the simulation results:

- Creating Config Sweep Variables
- Importing Config Sweeps from the Parasitic Setup
- Using Config Sweep Variables in the Corners Setup Form
- Viewing Config Sweep Variables in Run Preview
- Viewing Config Sweeps in the Simulation Results

Creating Config Sweep Variables

To create config sweep variables, perform the following steps:

1. Right-click *Global Variables* on the Data View pane and choose *Add Config Sweeps*.

© 2023 Cadence Design Systems, Inc.

The Add/Edit Config Sweep form is displayed, as shown below.

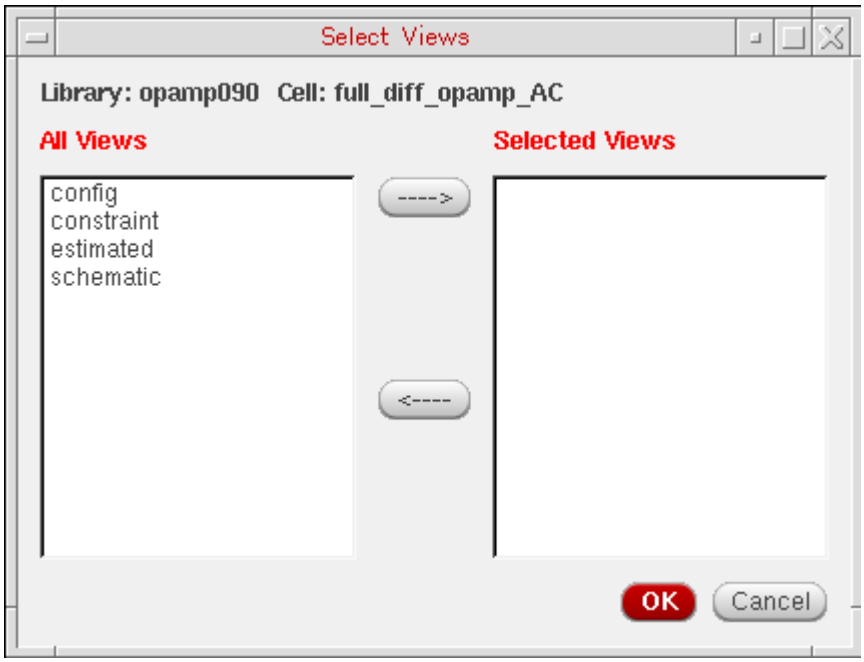
2. In the *Cell Name* drop-down list, select the cell name for which you need to sweep the config views.

   Note: The cell should not be a top-level cell in the design hierarchy.

3. In the *View Name List* field, enter a list of view names to be swept.

   Note: Ensure that the names of cellviews do not begin with a special character, such as `*`, `%`, `+` or `-`.

   Alternatively, click *Select* to open the Select Views form as shown below.



The list of available views for the cell is displayed in the *All Views* list. Select the required view names in this list and click ----> to add to the *Selected Views* list. Click *OK* to close the form. The view names are appended to list in the *View Name List* field.

© 2023 Cadence Design Systems, Inc.

Note: Ensure that the views exist for the given cell. Otherwise, while running simulations, ADE Assembler will ignore the view names that are not found.

4. Click *OK* to close the Add/Edit Config Sweep form.

   A config sweep variable is created and added to the *Global Variables* list on the Data View pane. The newly added config sweep variable is selected by default.

Important Points to Note

- You can also create CONFIG variables by adding a global variable using the *Click to add variable* command in the *Global Variables* list on the Data View pane. Specify a variable name and value in the Create Global Variable form. Ensure that the variable name is in the `CONFIG/<library-name>/<cell-name>` format and the value is a list of view names.
- The cells specified in the config sweep should not be a top-level cell in the design hierarchy.
- When you run a simulation with config sweeps, by default, ADE Assembler runs a check to ensure that all design variables are defined in the first cellview in the sweep list. If a design variable is missing in any other swept cellview, it would result in a simulation error. You can enable this check for all swept cellviews by setting the allowVariableCheckingDuringConfigSweep environment variable to `t`.
- When the checkInstanceBindings environment variable is set to `t`, ADE Assembler check for instance bindings in the Hierarchy Editor. If the instance or occurrence bindings in the config view conflict with the config sweeps, the tool displays a message prompt with a list of those instances and seeking confirmation whether the tool should clear the bindings in the config view.

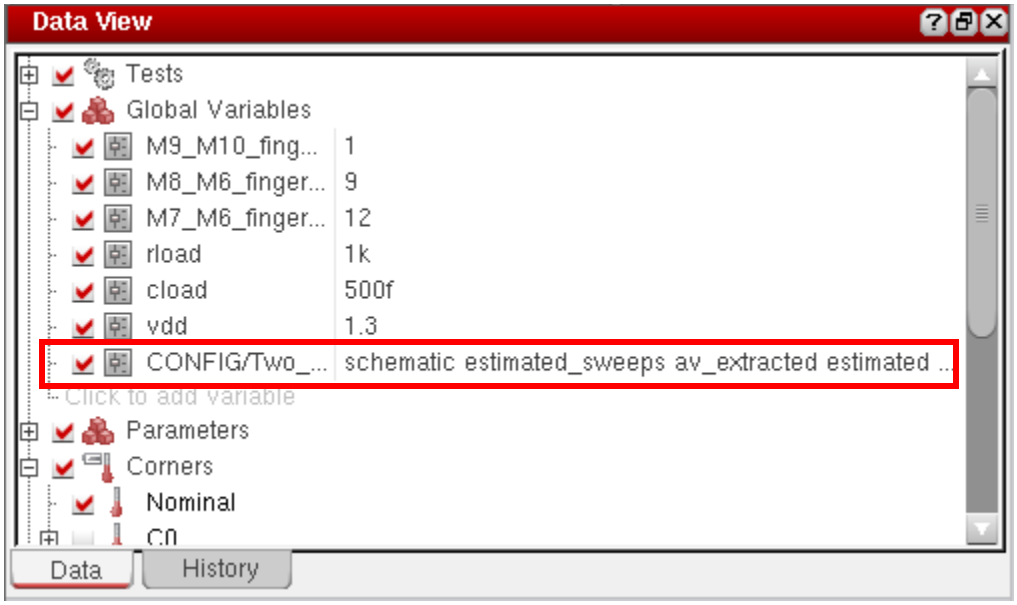
Importing Config Sweeps from the Parasitic Setup

If the parasitic setup already uses the estimated, netlist_layout, or extracted views, you can directly import those to create a config sweep variable. For this, right-click *Global Variables* and choose *Import Config Sweeps from Parasitic Setup*.

A config sweep variable is created and added to the existing list of global variables. The name of the variable is `CONFIG/<library-name>/<cell-name>`, where the library and cell are

© 2023 Cadence Design Systems, Inc.

the same as in the parasitic setup. The value of this variable contains a list of all the views referred in the parasitic setup, as shown below.



Using Config Sweep Variables in the Corners Setup Form

You can parameterize your cellview by using config sweep variables in the Corners Setup form. To create a config sweep variable in the corners, perform the following steps:

1. Click *click to add* in the *Design Variables* section of the Corners Setup form to display a list of available global and local design variables.
2. Choose a config variable from the list of variables.
3. In the row of the selected config variable, double-click in a column to make the cell editable.
4. Click the ellipses button to the right of the text box to open the *Add/Edit Config Sweeps* form.

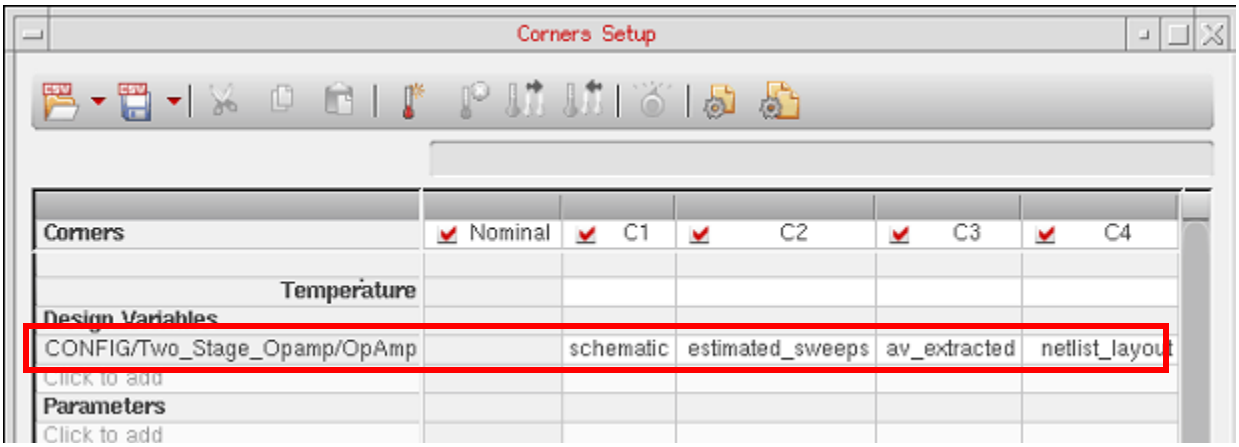
© 2023 Cadence Design Systems, Inc.

The Add/Edit Config Sweep form is displayed, as shown below.

Use this form to sweep the config views in the same way as done to create config sweep design variables in the Data View assistant.

5. Click *OK* to close the Add/Edit Config Sweep form.

   A config sweep variable is created and added to the *Design Variables* list in the Corners Setup form.



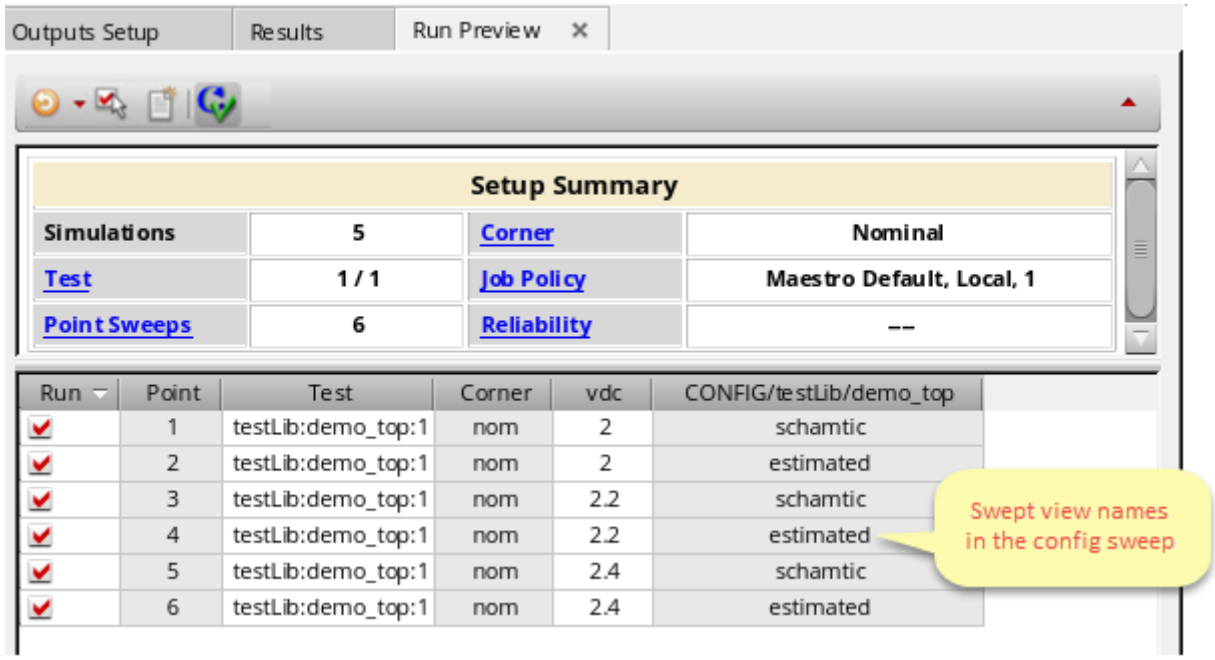
Related Topics

Creating Config Sweep Variables

© 2023 Cadence Design Systems, Inc.

Viewing Config Sweep Variables in Run Preview

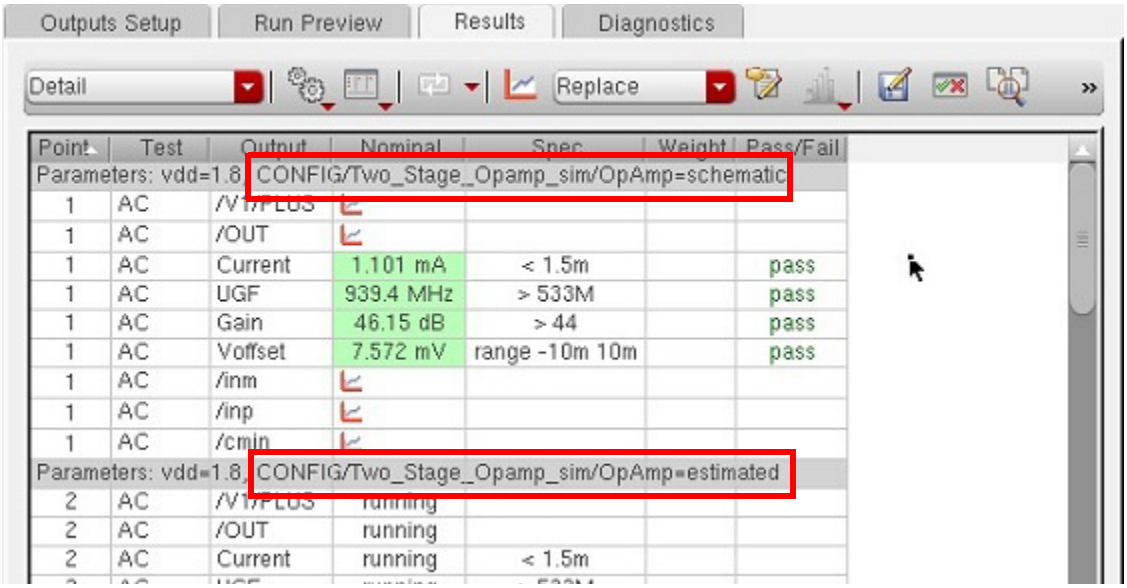
If config sweep variables are added to the setup, the *Run Preview* tab for the `Single Run, Sweeps and Corners` run mode shows the swept values in a column for the corresponding global variable, as shown below.



© 2023 Cadence Design Systems, Inc.

Viewing Config Sweeps in the Simulation Results

Similar to the other parameterized variables, config sweep variables are also shown in the gray colored rows on the *Results* tab, as shown below.



Important Points to Note

- You can use multiple config sweep variables in a maestro cellview. However, you need to ensure that there is no dependency between these variables.
- If multiple config sweep variables are specified, no specific order is maintained while modifying the config using these variables.
- You can create a parametric set containing more than one config sweep variables.
- Config sweeps are not considered if the config view contains instance bindings in Virtuoso Hierarchy Editor. You need to remove the bindings.
- When config sweep variables are used, ADE Assembler simulations do not consider the ignoreDesignChangesDuringRun and singleNetlistForAllPoints environment variables. If a simulation is already running and you make any changes in the design, the changes are considered for the simulation in progress. Also, a separate netlist is created for every data point.
- By default, the config information is saved in the netlist directory for every data point. This is because the moveConfigsToNetlistDir environment variable is set to `t`. If, for debugging purposes, you want to save the config information for every point in the

© 2023 Cadence Design Systems, Inc.

<library>/<cell>/ directory structure, you can set this variable to `nil`. This will show the generated config in Virtuoso Hierarchy Editor.

Importing Sweep Variables as Global Variables

If you have saved the sweep settings specified in the Parametric Analysis tool in the Virtuoso Analog Design Environment L (ADE L) in a parameters file (.il file), you can then import the sweep variables in the file as global variables. For more information about parametric analysis in ADE L, see the *Virtuoso Analog Design Environment L User Guide*.

To import sweep variables as global variables, do the following:

1. In the Data View pane or the Variables tab of the Variables and Parameters pane, right-click a global variable and choose *Import Sweep*.

   The Choose Parametric State File to Import form appears.

2. Select the file containing the sweep settings you saved in the Parametric Analysis tool in ADE L and click *Open*.

   The sweep variables in the file are displayed as global variables in the *Global Variables* tree on the Data View assistant pane and the Variables tab of the Variables and Parameters assistant pane.

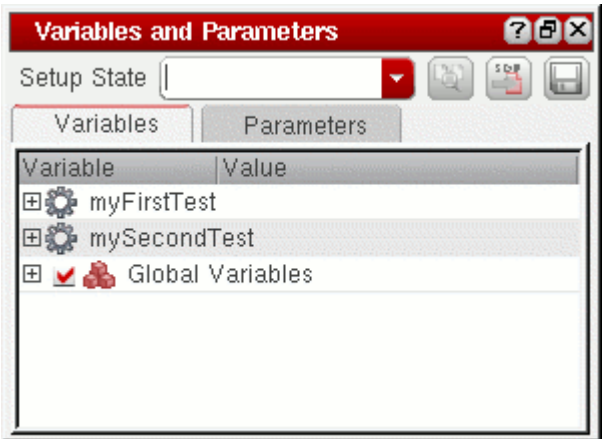
Saving Parameters and Global Variables to a Saved State

You can save the global variables and parameters from the Variables and Parameters assistant to an ADE state.

To save the variables and parameters, do the following:

© 2023 Cadence Design Systems, Inc.

1. After setting the variables and parameters and their values in the Variables and Parameters assistant, specify a name for the new ADE state in the *Setup State* field, as shown below.



2. Click *Save Variables and Parameters* ( ) to save the setup state.

Important

A saved stated to be used as a starting point should not have sweep values in variables and parameters.

Importing Global Variables from a Saved State

To import global variables and parameters along with their setup information from a saved state by using the Variables and Parameters assistant, do the following:

1. From the *Setup State* list, select name of the state from which you want to import the details.
2. Click *Load Variables and Parameters* ( ).

   All the variables and parameters are loaded from the given ADE state and displayed in their respective tabs.

   Note: All the global variables and parameters are overwritten.

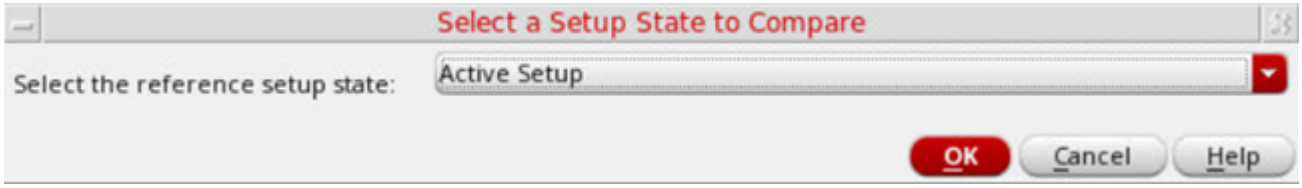
Comparing Variables and Parameters of Different Setup States

Before loading the global variables from a saved state, you can also choose to compare the variables and parameters with the currently active state or another setup state.

© 2023 Cadence Design Systems, Inc.

To do this,

1. Select the name of a state from the *Setup State* list and click *Compare Setup States*( ). The Select a Setup State to Compare form is displayed.

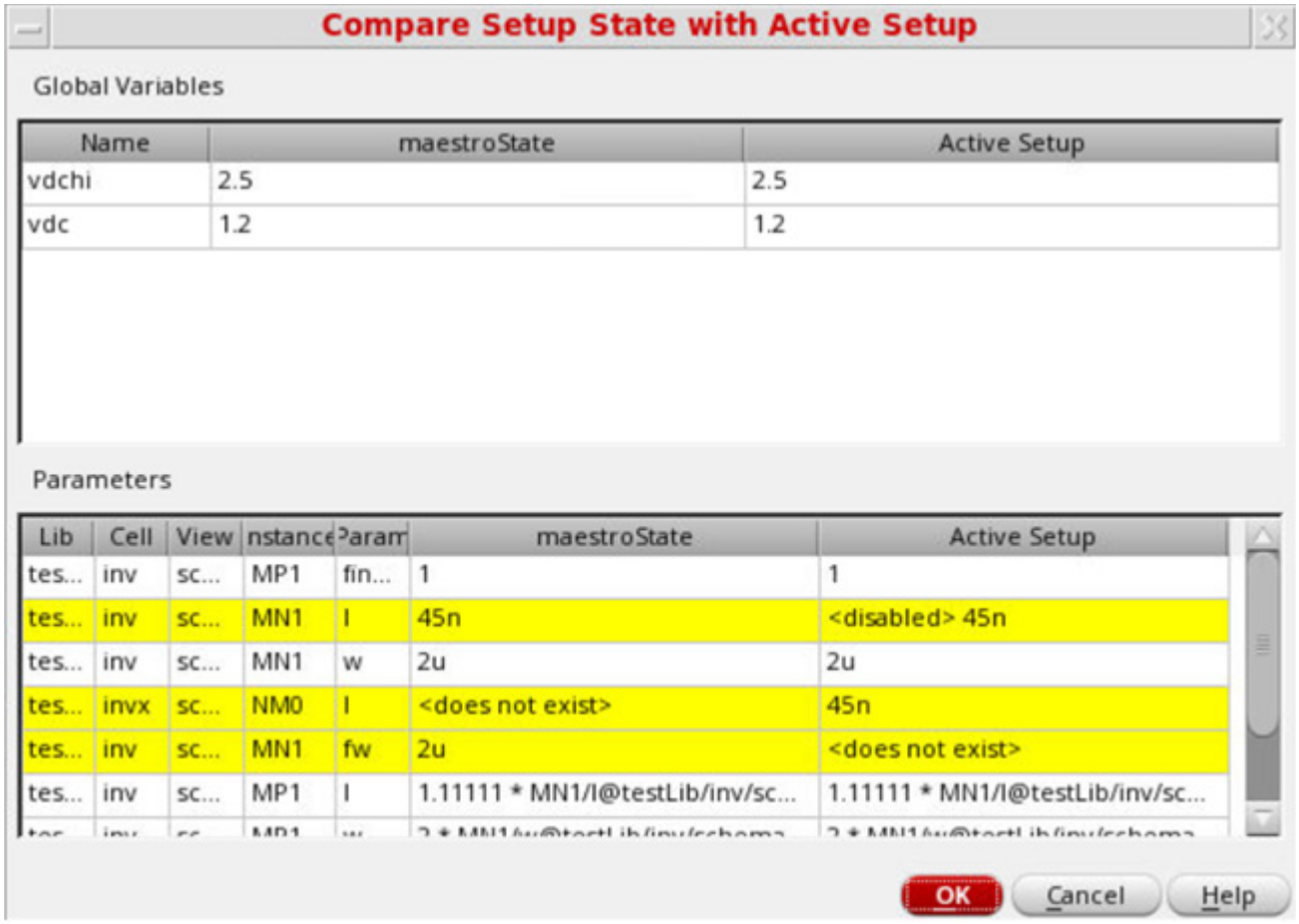


2. From the *Select the reference setup state* drop-down list, select the reference setup state.

The comparison is displayed in a form, where the form name gets customized according to the setup states being compared.

© 2023 Cadence Design Systems, Inc.

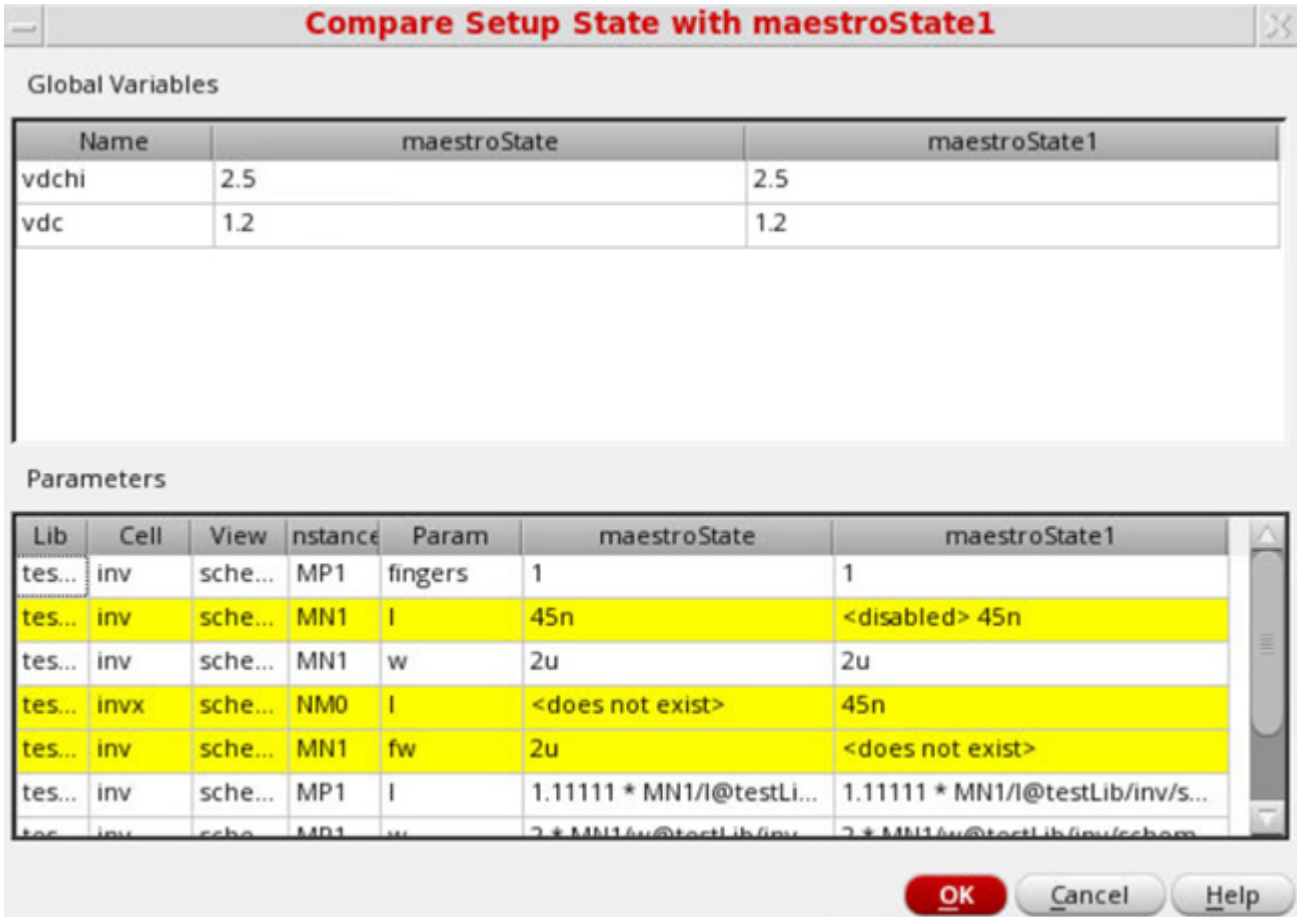
For example, on comparing the setup state with the currently active state, the Compare Setup State with Active Setup form is displayed as shown below.



In case a comparison is made between two different setup states, the form name gets customized accordingly.

© 2023 Cadence Design Systems, Inc.

The following form shows the comparison between the setup state and a state `maestroState1`.



The variables and parameters that have different values in the two setups are highlighted in yellow.

Click *OK* to close the form.

You can also import the global variables and parameters by using the **Loading State** form. For more details, see Loading State Information.

© 2023 Cadence Design Systems, Inc.

Exporting Global Variables to a CSV File

You can export the global variables from the ADE Assembler setup to a CSV file. You can later import them as global variables in the same or any other ADE Assembler setup, or as design variables in ADE Explorer.

Video

You can view a video demonstration for this feature at Export and Import Variables in the ADE Assembler.

To export the global variables defined in the active setup of ADE Assembler, do the following:

1. In the *Data* tab of the Data View assistant, right-click *Global Variables*.
2. Choose *Export to CSV*.

   The Export Variables form is displayed.
3. In this form, browse to a directory where you want to save the variables in a CSV file.
4. Specify the name of the CSV file in the *File Name* field.
5. Click *Save* to close the form and export the variables.

All the global variables, their values, notes, and the enabled or disabled status currently defined in the active setup are saved in the specified CSV file in a standard format, as shown in the example below.

```
Type,Name,Value,Enabled,Notes
variable,rload,1k,enabled,
variable,cload,100f 102f,enabled,
variable,vdd,1.3 1.7 2.0,disabled,
```

Note: If parametric sets were created for the variables, that information is also saved in the CSV file.

For more details, refer to:

- Importing Global Variables from a CSV File
- Exporting Variables to a CSV File

© 2023 Cadence Design Systems, Inc.

Importing Global Variables from a CSV File

If you have global variables or design variables saved in a CSV file, you can import them as global variables in the active setup.

To import the variables from a CSV file, do the following:

1. In the *Data* tab of the Data View assistant, right-click *Global Variables*.
2. Choose *Import from CSV*.

   The Import Variables form is displayed.
3. In the Import Variables form, browse to a directory and select the CSV file.
4. Click *Open* to close the form and import the variables.

The variables defined in the CSV file are added to the *Global Variables* list.

Note the following points:

- If a variable already exists with the same name as that of a variable being imported, it is overwritten by the imported variable.
- The enabled or disabled status of the imported variable is also imported, and set accordingly.
- If the information about parametric sets is available in the CSV file, the parametric sets are also created after importing the variables.

Enabling and Disabling Global Variables for All Tests

You can enable or disable global variables before starting your simulation. This allows you to specify the set of global variables you want to be swept for a particular simulation.

To enable a global variable for all tests,

➤ In the *Global Variables* tree on the Data View pane or the Variables tab of the Variables and Parameters pane, select the check box to the left of the variable.

Alternatively, right-click the variable and choose *Enable/Disable* from the context-sensitive menu.

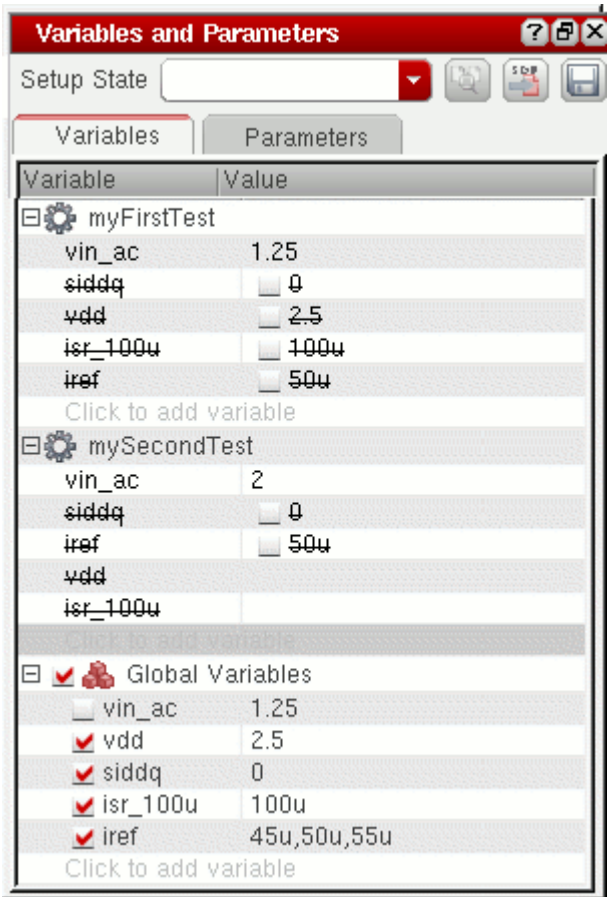
To disable a global variable for all tests,

© 2023 Cadence Design Systems, Inc.

➤ In the *Global Variables* tree on the Data View pane or the Variables tab of the Variables and Parameters pane, clear the check box to the left of the variable.

Alternatively, right-click the variable and choose *Enable/Disable* from the context-sensitive menu.

When a global variable is disabled, its value will not be used for simulation but its definition is retained in the *Global Variables* tree on the Data View pane and the Variables tab of the Variables and Parameters pane. The value of the design variable (specified for a test) that has the same name as the global variable will be used for the simulation. The ~~strikethrough~~ no longer appears for the design variable value in the test tree. For example, in the following figure, the value of design variable `vin_ac` specified for the tests named `myFirstTest` and `mySecondTest` will be used because the global variable `vin_ac` is disabled.



To toggle the enabling or disabling of more than one global variable,

© 2023 Cadence Design Systems, Inc.

➤ In the *Global Variables* tree on the Data View pane or the Variables tab of the Variables and Parameters pane, select the variables, right-click and choose *Enable/Disable* from the context-sensitive menu.

The enabled variables are disabled and the disabled variables are enabled.

When all global variables are disabled, global variable values will not be used for simulation but their definition is retained in the *Global Variables* tree on the Data View pane and the Variables tab of the Variables and Parameters pane. The value of design variables (specified for tests) that have the same name as the global variable will be used for the simulation.

Disabling Global Variables for Specific Tests

By default, global variables are swept over all the tests that are enabled in the Data View pane. You can disable a global variable for specific tests if you do not want to sweep the global variable over those tests. If a global variable is disabled for a test, that is, it overrides the global variable, you can specify a single and sweep values to be used only for that test.

You can disable a design variable for a test in any of the following ways:

- Using the Variables and Parameters assistant
- Using the Data View assistant

Video

You can view a video demonstration for this feature in ADE Assembler at Overriding Global Variable Values With Design Variables in the ADE Assembler.

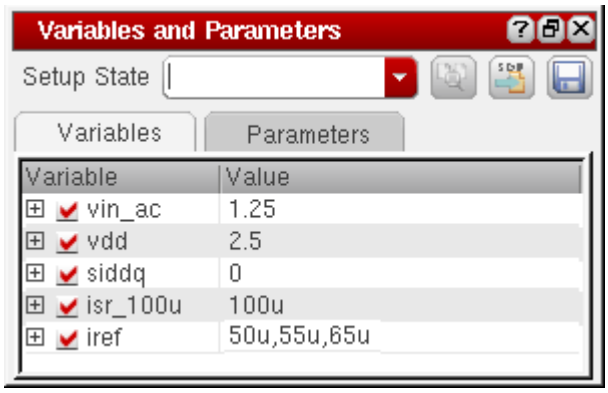
Disabling a Global Variable for a Test using the Variables and Parameters Assistant

To disable global variables using the Variables and Parameters assistant pane.

1. Click the Variables tab of the Variables and Parameters pane.

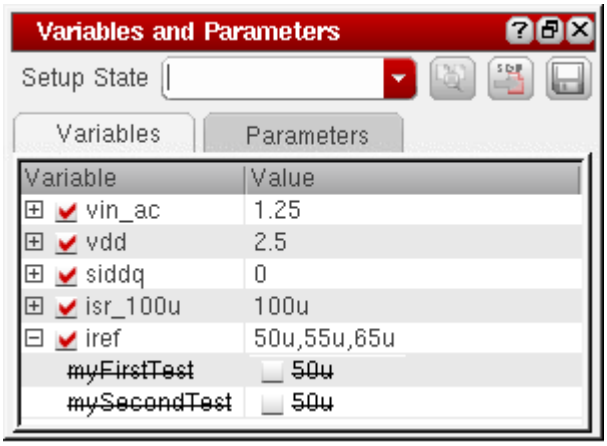
© 2023 Cadence Design Systems, Inc.

The *Variables* tab displays the list of global variables as shown below:



2. Click the plus sign to the left of the global variable name to view the names of tests over which the global variable will be swept during simulation.

   For example, the following figure indicates that the values `45u`, `50u` and `55u` specified for global variable `iref` will be swept over the tests named `myFirstTest and mySecondTest`. In this example, the `myFirstTest` and `mySecondTest` tests have a design variable named `iref` with the value `50u`. The test names and values are displayed with a ~~strikethrough~~ because the global variable `iref` overrides the design variable `iref`.

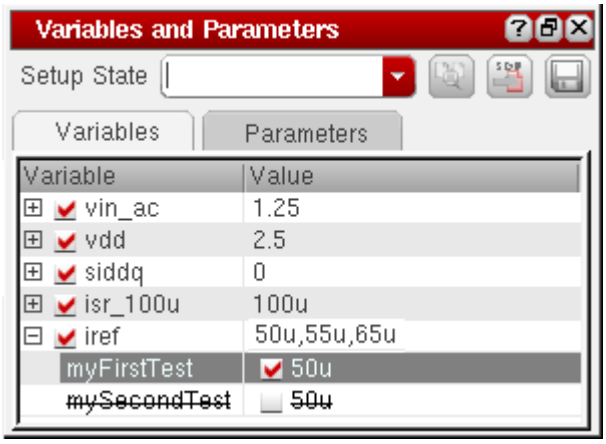


3. In the *Value* column, select the check box next to the name of the test for which you want to disable the global variable.

   The global variable is disabled for the test. For example, in the following figure, the global variable `iref` is disabled for the test named `myFirstTest`. Because of this, the value

© 2023 Cadence Design Systems, Inc.

`50u` will be used for the test during simulation. The global variable will be swept only for the test named `mySecondTest`.



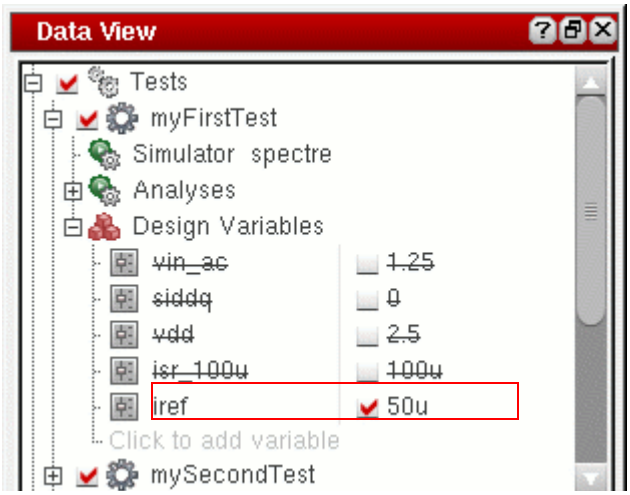
Also see:

- Hiding Overridden Design Variables for a Test on page 167
- Updating Global Variable Values with Design Variable Values on page 274

© 2023 Cadence Design Systems, Inc.

Disabling a Global Variable for a Test using the Data View Assistant

You can disable a global variable for a test using the Data View assistant also, as shown in the following figure:

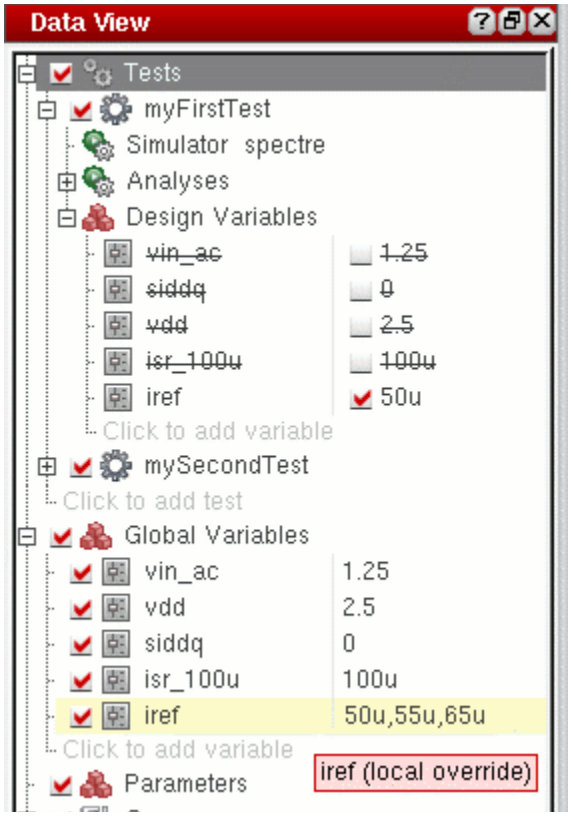


The global variable `iref` is using the value specified for the test `myFirstTest` instead of using the sweep points specified for the global variable.

The settings made using the Data View assistant are also reflected in the Variables and Parameters assistant and vice-versa.

© 2023 Cadence Design Systems, Inc.

In the Data View pane, if you place the mouse pointer on the global variable that is disabled for a test, a tooltip is displayed containing the text `(local override)`, as shown in the following figure:



Also see:

- Hiding Overridden Design Variables for a Test
- Viewing Simulation Results When Design Variables Use Swept Values

© 2023 Cadence Design Systems, Inc.

Updating Global Variable Values with Design Variable Values

To update the value of a global variable with that of a corresponding design variable, do the following:

➥ In an expanded test tree on the Data View pane or on the Variables tab of the Variables and Parameters pane, right-click a design variable and choose *Create/Update Global*.

The value of the global variable that has the same name as the design variable is updated.

Updating Design Variable Values with Global Variable Values

To update the test design variables with the value of the corresponding global variables, do the following:

➥ In the *Global Variables* tree on the Data View pane or the Variables tab of the Variables and Parameters pane, right-click a variable, choose *Backannotate to Test* and do one of the following:

Choose	To
All	Update the values of design variables that have the same name as the global variable in all the tests.
Testname	Update the value of a design variable that has the same name as the global variable in the selected test. **Note:** If a design variable with the same name does not exist for the test, the design variable is automatically added for the test.

Note: You cannot update a design variable whose corresponding global variable has a sweep value.

© 2023 Cadence Design Systems, Inc.

Using calcVal for Calibration

For calibration purposes, you often need to use the output values from a calibration test in the conditions of the test being verified. The multi-test environment of ADE Assembler allows you perform calibration. One of the ways to create calibration conditions or expressions is to use the `calcVal` function. You can also use pre-run scripts and run plans for calibration.

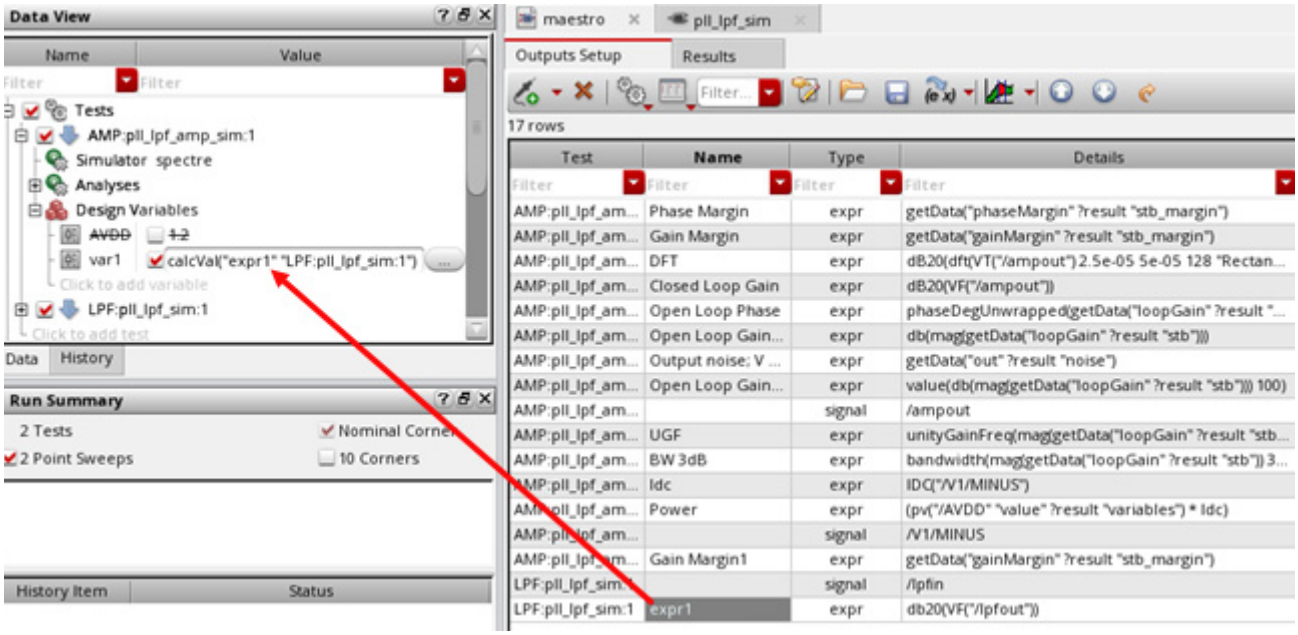
This section describes how to use `calcVal` for calibration when you need to pass a result value across tests, corners, or history results. You can use different arguments of the `calcVal` function to pass result values. For example, to use the result of the `nominal` corner in the calculation of a design variable used in other corners, you can use the `?cornerName` argument of `calcVal`.

You can use `calcVal` in design variables, global variables, outputs, corner setup, or OCEAN scripts.

Using calcVal in Design Variables

Set the value of design variable `var1` as `calcVal("<outputName>" "<testName>")`.

For this, create a design variable for a test as `var1`, and type in the calcVal expression. Alternatively, you can drag and drop an output from the Outputs Setup pane to the value of `var1`. A `calcVal` expression is automatically created in the value of `var1`. If required, you can further add more arguments to the expression.



© 2023 Cadence Design Systems, Inc.

Note that the calcVal expression is being added to the design variable for test `AMP`, whereas the output used in calcVal is for test `LPF`. To perform this calculation, ADE Assembler first runs simulation for test `LPF` and then calculates the value for var1 in test `AMP`.

You can also use the *calcVal lint check* button in the *Run Preview* tab to run pre-simulation checks on `calcVal` expressions. You can then view the resulting output messages in the CIW.

Using calcVal in Design Variables with Corners in Setup

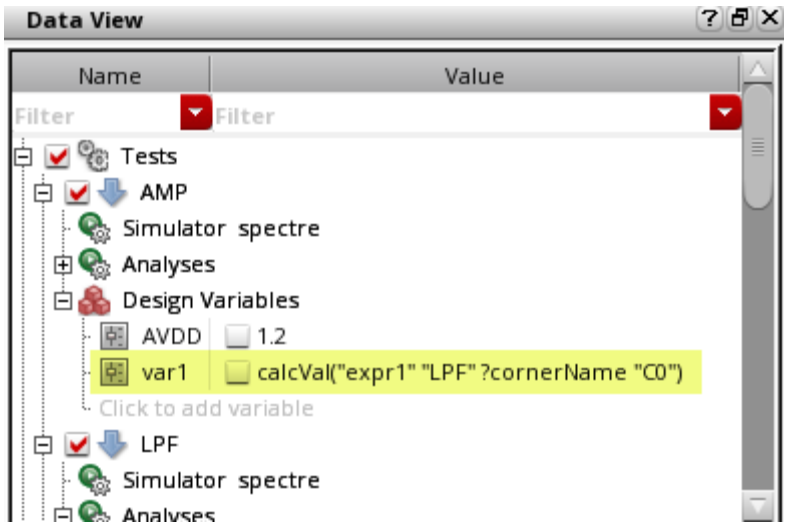
When the setup is using corners that are enabled for all tests, in the example shown above, the output value for each corner run for test `LPF` is used for the corresponding corner run for test `AMP`. That is the calibrated value for corners are used as shown below.

expr1 calculated for `LPF:C0` -> used for var1 calculated for `AMP:C0`

expr1 calculated for `LPF:C1` -> used for var1 calculated for `AMP:C1`

expr1 calculated for `LPF:C2` -> used for var1 calculated for `AMP:C2`

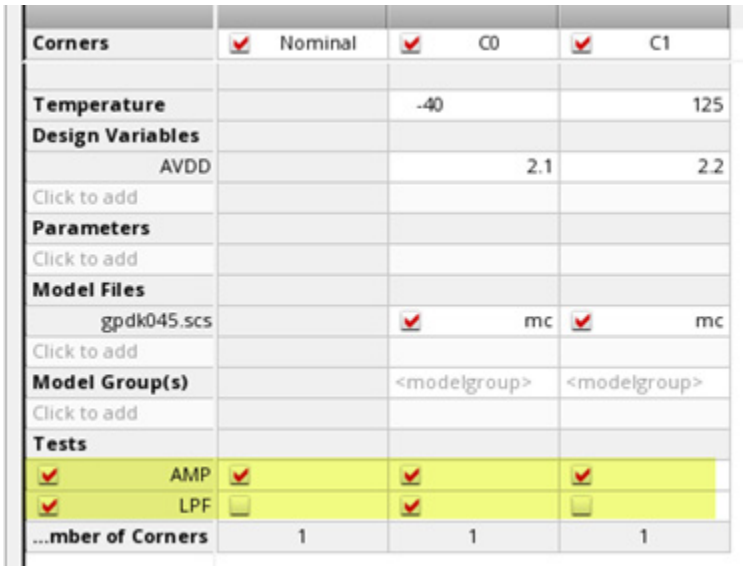
However, if you need to calibrate only one corner `C0` run for test `LPF` and use its output in all corners run for test `AMP`, you can use the `?cornerName` argument in the `calcVal` expression, as shown below.



You also need to disable corners `Nominal` and `C1` for test `LPF` in the Corners Setup, as

© 2023 Cadence Design Systems, Inc.

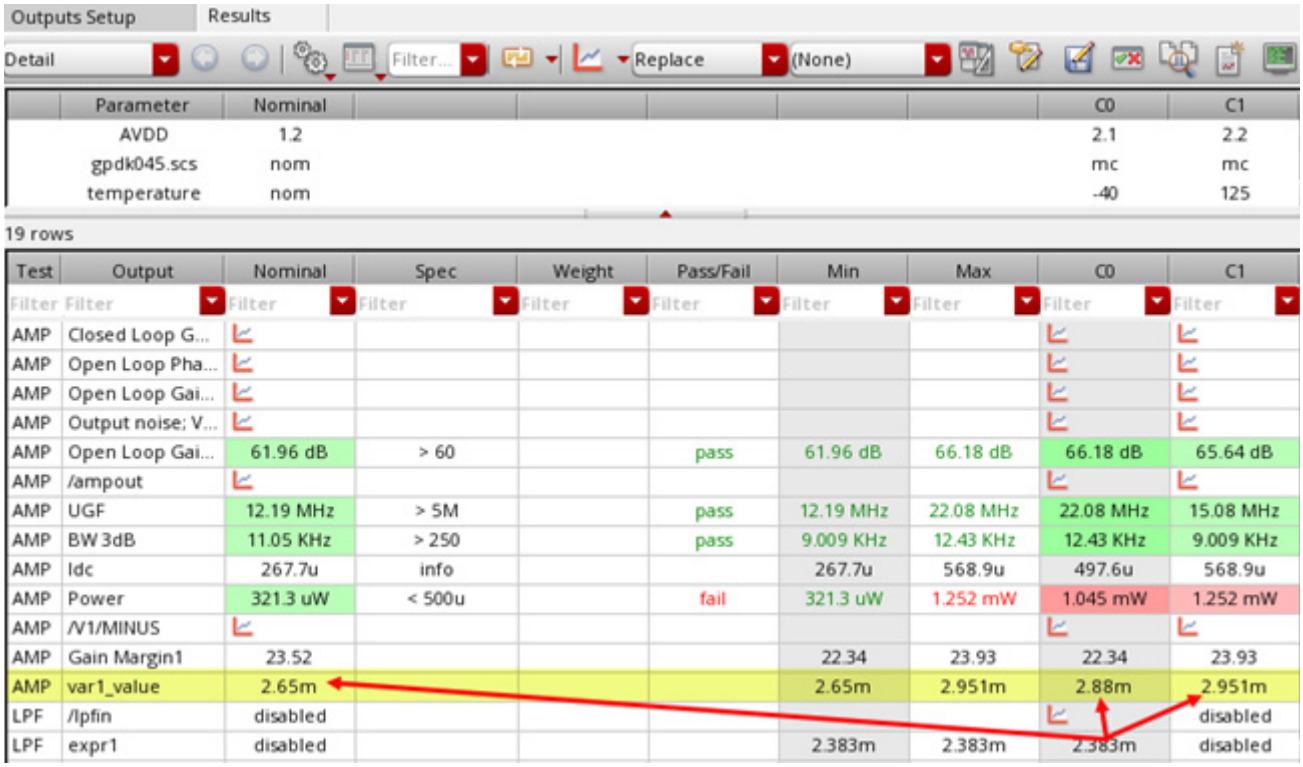
shown below.



Corners	Nominal	C0	C1
Temperature		-40	125
Design Variables			
AVDD		2.1	2.2
Click to add			
Parameters			
Click to add			
Model Files			
gpdk045.scs		mc	mc
Click to add			
Model Group(s)		<modelgroup>	<modelgroup>
Click to add			
Tests			
AMP			
LPF			
...mber of Corners	1	1	1

© 2023 Cadence Design Systems, Inc.

The calibration result of corner `C0` is used for the `Nominal` and `C1` run for test `AMP`, as shown below.



Value of **expr1** from the **C0** run for test **LPF** is used in the **var1_value** calculation for **Nominal** and **C1** runs for test **AMP**

For more details and examples on running calibrations using these methods, refer to the Techniques for Simulating Calibrated Circuits with Virtuoso ADE Assembler rapid adoption kit on Cadence Support portal. For syntax and description of the `calcVal` function, refer to *ADE SKILL Reference Guide*.

© 2023 Cadence Design Systems, Inc.

5

Working with Constraints

ADE Assembler supports the matched parameter and correlation constraints you specify in Virtuoso Constraint Manager.

- ADE Assembler creates parameters for the matched parameter constraints specified in Constraint Manager. These parameters are displayed on the Data View and the Parameters tab on the Variables and Parameters assistant panes. For more information about working with parameters created for matched parameter constraints, see Working with Parameters Created for Matched Parameter Constraints on page 281.
- Correlation constraints are used for Monte Carlo analysis. Correlation constraints are not displayed on the Data View or the Variables tab of the Variables and Parameters assistant panes—they are written to the netlist when you run a Monte Carlo analysis. For more information about Monte Carlo analysis, see Chapter 11, “Monte Carlo Analysis.”

For more information, see the following topics:

- Adding, Modifying, and Deleting Constraints on page 280
- Working with Parameters Created for Matched Parameter Constraints on page 281

© 2023 Cadence Design Systems, Inc.

Adding, Modifying, and Deleting Constraints

To add, modify or delete matched parameter and correlation constraints in Constraint Manager, do the following:

1. On the Data View assistant pane, right-click a test and choose *Open Design in Tab*.

   The schematic design associated with the test appears on a new tab in your session window.

2. Choose *Window – Assistants – Constraint Manager*.

   The Constraint Manager pane appears.

3. Add, modify or delete matched parameter and correlation constraints. For more information about using Constraint Manager, see the *Virtuoso Unified Custom Constraints User Guide*.

4. Click the *Save* button in Constraint Manager.

5. Click *Check and Save* in Virtuoso Schematic Editor.

© 2023 Cadence Design Systems, Inc.

Working with Parameters Created for Matched Parameter Constraints

To work with the parameters, you need to import the matched parameters constraints from Constraint Manager to ADE Assembler. These parameters are displayed in the *Parameters* tree on the Data View pane and the Variables tab of the Variables and Parameters assistant pane.

- In the *Parameters* tab on the Variables and Parameters assistant pane, you cannot modify or delete the matched parameter constraints that are created by Constraint Manager.
- If you make changes to matched parameter constraints in Constraint Manager while ADE Assembler is running, the changes are automatically reflected in ADE Assembler. For example, if you modify a matched parameter constraint in Constraint Manager while ADE Assembler is running, the corresponding parameters in ADE Assembler are also automatically modified.

Importing Matched Parameter Constraints to ADE Assembler

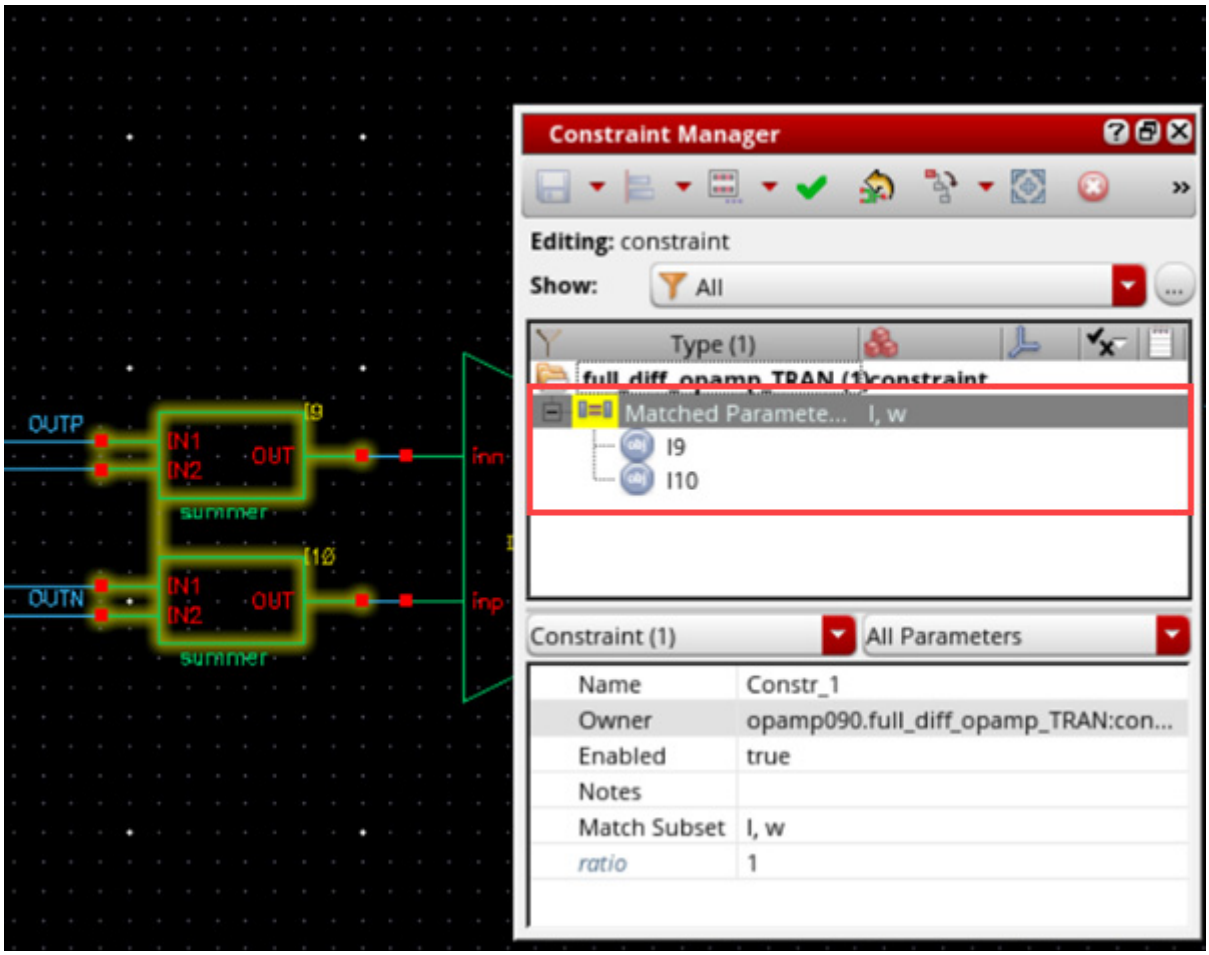
To import constraints from Constraint Manager to ADE Assembler:

1. In the Data View assistant, right-click the *Parameters* tree.
2. Choose *Import Constraints.*

   The parameters with matched constraints listed in Constraint Manager are imported to ADE Assembler and listed in the *Parameters* tree.

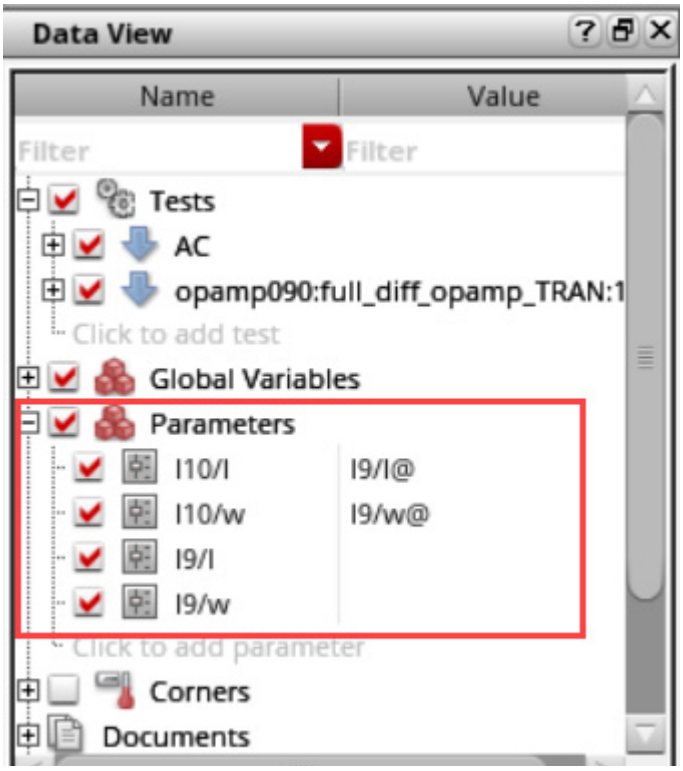
© 2023 Cadence Design Systems, Inc.

Consider the following example design with a matched parameter constraint defined for two devices I9 and I10.



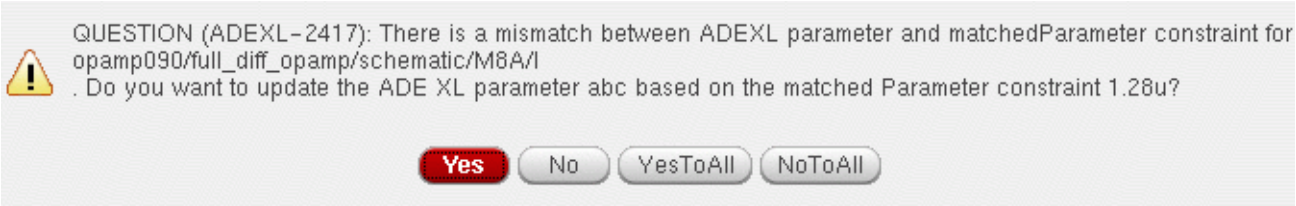
© 2023 Cadence Design Systems, Inc.

The Import Constraints command adds these constraints to the Data View assistant, as shown below.



Resolving Mismatch Between Matched Parameter Constraints and their Corresponding Parameters

When you launch ADE Assembler, or restore a history item, ADE Assembler checks if there is any mismatch between the matched parameter constraints in Constraint Manager and their corresponding parameters in ADE Assembler. The following message is displayed for each mismatch:



Do one of the following:

© 2023 Cadence Design Systems, Inc.

- Click *Yes* to update the parameter based on the corresponding matched parameter constraint.
- Click *No* to ignore the mismatch and mark the corresponding matched parameter constraint as *impossible* in Constraint Manager.
- Click *Yes to All* to update all mismatched parameters based on their corresponding matched parameter constraints.
- Click *No to All* to ignore all the instances where there is a mismatch and mark the corresponding matched parameter constraints as *impossible* in Constraint Manager.

© 2023 Cadence Design Systems, Inc.

6

Working with Corners

A corner is a combination of variables or process models that define a scenario in which you want to measure the performance of your design. In ADE Assembler, you can create corners using the Corners Setup form where you can vary the values for temperature, parameters, design variables, and model files from any test. You can then run the corners for one or more tests and measure how the tests perform in these varying conditions. If the corner settings are already available in pre-defined files, you can also load and use them to create corners.

This chapter describes the Corners Setup form and the procedures to create and use corners in ADE Assembler.

See the following topics for more details:

- The Corners Setup Form
- Creating Corners
- Filtering Corners
- Running Simulation with Corners
- Plotting Across Corners

Video

You can view the following video demonstrations related to this feature in ADE Assembler:

- Creating Corners and Overriding Specifications in the ADE Assembler
- Corner Filtering in ADE Assembler & Explorer

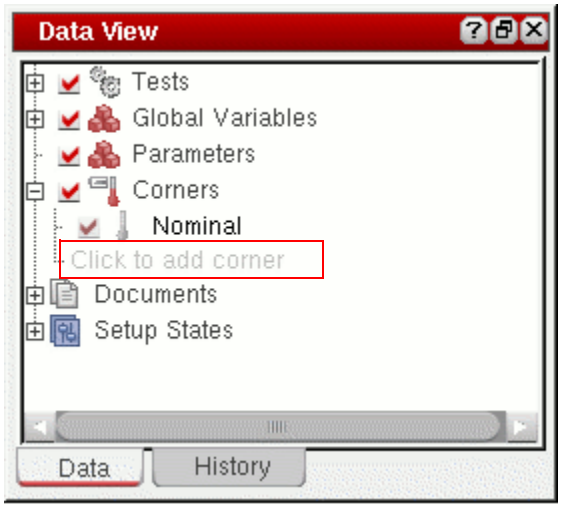
© 2023 Cadence Design Systems, Inc.

The Corners Setup Form

You can add or remove corners or modify corner settings by using the Corners Setup form.

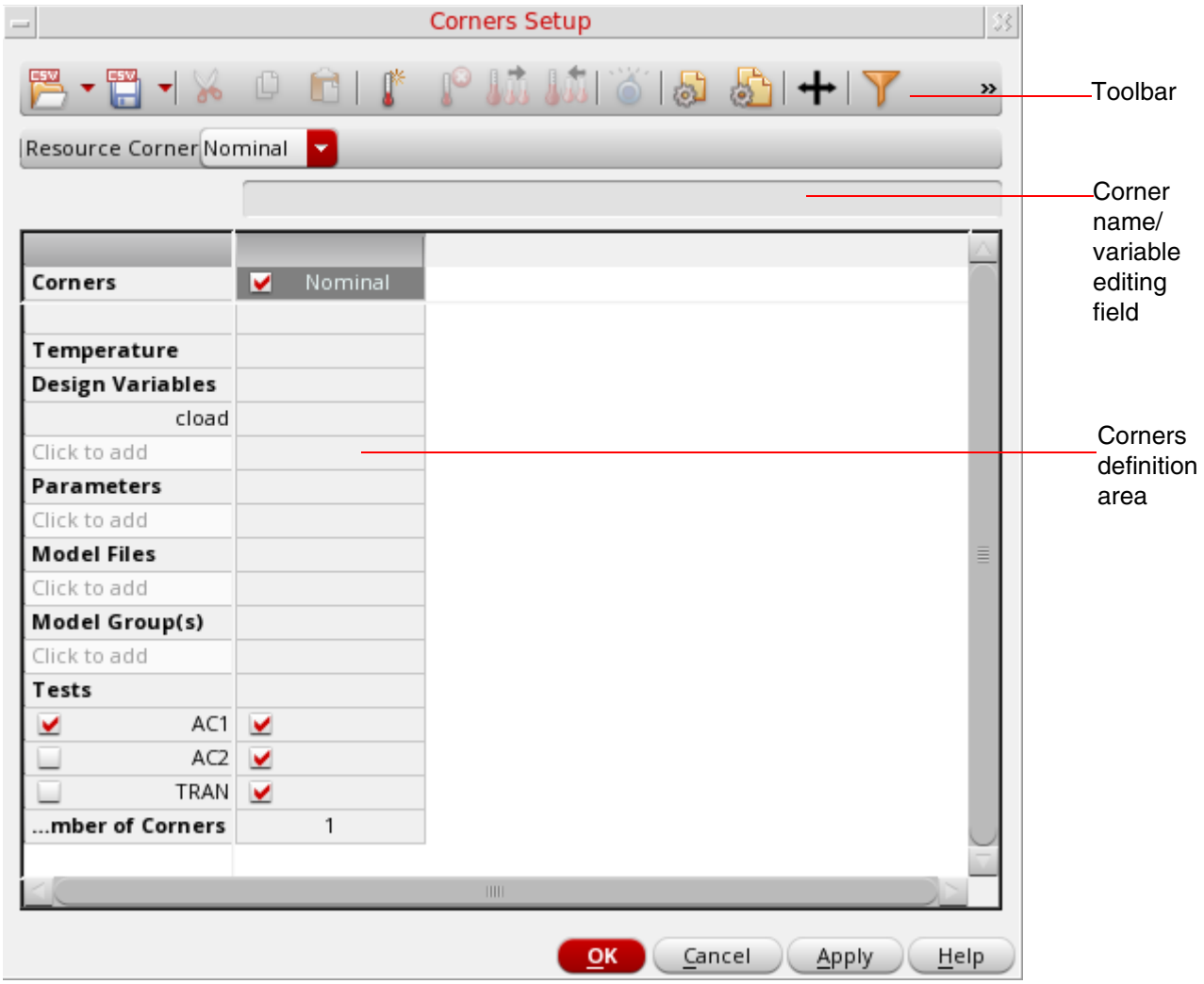
To open the Corners Setup form, do one of the following:

- Choose *Create – Corners*.
- On the Create toolbar, click .
- In the Data View pane, expand *Corners* and click where it says *Click to add corner*.



© 2023 Cadence Design Systems, Inc.

The Corners Setup form is displayed.



Note: Each corner is added as a new column on this form. By default, the table contains a default corner, named `Nominal`. This is the nominal corner and runs without varying the corners variables.

For more information on the Corner Setup form, refer to the *Virtuoso Analog Design Environment Explorer User Guide*.

Note: Read a blog related to this at Virtuosity: Handy UI Enhancements in ADE Assembler & ADE Explorer.

Useful Environment Variables:

- headerAlignmentSide

© 2023 Cadence Design Systems, Inc.

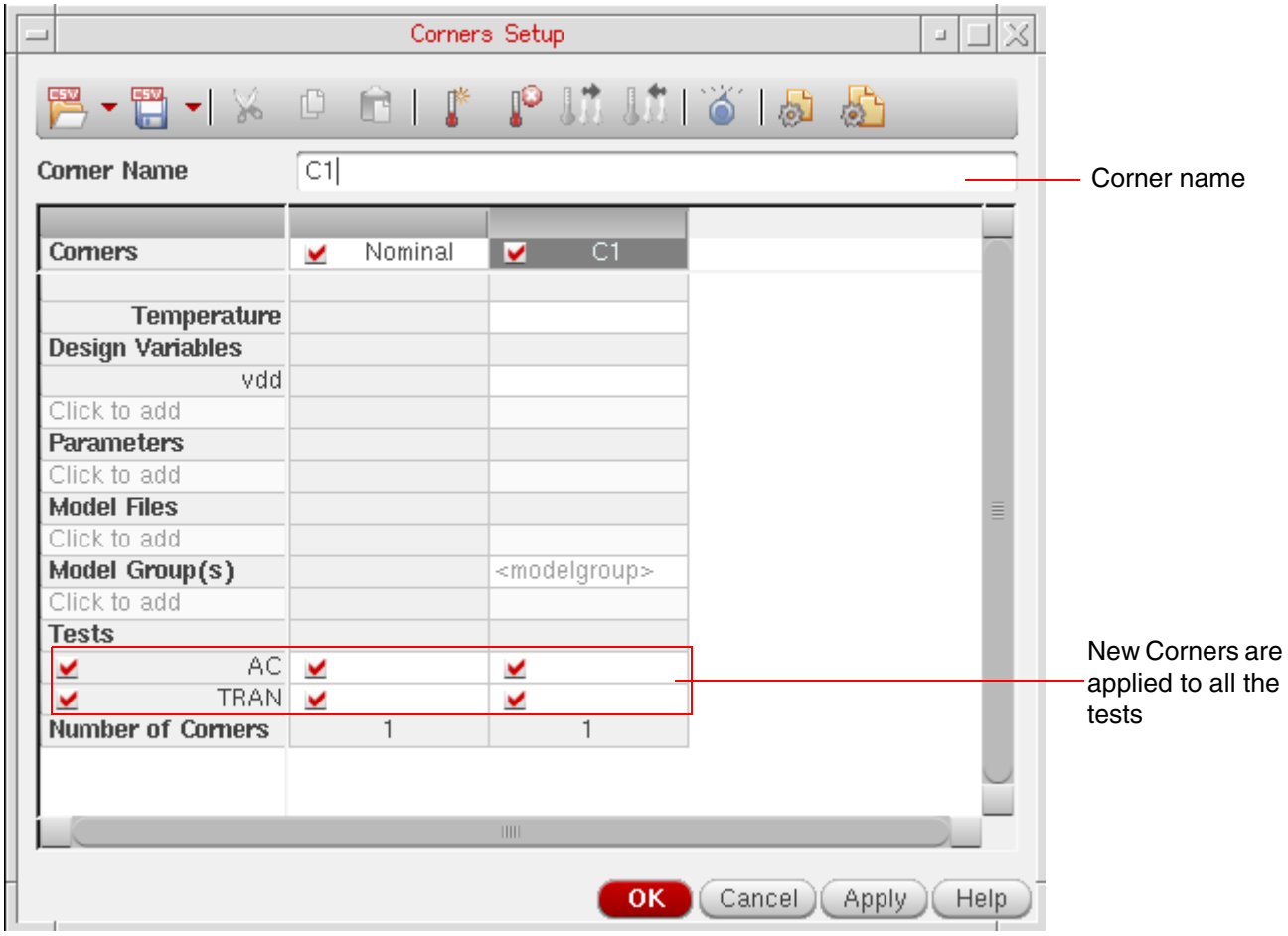
- headerTruncationWidth

Creating Corners

You can create a corner to define the conditions under which you want to check a design of your tests.

To create a new corner, in the Corners Setup form, click *Add new corner* on the toolbar.

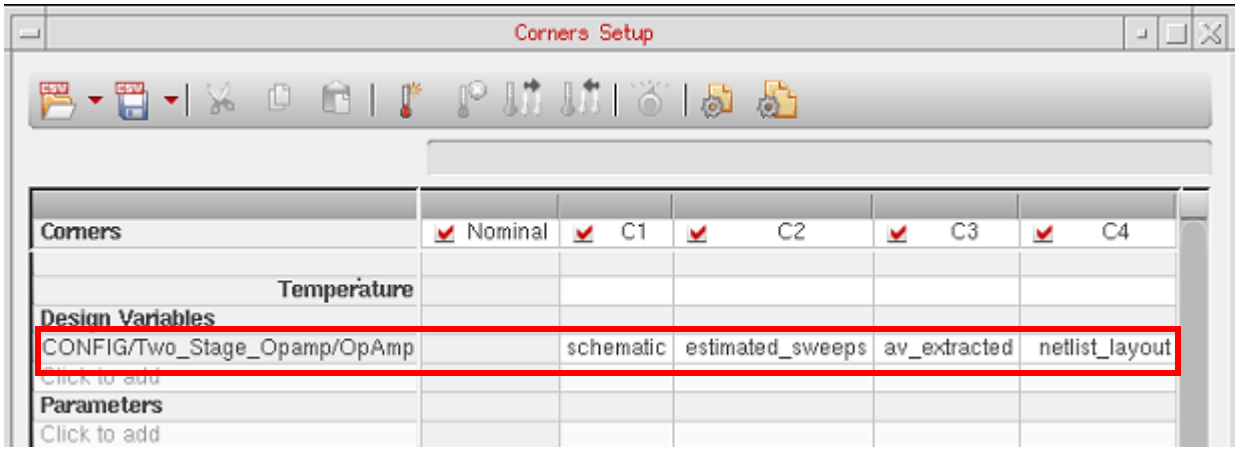
A new column is added in the Corners Setup form, as shown in the following figure.



By default, the corners are named sequentially, prefixed with `C`, such as `C1`, `C2`, and so on. You can change the names, if required.

© 2023 Cadence Design Systems, Inc.

After adding a new corner, specify values for the variables that define the corner conditions. You can also use config sweep variables to define corners, as shown below.



Note: In a corner column, all the text fields, for example, Temperature or a design variable, are by default edited in the append mode. When you click a field and type a value, it is appended to the already existing value, if any. To replace the existing value, select and delete it, and enter a new value.

Note: If you have saved a set of default corners in a `.sdb` file that you want to be used for a maestro view, set the defaultCorners environment variable to that file path. If no other corners are defined for the maestro view, corner definitions from the given file are loaded into the corners setup. Currently, you can load the default corners only from `.sdb` files. This feature is not supported for `.csv` files.

You can specify one or more of the following settings for a particular corner:

- Specify temperature
- Add design variables
- Add parameters
- Add model files
- Specify sections of model files
- Add model groups

For information on the tasks listed above, refer to the *Chapter 6, Working with Corners* in the Virtuoso ADE Explorer User Guide.

© 2023 Cadence Design Systems, Inc.

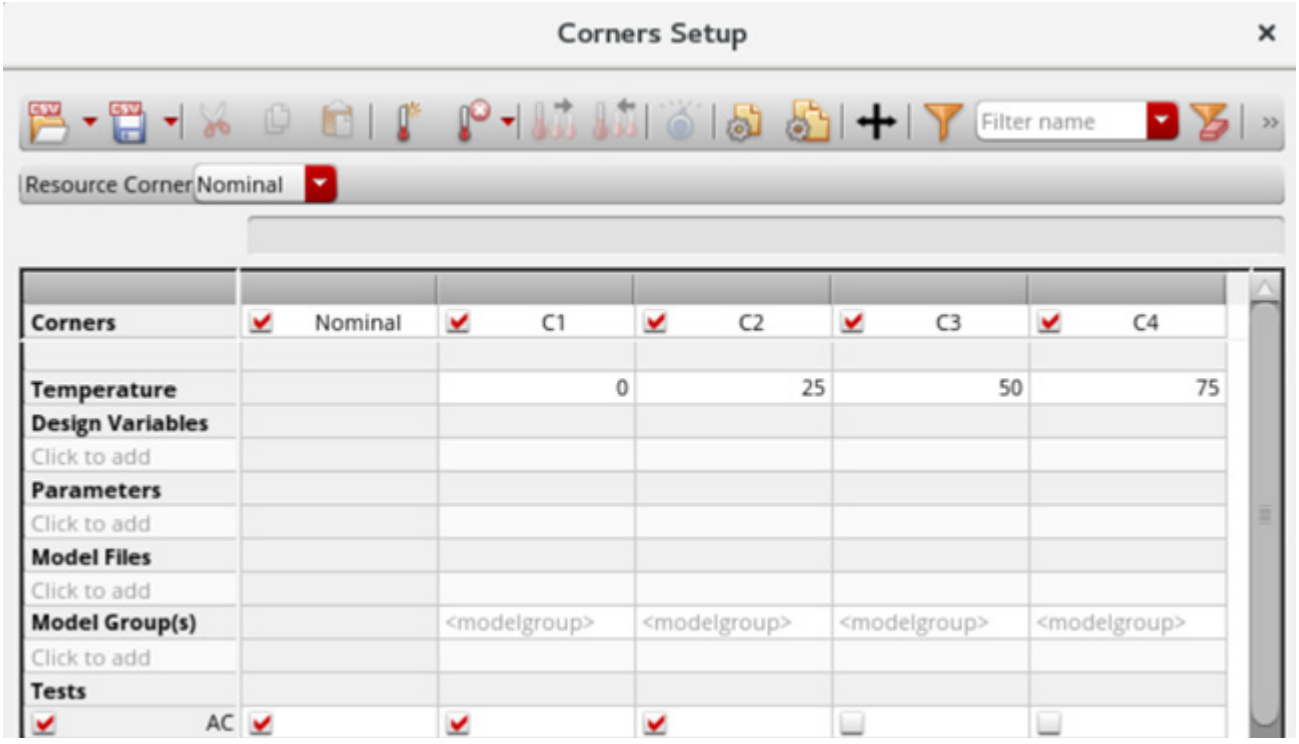
Important

In addition to these corners, you can also create statistical corners based on the results for a corner in a Monte Carlo sample. To know more about what statistical corners are and how to create these corners from the Monte Carlo results, refer to Managing Monte Carlo Results in the Yield View.

Disabling or Enabling Corners

You can use the *Corners* section in the Data View assistant to enable or disable corners. Disabled corners are automatically hidden in the results. To view these corners in the *Results* tab, select the *Disabled Corners* option in the *Configure what is shown in the table* drop-down list. The *Results* tab then displays the text `disabled` in the corresponding column.

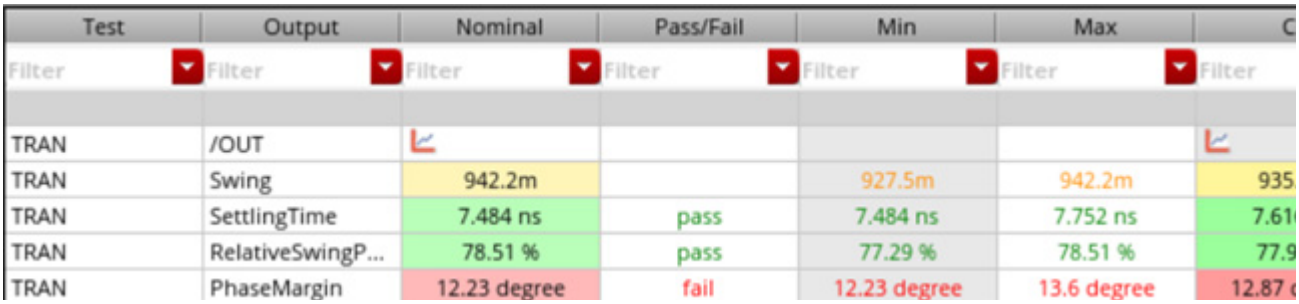
For example, let us consider the following setup.



In this setup, the corners `C1` and `C2` are enabled for test `AC`, and the corners `C3` and `C4` are enabled for test `TRAN`. If you disable the test `AC` through the *Select which tests are*

© 2023 Cadence Design Systems, Inc.

displayed in the table drop-down list, only the corners `C3` and `C4` are visible in the *Results* tab.



Test	Output	Nominal	Pass/Fail	Min	Max	C
TRAN	/OUT					
TRAN	Swing	942.2m		927.5m	942.2m	935
TRAN	SettlingTime	7.484 ns	pass	7.484 ns	7.752 ns	7.61
TRAN	RelativeSwingP...	78.51 %	pass	77.29 %	78.51 %	77.9
TRAN	PhaseMargin	12.23 degree	fail	12.23 degree	13.6 degree	12.87

To view the disabled corners `C1` and `C2` in the *Results* tab, select the *Disabled Corners* option in the *Configure what is shown in the table* drop-down list. The corresponding columns for corners `C1` and `C2` then display the text `disabled`.

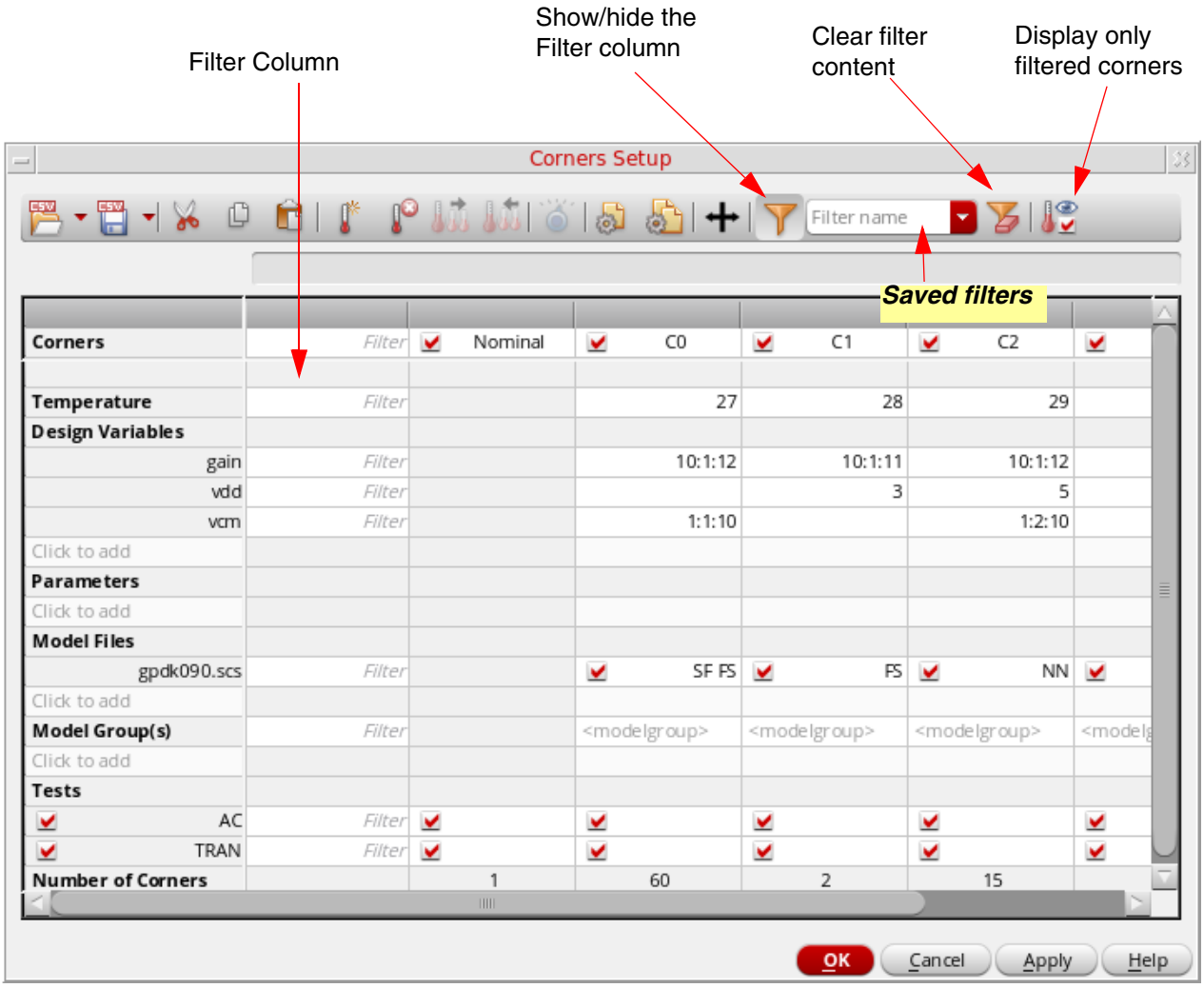
Test	Output	Nominal	Pass/Fail	Min	Max	C1	C2
TRAN	/OUT					disabled	disabled
TRAN	Swing	942.2m		927.5m	942.2m	disabled	disabled
TRAN	SettlingTime	7.484 ns	pass	7.484 ns	7.752 ns	disabled	disabled
TRAN	RelativeSwingP...	78.51 %	pass	77.29 %	78.51 %	disabled	disabled

For more details, see Disabling and Enabling Corners.

© 2023 Cadence Design Systems, Inc.

Filtering Corners

When the number of corners is large, you can apply filters to selectively display the corners that you want to work upon.



Video

You can view a video demonstration of this feature at Corner Filtering in ADE Assembler & Explorer.

To apply filters in the Corners Setup form:

1. Click the button in the toolbar.

   The Filter column is displayed as the first column.

© 2023 Cadence Design Systems, Inc.

2. Specify filters for the required rows.

   The accepted formats for specifying filters are:

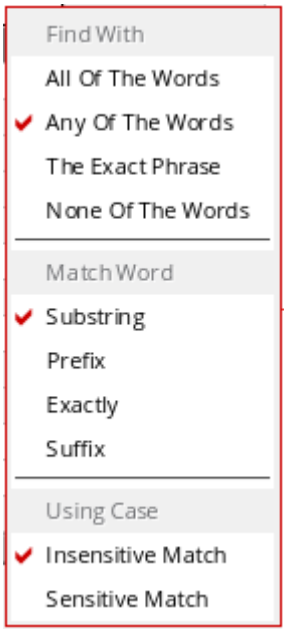
   - Space-separated list of values, such as `SF FS NN`
   - Comma-separated list of values, such as `27,28,29`
   - Range of values, such as `2:6` or `1.0:0.1:4.0`
   - Enabled or disabled status in the format `enabled:yes` or `enabled:no`

     For example, if you specify `enabled:yes` for a certain row, only the enabled columns for that row will be displayed.

   The corners are dynamically filtered as you type in the filter criteria.

3. (Optional) Right-click the *Filter* column to specify the following advanced filter options, as shown in the following figure.

   The options selected in this figure are the default selections.

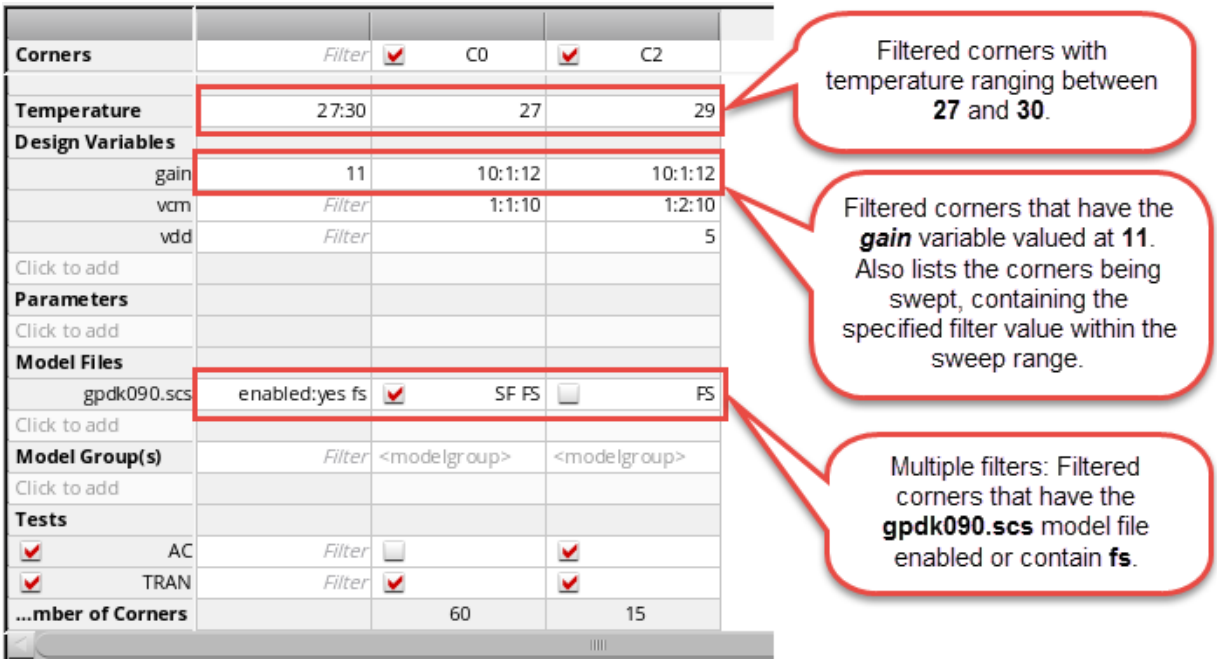


© 2023 Cadence Design Systems, Inc.

Advanced Filter	Description
Find With	Using this option, you can set your filter criteria to search: ■ All of the words in the selected filter ■ Any of the words in the selected filter ■ The exact phrase of the selected filter ■ None of the words in the selected filter
Match Words	Using this option, you can set your filter criteria to match: ■ *Substring*: displays the columns with values containing the filter criteria ■ *Prefix*: displays the columns with values starting with the filter criteria ■ *Exactly*: displays the columns with values exactly matching the filter criteria ■ *Suffix*: displays the columns with values ending with the filter criteria
Using Case	Using this option, you can specify whether the filter criteria should match casing or not.

© 2023 Cadence Design Systems, Inc.

The following figure shows examples of three filters applied in the Corner Setup form.



In the second example, the filter value 11 matches the corners with sweep value 10:1:12 also. This is because the sweep range in expanded form is 10 11 12 and it matches the default advanced filter, *Find With: Any Of The Words*.

To clear contents of all the filters at once, click .

4. (Optional) Click on the toolbar and choose the *Enable Visible, Disable Hidden* command to enable only the filtered corners in the simulation.

   Note: If you use this option to enable the filtered columns, the previously enabled corners that do not match the filter criteria will be disabled in the Data View pane. For more details, refer to Enabling or Disabling Filtered Corners.

Applying Multiple Filters

When you apply multiple filters to the rows with check boxes, the results are displayed based on the type of advanced filter applied to that filter. Let us consider the following example.

In the Corner Setup form, you want to filter the corners containing the `gpdk090.scs` model files with `FS` setting. To do so, apply a filter in the following format:

```
enabled:yes fs
```


© 2023 Cadence Design Systems, Inc.

This filter will give you results based on your selections of advanced filters.

Case: 1

Find With: All Of The Words

Match Word: Substring

Using Case: Insensitive Match

Corners	*Filter*	C2	C3
Temperature	*Filter*	29	30
Design Variables			
gain	*Filter*	10:1:12	10:2:13
vcm	*Filter*	1:2:10	
vdd	*Filter*	5	7
Click to add			
Parameters			
Click to add			
Model Files			
gpdk090.scs	enabled:yes fs	FS	FS
Click to add			
Model Group(s)	*Filter*	<modelgroup>	<modelgroup>

Gives results that match both the conditions.
So, the corners that have the `gpdk090.scs` model file enabled and also have `FS` set on them will be displayed.

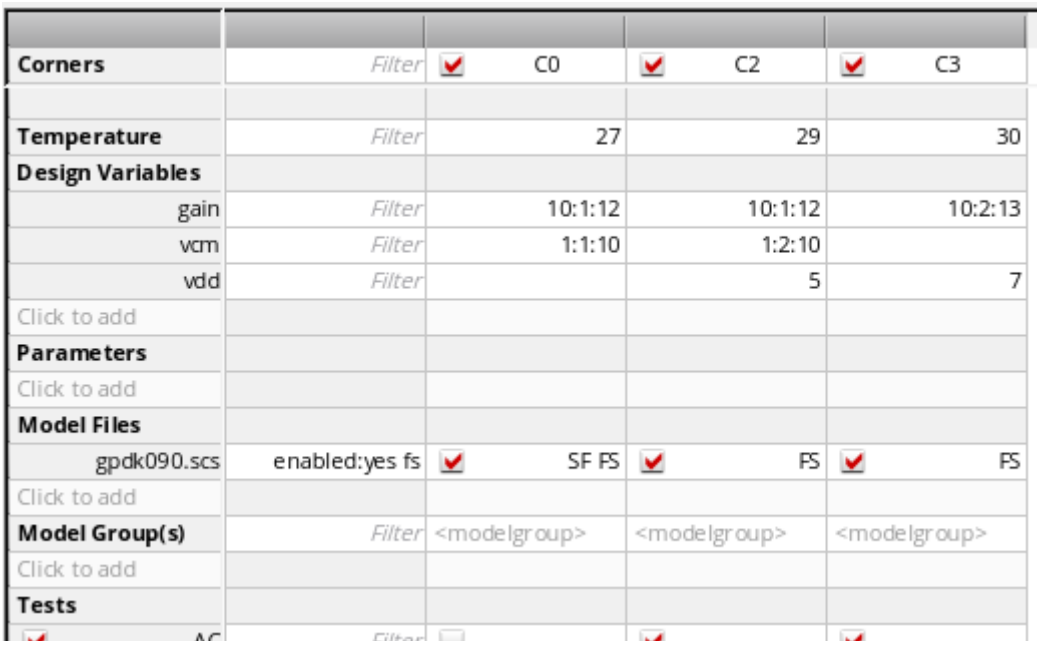
Case: 2

Find With: Any Of The Words

Match Word: Substring

© 2023 Cadence Design Systems, Inc.

Using Case: Insensitive Match



Corners	Filter	C0	C2	C3
Temperature	Filter	27	29	30
Design Variables				
gain	Filter	10:1:12	10:1:12	10:2:13
vcm	Filter	1:1:10	1:2:10	
vdd	Filter		5	7
Click to add				
Parameters				
Click to add				
Model Files				
gpdk090.scs	enabled:yes fs	SF FS	FS	FS
Click to add				
Model Group(s)	Filter	<modelgroup>	<modelgroup>	<modelgroup>
Click to add				
Tests				

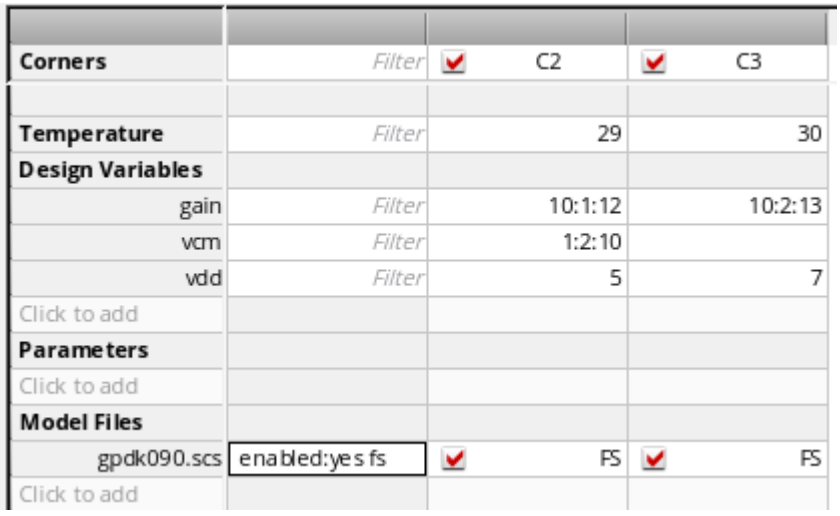
Gives results that match either of the conditions. So, the corners that have either the `gpdk090.scs` model file enabled, or have `FS` set on them will be displayed.

Case 3:

Find With: The Exact Phrase

Match Word: Exactly

Using Case: Insensitive Match



Corners	Filter	C2	C3
Temperature	Filter	29	30
Design Variables			
gain	Filter	10:1:12	10:2:13
vcm	Filter	1:2:10	
vdd	Filter	5	7
Click to add			
Parameters			
Click to add			
Model Files			
gpdk090.scs	enabled:yes fs	FS	FS
Click to add			

Gives results that exactly match the phrase used in the filter.
So, the corner that have the `gpdk090.scs` model file enabled and have only `FS` set on them will be displayed. No corners in this example.

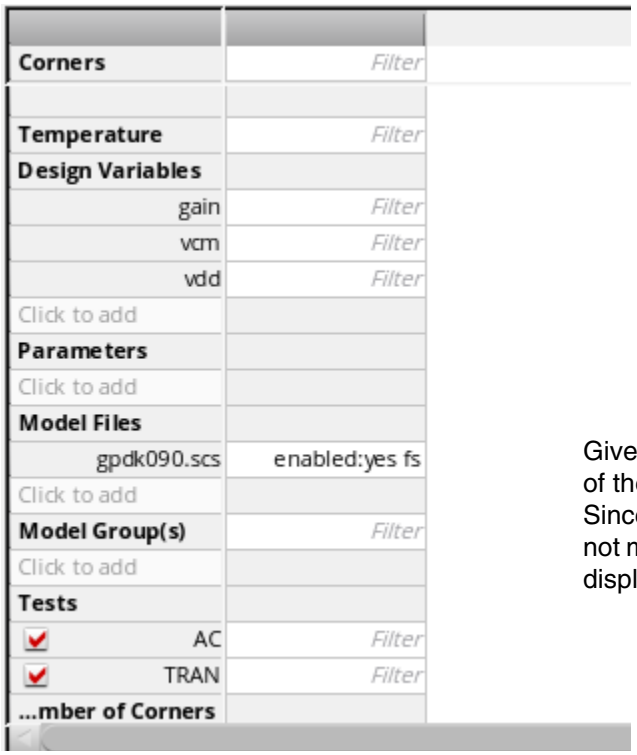
© 2023 Cadence Design Systems, Inc.

Case 4:

Find With: None Of The Words

Match Word: Substring

Using Case: Insensitive Match



Gives results that does not match any of the conditions.
Since the specified advance filters do not match any of the rows, no corner is displayed.

Enabling or Disabling Filtered Corners

After applying the filters to view the desired set of corners, you can enable or disable all the displayed or hidden corners by using the commands in the (*Enable or disable filtered corners*) drop-down list on the toolbar. This drop-down list contains the following commands:

© 2023 Cadence Design Systems, Inc.

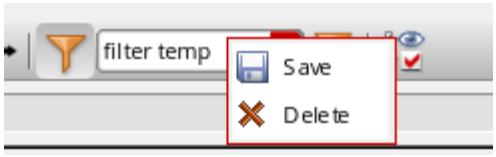
- *Enable Visible, Disable Hidden*: Enables all visible corners and disables all hidden corners. If any hidden corner was enabled earlier, it is disabled.
- *Enable Visible (Retain Status of Hidden Corners)*: Enables all visible corners. The status of hidden corners is retained as is. Therefore, after this command is used, the set of enabled corners includes the visible corners and the hidden, but enabled corners.
- *Disable Visible, Enable Hidden*: Disables all visible corners and enables all hidden corners. If any hidden corner was disabled earlier, it is enabled.
- *Disable Visible (Retain Status of Hidden Corners)*: Disables all visible corners. The status of hidden corners is retained as is. Therefore, after this command is used, the set of disabled corners includes the visible corners and the hidden, but disabled corners.

Note: Read a blog related to this at Virtuosity: What if I Want to Disable the Filtered Corners?.

Saving Filters

You can save a set of commonly used filters through the following steps:

1. Apply filters to the rows in the Corner Setup form.
2. In the *Filter Name* field on the toolbar, enter a name for the set of filters.
3. Right-click and select *Save*. Alternatively, you can also press `Enter` to save.



The saved filter name will start appearing in the *Filter Name* drop-down list. These filters are saved in the `.sdb` and will be available for the future sessions.

Deleting Saved Filters

To delete a saved filter:

➡ In the *Filter Name* drop-down list, right-click the named filter that you want to delete and click *Delete*.

© 2023 Cadence Design Systems, Inc.

Running Simulation with Corners

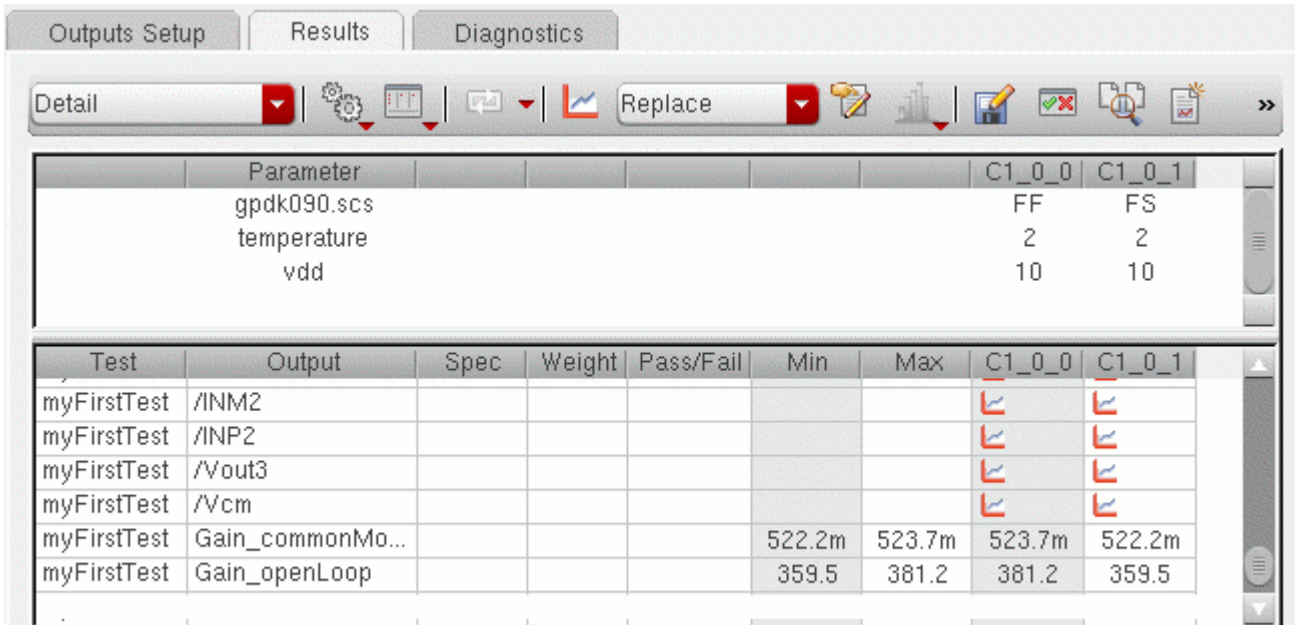
To run a simulation for a specific corner, do the following:

➡ In the Corners tree on the Data View pane, right-click a corner and choose *Run*.

To run a simulation for one or more corners, do the following:

1. In the Data View pane, ensure that the name of the corners for which simulation is required to be run are selected in the Corners tree.
2. Click *Run Simulation* on the Run toolbar.

The simulator runs simulation for the selected corners and shows results on the *Results* tab. If you run a corner group, simulations are run for all the corners in that group and results are displayed as shown in the following figure.



In case of a corner group, a column is displayed on the Results tab for each combination of corner parameters. Based on the results, if you want to create a corner by using the values of variables used for a particular data point, you can create a copy of that corner. For this, right-click in the corner column and choose *Create Copy Of Selected Corner*. A new corner is created with the name same as that of the corner column, prefixed with `Copy_`.

© 2023 Cadence Design Systems, Inc.

Important

While running simulations with corners, ADE Assembler sorts the values of variables and model sections in an alphabetical order. It also saves and displays the results in the sorted order. To disable sorting of variable values and model sections and to use the variables in the specified order, set the sortVariableValues variable to `nil`.

© 2023 Cadence Design Systems, Inc.

Plotting Across Corners

To plot waveforms across all corners for a signal or waveform expression, do the following:

1. Select the plot mode. For more information, see Selecting the Plot Mode.
2. Right-click any of the following on the Results tab and choose *Plot Across Corners*:
   - A test name in the *Test* column
   - A signal or waveform expression name in the *Output* column
   - In the *Nominal* column or in the column for a corner.

© 2023 Cadence Design Systems, Inc.

7

Working with Model Files and CDF

Model files contain definitions for models used in your design that are not defined in the library. You can specify the model files you want to reference for simulation on the Model Library Setup form. You can associate a model name (from one of your referenced model files) with a component on your schematic by setting the *Model name* CDF parameter. The *Model name* could refer either to a model definition or to a subcircuit (also called *macro*) definition. To netlist a subcircuit correctly, you must have a symbol cellview, a stopping cellview, and appropriate CDF information on the cell.

For example, the `opamp` schematic in the `aExamples` library contains instances of `npn` and `pnp` bipolar transistors (the *Model name* is set to `npn` or `pnp` accordingly). Cadence provides model definitions for Spectre circuit simulation of these parts in the file

your_install_dir`/tools/dfII/samples/artist/models/spectre/bipolar.scs`

You can declare this model library file by typing the path and file name above in the *Model Library File* field on the Model Library Setup form.

Note: See “Simple Device Models” in the Direct Simulation Models chapter of the *Direct Simulation Modeling Guide* for more information.

See also

- Specifying Model Libraries on page 115
- Associating a Model or Subcircuit Name with an Instance on page 304
- Editing Component CDF on page 305
- Adding a Model Name Parameter to a Component’s CDF on page 307
- Using Component CDF to Specify Simulation Information on page 309
- Creating a Stopping Cellview on page 310
- Varying the Model File and Section during Simulation on page 311
- Editing a Model File on page 314
- Disabling a Model File on page 314

© 2023 Cadence Design Systems, Inc.

Associating a Model or Subcircuit Name with an Instance

To associate a model or subcircuit name with an instance on your schematic do the following:

1. On the schematic, select the component.

   The instance name and its parameters appear in the scrolling list area in the top section of the *Parameters* tab on the Variables and Parameters assistant pane.

2. Locate the model-name parameter (such as *model*).

   You can use a filter to narrow the device instance parameter selection list.

   If your component does not have a model name parameter in its CDF, see _____.

   If your component does have a model name parameter but the parameter value is not editable, _____.

3. Double-click in the *Value* column and type a model name.

4. Use the Model Library Setup form to specify the file containing the model or subcircuit definition whose name you typed in the *Value* column (see).

Related Topics

Model Library Setup form

Specifying Model Libraries

Variables and Parameters Assistant

Filtering Device Instance Parameters

Adding a Model Name Parameter to a Component's CDF

Making the Model Name Parameter Editable

Toggling the View on the Variables tab of the Variables and Parameters Assistant Pane

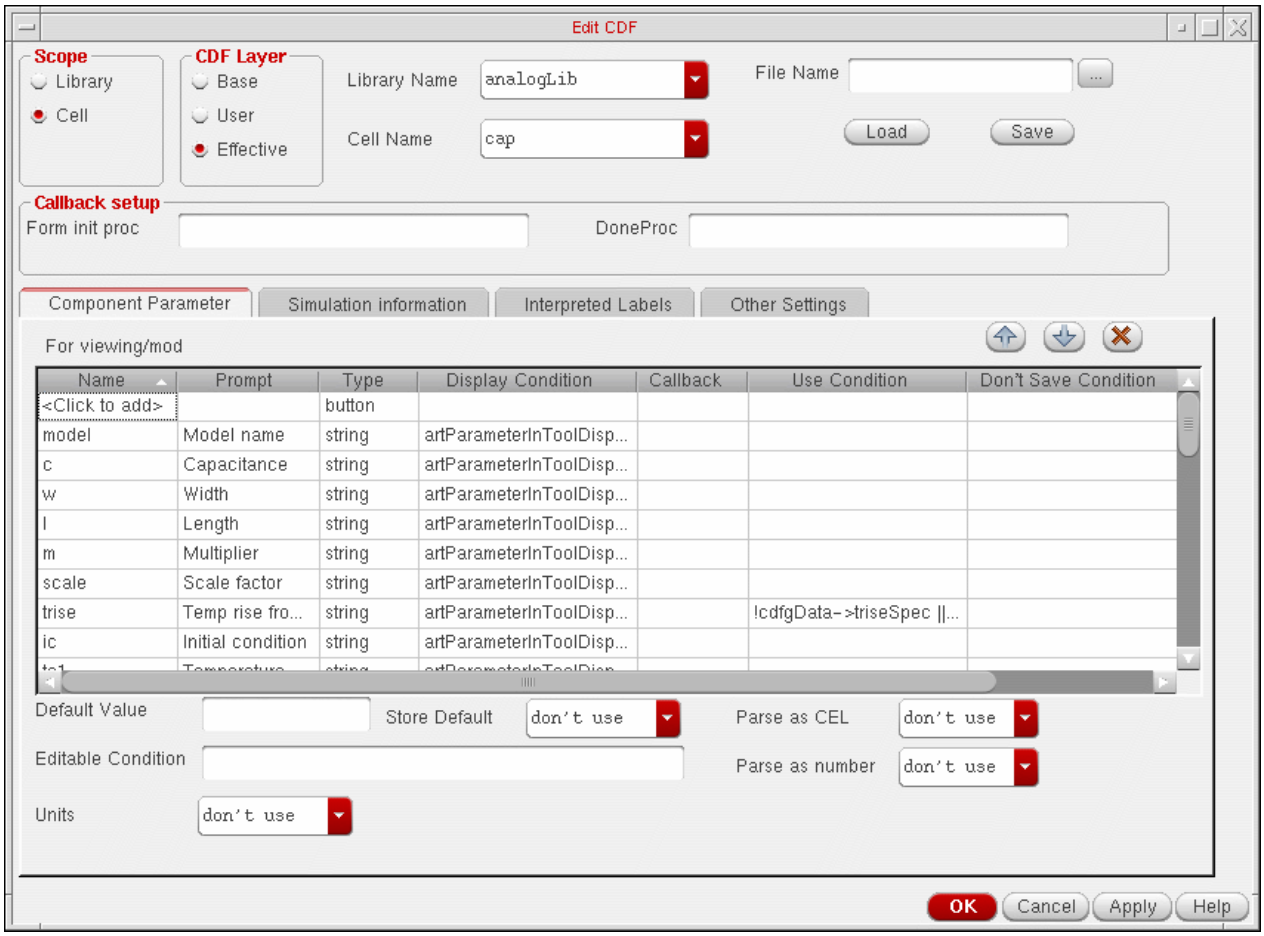
© 2023 Cadence Design Systems, Inc.

Editing Component CDF

To Edit CDF, do the following:

1. In the CIW, choose *Tools – CDF – Edit*.

   The Edit CDF form appears.



2. For *Scope*, select the *Cell* radio button.

   See “CDF Selection” in the "CDF Commands" chapter of the *Component Description Format User Guide* for information about this selection.

3. In the *CDF Layer* group box, select a CDF type: *Effective*, *Base*, or *User*.

   See “CDF Layer” in the "CDF Commands" chapter of the *Component Description Format User Guide* for information about this selection.

4. Click *Browse*.

© 2023 Cadence Design Systems, Inc.

The Library Browser form appears.

5. In the *Library Name* drop-down list, select a library.
6. In the *Cell* drop-down list, select a cell.

   The CDF information for the cell is displayed on the Edit CDF form.

See also

- Adding a Model Name Parameter to a Component's CDF on page 307
- Making the Model Name Parameter Editable on page 308
- Using Component CDF to Specify Simulation Information on page 309

© 2023 Cadence Design Systems, Inc.

Adding a Model Name Parameter to a Component's CDF

To add a parameter for model name to your component, do the following:

1. Open the Edit CDF form.
2. Specify the cell (component) whose CDF you want to edit.
3. In the *Name* column on the *Component Parameter* tab, click where it says *<Click to add>.*

   Specify your model name parameter as follows:

Column	Value
Name	Type `model`.
Prompt	Type `Model name`.
Type	Select *string* as the parameter type.
Parse as CEL	Select *no* from the drop-down list. You do not want your model name parsed as an expression.
Parse as number	Select *no* from the drop-down list. You do not want your model name parsed as a number.

4. Click *Apply* to apply your changes and leave the form open, or *OK* to apply your changes and close the form.

For more information, see "Defining Parameters" in the *Component Description Format User Guide*.

© 2023 Cadence Design Systems, Inc.

Making the Model Name Parameter Editable

To make the model name parameter editable (so that you can vary it during simulation), do the following:

1. Use the Edit CDF form to edit the component's CDF.
2. In the *CDF Layer* group box, select *Base*.
3. In the *Component Parameter* tab, select the model-name parameter (such as *model*).
4. In the *Editable Condition* field, type a non-nil value.
5. Click *OK*.
6. On the Edit CDF form, click *OK*.

   When you select this component instance on your schematic, the model-name parameter appears in the list of *CDF Editable* device instance parameters in the Parameters tab on the Variables and Parameters assistant pane.

© 2023 Cadence Design Systems, Inc.

Using Component CDF to Specify Simulation Information

To specify how you want the software to netlist your component, what parameters to pass to the underlying subcircuit or model definition, and the order of the input terminals (for netlisting), do the following:

1. Use the Edit CDF form to edit the component's CDF.
2. Click the *Simulation Information* tab.
3. Select the *By Simulator* radio button and select your simulator in the drop-down list.
4. In the following fields, you can specify how you want the software to netlist your component, what parameters to pass to the underlying subcircuit or model definition, and the order of the input terminals (for netlisting):

Field	Value
netlistProcedure	Type the name of a netlist procedure or leave this field blank to use the default netlist procedure.
otherParameters	Type `model` to specify a parameter for model or subcircuit name.
instParameters	Type the names of one or more component parameters (space-separated) you want to pass into the subcircuit or model.
termOrder	Type the names of the symbol's terminals in the order you want them netlisted (the order must match the node order on the `subckt` line or that of the model referenced).

Note: For more information, see "Modifying Simulation Information" in the *Component Description Format User Guide*.

© 2023 Cadence Design Systems, Inc.

Creating a Stopping Cellview

Important

You must have write privileges on the library to perform this task.

To create a stopping cellview for your simulator, do the following:

1. Edit the symbol cellview.
2. Choose *Design – Save As*.

   The Save As form appears.
3. In the *View Name* field, type a view name to represent your simulator.

   For example, for the Spectre circuit simulator, type `spectre`.
4. Click *OK*.

   The environment creates a new cellview (as a directory of files) in the specified library.

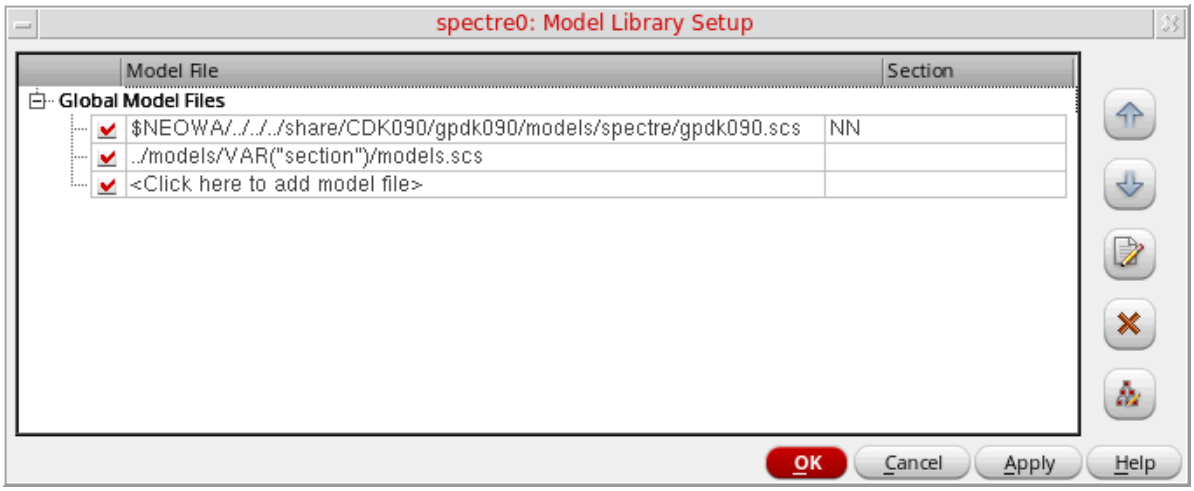
© 2023 Cadence Design Systems, Inc.

Varying the Model File and Section during Simulation

To make the model file or section something you can vary during simulation, do the following:

1. On the Data View assistant pane, right-click the test or analysis name and choose *Model Libraries*.

   The Model Library Setup form appears.



2. In the *Model File* column, double-click and type a variable name for the model file you want to vary (or as a part of the model file path) using the following format:

   `VAR("variableModelFileName")`

   For example:

   ```
   VAR("myModelFile")
   ```

   or

   ```
   ../models/VAR("section")/models.scs
   ```

3. In the *Section* column, double-click and type a variable name for the section you want to vary. For example:

   `VAR("myModelCorner")`

   Note: For more information about sections in model files, see "Corners Modeling" in the Direct Simulation Modeling User Guide.

4. Click *Apply*.

   Variables you specify using `VAR` (such as *myModelFile* or *section*, and *myModelCorner*) appear on the Variables tab of the Variables and Parameters assistant pane.

© 2023 Cadence Design Systems, Inc.

5. Double-click in the *Value* column for each global variable and type a value. For example:

Name	*Value*
Global Variables	
myModelCorner	`"NN"`
myModelFile	`"$PROJECT1/path/to/myModels.scs"`

You can vary these global variables in parametric sweeps and corners analysis.

© 2023 Cadence Design Systems, Inc.

Here is an example application using `VAR` in the specification of a model file to sweep through different model types.

If you have model files stored in a directory hierarchy such as the following:

`../models/`*`sectionIdentifier`*`/models.scs`

where *`sectionIdentifier`* is the model type (such as `typ`, `ff`, `ss`), you can create a sweep of model types as follows:

1. On the Model Library Setup form, double-click in the *Model File* column and type the following:

   ```
   ../models/VAR("section")/models.scs
   ```

2. Click *OK*.

   The *section* variable appears in the *Global Variables* tree on the Variables tab of the Variables and Parameters assistant pane.

3. Double-click in the *Value* column for *section* and click the ellipsis that appears at the right end of the field.

   The Parameterize form appears.

4. Click Add Specification and select *Inclusion List* from the drop-down list:

5. In the *Values* field, type the set of model types (valid section names from the model file) through which you want the program to sweep.

   For example:

   ```
   "typ", "ff", "ss"
   ```

   The program sweeps through all values of `section`.

© 2023 Cadence Design Systems, Inc.

Editing a Model File

To edit a model file, do the following:

1. On the Data View assistant pane, right-click the test or analysis name and choose *Model Libraries*.

   The Model Library Setup form appears.

2. Select the model library you want to edit and click *Edit File*.

   The model file appears in an editing window.

3. Make the edits you want to make.
4. Save and exit the file.
5. On the Model Library Setup form, click *OK*.

Disabling a Model File

To disable a model file for a test, do the following:

1. On the Data View assistant pane, right-click the test or analysis name andchoose *Model Libraries*.

   The Model Library Setup form appears.

2. Deselect the check box for the model library you want to disable.
3. Click *OK*.

© 2023 Cadence Design Systems, Inc.

8

Working with DSPF Files

The Detailed Standard Parasitic Format files (DSPF files) contain the post-layout data, which includes the designed and parasitic devices in a design. These files are created using the parasitic extraction tools. Therefore, the content and format of each DSPF file is dependent on the extraction tool. To run simulations for designs containing devices represented by DSPF files, ADE Assembler needs to consider these components in a way different than the devices with schematic.

The following sections explain how you can use DSPF files while running simulations in ADE Assembler:

- Including DSPF Files in the Setup
- Rules for Adding Prefixes to Instance Names
- Customizing Netlist Creation Using the Options in .simrc
- Customizing Netlist Creation Using Environment Variable

© 2023 Cadence Design Systems, Inc.

Including DSPF Files in the Setup

You can include DSPF files in the ADE Assembler setup in the following ways:

- Specifying DSPF Files in the Simulation Files Setup Form
- Sweeping DSPF Files in the Corners Setup Form
- Sweeping DSPF Files in the Simulation Files Setup Form Using the VAR Function
- Sweeping DSPF Files in the Corners Setup Form Using the VAR Function
- Binding a DSPF View to a Design

Specifying DSPF Files in the Simulation Files Setup Form

To specify a DSPF file for a test, perform the following steps:

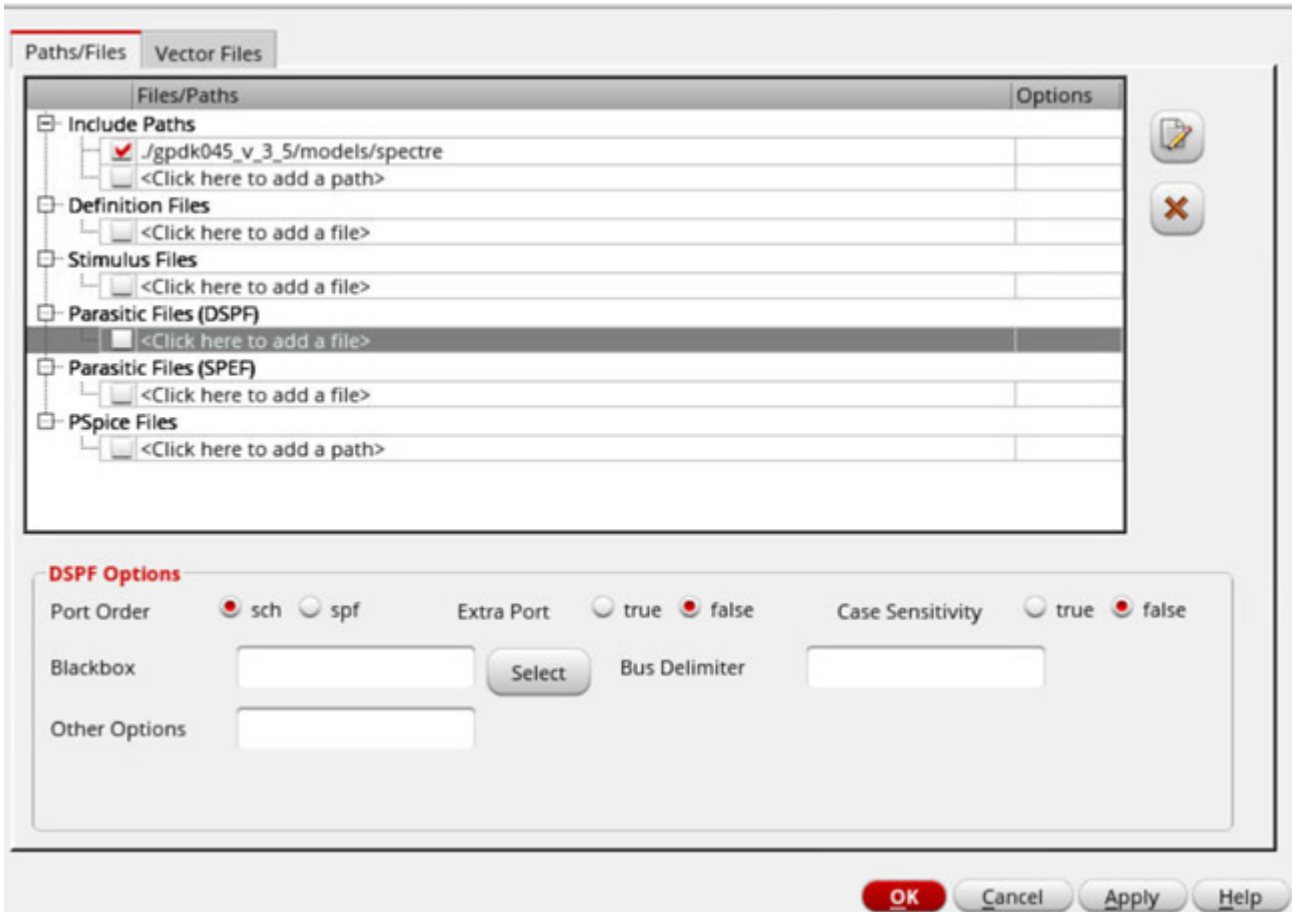
1. In the Data View assistant, right-click a test and choose *Simulation Files*.

   The Simulation Files Setup form is displayed.

2. On the *Paths/Files* tab of the Simulation Files Setup form, specify the name of the DSPF file in the *Parasitic Files (DSPF)* field.

© 2023 Cadence Design Systems, Inc.

A few additional options are displayed in the *DSPF Options* section on the form.



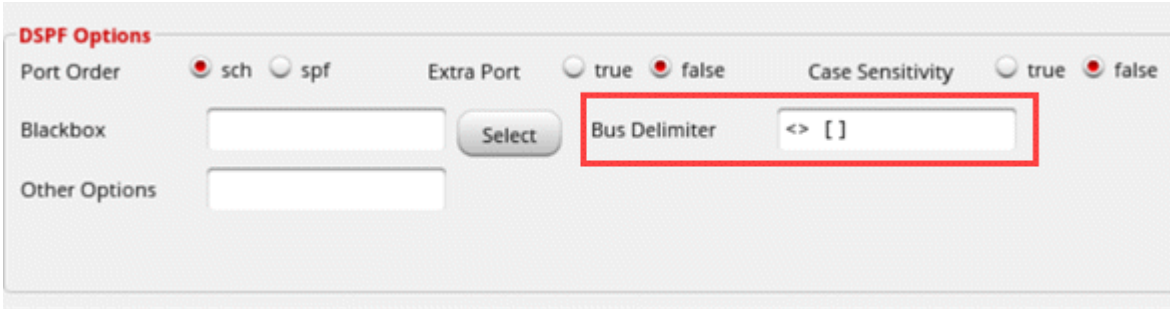
3. Specify the additional options to be used for the included DSPF file.

   - *Port Order*: Specifies whether to take the port order of the subcircuit definition from the pre-layout schematic netlist or from the DSPF file subcircuit definition.
     - *sch* specifies that the port order is taken from schematic subcircuit definition.
     - *spf* specifies that the port order is taken from the DSPF subcircuit definition
   - *Extra Port*: Specifies how to handle extra ports in the DSPF or schematic subcircuit definition. By default, this option is set to false, which implies that the port number in the schematic and the DSPF file needs to be the same. If not, the simulation will stop. When you set this option to true, the extra ports in the DSPF subcircuit call are changed to internal nodes. In addition, the extra ports in the schematic subcircuit definition are connected to nodes with the same name in the DSPF netlist, otherwise, they will be floating.

© 2023 Cadence Design Systems, Inc.

- *Case Sensitivity*: By default, the DSPF file is considered case insensitive. To make the DSPF file case-sensitive, set this option to `true`.
- *Blackbox*: Specifies the instances to be considered as blackboxes. For the blackbox subcircuit instances, Spectre takes the port order from the DSPF subcircuit definition, and the content from the schematic (blackbox) subcircuit definition
- *Bus delimiter*: Enables you to map the bus delimiter between the schematic and DSPF or SPEF files. This option defines the bus delimiter in the schematic netlist, and optionally the bus delimiter in the DSPF file. By default, the bus delimiter of the DSPF file is automatically taken from the DSPF file header (that is, `*|BUSBIT [ ]` or `*|BUS_DELIMITER [ ]`). If the bus delimiter is not defined in the parasitic file header, you need to specify it in the `dspf_include bus_delim` statement. If both are specified, then the parasitic file bus delimiter is taken from the `dspf_include bus_delim`. This option specifies the value for `dspf_include bus_delim`.

  Buses are typically specified in the schematic using angular brackets as the delimiters, for example, bus<0:4>. Buses in the DSPF are flattened and use square brackets, for example, bus[0], bus[1], bus[2], bus[3], and bus[4]. To map between the different bus delimiters, specify the schematic bus delimiters followed by the DSPF bus delimiters in the Bus Delimiter field, as shown in the figure.



- *Other Options*: Any `dspf_include` option other than those available on this form. For example, you can enter the `auto_blackbox=true` option to specify that Spectre needs to automatically consider a subckt as a blackbox subckt if it does not have the following:
  - `*|NET` section
  - `*|GROUND_NET` section
  - Any valid device in the `*Instance` section

4. Click *OK* to save and close the form.

   ADE Assembler uses the DSPF file for the subcircuit defined in it.

© 2023 Cadence Design Systems, Inc.

Important

If you specify the same DSPF file in the Simulations Files Setup as well as in the Corners Setup, the simulations will not run.

Annotating DSPF Results for DC node Voltage and DC Operating Points

When you specify DSPF files in your simulation setup, you must manually add the `cap` parameter in the CDF Editor to view the DC node voltage and DC Operating points annotation for all devices.

To add the cap parameter in the CDF Editor:

1. In the CIW, choose *Tools – CDF – Edit*.

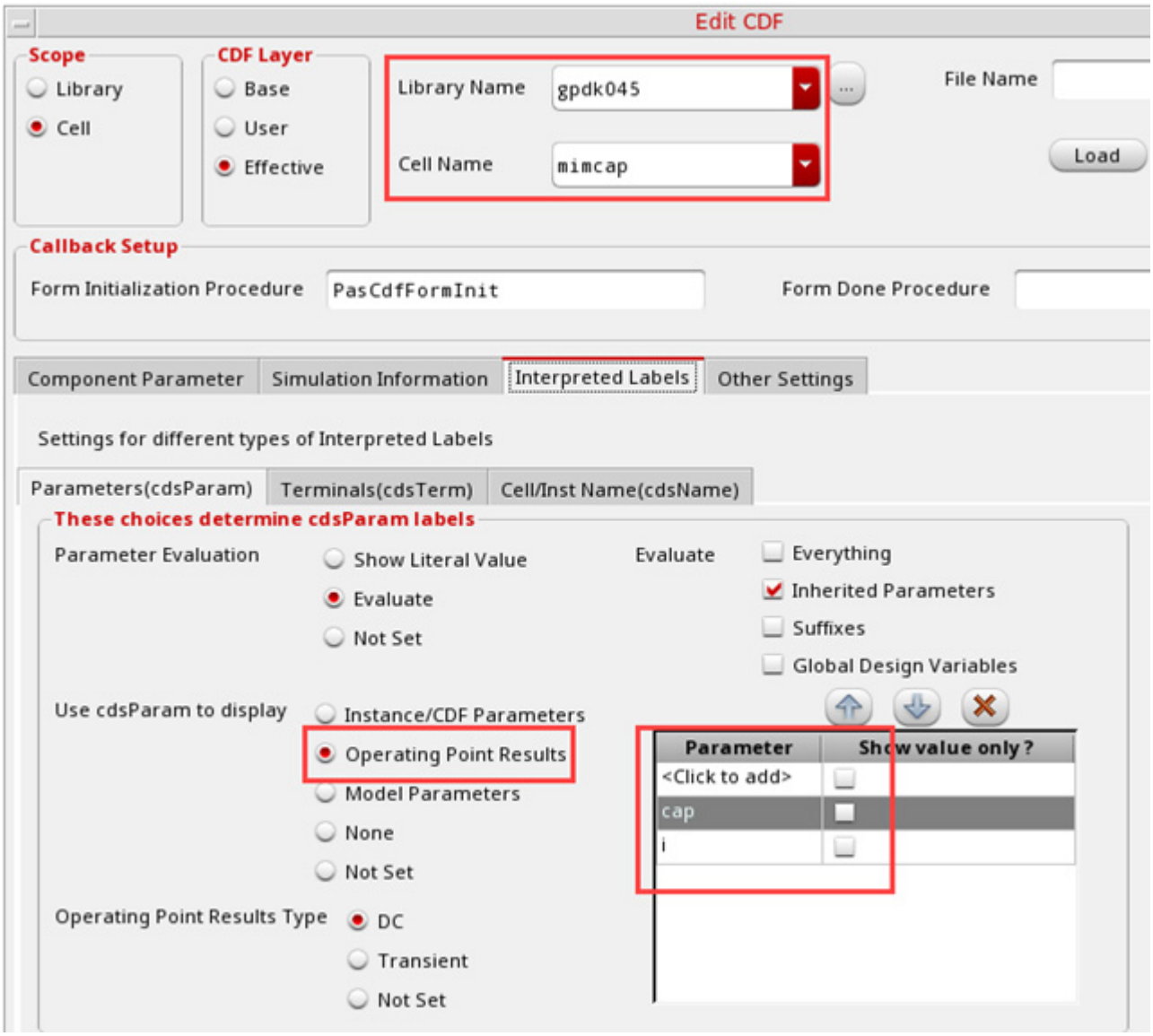
   The Edit CDF form appears.

2. On the Edit CDF form:
   - In the *Library Name* field, select the library whose CDF you want to edit.
   - In the Cell Name list, select `mimcap`.
   - Click the *Interpreted Labels* tab.
   - On the Interpreted Labels tab, click the *Parameters(cdsParam)* tab.
   - On the *Parameters(cdsParam)* tab, select *Operating Point Results* from the *Use cdsParam to display* group of options.

     This enables the Parameters list box.

© 2023 Cadence Design Systems, Inc.

3. In the *Parameter* column, add `cap`.



4. Click *OK*.

To view the annotation for DC node voltage and DC Operating Points results in the schematic:

1. On the *Results* tab of the *Outputs* pane, right-click a data point and choose *Annotate*.
2. In the sub-menu that appears, select *DC Node Voltages* and *DC Operating Points*.

   Assembler annotates your design schematic with the DC node voltage values and DC operating point data.

© 2023 Cadence Design Systems, Inc.

Sweeping DSPF Files in the Corners Setup Form

If you have multiple DSPF files that you can use for a particular device, and you want to use one file at a time, you can sweep the files using the Corners Setup form.

To sweep the DSPF files:

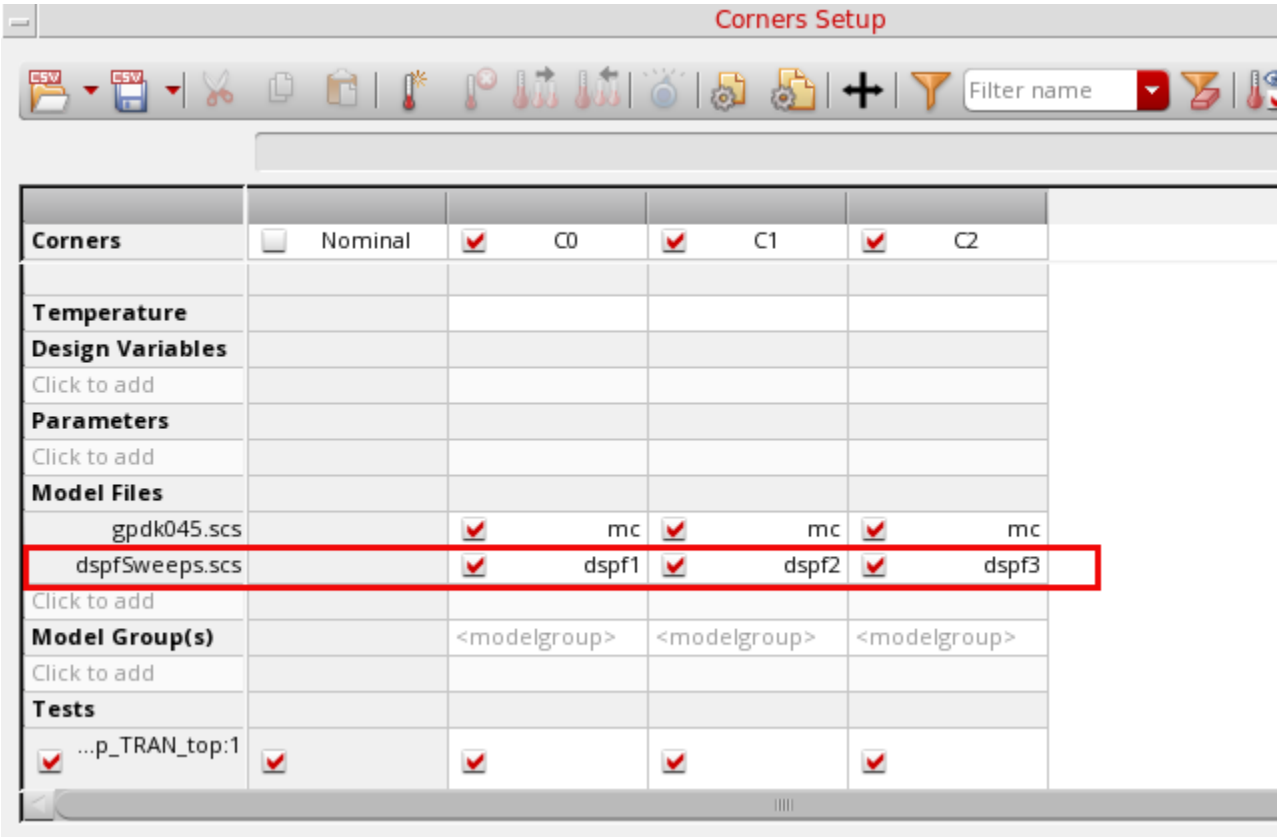
1. Create a model file that a separate section for the each DSPF file to be swept.

   The following code shows a sample content of a model file:

```
section dspf1
dspf_include "qrc_diffOpAmp.dspf"
endsection dspf1

section dspf2
dspf_include "qrc_diffOpAmp2.dspf"
endsection dspf1section dspf2

section dspf3
dspf_include "qrc_diffOpAmp3.dspf"
endsection dspf3
```


© 2023 Cadence Design Systems, Inc.

2. In the Corners Setup form, include the model file and select a unique section for each corner.

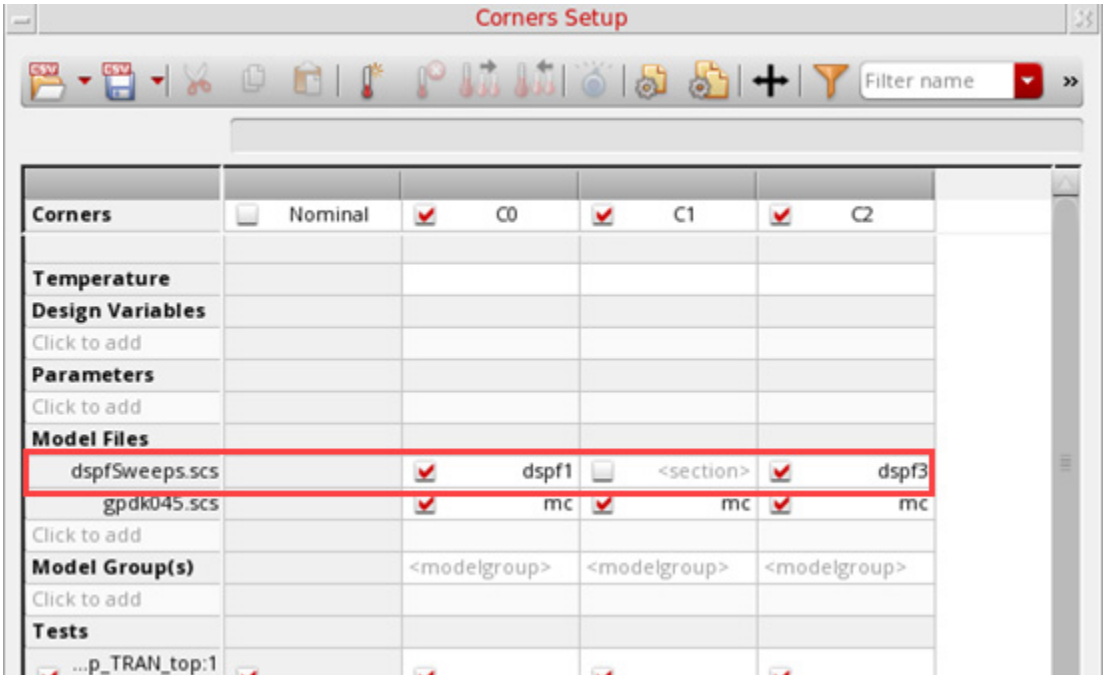


ADE Assembler uses the DSPF file for the subcircuit defined in it.

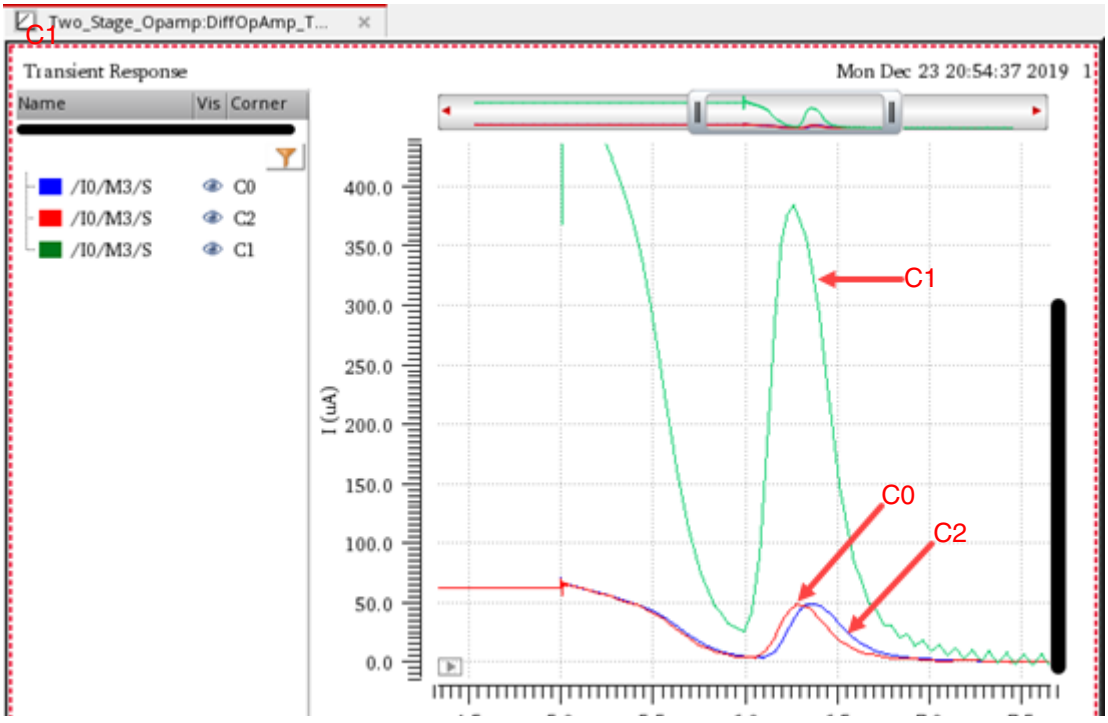
If required, you can use the schematic view for selected corners and a DSPF file for the remaining. For example, in the following corners setup, corner `C1` is not using the DSPF

© 2023 Cadence Design Systems, Inc.

file. Therefore, it will use the schematic view. In this case, the output shows the difference in values of corners `C0` and `C2` as compared to corner `C1`.



The output shows the difference in values of corners `C0` and `C2` as compared to corner `C1`.



© 2023 Cadence Design Systems, Inc.

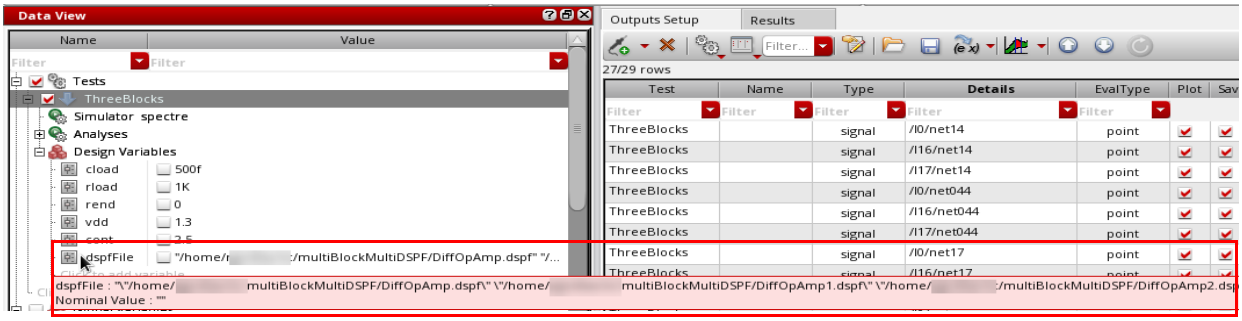
Important

If you specify the same DSPF file in the Simulations Files Setup as well as in the Corners Setup, the simulations will not run.

Sweeping DSPF Files in the Simulation Files Setup Form Using the VAR Function

To specify multiple DSPF files for a test in the Simulation Files Setup form:

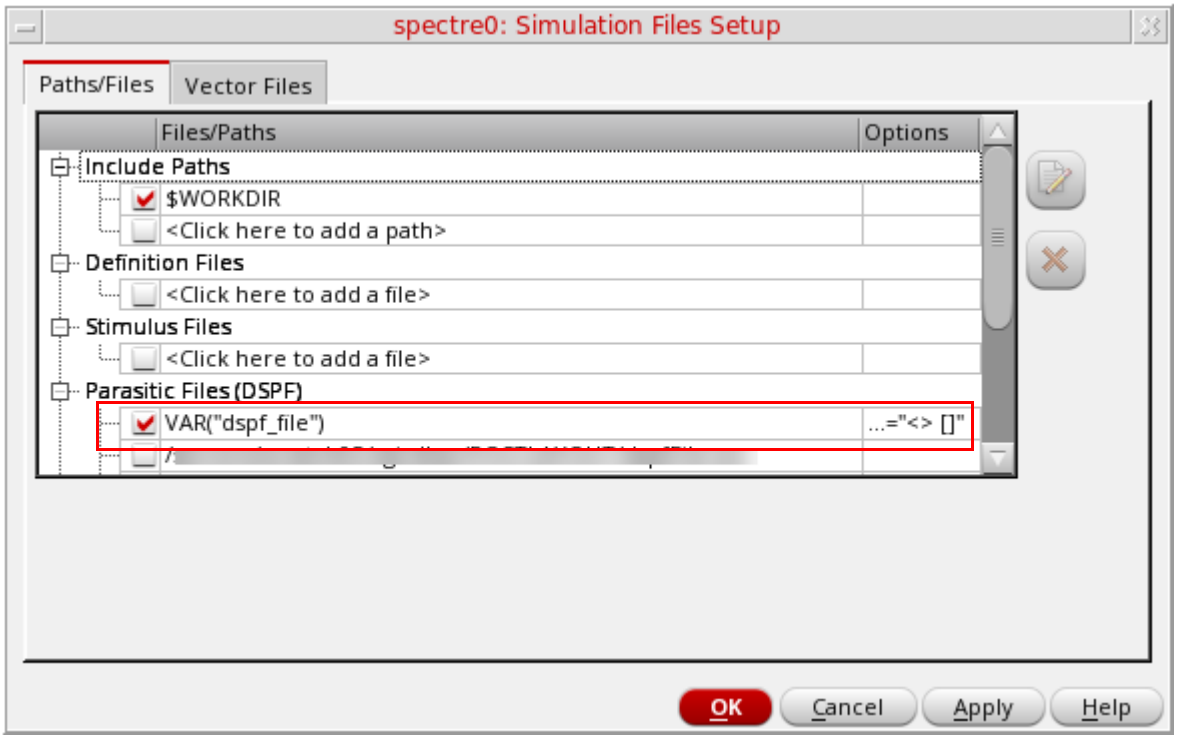
1. In the Data View assistant, add a design variable or global variable to specify the path to one or more DSPF files.



Here, the design variable is named `dspfFile` and its value has been set to the path of three DSPF files.

© 2023 Cadence Design Systems, Inc.

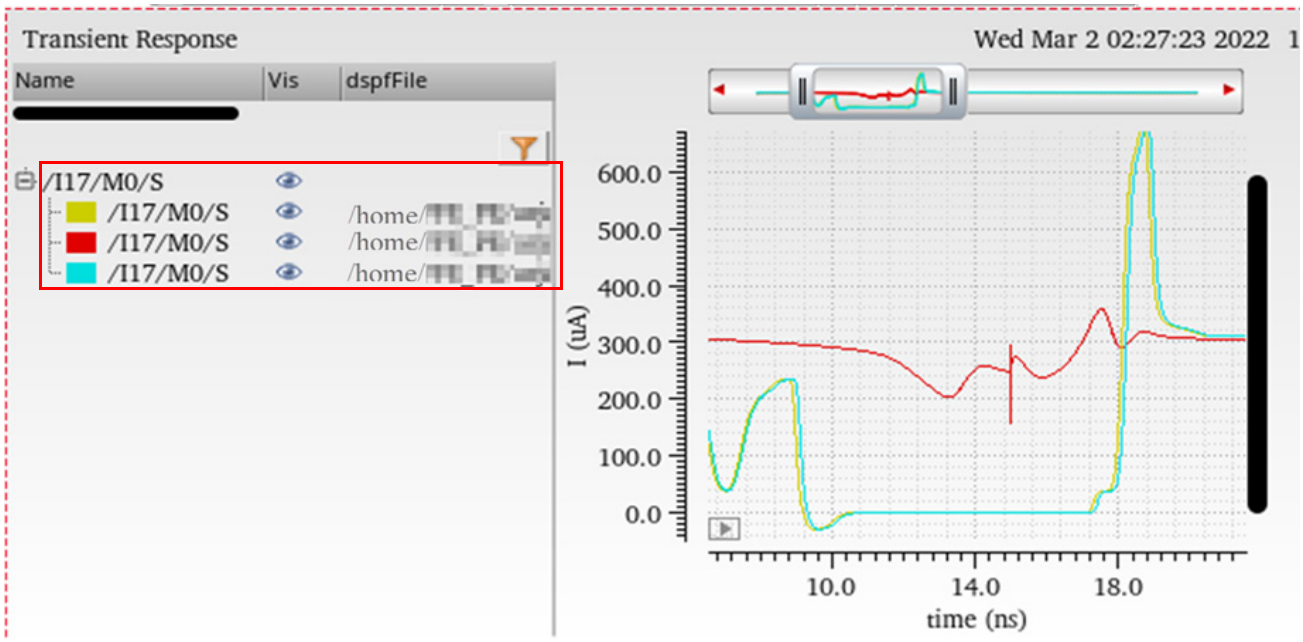
2. On the *Paths/Files* tab of the Simulation Files Setup form, specify the name of the design variable enclosed within the VAR function in the *Parasitic Files (DSPF)* field.



3. Click *Run* on the toolbar to run the simulation.
4. Right-click a point from the *Results* tab and select *Plot Across Design Points* from the context menu.

© 2023 Cadence Design Systems, Inc.

The Results Browser displays the plotted outputs for the three DSPF files that are swept.



Sweeping DSPF Files in the Corners Setup Form Using the VAR Function

To specify different DSPF files in the Corners Setup form:

1. Create a model file that contains sections to be used by multiple DSPF files.

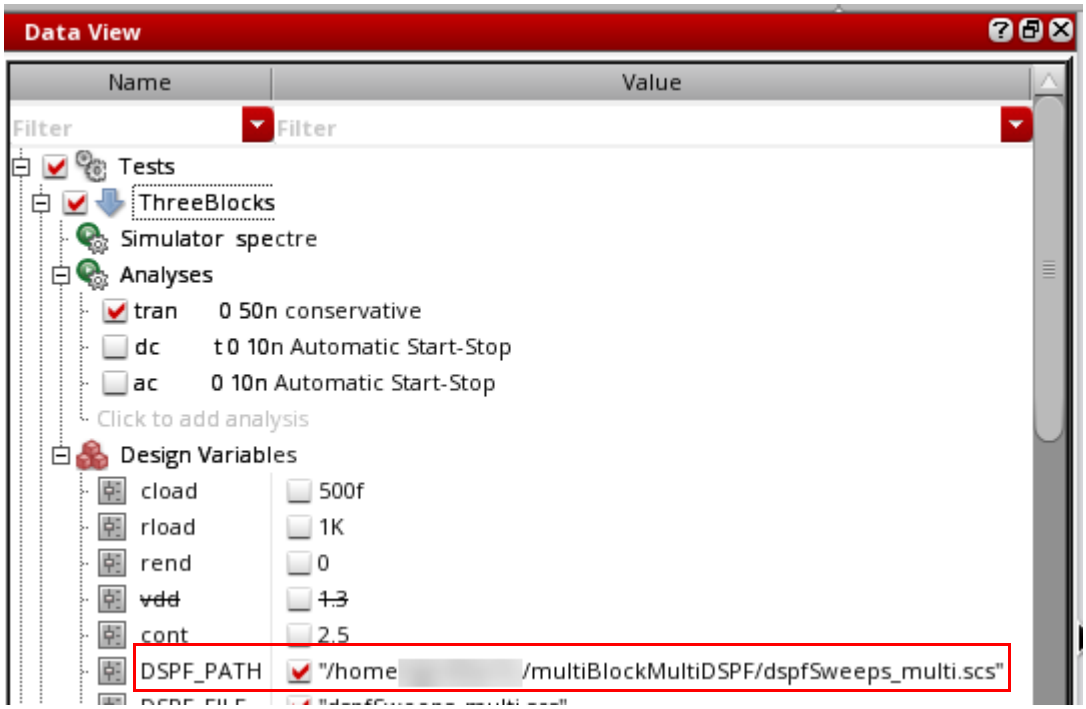
   The following code shows the sample content of a model file:

   ```
   section dspf1
   dspf_include "DiffOpAmp.dspf"
   dspf_include "DiffOpAmpB.dspf"
   dspf_include "DiffOpAmpC.dspf"
   endsection dspf1

   section dspf2
   dspf_include "DiffOpAmp1.dspf"
   dspf_include "DiffOpAmpB1.dspf"
   dspf_include "DiffOpAmpC1.dspf"
   endsection dspf2

   section dspf3
   dspf_include "DiffOpAmp2.dspf"
   dspf_include "DiffOpAmpB2.dspf"
   dspf_include "DiffOpAmpC2.dspf"
   endsection dspf3
   ```

2. Save this model file at the required location.

© 2023 Cadence Design Systems, Inc.

3. In the Data View assistant, specify a design variable, such as *DSPF_PATH*, that points to the location of the model file.

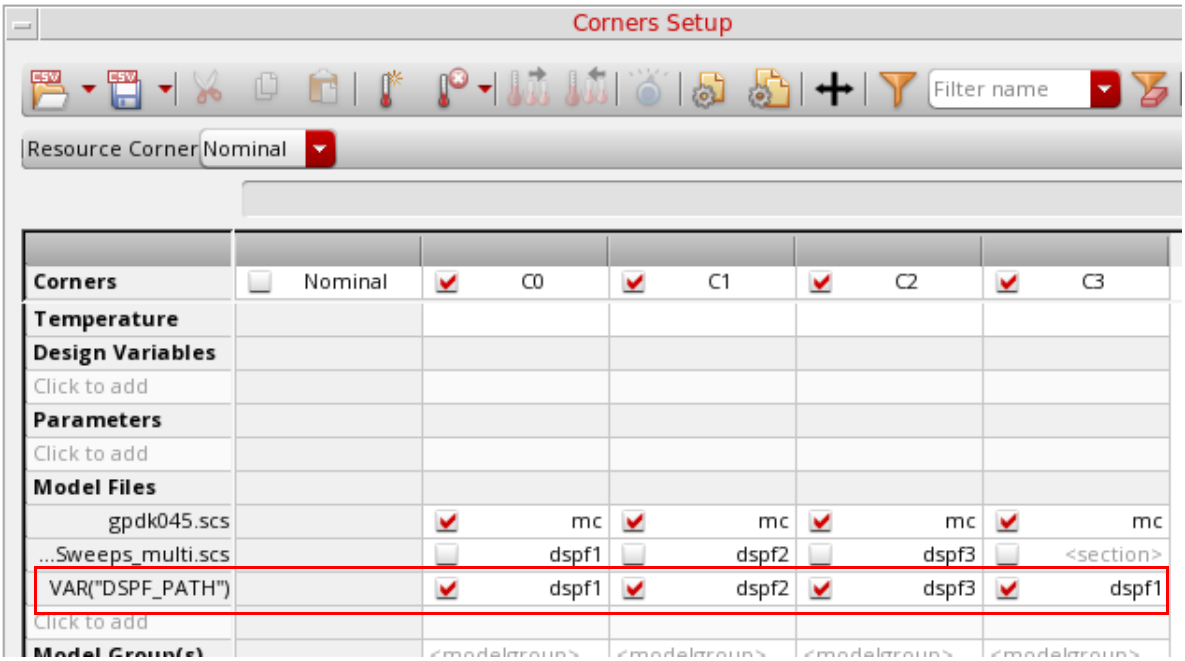


4. In the Data View assistant, right-click the *Corners* group and select *Open Corners Setup*.

   The Corners Setup form is displayed.

© 2023 Cadence Design Systems, Inc.

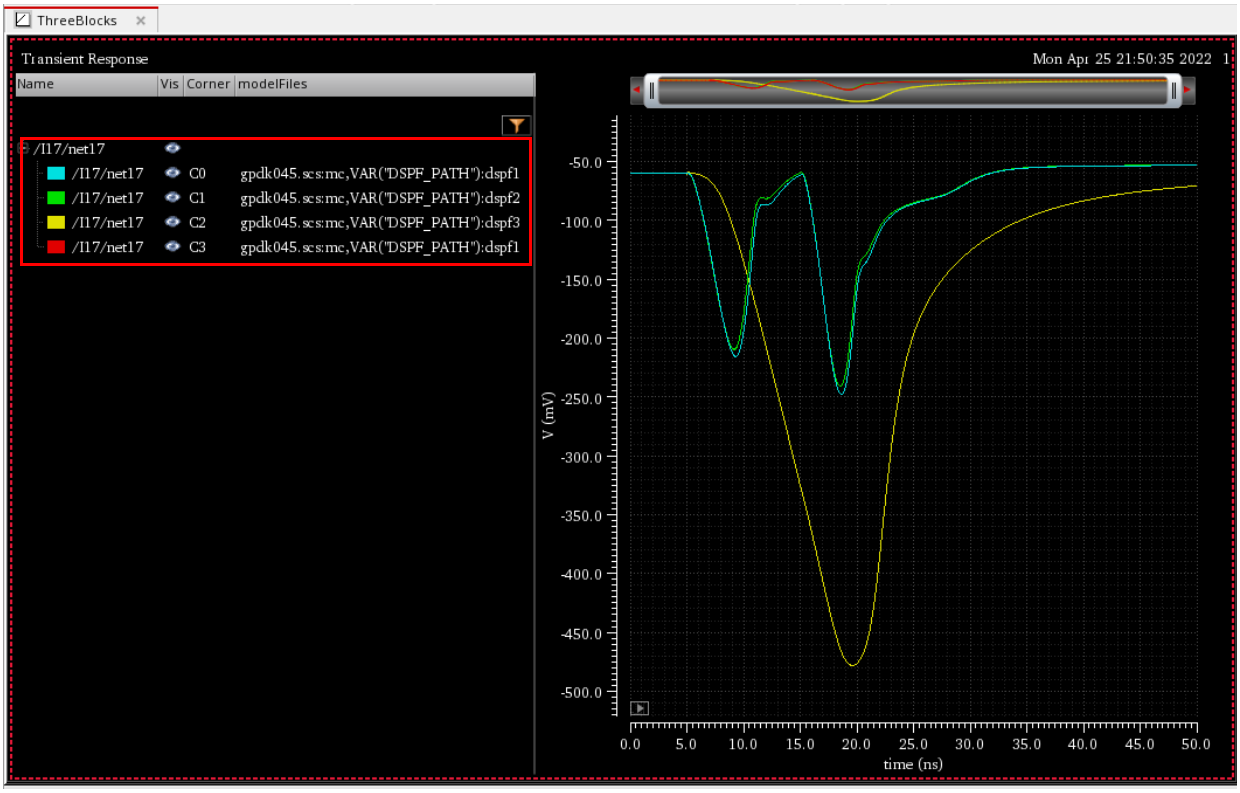
5. In the *Model Files* field, specify the model file by enclosing it within the VAR function and specify a section for each corner.



6. Click *OK* on the Corners Setup form.
7. Click *Run* on the toolbar to run the simulation.
8. Right-click a point from the *Results* tab and select *Plot Across Corners* from the context menu.

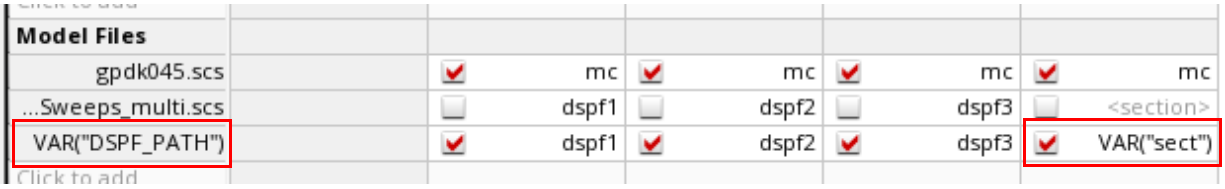
   The Results Browser displays the plotted outputs for all corners where different DSPF files have been swept.

© 2023 Cadence Design Systems, Inc.



Here, each corner uses a different set of DSPF files.

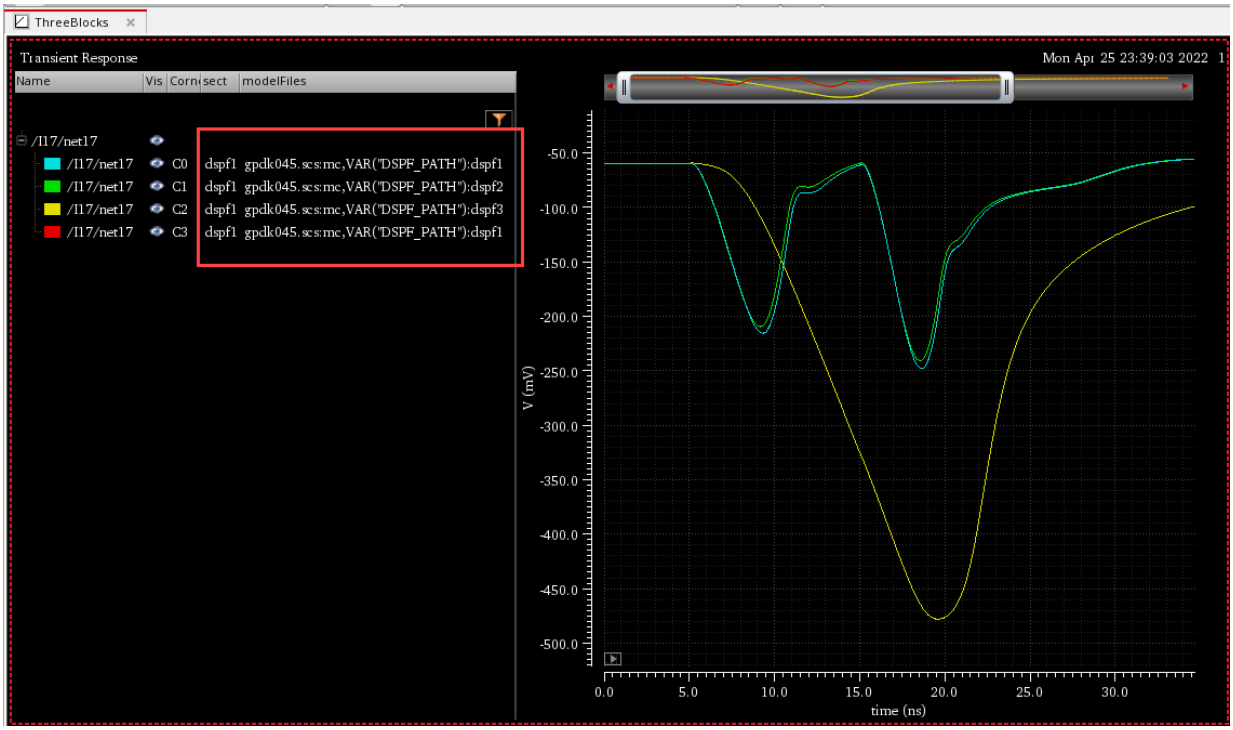
9. Optionally, in the Corners Setup form, use the VAR function to specify different model sections for a corner.



10. Click *OK* on the Corners Setup form.
11. Click *Run* to run the simulation.
12. Right-click a point from the Results tab and select *Plot Across Corners* from the context menu.

© 2023 Cadence Design Systems, Inc.

The Results Browser displays the plotted outputs for all corners where different DSPF files as well as model sections have been swept.



Binding a DSPF View to a Design

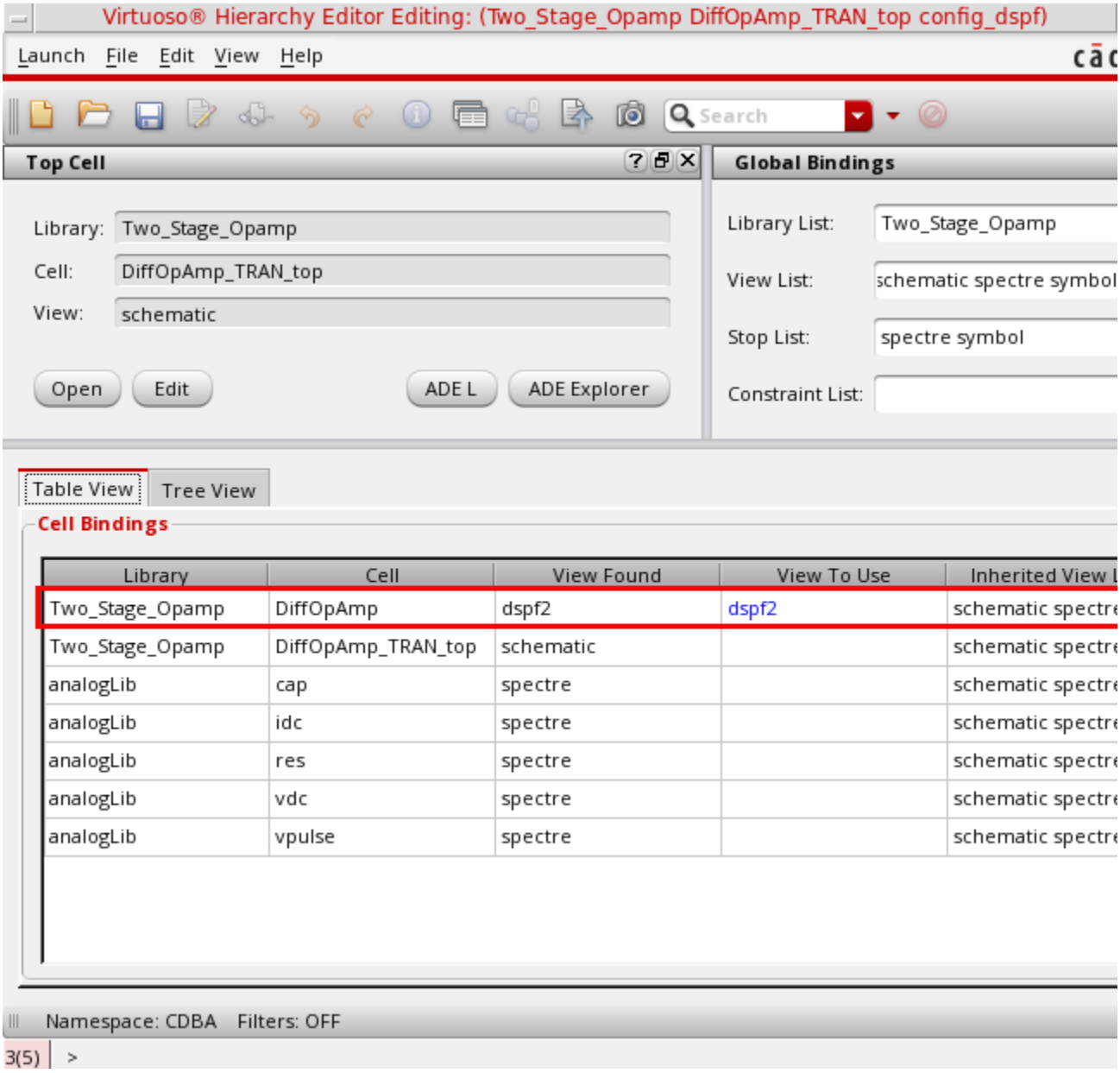
If you need to replace the schematic of a particular device in your design with the DSPF view, you can bind the cellview to the corresponding DSPF view.

To do this:

1. Create or edit the config view for its parent cellview in Virtuoso Hierarchy Editor.

© 2023 Cadence Design Systems, Inc.

2. Specify the DSPF view to be used in the *View to Use* column in the *Cell Bindings* table.



3. Save the changes in the Hierarchy Editor.

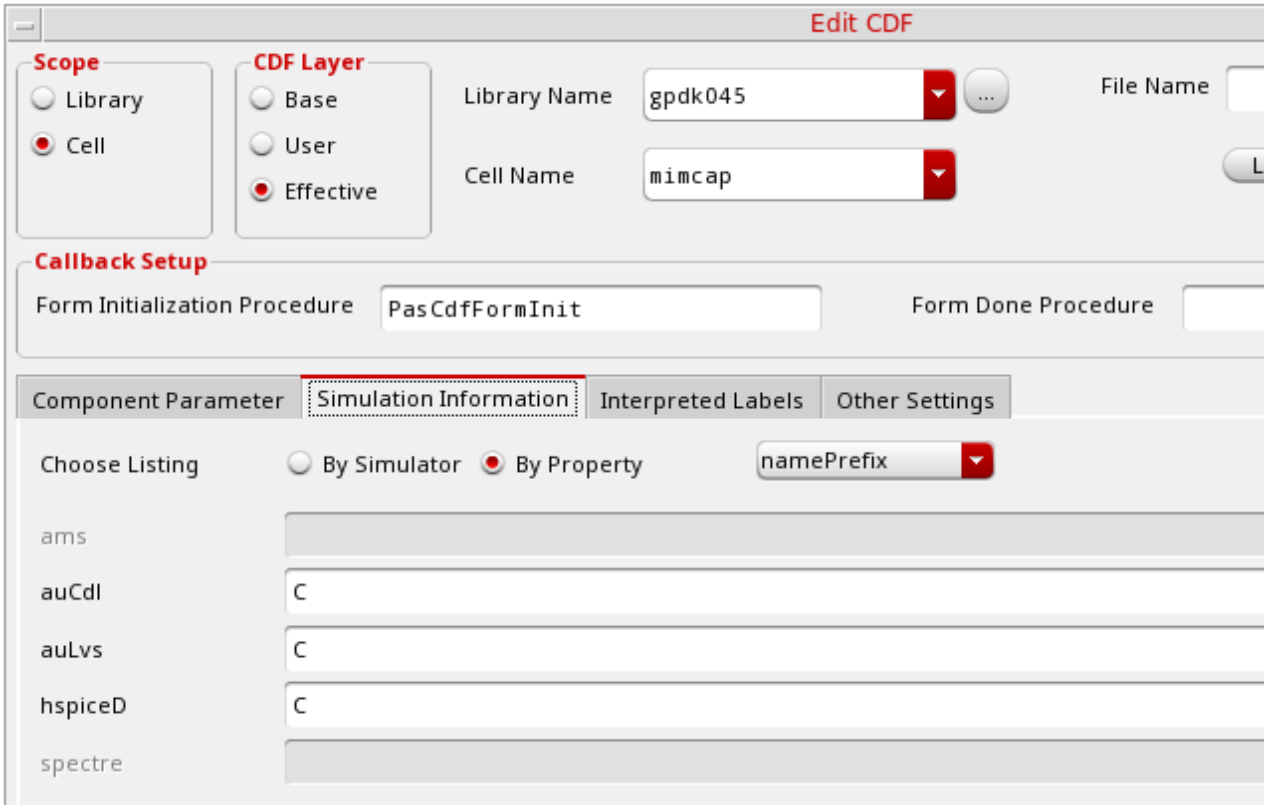
Tip

If required, you can vary the DSPF view to be used for simulation by using the CONFIG sweep feature of ADE Assembler.

© 2023 Cadence Design Systems, Inc.

Rules for Adding Prefixes to Instance Names

The netlister determines the naming prefix for instances using the `namePrefix` property set in the *Simulation Information* section of the Edit CDF form.



The `namePrefix` for instances is decided as follows:

- If the `namePrefix` is not found in the CDF *Simulation Information* section, the netlister prints `X` as the namePrefix.
- If the `namePrefix` is the same as the first character of the device name, the netlister does not add a prefix to the instance name in the netlist.

If none of these conditions are met, the netlister adds the prefix `namePrefix` in the output of the instance. You can modify the prefixes that are chosen by either using the `dspfNamePrefixSimInfo` environment variable or the `useSelfCDFInPostLayoutNetlist` environment variable.

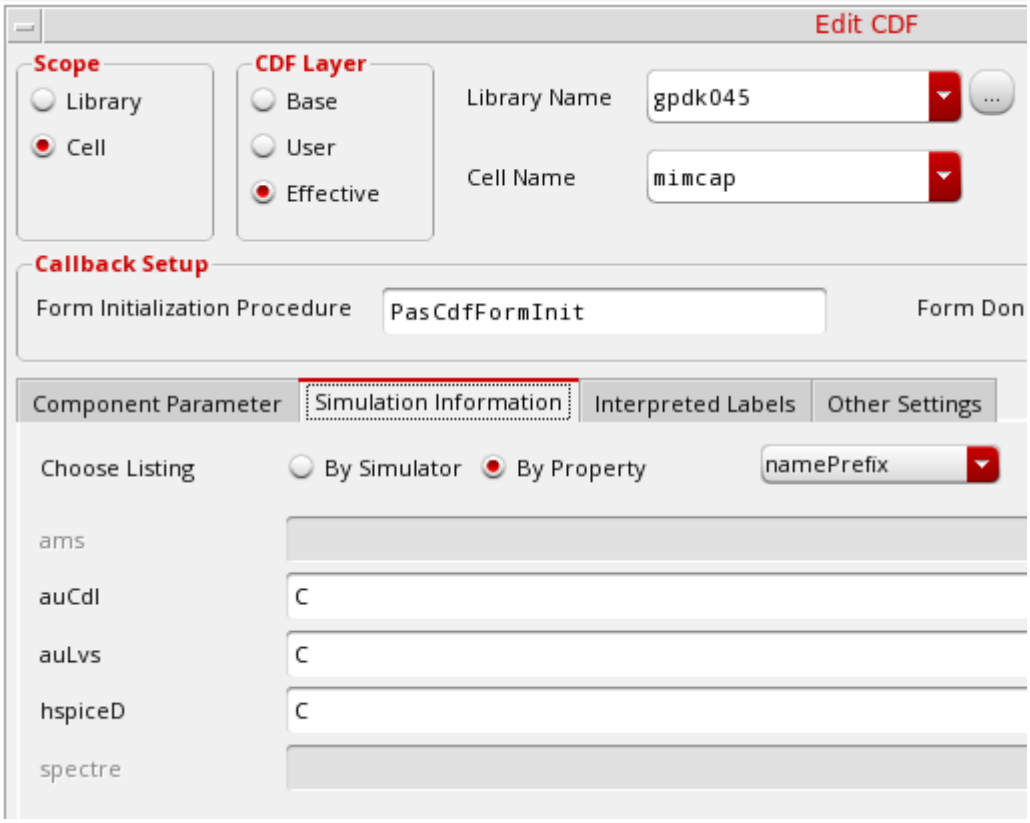
© 2023 Cadence Design Systems, Inc.

Examples

- When the `namePrefix` and the first character of the device name are the same

  The `mimcap` device is netlisted in the DSPF as `C0`, when the simulator used is *auCdl*.

```
C0 outm#64 net044#67 AVSS g45cmim w=5e-06 l=5e-06 m=1 $x=15.29 $y=25.58
```



Here, the `namePrefix` in the Edit CDF form is set to `C`, which is the same as the first character of the device name. According to the pre-defined rules, no prefix is added for `mimcap` in the DSPF file.

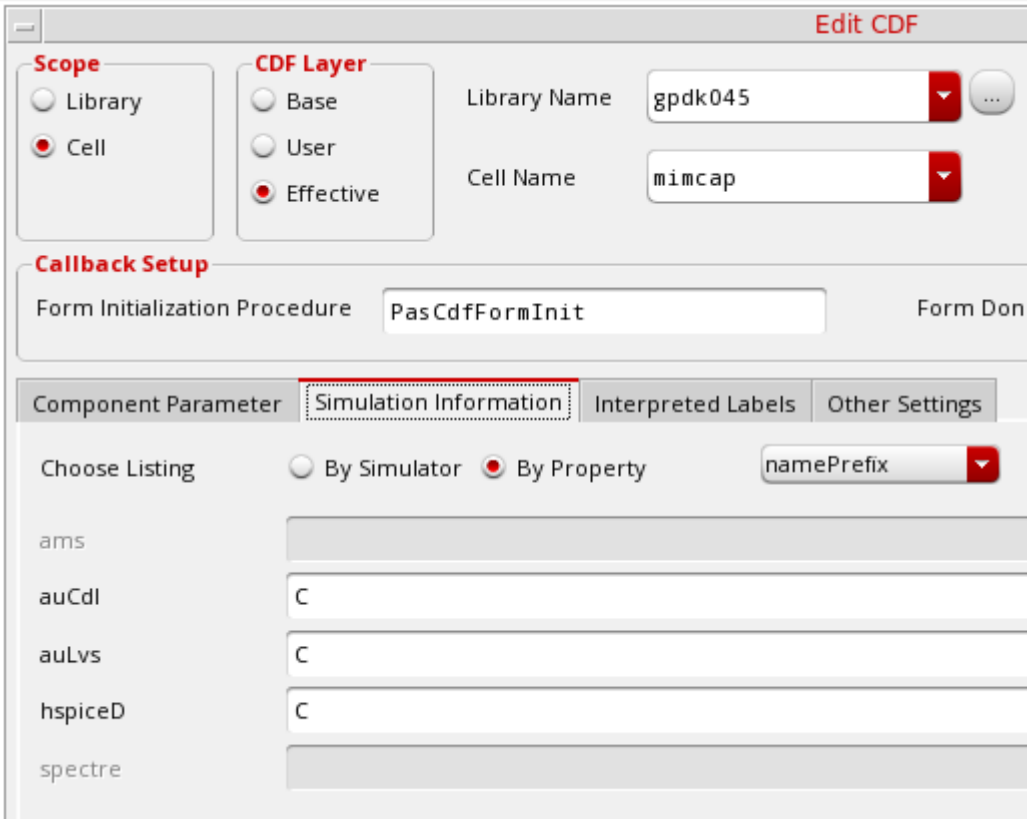
Setting `dspfNamePrefixSimInfo` to *ams* or *spectre* will add an `X` prefix and the DSPF will print the device as follows:

```
XC0 outm#64 net044#67 AVSS g45cmim w=5e-06 l=5e-06 m=1 $x=15.29 $y=25.58
```


© 2023 Cadence Design Systems, Inc.

- When `namePrefix` has no specified value

  Set the `dspfNamePrefixSimInfo` environment variable to a simulator, such as *spectre*, that does not have a `namePrefix` specified for `mimcap`.



  Based on the `namePrefix` rules, the `X` prefix is added to the devices.

```
XC0 outm#64 net044#67 AVSS g45cmim w=5e-06 l=5e-06 m=1 $x=15.29 $y=25.58
```

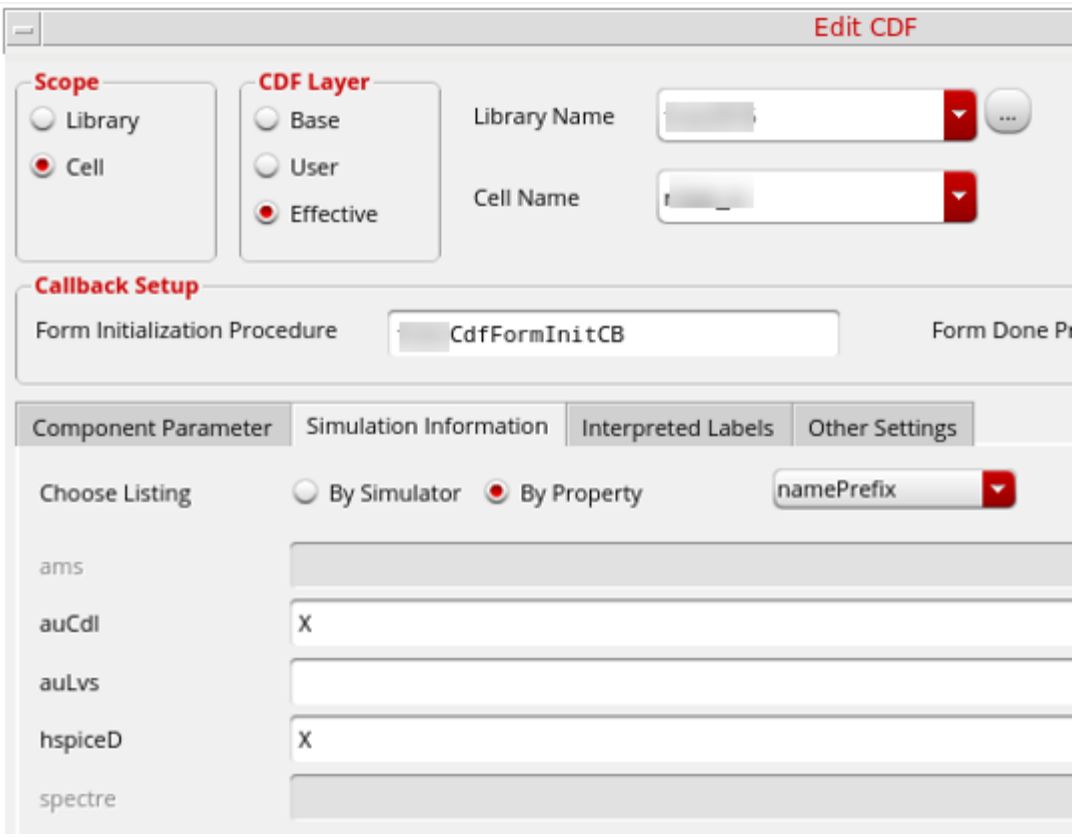

© 2023 Cadence Design Systems, Inc.

- When the device name and `namePrefix` are different

  Consider that you have resistors in the netlist as shown below:

  ```
  R14@2 (net#222 net0178#12 gnd#1330 ) rhim_m lr=3.5e-06 wr=8e-07 + multi=(1)
  $x=69.656 $y=38.422
  ```

  To netlist this correctly with an `X` prefix, set `dspfNamePrefixSimInfo` to *auCdl* because the `namePrefix` for *auCdl* is `X`.



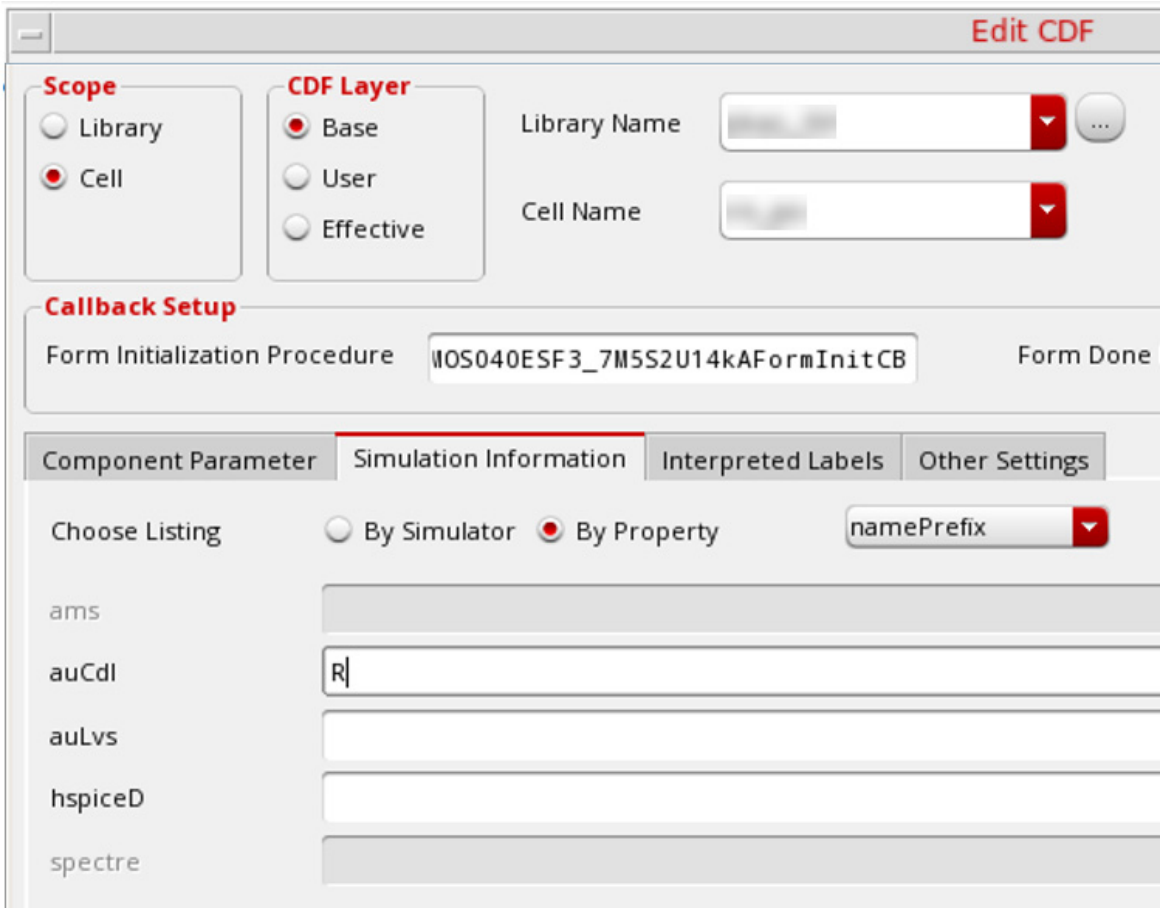

  ```
  XR14@2 (net#222 net0178#12 gnd#1330 ) rhim_m lr=3.5e-06 wr=8e-07 + multi=(1)
  $x=69.656 $y=38.422
  ```

  This is applicable in cases when the *auCdl* prefix is not the same as the first character of the device name.

© 2023 Cadence Design Systems, Inc.

- When `namePrefix` has a value for a specific simulator and none for others

  Consider that a device has the `namePrefix` set to `R` for only the *auCdl* simulator, and no value for other simulators in the Edit CDF form. On netlisting, no prefix will be added to resistors if you set `dspfNamePrefixSimInfo` to *auCdl*.

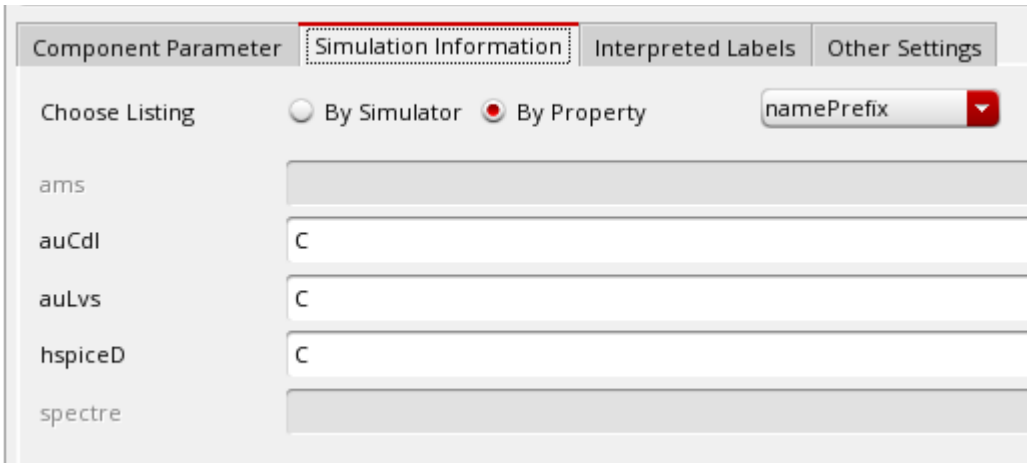


  If you set `dspfNamePrefixSimInfo` to any other simulator name, the prefix `X` will be added as per the `namePrefix` rules.

© 2023 Cadence Design Systems, Inc.

- When `namePrefix` value is not shown

  The `namePrefix` for `mimcap` is `C` for the *auCdl*, *auLvs*, and *hspiceD* simulators, and disabled for *ams* and *spectre*.



  If you set `dspfNamePrefixSimInfo` to *auCdl*, *auLvs*, and *hspiceD*, no prefix is added, but if you set the environment variable to *ams*, or *spectre*, the prefix `X` is added to the devices.

```
XC0 outm#64 net044#67 AVSS g45cmim w=5e-06 l=5e-06 m=1 $x=15.29 $y=25.58
```

Related Topics

Prefixes in Instance Names in the DSPF File

dspfNamePrefixSimInfo

useSelfCDFInPostLayoutNetlist

© 2023 Cadence Design Systems, Inc.

Customizing Netlist Creation Using the Options in .simrc

The content and format of a DSPF file is heavily dependent on the extraction tool and the settings. Often, simulation problems occur because the syntax used for the nets and terminals in the DSPF file is different from the syntax used in the netlist created by Spectre. For example, the hierarchy delimiter used in a DSPF file is '.' (`I0.M9`) whereas the delimiter used in the Spectre netlist is '/' (`I0/M9`).

Similarly, for terminal currents, mapped name of top level devices in the DSPF file can have an extra prefix, such as XX instead of X, which does not exist in the netlist.

If your DSPF files show such differences, you can set options in the `.simrc` file to update the save statement in the netlist so that the device names match with those in the DSPF file. For this, you need to read the contents of the DSPF file and set the variables in the `.simrc` file to update the name mapping.

The following table describes the options you can set in the `.simrc` file:

Variable	Description
`spfFileNameMappingFormat`	Specifies the format to be used for name mapping. Valid values:`"cdl"`, `"spectre"`. If you give any other value it is internally reset to `"cdl"`. Default value: `"cdl"`
`spfFileTermDelimiter`	Specifies the delimiter to be used to save terminal currents. This value is not taken from the DSPF because Spectre requires this to be set to a colon. Therefore, it is mandatory to set it to the default value, which `":"`.
`spfFileAddPrefixToDevice`	Specifies whether the prefixes defined by other `spf*` variables, for example, `spfFileDeviceDefaultPrefix`, `spfFileDevicePrefixForTermCurrent` and `spfFileDevicePrefixForOppoints`, are to be added to the device names printed in the save statements in the netlist. Default value: `t`

© 2023 Cadence Design Systems, Inc.

`spfFileAddContextSensitivePrefix`

Specifies how many prefixes need to be added to the names of the devices at a level below the DSPF block. This variable does not affect the name of the DSPF block.

When this variable is set to `"yes"`, the device names get a single prefix that is specified by the *namePrefix* field in the Component Description Format form or the `spfFileDeviceDefaultPrefix` variable.

When this variable is set to `"no"`, the device names get all prefixes specified by the .simrc variables or CDF properties.

Example:

A device named `I0.I1.I2.XM0*:1` is changed to one of the following depending on the value of this variable:

- `I0.I1.I2.XMM0*:1`, when this variable is set to `yes`
- `I0.I1.I2.XXMM0*:1`, when this variable is set to `no`

where the first `M` is specified by the *namePrefix* field in the Component Description Format form and `X` is specified by the `spfFileDevicePrefixForTermCurrent` variable.

`spfDevicePrefixForMimcap`

Specifies the prefix to be used to identify the top-level devices while mapping `mimcap` devices.

Default value: `"C"`

If the DSPF has `mimcap` devices, you can print these devices in the DSPF with a different prefix to the other devices, using the following definition:

```
spfDevicePrefixForMimcap "C"
```

For example, using this variable will map a capacitance, `CC0@1`, and show the following mapping in the netlist:

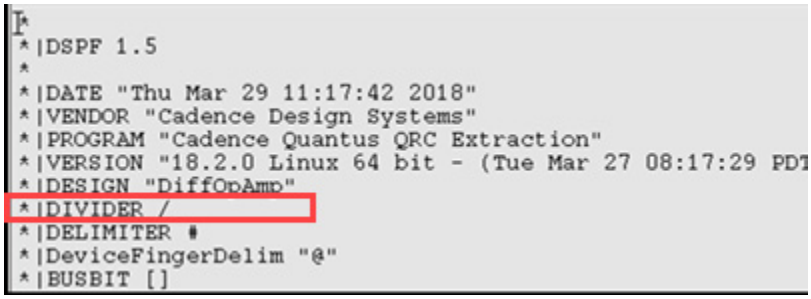
```
save I0.CC0:cap I0.CC1:cap
```


© 2023 Cadence Design Systems, Inc.

`spfFileHierDelimiter`

Specifies the delimiter to be used in hierarchical names. This variable should be set to the DIVIDER setting in the DSPF Header section.

Default value: `.`



`spfFileTerminalMapping`

Specifies the character case to be used while mapping the terminal names.

Valid values:

- `"mixed"`: Uses a mix of upper or lower case. For example, `M0#Term1`
- `"lower"`: Uses the lower case. For example, `M0#term1`
- `"upper"`: Uses the upper case. For example, `M0#TERM1`
- `"spectre"`: Uses only numbers to identify the terminals. For example, `M0#1`

Default value: `"lower"`

© 2023 Cadence Design Systems, Inc.

`spfFileFingerDelimiter`

Specifies the delimiter to be used to mention the device fingers. Set this value to the delimiter used in the DSPF file. The example shown below uses the `@` delimiter.

```
*Net Section
*
*|GROUND_NET AVSS
*
*|NET inm 4.77871e-15
*|P (inm X 0 11.4700 0.0450)
*|I (MM1@5#g MM1@5 g X 0 22.5750 15.2950)
*|I (MM1@4#g MM1@4 g X 0 21.1650 15.2950)
*|I (MM1@3#g MM1@3 g X 0 20.6950 15.2950)
*|I (MM1@1#g MM1@1 g X 0 21.6350 15.2950)
*|I (MM1#g MM1 g X 0 22.1050 15.2950)
*|I (MM1@2#g MM1@2 g X 0 23.0450 15.2950)
*|S (inm#1 11.4700 12.1250)
*|S (inm#2 20.6950 12.6350)
*|S (inm#4 21.1650 12.6350)
*|S (inm#6 21.6350 12.6350)
```

Default value: `@`

In the expressions that calculate the sum total of current for a multi-fingered device, you must use the same delimiter that is mentioned in the DSPF file.

`spfFileDevicePrefixForMacroModels`

Specifies the prefix for any macro models in the design. In some PDKs, the MOS transistors are implemented as macro models to which an X prefix is added during LVS and extraction.

To ensure that the save statements adds the X prefix so that the content saved by simrc matches the DSPF, set the `spfFileDevicePrefixForMacroModels` variable to `X`.

Default value: `X`

`spfFileDeviceDefaultPrefix`

Specifies the default device prefix to be used when no prefix is specified in the *namePrefix* field in the *simInfo* section of the Component Description Format form, but a prefix is required to correctly identify a device in the DSPF.

Default value: `nil`

`spfFileNetMapping`

Specifies the character case to be used while mapping the terminal names.

Valid values: `"mixed"`, `"lower"`, `"upper"s`

© 2023 Cadence Design Systems, Inc.

`spfFileDevicePrefixToOverrideCDF`	Overrides the value specified in the *namePrefix* field in the *simInfo* section of the Component Description Format form. Default value: `X`
`spfFileDevicePrefixForTermCurrent`	Specifies the prefix to be used to identify the top level devices while mapping the terminal currents. For terminal currents the mapped name of the top level devices in the DSPF file can have an extra prefix, such as `XX` instead of `X`. Set this variable to the prefix or to `nil` if no extra prefix is required. Default value: `X` **Note:** This variable is applicable to MOS devices only.
`spfFileDevicePrefixForOppoints`	Specifies the prefix to be used to identify the top level devices for operating points. Set this variable to the prefix or to `nil` if no extra prefix is required. Default value: `nil` **Note:** This variable is applicable to MOS devices only. For example, if the saved device names in the netlist are prefixed with `X`, as shown below, set this variable to `"X"`. `save I0.XMM4:ids`
`spfFilePrefixForBoxedDevice`	Specifies the prefix to be assigned to the devices that are considered to be boxed, that is a DSPF embedded inside a DSPF, by the extractor. The names of such boxed devices are specified by the `spfFileBoxedDeviceList` variable in the `.simrc` file. In the netlist saved for simulation, the names of the boxed devices are printed with the assigned prefix. You can probe only the terminals of such boxed subcircuits or devices, not the objects inside them. Default value: `X`

© 2023 Cadence Design Systems, Inc.

`spfFileBoxedDeviceList`	Specifies a list of names of devices to be considered as boxed. These devices do not get a prefix in the netlist. While creating netlist, the names of these boxed devices are assigned a prefix specified by the `spfFilePrefixForBoxedDevice` variable. Default value: `nil`

The following example of the .simrc file shows how you can specify these settings:

```
dspfFileEnvOptions = '(
        (nil spfFileNameMappingFormat "cdl"
            spfFileTermDelimiter ":"
                spfFileTerminalMapping "lower"
                spfDevicePrefixForMimcap "C"
                spfFileFingerDelimiter "@"
                spfFileDeviceDefaultPrefix "M"
                spfFileHierDelimiter "/"
                spfFileDevicePrefixForMacroModels "X"
                spfFileDevicePrefixToOverrideCDF "X"
                spfFileDevicePrefixForTermCurrent "X"
                spfFileDevicePrefixForInstCurrent " "
                spfFileNetMapping "lower"
                spfFileBoxedDeviceList ("devicename1" "devicename2")
                spfFilePrefixForBoxedDevice "XB"

))
```

Caution

It is recommended not to specify the following options in the .simrc file to be used for netlist customization when using DSPF files:

- `spfAlwaysAddPrefixWithInstanceNames`
- `spfSimAlwaysAddPrefixInInstName`
- `simAlwaysAddPrefixInInstName`

© 2023 Cadence Design Systems, Inc.

Customizing Netlist Creation Using Environment Variable

When using DSPF files, you can also use the saveFingersForOpParams and sumDeviceFingerCurrents environment variables to customize the save statement in the netlist.

You can use the dspfNamePrefixSimInfo environment variable to specify the simulator from which the `namePrefix` value is used to modify prefixes for device names in a DSPF file. To use the `namePrefix` from the current simulator, you can use the useSelfCDFInPostLayoutNetlist environment variable.

© 2023 Cadence Design Systems, Inc.

9

Netlisting

The environment generates the following files for simulation:

File	Description
Netlist	Contains component and design connectivity information only
Simulator input file	Contains the netlist and simulator control information

Note: The environment passes the simulator input file to your configured simulator.

ADE Assembler creates or updates the simulator input file automatically when you give the command to run a simulation. The environment creates hierarchical netlists incrementally: incremental netlisting is faster than full hierarchical netlisting because the environment updates the netlist only for those schematics that have changed since it created the previous netlist.

Alternatively, you can generate a netlist and simulator input file when

- You want to run the simulator in stand-alone mode
- You want to modify the netlist to take advantage of features that the design environment interface to your simulator does not support
- You want to read the netlist before running the simulation

See the following topics for more information:

- Creating a Netlist on page 346
- Displaying a Netlist on page 346
- Expanding Hierarchy to Netlist a Design on page 347

© 2023 Cadence Design Systems, Inc.

Creating a Netlist

➡ To create and display the simulation input file, which contains the netlist, do the following:

On the Data View assistant pane, right-click the test or analysis name and choose *Netlist – Create*. In this case, ADE Assembler updates the netlist only for those cells that have changed since the previous netlist was created. Netlist is not created for the cells that are not checked and saved. Instead, their existing netlist is used. The netlist appears in a text browser window.

Note: If you include DSPF files in your simulation setup, you can set options in the `.simrc` file to update the save statement in the netlist according to the format and content of the DSPF files, which are dependent on the extraction tool used to create the files. For more details, refer to Customizing Netlist Creation Using the Options in .simrc.

➡ To recreate the netlist, choose *Netlist – Recreate*. In this case, ADE Assembler runs full hierarchical netlisting for all the cells or schematics in the hierarchy.

Important

If the `nlReNetlistAll` variable is set to `t` in the `.simrc` file, the command to create a netlist always recreates it.

Displaying a Netlist

To display an existing simulator input file, do one of the following:

➡ On the Data View assistant pane, right-click the test and choose *Netlist – Display*.

➡ On the Results tab of the Outputs pane, right-click a test and choose *Netlist – Display*.

© 2023 Cadence Design Systems, Inc.

Expanding Hierarchy to Netlist a Design

While netlisting a hierarchical design, the environment expands every cell (instance) into lower level cells until it reaches one designated as a primitive. The environment adds each primitive to the netlist.

At each level in your design hierarchy, you can have one or more views for each cell. You use a view list to specify which view the design environment selects for expansion during netlisting. View lists can be global to the entire design or specific to an instance as specified by its property values.

For analog simulation, you specify the global or default view list in the *Switch View List* field of the Environment Options form when the selected cellview is not a configuration (config) view. If an instance does not have any of the views listed in the switch view list, the netlister reports an error.

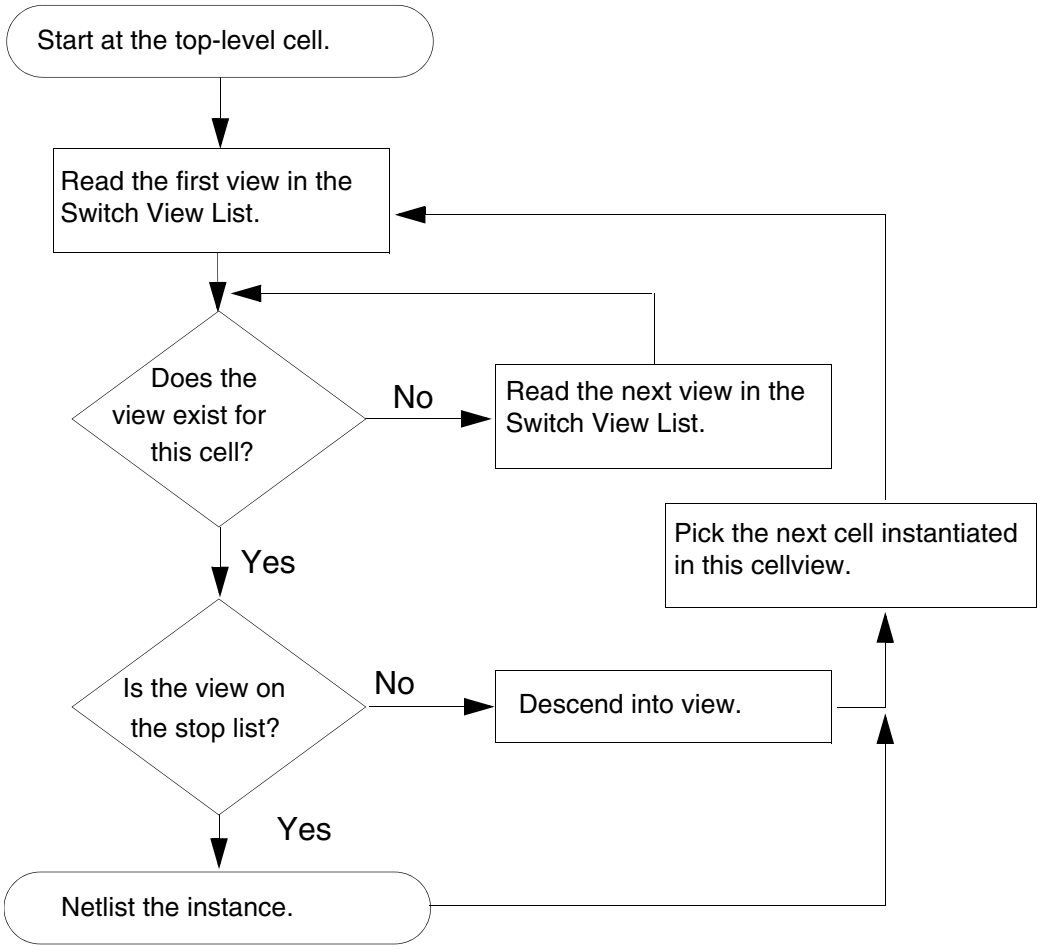
The netlister uses the stop list, which you specify in the *Stop View List* field on the Environment Options form, to identify primitives. When the netlister reaches a view that is specified in both the switch and the stop list, it netlists the instance and does not expand beyond this level.

If you are using DSPF subcircuits that contain blackboxes, the netlister traverses inside each blackbox device and uses the schematic name corresponding to each object to write the save statement in the netlist. This helps in probing the device terminals or objects inside the blackbox devices.

Note: Parasitic simulation and mixed-signal simulation use different processes for creating switch and stop view lists. See the *Virtuoso Parasitic Aware Design User Guide* and the Virtuoso Mixed-Signal Circuit Design Environment User Guide for more information.

© 2023 Cadence Design Systems, Inc.

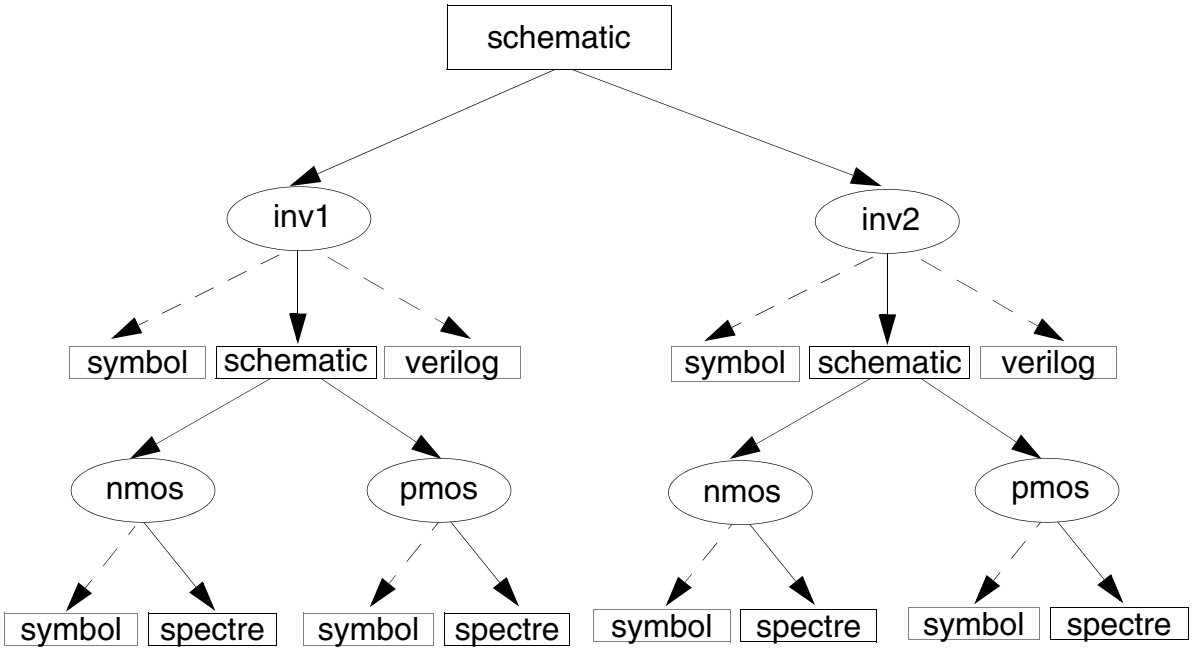
The following flowchart shows how a typical OSS-based netlister netlists a design.



© 2023 Cadence Design Systems, Inc.

The following figure illustrates how hierarchy expansion is performed on a simple design. The solid lines show the view selection and design expansion based on the switch view list and stop view list shown.

Switch View List: spectre schematic cmos.sch verilog
Stop View List: spectre verilog



© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

10

Selecting Data to Save and Plot

From the Outputs Setup tab of the Outputs pane, you can specify nets, terminals, and measurements you want to save and plot. You can save all node voltages and terminal currents or specify a set of voltages and currents you want to save. You can select output nodes and terminals on your schematic and build expressions using the expression builder to analyze or measure particular results.

Each item that appears on the Outputs Setup tab has a *Plot* check box and a *Save* check box. You can select the check box for those outputs you want to plot or save, or deselect the check boxes of those items you do not want to plot or save.

See the following topics for more information:

- Opening the Outputs Setup Tab on page 353
- Selecting Output Signals on the Schematic on page 355
- Specifying Whether a Result Will Be Saved or Plotted on page 366
- Adding or Modifying an Output Expression on page 367
- Creating calcVal Expressions Using calcVal Wizard on page 372
- Creating Dependent Expressions on page 381
- Creating a Combinatorial Expression on page 385
- Creating Expressions to be Measured Across Dimensions on page 389
- Using MATLAB Expressions in Outputs on page 412
- Loading an OCEAN or a MATLAB Measurement on page 430
- Saving Operating Point Parameters on page 434
- Copying Outputs on page 442
- Adding User-Defined Columns in the Outputs Setup Tab on page 444
- Exporting Outputs to a CSV File on page 448

© 2023 Cadence Design Systems, Inc.

- Importing Outputs from a CSV File on page 450
- Configuring How Outputs Appear on the Outputs Setup Tab on page 452
- Filtering and Searching in Outputs Setup Pane on page 455
- Selecting Outputs to Save or Plot on page 459
- Removing Outputs on page 459
- Saving All Voltages or Currents on page 461

© 2023 Cadence Design Systems, Inc.

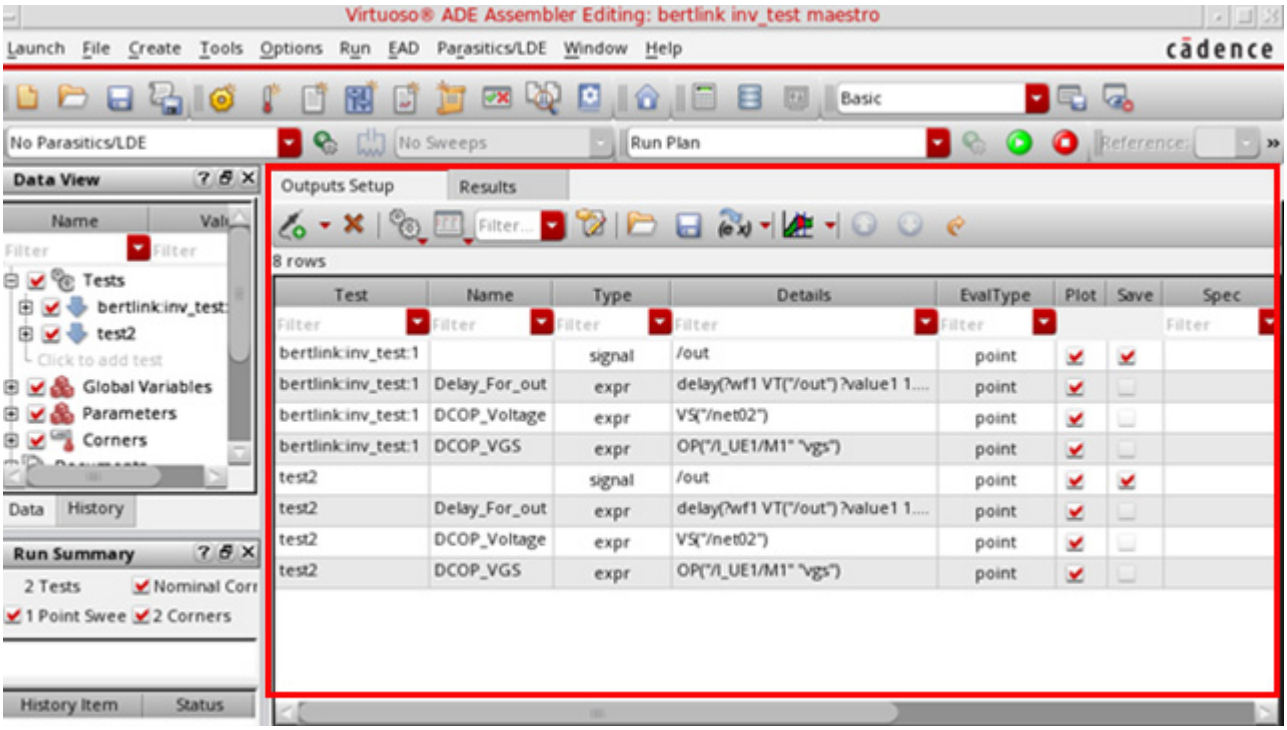
Opening the Outputs Setup Tab

You can use the Outputs Setup tab of the Outputs pane to add a new output signal/ expression/OCEAN or MATLAB script or to change an existing output.

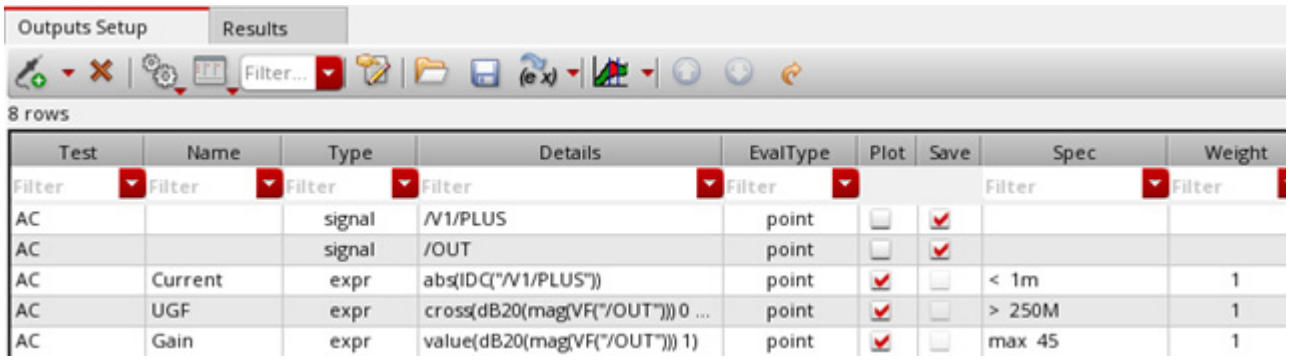
In a new test, the Outputs pane is not visible by default. To open the *Outputs Setup* tab, do the following:

➤ On the Welcome to ADE Assembler pane, click the hypertext link below *Outputs*.

The Outputs pane appears displaying the *Outputs Setup* tab.



On this tab, you can specify the outputs to be saved or plotted.



© 2023 Cadence Design Systems, Inc.

See the following topics for more information:

- Selecting Output Signals on the Schematic on page 355
- Adding or Modifying an Output Expression on page 367
- Specifying Whether a Result Will Be Saved or Plotted on page 366
- Using MATLAB Expressions in Outputs on page 412
- Loading an OCEAN or a MATLAB Measurement on page 430
- Saving Operating Point Parameters on page 434
- Copying Outputs on page 442
- Adding User-Defined Columns in the Outputs Setup Tab on page 444
- Configuring How Outputs Appear on the Outputs Setup Tab on page 452
- Selecting Outputs to Save or Plot on page 459
- Removing Outputs on page 459

© 2023 Cadence Design Systems, Inc.

Selecting Output Signals on the Schematic

You can select outputs on the schematic and specify whether you want to save or plot them.

To select an output on the schematic, do the following:

Note: This procedure allows you to select only one output on the schematic. If you want to select multiple outputs on the schematic at a time, see Selecting Outputs to Save and Selecting Outputs to Plot.

1. On the Outputs Setup tab, click .
2. In the drop-down list, select a test and choose *Signal*.

   A new row is added for the test with the output type *signal*.

Tip

You can double-click the test name or output type to select a different test or output type from the drop-down list that appears in the *Test* or *Type* column.

3. (Optional) In the *Name* field, type the name for the output.

   This name appears in the Waveform window.
4. Double-click the *Details* field and click the ellipsis button.

   The schematic window appears in a tab.
5. In the schematic, select a signal.

   The signal name appears in the *Details* field on the Outputs Setup tab.

   Alternatively, you can manually enter a signal. In this case, the type of data to be saved—current, voltage, or both— is decided through the terminal selection type specified in the Printing/Plotting Options form.

   Note: By default, the current data is saved for all selected terminals. However, you can choose to save or plot voltage data, or both current and voltage. For more details, refer to Specifying the Type of Data for Signal Outputs.
6. (Optional) You can select or deselect either or both of the following check boxes:
   - *Plot* — The program plots the specified signal output after simulation.
   - *Save* — The program saves the signal output to the simulation results file.

© 2023 Cadence Design Systems, Inc.

Related Topics

defaultOutputColumns

Specifying the Type of Data for Signal Outputs

Selecting Outputs to Save

Selecting Outputs to Plot on page 360

Selecting Nodes, Nets, and Terminals on page 361

Selecting a Digital Bus or a Bundle of Buses on page 363

Searching for Conditional Results on page 1243

Viewing and Editing Output Signals or Expressions in Calculator

© 2023 Cadence Design Systems, Inc.

Specifying the Type of Data for Signal Outputs

Important

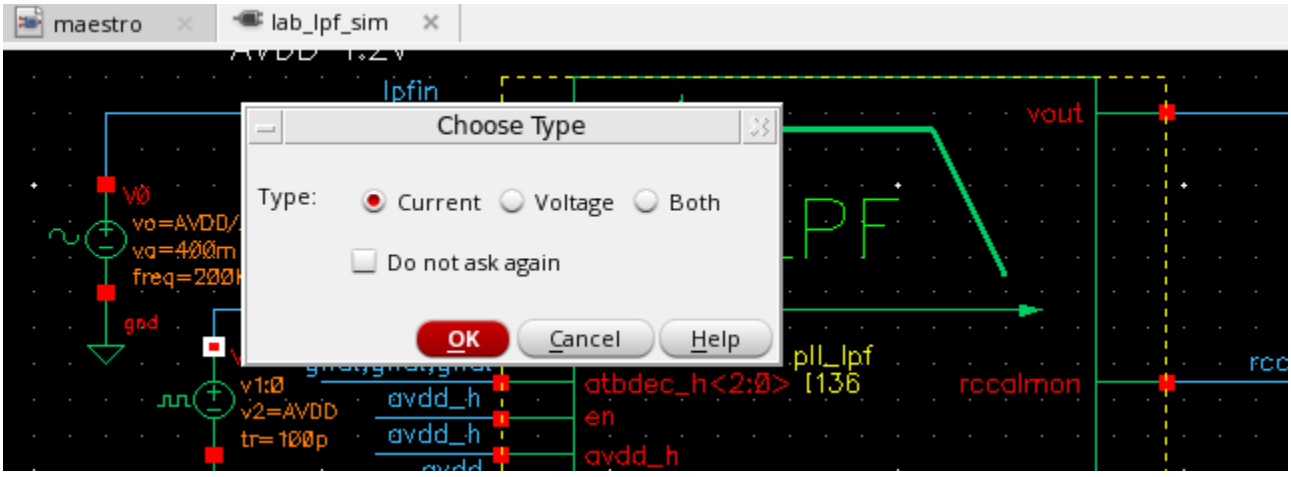
This feature is not supported for the AMS simulator.

By default, ADE Assembler saves the current values for all selected terminals, but you can choose to save or plot voltage data, or both current and voltage.

To control the type of data to be saved for the selected terminals, do one of the following:

- Modify the *Terminal Selection Type* drop-down list on the ADE Assembler Plotting/ Printing Options form.
- Modify the `terminalSelectionType` environment variable, which is by default set to `"current"`.

Using the two ways specified above, you can choose to save either current, voltage, or both types of data for all selected terminals. If you need flexibility to choose the type of data to be saved while selecting a terminal, you can set these settings to `"prompt"`. This setting displays the Choose Type message prompt for every terminal output selection. In this message prompt, you can select the type of data to be saved or plotted for the output.

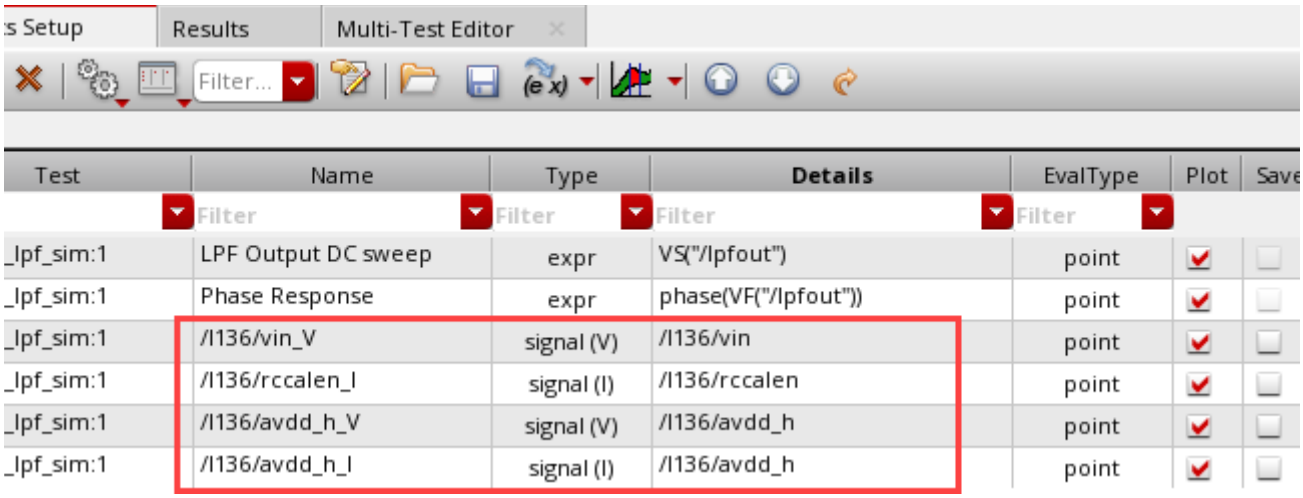


Depending on your selection, ADE Assembler saves the desired type of data. The data type of a signal output is indicated by its name and output type. For signals with current type of data, ADE Assembler adds the `_I` suffix in the name and (I) in the output type. Similarly, it adds the `_V` suffix in the name and (V) in the output type for the outputs that save the voltage data.

If you choose to save both, ADE Assembler creates two outputs, one for current and the other for voltage.

© 2023 Cadence Design Systems, Inc.

The following example shows the signal outputs created for all of these choices.

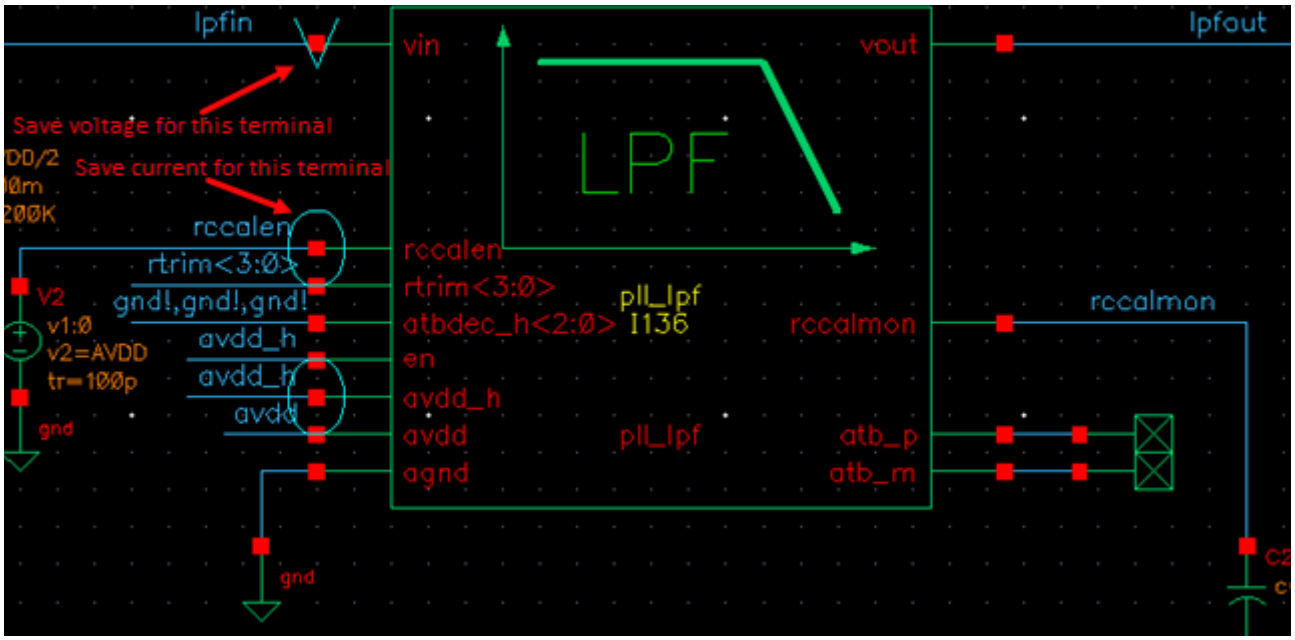


Test	Name	Type	Details	EvalType	Plot	Save
	Filter	Filter	Filter	Filter		
_lpf_sim:1	LPF Output DC sweep	expr	VS("/lpfout")	point	☑	☐
_lpf_sim:1	Phase Response	expr	phase(VF("/lpfout"))	point	☑	☐
_lpf_sim:1	/I136/vin_V	signal (V)	/I136/vin	point	☑	☐
_lpf_sim:1	/I136/rccalen_I	signal (I)	/I136/rccalen	point	☑	☐
_lpf_sim:1	/I136/avdd_h_V	signal (V)	/I136/avdd_h	point	☑	☐
_lpf_sim:1	/I136/avdd_h_I	signal (I)	/I136/avdd_h	point	☑	☐

Indicators are also added to the schematic view, as described below.

- For the signals saving the current data, the terminal is **circled** in the schematic view.
- For voltage data, the terminal is **marked with a V**.
- For signals saving both current and voltage, both types of indicators are used.

The following figure shows two examples.

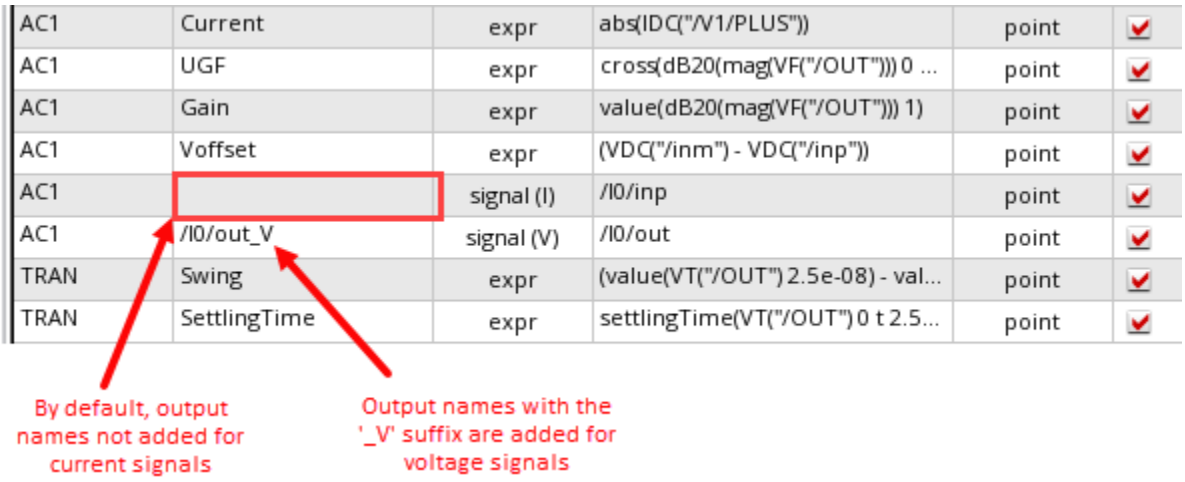


© 2023 Cadence Design Systems, Inc.

The revised output names and types, and the visual indicators on schematic are helpful in identifying the purpose of signal outputs in maestro views as well as in plotted graphs.

Important Points to Note

- Any change in the value of `terminalSelectionType` is reflected in the *Terminal Selection Type* drop-down list on the ADE Assembler Plotting/Printing Options form. However, if you update the value of the *Terminal Selection Type* drop-down list on the form, the corresponding `.cdsenv` setting is ignored in the current Virtuoso session.
- To keep consistency with the previous versions, by default, no output names are added for the terminals selected to save current data when `terminalSelectionType` is set to `"current"`. However, when this variable is set to `"voltage"`, an output name is added with the `_V` suffix for outputs saving voltage data, as shown below.



AC1	Current	expr	abs(IDC("/V1/PLUS"))	point	☑
AC1	UGF	expr	cross(dB20(mag(VF("/OUT"))) 0 ...	point	☑
AC1	Gain	expr	value(dB20(mag(VF("/OUT"))) 1)	point	☑
AC1	Voffset	expr	(VDC("/inm") - VDC("/inp"))	point	☑
AC1		signal (I)	/I0/inp	point	☑
AC1	/I0/out_V	signal (V)	/I0/out	point	☑
TRAN	Swing	expr	(value(VT("/OUT") 2.5e-08) - val...	point	☑
TRAN	SettlingTime	expr	settlingTime(VT("/OUT") 0 t 2.5...	point	☑

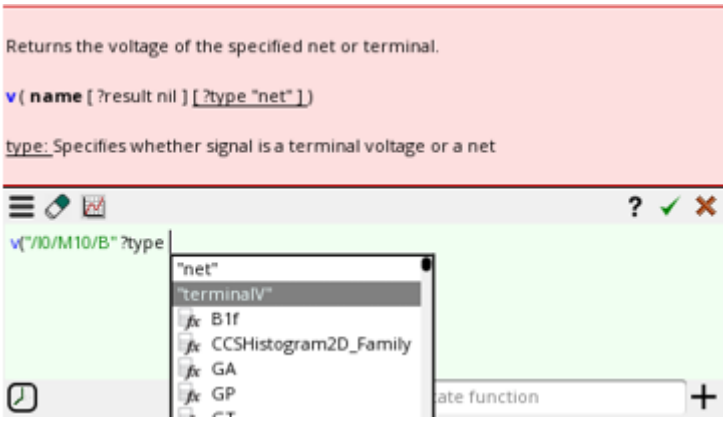
You can set the `alwaysNameTerminalOutputs` environment variable to `t` to always add the `_I` suffix to the name of outputs saving current data, as shown below.

AC1	Current	expr	abs(IDC("/V1/PLUS"))	point
AC1	UGF	expr	cross(dB20(mag(VF("/OUT"))) 0 ...	point
AC1	Gain	expr	value(dB20(mag(VF("/OUT"))) 1)	point
AC1	Voffset	expr	(VDC("/inm") - VDC("/inp"))	point
AC1	/I0/out_V	signal (V)	/I0/out	point
AC1	/I0/inp_I	signal (I)	/I0/inp	point
TRAN	Swing	expr	(value(VT("/OUT") 2.5e-08) - val...	point

- Similar to other common variables, the value of `terminalSelectionType` is also shared by ADE Explorer and ADE Assembler.

© 2023 Cadence Design Systems, Inc.

- The output type and name of signal outputs are changed in the existing maestro views too.
- To probe terminal voltage from an expression, set the `?type` argument of the `v` calculator function to `"terminalV"`, as shown below.



Selecting Outputs to Save

To select outputs to be saved, do the following:

1. On the Outputs Setup tab of the *Outputs* pane, right-click a test and choose *To be saved*.

   The schematic window appears in a tab.

2. On the schematic, select one or more nodes, nets, or terminals.

   Note: You can set the terminalSelectionType environment variable to define the default type of data to be saved for each signal output selected on the schematic. When the variable is set to `"prompt"`, the Choose Type form is displayed for each selected terminal. For more details, refer to Specifying the Type of Data for Signal Outputs.

3. Press the *Esc* key when you are done.

   The name of each selected item appears (such as */out* or */I2/PLUS*) in different rows in the Outputs Setup tab. The *Save* check box is selected.

Selecting Outputs to Plot

To select outputs to be plotted, do the following:

© 2023 Cadence Design Systems, Inc.

1. On the Outputs Setup tab of the *Outputs* pane, right-click a test and choose *To be plotted*.

   The schematic window appears in a tab.

2. On the schematic, select one or more nodes, nets, or terminals.

   Note: You can set the terminalSelectionType environment variable to define the default type of data to be saved for each signal output selected on the schematic. When the variable is set to `"prompt"`, the Choose Type form is displayed for each selected terminal. For more details, refer to Specifying the Type of Data for Signal Outputs.

3. Press the *Esc* key when you are done.

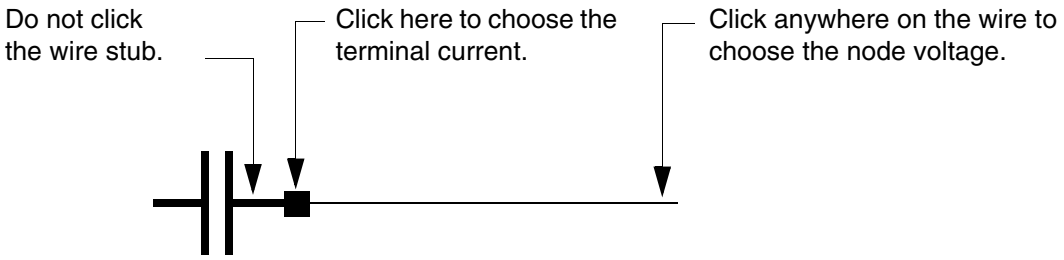
   The name of each selected item appears (such as */out* or */I2/PLUS*) in different rows in the Outputs Setup tab. The *Plot* check box is selected.

   Note: For nets, both the *Plot* and *Save* check boxes are selected.

   The program plots the selected items in a waveform window at the end of the simulation.

See also Searching for Conditional Results on page 1243.

Selecting Nodes, Nets, and Terminals



A circle appears around each pin when you choose a terminal and wires appear highlighted when you choose a net. You can alternate between selecting a terminal and selecting the wire to which it is connected.

You can

- Click an instance to choose all instance terminals.
- Click the square pin symbols to choose currents.
- Click wires to choose voltages.
- Click and drag to choose voltages by area.

© 2023 Cadence Design Systems, Inc.

- Select nodes and terminals on lower-level schematics by doing the following:
  1. In the schematic window, choose *Design – Hierarchy – Descend Edit*.
  2. Select a hierarchical instance.

     The Descend form appears.
  3. Click *OK*.

     The lower-level schematic appears.
  4. In the schematic window, select one or more nodes, nets, or terminals.

© 2023 Cadence Design Systems, Inc.

Selecting a Digital Bus or a Bundle of Buses

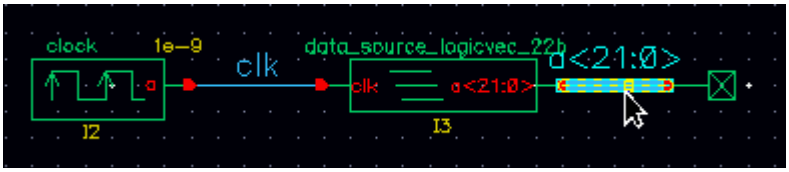
In the Outputs Setup pane, you can specify a digital bus output to be saved or plotted. The digital bus output can include a single bit or multiple bits bundled together in one output.

To add a single bit or a bundle of bits from a bus as outputs in the ADE Assembler *Outputs Setup* pane, perform the following steps:

1. Add a new output row of type `signal`. Then double-click in the *Details* column and click the ellipses button.

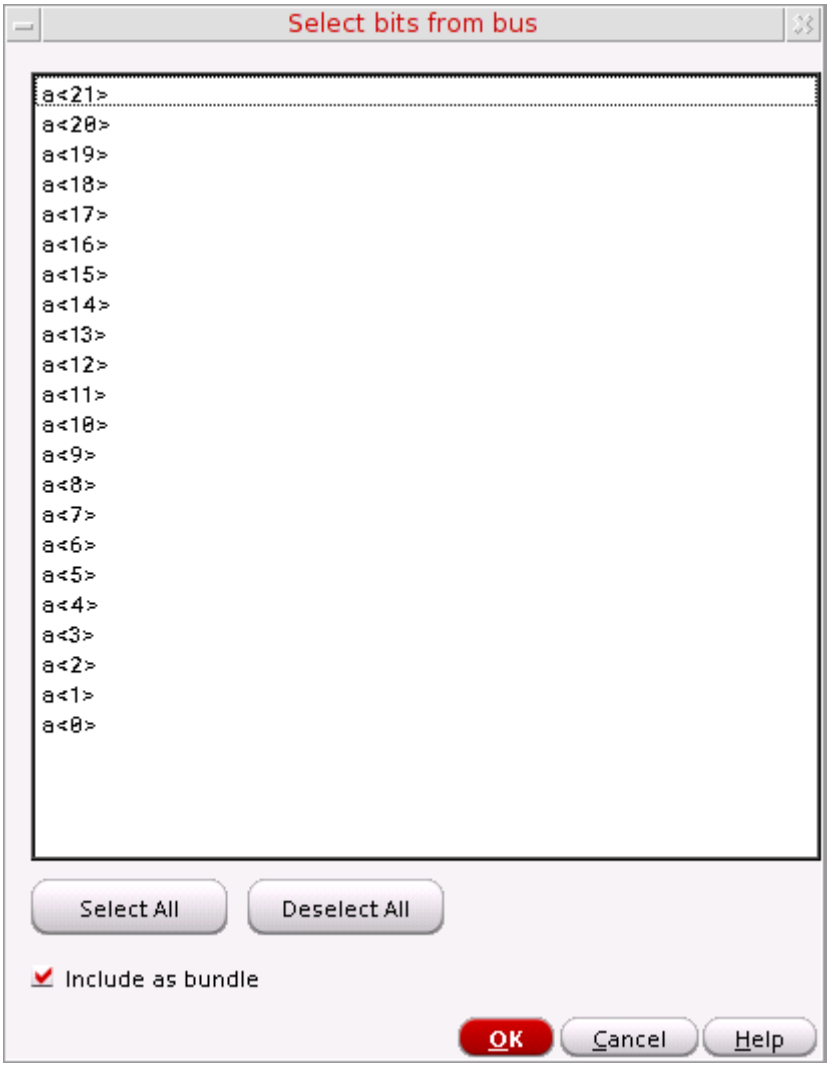
   Alternatively, right-click the output row and choose *To Be Saved* or *To Be Plotted.* The schematic of the design is opened in a new tab.

2. In the schematic, click the bus.



© 2023 Cadence Design Systems, Inc.

The Select bits from bus form appears.



Perform one of the following steps to specify bits from the bus as outputs to be saved or plotted:

- To add a single bit as an output, select the bit and click *OK*.
- To add multiple bits bundled together as an output,
  - Press the `Ctrl` key and select the bits.
  - Ensure that the *Include as bundle* check box is selected and click *OK*.

  The selected bits appear in one row in the *Outputs Setup* pane.

  Note: If the *Include as bundle* check box is not selected while you are adding or

© 2023 Cadence Design Systems, Inc.

editing an output, an error message is displayed.

- To add multiple bits to individual outputs, right-click the output row and choose *To Be Saved* or *To Be Plotted.* The schematic of the design is opened in a new tab.

  In the schematic, click the bus to open the Select bits from bus form and perform the following steps:

  - Press the `Ctrl` key and select the bits.
  - Clear the *Include as bundle* check box and click *OK*.

  Each selected bit appears in a different row in the *Outputs Setup* pane.

  You can also click the *Select All* button to select all the available bits. Or use the *Deselect All* button to clear the selection.

The following image shows how the selected bits appear as outputs in the *Outputs Setup* pane in ADE Explorer.

Name	Type	Details	Plot	Save	Spec
	signal	/a<6>	☑	☑	
	signal	/a<0,6:1>	☑	☐	
	signal	/a<15,11,5,1,17>	☑	☑	
	signal	/a<8,2>	☑	☐	
	signal	/a<21:19,8:5>	☑	☑	
	signal	/a<9>	☐	☑	
	signal	/a<14>	☐	☑	

3. In the *Name* field on the *Outputs Setup* pane, type the name for the signal output. This name appears in the graph window.
4. You can click or clear the *Plot* or *Save* check box corresponding to the output as required.
5. If you have used the *To Be Saved* or *To Be Plotted* command to open the schematic, press the `Esc` key to end the selection.

© 2023 Cadence Design Systems, Inc.

Specifying Whether a Result Will Be Saved or Plotted

You can specify whether an item will be saved to the results database or plotted in a waveform window on the Outputs Setup tab of the Outputs pane by selecting or clearing the *Plot* or *Save* check box for the item. Selecting the *Plot* check box causes the selected item or items to appear in a waveform window at the end of the simulation.

Note: If none of the outputs specified for a test are selected for plotting, the following warning message is displayed in the CIW:

```
Following tests do not have any outputs selected for plotting:
<testname1>
<testname2>
...
```


© 2023 Cadence Design Systems, Inc.

Adding or Modifying an Output Expression

You can use the Outputs Setup tab of the Outputs pane to add or modify an output expression. You can type an output expression directly in the *Details* field or use the expression builder to build an output expression.

Video

You can view a video demonstration for this feature at Expression Builder in ADE Assembler & Explorer.

To add an output expression, do the following:

1. On the Outputs Setup tab, click .
2. In the drop-down list, select a test and choose *Expression* from the sub menu.

   A new row is added for the test with the output type *expr*.

Tip

You can double-click the test name or output type to select a different test or output type from the drop-down list that appears in the *Test* or *Type* column.

3. (Optional) In the *Name* field, type a name for the expression.

   This expression name appears in the Virtuoso Visualization and Analysis XL window.
4. (Optional) The default value in the *EvalType* column is `point`. This signifies that by default the expression will run for the given data point. If you want to run the expression for all corners across the design point, double-click in the *EvalType* cell and select `corners` from the pull-down list.

   When you set the *EvalType* as `corners`, the color of the row is changed to blue. For more details, refer to Creating Expressions to be Measured Across Dimensions.

   Similarly, you can choose other values in *Eval Type* to evaluate expressions across sweeps, all dimensions, or any specific dimension.
5. Double-click the *Details* field.
6. Type an output expression directly in the *Details* field or use the expression builder to build an output expression. To add the output expression using the expression builder, do the following:
7. Click *Open expression builder* ( ).

   The expression builder appears.

© 2023 Cadence Design Systems, Inc.

Build an output expression in the space given on the expression builder. For example, you can build an expression for the 3 dB point of an output signal as shown below.

Note: To refer to the temperature value in an output expression or specification, use `VAR("temp")` instead of `VAR("temperature")`. Even if the temperature for a particular test is specified using the `temperature` design variable, you can access it as `VAR("temp")` in the results.

8. Click ✓ on the expression builder to copy the expression to the *Details* field of the output.

   The expression appears in the *Details* field. If the expression uses a signal or net that does not already exist in the outputs table, a new row for an output of type `Implicit` is added to the table. For more details, see Viewing Implicit Signals in Outputs.

   Note: While performing the division of two integer or floating point values, if denominator is greater than numerator, the division results are truncated. For example, if you perform the division of 1/2, the int function displays the result as 0 instead of 0.5.

   For more information about how the integer and floating point division is performed using SKILL, see Integer vs. Floating-Point Division in the Arithmetic and Logical Expressions chapter of the *Cadence SKILL Language User Guide*.

9. (Optional) Change the *Plot* and *Save* options for the expression, if required. See also Specifying Whether a Result Will Be Saved or Plotted on page 366.

10. (Optional) In the *Spec* and *Weight* columns next to the expression, define the specification and the weighting factor for the specification for the output expression. For more information, see Defining a Specification.

11. (Optional) In the *Units* column next to the expression, specify the unit value to be used for displaying the measured results for the expression in the Results tab of the Outputs pane.

    For example, if you specify the unit value as `dB` for an expression named `Gain`, the measured results for the expression is displayed with the unit value `dB` in the Results tab of the Outputs pane.

© 2023 Cadence Design Systems, Inc.

12. (Optional) In the *Digits* column next to the expression, specify the number of significant digits in which the measured results for the expression needs to be displayed in the Results tab of the Outputs pane. Valid values are `2` to `15`.

Note: If you do not specify the number of significant digits, the default number of significant digits specified in the Default Formatting Options form will be used to display the measured results for the expression in the Results tab. For more information, see Specifying the Default Formatting Options on page 1084.

13. (Optional) Double-click in the *Notation* column next to the expression. From the drop-down list that appears, choose the notation style in which the measured results for the expression needs to be displayed in the Results tab of the Outputs pane.

Choose	To
default	Display results using the default notation style specified in the Default Formatting Options form. For more information, see Specifying the Default Formatting Options on page 1084.
eng	Display results in the engineering notation.
sci	Display results in the scientific notation.
suffix	Display results in the suffix notation. If you choose suffix, double-click in the *Suffix* column next to the expression and choose the suffix from the drop-down list. If you specify the suffix as *auto* or do not specify the suffix in the Suffix column, the suffix is automatically assigned.

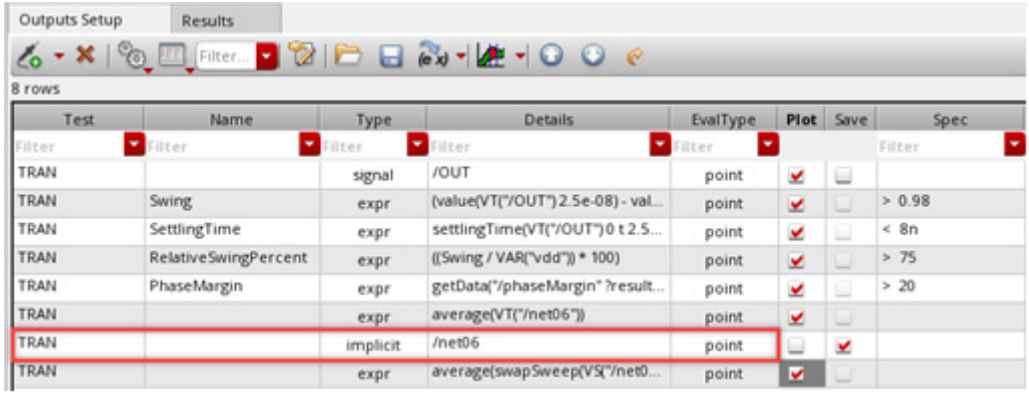
Note: If you do not specify the notation style, the default notation style specified in the Default Formatting Options form will be used to display the measured results for the expression in the Results tab. For more information, see Specifying the Default Formatting Options on page 1084.

Viewing Implicit Signals in Outputs

When you use signals in an expression and those signals are not already added in the outputs table, ADE Assembler automatically creates new outputs of type `Implicit` to save those signals.

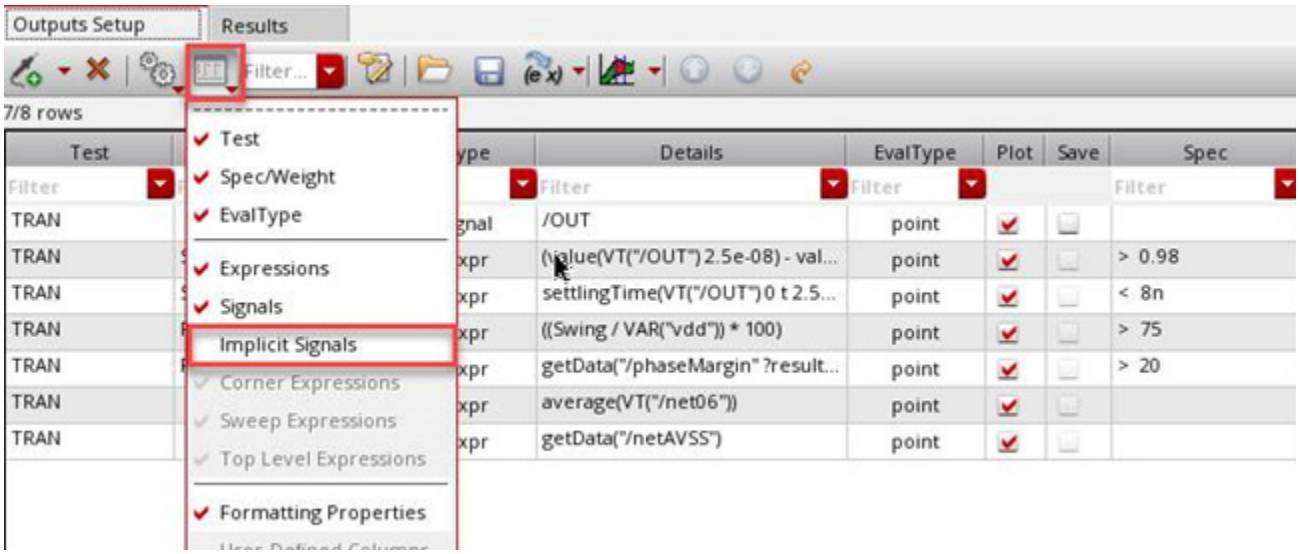
© 2023 Cadence Design Systems, Inc.

As shown in the example below, when you create `average(VT("/net06"))` as an output expression, an implicit output is created for signal `/net06`.



Please note the following points for implicit signals:

- By default, implicit outputs are not shown in the outputs table. To show these outputs, select *Implicit Signals* from the C*onfigure what is shown in the table* command menu in the toolbar of *Outputs Setup* tab.



- By default, implicit outputs are not saved or plotted. To save an implicit output in the results database, select the *Save* check box in the *Save* column.
- In the following two scenarios, the implicit signals are not saved in the netlist:
  - The outputs setup for a maestro cellview already contains expressions for which implicit signals are not created

© 2023 Cadence Design Systems, Inc.

- The *Select signals to output (save)* option on the Save Options form (which is opened by using the Save All... command in the context-sensitive menu of an output) is set to *Selected* during the creation on an output expression, and later you change the setting to *all*.

In these cases, you can set the saveImplicitSignalsDuringRun environment variable to `t`. This ensures that the implicit signals are added to the `save` statement in the netlist during the simulation run. However, those implicit signals will not be visible in the *Outputs Setup* or *Results* tab.

Related Topics

- Creating Dependent Expressions on page 381
- Creating a Combinatorial Expression on page 385
- Creating Expressions to be Measured Across Dimensions on page 389
- Viewing and Editing Output Signals or Expressions in Calculator
- Re-Evaluation of Expressions and Specifications
- Using MATLAB Expressions in Outputs on page 412
- Loading an OCEAN or a MATLAB Measurement on page 430
- Searching for Conditional Results on page 1243
- Working with Specifications on page 1329
- Specifying the Default Formatting Options on page 1084

© 2023 Cadence Design Systems, Inc.

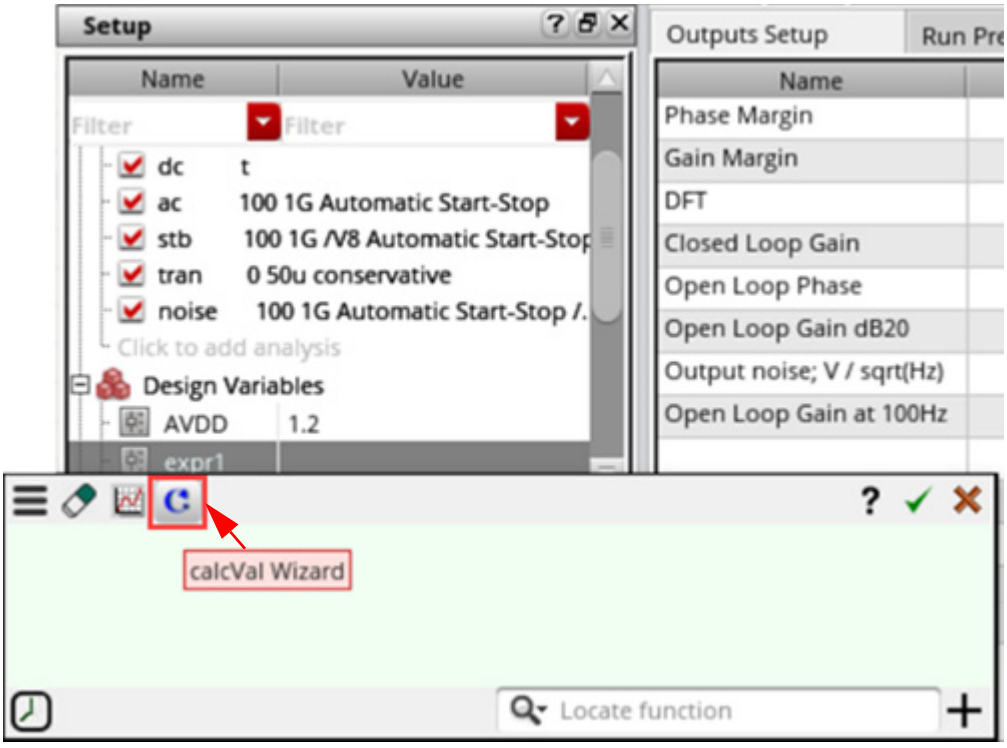
Creating calcVal Expressions Using calcVal Wizard

You can directly edit the calcVal expressions in the editing fields of the variable expressions or the Expression Builder. However, in large setups, calcVal expressions can become complex when you need to compare parameter values in multiple points. In such scenarios, calcVal Wizard is helpful in editing and debugging expressions.

The calcVal wizard fetches information from the source test and history and populates the details in the form controls for a convenient expression editing. You can focus on the inputs for the expression while the wizard manages the correctness of the syntax. For example, it automatically adds the names of key arguments and creates lists of values for the `?matchParam` argument.

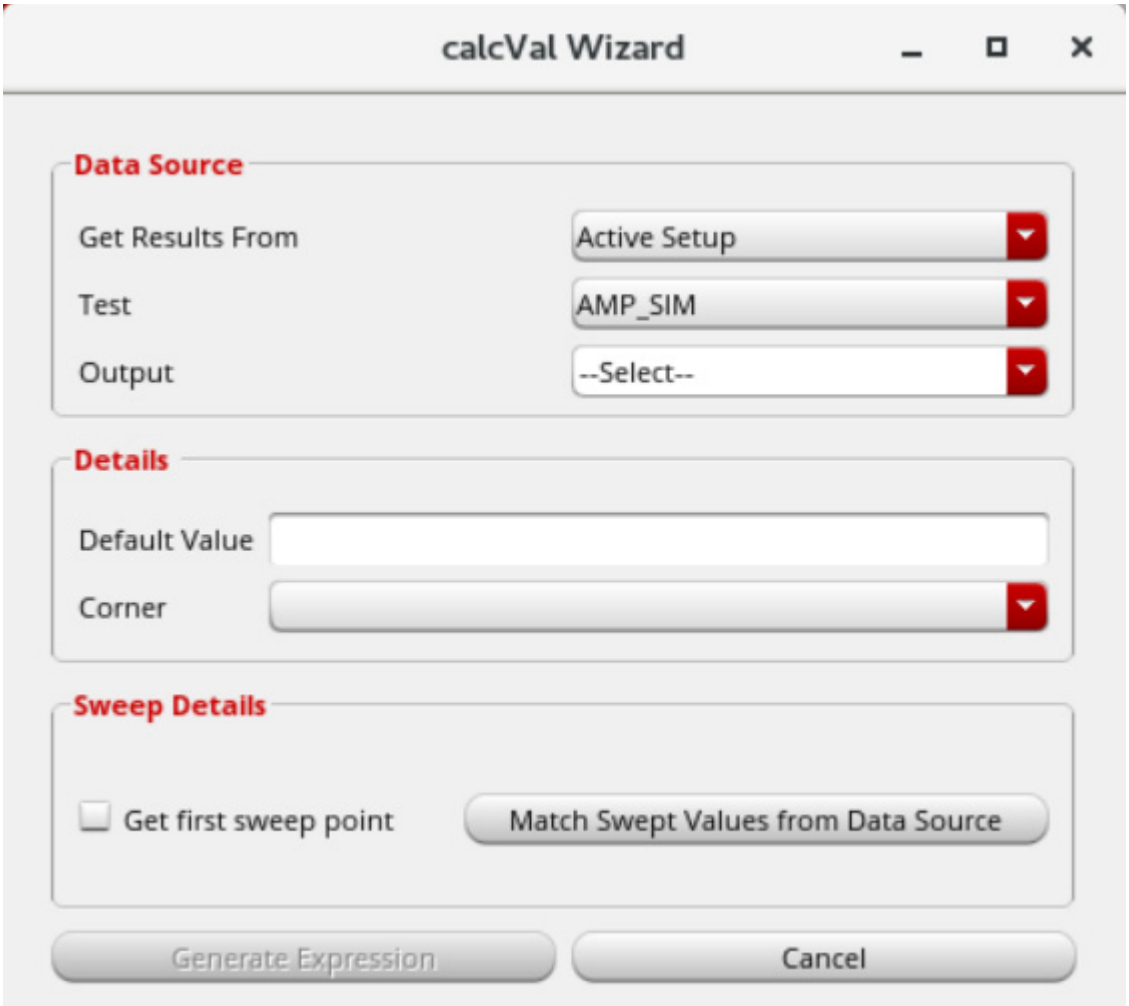
To use the calcVal wizard for a local test variable, perform the following steps in ADE Assembler:

1. Add a new design variable in the *Design Variables* tree of the Data View assistant.
2. Double-click in the *Value* column for the variable and click Click *Open expression builder* to open Expression Builder for the variable.
3. Click *calcVal Wizard* in the title bar of Expression Builder to open the calcVal Wizard form.



© 2023 Cadence Design Systems, Inc.

The calcVal Wizard form is displayed with the default values populated from the active setup, as shown below.

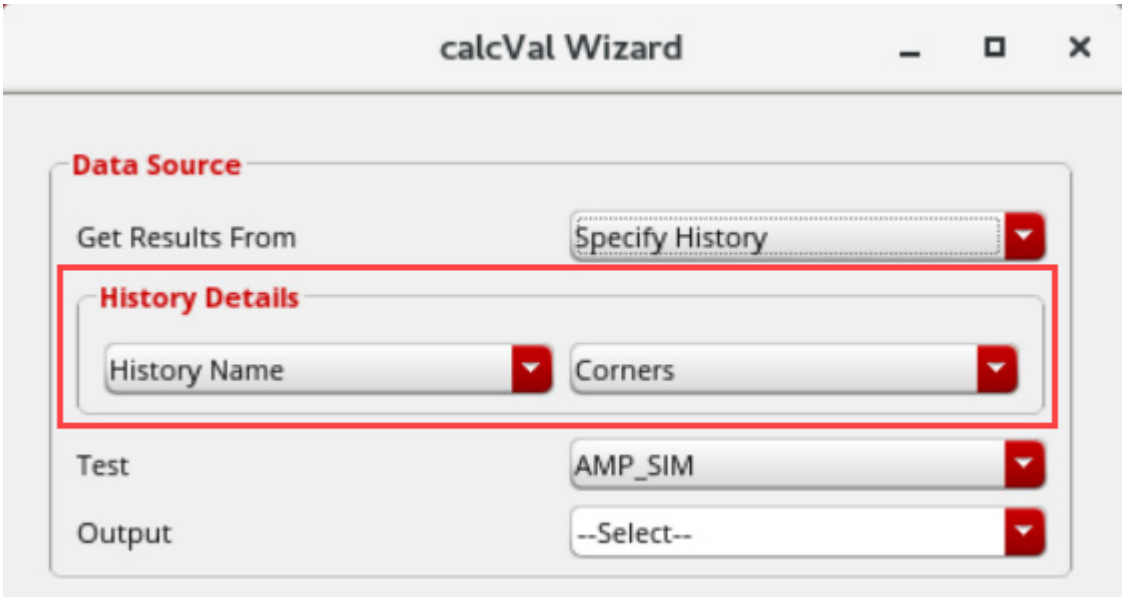


4. To use the result of a particular output of a test from the current simulation run, keep the default value, *Active Setup*, selected in the *Get Results From* drop-down list.

   Alternatively, to use the results from a specific history, choose *Specify History* from the *Get Results From* drop-down list.

© 2023 Cadence Design Systems, Inc.

An additional subsection, *History Details*, is displayed in the *Data Source* section of the form. In this section, choose a history and corner to be used as the source of results data.



5. From the *Test* drop-down list in the *Data Source* group, select the name of a source test.

   The *Output* drop-down list is populated with the outputs defined for the selected test.

6. From the *Output* drop-down list, select the name of an output. The results of this output of the selected test are used in the calcVal expression.

7. (Optional) In the *Default Value* field, specify a default value to be returned in case calcVal does not return a valid result.

8. In the *Corner* field, specify the name of the corner for which the results are to be used in calcVal.

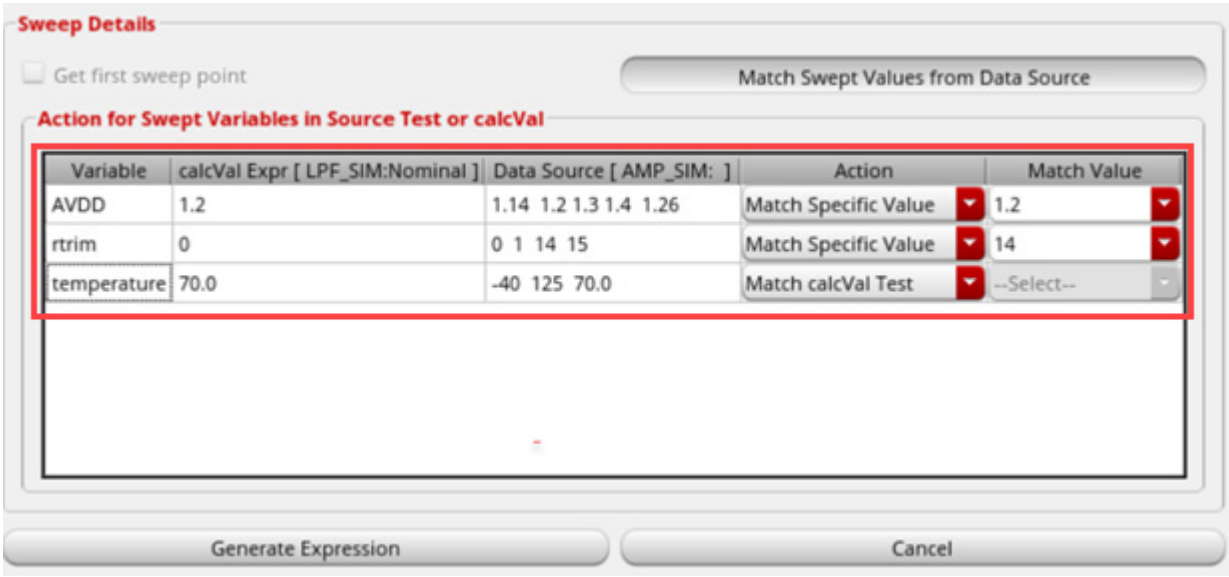
   If the setup of the current test and the test in data source do not have any common corner, it is mandatory to specify a corner in the *Corner* field.

9. If the setup contains local swept variables and you choose to use the result from the first sweep point, select *Get first sweep point* in the *Sweep Details* section.

   Alternatively, click *Match Swept Values from Data Source* to display the *Action for Swept Variables in Source Test or calcVal* table. This table lists all swept variables from the source test. The swept values from the chosen test or global variables or the corners are listed in the *Data Source* column. Values of the local design variable for the test with calcVal expression are given in the *calcVal Expr* column. If the complete test

© 2023 Cadence Design Systems, Inc.

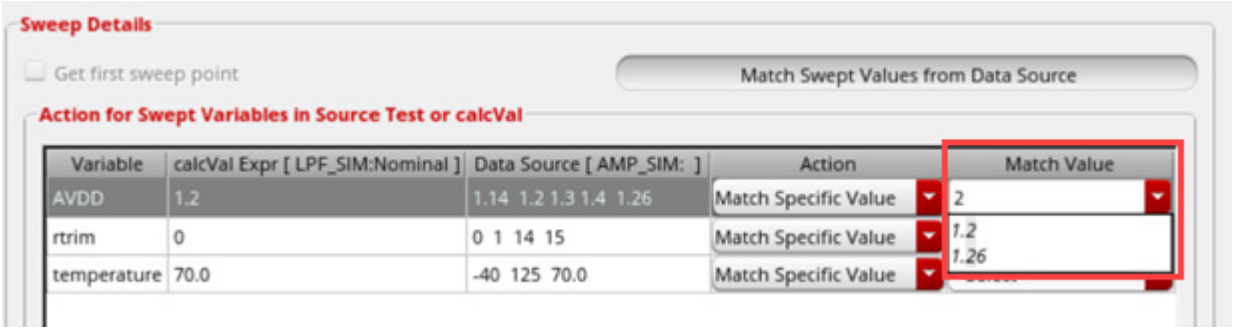
name and corner name information is not visible in the header, you can view the details in the tooltip of the column headers.



For each variable listed in the table, do the following:

a. In the *Action* field, specify an approach to get the result from the point with a matching parameter or from a point that matched the value in the calcVal test to build the `?matchParams` argument of calcVal.

b. Use the drop-down list in the *Match Value* column to scroll and choose the desired value. This field is editable. You can either type in a value or scroll through the list to select an existing value. If the list of available values is extremely long, type in characters to filter the list. Only those values in which the specified characters are found are displayed in the drop-down list to make selection convenient. In the example shown below, all values that contain `2` are filtered and displayed.



If the specified value is not found in the drop-down list, it is considered as invalid and the expression cannot be generated.

© 2023 Cadence Design Systems, Inc.

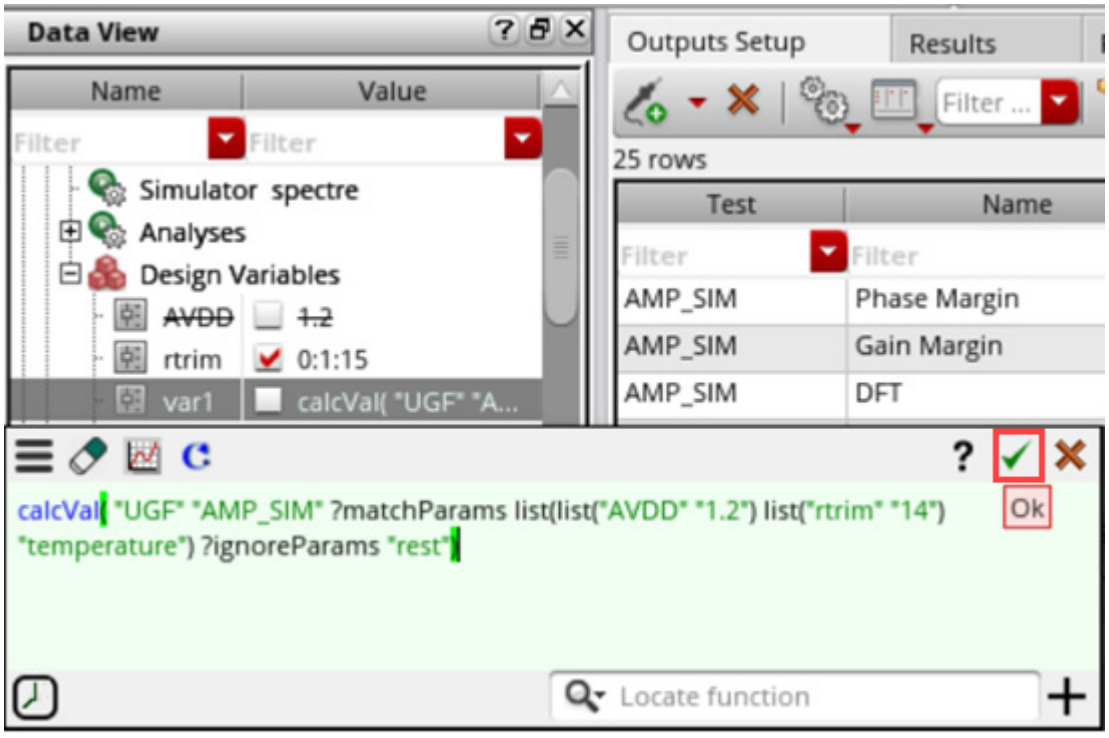
Note: If the chosen source test does not contain swept variables, *Match Swept Values from Data Source* and the corresponding table are disabled.

10. After completing values in the form, click *Generate Expression* to create the calcVal expression and send it back to Expression Builder and close this form.

The *Generate Expression* command is disabled by default. Depending on your setup, enter a corner name in the *Corner* field or details in the *Action for Swept Variables in Source Test or calcVal* table. The tool automatically enables *Generate Expression* after you have entered valid values in all the mandatory fields. If this command is not enabled, review the form for valid and ensure that all values are complete and valid.

After the form is closed, the calcVal expression is displayed in the Expression Builder. Note that if `?matchParams` is added, the tool automatically adds `?ignoreParams "rest"` to the expression.

11. If required, you can make further edits in the expression and click *Ok* (✓) to send the expression to the setup.



Note: The calcVal Wizard form is of modal type. You must close the form to access the Virtuoso session for which you are creating the calcVal expression.

© 2023 Cadence Design Systems, Inc.

Related Topics

calcVal

calcVal Wizard Form

© 2023 Cadence Design Systems, Inc.

calcVal Wizard Form

The calcVal Wizard form lets you create and edit calcVal expressions.

Field	Description
Data Source	This section provides fields that define the source of data to be used in an expression.
Get Results From	Specifies the source of data to be used in a calcVal expression. ■ *Active Setup*: Gets the data from the setup of the current simulation run. ■ *Specify History*: Gets the data from the results saved for the selected history. This field corresponds to the `?historyName` argument of calcVal.
History	Specifies the name of the history to be used as a data source. You can choose a test from the drop-down list that shows all the test names from the active setup. This field corresponds to the `?testName` argument of calcVal.
Corner	Specifies the name of the corner to be used as a data source. This field corresponds to the `?cornerName` argument of calcVal.
Test	Specifies the name of the test to be used as a data source. You can choose a test from the drop-down list that shows all the test names from the active setup. This field corresponds to the `?testName` argument of calcVal.
Output	Specifies the name of the output from the selected test. You can choose an output from the drop-down list that populates all output names from the selected test. This field corresponds to the `?outputName` argument of calcVal.
Details	This section provides fields that define the source of data to be used in an expression.

© 2023 Cadence Design Systems, Inc.

Default Value
The default value to be returned if calcVal fails for any reason.

This field corresponds to the `?defaultVal` argument of calcVal.

Corner
Specifies the name of the corner for which the results are to be used in calcVal. When not specified, the value of nominal is used.

This field corresponds to the `?cornerName` argument of calcVal.

Sweep Details
This section provides options to specify sweep points and the matching values to be considered while fetching results from the data source.

Get First Sweep Point
Specifies that when sweep values are used, the result of only the first swept point is to be used for calculation.

This field corresponds to the `?getFirstSweepPoint` argument of calcVal.

Match Swept Values from Data Source

Opens the *Action for Swept Variables in Source Test or calcVal* table where you can specify the approach to match parameters when the variables in source test have swept values.

This command creates the `?matchParams` argument of calcVal.

Action for Swept Variables in Source Test or calcVal

© 2023 Cadence Design Systems, Inc.

	A table that is dynamically generated to shows a list of all variables that have swept values either in the source test or the test for which you are creating the calcVal expression. This table is displayed when you click *Match Swept Values from Data Source* and contains the following columns: ■ *Variable*: Name of the variable that has swept values in the source test. ■ *calcVal Expr [*`TestName`: `cornerName`*]*: Displays the value of the variable from the selected corner in the test for which you are creating the calcVal expression. ■ *Date Source [*`TestName`: `cornerName`*]*: Displays the swept values of the variable from the source test. ■ *Action*: Lets you choose from the two approaches: - *Match Specific Value*: Matches a specific value from the swept set of values - *Match calcVal Test*: Matches the value of the same variable in the current point of the calcVal test. ■ *Match Value*: Specifies the specific value of the swept variable corresponding to which the output value is to be used in the calcVal expression. This column is disabled when the Action column is set to *Match calcVal Test*.
Generate Expression	Generates a calcVal expression using the information provided on this form, sends it back to Expression builder, and closes this form.
Cancel	Discards the provided information and closes the form.

Related Topics

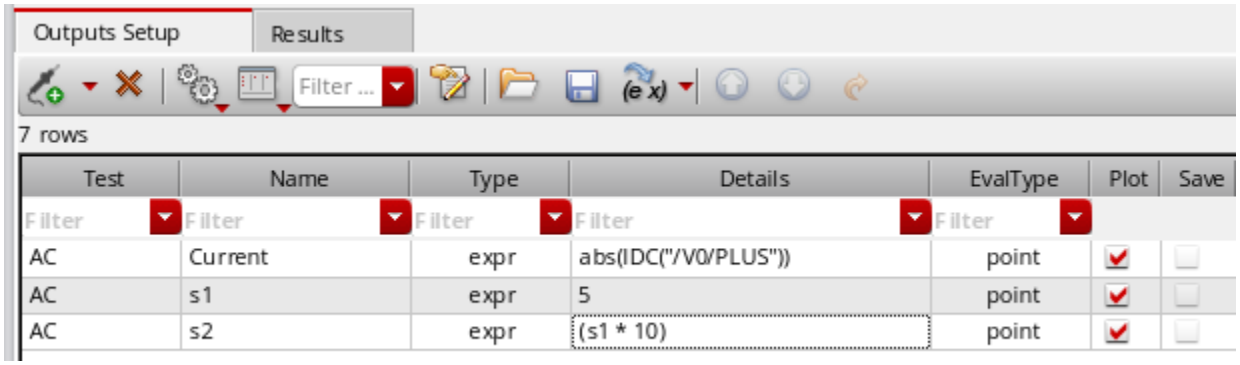
Creating calcVal Expressions Using calcVal Wizard

calcVal

© 2023 Cadence Design Systems, Inc.

Creating Dependent Expressions

You can create expressions based on other expressions. For example, assume that you have an expression named `s1` with the expression `5`. If you want another expression, say, `s2` to be ten times the value of `s1`, type `s1*10` in the *Details* field as shown in the figure below.



Note the following when you create dependent expressions:

- Expressions can be added in any order, irrespective of their dependencies. For example, if the expression `s2` is based on expression `s1`, it is not necessary to add the expression `s1` before adding the expression `s2`.
- An expression can be based on any number of other expressions.
- Ensure that there is no cyclic dependency between the dependent expressions.

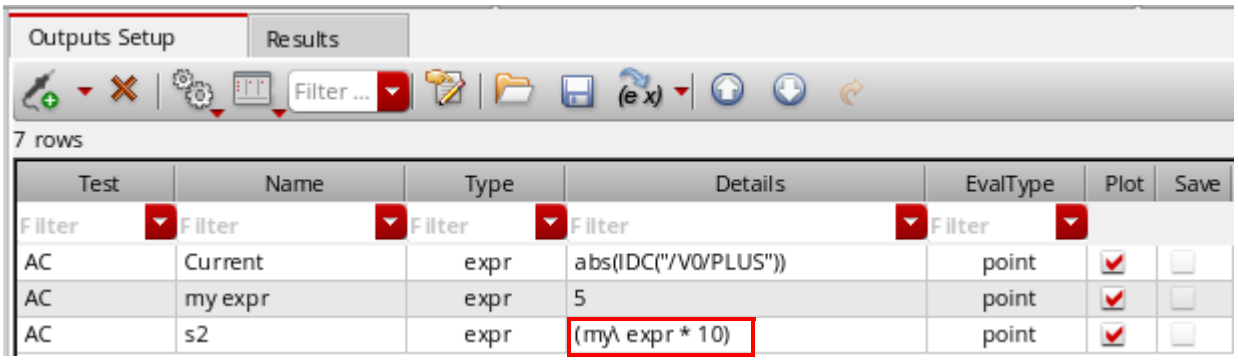
  In the following example, a cyclic dependency exists because expression `myExpr` depends on expression `myExpr1`, and expression `myExpr1` also depends on expression `myExpr`.

  ```
  myExpr=myExpr1*5
  myExpr1=myExpr+10
  ```

- If the expression name that is being used in a dependent expression has special characters such as spaces or dot (.), use the \ character to escape these special

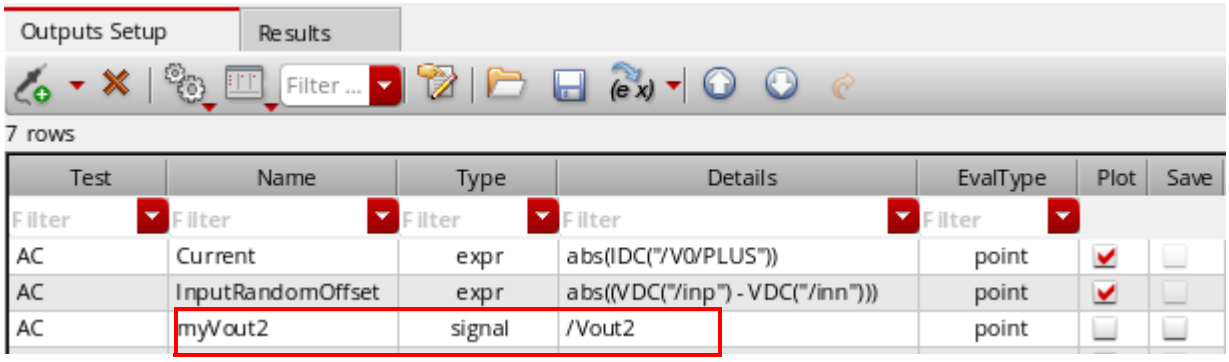
© 2023 Cadence Design Systems, Inc.

character. For example, if you are using an expression named `my expr` in another expression, escape the space character in `my expr` as shown in the figure below:



Note: Expression names can have special characters as they are valid string values. However, when these expression names are used in dependent expressions, they are treated as SKILL symbols and should be in a valid SKILL format. Therefore, special characters in such expression names should be escaped using the \ character.

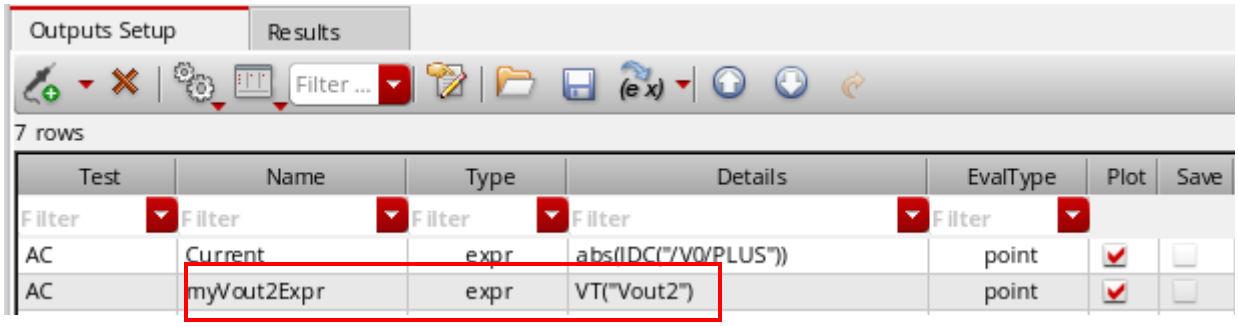
- It is not necessary to plot or save the expressions on which other expressions are based. For example, if the expression `s2` is based on expression `s1`, it is not necessary to select the *Plot* or *Save* check box for the expression `s1` in the Outputs Setup tab.
- You cannot assign a name to an output of type signal and then use that name in a dependent expression. For example, you cannot assign the name `myVout2` for the `Vout2` signal as shown below, and then use `myVout2` in a dependent expression.



If you want a dependent expression to be based on a signal, create an expression based on that signal and use the name of that expression in a dependent expression. For

© 2023 Cadence Design Systems, Inc.

example, create an expression named `myVout2Expr` based on the signal `Vout2` as shown below, and then use `myVout2Expr` in a dependent expression.

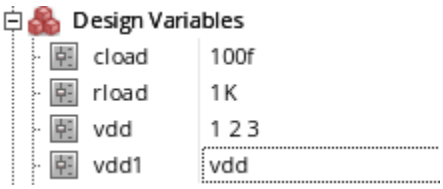


- Error messages, if any, regarding dependent expressions are displayed in the Command Interpreter Window (CIW).
- Dependent expressions will appear on the *Results* tab only if the *Plot* check box is checked. However, if you want to display of the results of intermediate expressions irrespective of the setting of the *Plot* check box, set the forceShowAutomaticExpressions environment variable to `t`.

Plotting Dependent Expressions

When sweep includes dependent design variables, scalar expressions outputs are plotted as continuous line.

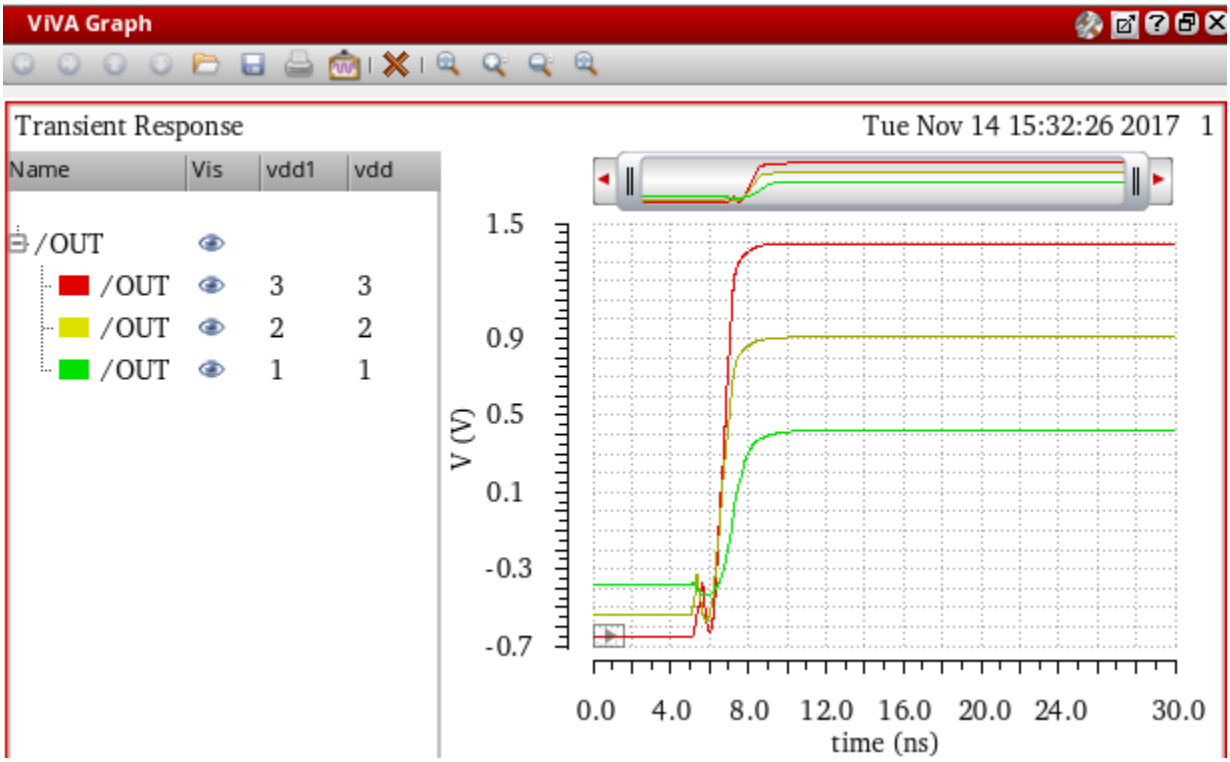
For example, consider the ADE Explorer setup in which you have the following design variables:



Here, the variable vdd1 is dependent on another variable `vdd`.

© 2023 Cadence Design Systems, Inc.

Now, if you run simulation and plot the `/OUT` signal, the following waveforms are generated for the three values of `vdd`, as shown in the figure below.



© 2023 Cadence Design Systems, Inc.

Creating a Combinatorial Expression

A combinatorial expression is one created using more than one output from one or more tests. For example, you might want to create an expression such as

```
Test1_Output1/Test2_Output2
```

To use values from another test, it is required to use the `calcVal` function.

To create a combinatorial expression such as the one above, do the following:

1. On the Outputs Setup tab, click .
2. In the drop-down list, select a test and choose *Expression*.

   A new row is added for the test with the output type *expr*.

Tip

You can double-click the test name or output type to select a different test or output type from the drop-down list that appears in the *Test* or *Type* column.

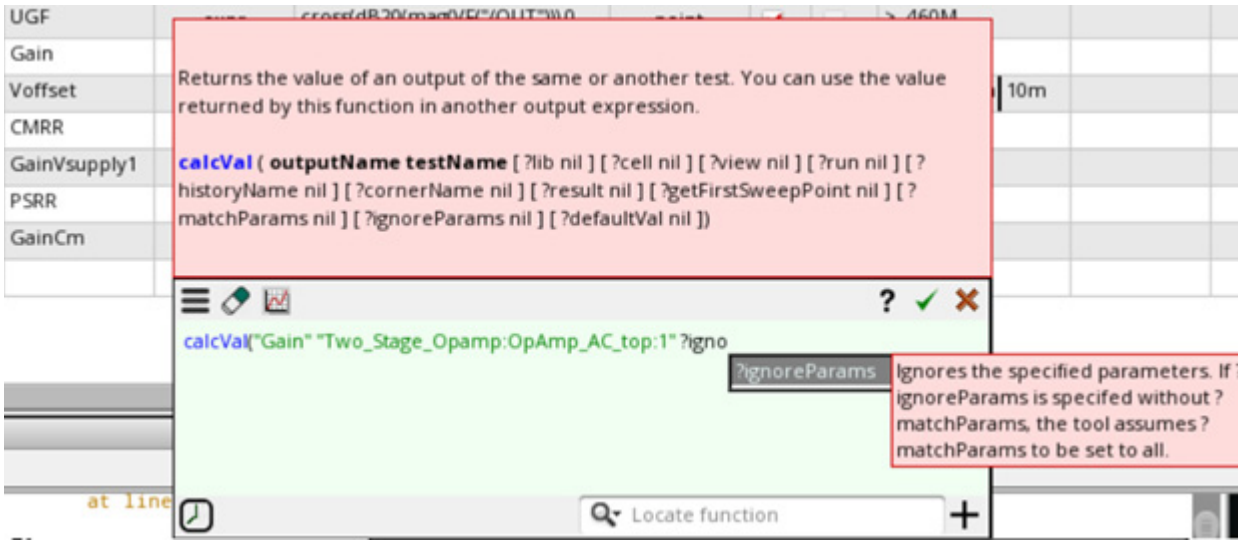
3. (Optional) In the *Name* field, type the name for the expression (such as `Out1DivOut2`).

   This name appears in the Waveform window.
4. Double-click the *Details* field.
5. Click *Open expression builder* ( ).

   The expression builder appears.
6. Type `calcVal` in the given editing area.

© 2023 Cadence Design Systems, Inc.

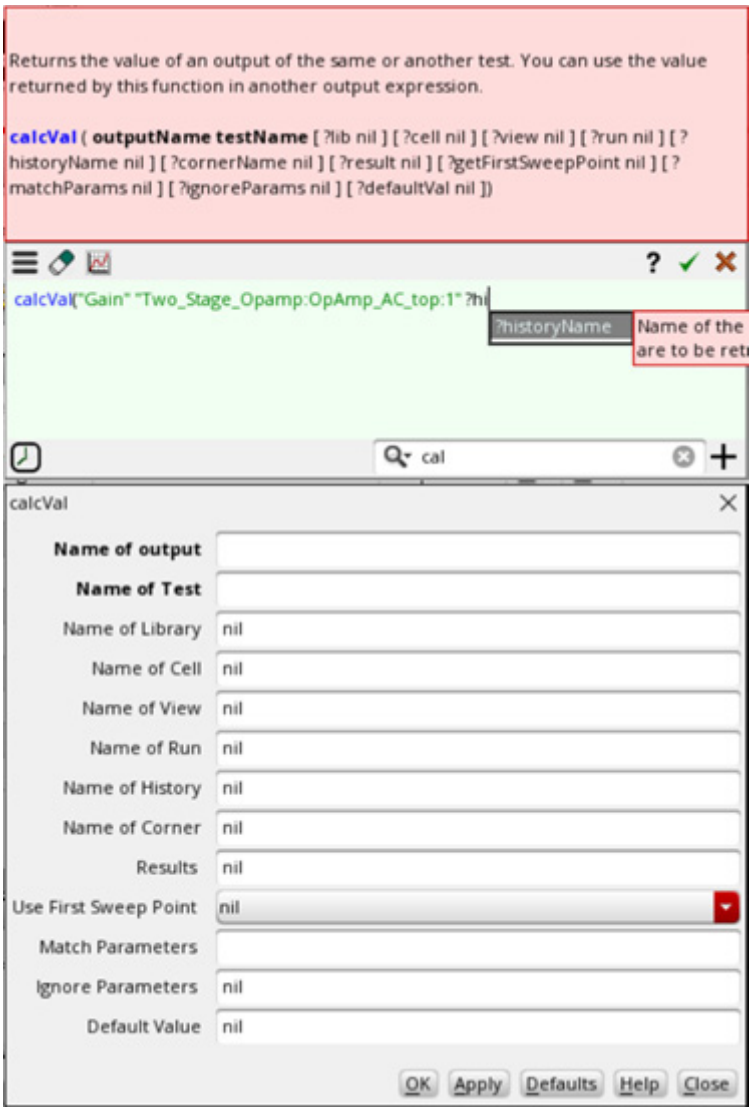
The expression builder identifies the function and displays help for that. In addition, it also helps with the possible argument values from the active setup. This enables quick function editing.



© 2023 Cadence Design Systems, Inc.

Tip

Instead of typing or choosing argument names and editing their values in the expression editing area, you can also type calcVal in the Locate Function textbox and choose the function name from the displayed list. A form showing the fields for all arguments is displayed. You can edit the values in that form.



7. Select the name of output from the given argument list that appears and specify other required arguments.
8. Type the `/` operator.
9. Type `calcVal` again and complete the other calcVal expression.

© 2023 Cadence Design Systems, Inc.

10. Click ✓ on the expression builder to add the expression to the *Details* field of the output.

A expression using results from two tests in the setup has been created.

© 2023 Cadence Design Systems, Inc.

Creating Expressions to be Measured Across Dimensions

Important

Measurements across dimensions are supported only for the following run modes:
- Single Run, Sweeps, and Corners
- Monte Carlo
- Local Optimization
- Global Optimization

Simulation results data is spread across multiple dimensions—sweeps, corners, or Monte Carlo iterations. Therefore, at times, it is required to measure outputs across these dimensions. You can do this by specifying the dimension across which you want to evaluate a result. The tool keeps only your chosen dimension across which the result is to be calculated and flattens the remaining dimensions.

ADE Assembler provides the following ways to evaluate expressions across different dimensions:

- Measurements across corners
- Measurements across sweeps or across all dimensions
- Measurements across any dimension

You can choose a dimension in the *EvalType* column of the Outputs Setup tab.

How is expression evaluation different when measured across dimensions?

When evaluated across dimensions, the results of data access functions, such as `VF`, `VT`, `VAR`, or `getData`, are different in comparison to their evaluation for a single point. This is because the data access functions read data for all the points selected by the tool depending on the evaluation type, for example, evaluation across corners or sweeps.

Consider the following example expression:

```
value(mag(VF("/OUT")) 20000000)
```

When this expression is evaluated for a single point, the result is a scalar value. However, when it is evaluated across corners, the tool gets a list of points corresponding to the leaf point directories for which evaluation has to be done and then returns a waveform output.

© 2023 Cadence Design Systems, Inc.

A constant output expression that does not need to access results data always returns the same value when evaluated for a single point or across dimensions.

When the expressions contain data access functions, you can use one of the following functions to run a second level of consolidation of the expression evaluated across dimension:

- `average`
- `ymax`
- `ymin`
- `peakToPeak`
- `stddev`

The tool automatically adds the `?overall t` argument to these functions when they are written in the measurements to be evaluated across dimensions.

```
average(value(mag(VF("/OUT")) 20000000) ?overall t)
```

After consolidation, the tool returns a scalar value.

Related Topics

Results Data Access Functions

Video

You can view the following video demonstrations related to this feature in ADE Assembler:

- Performing Measurements Across Corners in the ADE Assembler
- Performing Measurements Across Sweeps and Measurements Across All in the ADE Assembler

Measurement Across Corners

By measuring an output across corners, you evaluate an expression once for every sweep point across all the corners. It results in a single output value that is displayed in the column for the Nominal corner.

To evaluate expressions across corners, perform the following steps:

© 2023 Cadence Design Systems, Inc.

1. Right-click in the *Outputs Setup* tab and choose *Add Expression*.

   A new blank output row with output type `expr` is added to the outputs table.

2. In the new row, double-click in the *Test* column and select a test name.

3. Double-click in the *EvalType* column and specify the evaluation type as `corners`.

   The color of the row changes to blue.

   Note: Outputs of type Signal, Matlab script, Area Spec, and Op Region Spec are always evaluated for design points, that is, when evalType is equal to `point`.

4. (Optional) Add a name for the expression in the *Name* column.

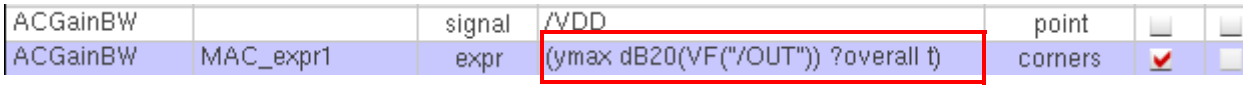
5. Enter an expression in the *Details* column.

   When the evaluation type is set as `corners`, the tool automatically adds an additional argument, `overall` and sets it to `t`.

   You can calculate measurements across corners only for the following functions:

   - average
   - ymax
   - ymin
   - peakToPeak
   - stddev

   An example is shown in the following figure.



   When the argument *`overall`* is set to `t` and the evalype is `corners`, it performs the calculation on the results of corner simulations for each design point that are treated as discrete values for evaluation and not waveforms.

   For more details on how the measures are calculated across corners, refer to Calculations of Measurements Across Corners.

© 2023 Cadence Design Systems, Inc.

Caution

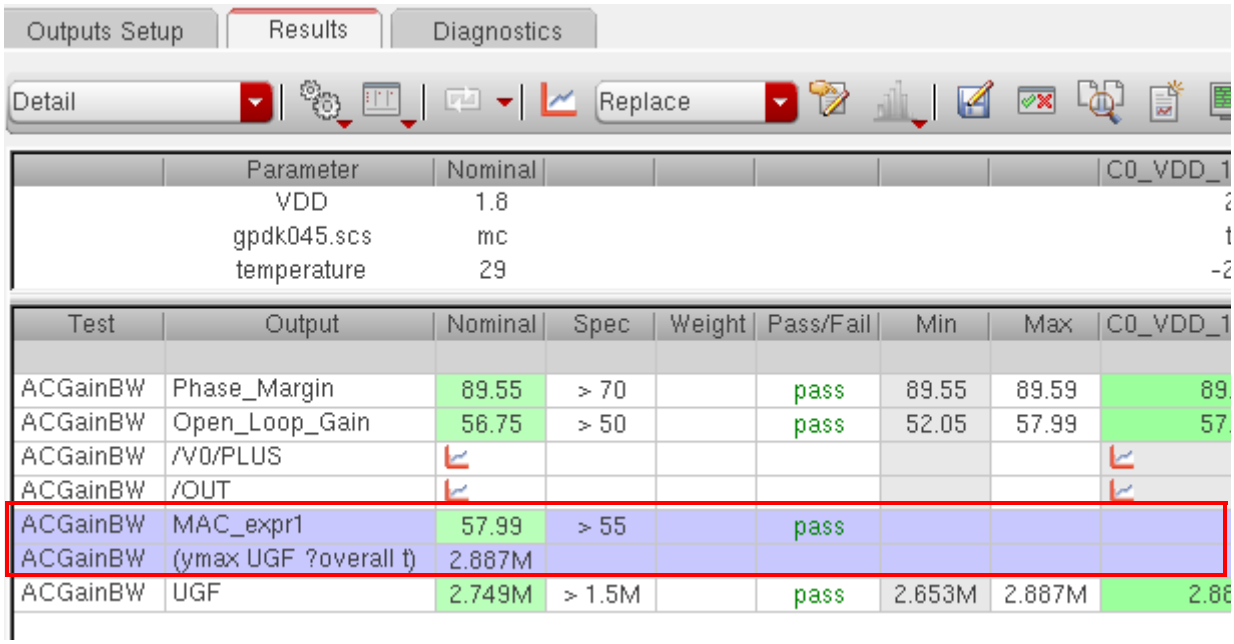
The argument, overall, has been added to only five most-frequently used Calculator functions for which measurements across dimensions are supported. Usage of this argument in any other function might generate erroneous results.

There are some more ways in which you can create expressions to be measured across corners. For more details, refer to Alternate Ways to Create Measurements Across Corners.

6. (Optional) Add specification details in the *Spec* column.

   Note: The *Override Specification* command in the popup menu on the Output Setup tab is disabled for measurements across corners.

When you run simulations, results of the expressions measured across corners are displayed on the *Results* tab, as shown in the following figure.



Note the following on the *Results* tab:

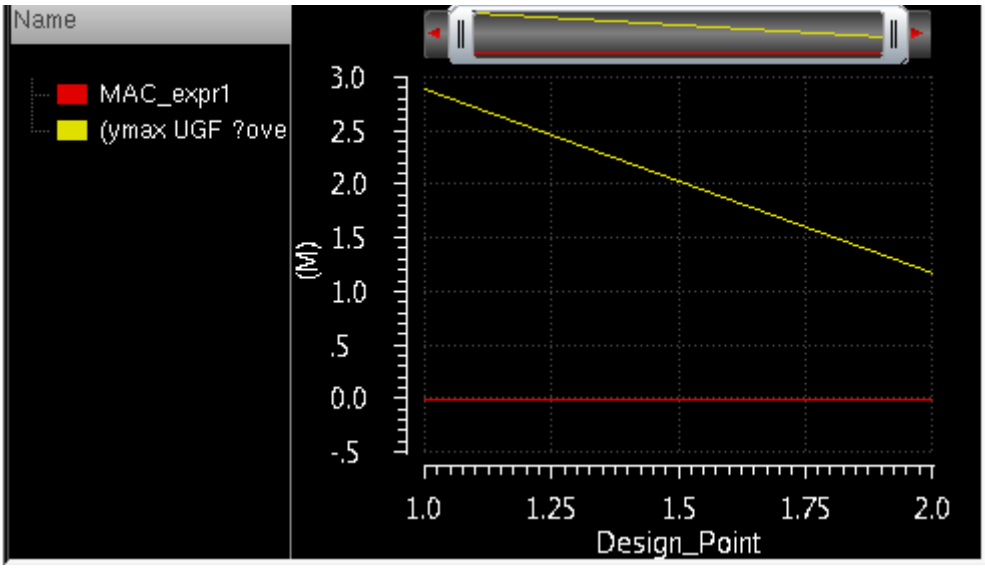
- Rows that display results of the expressions measured across corners appear in blue.
- Results of the expressions measured across corners are displayed in the *Nominal* column and the columns for other corners are blank. No new column is created to show this result.

© 2023 Cadence Design Systems, Inc.

- If you disable the nominal corner before running simulation, the results are still displayed in the *Nominal* column. Only the cells that display results for expressions measured across corners show results. Other cells in the column do not have any results.
- If you specify performance specifications for the expressions measured across corners, depending on the measured value, the tool displays the pass/fail status.
- If you open the debug environment, the outputs that are measured across corners are not shown in the *Outputs* section of the debug environment.
- Re-evaluation is not supported for the outputs that are measured across corners.
- For Monte Carlo simulations, if you clear the *Save Data to Allow Family Plots* check box and if you have outputs to be measured across corners as independent expression then it will not be evaluated and show eval error. This is because the evaluation is based on simulation results that are not saved. However, if an expression to be measured across corners is fully dependent on other expressions for which results are saved in the results database, it is evaluated and results are shown.

Other important points to note:

- When you plot the results using Auto Plot, the results of measurements across corners are plotted across all design points, as shown in the following figure.



- The results of measurements across corners are displayed in all views on the *Results* tab.

© 2023 Cadence Design Systems, Inc.

- For Monte Carlo results, in the yield view, only a single row is displayed for the expressions measured across corners. However, for other expressions, result for each corner is displayed in a separate row.

Alternate Ways to Create Measurements Across Corners

Alternate ways in which you can create expressions to be evaluated across corners are:

- By creating OCEAN measures in a .ocn file and loading that file. For example:

```
MAC_ocn1= ymax(bandwidth(dB20(mag(VF("/OUT"))) 10 "low") ?overall t)
axlOutputResult(MAC_ocn1 "MAC_ocn1")
```

  While creating an OCEAN measure in one file, you can also use an expression defined in another OCEAN file. For example, ocean1.ocn is defined as given below.

```
BW=bandwidth(VF("/net1") 10 "low")
axlOutputResult(BW "BW")
```

  In ocean2.ocn, you can create another measure that uses BW that was defined in ocean1.ocn. This is shown in the example given below.

```
MAC_ocn1=ymax(calcVal("BW" "testLib:top:1") ?overall t)
axlOutputResult(MAC_ocn1 "MAC_ocn1")
```

Important

  If you want to create an OCEAN measure of both the types, point and corners, you need to create separate OCEAN script files where one file contains functions of a single type only. Use a separate *axlOutputResult* statement for each output.

- By using the ocnxlOutputExpr OCEAN function in an OCEAN script to create an output. This function supports the `evalType` argument that specifies how to evaluate the expression.

Calculations of Measurements Across Corners

When the *evalType* column is set to `corners`, the measurements are calculated across corners. This section describes how different scalar and waveform expressions are calculated to measure the outputs across corners.

© 2023 Cadence Design Systems, Inc.

Measurements of Scalar Expressions

When the *evalType* column is set to `corners`, by default the *`overall`* argument is set to `t`. With this default value of *`overall`*, the measurement is calculated by using the result values obtained for each corner, irrespective of the number of sweep variables. This gives a scalar value as result.

In the example shown below, **average** of risetime is calculated by adding the risetime calculated for all the sweep variables and then dividing the sum by the count of values.

rt_avg = average of risetime scalar values calculated for each corner

See the results of rt_avg in the figure given below.

TRAN	risetime	expr	riseTime(VT("/OUT") 0 t 1e-08 t 10 90 nil "time")	point
TRAN	rt_avg_t	expr	(average risetime ?overall t)	corners
TRAN	rt_avg_nil	expr	(average risetime ?overall nil)	corners
TRAN	rt_max_t	expr	(ymax risetime ?overall t)	corners
TRAN	rt_max_nil	expr	(ymax risetime ?overall nil)	corners

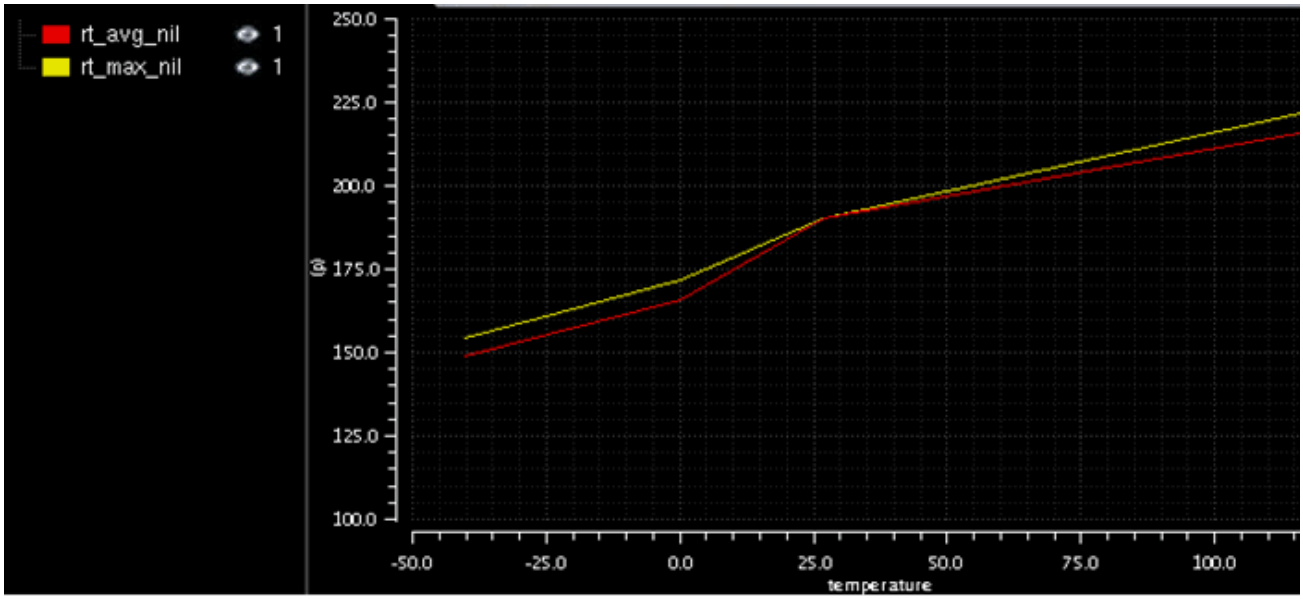
	Parameter	Nominal	C0_0	C0_1	C0_2	C0_3	C0_4	C0_5	C0_6	C0_7
	temperat...	27	-40	0	125	-40	0	125	-40	0
	vdd	1.8	1.9	1.9	1.9	2	2	2	2.1	2.1

Test	Output	Nominal	C0_0	C0_1	C0_2	C0_3	C0_4	C0_5	C0_6	C0_7
TRAN	risetime	190.2p	154.7p	171.7p	225.1p	148.7p	165.6p	218p	143.6p	160.3p
TRAN	rt_avg_t	178.9p								
TRAN	rt_avg_nil									
TRAN	rt_max_t	225.1p								
TRAN	rt_max_nil									

If you change the default value of the *`overall`* argument and set it to `nil`, this returns a scalar value in case of a single sweep variable, but a waveform output in case of multiple sweep variables.

In the example given above, for rt_max_nil, ymax is calculated across vdd at each temperature value and a waveform is plotted for each temperature. The result is shown in the figure given below.

© 2023 Cadence Design Systems, Inc.



Note: Getting a waveform in case of multiple sweep variables is not the desired output for measurements across corners.

Measurements of Waveform Expressions

For the default value of the `overall` argument, a measurement is first calculated on the waveform for each corner thereby resulting into a set of values. The measurement is further applied on this resulting set, thereby giving a scalar value as a result.

Consider the example shown below.

© 2023 Cadence Design Systems, Inc.

TRAN	avg	expr	average(VT("/OUT"))	point
TRAN	wave_max	expr	ymax(VT("/OUT"))	point
TRAN	wave_min	expr	ymin(VT("/OUT"))	point
TRAN	wave_avg_t	expr	(average VT("/OUT") ?overall t)	corners
TRAN	wave_avg...	expr	(average VT("/OUT") ?overall nil)	corners
TRAN	wave_pp_t	expr	(peakToPeak VT("/OUT") ?overall t)	corners

	Parameter	Nominal			C0_0	C0_1	C0_2	C0_3	C0_4	C0
	temperature	27			-40	0	125	-40	0	12
	vdd	1.8			1.9	1.9	1.9	2	2	2

Test	Output	Nominal	Min	Max	C0_0	C0_1	C0_2	C0_3	C0_4	C0
TRAN	avg	717.2m	717.2m	843.8m	762.4m	760.5m	752.5m	803.2m	801.8m	794.
TRAN	wave_max	902.5m	902.5m	1.055	952.4m	952.7m	953.5m	1.003	1.003	1.00
TRAN	wave_min	-623.6m	-657.5m	-614.4m	-614.4m	-628m	-637.6m	-624.4m	-638m	-647.
TRAN	wave_avg_t	791.4m								
TRAN	wave_avg_t1									
TRAN	wave_pp_t	1.712								

In this example, for **wave_avg_t**, first average of `VT("/OUT")` is calculated for each corner individually and then discrete average of the resulting values is taken, as shown below.

```
average = average values obtained for corners
```

Similarly, for **wave_pp_t**, first, ymax and ymin are calculated for `VT("/OUT")` for each corner. Next, ymax and ymin of the resulting values is calculated.

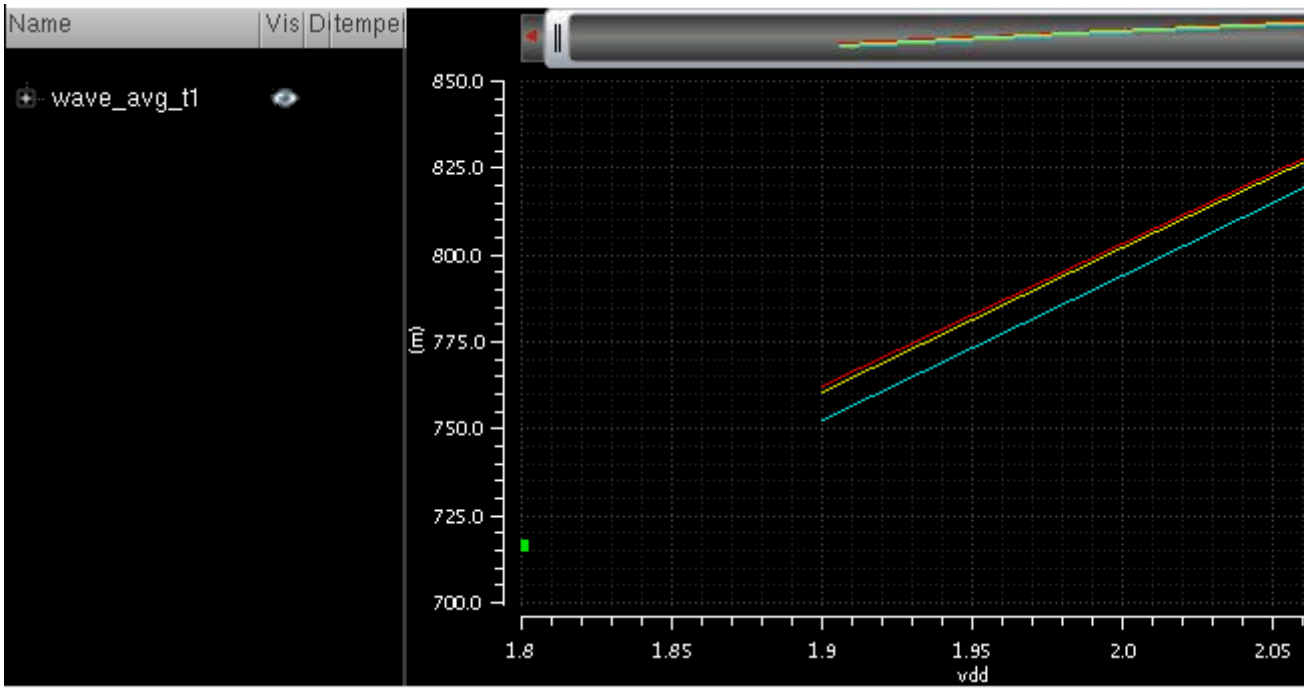
```
ymax_all = ymax of the ymax values of individual waveforms
ymin_all = ymin of the ymin values of individual waveforms
```

Next, peakToPeak is calculated using the resulting ymax_all and ymin_all values, thereby giving a scalar result.

```
peakToPeak = ymax_all - ymin_all
```

However, if you set the *`overall`* argument to `nil`, a measurement is calculated on the waveform for each corner and the resulting values are plotted. This is not the expected result for measurements across corners. In the example shown above, for wave_avg_t1, average is calculated for each combination of VDD and temperature and the resulting values are plotted, as shown in the figure shown below.

© 2023 Cadence Design Systems, Inc.



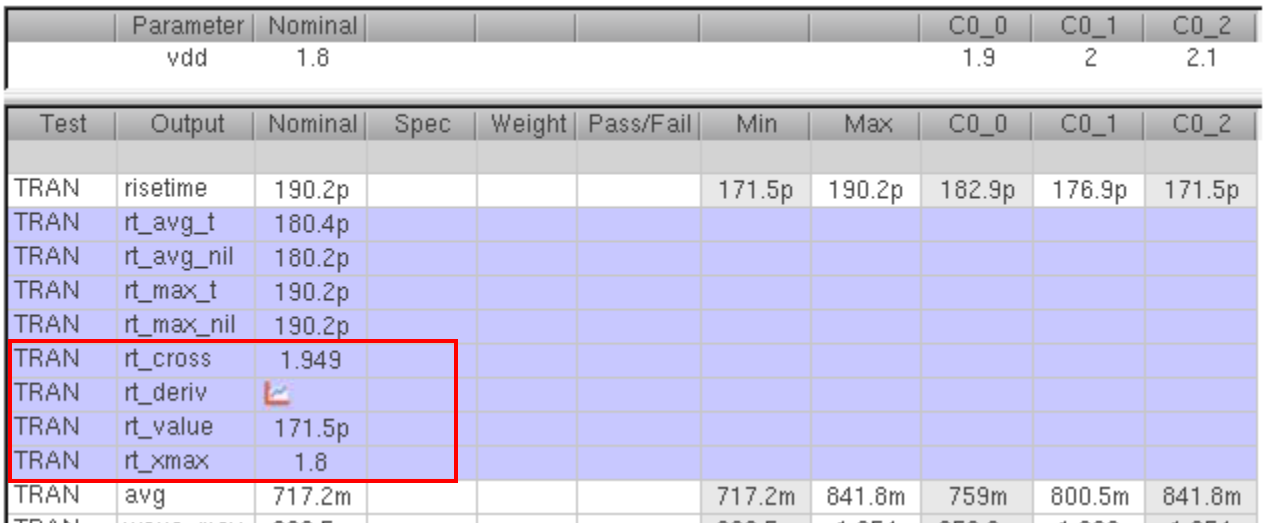
Measurements Across Corners for Functions that Run on a Single Corner Variable

Some of the calculator functions that do not support the new argument, `overall`, give correct results when calculated across corners. For example, functions such as `xmin`, `xmax`, `deriv`, `cross`, and `value`, that work on a single dimension, give correct results when calculated across corners, as shown below.

TRAN	rt_cross	expr	cross(risetime 1.8e-10 1 "falling")	corners
TRAN	rt_deriv	expr	deriv(risetime)	corners
TRAN	rt_value	expr	value(risetime "vdd" 2.1)	corners
TRAN	rt_xmax	expr	xmax(risetime)	corners

The results for these outputs are shown below.

© 2023 Cadence Design Systems, Inc.



	Parameter	Nominal						C0_0	C0_1	C0_2
	vdd	1.8						1.9	2	2.1

Test	Output	Nominal	Spec	Weight	Pass/Fail	Min	Max	C0_0	C0_1	C0_2
TRAN	risetime	190.2p				171.5p	190.2p	182.9p	176.9p	171.5p
TRAN	rt_avg_t	180.4p								
TRAN	rt_avg_nil	180.2p								
TRAN	rt_max_t	190.2p								
TRAN	rt_max_nil	190.2p								
TRAN	rt_cross	1.949								
TRAN	rt_deriv									
TRAN	rt_value	171.5p								
TRAN	rt_xmax	1.8								
TRAN	avg	717.2m				717.2m	841.8m	759m	800.5m	841.8m

However, it is not recommended to measure outputs across corners for other functions that work on multiple dimensions and do not support the `overall` argument.

Measurement Across Sweeps or Measurement Across All

By measuring an output across sweeps, you evaluate an expression once across all the sweep points for each corner.

Similarly, by measuring an output across all the sweep points and corners, you evaluate an expression once across all the results for the available sweep points and corners. These measurements provide a single resulting value.

The results for both the types of measurements are displayed in a separate table displayed at the top of the Detail results view.

To evaluate expressions across sweeps or anything, perform the following steps:

1. Right-click in the *Outputs Setup* tab and choose *Add Expression*.

   A new blank output row with output type `expr` is added to the outputs table.

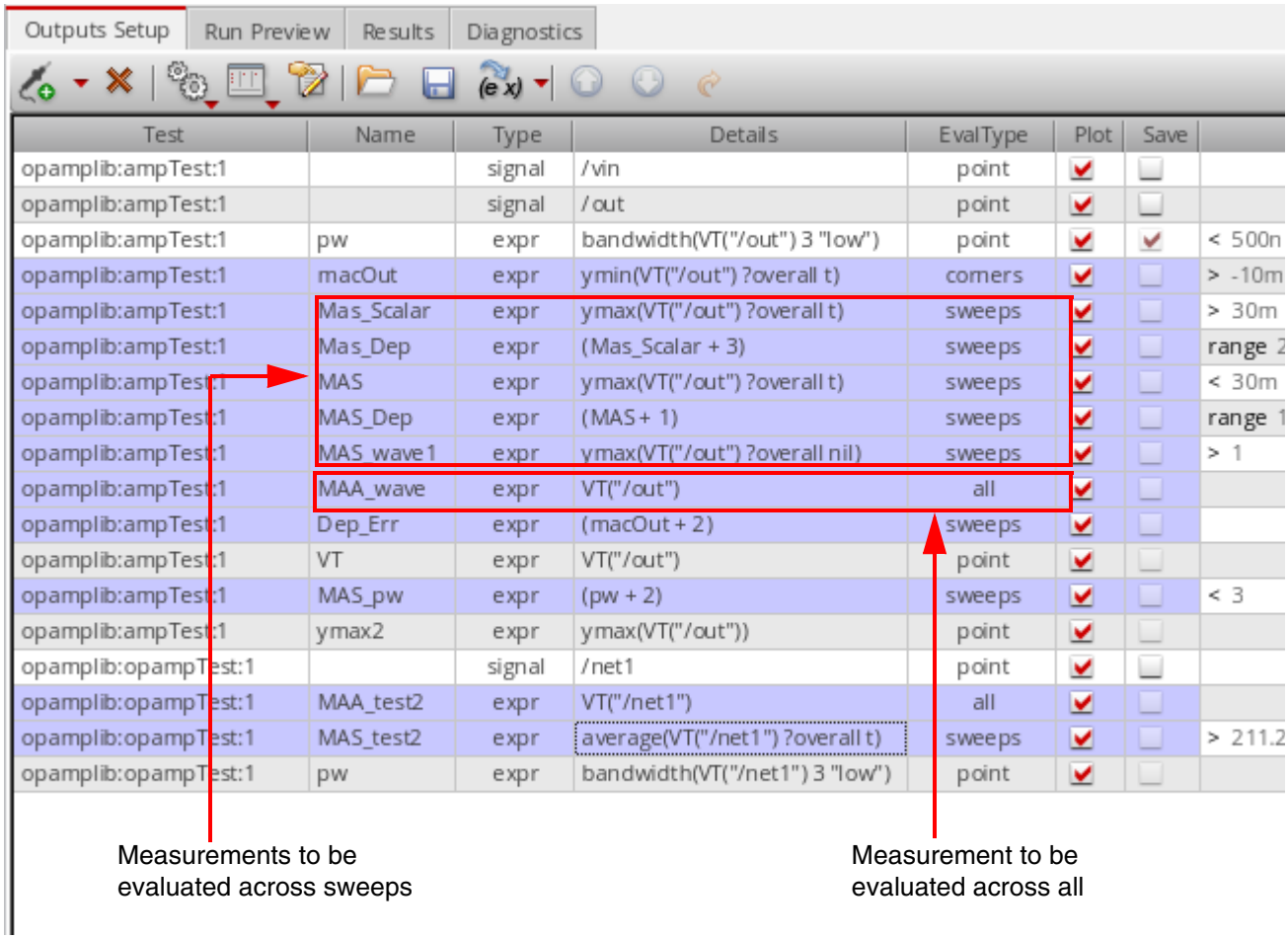
2. In the new row, double-click in the *Test* column and select a test name.

3. For a measurement to be evaluated across sweeps, double-click in the *EvalType* column and specify the evaluation type as `sweeps`.

   For a measurement to be evaluated across all, specify the evaluation type as `all`.

© 2023 Cadence Design Systems, Inc.

The color of the row changes to blue.



Test	Name	Type	Details	EvalType	Plot	Save	
opamplib:ampTest:1		signal	/vin	point	☑	☐	
opamplib:ampTest:1		signal	/out	point	☑	☐	
opamplib:ampTest:1	pw	expr	bandwidth(VT("/out") 3 "low")	point	☑	☑	< 500n
opamplib:ampTest:1	macOut	expr	ymin(VT("/out") ?overall t)	corners	☑	☐	> -10m
opamplib:ampTest:1	Mas_Scalar	expr	ymax(VT("/out") ?overall t)	sweeps	☑	☐	> 30m
opamplib:ampTest:1	Mas_Dep	expr	(Mas_Scalar + 3)	sweeps	☑	☐	range 2
opamplib:ampTest:1	MAS	expr	ymax(VT("/out") ?overall t)	sweeps	☑	☐	< 30m
opamplib:ampTest:1	MAS_Dep	expr	(MAS + 1)	sweeps	☑	☐	range 1
opamplib:ampTest:1	MAS_wave1	expr	ymax(VT("/out") ?overall nil)	sweeps	☑	☐	> 1
opamplib:ampTest:1	MAA_wave	expr	VT("/out")	all	☑	☐	
opamplib:ampTest:1	Dep_Err	expr	(macOut + 2)	sweeps	☑	☐	
opamplib:ampTest:1	VT	expr	VT("/out")	point	☑	☐	
opamplib:ampTest:1	MAS_pw	expr	(pw + 2)	sweeps	☑	☐	< 3
opamplib:ampTest:1	ymax2	expr	ymax(VT("/out"))	point	☑	☐	
opamplib:opampTest:1		signal	/net1	point	☑	☐	
opamplib:opampTest:1	MAA_test2	expr	VT("/net1")	all	☑	☐	
opamplib:opampTest:1	MAS_test2	expr	average(VT("/net1") ?overall t)	sweeps	☑	☐	> 211.2
opamplib:opampTest:1	pw	expr	bandwidth(VT("/net1") 3 "low")	point	☑	☐	

4. (Optional) Add a name for the expression in the *Name* column.
5. Enter an expression in the *Details* column.

   You would notice that if you use any of the following functions, ADE Assembler automatically adds an additional argument, `overall` and sets it to `t`:

   - ❑ average
   - ❑ ymax
   - ❑ ymin
   - ❑ peakToPeak
   - ❑ stddev

© 2023 Cadence Design Systems, Inc.

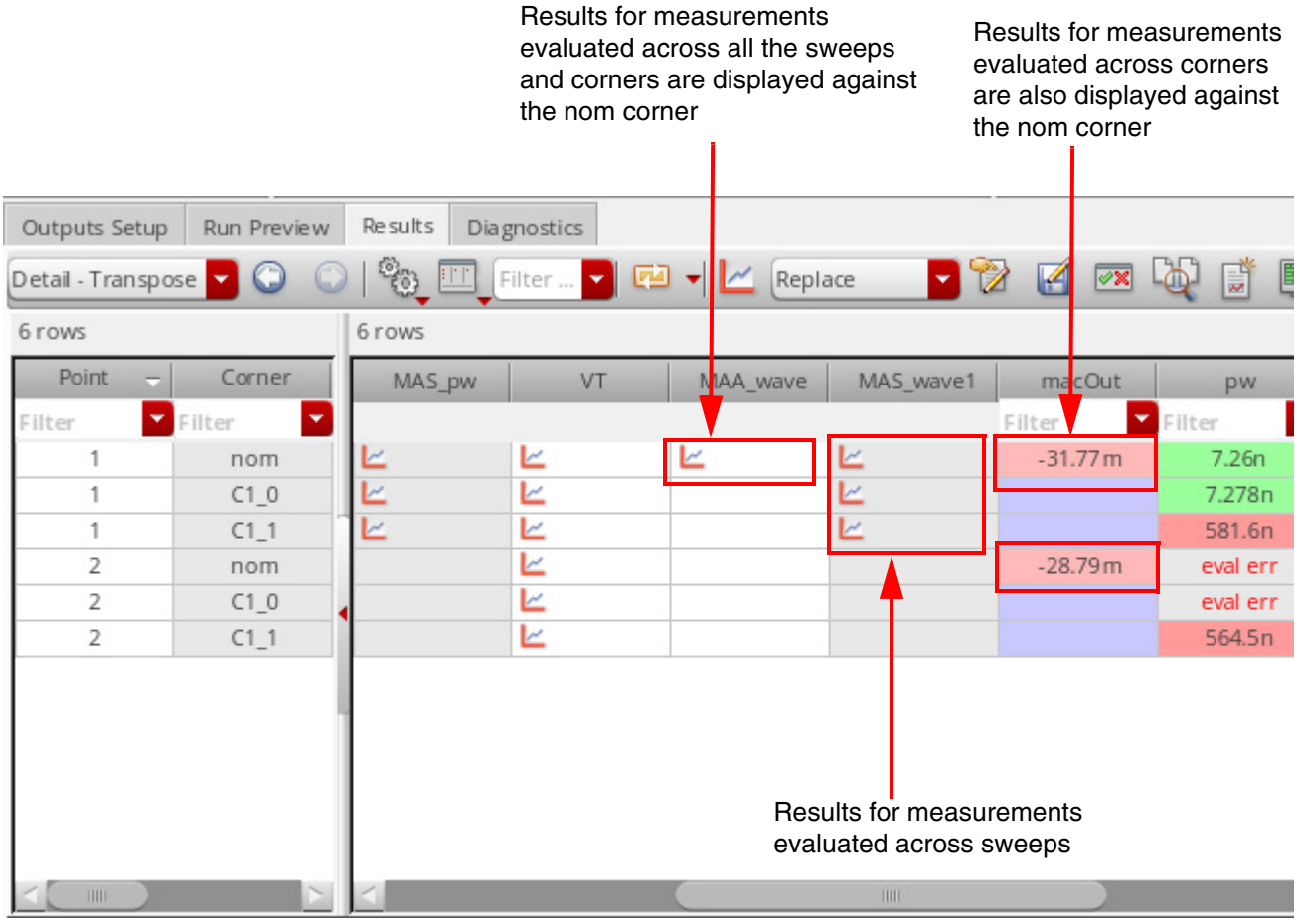
When you run simulations, results for the outputs measured across sweeps are displayed in a separate table on top of the detailed results.

The following example shows how the results are displayed:

© 2023 Cadence Design Systems, Inc.

Important Points to Note

- Outputs measured across sweeps can be dependent only on other outputs of type `point` or `sweeps`. They cannot be dependent on the outputs of type `corners` or `all`.
- Similarly, outputs measured across all can be dependent only on other outputs of type `point` or `all`.
- Re-evaluation of results is supported for the outputs measured across sweeps or across all.
- Results for measurements across different dimensions are displayed in the Detail - Transpose results view as shown below.



- You can toggle the *Configure what is shown in the table – Post Run Results* command on the Results toolbar to show or hide the results of outputs of type `sweeps` or `all`.

© 2023 Cadence Design Systems, Inc.

Creating Expressions to be Measured Across Any Dimension

The expressions that can be measured across any dimension give you the flexibility to specify a dimension or a combination of dimensions across which you want to evaluate outputs. Accordingly, ADE Assembler slices the result data and evaluates expressions.

To create an expression that is measured across any dimension:

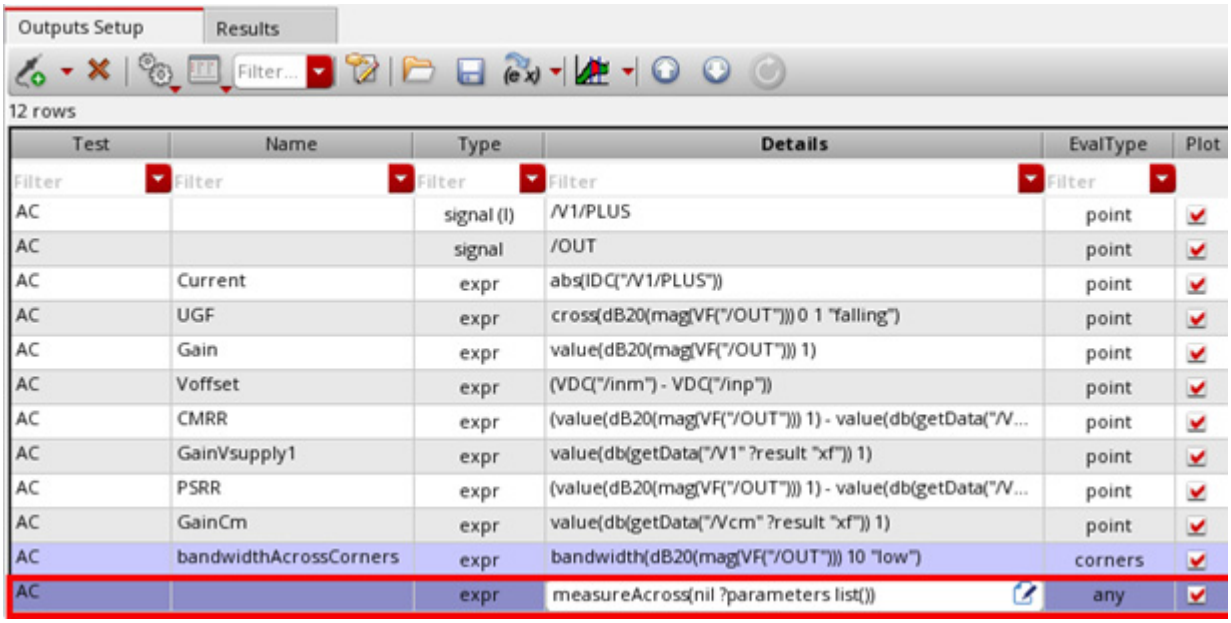
1. Right-click in the *Outputs Setup* tab and choose *Add Expression*.

   A new blank output row with output type `expr` is added to the outputs table.

2. In the new row, double-click in the *Test* column and select a test name.

3. Double-click in the *EvalType* column and choose `any` from the drop-down list.

   The tool adds a new function `measureAcross(nil ?parameters list())` in the *Details* cell for this new row. The color of this row is changed to dark purple to indicate its type.



4. Edit the `measureAcross` function. Remove `nil` from the first argument and add an expression. In the `?parameters` argument, specify a list of dimensions across which the expression is to be calculated. You can specify a list of corners, swept variables, device parameters, or aliases as a dimension.

   Currently, the following aliases are supported in `?parameters`:

   - `Sweeps`: to specify that the calculation is to be done across all swept variables

© 2023 Cadence Design Systems, Inc.

- `Corners:` to specify that the calculation is to be done across all corners
- `mc_iteration:` to specify that the calculation is to be done across the Monte Carlo parameter `monteCarlo::param::sequence`. This alias is valid for the Monte Carlo Sampling run mode only.

A few examples of `measureAcross` are shown below.

The following expression calculates the average phase margin across two dimensions, the `vdd` design variable and Monte Carlo iterations.

```
measureAcross(average(PhaseMargin ?overall t) ?parameters list("vdd"
"mc_iteration"))
```

The following expression calculates the average phase margin across two design variables, the `vdd` and `temperature` swept variables.

```
measureAcross(average(PhaseMargin ?overall t) ?parameters list("vdd"
"temperature"))
```

The following expression calculates the average phase margin across the `vdd` design variable and all corners.

```
measureAcross(average(PhaseMargin ?overall t) ?parameters list("vdd"
"Corners"))
```

The following expression calculates the average phase margin across the device parameters PM4/w and PM4/l and all corners.

```
measureAcross(average(PhaseMargin ?overall t) ?parameters list("PM4/w" "PM4/l"
"Corners"))
```

The following expression calculates the current across the varying values of a parameter `M3/fw` that is swept as `5.1u:0.1u:5.5u`:

```
measureAcross(Current ?parameters list( "M3/fw"))
; In this example, Current is another expression, Current=abs(IDC("/V1/PLUS"))
```

The following expression calculates the average phase margin across all Monte Carlo iterations.

```
measureAcross(average(PhaseMargin ?overall t) ?parameters list(
"mc_iteration"))
```

A new output expression is created.

Note: If you use `average`, `ymax`, `ymin`, `peakToPeak`, or `stddev` function in `measureAcross`, the tool automatically adds an additional argument, `?overall`, and sets it to `t`. With this argument, the tool performs the calculation on the results across the swept conditions. If this is not required for your expression, you can set `?overall` to `nil`.

The expressions with *Type* set to `any` are evaluated after all simulations are run and other expressions are calculated.

© 2023 Cadence Design Systems, Inc.

Related Topics

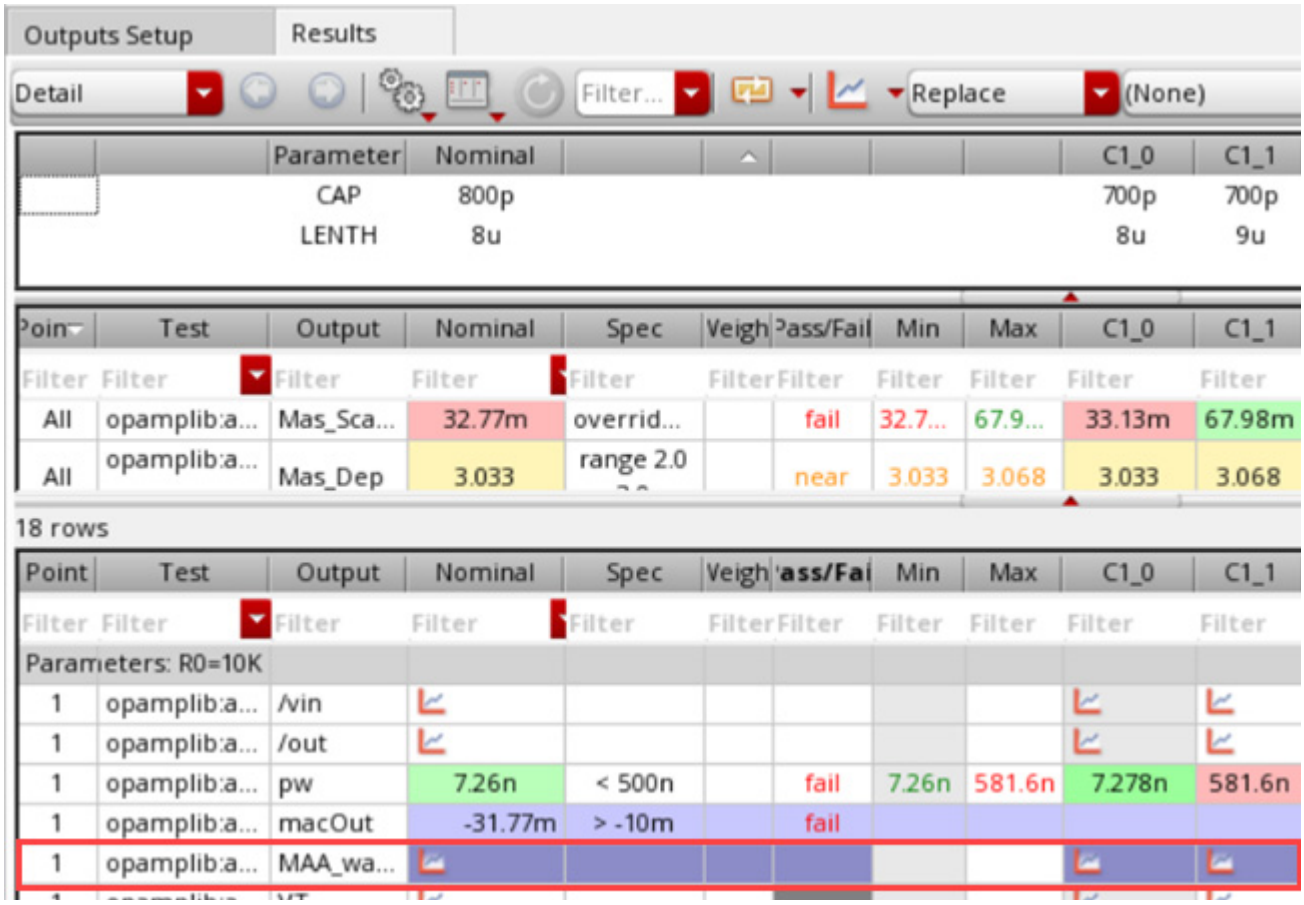
measureAcross

Viewing Results of Outputs Measured Across Any Dimension

© 2023 Cadence Design Systems, Inc.

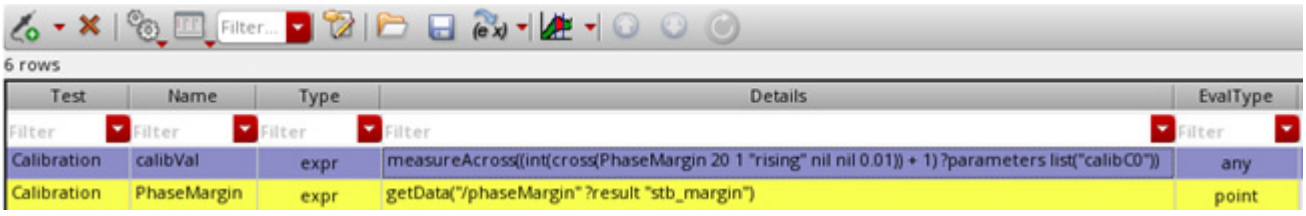
Viewing Results of Outputs Measured Across Any Dimension

The results of expressions evaluated across any dimension are highlighted in dark purple in the results table. In the example shown below, output `MAA_wave` has been evaluated across parameter `R0`.



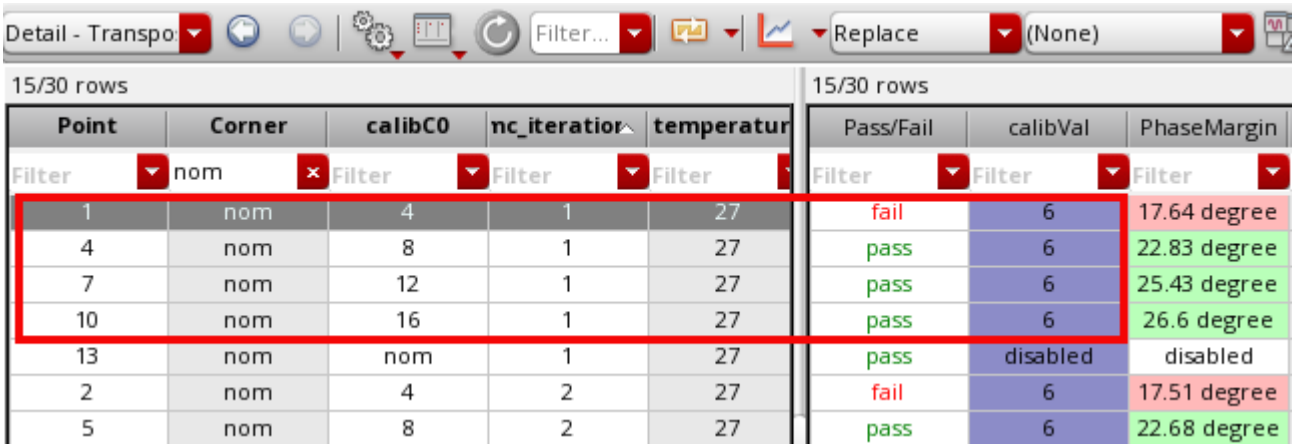
The results of outputs with type `any` are calculated across different dimensions that might not match the row and column combination of the Results table. As a result, the values for these outputs are repeated for the dimensions across which they are not measured.

In the following example, output `calibVal` calculates a `cross` of `PhaseMargin` across the variable `calibC0`.



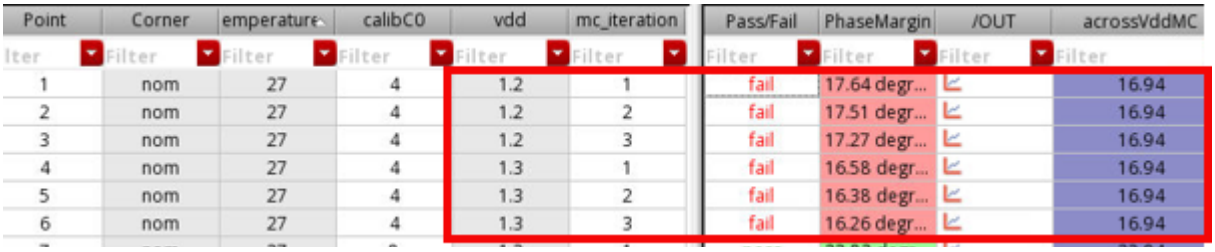
© 2023 Cadence Design Systems, Inc.

The `cross` function runs on the `PhaseMargin` result across all values of `calibC0` when other dimensions are constant. The resulting `calibVal` value remains constant for each varying value of `calibC0` while other dimensions remain constant. The same is visible in the `calibVal` column in the Detail-Transpose results view.



Point	Corner	calibC0	nc_iteration	temperatur	Pass/Fail	calibVal	PhaseMargin
1	nom	4	1	27	fail	6	17.64 degree
4	nom	8	1	27	pass	6	22.83 degree
7	nom	12	1	27	pass	6	25.43 degree
10	nom	16	1	27	pass	6	26.6 degree
13	nom	nom	1	27	pass	disabled	disabled
2	nom	4	2	27	fail	6	17.51 degree
5	nom	8	2	27	pass	6	22.68 degree

Similarly in the following example, the output `acrossVddMC` is measured across two dimensions, `vdd` and `mc_iteration`, and the resulting value is constant for all varying combinations of these two dimensions when other dimensions are constant.



Point	Corner	emperature	calibC0	vdd	mc_iteration	Pass/Fail	PhaseMargin	/OUT	acrossVddMC
1	nom	27	4	1.2	1	fail	17.64 degr...		16.94
2	nom	27	4	1.2	2	fail	17.51 degr...		16.94
3	nom	27	4	1.2	3	fail	17.27 degr...		16.94
4	nom	27	4	1.3	1	fail	16.58 degr...		16.94
5	nom	27	4	1.3	2	fail	16.38 degr...		16.94
6	nom	27	4	1.3	3	fail	16.26 degr...		16.94

For any invalid expression in `measureAcross`, the tool displays `eval error` in results.

Note: `calcVal` expressions can also refer to outputs that are measured across any dimension. However, you might need to use the `?matchParams` or `?getFirstSweepPoint` argument to use the desired result of that output. For example, `calcVal("calibVal" "AC_test" ?getFirstSweepPont t)` where `calibVal` is an expression with *EvalType* as `any`.

Related Topics

measureAcross

Creating Expressions to be Measured Across Any Dimension

© 2023 Cadence Design Systems, Inc.

Viewing and Editing Output Signals or Expressions in Calculator

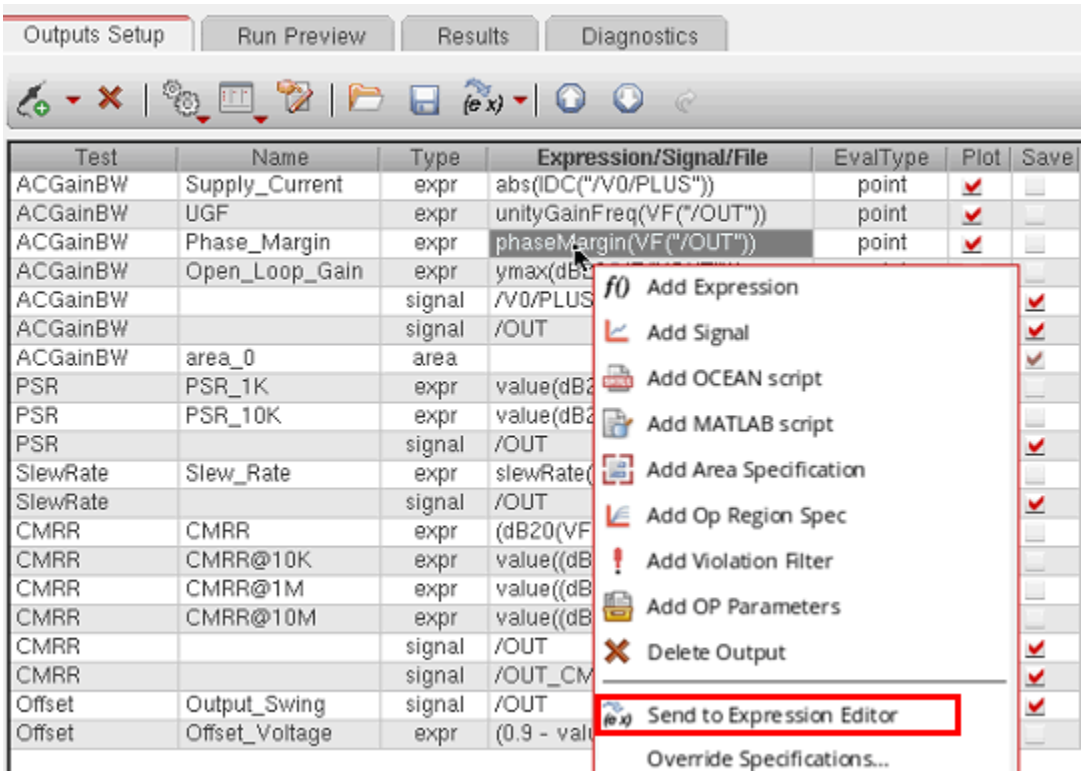
You can send the outputs from the Outputs Setup tab of the ADE Assembler Outputs pane to the Expression Editor in Virtuoso Visualization and Analysis XL Calculator. In the Calculator window, you can copy an expression from the Expression Editor to the buffer, make changes, and send it back to the Outputs pane.

Depending on your requirement, you can either send a single output (signal or expression), a set of outputs, or all the outputs to the Calculator.

Note: This feature is not supported for other output types such as OCEAN script, MATLAB script, area spec, violation filter, and .measure.

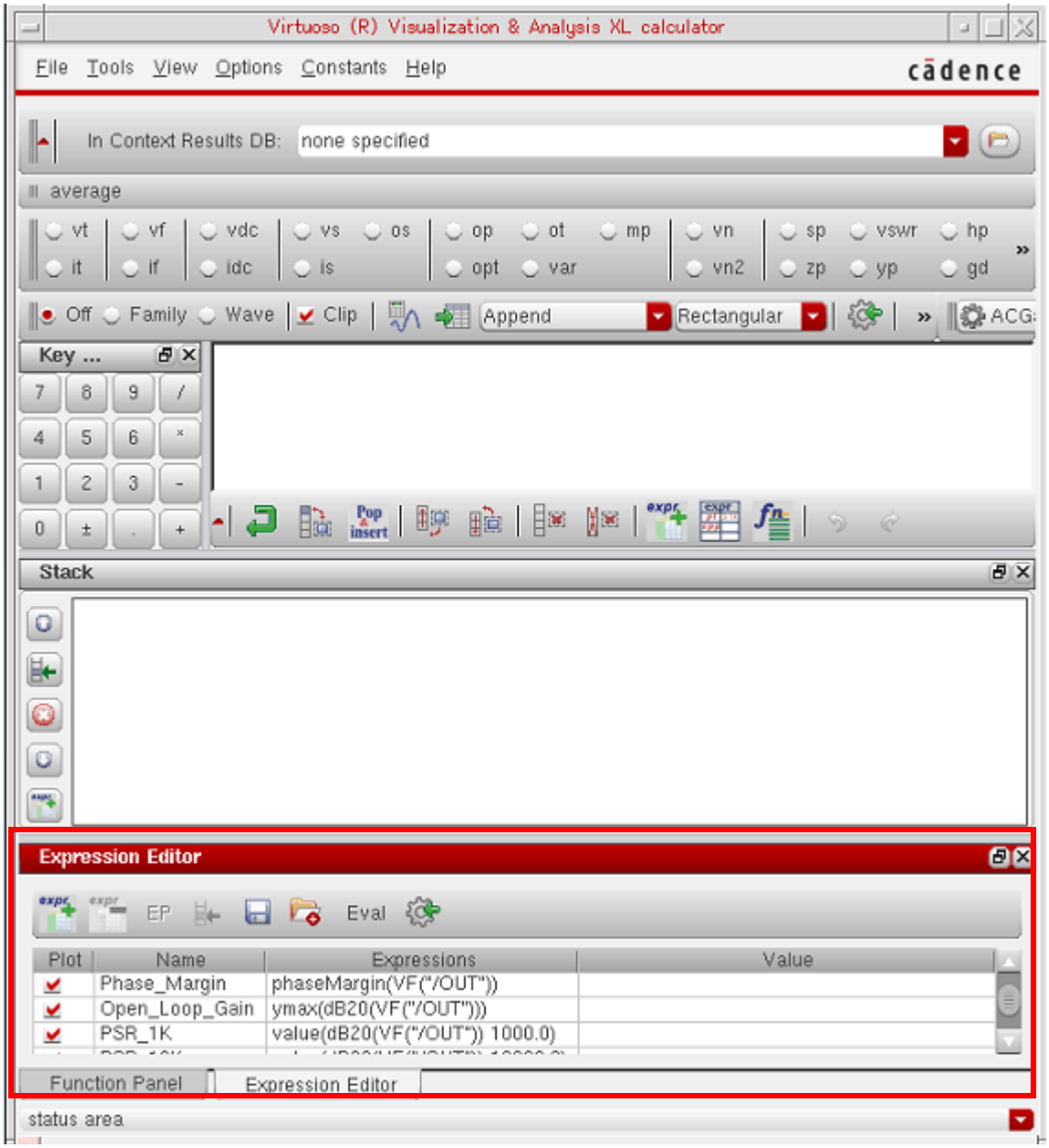
To send a single or a selected set of outputs to the Calculator, do the following:

1. On the Outputs Setup tab, press the `Ctrl` key and click the expressions or signals that you want to copy to the Calculator.
2. Right-click any one of the selected output and choose *Send to Expression Editor*.



© 2023 Cadence Design Systems, Inc.

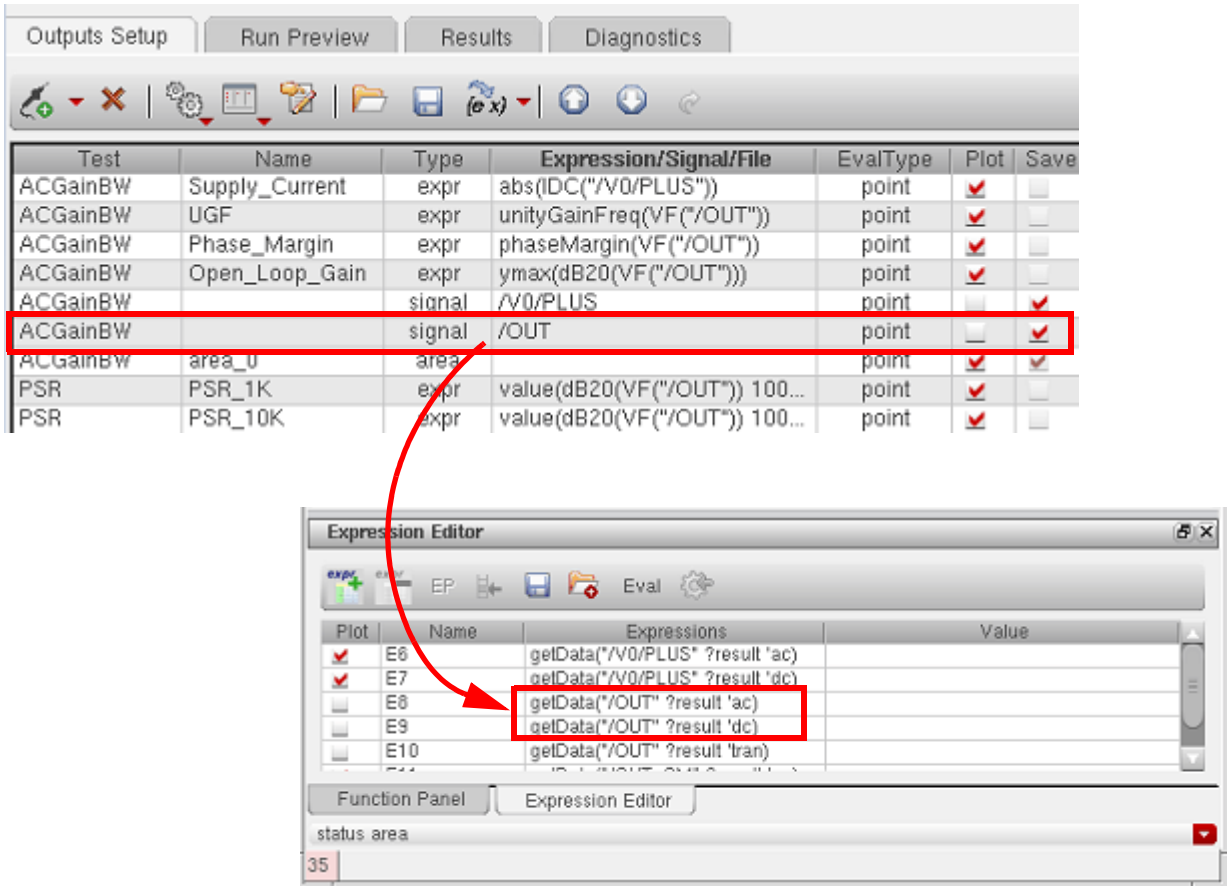
The selected outputs are copied to the Expression Editor in Virtuoso Visualization and Analysis XL Calculator window, as shown below.



If the Expression Editor is not visible, you might need to expand the Calculator window.

© 2023 Cadence Design Systems, Inc.

The names of expressions are the same as those mentioned in ADE Assembler. If no name is given for the source output in ADE Assembler, a default name `E<count>` is used, as shown in the following figure.



In the above figure, the signals have been copied from the ADE Assembler Outputs Setup tab to the Expression Editor. These signals have been assigned default names and they have been converted into expressions.

If more than one analysis is enabled for the ADE Assembler test, a separate expression is created to get the signal data from each analysis. In the example shown above, the test `ACGainBW` contains two tests, `ac` and `dc`. When the signal `/OUT` is sent to the Expression Editor, two separate expressions are created to get the result data for `/OUT` from the two analysis. The `?result` argument indicates the name of the analysis from which the data is taken.

To send all the output expressions or signals from the ADE Assembler Outputs Setup tab to the Expression Editor in Calculator, do the following:

© 2023 Cadence Design Systems, Inc.

1. Click the drop-down arrow to the right of the *Send to Expression Editor* command to view the command options.

   By default, *Expressions* is selected, which implies that only the expressions will be copied to the Calculator.

2. To copy all the expressions and signals, select *All*.

   Note: You can change the default selection by setting the value of the sendOutputsToEEFilter environment variable.

3. Click *Send Outputs to Expression Editor*.

   All the expressions and signals, are copied to the Expression Editor in Calculator.

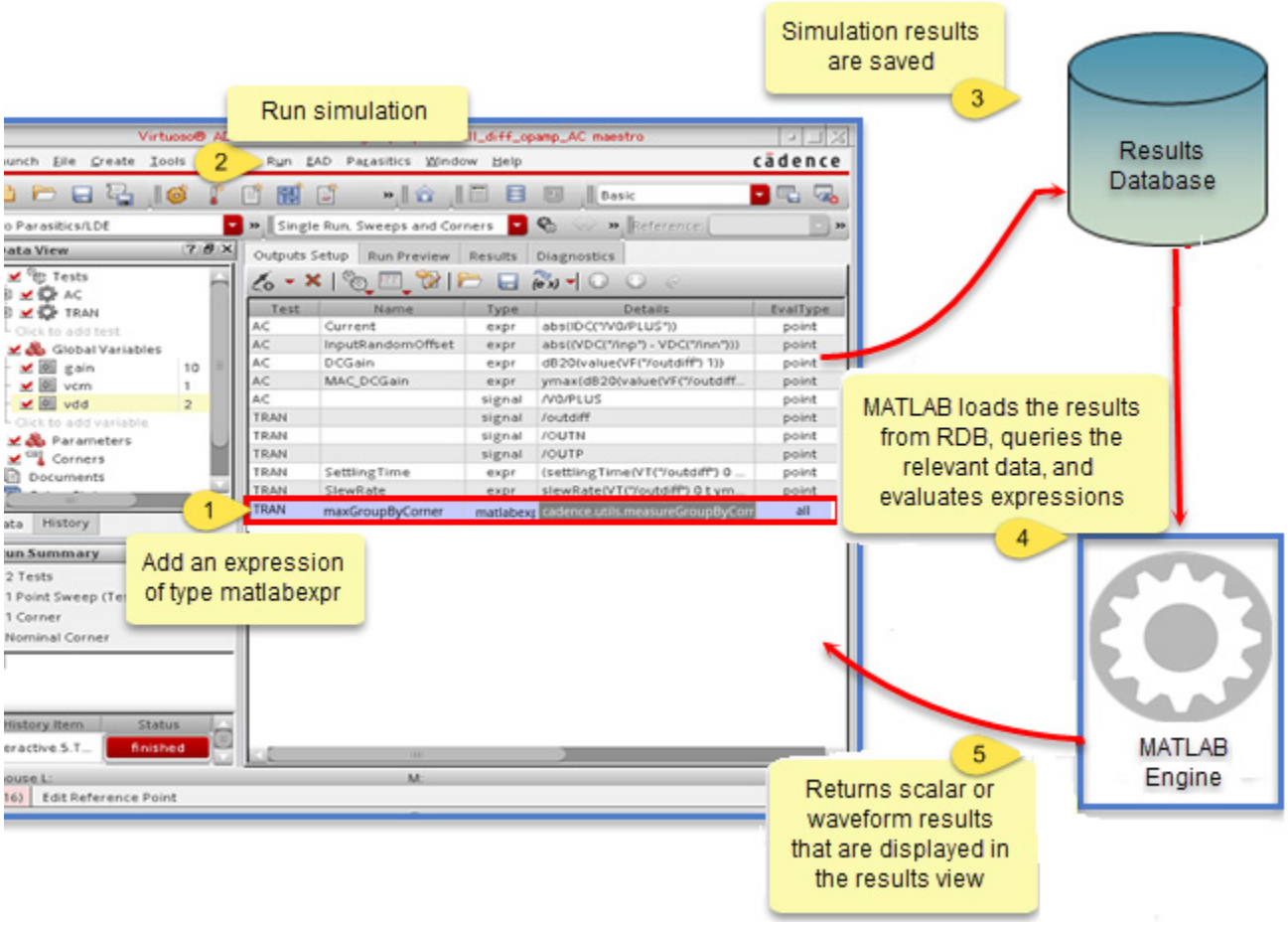
   Note: The *Send to Expression Editor* command is also available on the right-click menu of the Output Setup pane of the ADE Explorer.

© 2023 Cadence Design Systems, Inc.

Using MATLAB Expressions in Outputs

Fast, accurate, and efficient analysis of simulation results is one of the key requirements for circuit simulation and verification. Integration of MATLAB® with ADE Assembler provides the capability to apply various data slicing and querying methods for advanced data analysis and waveform processing using a high-level language.

To use MATLAB® in ADE Assembler, you can add MATLAB expressions as outputs that are evaluated in the MATLAB engine. The MATLAB engine runs independent of ADE Assembler, but there is a channel through which ADE Assembler exposes the circuit simulation data. This provides MATLAB a capability to access both the scalar and waveform data that can be used in measurements for single points, sweeps or corners as well as for Monte Carlo analysis. The results of these expressions are returned by MATLAB and displayed on the *Results* tab in ADE Assembler.



Note: To evaluate MATLAB expressions, you need to run MATLAB vR2020b or later. You also require the Virtuoso MATLAB Integration Option (95252) license.

© 2023 Cadence Design Systems, Inc.

The following topics describe how you can use MATLAB expressions in ADE Assembler:

- Adding Outputs of Type MATLAB Expression in ADE Assembler
- Using an Interactive MATLAB Session Opened from ADE Assembler
- Evaluating MATLAB Expressions Across All Dimensions
- Evaluating MATLAB Expressions for a Data Point
- Examples of MATLAB Expressions
- Debugging MATLAB Expressions in the MATLAB Window

Adding Outputs of Type MATLAB Expression in ADE Assembler

To add a MATLAB expression in ADE Assembler, perform the following steps:

1. On the Outputs Setup tab, click , and then the test for which you need to add the expression.
2. Choose *MATLAB Expression* from the drop-down list.

   A new row with `matlabexpr` as *Type* is added to the outputs table.
3. Double-click the *Details* column for this row to enable editing.
4. Enter a MATLAB expression or a function already defined in MATLAB. This function can be a built-in function in the MATLAB engine or the additional functions provided by Virtuoso for integration of MATLAB with ADE. If you need to perform a custom calculation, you can also define a function in the MATLAB window and use that in the MATLAB expression.

Alternatively, click *Open MATLAB* (…) in the *Details* column to open the MATLAB window. Create a MATLAB expression, copy it to the clipboard, and paste it in the new row added in **step 2**.

Note: By default, *EvalType* for MATLAB Expressions is set to `all`, which indicates that the MATLAB expression is to be evaluated after the simulation results for all the points are ready. These expressions can be measured for a single point or across all data points or corners. If an expression needs to be evaluated using the simulation results of a single design point, change the *EvalType* to `point`.

Related Environment Variables:

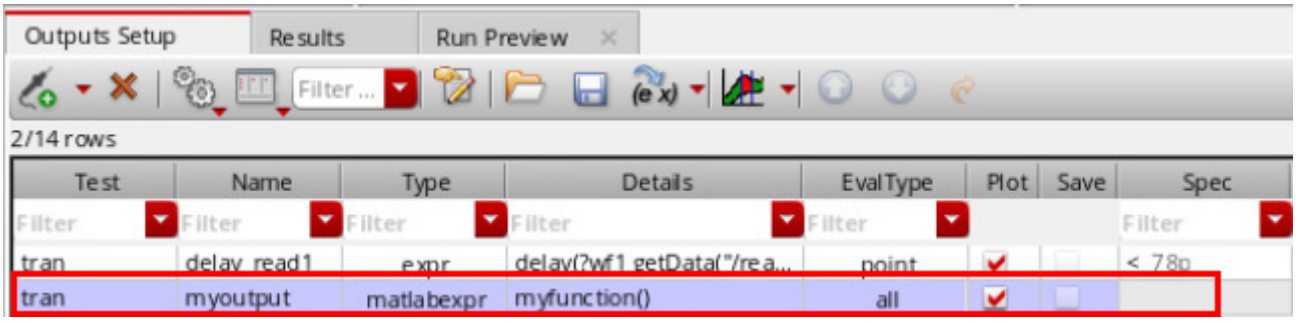
- startMatlabTimeout
- matlabIdleTimeout

© 2023 Cadence Design Systems, Inc.

Example: If you have defined a function in MATLAB, as shown below,

```
function result=myFunction()
        result=100
end
```

you can call this function from the `matlabexpr` type of output in ADE Assembler, as shown below.

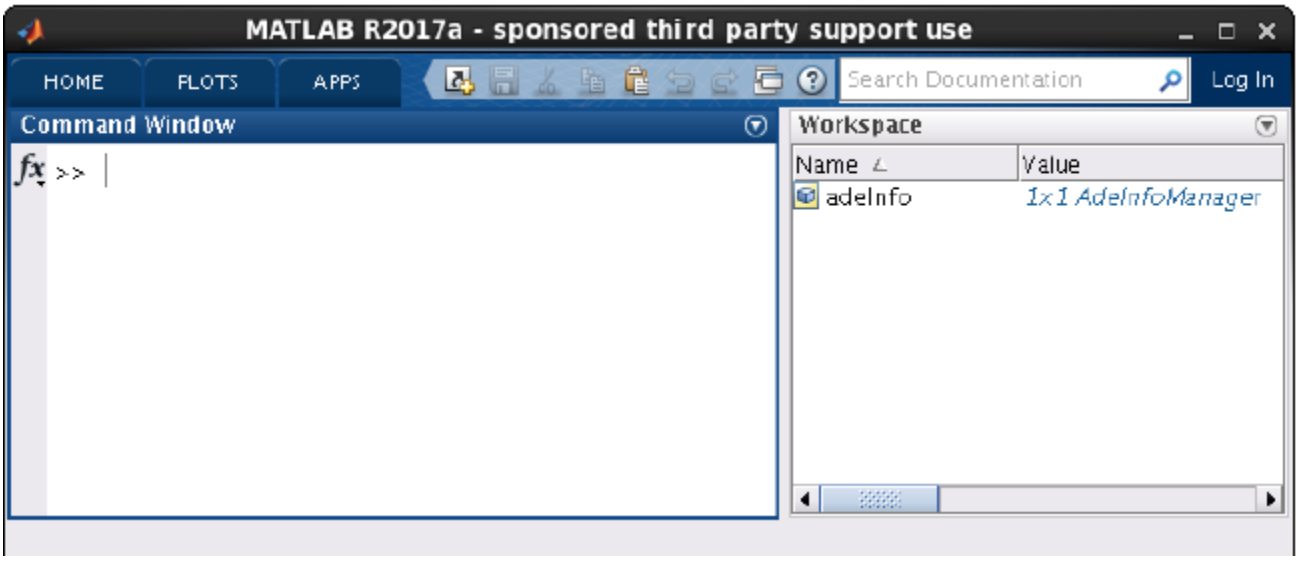


Using an Interactive MATLAB Session Opened from ADE Assembler

Instead of passing on MATLAB expressions to the MATLAB engine for evaluation during simulation run, you can open the MATLAB application from the ADE Explorer or ADE Assembler environment, and define a function or expression in that to query the already available simulation data.

For this, perform the following steps:

1. Click *Open MATLAB* ( ) on the toolbar of the Results tab.



© 2023 Cadence Design Systems, Inc.

Note the following:

- The following default libraries are imported in this interactive session:
  - `cadence.*`
  - `cadence.srrdata.*`
  - `cadence.calculator.*`
  - `cadence.help`
- The `adeInfo` object is automatically added to MATLAB. This is an object of the AdeInfoManager class defined for the integration of ADE with MATLAB. It contains a reference to the current history in ADE Assembler.

2. Use the properties and functions of the `adeInfo` object to access the history results.

   For example, you can define a function to use the simulation results as shown below.

```
function result=myMeasurement()
  adeInfo.loadResult;
  rdb = adeInfo.adeRDB;
  rdb.corners;
  ...
  ...
end
```

Tip

By default, the result context for the `loadResult` function in the MATLAB window is set to the current ADE simulation history. You can change the context to another history by setting the history argument of the `loadResult` function.

```
adeInfo.loadResult('history', 'Interactive.2')
```

Important

The functions used in a MATLAB expression must be found in the location specified by the MATLABPATH environment variable.

3. Save the function in a file. By default, the file is saved in the MATLAB working directory, which is the same as the current working directory for Virtuoso. The MATLAB files are saved with the `.m` extension.

   A MATLAB function can return the following types of values to ADE Assembler:

   - **Scalar values:** A single value is returned. When a MATLAB expression that returns a single value is added as an output expression, a single value is returned for each point and displayed on the Results tab of ADE Assembler.

© 2023 Cadence Design Systems, Inc.

- W**aveform objects:** When a MATLAB expression that returns a single waveform is added as an output expression in ADE Assembler, a waveform icon is displayed for each point on the Results tab.

  If you run a MATLAB function from the MATLAB window, you can use the `cdsPlot` function, which is included in the `cadence.utils` package, to plot a waveform result, as shown below.

  ```
  import cadence.utils.cdsPlot
  %Load results of simulation data point #1
  adeInfo.loadResult('test','tran','DataPoint', 1);
  wave = VT('/I0/I2/a');
  cdsPlot(wave)
  ```

  If the returned value contains multiple waveforms, you can refer to the individual waveform by using the dot (.) operator.

  ```
  cdsPlot(waves.wave(1))
  cdsPlot(waves.wave(2))
  ```

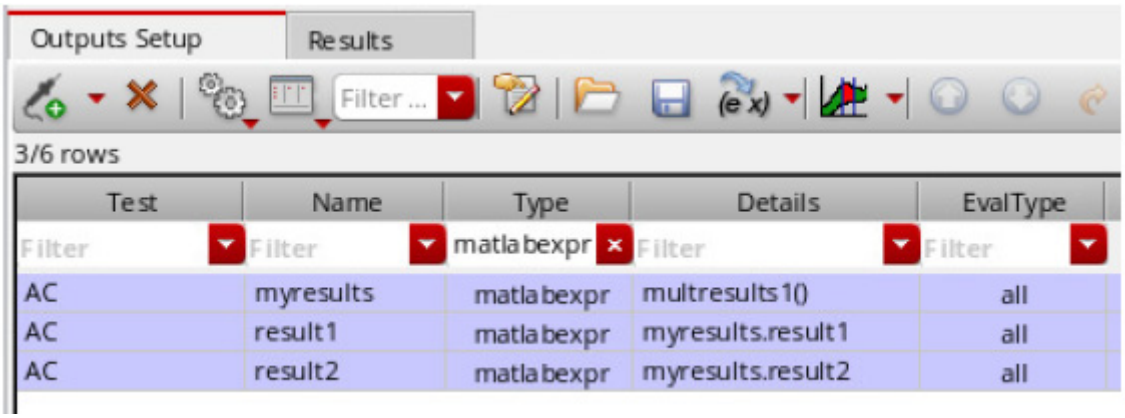
- **String Values:** These are helpful in returning string values from a MATLAB expression.

- **MATLAB structs:** These are helpful in returning multiple results from a single MATLAB function call. You need to use the dot (.) operator to access individual values from the struct.
  In case of errors, the MATLAB expressions return `Error`.

  A sample MATLAB function that returns multiple values is given below.

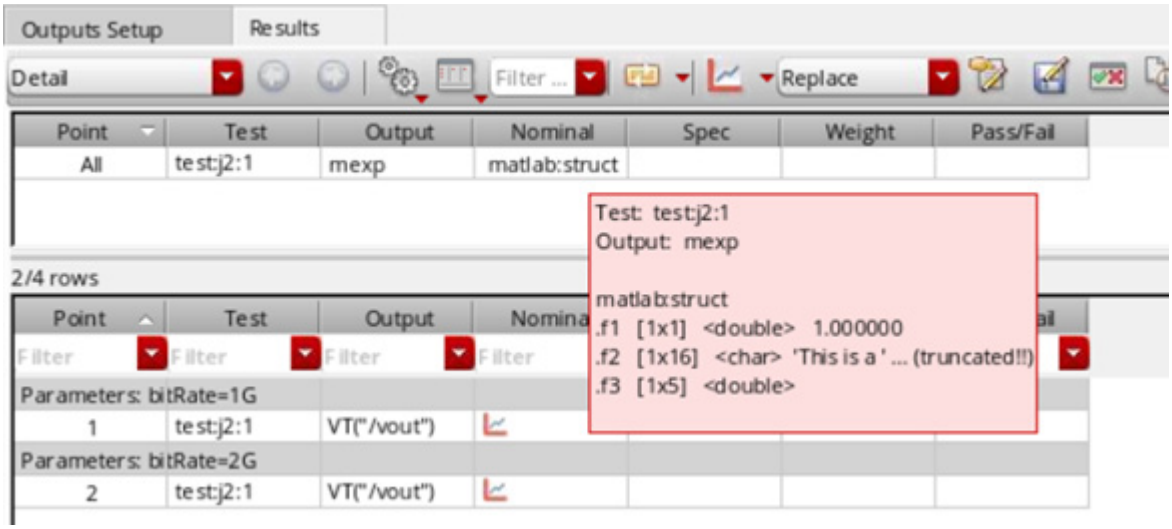
  ```
  function result = multresults1()
    res1 = 1;
    res2 = 200;
    result = struct('result1', res1, 'result2', res2);
  end
  ```

  To refer to individual values in the struct returned by this object, you need separate expressions on the Outputs table of ADE Assembler, as shown below.



© 2023 Cadence Design Systems, Inc.

To get a hint about the individual members of a struct, move the mouse over a struct type of result. A tooltip displays the details of the struct, as shown below.



For more example, refer to Examples of MATLAB Expressions.

© 2023 Cadence Design Systems, Inc.

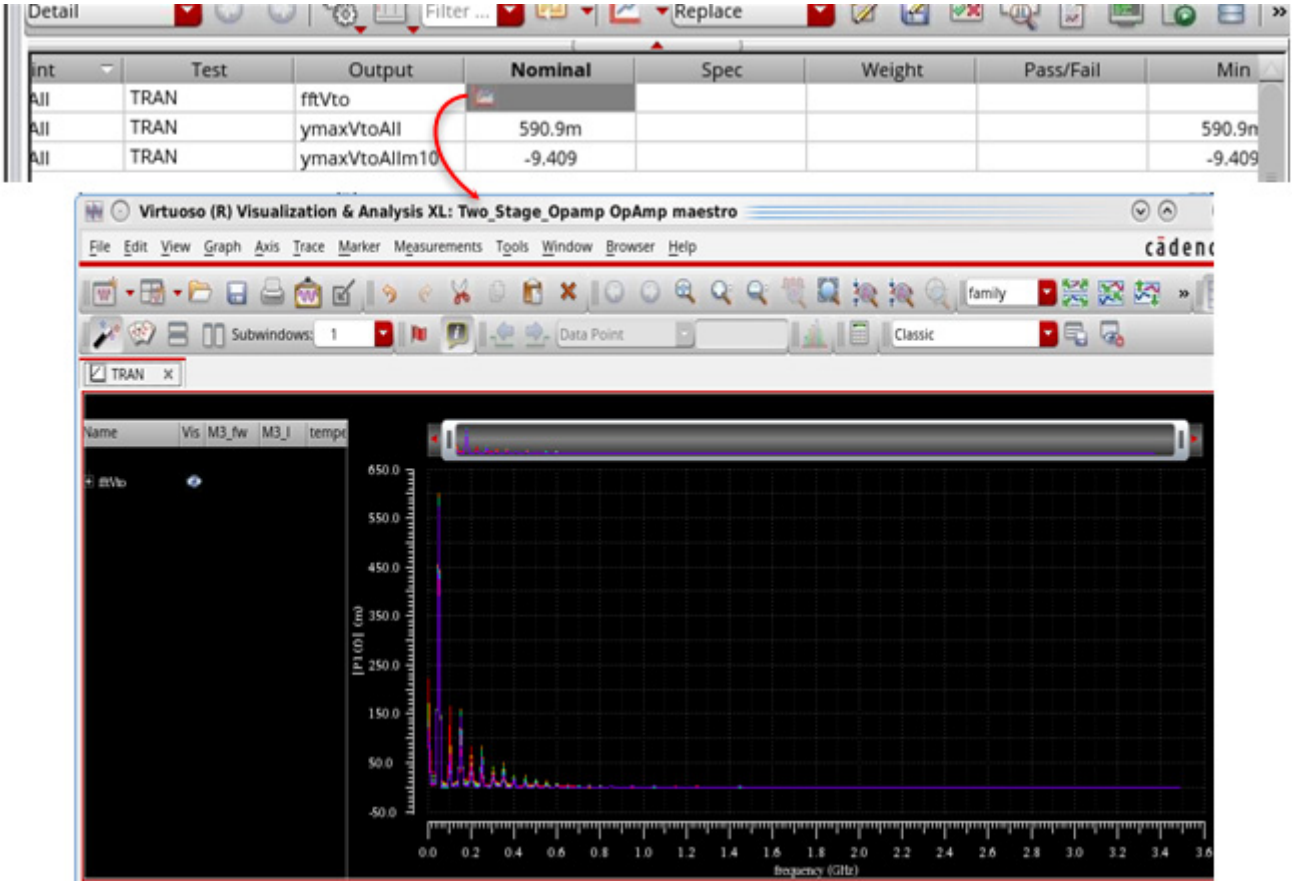
Evaluating MATLAB Expressions Across All Dimensions

By default, the MATLAB expressions have *EvalType* set to `all`. Therefore, these expressions are evaluated across all dimensions, and their results are displayed in a separate section at the top of the *Results* tab, as shown below.

Outputs in the Outputs Setup tab

TRAN	fftVto	matlabexpr	familyfun(@(w) singleSidedAmplitudeSpectrum(w, 7.0000e+...	all
TRAN	calcVal_ymaxVfo	expr	(calcVal("ymaxVfo" "AC") + 1)	point
TRAN	calcVal_cv	expr	VAR("cv")	point
TRAN	ymaxVtoAll	matlabexpr	ymax(VT("/OUT"))	all
TRAN	ymaxVtoAllm10	matlabexpr	ymaxVtoAll - 10	all

Results in the Results tab



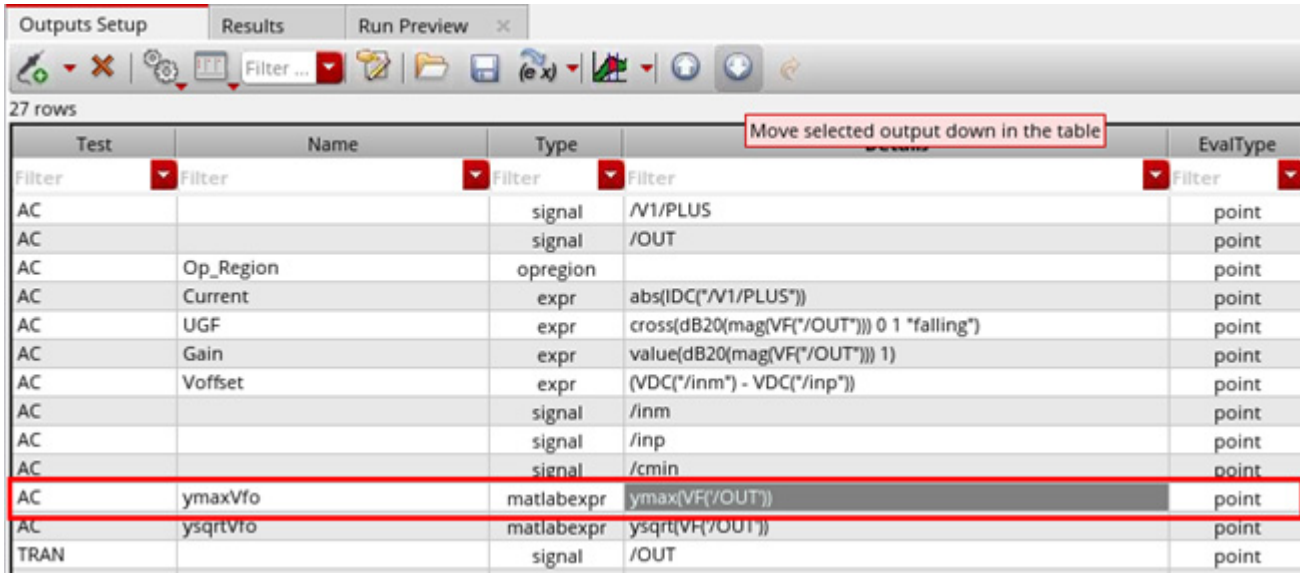
© 2023 Cadence Design Systems, Inc.

Important

The outputs to be measured across all are evaluated only after the simulation run is complete, and the results for all the points and corners are available.

Evaluating MATLAB Expressions for a Data Point

Similar to other output types, if a MATLAB expression is to be evaluated for a data point, set the *EvalType* value for that expression to `point`, as shown in the example given below.

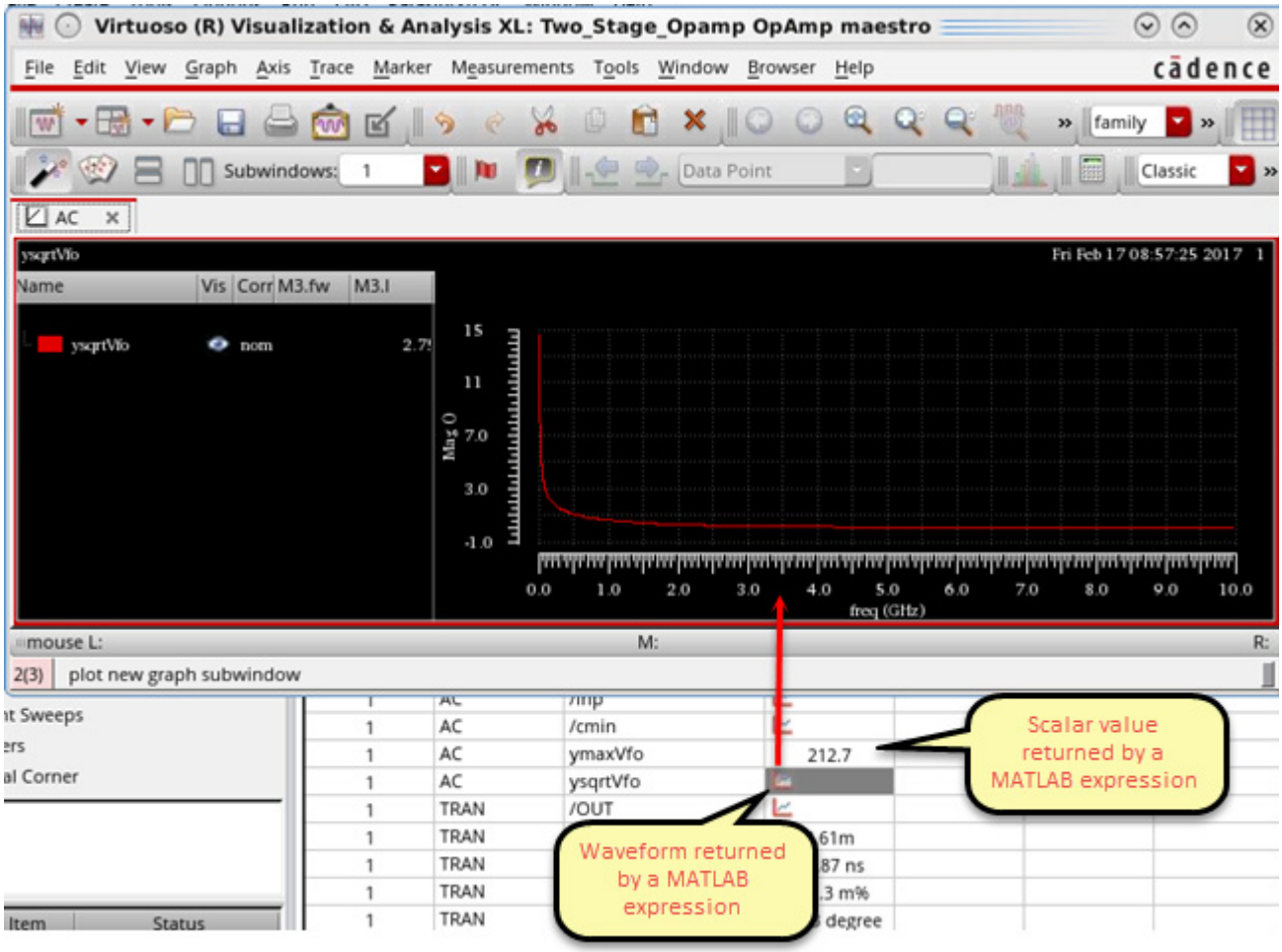


Test	Name	Type	Details	EvalType
Filter	Filter	Filter	Filter	Filter
AC		signal	/V1/PLUS	point
AC		signal	/OUT	point
AC	Op_Region	opregion		point
AC	Current	expr	abs(IDC("/V1/PLUS"))	point
AC	UGF	expr	cross(dB20(mag(VF("/OUT"))) 0 1 "falling")	point
AC	Gain	expr	value(dB20(mag(VF("/OUT"))) 1)	point
AC	Voffset	expr	(VDC("/inm") - VDC("/inp"))	point
AC		signal	/inm	point
AC		signal	/inp	point
AC		signal	/cmin	point
AC	ymaxVfo	matlabexpr	ymax(VF("/OUT"))	point
AC	ysqrtVfo	matlabexpr	ysqrt(VF("/OUT"))	point
TRAN		signal	/OUT	point

While running a simulation, ADE sends this expression to the MATLAB engine and also shares the reference of the current test and data point for the expression. In addition, it sets the context for the session and history in MATLAB to the current session and history in ADE Assembler.

© 2023 Cadence Design Systems, Inc.

If MATLAB returns a scalar value, the value is displayed on the *Results* tab. If MATLAB returns a waveform, a wave icon is displayed. Click that icon to view the result, as shown below.



Important

The outputs to be measured for a single data point are evaluated only after each point is simulated. It is possible that when an expression is evaluated for a data point, the results for that point are not available, which can be due to multiple reasons, such as the scheduled job to run the simulation for that point is waiting or is not complete, or the results are not received from a remote computer. In such cases, the expression might return an error. It is recommended to be careful when using `point` as an eval type for MATLAB expressions.

© 2023 Cadence Design Systems, Inc.

Related Environment Variables

- matlabExprReevalSearchUpdateLevel
- matlabStartTimeout
- matlabIdleTimeout

Examples of MATLAB Expressions

A MATLAB expression can be used to perform different types of calculations on the input data. A few examples are:

- To use the scalar result of an output in another output using dependent expressions
- To return the results of multiple corners or sweep points in the current history
- To return selective values from the results by using query constraints
- To use the results from other tests in the same history
- To use the results of another simulation saved in a different history

To create these MATLAB expressions, you can use the built-in functions in the MATLAB engine or the additional functions provided by Virtuoso for integration of MATLAB with ADE. The additional functions are provided in a package named `cadence`, which contains various sub-packages. You can also define custom functions using the available functions.

The following table shows a summary of the packages included in the `cadence` package.

	cadence		
srrdata	**Query**	**calculator**	**utils**
Contains functions to access data	Contains constraints that you can use to filter results based on certain criteria	Contains calculator functions for waveforms	Contains special functions to query or group simulation results saved in RDB

The following examples show how you can create MATLAB expressions for the different types of calculations:

© 2023 Cadence Design Systems, Inc.

Example 1: MATLAB Expression that Uses a Waveform from the Current Simulation

To access the waveform of a signal, say `/OUT`, from the current simulation results, use the `VF` function.

```
VF('/OUT')
; returns a waveform for each data point
```

For a waveform value returned by a MATLAB expression or function, a waveform icon is displayed on the ADE Results tab. Similar to the other waveform results, you can double-click that icon to view the plot. However, if you are running the command in the MATLAB window, you can use the `cdsPlot` function of the c`adence.utils.cdsPlot` library to display the graphs.

Now, to get the maximum Y-axis value for waveform `/OUT`, you need to use `VF('/OUT')` inside `ymax` function.

```
ymax(VF('/OUT'))
; returns a scalar for each waveform returned for each data point
```

In the example given above, `VF` returns a waveform of the ac sweep net voltage. This waveform is an input for `ymax`, which returns a scalar value—the maximum value on the Y-axis of a waveform.

Note: The `ymax(VF('/OUT'))` expression is equivalent to `max(VF('/OUT').y)`, where `VF('/OUT')` returns an object of `cadence.Waveform` and `VF('/OUT').y` returns the Y-axis data of `VF('/OUT')` in the form of a column vector in MATLAB. `max`, which is a native MATLAB function, returns the maximum value from that column vector.

`VF` is defined in the `cadence.srrdata` package and `ymax` is defined in `cadence.calculator`. Both the packages are imported by ADE while sending a request to MATLAB for expression evaluation.

In addition, ADE internally shares with MATLAB the details of the current ADE session. It also calls the `cadence.AdeInfoManager.loadResult` function to load the simulation results for the current test and the current data point by internally running the following commands:

```
import cadence.srrdata.*
import cadence.utils.cdsPlot
%Load results of simulation data point #1
adeInfo.loadResult('test','<name-of-current-test>','DataPoint', 1);
```

Therefore, by default, an expression is evaluated for the current point.

For more details about `loadResult`, run the `adeMatlabHelp` command in the MATLAB window to open the help describing the integration of MATLAB with ADE, and read the *Loading Simulation Results* section.

© 2023 Cadence Design Systems, Inc.

Note: As the commands given above are run internally by ADE while sending a request to MATLAB, you can avoid adding these explicitly in your MATLAB expressions or custom functions written in ADE.

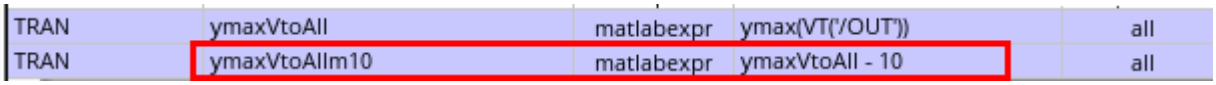
Tip

To read more about a package in MATLAB, use the `doc <package-name>` command in the MATLAB window.

Example 2: MATLAB Expression that Uses a Scalar Value from Another Output in the Current Simulation

If you need to access a scalar value of another output from the current simulation result, you can create a dependent MATLAB expression in the same way as for other types of expressions.

In the example shown below, `ymaxVtoAllm10` is dependent on the result of a pre-defined MATLAB function `ymaxVtoAll`.



In dependent expressions, ADE automatically analyzes the dependency between MATLAB expressions, and evaluates those according to their dependency order.

Important

MATLAB expressions are independent of SKILL expressions. They cannot be used together in a MATLAB function.

Example 3: A MATLAB Expression to Group the Results of Sweep Points

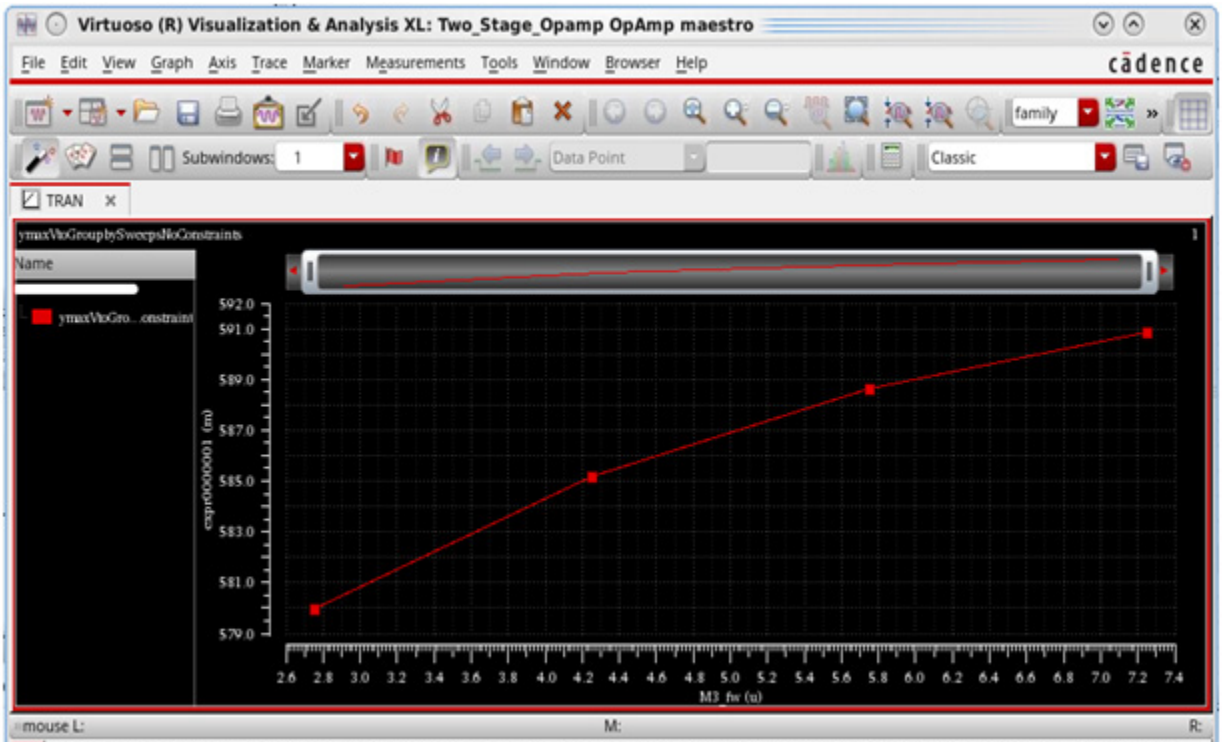
You can use the functions provided in the `cadence.utils` package to work on the results of sweep points.

For example, you run a simulation by sweeping the values of parameter `M3.fw`. To find `ymax` of output `/OUT` for certain specific sweep points, you can create a MATLAB expression as shown below.

```
cadence.utils.measureWaveGroupBySweeps( @ymax, @VT, '/OUT', 'M3.fw', 'test',
'TRAN')
```


© 2023 Cadence Design Systems, Inc.

The above example returns a waveform that plots the `ymax` value of `/OUT` for each value of `M3.fw`.



Example 4: To return selective values from the results by using query constraints

For the case explained in example 3, if you need to consider only selected data points that meet the following criteria:

- `temperature` is `0`
- `M3.l` equals to `350e-09`
- `M3.fw` is not equal to `4.25e-06`

To run the `measureWaveGroupBySweeps` function on a selected set of values from the simulation results, you can specify query constraints, as shown below.

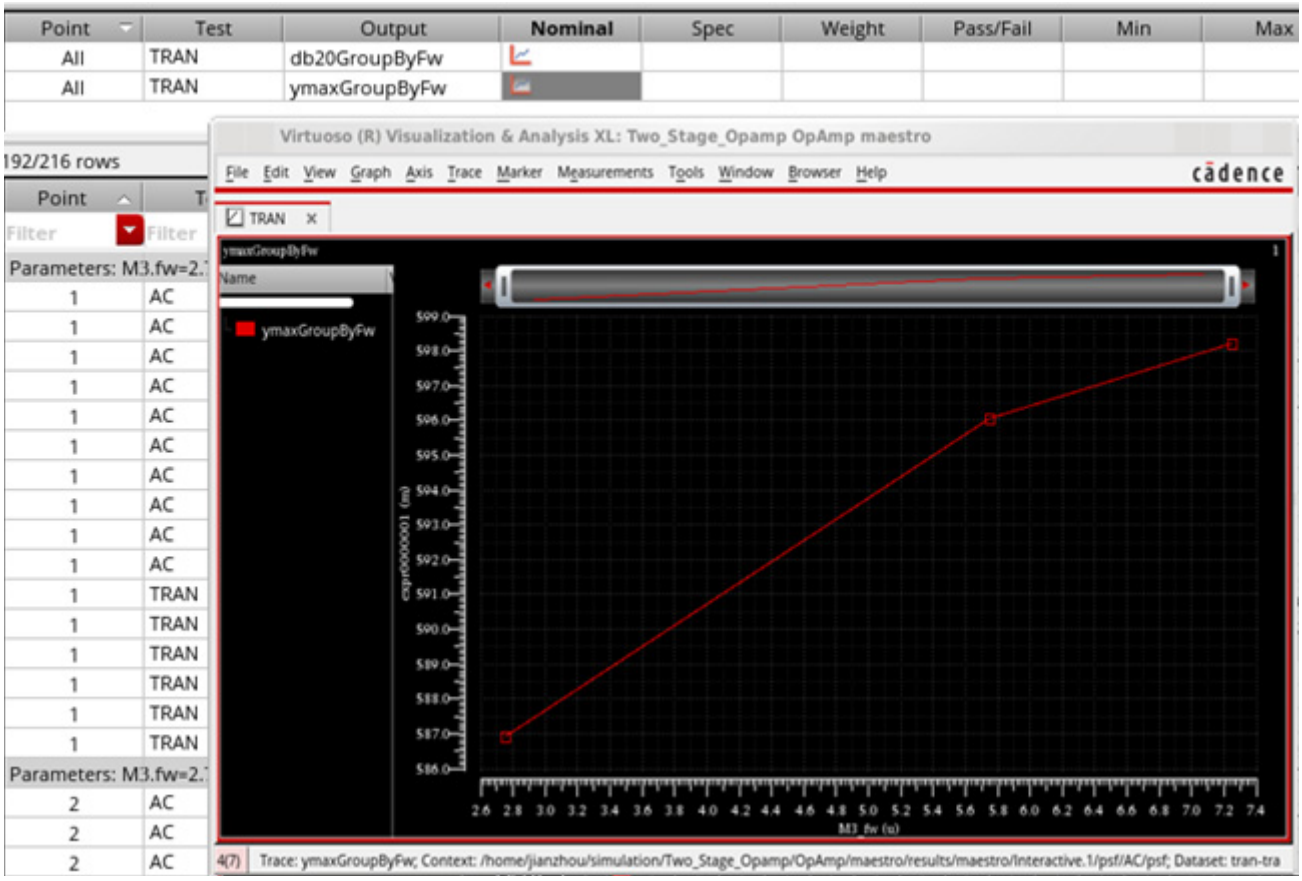
```
cadence.utils.measureWaveGroupBySweeps( @ymax, @VT, '/OUT', 'test', 'TRAN' ...
      'constraints', cadence.Query.Parameter('temperature') == 0 &
      cadence.Query.Parameter('M3.l') == 350e-09 & ...
      cadence.Query.Parameter('M3.fw') ~= 4.25e-06)
```

The `measureWaveGroupBySweeps` function is run for all the data points. Therefore, MATLAB executes the function only after the entire simulation run is complete. When ADE calls the `cadence.AdeInfoManager.loadResult` function, it sets the context to the

© 2023 Cadence Design Systems, Inc.

current ADE session and the current test. The context for data point is set to -1, which implies that the calculation is to be done across all.

The example returns a waveform that plots the ymax value of `/OUT` for each value of `M3.fw` where the given criteria is met.



Additional reference materials:

- For more details about a query constraint, run the following command in the MATLAB window:

  ```
  doc cadence.Query
  ```

- For more details about `measureWaveGroupBySweeps`, run the following command in the MATLAB window:

  ```
  doc cadence.utils
  ```

- To know about the internal implementation details of `measureWaveGroupBySweeps`, run the `adeMatlabHelp` command to open the help document on MATLAB integration and read the example on grouping waveform results for sweep variables.

© 2023 Cadence Design Systems, Inc.

Example 5: A MATLAB Expression to Perform Calculations on the Grouped Results of Corners

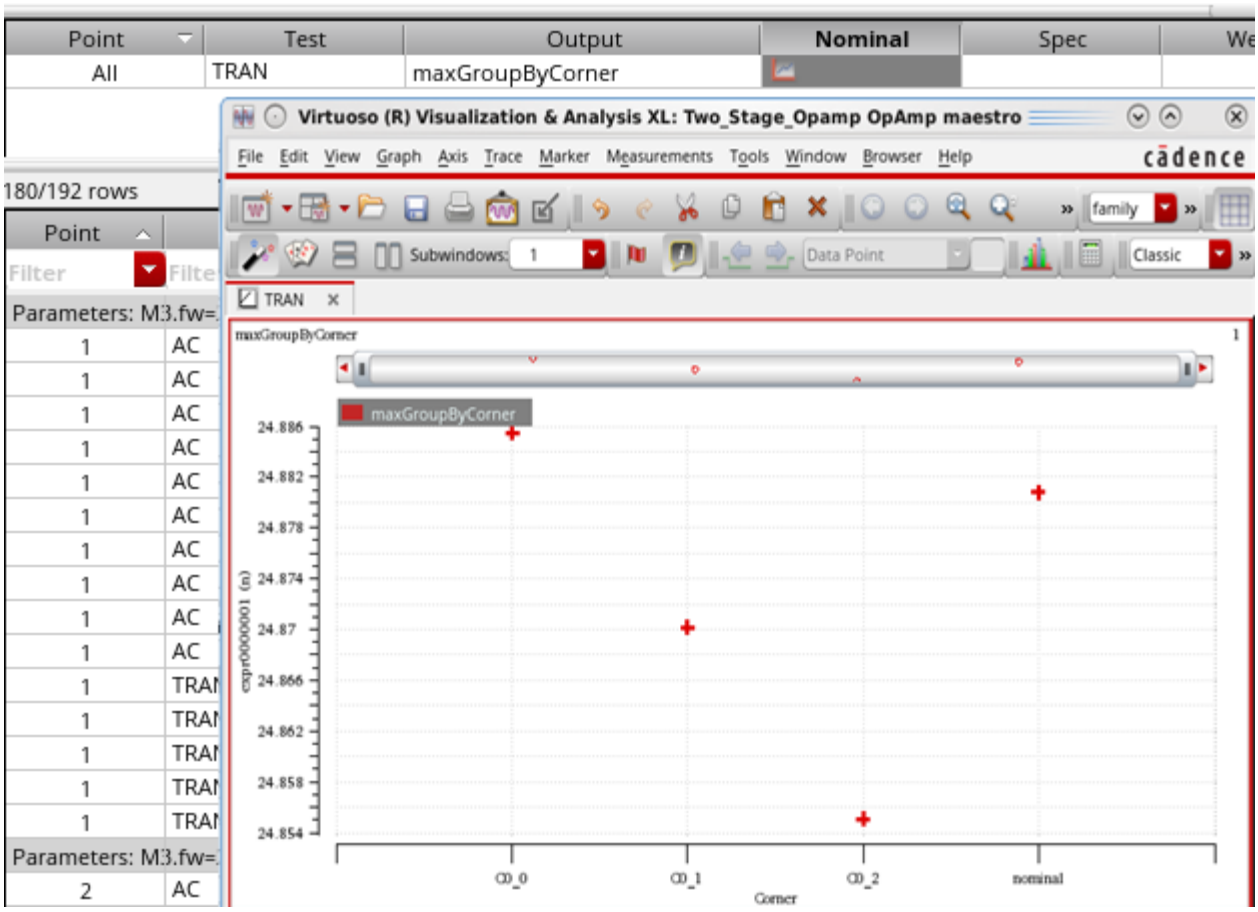
If you need to group the scalar results for corners and then perform some mathematical calculations on the grouped data, you can use the `cadence.utils.measureGroupByCorners` function.

For example, you need to find the maximum value for an output, `SettlingTime`, across all the data points in each corner group. For this, you can use the `measureGroupByCorners` function, as shown below.

```
cadence.utils.measureGroupByCorners(@max, 'constraints', cadence.Query.Output()
== 'SettlingTime');
```

In `measureGroupbyCorners` function runs on scalar values saved in the results database. Similar to `measureWaveFormsBySweeps`, for this function too, ADE sets the context to the current ADE session and the current test.

The example returns a waveform that plots the `max` value of `SettlingTime` for each corner, as shown below.



© 2023 Cadence Design Systems, Inc.

Additional reference materials:

- For more details about query constraint, run the `doc Query` command in the MATLAB window.
- For more details about `measureGroupByCorners`, run the `doc utils` command in the MATLAB window.
- To know about the steps performed by `measureGroupByCorners`, run the `adeMatlabHelp` command to open the help document on MATLAB integration and read the example on grouping results for corners.

Note: For more details and examples, refer to the Virtuoso ADE MATLAB Integration rapid adoption kit on Cadence Online Support.

Debugging MATLAB Expressions in the MATLAB Window

If the simulation results for a MATLAB expression do not return the required result or show errors, you can edit those and re-evaluate without running the simulation again. Expressions can be edited in the Outputs Setup pane or in the MATLAB window.

To edit and debug an expression in the MATLAB window:

- When debugging a result in the *Detail* results view, right-click the result value and choose *Open MATLAB*.

  In this case, ADE sets the context to the current session, history, test, and data point. You can access the other output values for that point, and verify the result while editing the expression.

- When debugging a result in the *Detail-Transpose* results view, click *Open MATLAB* on the toolbar of the Results tab.

  In this case, ADE sets the context to the current session and history. The test is set to `""` and the data point is set to `-1`.

You can also access operating point parameters from an extracted view in MATLAB.

For example, after launching MATLAB through the *Open MATLAB* option on the Results toolbar, let us run the following commands.

```
import cadence.srrdata.*
adeInfo.loadResult('datapoint',1,'test','testLib:buf_sim:1');
psfdir = adeInfo.adeCurrentResultsPath
    cds_srr(psfdir)
    dcinfo=cds_srr(psfdir,'dcOpInfo-info')
```


© 2023 Cadence Design Systems, Inc.

```
    op = dcinfo.bsim4;  %This is a cell array in this testcase
```

In case you wish to filter a large cell array, you can also use the following commands.

```
opi1 = find(contains(op, 'PM0'));  %Find index that contains PM0
opi2 = find(contains(op, 'vdsat'));  %Find index that contains vdsat
opi = intersect(opi1, opi2); % index common to opi1 and opi2
opf = op(opi)  %Filtered list
val1 = DATA('I0.PM0\(1\):vdsat', 'dcOpInfo-info', 'OP')
```

Or

```
val1 = getData('I0.PM0\(1\):vdsat', 'result', 'dcOpInfo')
val2 = DATA('I0.PM0\(0\)_1__rcx:vdsat', 'dcOpInfo-info', 'OP')
```

Or

```
val2 = getData('I0.PM0\(0\)_1__rcx:vdsat', 'result', 'dcOpInfo-info')
```

On using the above commands, the following output is displayed.

```
mytest1
psfdir =
        '/home/user/simulation/testLib/buf_iter/maestro/results/maestro/
Interactive.0/1/testLib:buf_sim:1/psf'
    ans =
      11x1 cell array
        {'tran-tran'                }
        {'finalTimeOP-info'         }
        {'dcOp-dc'                  }
        {'dcOpInfo-info'            }
        {'modelParameter-info'      }
        {'element-info'             }
        {'outputParameter-info'     }
        {'designParamVals-info'     }
        {'primitives-info.primitives'}
        {'subckts-info.subckts'     }
        {'variables'                }
    dcinfo =
      struct with fields:
           signal_info: {1x14 cell}
                  prop: {23x1 cell}
            sweep_info: {}
                 bsim4: {1020x1 cell}
        bsource_2b273e: {60x1 cell}
             capacitor: {43x1 cell}
```


© 2023 Cadence Design Systems, Inc.

```
          resistor: {236x1 cell}
           vsource: {9x1 cell}
opf =
  8x1 cell array
    {'I0.PM0\(1\):vdsat'            }
    {'I0.PM0\(1\):vdsat_marg'       }
    {'I0.PM0\(1\)_1__rcx:vdsat'     }
    {'I0.PM0\(1\)_1__rcx:vdsat_marg'}
    {'I0.PM0\(0\):vdsat'            }
    {'I0.PM0\(0\):vdsat_marg'       }
    {'I0.PM0\(0\)_1__rcx:vdsat'     }
    {'I0.PM0\(0\)_1__rcx:vdsat_marg'}
val1 =
   -0.0481
val1 =
   -0.0481
val2 =
   -0.0481
val2 =
   -0.0481
```


© 2023 Cadence Design Systems, Inc.

Loading an OCEAN or a MATLAB Measurement

Note: See `axlOutputResult` in the *Virtuoso ADE SKILL Reference*. See the *OCEAN Reference* for information about OCEAN script commands.

To load an OCEAN script file containing one or more output measurements or a MATLAB measurement defined in a script file (one value per script) for a test, do the following:

1. On the Outputs Setup tab, click the button.
2. In the drop-down list, select a test and choose *OCEAN script* or *MATLAB script*.

   A new row is added for the test with the output type *ocean* or *matlab*.

Tip

You can double-click the test name or output type to select a different test or output type from the drop-down list that appears in the *Test* or *Type* column.

3. (Optional) In the *Name* field, type a name to represent the measure.
4. Double-click the *Details* field.
5. Type the name of (and location/path to) the script file.

Tip

Alternatively, you can click the browse button to open a browser window so that you can browse to locate a file. After you select a file and click *Open*, the relative path to the file appears in the *Details* field.

Note: Once the script file name appears in this field, you can edit it by clicking the Edit File button .

See also

- Editing an OCEAN or a MATLAB Script File on page 431
- Writing a MATLAB Script on page 431
- Specifying Whether a Result Will Be Saved or Plotted on page 366
- Searching for Conditional Results on page 1243

© 2023 Cadence Design Systems, Inc.

Editing an OCEAN or a MATLAB Script File

To edit an OCEAN or a MATLAB measurement in a script file, do the following:

1. On the Outputs Setup tab, double-click the OCEAN or MATLAB script file you want to edit in the *Details* field.
2. Click clicking the *Edit File* E button.

   The script file appears in a text editor.

   Note: By default, ADE Assembler opens the text editor set by using the `$EDITOR` shell variable in UNIX. You can specify a preferred text editor by using the `editor` SKILL variable.
3. Edit the script file and save your changes.

   The program uses the new script the next time you run the test.

Writing a MATLAB Script

When writing a MATLAB measure, you create a script in a `.m` file. You can use additional tools such as the `cds_srr` function provided in the Spectre/RF MATLAB Toolbox[1] to read the results data. You can also use the `axlCurrentResultsPath` special function in your MATLAB script file to determine the current ADE Assembler results path. If you want your script to write results to the Results tab of the Outputs pane, you must assign the value to the `axlResult` variable. You can plot your results in MATLAB if you include a MATLAB `plot` command in your script file.

To write a MATLAB measure for a test, do the following:

1. Use a text editor to create a `.m` file.
2. (Optional) Use the `cds_srr` function to access results data as follows:

   ```
   cds_srr( 'path_to_psf_dir', 'analysisName', 'outputVariable' )
   ```

1. Installed in your MMSIM 6.x hierarchy under `tools/spectre/matlab`.

© 2023 Cadence Design Systems, Inc.

where

path_to_psf_dir is the path to the `psf` directory; for example:

```
simulation/ampTest/schematic/psf
```

Tip

You can use the `axlCurrentResultsPath` function (instead of `'path_to_psf_dir'`) as follows to return the current ADE Assembler results path:

```
cds_srr( axlCurrentResultsPath, … )
```

analysisName is the analysis name string that corresponds to the `psf` subdirectory that contains the results; for example:

```
tran-tran
```

outputVariable is the output variable name

For example:

```
cds_srr( 'simulation/ampTest/schematic/psf', 'tran-tran', 'out' )
```

3. If you want your script to write results to the Results tab of the Outputs pane, you must assign the value to the `axlResult` variable as follows:

   ```
   axlResult = resultStatement
   ```

   For example:

   ```
   axlResult = max( out.time )
   ```

4. Save your changes and load this `.m` file.

Important

Matlab prefers current directory over any other path. So, if the Matlab script file (.m file) exists in the current directory, it is always run from there. To run a matlab script saved at a location other than the current run directory, delete the script from the current directory.

Example scripts

The following script creates a MATLAB plot using the specified X and Y values.

```
figure
x = [1 2 3]
```


© 2023 Cadence Design Systems, Inc.

```
y = [2 0 6]
plot(x,y)
```

The following script accesses the value of the `out` output variable from the current transient results directory using `axlCurrentResultsPath`, plots `out` using MATLAB (Voltage on the X axis and time on the Y axis), and writes the measured value (for total simulation time) to the Results tab of the Outputs pane.

```
out = cds_srr( axlCurrentResultsPath,'tran-tran', 'out' )
fig = figure
plot( out.V,out.time )
axlResult = max( out.time )
```


© 2023 Cadence Design Systems, Inc.

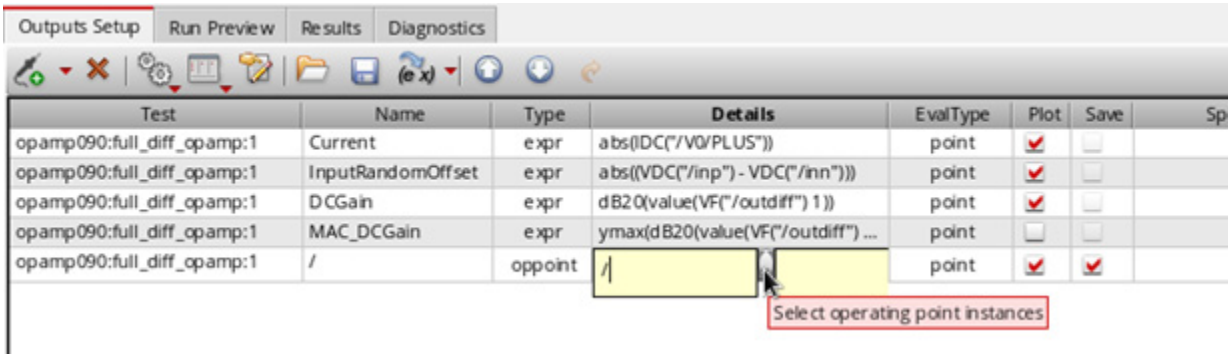
Saving Operating Point Parameters

In the Outputs Setup pane, you can specify the instances and their operating point parameters that you want the simulator to save. Currently, this feature is available only in the Spectre and operating point parameters for only DC and transient analyses are saved.

By default, the instance column on the Outputs setup pane is blank. To add instances and their operating point parameters that are to be saved, perform one of the following steps:

1. On the Outputs Setup tab, click .
2. In the drop-down list, select a test and choose *Op Parameters* from the sub menu.

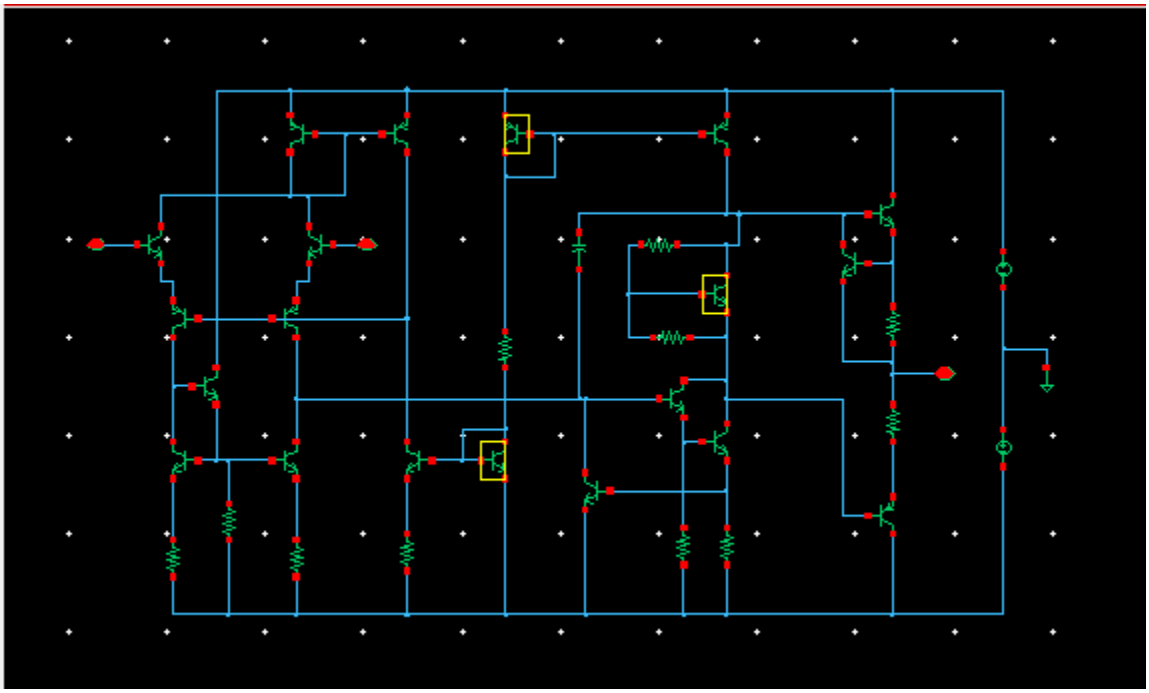
   A new row is added for the test with the output type *oppoint*.
3. In the new row, click *Select operating point instances*, as shown below.



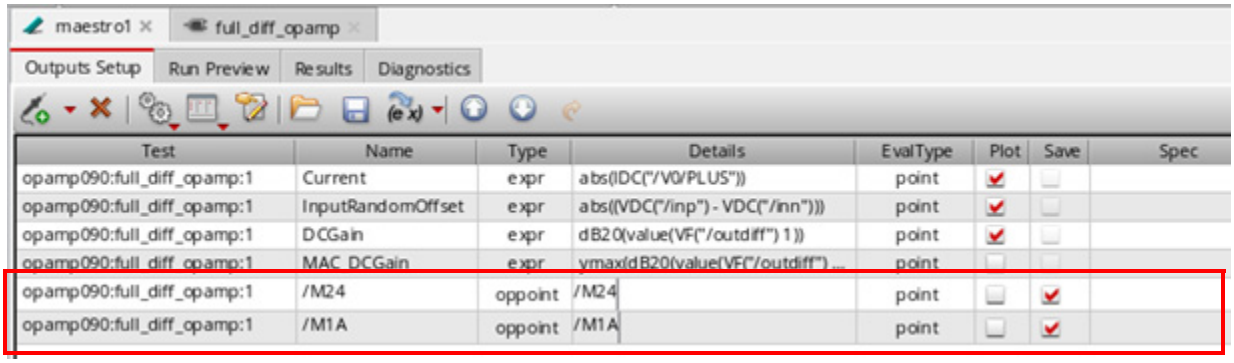
The Schematic Editor window appears.

© 2023 Cadence Design Systems, Inc.

4. Select the required instances in this window. The selected instances are highlighted in the Schematic Editor window, as shown in the figure below.



The selected instances are listed in the table as shown in the figure below.



Test	Name	Type	Details	EvalType	Plot	Save	Spec
opamp090:full_diff_opamp:1	Current	expr	abs(IDC("/V0/PLUS"))	point	☑	☐	
opamp090:full_diff_opamp:1	InputRandomOffset	expr	abs((VDC("/inp") - VDC("/inn")))	point	☑	☐	
opamp090:full_diff_opamp:1	DCGain	expr	dB20(value(VF("/outdiff") 1))	point	☑	☐	
opamp090:full_diff_opamp:1	MAC_DCGain	expr	ymax(dB20(value(VF("/outdiff") ...	point	☐	☐	
opamp090:full_diff_opamp:1	/M24	oppoint	/M24	point	☐	☑	
opamp090:full_diff_opamp:1	/M1A	oppoint	/M1A	point	☐	☑	

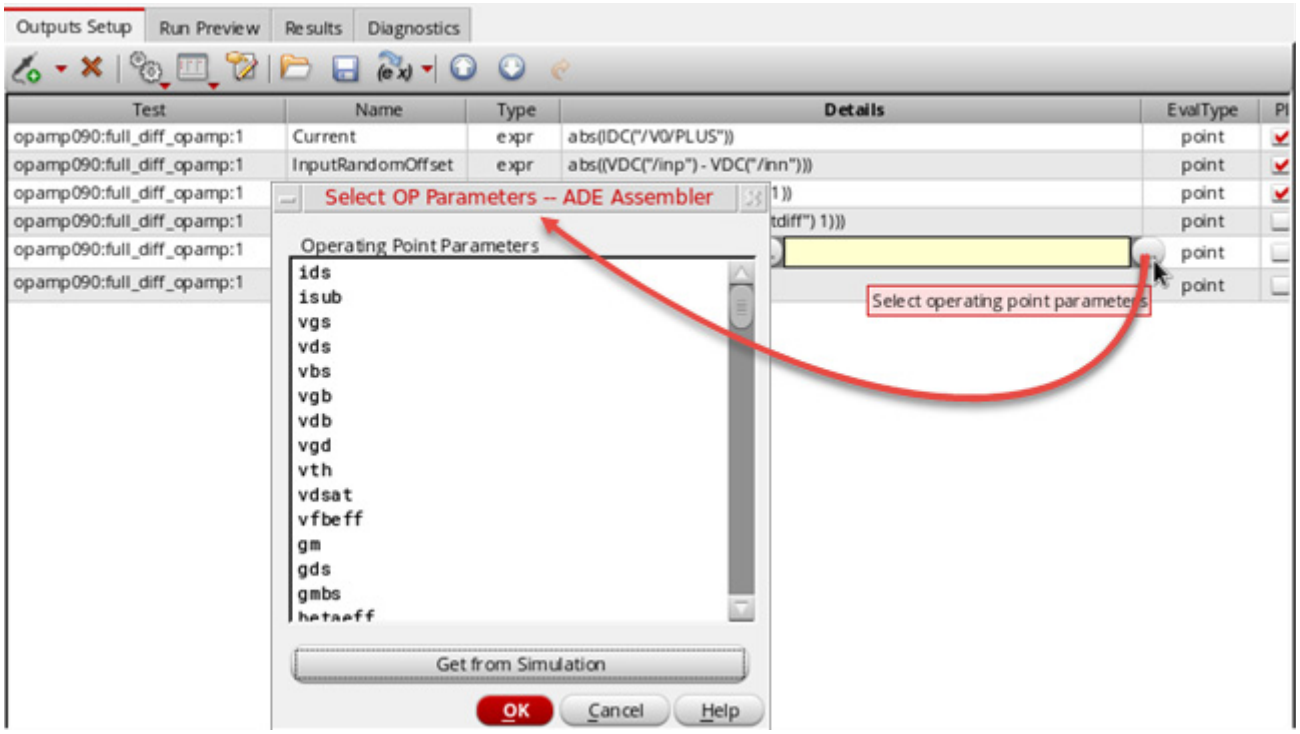
Adding Operating Point Parameters

In the *Details* column, do one of the following to specify the operating point parameters to be saved for each instance:

- Double-click in the *Details* column for each instance and type the name of the operating point parameters that are to be saved. The specified parameters must exist for that instance.

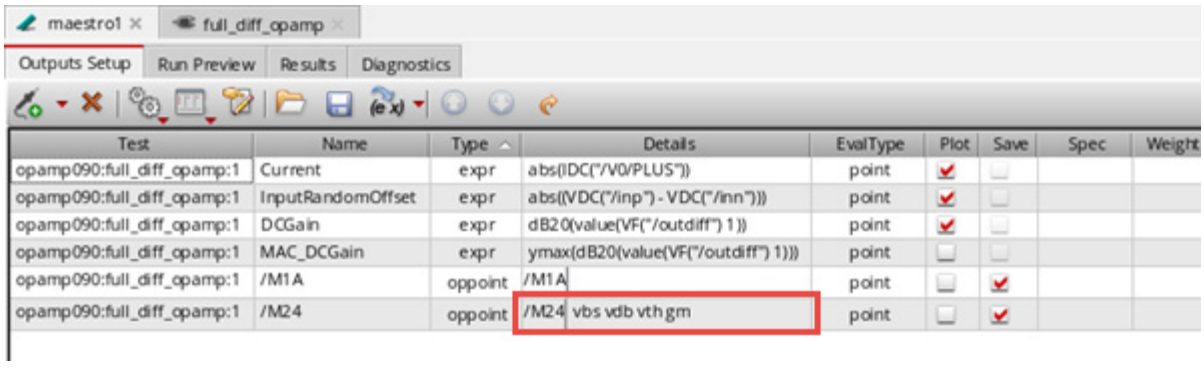
© 2023 Cadence Design Systems, Inc.

- Click the ellipses (...) button adjacent to the cell in *Details* column, as shown in the figure below.



The Select OP Parameters form appears. Choose the operating point parameters from the Select OP Parameters form. If the instance is not selected or invalid, the Save OP Parameters form will not open and a warning message is displayed in CIW.If you do not find the operating point parameters listed in the form, click the *Get from Simulation* button to run the simulation. If the simulation run is successful, the operating point parameters are listed in the form; otherwise, an error message is displayed in CIW.

Next, in the Select OP Parameters form, select the operating point parameters that you want to save by using the `Ctrl` key and click *OK*. The selected operating point parameters are added to the table.



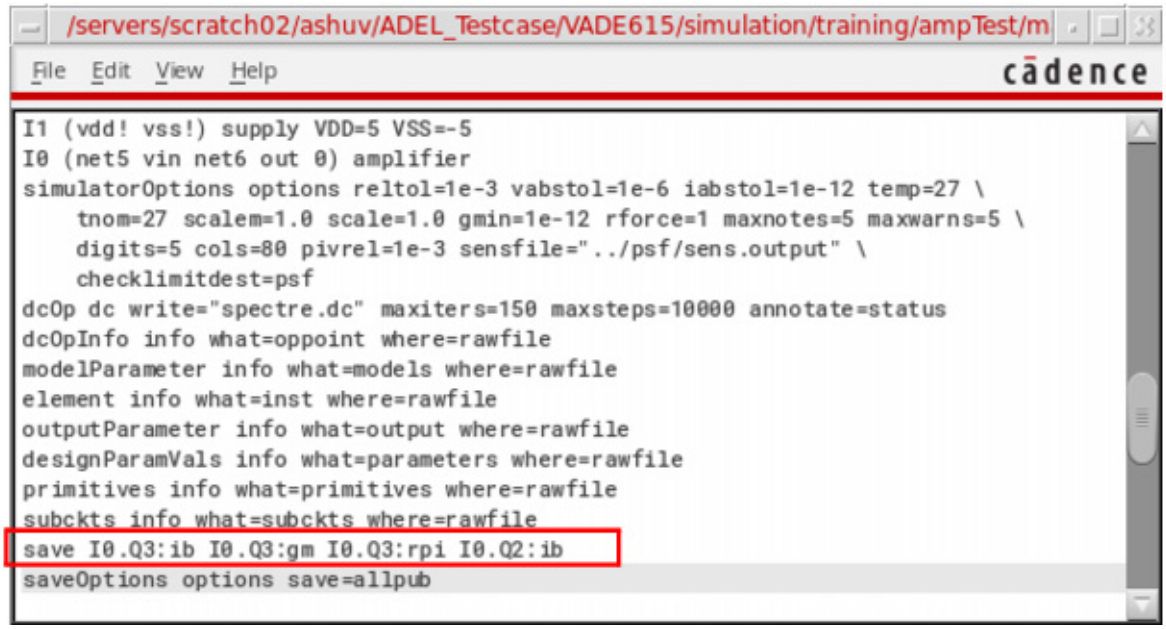
© 2023 Cadence Design Systems, Inc.

Note: Consider a scenario in which you have added two instances, `/I0/Q3` and `/10/Q2`, to the table from the schematic. You open the Select OP Parameters form for `/I0/Q3` and do not find anything listed in this form. In such a scenario, you need to click the *Get from Simulation* button to retrieve operating point parameters from the simulation run. If you now open the Select OP Parameters form for instance `/I0/Q2`, you will find that the operating point parameters for this instance are already listed in the form and you do not need to click the *Get from Simulation* button again.

If you next add some more instances from schematic to the table and open the Select Operating Points form for one of them, the operating point parameters for these instances will not be listed in the form and you will need to click the *Get from Simulation* button again to retrieve operating point parameters for the newly added instances.

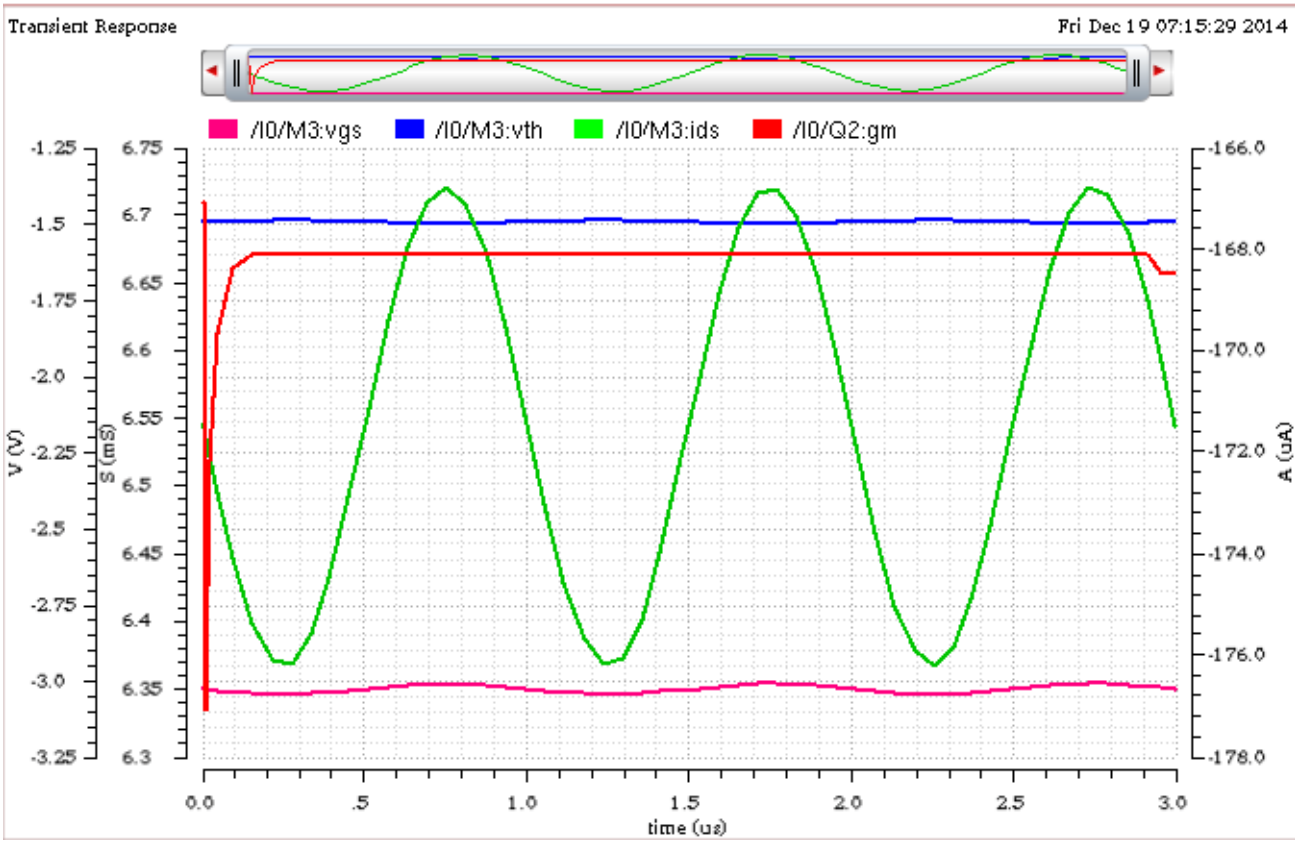
The operating point parameters that have already been added in the *Operating Parameters* field are always selected in the Select OP Parameters form if present.

Run the simulation. To ensure that the operating point parameters were saved, open the `input.scs` file (the link to this file is provided in the netlist). For example, in the figure below, you can see that the `ib`, `gm`, and `rpi` operating point parameters were saved for instance `M3`, and `ib` operating point parameter was saved for instance `Q2` in the `input.scs` file.



© 2023 Cadence Design Systems, Inc.

You can click the *Plot* button to plot the operating point parameters for instances `Q2` and `M3`. The selected parameters for these instances are plotted in append mode.



You can also add and delete operating points by using the saveOpPoint and deleteOpPoint OCEAN commands.

Viewing Tabular Report for Operating Point Parameters

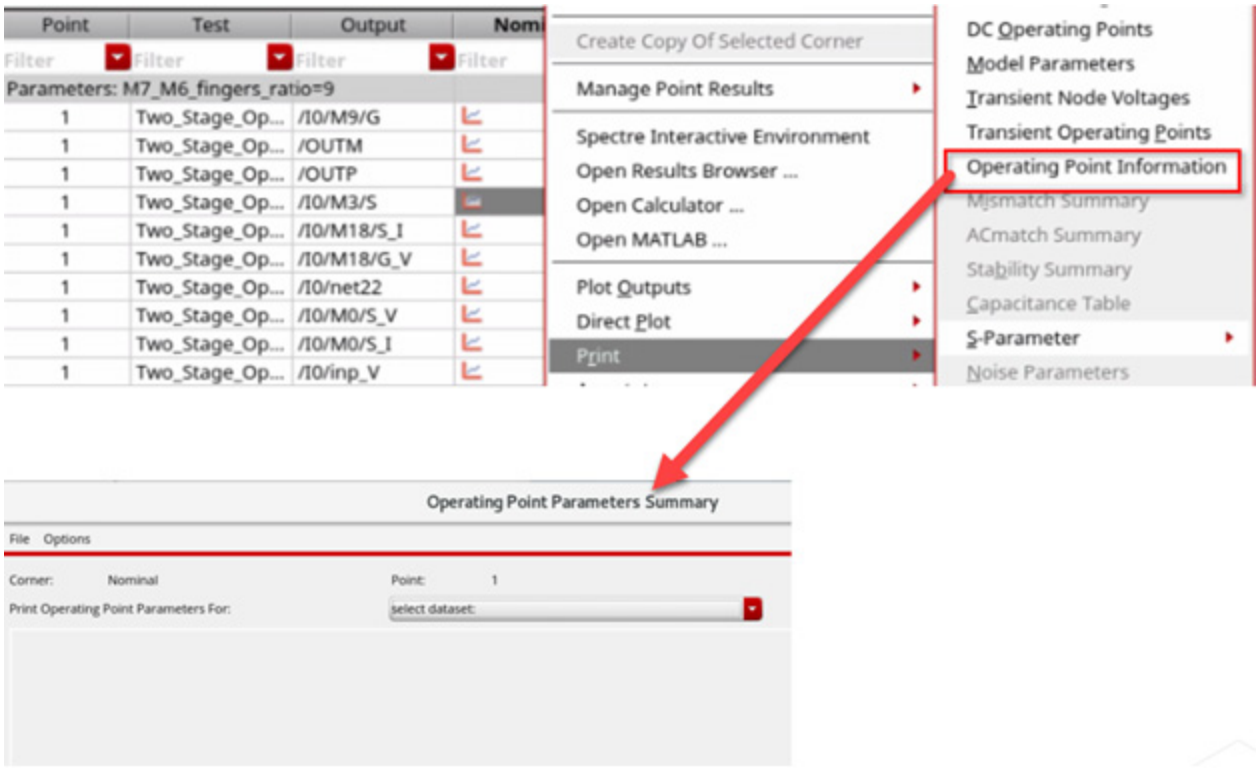
To use this feature, set the following environment variable in the CIW:

```
envSetVal("maestro.gui" "showOperatingPointsTableUI" 'boolean t)
```

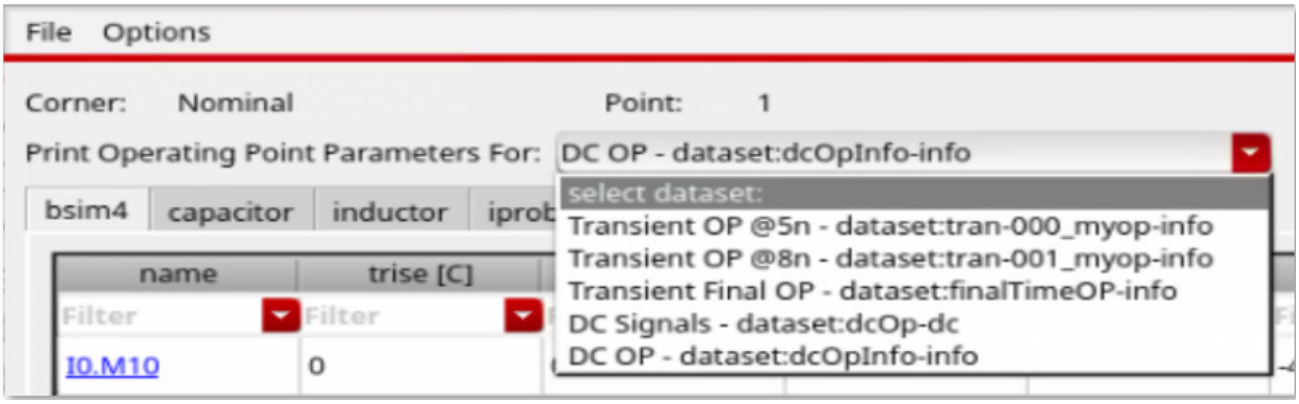
To open the tabular report for operating point parameters:

1. In the *Results* tab, right-click any results for a parametric sweep or corner result, choose *Print – Operating Point Information* to open the Operating Point Parameters Summary form.

© 2023 Cadence Design Systems, Inc.

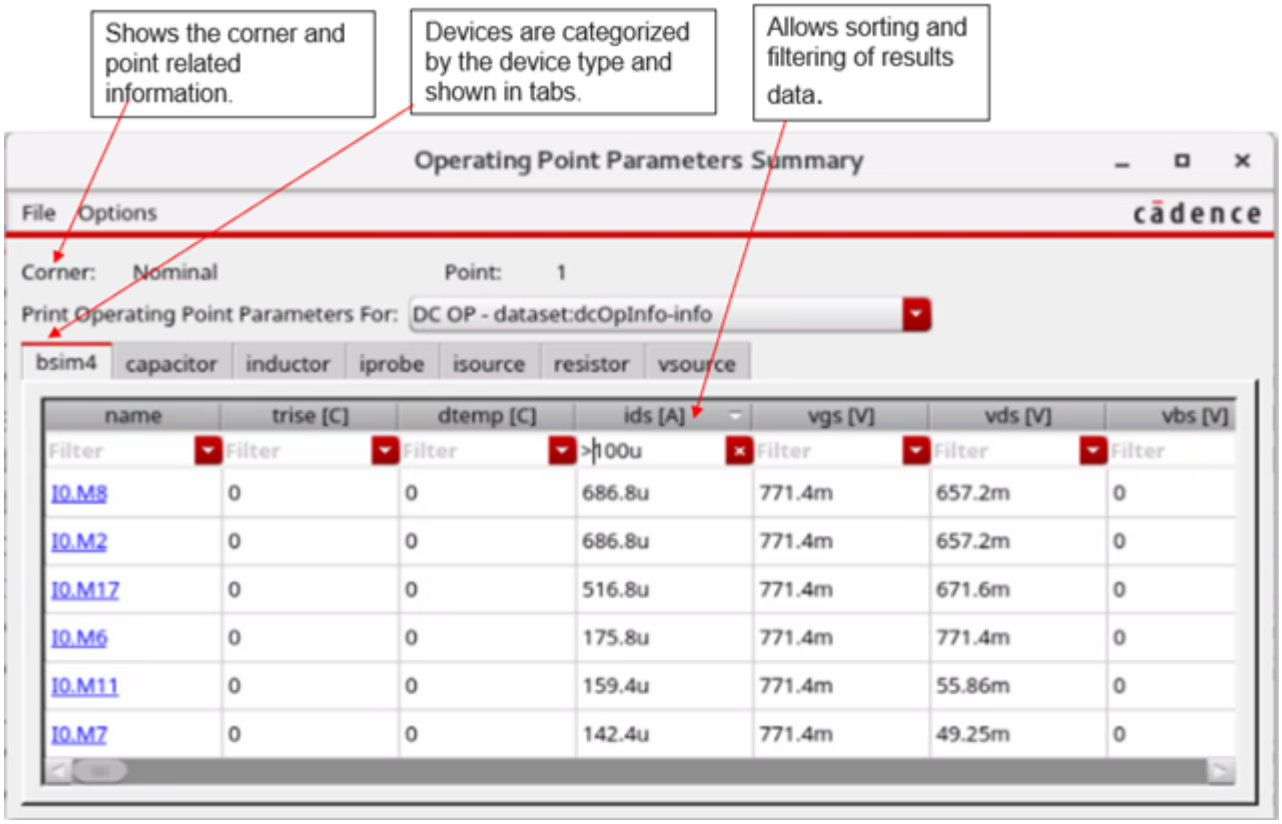


2. In the *select dataset:* drop-down list, select any one of the available transient and DC result datasets to view its results.



© 2023 Cadence Design Systems, Inc.

The following table shows the results for the operating point parameters and categorizes the devices based on their type. The corner name and point number are displayed in the labels on top of the form. You can sort and filter the results data to view selected data.



In the tabular report, you can do the following:

- To export the table content, click *File* and choose *Export to CSV*.
- To set the notation style and the decimal digits, click *Options* and choose *Set Display Options*.

© 2023 Cadence Design Systems, Inc.

- To cross-probe any device on the schematic, click the hyperlink for the device name. The corresponding device is highlighted on the schematic.

© 2023 Cadence Design Systems, Inc.

Copying Outputs

You can copy outputs within a test or from one test to another. You can also copy and paste the contents of an output. This allows you to quickly setup outputs for your tests.

For more information, see the following topics:

- Copying Outputs Within and Across Tests on page 442.
- Copying the Contents of an Output on page 443

Copying Outputs Within and Across Tests

You can copy outputs within a test or from one test to another. For example, if you have specified an output expression for one test, you can do the following:

- Make a copy of the output expression within the same test.
- Copy the output expression to another test.

Important

An operating region specification is copied from one test to another test only if the design and testbench of both the tests is same.

To copy outputs within and across tests, do the following:

➤ On the Outputs Setup tab of the Outputs pane, right-click the row for the output you want to copy, select *Copy To Test*, then do one of the following:

- ❑ To copy the output within a test, select the name of the same test.
- ❑ To copy the output to another test, select the name of that test.
- ❑ To copy the output to all tests, select *All*.

To copy multiple outputs, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection) and click the output to add it to the selection set. Right-click and choose *Copy To Test*, then select the name of the target test or *All*.

Note: An output will be copied across tests only if an output with the same name does not exist in the target test. For example, if you are copying an output expression named `DCGain` from one test to another, and the target test has an output with the same name, the output expression will not be copied.

© 2023 Cadence Design Systems, Inc.

Copying the Contents of an Output

To copy the contents (name, expression, file name, specification, weight, units or digits) of an output, do the following:

1. On the row for the output in the Outputs Setup tab of the Outputs pane, select the content you want to copy.

   For example, to copy an expression, select the expression in the *Details* column.

2. Press *Ctrl+C* to copy the content.
3. Select the cell where you want to paste the content.

   For example, to paste an expression, select a cell in the *Details* column.

4. Press *Ctrl+V* to paste the content.

© 2023 Cadence Design Systems, Inc.

Adding User-Defined Columns in the Outputs Setup Tab

In addition to the default set of columns that appear in the *Outputs Setup* tab, you can add one or more user-defined or custom columns. These user-defined columns can be used to save additional information, such as comments, target values for specifications, or other notes related to a measurement.

To help you use the information provided in the user-defined columns while analyzing results, ADE Assembler makes this information available in the following ways:

- User-defined columns are displayed on the *Results* tab. These columns are visible in the `Detail`, `Optimization`, `Summary`, and `Yield` result views.

  Note: Currently, these columns are not available in the `Detail-Transpose` view.

- If you add a new column and re-evaluate results, the new column is displayed on the *Results* tab.
- When you export the outputs to CSV or HTML files, the user-defined columns are also saved with other columns.
- These columns are displayed in the Specification summary sheets and datasheets.

For more information, see the following topics:

- Adding a User-Defined Column
- Renaming a User-Defined Column
- Deleting a User-Defined Column
- Hiding a User-Defined Column

Tip

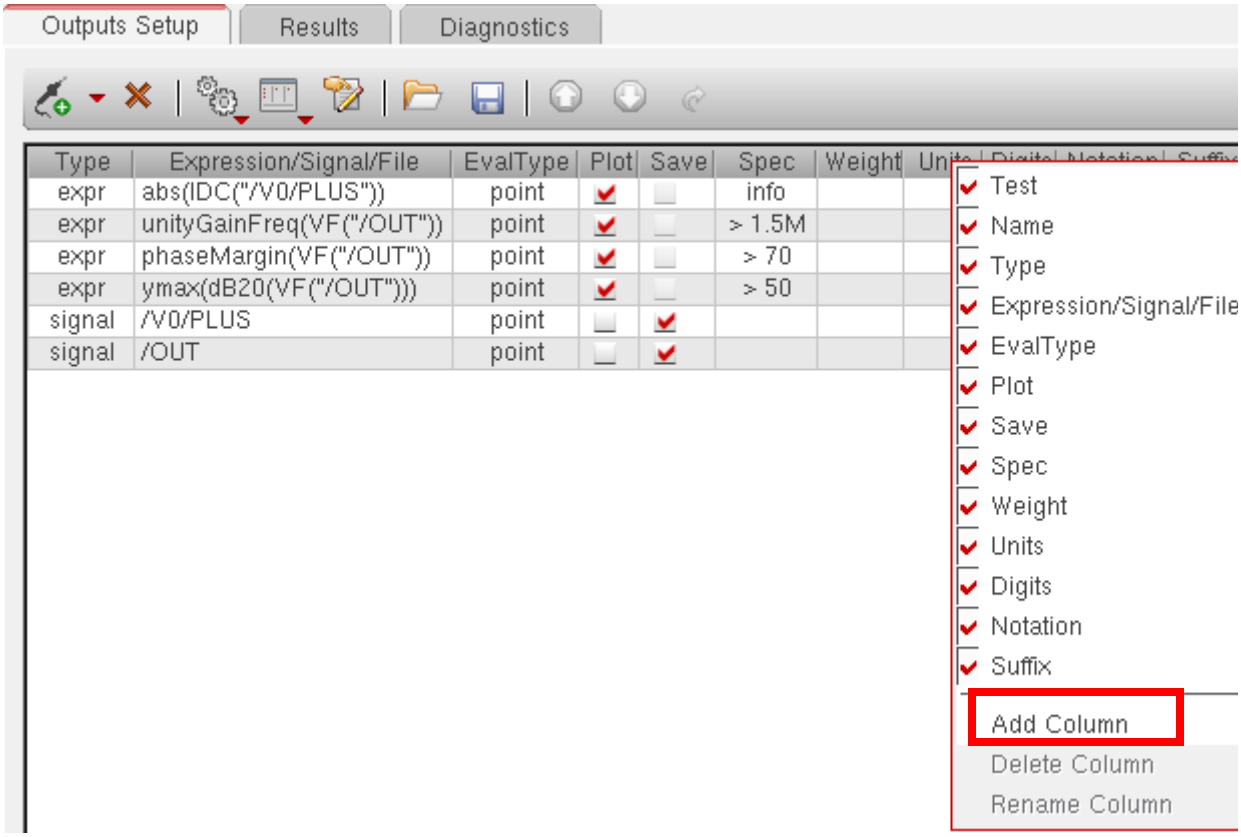
To know more about the SKILL APIs required to add a new user-defined column or to get or set value in a SKILL script, refer to *ADE SKILL Reference*.

Adding a User-Defined Column

To add a user-defined column in the *Outputs Setup* tab, do the following:

© 2023 Cadence Design Systems, Inc.

1. Right-click the title of any existing column and choose *Add Column* from the context-sensitive menu that appears.

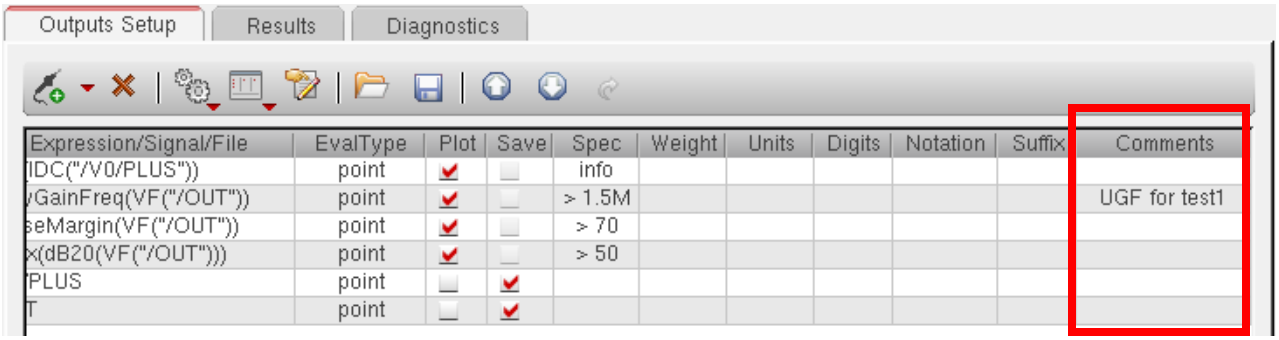


The Add Column form is displayed.

2. In the *New Column Name* field, specify a name for the new column to be added.
3. Click *OK*.

© 2023 Cadence Design Systems, Inc.

A new column is added to the right of the existing columns.



You can add any information in a custom column. The information in this column is retained and displayed on the *Results* tab post simulation run.

Also see:

- Renaming a User-Defined Column
- Deleting a User-Defined Column
- Hiding a User-Defined Column

Renaming a User-Defined Column

If required, you can rename a user-defined column. For this, do the following:

1. Right-click the column on the *Outputs Setup* tab and choose *Rename Column* from the context-sensitive menu.

   The Rename Column form is displayed.

2. In the *Rename Column to* field, specify a new name for the column.
3. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Deleting a User-Defined Column

To delete a user-defined column, do the following:

➡ Right-click the column and choose *Delete Column*.

The column is removed from the *Outputs Setup* tab.

Important

You can only delete a custom or user-defined column. It is not possible to delete a standard column from the *Outputs Setup* tab. However, you can hide or show selected columns, as required.

Hiding a User-Defined Column

To hide a user-defined column on the *Outputs Setup* tab, do the following:

➡ Click (*Configure what is shown in the table*) on this tab and clear *User-Defined Columns*.

To hide a user-defined column on the Results tab, do the following:

➡ Click (*Configure what is shown in the table*) on this tab and clear *User-Defined Columns*.

All the custom columns are hidden.

© 2023 Cadence Design Systems, Inc.

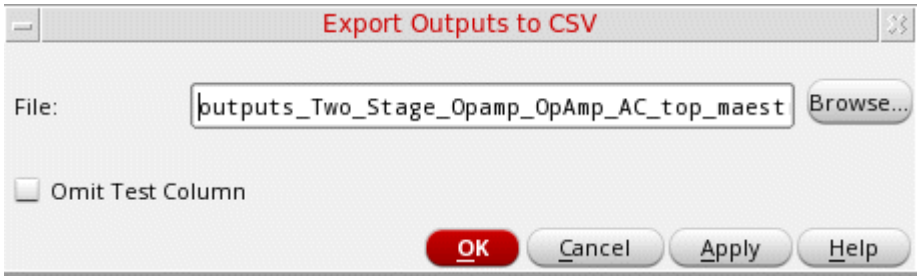
Exporting Outputs to a CSV File

You can save the outputs from the *Outputs* tab to a CSV file. This helps in saving outputs in a format from which they can be later reused and imported in another maestro view.

To export outputs to a CSV file, do the following:

1. Ensure that all the required outputs are defined on the *Outputs Setup* tab.
2. Click *Export Outputs to CSV File* on the toolbar on the *Outputs Setup* tab.

The **Export Outputs to CSV** form is displayed, as shown below.



3. (Optional) Specify a file name.

   The default file name is `outputs_<lib>_<cell>_<view>.csv` where lib, cell, and view are the library name, cell name, and view name associated with the test. By default, it is saved in the testcase directory.

4. (Optional) Select the *Omit Test Column* check box to specify that the test names are not to be saved in the CSV file.

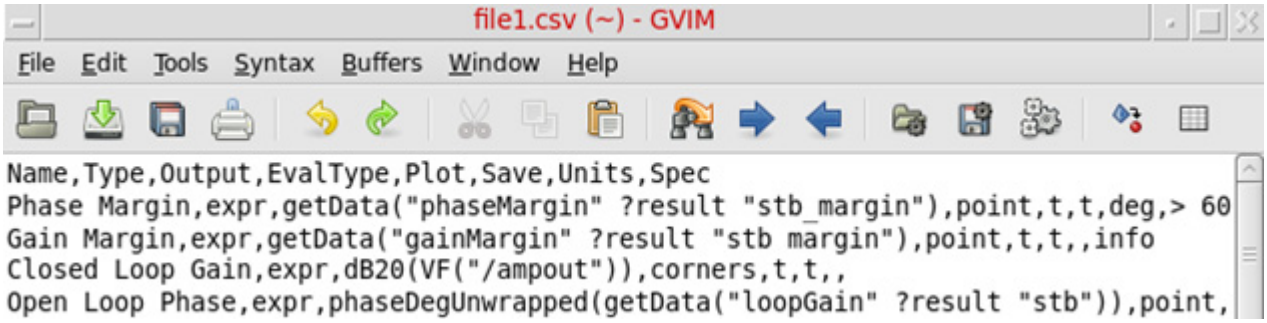
   By default, ADE Assembler saves the test name along with each saved output to indicate the test to which this belongs. Next time, when you import the outputs from the CVS file, the outputs will be assigned to the given test name only.

   When the outputs are saved without the test name, you can load them later for another test as well. For more details, refer to Importing Outputs from a CSV File.

5. Click *OK* to export the outputs and to close the form.

© 2023 Cadence Design Systems, Inc.

All the outputs defined on the *Outputs* tab are saved in the specified CSV file, as shown below.



Important Points to Note:

- In the saved CSV file, the *Test* column has been saved to indicate the test for which an output is saved. If you omit to save the test column, this information is not saved in the CSV file.
- An appropriate output type, such as, `net`, `terminal`, `expr`, or `corners` is used to indicate the output type. `corners` is the output type used for the expressions measured across corners.
- If no value is specified for any column, it is considered as `null`.
- It is mandatory to specify at least the *Name* column or the *Signal* and *Expression* columns in the CSV file.
- If you override an output specification for a corner, the overridden spec is also saved with the exported output. When you import such an output, the overridden spec value is applied to the same corner, if it exists in the ADE Assembler setup.
- Outputs that include the following are ignored while exporting outputs to CSV:
  - OCEAN measures
  - MATLAB scripts
  - area specifications
  - operating region specifications
- By default, only the *Test*, *Name*, *Type*, *Output*, *Eval Type*, *Plot*, *Save*, and *Spec* columns are saved. However, if you have specified values in other columns as well, all other columns, such as Notation, are also saved in the CSV file. In case of overridden specifications, multiple columns are saved for specifications.

© 2023 Cadence Design Systems, Inc.

Related topics:

- Importing Outputs from a CSV File on page 450
- axlOutputsExportToFile

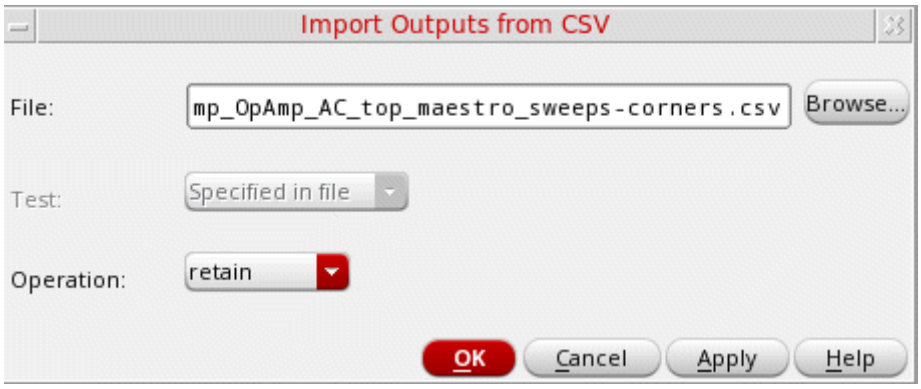
Importing Outputs from a CSV File

If you have outputs exported and saved in a CSV file, you can import those to the same or a different maestro view, as required.

To import the outputs from a CSV file, do the following:

1. Open the maestro view to which you need to import the outputs.
2. Click *Import Outputs from CSV File* on the toolbar on the *Outputs Setup* tab.

The Import Outputs from CSV form is displayed, as shown below.



The most recently exported file name appears in the File field.

3. Click *Browse* to locate and select a different file.

   ADE Assembler reads the specified file. If the details in the file were saved with the test column, the *Test* drop-down list on this form remains disabled.

   If the selected CSV file does not contain the details of the test column, the *Test* drop-down list is enabled and the names of all the tests in the current maestro view are listed in it.

© 2023 Cadence Design Systems, Inc.

Note: Outputs that were earlier exported from the ADE L environment in a `.txt` file can be imported in ADE Assembler.

4. (Optional) If the *Test* drop-down list is enabled, select the name of the test for which you wish to import the outputs from the specified CSV file.

   Select `Import to All Tests` to import the complete set of outputs for all the tests in the current maestro view.

5. Select an appropriate option from the *Operation* field to specify how you want to use the output rows that already exist before you import details from the CSV file.

   Following are the three possible operations to choose from:

   - `retain`: If an output already exists in the Outputs Setup tab with a name which is same as that of an output being imported, the import is not done and the existing output is retained.

     In this case, a message similar to the one shown below is displayed in CIW:

     ```
     *Info*    Output "Supply_Current" exists in the current setup and the
     operation is "retain". Skipping import of row 2 from CSV file "./out2.csv".
     ```

     An existing output which has the same signal name as that of an output being imported is also retained.

   - `merge`: If an existing output for a given test has the same name as that of an output being imported, it is deleted and is overwritten by the imported output.

     In this case, a message similar to the one shown below is displayed in CIW:

     ```
     *Info*    Output "CMRR@10M" exists in the current setup and the operation
     is "merge". Deleting this output from the current setup and importing it
     from CSV file "./out1.csv".
     ```

     Note: Other outputs for which the names do not match with the outputs being imported are also retained.

   - `overwrite`: All existing outputs for the given test are deleted and all the imported outputs are used.

     In this case, a message similar to the one shown below is displayed in CIW:

     ```
     *Info*    The import outputs from CSV file operation is "overwrite". The
     original outputs in test "Offset" are deleted.
     ```

6. Click *OK* to import the outputs.

Related topics:

- Exporting Outputs to a CSV File on page 448
- axlOutputsImportFromFile

© 2023 Cadence Design Systems, Inc.

Configuring How Outputs Appear on the Outputs Setup Tab

By default, the Outputs Setup tab of the Outputs pane displays the outputs added for all the tests. You can change the order in which outputs or columns are displayed or you can show or hide the outputs or columns by using the toolbar on top of this tab.

For more information about viewing outputs in the Outputs Setup tab, see the following topics:

- Changing the Order of Outputs on page 452
- Sorting the Outputs on page 453
- Hiding and Showing Outputs on page 453
- Hiding and Showing Output Details on page 454
- Changing the Order of Columns on page 454
- Hiding and Showing Columns on page 455
- Filtering and Searching in Outputs Setup Pane on page 455

Changing the Order of Outputs

To change the order of outputs, do one of the following:

- Click the row for an output and:
  - Click the button to move it one level up.
  - Click the button to move it one level down.
- Drag and drop the output at the desired location.

When you run a simulation, the outputs for a test are displayed in the same order in the Results tab of the Outputs pane.

Note: Order of output rows is overridden if you sort the outputs by a column. For details, see Sorting the Outputs.

© 2023 Cadence Design Systems, Inc.

Sorting the Outputs

By default, outputs are sorted by test names. However, you can sort them by any column on this tab. For example, you can sort the outputs by output name or by output type.

To change the sorting order by a column:

➡ Click the header of that column.

The up or down arrow on the top-right area of the column header shows if the rows are being arranged in the ascending or the descending order. All the output rows are sorted according the values in the selected column.

Note: Before sorting the rows, if you had rearranged the output rows in a specific order by using the or buttons, those settings are overridden after you sort the outputs by a column.

To undo sorting or to restore the original arrangement of rows:

➡ Click *(Reset table to display per-test evaluation order)* button on the *Outputs Setup* tab.

The tool groups the outputs by test name and within each test, displays the outputs in the user-specified order.

Hiding and Showing Outputs

You can use the drop-down menu of the command on the Outputs Setup tab to specify the outputs to be displayed on this tab. The drop-down menu provides the following commands:

- *Show All Tests*: Shows the outputs for all the tests defined in the Data View pane.
- *Hide All Tests*: Hides all the outputs.
- *Show Enabled Tests*: Shows the outputs for all the tests enabled in the Data View pane.
- *Show Disabled Tests*: Shows the outputs for all the tests disabled in the Data View pane.
- `<test-name>`: Shows the outputs for the test named `<test-name>`.

© 2023 Cadence Design Systems, Inc.

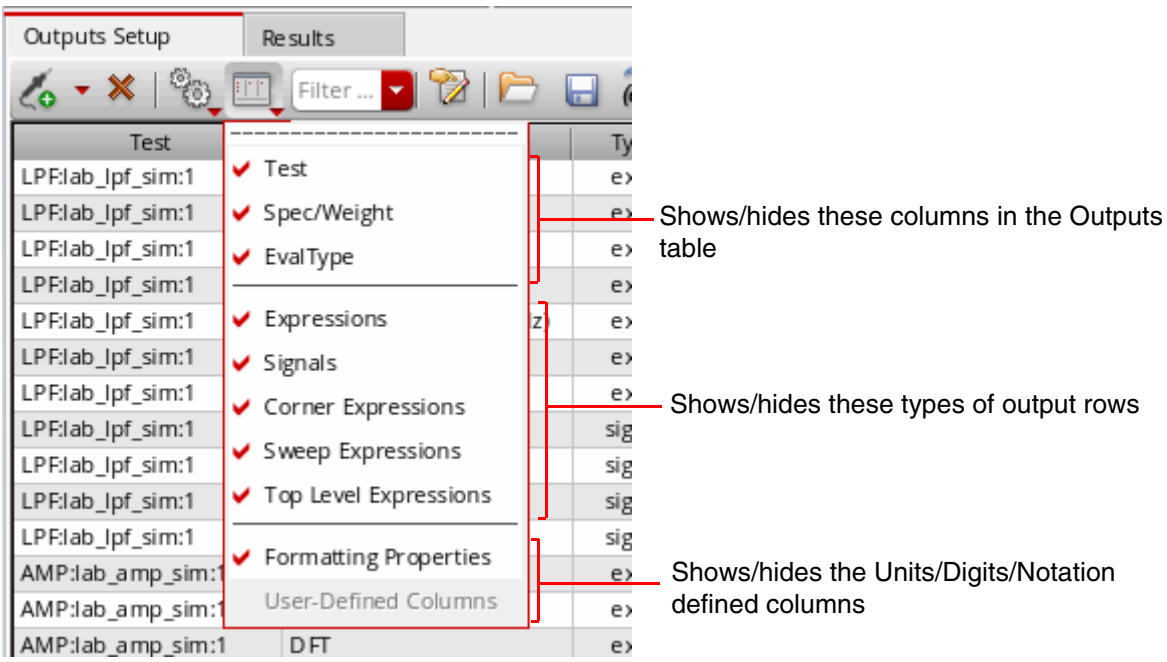
Important

By default, ADE Assembler displays outputs for all the tests. You can specify the default hide or show criteria for outputs by using the testsShownInOutputsSetup environment variable. After you change the default display criteria in an maestro view, the value specified for this variable is ignored for that view.

Hiding and Showing Output Details

To show or hide output details in the Outputs table, do the following:

1. Click the *Configure what is shown in the table* ( ) button on the Outputs Setup tab. The following drop-down list appears.



2. Select/deselect the check box next to an item in this list to show or hide the related information in the Outputs table.

Changing the Order of Columns

To change the order of columns in the Outputs Setup tab of the Outputs pane, do the following:

1. Click a column name.

© 2023 Cadence Design Systems, Inc.

2. Drag and drop the column at the desired location.

Hiding and Showing Columns

To hide the columns in the Outputs Setup tab of the Outputs pane, do the following:

Right-click a column name to display a context-sensitive menu listing all the columns. A check mark next to a column name indicates that the column is being displayed.

- To hide a column, click the column name.
- To unhide the column, right-click a column name and click the column once again in the context-sensitive menu.

Filtering and Searching in Outputs Setup Pane

Video

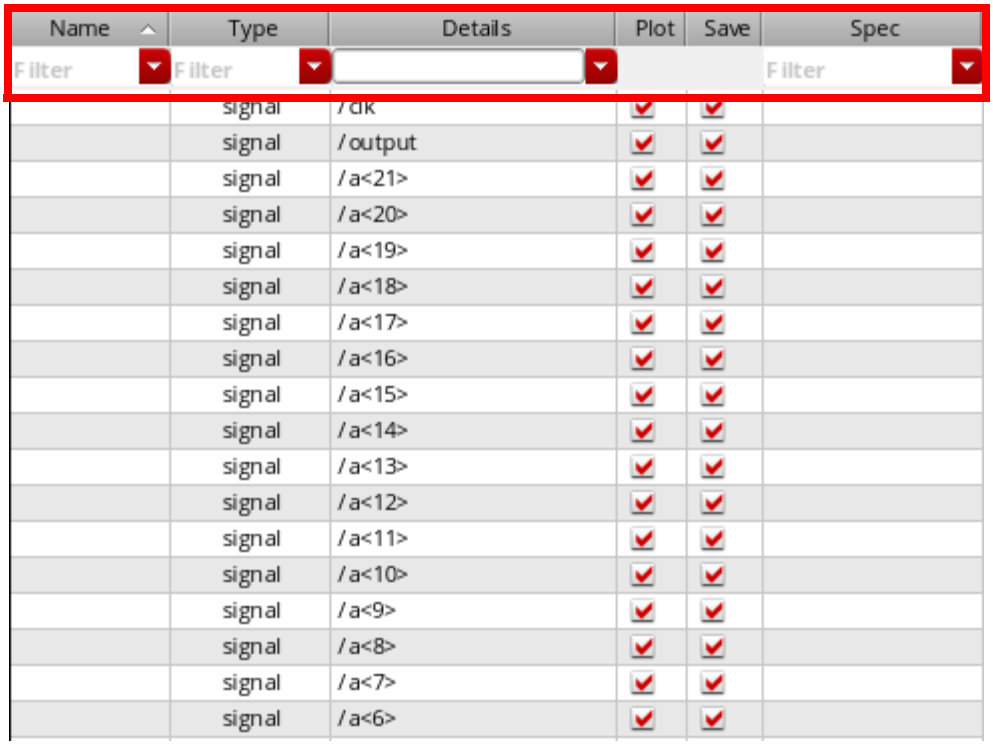
You can view a video demonstration for this feature at Filtering Data on the Outputs Setup and Results Panes.

You can use the search filters (*Filters*) displayed at the top of the Outputs Setup pane to filter the outputs based on specified search words or strings. These filters are displayed for all the fields in the Outputs Setup pane except for the *Plot* and *Save* check boxes.

Note: By default, the filters are always displayed in the ADE Assembler setup because the outputSetupFilteringNumOutputThreshold environment variable is set to `0`. However, you can hide these by setting this variable to the minimum number of rows for which you want to start showing the filters.

© 2023 Cadence Design Systems, Inc.

Regardless of the setting of this variable, you can also show or hide filters manually for any number of output rows. To do this, right-click anywhere on the outputs header and choose *Show Filter*s or *Hide Filters*.



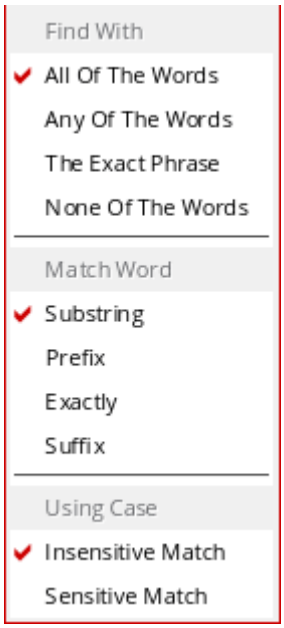
To filter the output rows in the Outputs Setup pane, do one of the following

- Type search words or strings in the *Filter* drop-down.
- Select the previously searched words or strings that are saved in the *Filter* drop-down. For more information, see Viewing Search History.
- Select the saved named filter from the *Named filters* toolbar. For more information, see Using Named Filters Toolbar.

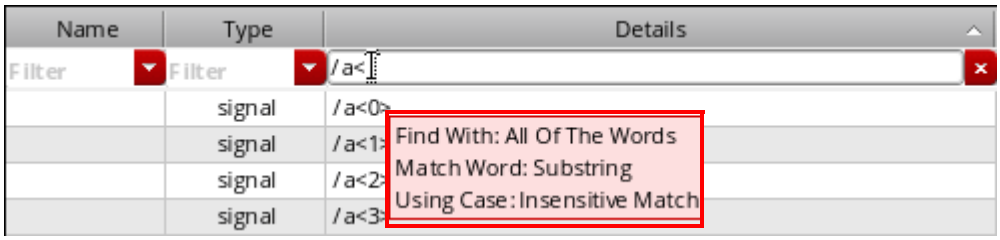
To specify filter conditions while searching or filtering the outputs,

© 2023 Cadence Design Systems, Inc.

➡ Right-click in the *Filter* drop-down list and select the required conditions. The following figure displays the conditions that are selected by default.



The tooltip on the *Filter* drop-down list displays the selected search conditions, as shown in the figure below.

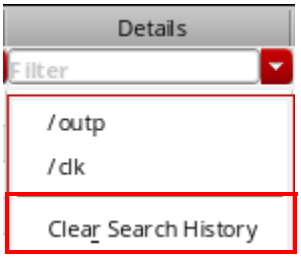


Viewing Search History

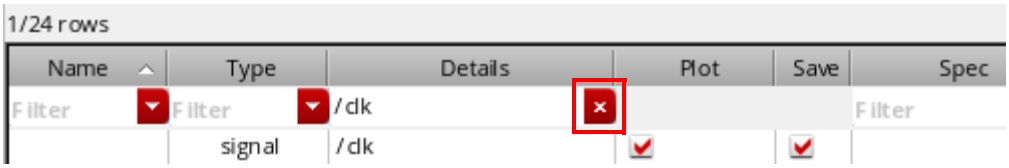
A history of the searched words or strings is also maintained in the *Filter* drop-down list. This history can be reused later, and therefore, helps in making the filtering or searching faster for the frequently used outputs. In addition, you do not need to type the search text again to search for particular outputs.

© 2023 Cadence Design Systems, Inc.

To clear the search history, select *Clear Search History* from the drop-down list of search filter.



In addition, after the searched outputs are filtered and listed, the filter drop-down list changes into a Cancel button that you can use to cancel the search and display all the specified outputs.



Using Named Filters Toolbar

You can use the *Named filters* toolbar to create and save named filters for combination of column searches that you use or refer to more frequently. These named filter include search words or strings used to filter the outputs for all columns.

To open the *Named filters* toolbar,

- Right-click anywhere on the menu bar and choose *Toolbars – Named filters*.
- Right-click anywhere on the toolbar area and choose *Named filters*.

To save a named filter,

➡ Type a name in the drop-down list and click the *Save named filter* button.

The specified name is added as a named filter and displayed in the drop-down list.

To delete a saved filter,

© 2023 Cadence Design Systems, Inc.

➡ In the drop-down list, select a named filter that you want to delete and click the *Delete named filter* button.

The selected named filter is deleted.

Selecting Outputs to Save or Plot

You can select outputs (signals and expressions) to save or plot by selecting the *Plot* or *Save* check boxes on the Outputs Setup tab of the Outputs pane.

To select an output for plotting, do the following:

➤ In the *Plot* column, select the *Plot* check box.

The program plots the item when the simulation finishes.

Tip

To select more than one output for plotting, select the outputs, right-click and choose *Enable Plot* from the context-sensitive menu.

To select an output for saving, do the following:

➤ In the *Save* column, select the *Save* check box.

The program saves the item to the results database.

Tip

To select more than one output for saving, select the outputs, right-click and choose *Enable Save* from the context-sensitive menu.

See also "Specifying Whether a Result Will Be Saved or Plotted" on page 366.

Removing Outputs

You can remove outputs (signals and expressions) from the save or plot set by deselecting the *Plot* or *Save* check boxes on the Outputs Setup tab of the Outputs pane. You can also remove outputs altogether from the test setup.

To remove an output from the plot set, do the following:

➤ In the *Plot* column, deselect the *Plot* check box.

© 2023 Cadence Design Systems, Inc.

The program does not plot the item when the simulation finishes.

Tip

To remove more than one output from the plot set, select the outputs, right-click and choose *Disable Plot* from the context-sensitive menu.

To remove an output from the save set, do the following:

➤ In the *Save* column, deselect the *Save* check box.

The program does not save the item to the results database.

Tip

To remove more than one output from the save set, select the outputs, right-click and choose *Disable Save* from the context-sensitive menu.

To remove an output from the test setup, do the following:

➤ In the Outputs Setup tab, select an output, right-click and choose *Delete Output*.

The program removes the item from the test setup.

Tip

To remove more than one output from the test setup, select the outputs, right-click and choose *Delete Output*.

See also "Specifying Whether a Result Will Be Saved or Plotted" on page 366.

© 2023 Cadence Design Systems, Inc.

Saving All Voltages or Currents

To save all of the node voltages and terminal currents, do the following:

1. On the Outputs Setup tab of the Outputs pane, right-click a test and choose *Save All*.

   The Save Options or Keep Options form appears, depending on your target simulator. The options that appear on this form vary according to your target simulator:

   - Save Options form for Spectre Simulations on page 462
   - Keep Options form for UltraSim Simulations on page 463
   - Save Options form for AMS Simulations on page 464
   - Save Options form for HspiceD Simulations on page 465

2. Select the values you want to save and click *OK*.

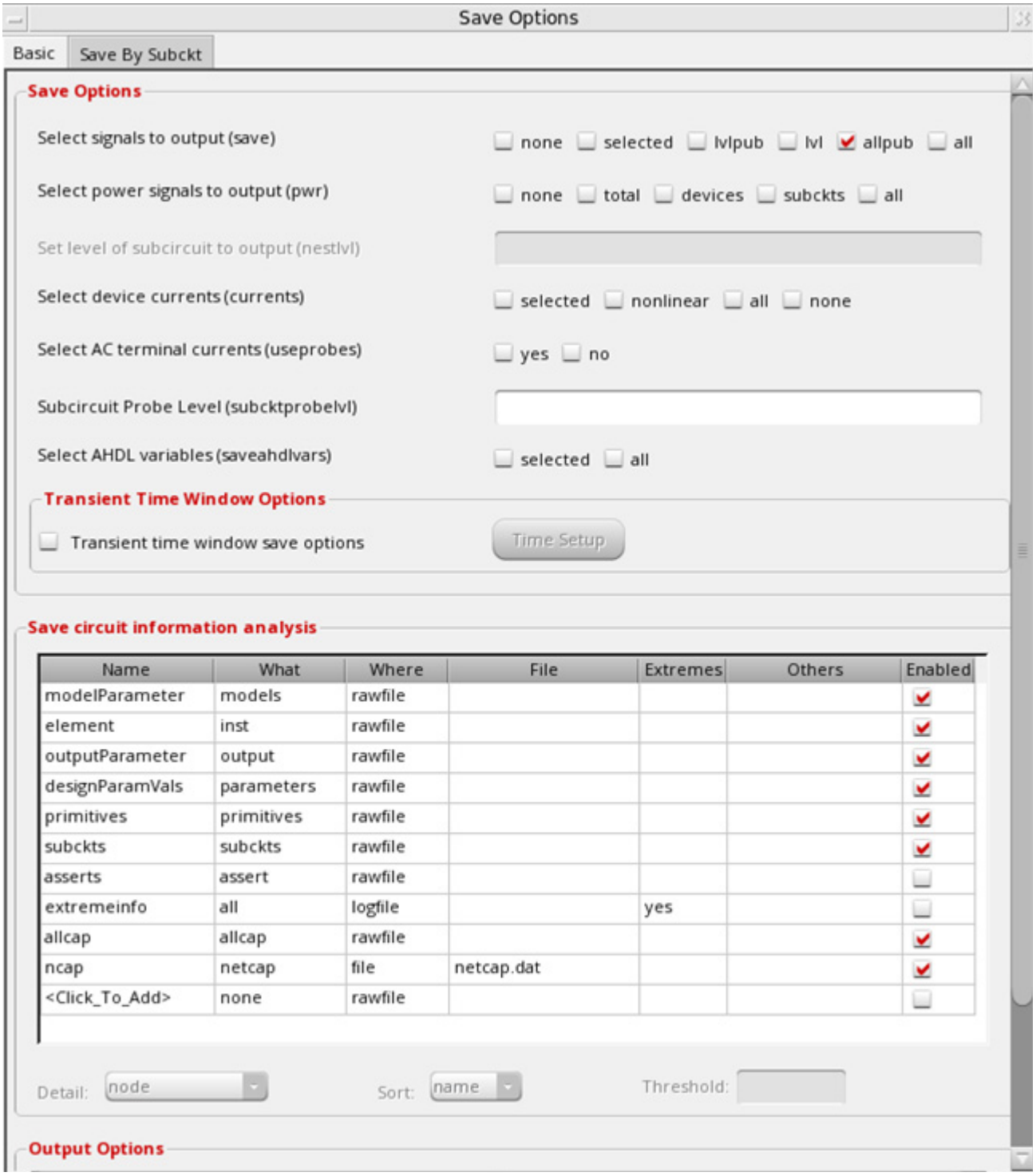
Important

Spectre never saves the current data for global pins even if you select Save all on the Outputs tab.

© 2023 Cadence Design Systems, Inc.

Save Options form for Spectre Simulations

For Spectre simulations, the following Save Options form appears:

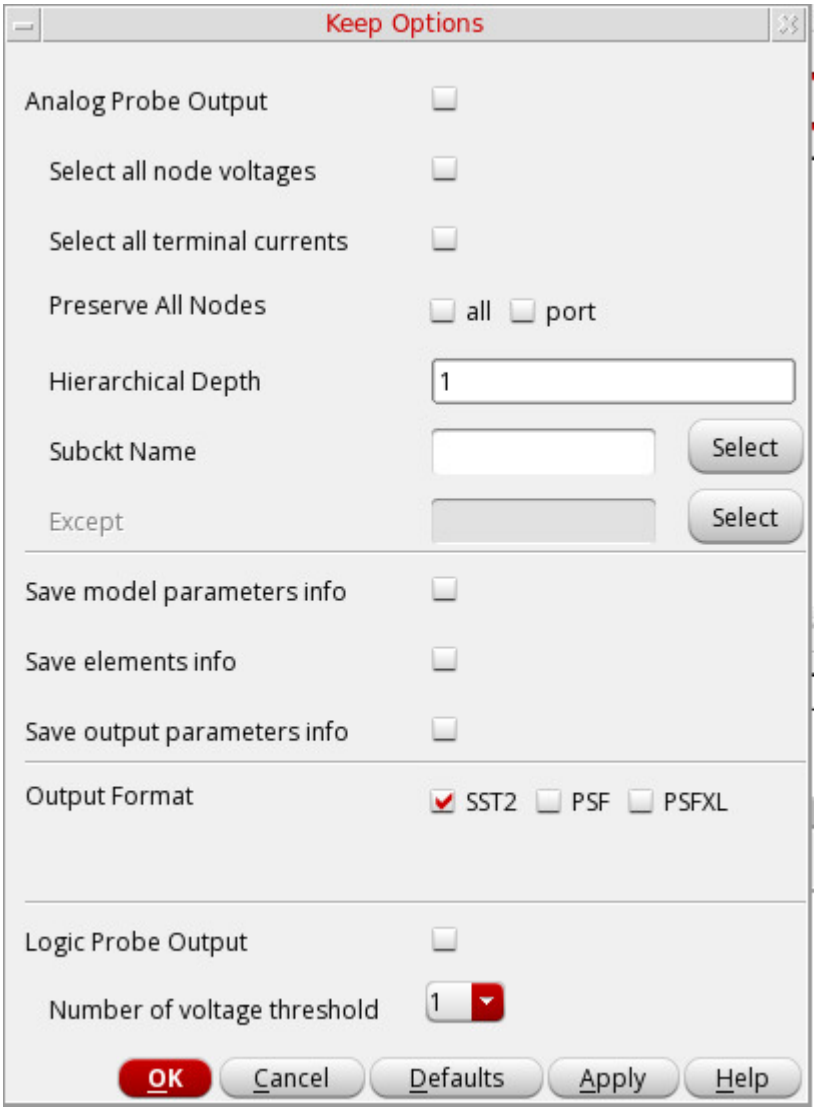


© 2023 Cadence Design Systems, Inc.

For more information about the fields on this form, see Saving All Voltages or Currents in the *Virtuoso ADE Explorer User Guide*.

Keep Options form for UltraSim Simulations

For UltraSim Simulations, the following form appears:

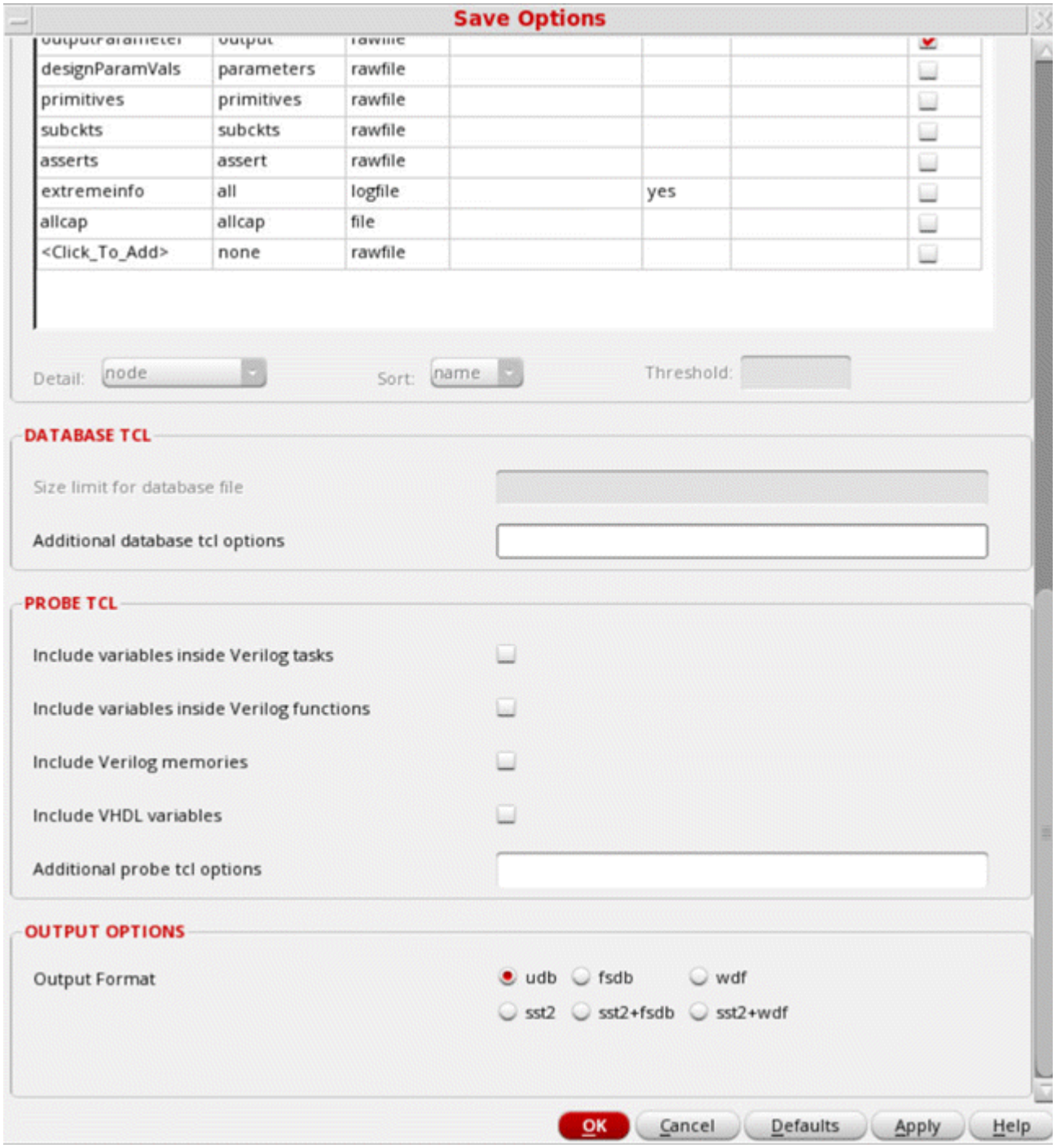


For more information on using the UltraSim simulator, refer to Working with UltraSim Simulator in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

Save Options form for AMS Simulations

For AMS Simulations, the following form is displayed.

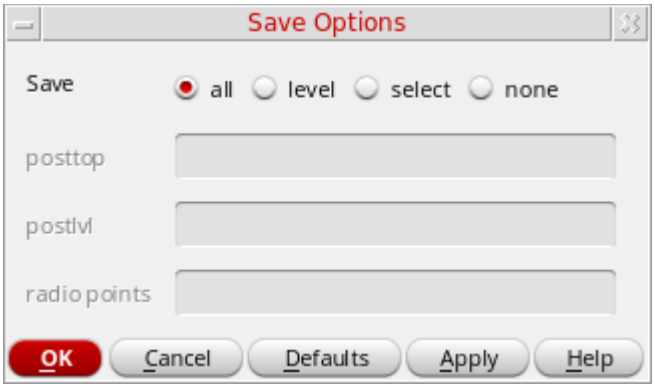


© 2023 Cadence Design Systems, Inc.

For more information about the fields on this form, see Saving All Voltages or Currents in the *Virtuoso ADE Explorer User Guide*.

Save Options form for HspiceD Simulations

For HspiceD Simulations, the following form appears:



For details on working with the HSPICE simulator, refer to Working with HSPICE Simulator in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

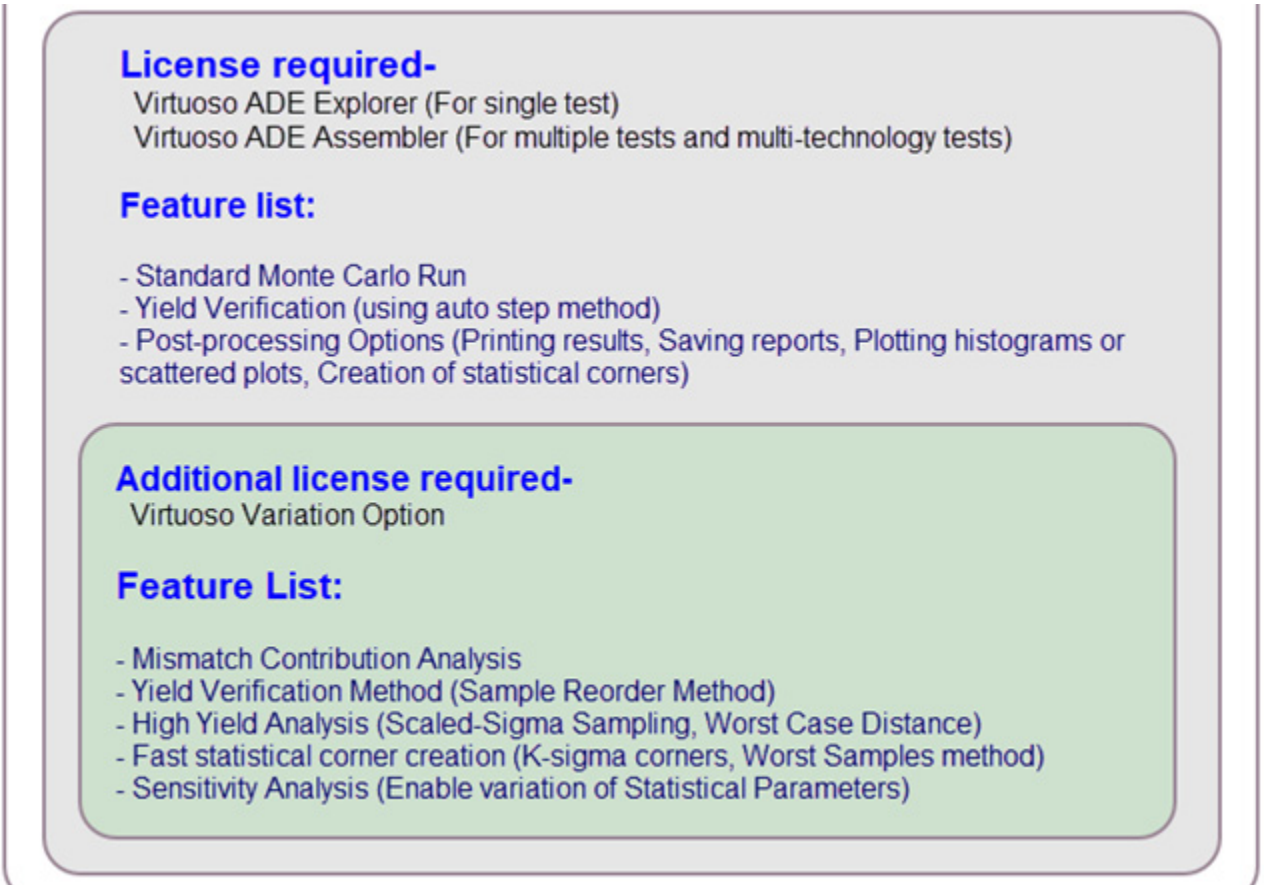
11

Monte Carlo Analysis

Monte Carlo analysis can be used to perform statistical analysis to estimate parametric yields and generate information about the performance characteristics of the circuits you design.

To perform statistical analysis for a single test, you can use ADE Explorer. For multiple tests or for multi-technology tests, you can use Monte Carlo in ADE Assembler.To used advanced run options or features, such as, high yield estimation, you require an additional license, `Virtuoso Variation Option`.

The following figure gives a snapshot of the licenses required to run various Monte Carlo methods and features.



© 2023 Cadence Design Systems, Inc.

The following topics in this chapter describe how to run Monte Carlo using ADE Assembler:

- Preparing Setup for Monte Carlo Analysis
- Running a Monte Carlo Simulation and Viewing Results
- Post-Processing Monte Carlo Results
- Advanced Monte Carlo Methods

© 2023 Cadence Design Systems, Inc.

Preparing Setup for Monte Carlo Analysis

The following sections provide details on how to set up Monte Carlo analysis:

- Preparing Setup in the Data View Assistant of ADE Assembler
- Setting Run Options in the Monte Carlo Form

Preparing Setup in the Data View Assistant of ADE Assembler

You can run Monte Carlo analysis over more than one test and corner. To selects the tests and corners that you want to use in analysis, do the following:

1. In the Data View assistant pane, identify the tests over which you want to run Monte Carlo and select the check boxes next to their names.

   The Run Summary pane reflects the number of tests you have selected.

2. Ensure that the simulator specified for the selected tests is Spectre, APS, or AMS Designer (with Spectre as the solver).

   You can run Monte Carlo analysis only with these simulators. For details on simulators that support Monte Carlo, refer to Before Running a Monte Carlo Analysis in *Virtuoso ADE Explorer User Guide*.

3. Disabled output expressions—expressions for which the *Plot* check box is not selected in the Outputs Setup tab—will not be evaluated for Monte Carlo simulations. So ensure that the *Plot* check box is selected for the output expressions that you want to be evaluated.

4. (Optional) Specify correlation constraints in Constraint Manager. The correlation constraints are evaluated during the Monte Carlo run. For more information, see Chapter 5, “Working with Constraints.”

5. (Optional) To use corners in the analysis, ensure that the *# Corners* check box on the Run Summary pane is selected.

   Note: You can modify the corners you have defined in the Corners Setup form.

6. (Optional) If you do not want to include the nominal corner in the Monte Carlo analysis, you can deselect the *Nominal Corner* check box on the Run Summary pane.

© 2023 Cadence Design Systems, Inc.

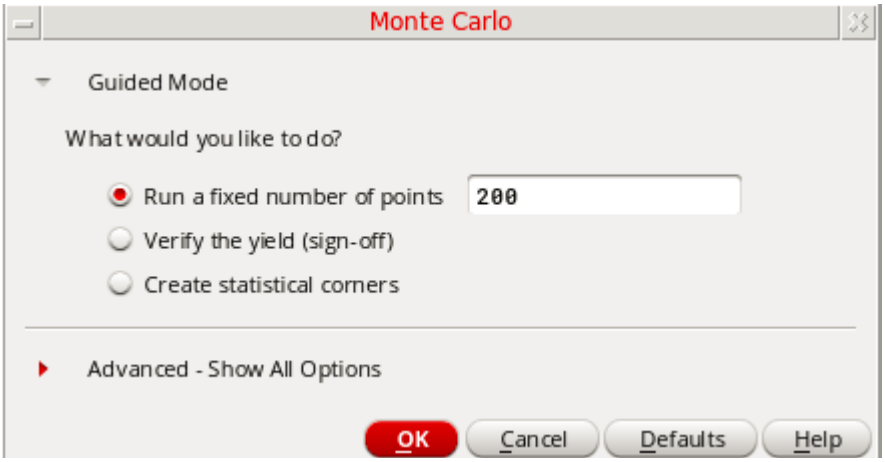
Setting Run Options in the Monte Carlo Form

Select the task that you want to perform and specify related options in the Monte Carlo run options form.

1. Choose Run – *Monte Carlo Sampling* to open the Monte Carlo form.

   Alternatively, select *Monte Carlo Sampling* in the *Select a Run Mode* drop-down list on the Run toolbar, then click (*Simulation Options*) on the Run toolbar.

   The Monte Carlo form appears, as shown below.



   This form includes two sections:

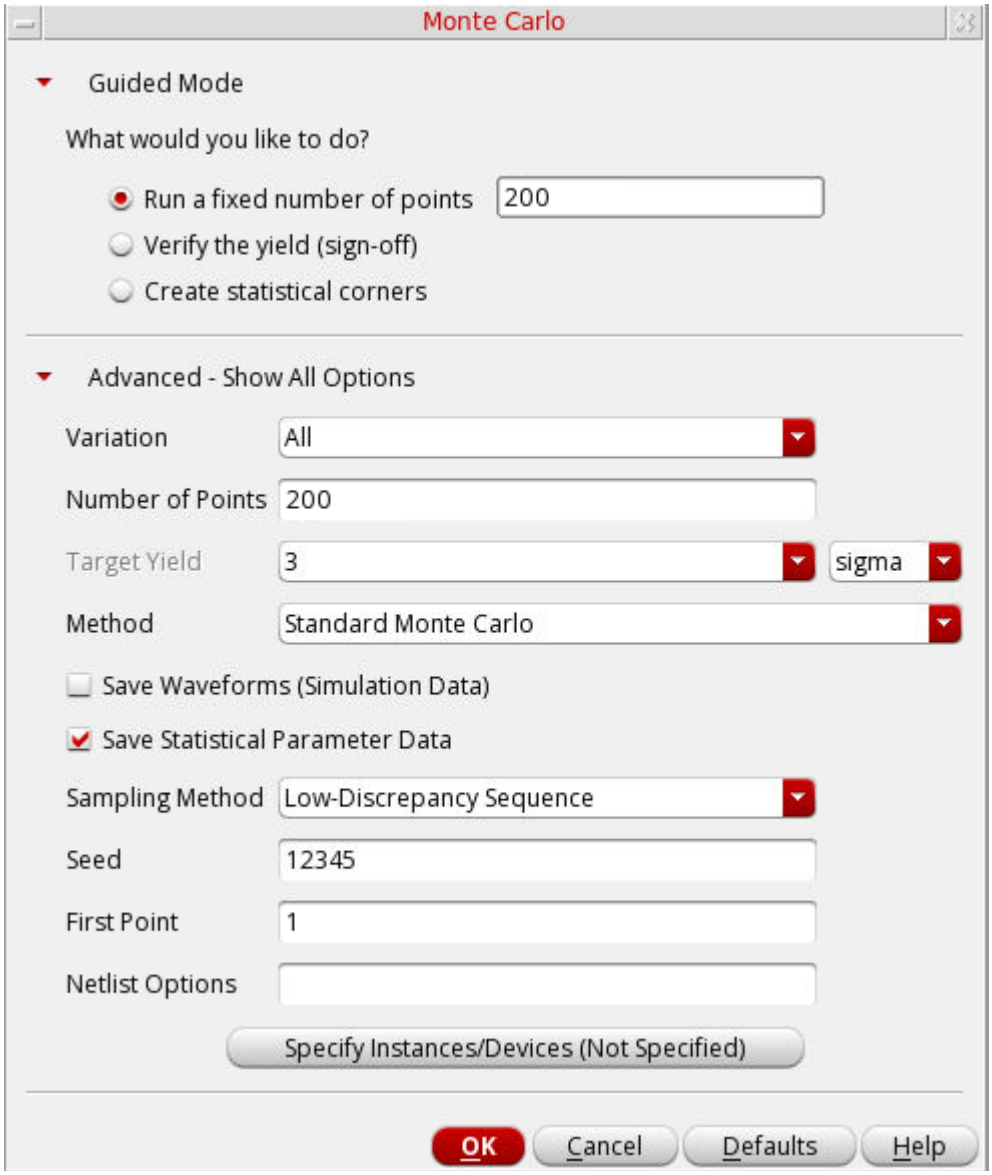
   – *Guided Mode*: Provides guidance in terms of the tasks you need to perform using Monte Carlo.

   – *Advanced – Show All Options*: Provides a relevant set of options to be set based on the task selected in the *Guided Mode* section.

2. Select an appropriate task that you want to perform.
3. Click the red arrow to the left of the to expand the *Advanced – Show All Options* section.

© 2023 Cadence Design Systems, Inc.

An example of the advanced options displayed for the default run mode, *Run a fixed number of points*, is shown below.



In this section, you can specify various options as listed below:

- The devices or device models for which you have specified statistically varying parameter values. You must specify either global (process) or mismatch (per-instance) variations or both.
- A method for statistical variation
- Number of points to run

© 2023 Cadence Design Systems, Inc.

- A sampling method to generate samples, seed value, and first point
- Other Monte Carlo options

  For details about the various options that can be set on this form for a standard Monte Carlo run, refer to Performing a Standard Monte Carlo Run section in the *Virtuoso ADE Explorer User Guid*e.

  By default, variation is applied to all subcircuit instances in the design. You can use the Specify Instances for Mismatch form to do one of the following:

  - Select specific instances to which variation must be applied. The variation is also applied to all subcircuits instantiated under the selected instances.
  - Select specific instances to which variations must not be applied. The variations is not applied to all subcircuits instantiated under the selected instances.

If multiple ADE Assembler tests are enabled, ensure that the same single top level DUT (Device Under Test) instance is selected for all the tests, so that you get the same statistical variation across all tests. This allows you to accurately predict the total yield across multiple tests.

To open the Specify Instances for Mismatch form, click *Specify Instances/Devices* on the Monte Carlo form.

Note: If the button name appears as *Specify Instances/Devices (Not Specified)*, it indicates that no instances or devices are selected. Therefore, mismatch variations are applied to all subcircuit instances in the design. If the button name appears as *Specify Instances/Devices (Exists)*, it indicates that you have selected specific instances and devices to include or exclude for applying mismatch variations.

© 2023 Cadence Design Systems, Inc.

The Specify Instances for Monte Carlo form appears as shown below.

For details on including or excluding instance and devices for applying mismatch variation, refer to Including or Excluding Instances and Devices for Applying Mismatch Variations in the *Virtuoso ADE Explorer User Guide*.

An additional point to note about this form in ADE Assembler is that if `All Tests` is selected in the *Tests* drop-down list, you can add only instances of cellviews that exist in the designs for all the tests. If instances of a cellview do not exist in the designs for all the tests, the *View* field displays the text `No Common View` indicating that you cannot select instances of the cellview.

Note: By default, the *Type* and *Sch/Master/Subcircuit* columns are not visible in the table that lists the selected instances. To view these columns, right-click the title bar and

© 2023 Cadence Design Systems, Inc.

select the name of the column to be displayed in this table.

Also see:

- Running Monte Carlo with Sweep Values
- Performing Yield Verification

© 2023 Cadence Design Systems, Inc.

Running a Monte Carlo Simulation and Viewing Results

To run a Monte Carlo simulation, click *Run Simulation* on the Run toolbar.

ADE Assembler calculates the number of simulations to be run. If the number exceeds the threshold limit specified using the warnWhenSimsExceed environment variable, a warning message is displayed to confirm if the run is to be continued.

Depending on the run options, ADE Assembler runs simulation for the identified samples. If the auto stop options are selected, the run is stopped automatically when the defined criteria is selected.

The following sections provide more details about running Monte Carlo simulations and viewing results:

- Running Incremental Monte Carlo Analyses
- Running Multi-Technology Simulations for Monte Carlo Analysis
- Viewing Results
- Viewing Statistical Parameters for Monte Carlo Samples
- Viewing Sensitivity Results
- Debugging in Debug Environment

Running Incremental Monte Carlo Analyses

You can use the Manual Tuning run mode in ADE Assembler to run Monte Carlo incrementally. In this run mode, you can append the results of multiple Monte Carlo runs to the same history and results view.

Important Points to Note

Consider the following points while running incremental Monte Carlo analyses in the Manual Tuning run mode:

- In order to run incrementally, all Monte Carlo runs must be performed within the Manual Tuning mode. You cannot append to a run performed outside of the Manual Tuning mode.
- If the chosen sampling method is `Random`, for every subsequent Monte Carlo run, specify an incremented value in the *Starting Run Number* field. For example, if you run the first Monte Carlo run for 20 points, in the subsequent run, set the *Starting Run Number* field on the Monte Carlo form to `21`. Similarly, the value in the *Number of Points*

© 2023 Cadence Design Systems, Inc.

field can vary across different Monte Carlo runs. For example, if you run the first Monte Carlo run for 20 points, for the second one, you can set the *Number of Points* field to 40.

- If the chosen sampling method is `Latin Hypercube`, specify a different seed value in the *Monte Carlo Seed* field or each subsequent Monte Carol run.

For more details on the Manual Tuning run mode, refer to Performing Manual Tuning.

Running Multi-Technology Simulations for Monte Carlo Analysis

You can run a Monte Carlo analysis in the multi-technology mode by enabling the Multi-Technology Mode option for a simulator and specifying the related Multi-Technology Simulation (MTS) options. For details on how to enable these settings, refer to Multi-Technology Designs and Locally Scoped Models and Options.

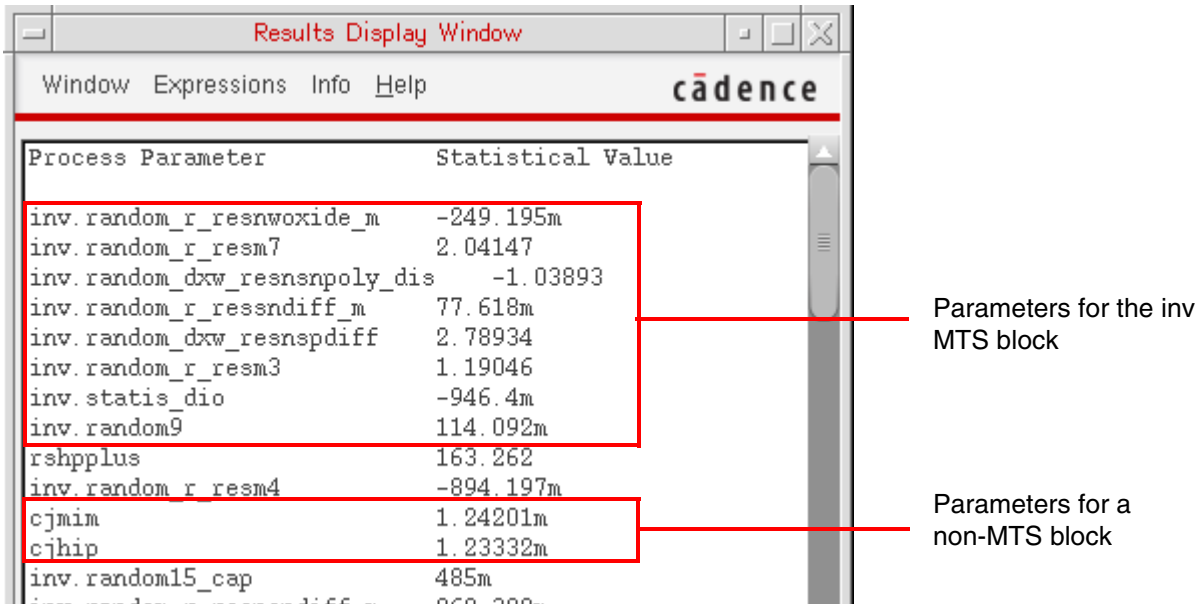
After identifying the MTS blocks and specifying MTS options for them, run the Monte Carlo analysis. If you have specified different model libraries for the MTS blocks, the statistical and process parameters used for those blocks are different than the parameters used for the non-MTS blocks.

Viewing Results of Monte Carlo Analysis with MTS

After a multi-technology simulation is run, you can view the results on the *Results* tab. To view the process and statistical parameters used for simulation of a data point, change the results view to the *Yield* view, right-click a result data cell and choose *Print Statistical Parameters*.

The Results Display Window shows a list of all the process and statistical parameters along with the statistical value used for simulation. The names of parameters for the MTS blocks are prefixed with the name of their corresponding block. For example, in the following figure, the names of parameters for the `inv` MTS block are prefixed with `inv`.

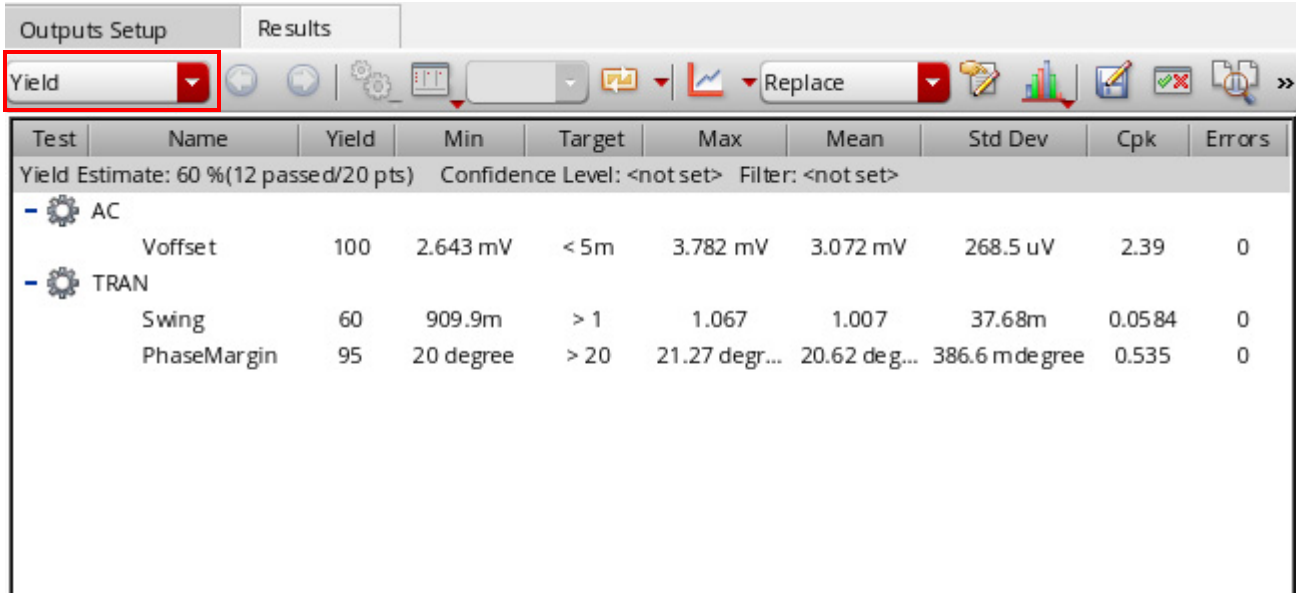
© 2023 Cadence Design Systems, Inc.



Similarly, in the Sensitivity Analysis results window, the parameter names for MTS blocks are prefixed with the name of their corresponding block. For more details, refer to Support for MTS in Sensitivity Analysis.

Viewing Results

After the run is complete, ADE Assembler displays the results in the *Yield* view on the Results tab of the Outputs pane, as shown below.

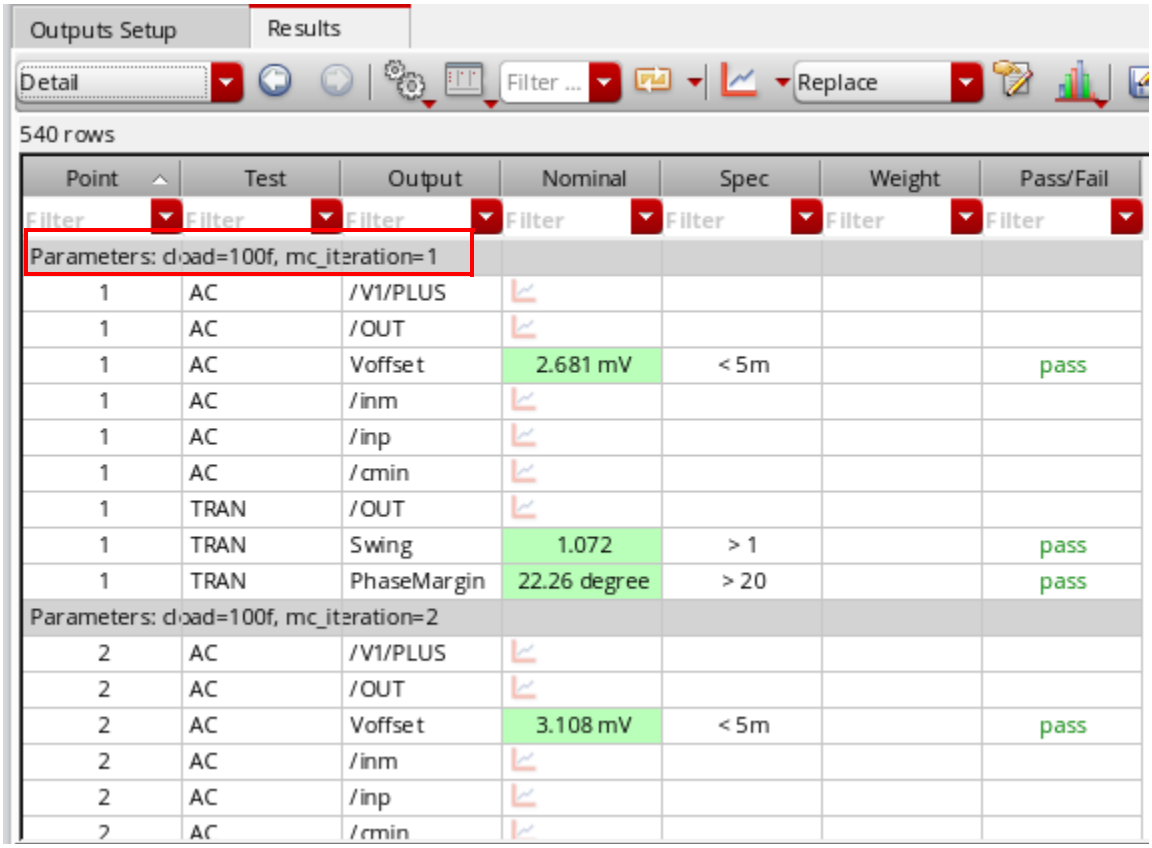


© 2023 Cadence Design Systems, Inc.

Results are reported for each test, corner and measurement expression and the total yield is reported for the circuit. For details about this view, refer to *Yield* view in Chapter 19, "Viewing, Printing, and Plotting Results.".

You can also use the Summary view or the Detail view on the Results tab to view the detailed results for the Monte Carlo analysis. In these views, the results for each Monte Carlo sample is displayed under a row named
`Parameters: parameter-name=parameter-value, mc_iteration=n`

where *n* is the number of the sample. For example, in the following figure of the Detail view, the results under the row `Parameters: cload=100f, mc_iteration=1` displays the results for the first iteration.



Point	Test	Output	Nominal	Spec	Weight	Pass/Fail
Filter	Filter	Filter	Filter	Filter	Filter	Filter
Parameters: cload=100f, mc_iteration=1						
1	AC	/V1/PLUS				
1	AC	/OUT				
1	AC	Voffset	2.681 mV	< 5m		pass
1	AC	/inm				
1	AC	/inp				
1	AC	/cmin				
1	TRAN	/OUT				
1	TRAN	Swing	1.072	> 1		pass
1	TRAN	PhaseMargin	22.26 degree	> 20		pass
Parameters: cload=100f, mc_iteration=2						
2	AC	/V1/PLUS				
2	AC	/OUT				
2	AC	Voffset	3.108 mV	< 5m		pass
2	AC	/inm				
2	AC	/inp				
2	AC	/cmin				

To view the description of the each column in the Detail view of the Results tab, refer to Hiding and Showing Data on the Results Tab. In some cases, the results are shown as `Canceled`. To know more about this status, refer to Starting a Simulation.

For more details on how to view and use the Monte Carlo results, refer to the following topics in *Virtuoso ADE Explorer User Guide*:

- Managing Monte Carlo Results in the Yield View

© 2023 Cadence Design Systems, Inc.

- Generating Plots, Tables, and Reports

Viewing Statistical Parameters for Monte Carlo Samples

After running a Monte Carlo analysis, you can switch to the *Detail* view on the Results tab of the Outputs pane to view the detailed results for each Monte Carlo sample. If the results for any sample need further investigation, you can view the statistical parameters for that sample so that you can debug your design.

To view and print the statistical parameters for a Monte Carlo sample, do the following:

1. In the *Select the results view* drop-down list on the Results tab of the Outputs pane, select *Detail*.

   The Detail view is displayed.

2. Right-click a result for a sample in the *Nominal* column and select *Print Statistical Parameters*.

   The statistical parameters for that sample are displayed in a Results Display Window.

For information about printing the contents in the Results Display Window, see Printing Results from the Results Display Window on page 1229.

Viewing Sensitivity Results

To view data on the sensitivity of variables and measurement expressions for a Monte Carlo run, do one the following:

- Click the *Sensitivity Results* button on the Results tab.
- On the History tab in the Data View pane, right-click the history item for a Sensitivity Analysis or Monte Carlo run and choose *Sensitivity Results*.

  The Sensitivity Analysis form appears displaying the sensitivity results.

  For more information about the Sensitivity Analysis form, refer to Sensitivity Analysis.

Debugging in Debug Environment

If any particular data point fails or shows undesired results, you can select the result in the Detail view and load it in the debug environment. Here, you can modify the setup of that data point to debug it and run a single Monte Carlo simulation to verify the results, and bring back the updated test to the active maestro view.

© 2023 Cadence Design Systems, Inc.

Varying Design Variables with Statistical Distribution

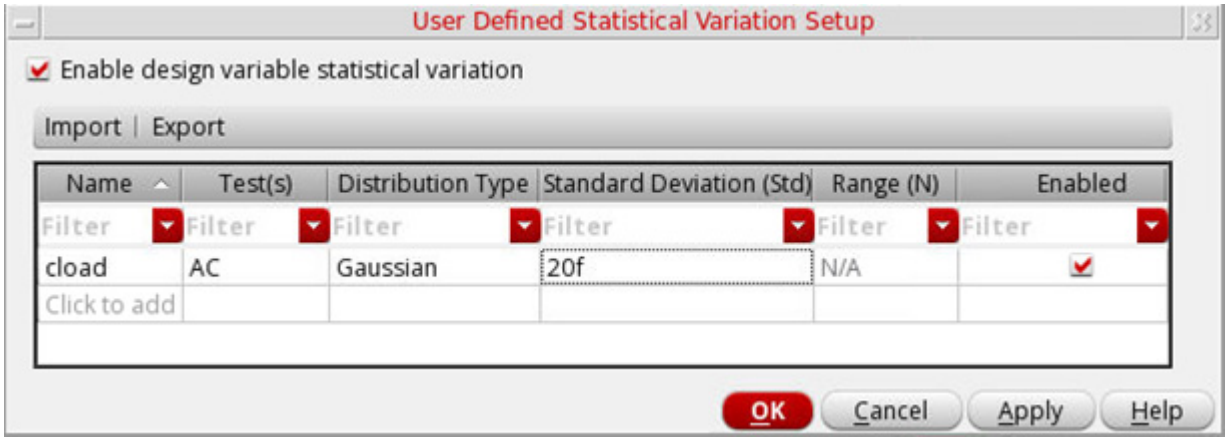
You can specify design variables that you want to vary with statistical distribution in Monte Carlo analysis.

To vary design variables with statistical distribution:

1. In the Data View assistant, right-click *Global Variables*, and then select *Define Statistical Variation*.

   Alternatively, click *Specify Design Variables* on the Monte Carlo form.

   The User-Defined Statistical Variation Setup form opens.



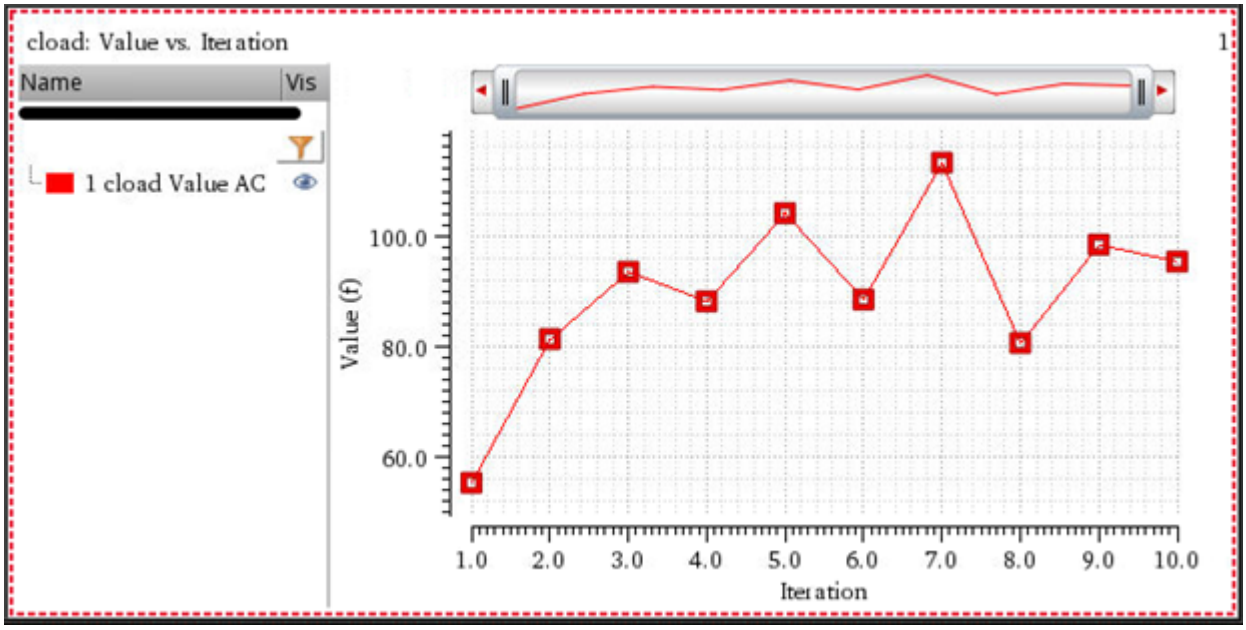
2. Click the *Enable design variable statistical variation* check box.
3. From the *Name* drop-down list, select a global design variable that you want to vary.

   Note: Ensure that the selected variable is not used in an enabled corner. If a variable is used in both corners and statistical variation, an error is displayed before running a simulation.
4. From the *Test(s)* drop-down list, select the tests for which you want to vary the design variables.
5. From the *Distribution Type* drop-down list, select one of the following distribution methods: *Gaussian*, *LogGaussian*, *Uniform*, or *LogUniform*.
6. In the *Standard Deviation (Std)* field, specify the standard deviation for the *Gaussian* or *LogGaussian* method. This field is not available for *Uniform* and *LogUniform* methods.

© 2023 Cadence Design Systems, Inc.

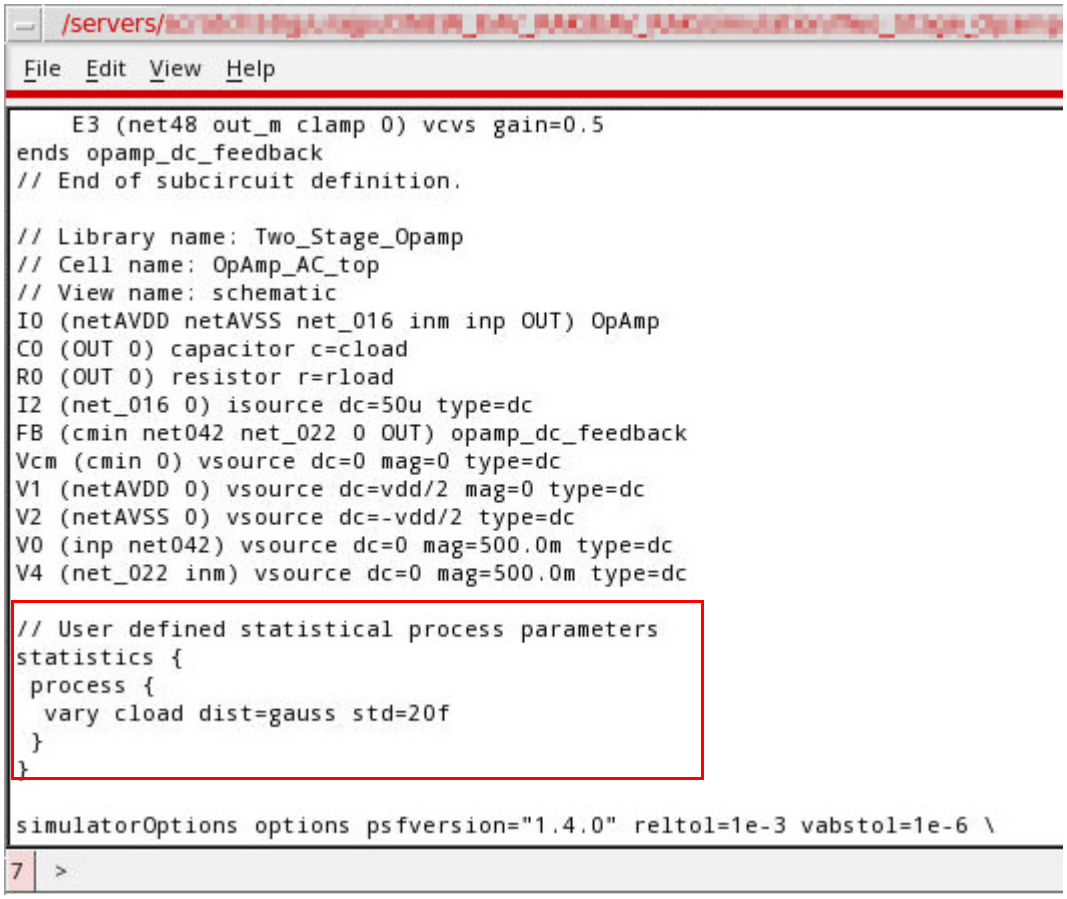
7. In the *Range (N)* field, specify a range for the *Uniform* or *LogUniform* method. Note that the parameter is varied in the range of the nominal value $\pm N$. This field is not available for *Gaussian* and *LogGaussian* methods.
8. Select the check mark in the *Enabled* column.
9. Repeat steps *3–8* to add another design variable.

   Note: After the setup is complete, you can export this setup to a CSV file by clicking the *Export* command. You can later import this CSV file to reuse this setup by clicking the *Import* command.
10. Click *OK* to close the User Defined Statistical Variation Setup form.
11. Run Monte Caro analysis and plot the values of user-defined statistical parameter to see how it varies according to the specified statistical distribution.



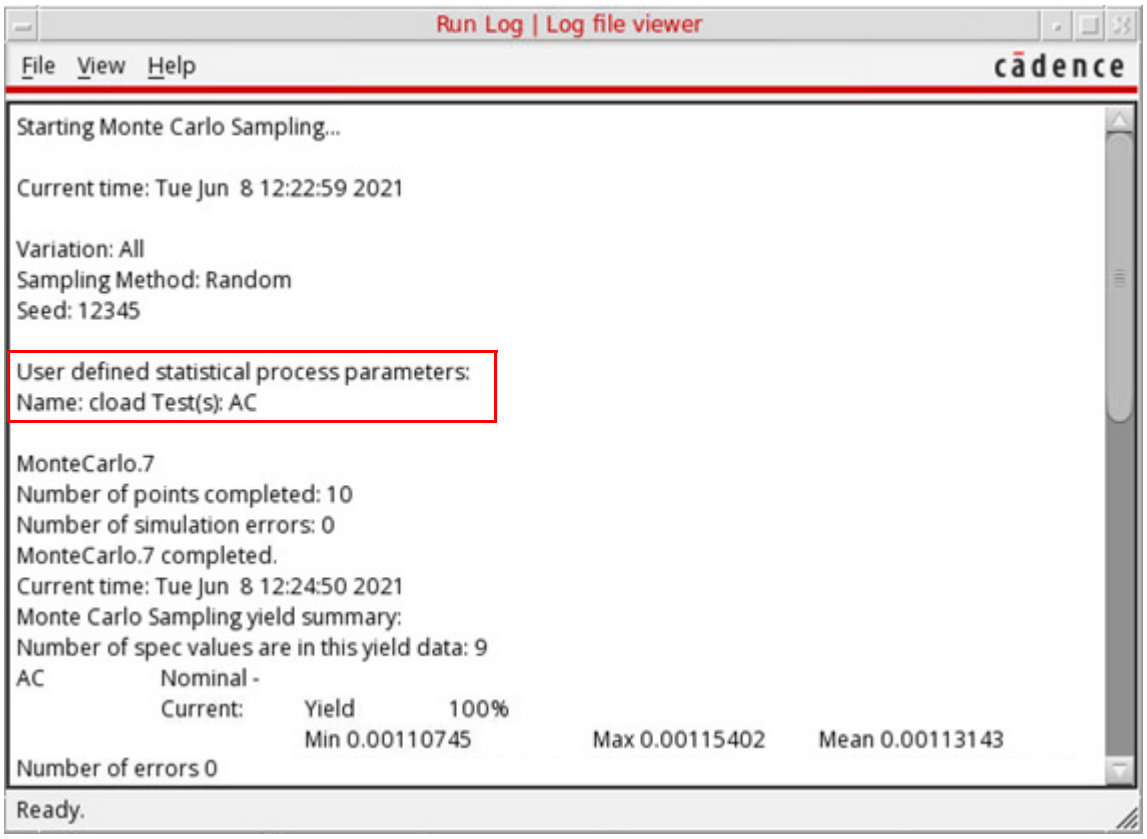
© 2023 Cadence Design Systems, Inc.

ADE Assembler generates the statistics block of user-defined statistical process parameters to include it in the netlist, as shown in the following figure.



© 2023 Cadence Design Systems, Inc.

The Run Log also reports information about the user-defined statistical parameters.



Post-Processing Monte Carlo Results

The default results view for the Monte Carlo Sampling run mode is the *Yield* view. This view displays the overall yield estimate based on the pass or fail status of all the specifications. For details on the different columns in this view, refer to the *Yield* view.

Note: You may change the default results view for the Monte Carlo Sampling run mode using the defaultResultsViewForMonteCarlo environment variable.

Also refer to the following topics for the post-processing commands for Monte Carlo results:

- Managing Monte Carlo Results in the Yield View
- Generating Plots, Tables, and Reports
- Saving Monte Carlo Results in CSV Format

© 2023 Cadence Design Systems, Inc.

Managing Monte Carlo Results in the Yield View

The following topics explain how you can manage the Monte Carlo results either to show the most relevant data or to create statistical corners that can be used in the next simulation runs:

- Viewing Data for a Specific Confidence Interval
- Creating Statistical Corners
- Filtering Out Data from the Yield View

Viewing Data for a Specific Confidence Interval

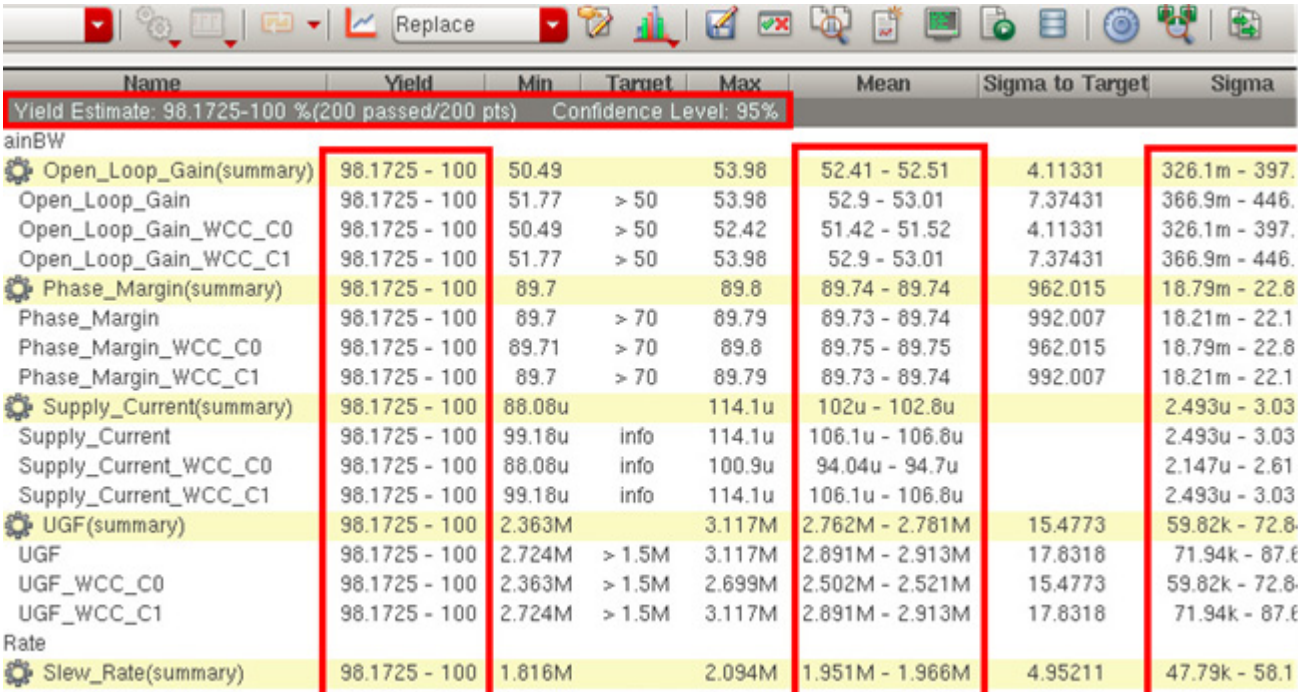
During or after a Monte Carlo analysis run, you can specify a confidence level to display the confidence intervals for the estimated yield, mean, and sigma.

To set the confidence level, right-click the gray-colored row at the top and choose *Set Confidence Level*. The Set Confidence Level form is displayed, as shown in the following figure.

Specify a percentage value in the *One-sided Confidence Level* field and click *OK*.

© 2023 Cadence Design Systems, Inc.

ADE Assembler displays the confidence intervals for the yield, mean, and sigma values, as shown in the following figure.



Name	Yield	Min	Target	Max	Mean	Sigma to Target	Sigma
Yield Estimate: 98.1725-100 %(200 passed/200 pts) Confidence Level: 95%							
ainBW							
Open_Loop_Gain(summary)	98.1725 - 100	50.49		53.98	52.41 - 52.51	4.11331	326.1m - 397.
Open_Loop_Gain	98.1725 - 100	51.77	> 50	53.98	52.9 - 53.01	7.37431	366.9m - 446.
Open_Loop_Gain_WCC_C0	98.1725 - 100	50.49	> 50	52.42	51.42 - 51.52	4.11331	326.1m - 397.
Open_Loop_Gain_WCC_C1	98.1725 - 100	51.77	> 50	53.98	52.9 - 53.01	7.37431	366.9m - 446.
Phase_Margin(summary)	98.1725 - 100	89.7		89.8	89.74 - 89.74	962.015	18.79m - 22.8
Phase_Margin	98.1725 - 100	89.7	> 70	89.79	89.73 - 89.74	992.007	18.21m - 22.1
Phase_Margin_WCC_C0	98.1725 - 100	89.71	> 70	89.8	89.75 - 89.75	962.015	18.79m - 22.8
Phase_Margin_WCC_C1	98.1725 - 100	89.7	> 70	89.79	89.73 - 89.74	992.007	18.21m - 22.1
Supply_Current(summary)	98.1725 - 100	88.08u		114.1u	102u - 102.8u		2.493u - 3.03
Supply_Current	98.1725 - 100	99.18u	info	114.1u	106.1u - 106.8u		2.493u - 3.03
Supply_Current_WCC_C0	98.1725 - 100	88.08u	info	100.9u	94.04u - 94.7u		2.147u - 2.61
Supply_Current_WCC_C1	98.1725 - 100	99.18u	info	114.1u	106.1u - 106.8u		2.493u - 3.03
UGF(summary)	98.1725 - 100	2.363M		3.117M	2.762M - 2.781M	15.4773	59.82k - 72.8
UGF	98.1725 - 100	2.724M	> 1.5M	3.117M	2.891M - 2.913M	17.8318	71.94k - 87.8
UGF_WCC_C0	98.1725 - 100	2.363M	> 1.5M	2.699M	2.502M - 2.521M	15.4773	59.82k - 72.8
UGF_WCC_C1	98.1725 - 100	2.724M	> 1.5M	3.117M	2.891M - 2.913M	17.8318	71.94k - 87.8
Rate							
Slew_Rate(summary)	98.1725 - 100	1.816M		2.094M	1.951M - 1.966M	4.95211	47.79k - 58.1

Creating Statistical Corners

After running Monte Carlo simulations, you can analyze the yield and identify the specifications for which the results need improvement. You can then create statistical corners to be used in further analysis and design optimization.

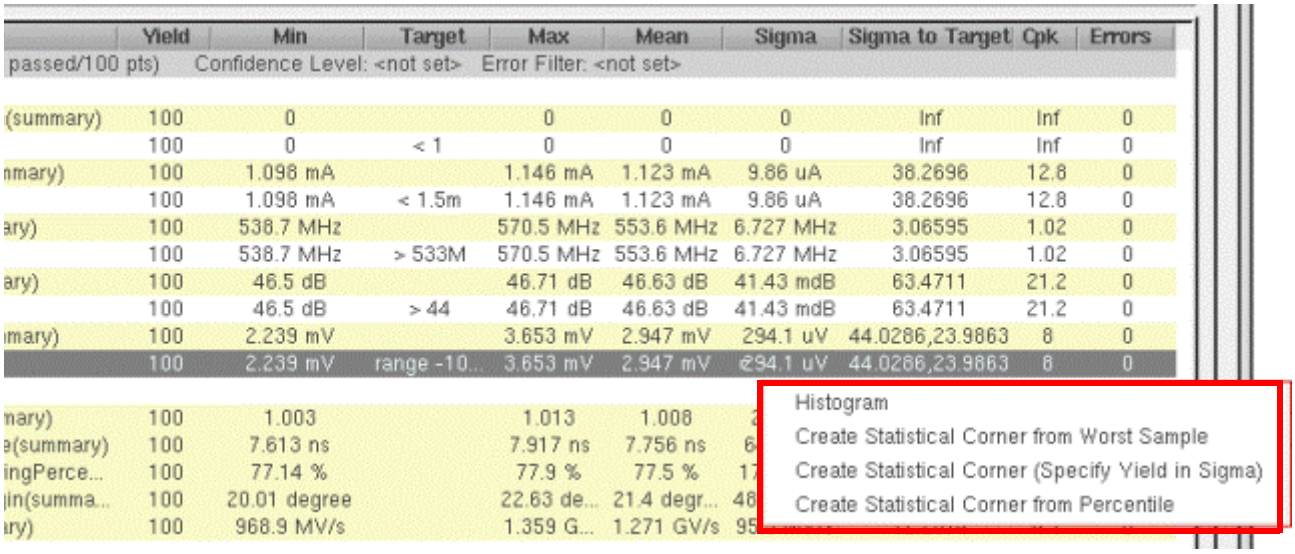
You can create statistical corners by using any of the following methods:

- From a selected sample in the Monte Carlo results
- From a worst sample
- From a percentile

For the first method, you can use the *Create Statistical Corner* command in the right-click context-sensitive menu of the *Detail* or *Detail – Transpose* results views. The commands

© 2023 Cadence Design Systems, Inc.

for these methods are available in the context-sensitive menu for the specification results in the *Yield* results view, as shown below.



For more details on these ways, refer to the topics given below.

Creating a Statistical Corner Using the Monte Carlo Results

By using the results of a Monte Carlo simulation, you can create statistical corners of two types:

- **Statistical corners based on statistical parameter values**: These corners are robust and do not need to be recreated if there are minor design changes. However, in this case, you need to save the statistical parameters while running the Monte Carlo simulations.
- **Statistical corners based on the sequence ID of a point**: These corners are not robust because if you make slight changes in the design, such as an addition of a new device instance, Spectre can no longer recreate the same statistical parameter values that were created for the original Monte Carlo run using the sequence ID.

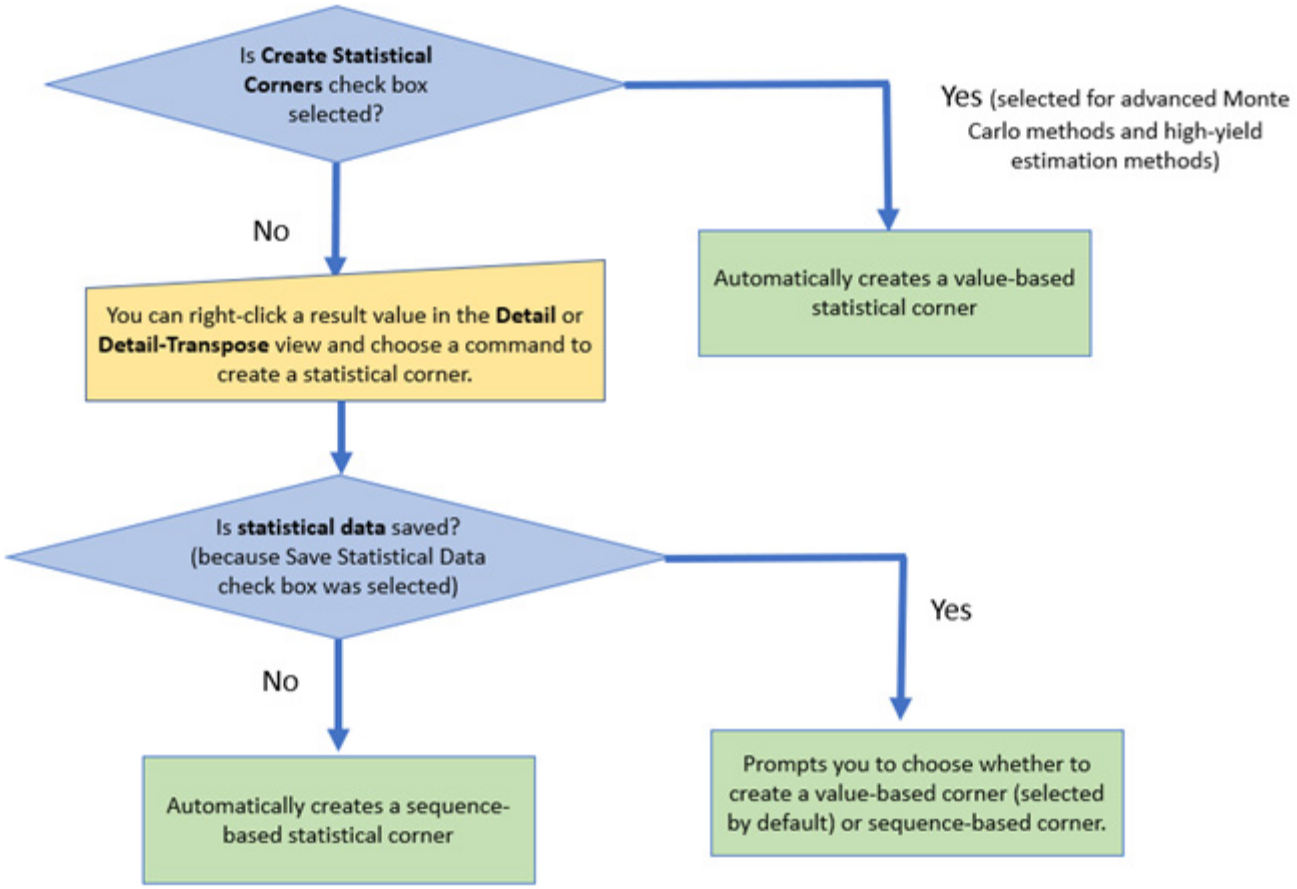
You can specify the default type for statistical corners by setting the value of the createStatisticalCornerType environment variable. By default, this variable is set to `"auto"`, which allows ADE Assembler to automatically choose a type for the statistical corner based on the status of the *Create Statistical Corner* check box. For details, see How does ADE Assembler choose a type for statistical corner when `createStatisticalCornerType` is set to `"auto"`?.

© 2023 Cadence Design Systems, Inc.

Set this variable to `"sequence"` to always create a statistical corner by using a sequence ID of a sample. Set it to `"values"` to always create a statistical corner by using the statistical parameter values of a sample. Set it to `"prompt"` to always display the Create Statistical Corner form in which you can confirm which one of the two types mentioned above you want to use to create a statistical corner.

How does ADE Assembler chooses a type for statistical corner when createStatisticalCornerType is set to "auto"?

The following figure describes the process followed by ADE Assembler to create a statistical corner when createStatisticalCornerType is set to `"auto"`:



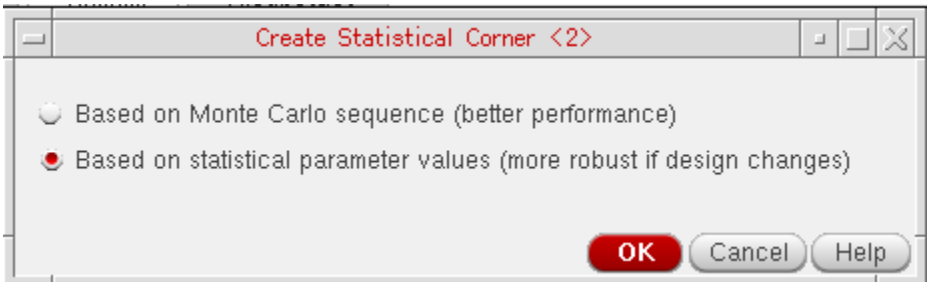
If the *Create Statistical Corners* check box is selected in the Monte Carlo options form and a high-yield estimation Virtuoso Variation Option (VVO) method is used to run a Monte Carlo simulation, ADE Assembler automatically creates values-based statistical corners.

If the *Create Statistical Corners* check box is not selected, you need to interactively create statistical corners from the results of a Monte Carlo run. For this, open the `Detail` or `Detail`

© 2023 Cadence Design Systems, Inc.

`- Transpose` results view, right-click a point and choose *Create Statistical Corner* from the context-menu. Depending on the status of the *Save Statistical Parameter Data* check box in the Monte Carlo options form for that run, ADE Assembler creates a statistical corner in two ways:

- If the *Save Statistical Parameter Data* check box was deselected, ADE Assembler automatically creates a sequence-based statistical corner by using the sequence of the selected point. The sequence-based corner is named as `Stat_Seq_<seq_ID>`, where `<seq_ID>` is the sequence ID of that sample. Whenever you resimulate the design, Spectre recreates the statistical corner based on the sequence ID.
- If the *Save Statistical Parameter Data* check box was selected, ADE Assembler displays the following dialog box prompting you to choose a method to create a corner:



By default, the *Based on statistical parameter values* option is selected.

You can also create a statistical corner for a particular point in the histogram for a specification. For this, plot a histogram for the desired specification, right-click a point for which you want to create a corner, and choose *Create Statistical Corner*. ADE Assembler creates a corner using the parameter values for that point.

Creating a Statistical Corner from a Worst Sample

In this method, ADE Assembler helps in identifying a sample that generated the worst result for a specification and creates a statistical corner using the sample details.

To create a statistical corner from the worst sample, do the following:

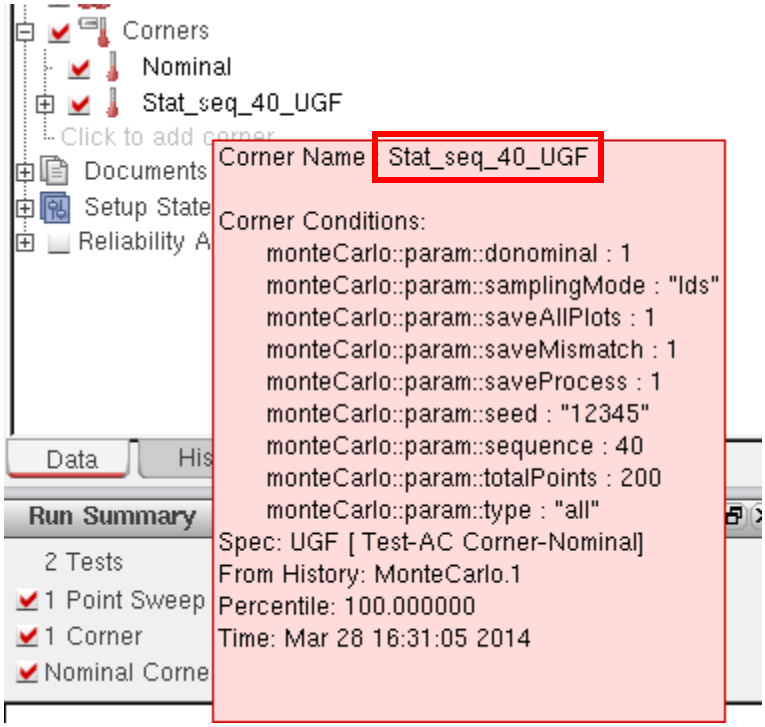
1. Open the Monte Carlo results in the *Yield* view and identify the specification for which you need to improve the yield.
2. Right-click the row for that specification and choose *Create Statistical Corner from Worst Sample*.

ADE Assembler looks into the simulation results to find a sample that gives the worst result for that specification and creates a statistical corner using that sample. The new corner is

© 2023 Cadence Design Systems, Inc.

added it to the *Corners* list in the Data View assistant pane. The corner is named as `Stat_Seq_<seq_ID>_<spec_name>`, where `<seq_ID>` is the sequence ID of that sample and `<spec_name>` is the name of the specification. For example, if you choose to create a corner for the worst sample for `UGF`, ADE Assembler creates a corner with the name `Stat_seq_40_UGF`, where the sequence ID of the sample is 40.

Note the details of the statistical corner in the following figure.



The figure given above shows that ADE Assembler saves the sequence ID, seed, and other Monte Carlo settings to the statistical corner when simulating it. Using this information, the simulator can recreate the statistical parameter values for the corner. This type of corner is efficient and does not require saving all the statistical parameter data for each sample in Monte Carlo. However, one disadvantage is that some changes to the design topology, such as, addition of a new instance to the schematic, can invalidate the corner. In that case, the simulator cannot recreate the same set of statistical parameter values when simulating the statistical corner. You would then need to rerun the Monte Carlo simulation to recreate statistical corners.

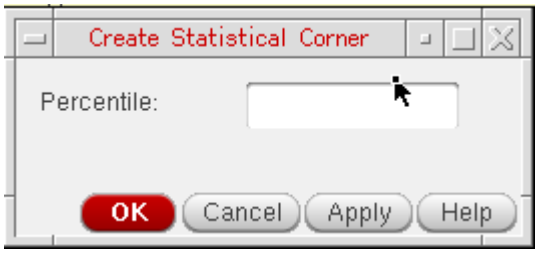
To avoid this, you can create statistical corners by directly saving the parameter values from the sample in the corner details and by not referring to any sequence ID. For more details, refer to Creating a Statistical Corner Using the Monte Carlo Results.

© 2023 Cadence Design Systems, Inc.

Creating a Statistical Corner from Percentile

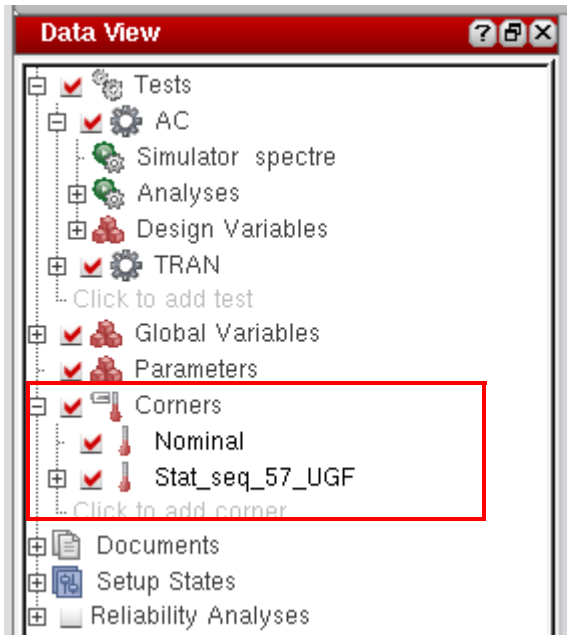
In this method, you can create a corner for the nth worst percentile for a given specification. That is, a statistical corner is created from the Monte Carlo sample that gives the nth worst output for a specification.

To create a statistical corner from percentile, right-click the output of a specification in the Yield view and choose Create Statistical Corner from Percentile. The Create Statistical Corner form is displayed, as shown in the following figure.



Specify a percentile value for which you want to create a sample and click OK.

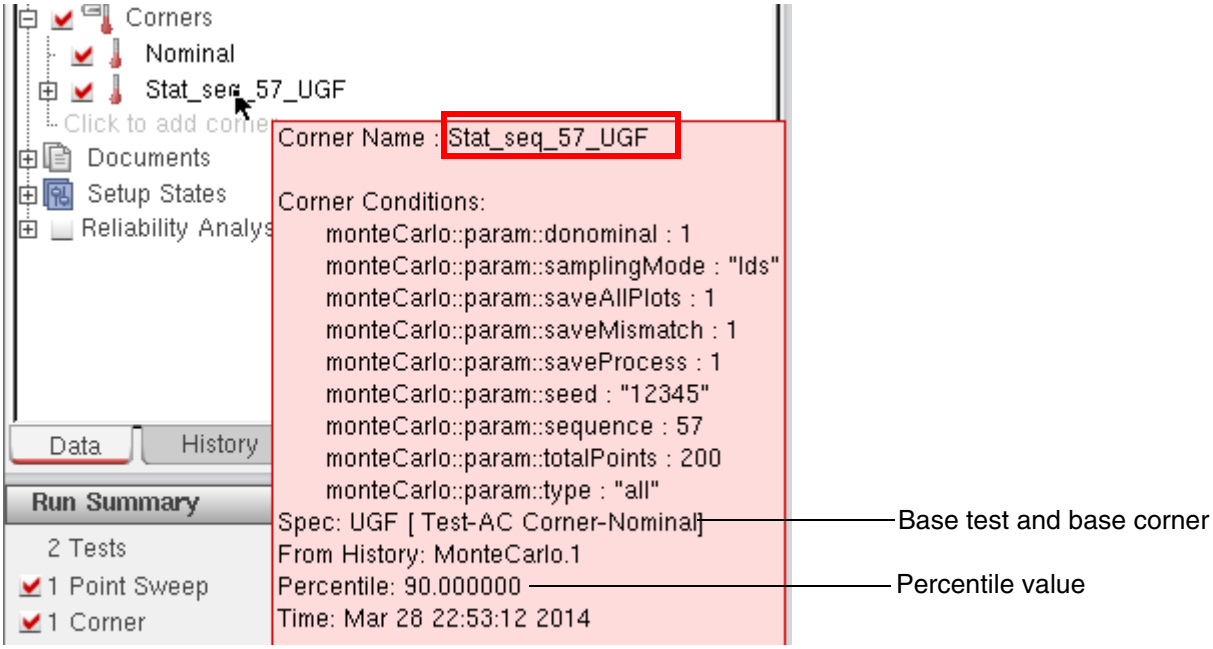
ADE Assembler creates a new statistical corner and adds it to the Corners node in the Data View assistant pane, as shown in the following figure.



The default name of a new statistical corner consists of a prefix `Stat_seq` followed by the sequence ID of the sample and the specification name. The tooltip for a corner displays the

© 2023 Cadence Design Systems, Inc.

values for all corner conditions, names of the base specification and base corner, and the percentile value used to create that corner, as shown in the following figure.



Important

If the specification type is < or >, only one statistical corner is created from the percentile. A < specification type represents a corner larger than the average value. A > specification type represents a corner smaller than the average value. If the specification type is `range` or if a specification is not defined, two corners are created–one in each direction.

By default, the corner uses the sequence ID of the sample that gives the nth worst output for a specification. Using this information, the simulator can then recreate the statistical parameter values for the corner when simulating it. This type of corner is efficient and does not require saving all the statistical parameter data for each sample in Monte Carlo. However, one disadvantage is that some changes to the design topology, such as, addition of a new instance to the schematic, can invalidate the corner. In that case, the simulator cannot recreate the same set of statistical parameter values when simulating the statistical corner. You would then need to rerun the Monte Carlo simulation to recreate statistical corners.

To avoid this, you can create statistical corners by directly saving the parameter values from the sample in the corner details and by not referring to any sequence ID. For more details, refer to Creating a Statistical Corner Using the Monte Carlo Results.

© 2023 Cadence Design Systems, Inc.

Important

A statistical corner is enabled only for the test associated with the specification for which the corner is created.

Creating Statistical Corners by Using Target Yield Value (Specified in Sigma)

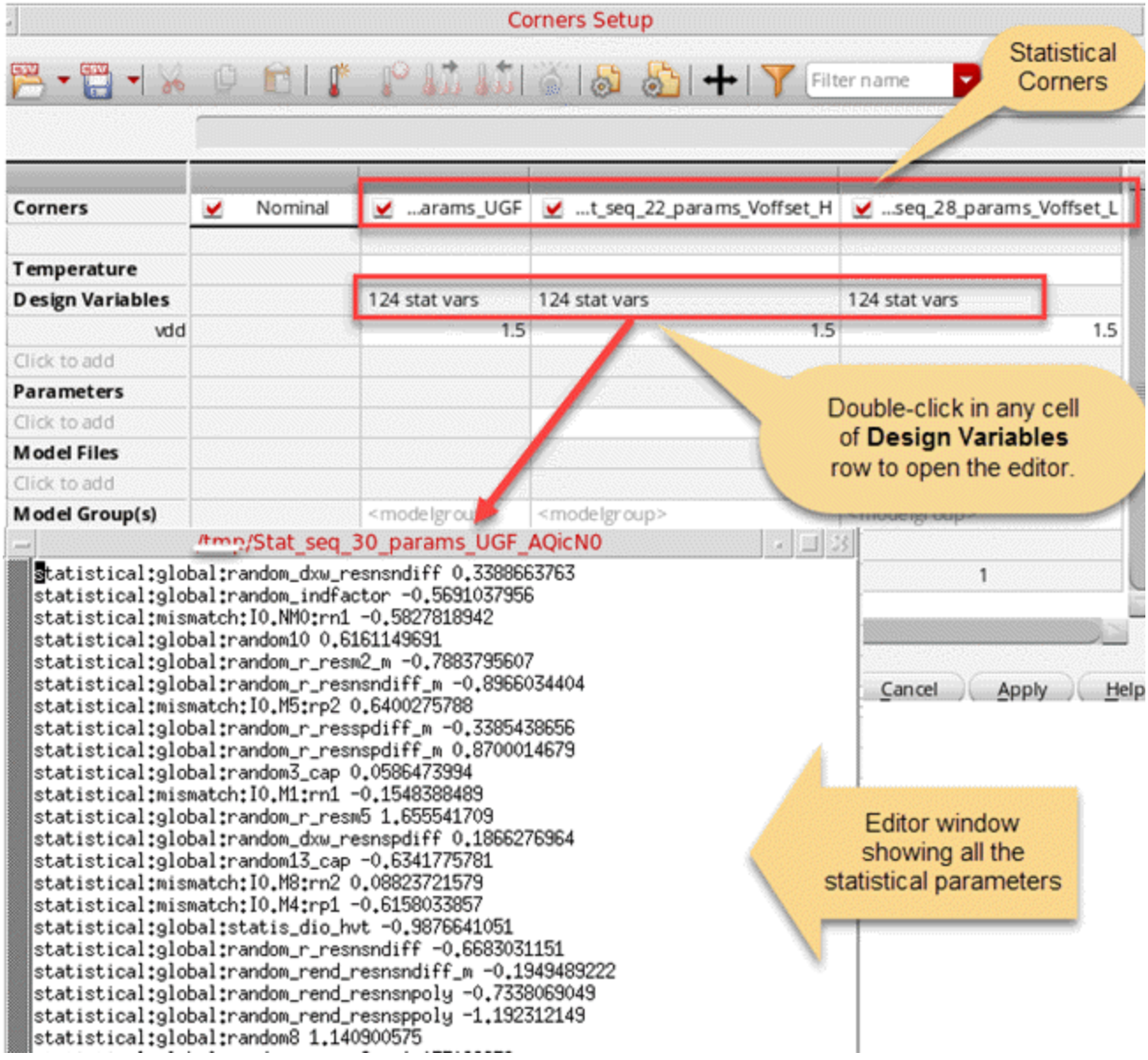
This is a fast method to create 3-sigma or user-specified sigma statistical corners without requiring a large number of Monte Carlo samples.

Note: Creating statistical parameters by this method requires the `Virtuoso Variation Option` license. For more information, see The K-Sigma Corners Method in *Virtuoso Variation Aware User Guide*.

© 2023 Cadence Design Systems, Inc.

Displaying Statistical Parameters of Statistical Corners

After you create the statistical corners, you can view them in the Corners section of the Data View assistant. You can also view them in the Corner Setup form, as shown below.



The *Design Variables* row displays the statistical parameters count in the `N stat vars` format, where `N` indicates the number of statistical parameters. To view the complete list of these parameters, double-click in the *Design Variables* cell for a statistical corner. The complete list of statistical parameters is displayed in the default text editor window for Virtuoso, as shown in the figure above.

© 2023 Cadence Design Systems, Inc.

Note: If the default text editor for Virtuoso specified by the `$EDITOR` shell variable in UNIX or the `editor` SKILL variable is Vim, the editor might appear blank. Either use the `-f` option with gvim by setting `editor="gvim -f"` in CIW or change the default text editor.

Filtering Out Data from the Yield View

By default, the statistical calculations displayed in the *Yield* view are based on all the points in Monte Carlo results. However, you can filter out and eliminate the points that were either partially complete or they resulted in simulation or evaluation errors so that the yield estimate is calculated only on the points for which simulation was successfully completed.

Important

A typical Monte Carlo error point represents a real failure of the design. Therefore, it is recommended to filter the error points only when it is certain that the error points are invalid.

To apply a filter in the *Yield* view, right-click the gray-colored row at the top and choose *Filter Points from Yield Estimate*.

© 2023 Cadence Design Systems, Inc.

Generating Plots, Tables, and Reports

ADE Assembler provides the following post-processing commands that you can use to print and plot the results data in different ways that suit your data analysis requirements:

- *Histogram*
- *Plot/Print versus Iteration*
- *Scatter Plot*
- *Print Correlation Tables*
- *Export Data to CSV*

You can find these commands in any of the following ways:

➡ Click on the Results tab, select a test and choose the required command.

➡ Open the *Yield* results view, right-click a test name, and choose the required command.

For details on these commands, refer to Generating Plots, Tables, and Reports in *Virtuoso ADE Explorer User Guide*.

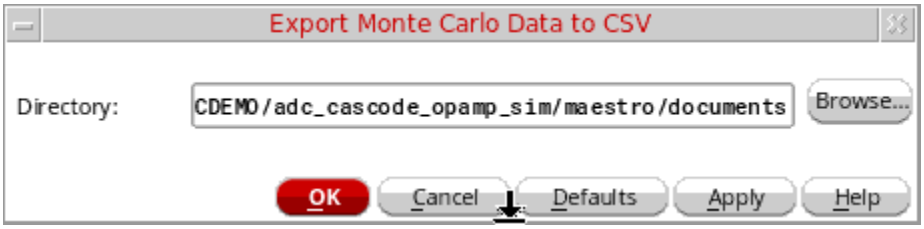
© 2023 Cadence Design Systems, Inc.

Saving Monte Carlo Results in CSV Format

After a Monte Carlo run is complete, you can save the results in CSV format. ADE Assembler saves the results for each corner in a separate `.csv` file. For this, do the following:

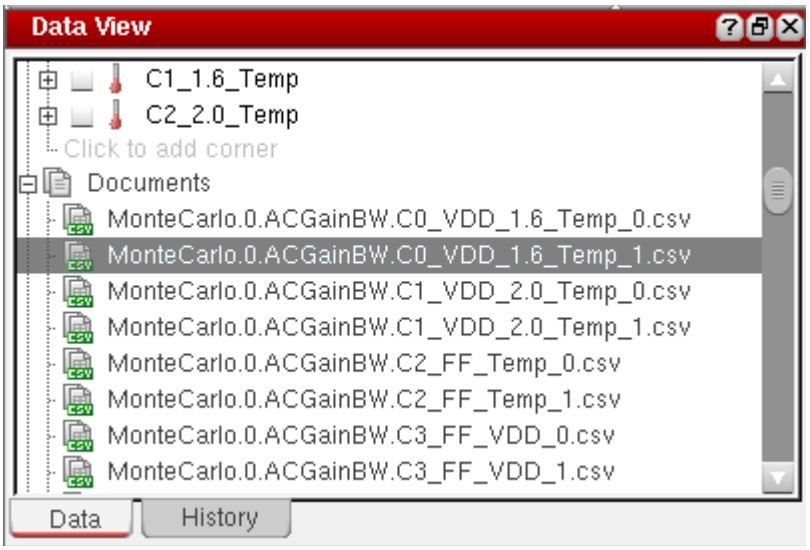
➡ In the History tab on the Data View pane, right-click a Monte Carlo run history and choose *Export Data to CSV*.

The Export Monte Carlo Data to CSV form is displayed.



The *Directory* field shows the default directory, `<path-to-the-maestro-view-dir>/documents`, where the `.csv` files are saved. If required, you can change this path and click *OK*.

ADE Assembler exports the output data for each corner in a separate `.csv` file and saves it in the specified directory. The names of these `.csv` files are also listed in the *Documents* list in the Data View assistant.



To open a `.csv` file fro the ADE Assembler window, right-click a file in the Data View assistant and choose *Open*. The file is opened in the default text editor.

© 2023 Cadence Design Systems, Inc.

Related Topics

axlWriteMonteCarloResultsCSV.

© 2023 Cadence Design Systems, Inc.

Introduction to Mismatch ID

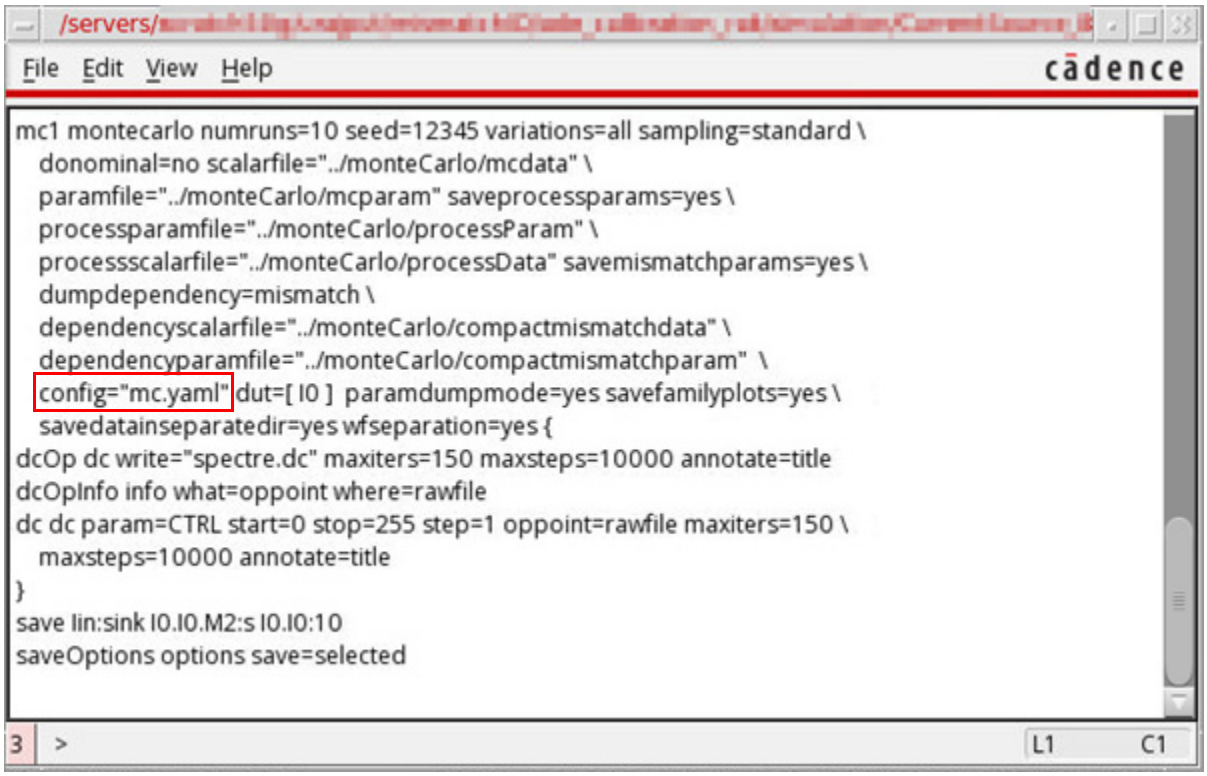
During Monte Carlo analysis, Spectre generates random parameter values based on the defined model distributions. Circuit calibration and yield estimation in a multi-test setup require that identical mismatch variation is applied to the design under test for each Monte Carlo sample point.

By specifying a mismatch ID, you can ensure that the same values for each statistical parameter are generated across all tests in your setup. You can specify a mismatch ID either at the subcircuit level or at the instance level.

The Configuration File for Mismatch ID

When you run Monte Carlo analysis after specifying mismatch ID, a configuration file with the name `mc.yaml` is created in the netlist directory of the simulation run.

`"mc.yaml"` is specified as the value of the Spectre `montecarlo` parameter `config` in the netlist, as shown in the following figure.



© 2023 Cadence Design Systems, Inc.

The contents of the `mc.yaml` file for a test are as follows:

Scope: Subcircuit	**Scope: Instance**

`# Monte Carlo Yaml configuration for test: nameOfTheTest`
`version: versionNumber`
`montecarlo:`
`  mismatchid:`
`  sub:`
`  nameOfTheSubcircuit: mismatchID`
	`# Monte Carlo Yaml configuration for test: nameOfTheTest`
	`version: versionNumber`
	`montecarlo:`
	`  mismatchid:`
	`  inst:`
	`  nameOfTheInstance: mismatchID`

Example:	**Example:**

`# Monte Carlo Yaml configuration for test: dc`
`version: 1.0`
`montecarlo:`
`  mismatchid:`
`  sub:`
`  DUTtop: 1`
	`# Monte Carlo Yaml configuration for test: dc`
	`version: 1.0`
	`montecarlo:`
	`  mismatchid:`
	`  inst:`
	`  I0: 1`

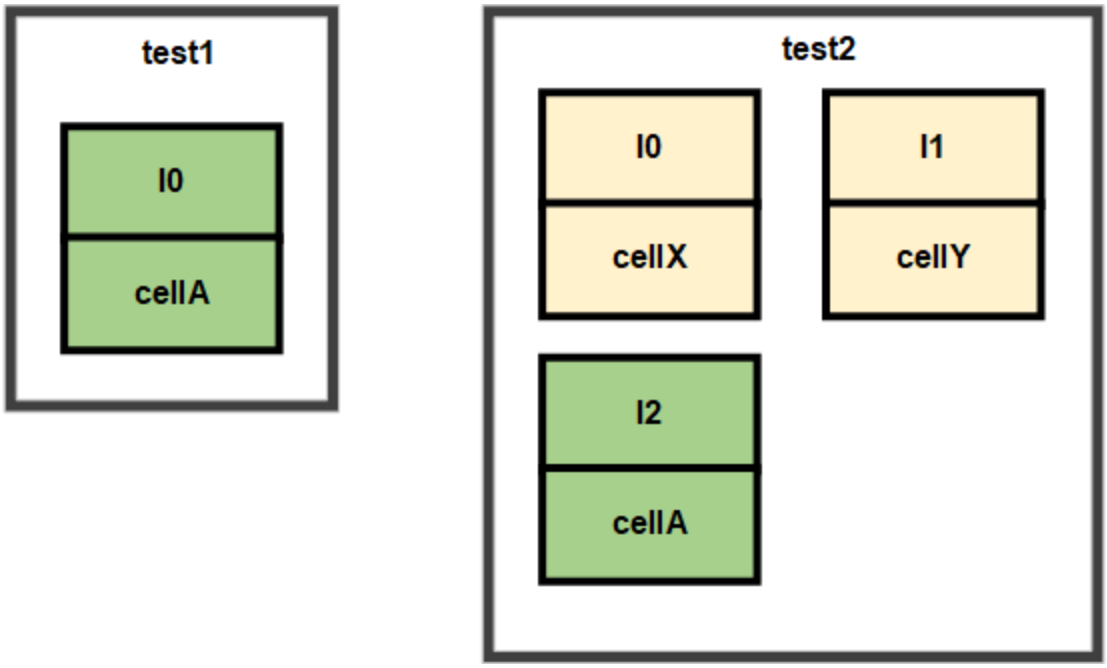
Examples of Mismatch ID Configuration

This section provides some examples of how you can configure mismatch IDs to ensure that statistical parameter values for instances or subcircuits in your design are the same across all tests in your setup.

© 2023 Cadence Design Systems, Inc.

Example 1

Consider the following design in which you want to generate the same values of statistical parameters for the blocks highlighted in green.



© 2023 Cadence Design Systems, Inc.

You can do so by specifying the same mismatch ID either at the subcircuit level or at the instance level for both the tests, as shown in the following YAML configurations for each test.

Scope: Subcircuit

```
# Monte Carlo Yaml configuration for
test: test1
version: 1.0
montecarlo:
  mismatchid:
    sub:
      cellA: 1
```

```
# Monte Carlo Yaml configuration for
test: test2
version: 1.0
montecarlo:
  mismatchid:
    sub:
      cellA: 1
```

Scope: Instance

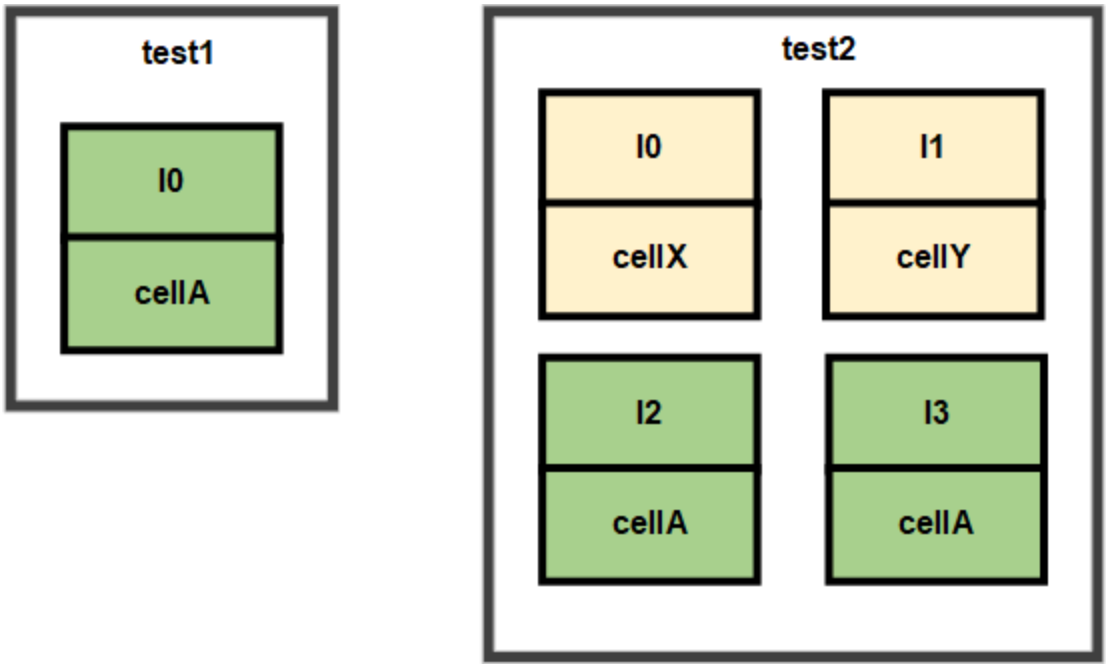
```
# Monte Carlo Yaml configuration for
test: test1
version: 1.0
montecarlo:
  mismatchid:
    inst:
      I0: 1
```

```
# Monte Carlo Yaml configuration for
test: test2
version: 1.0
montecarlo:
  mismatchid:
    inst:
      I2: 1
```


© 2023 Cadence Design Systems, Inc.

Example 2

Consider the following design in which you want to generate the same values of statistical parameters for the blocks highlighted in green.



© 2023 Cadence Design Systems, Inc.

You can do so by specifying the same mismatch ID either at the subcircuit level or at the instance level for both the tests, as shown in the following YAML configurations for each test.

Scope: Subcircuit

```
# Monte Carlo Yaml configuration for
test: test1
version: 1.0
montecarlo:
  mismatchid:
    sub:
      cellA: 1
```

```
# Monte Carlo Yaml configuration for
test: test2
version: 1.0
montecarlo:
  mismatchid:
    sub:
      cellA: 1
```

Scope: Instance

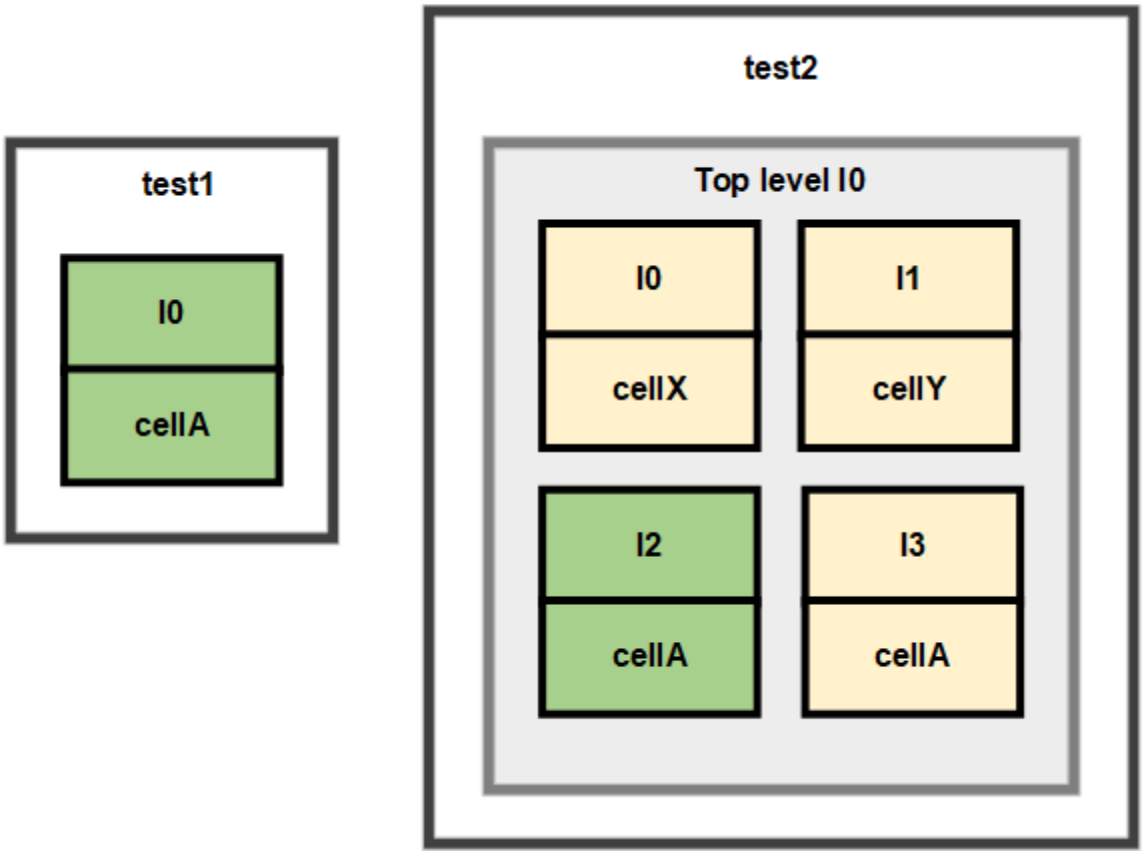
```
# Monte Carlo Yaml configuration for
test: test1
version: 1.0
montecarlo:
  mismatchid:
    inst:
      I0: 1
```

```
# Monte Carlo Yaml configuration for
test: test2
version: 1.0
montecarlo:
  mismatchid:
    inst:
      I2: 1
      I3: 1
```


© 2023 Cadence Design Systems, Inc.

Example 3

Consider the following design in which you want to generate the same values of statistical parameters for the blocks highlighted in green.



© 2023 Cadence Design Systems, Inc.

You can do so by specifying the same mismatch ID at the instance level for both the tests, as shown in the following YAML configurations for each test.

Scope: Instance

```
# Monte Carlo Yaml configuration for
test: test1
version: 1.0
montecarlo:
  mismatchid:
    inst:
      I0: 1
```

```
# Monte Carlo Yaml configuration for
test: test2
version: 1.0
montecarlo:
  mismatchid:
    inst:
      I0.I2: 1
      I0.I3: 2
```

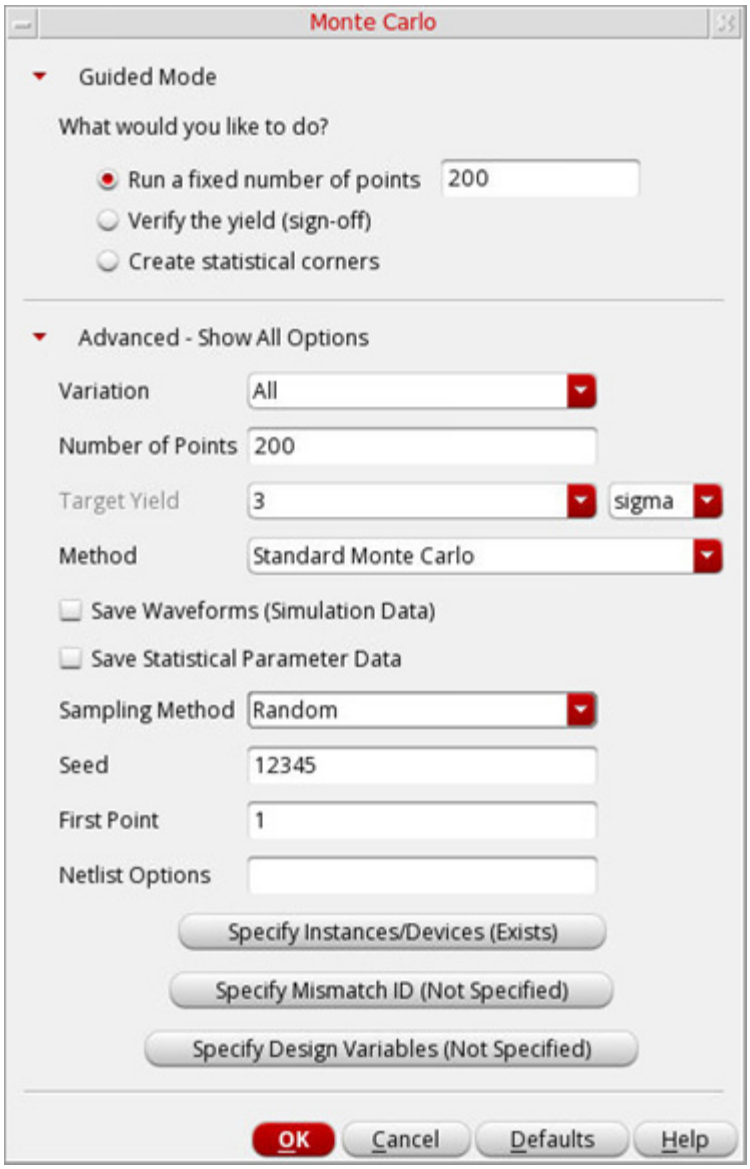

© 2023 Cadence Design Systems, Inc.

Specifying a Mismatch ID

To specify a mismatch ID:

1. Open your design in ADE Assembler or ADE Explorer.
2. Select *Monte Carlo Sampling* from the *Run Mode* drop-down list.
3. Click the *Simulation Options* command.

   The Monte Carlo form opens.



4. From the *Sampling Method* drop-down list, select *Random*.

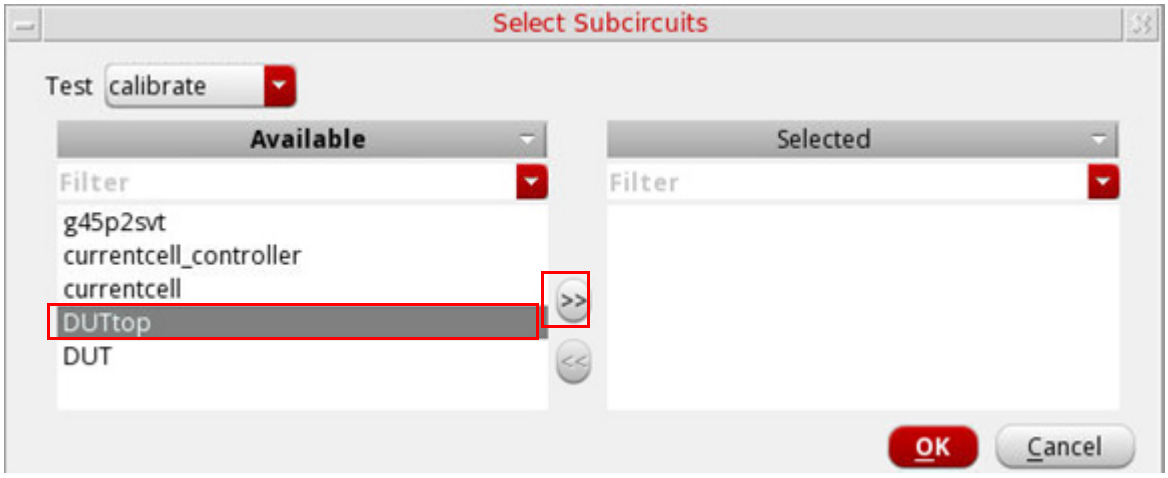
© 2023 Cadence Design Systems, Inc.

5. Click *Specify Mismatch ID (Not Specified)*.

   The Mismatch ID form opens.

6. Select the *Enable Mismatch ID* check box.
7. In the *Test* column, click a cell to select either *All Tests* or a specific test in your setup.
8. In the *Scope* column, double-click a cell to select either *Subcircuits* or *Instances.*
9. In the *Data* column, double-click a cell, and then click ... to select the subcircuit or instance for which you want to specify the mismatch ID. Depending upon what you chose in step 8, either the Select Subcircuits or the Select Instances form opens.

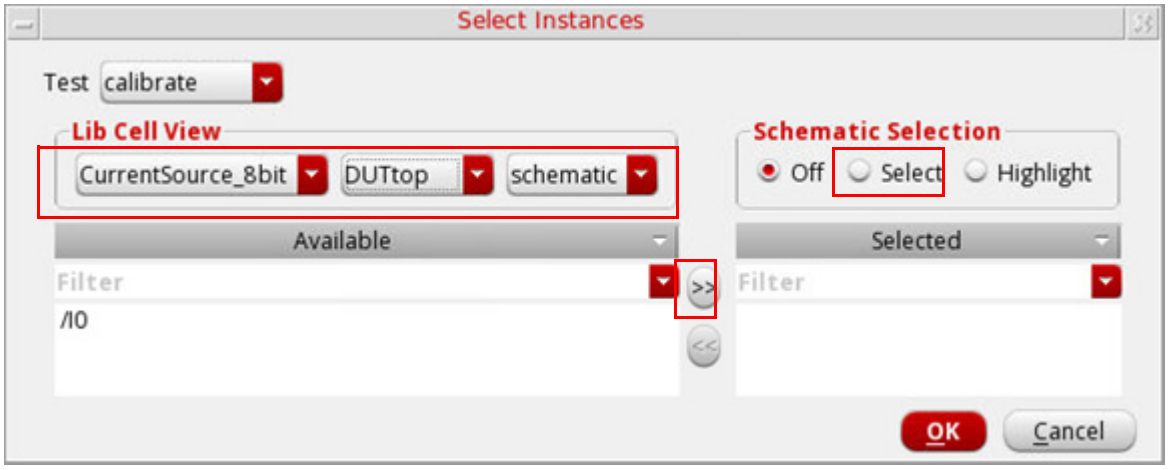
   - On the Select Subcircuits form, from the list of available subcircuits, select a circuit and then click >> to add it to the selected list. Click *OK* to return to the Mismatch ID form.



   - On the Select Instances form, select a library cellview to list available instances. From the list of available instances, select an instance and then click >> to add it to

© 2023 Cadence Design Systems, Inc.

the selected list. Alternatively, you can click *Select* in the *Schematic Selection* section to select an instance from the schematic.



10. In the *ID* column, specify an integer value as the mismatch ID. For example, `1`.

11. In the *Enable* column, ensure that the check mark is selected.

12. Repeat steps 6–11 to add other subcircuits or instances.

Note: Ensure that you specify the same mismatch ID against the subcircuits and instances for which you want to generate the same values of statistical parameters across all tests.

13. Click *OK* to close the Mismatch ID form.

14. Click *OK* on the Monte Carlo form.

15. Click *Run Simulation*.

© 2023 Cadence Design Systems, Inc.

The Mismatch ID Form

The Mismatch ID form lets you specify mismatch IDs for the instances and subcircuits so that the values of statistical parameters generated for those instances or subcircuits remains same across all tests.

Field	Description
Enable Mismatch ID	Enables mismatch IDs.
Import	Imports the mismatch ID configuration from an existing YAML configuration file. This is helpful when you do not want to manually configure mismatch IDs one by one.
Export	Exports the current mismatch ID configuration on the Mismatch ID form to a YAML file for the specified test. This is helpful when you want to reuse an existing mismatch ID configuration defined on the Mismatch ID form.
Tests	Selects all tests or a specific test from the setup.
Scope	Specifies whether to define the mismatch ID at the subcircuit level or at instance level.
Data	Selects the subcircuit or the instance for which you want to specify the mismatch ID.
ID	Specifies an integer value as the mismatch ID.
Enable	Enables the specified mismatch ID.

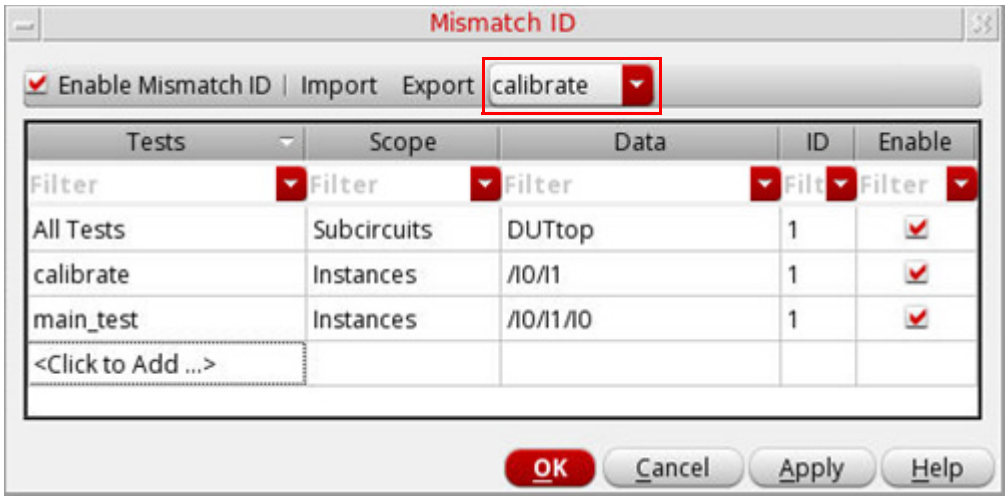
© 2023 Cadence Design Systems, Inc.

Exporting a Mismatch ID Configuration to a YAML File

After you have specified mismatch IDs on the Mismatch ID form, you can export this configuration to a YAML file that you can later reuse by importing the file.

To export a mismatch ID configuration to a YAML file:

1. On the Mismatch ID form, select a test for which you want to export the mismatch ID configuration from the drop-down list next to the *Export* command.



2. Click *Export* to save the mismatch ID configuration defined on the form to a YAML file.

   The Select File for Mismatch ID Export form opens.

3. Select a directory where you want to save the YAML file.
4. In the *File name* field, specify the name of the file.
5. Click *Save* to save the mismatch ID configuration to a YAML file.

© 2023 Cadence Design Systems, Inc.

Importing a Mismatch ID Configuration from a YAML File

You can reuse a YAML file that you saved earlier to configure mismatch IDs without manually specifying them for each subcircuit and instance one-by-one.

To import a mismatch ID configuration:

1. On the Mismatch ID form, click the *Enable Mismatch ID* check box.
2. Click *Import*.

   The Select File for Mismatch ID Import form opens.
3. Select the YAML file containing the mismatch ID configuration that you want to import.
4. Click *Open*.

   The mismatch IDs as defined in the YAML file are updated in the Mismatch ID table.

© 2023 Cadence Design Systems, Inc.

Advanced Monte Carlo Methods

When you have the have the `Virtuoso Variation Option` (VVO) license, you can use the following advanced methods that are available on the Monte Carlo run options form:

- Yield verification using sample reordering - to verify the yield without simulating such a large number of samples
- Scaled-sigma sampling method - for estimation of a high yield (> 3 sigma) by generating samples where the standard deviation has been scaled up.
- Worst case distance method - for estimation of a high yield (> 3 sigma) by finding the shortest distance, also called the Worst Case Distance or WCD, from the nominal point to the specification boundary in the process/mismatch parameter space.
- K-sigma corners method - for fast creation of statistical corners when the number of statistical parameters is large (> 1000)
- Worst samples method - for creation of statistical corners based on the sample reorder method
- Sensitivity Accuracy method for mismatch variation - to identify the devices that are important contributors to mismatch variation and to make the design less sensitive to mismatch variation.
- Confidence Interval - Auto Stop - to automatically stop Monte Carlo analysis based on confidence interval of outputs standard deviation

For more information about these methods, refer to the Virtuoso Variation Option User Guide.

© 2023 Cadence Design Systems, Inc.

12

Working with the Simulation Setup

This chapter describes the following topics:

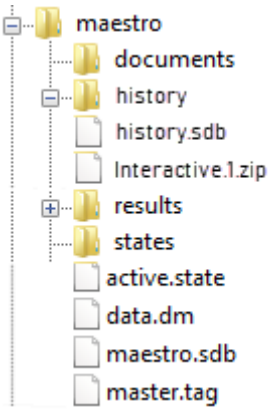
- Directory Structure of a Maestro Cellview on page 514
- Saving the Simulation Setup on page 518
- Copying a maestro Cellview on page 519
- Deleting a maestro View on page 523
- Working with Read-Only or Editable maestro Views on page 524
- Importing the Simulation Setup on page 532
- Working with Setup States on page 536
- Running a Simulation Using a Setup State on page 547
- Creating a Plan Using Setup States on page 548

© 2023 Cadence Design Systems, Inc.

Directory Structure of a Maestro Cellview

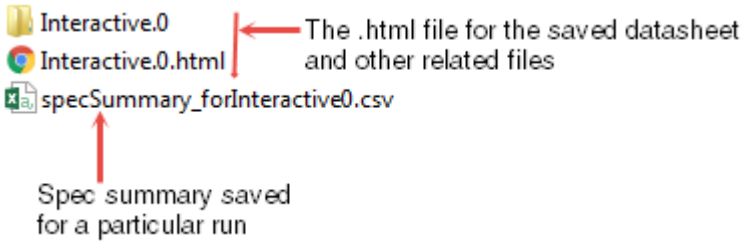
The simulation setup in ADE Assembler is saved in a maestro cellview. The default name for this type of cellview is `maestro`.

A maestro cellview has the following directory structure:



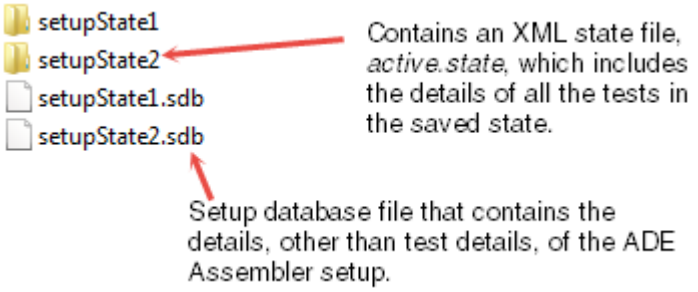
The following table describes the contents of files and directories in the maestro cellview.

Table 12-1 Contents of a Maestro Cellview

Directory/File	Description
`documents`	A directory containing the documents you created in the maestro view, including the specification summaries and datasheets you created in ADE Assembler. 

© 2023 Cadence Design Systems, Inc.

Table 12-1 Contents of a Maestro Cellview

Directory/File	Description
`results`	A directory containing the results database (`.rdb`) and run log (`.log`) files for all the history checkpoints saved from the maestro view. The `.rdb` file contains the results of all the scalar outputs evaluated using the simulation results returned by the simulator. These details are used to populate the *Results* tab in the ADE Assembler environment window. The `.log` file contains the summary of the simulations run for the saved checkpoint. Simulation results are not saved in the `results` directory. For information about the location of simulation results information, see Specifying Options for Saving Simulation Results. You can specify a different location where you want the program to save the results database and run log files. For this, do one of the following: ■ Use the *ADE Assembler Results Database Location* field in the Save Options form. For more information, see Specifying Results Database Location. ■ Use the saveDir environment variable. If you do not specify a different location and open the maestro view in read-only mode or you do not have write permissions for the maestro view, the program writes this information to *`libraryName/cellName`*`/viewName/maestro/data` in the location specified using the `asimenv.startup projectDir` environment variable. The default setting for this environment variable is `$HOME/simulation`.
`states`	A directory containing the setup states you created in ADE Assembler. For every state, a `.sdb` file and a directory is saved. 
`active.state`	XML state file that contains the details of all the tests in the **currently active** ADE Assembler setup.

© 2023 Cadence Design Systems, Inc.

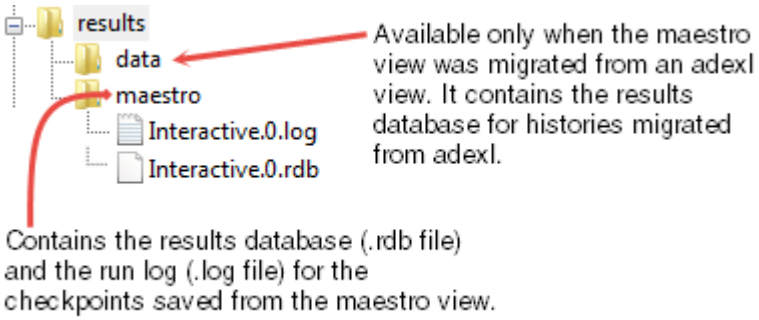
Table 12-1 Contents of a Maestro Cellview

Directory/File	Description
`maestro.sdb`	The main setup database file that contains the setup details, other than test details, of the currently active ADE Assembler setup. The history setup information for each history is saved in a `.zip` file in the `history` directory.
`maestro.tag`	Contains master information for the maestro cellview.
`history`	This directory contains `history.sdb` and `.zip` files for each history. Each zip file contains the setup information for a history. The compressed format of saving history setup saves space. `history.sdb` maintains an inventory list of all history checkpoints saved for a cellview.
`data.dm`	The property bag file that contains the properties of the maestro cellview.

Additional Files or Directories Visible in a Maestro View Migrating from ADE XL

When you migrate an adexl view to a maestro view for ADE Assembler, the following additional files or directories are also visible inside the `maestro` view:

- `data`: This directory is created under the `results` directory. This contains the `.rdb` and `.log` files for the history checkpoints saved in the ADE XL view.



- Additional files and directories under `test_states`:
  - `<library-name>\<cell-name>\<simulator-name>` directories for the histories migrated from the ADE XL view.
  - `<history-name>.ScriptsDir` directory that contains the pre-run scripts for the history migrated from ADE XL.

© 2023 Cadence Design Systems, Inc.

Note: The setup states saved from the ADE XL view are copied to the `state` directory in the maestro view. While copying these setup states, the number of files are reduced and all the state information is condensed into a single XML file called `active.state` under the directory with same name as that of the state name.

© 2023 Cadence Design Systems, Inc.

Saving the Simulation Setup

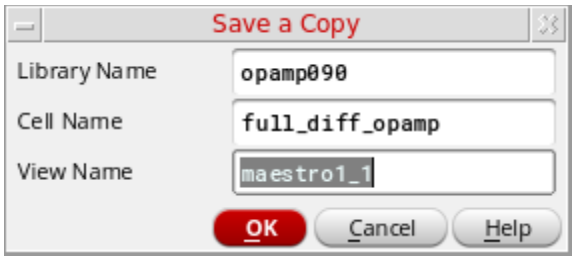
To save the simulation setup, do the following

➤ Choose *File – Save*.

To save the simulation setup to a different maestro view, do the following:

1. Choose *File – Save A Copy*.

   The Save A Copy form appears.



2. Enter the name of the library, cell and view in which you want to save the setup.
3. Click *OK*.

Note: This action does not save history data, only the setup database.

See also

■ Saving Setup Changes in Read-Only maestro Views on page 530

© 2023 Cadence Design Systems, Inc.

Copying a maestro Cellview

Do not copy maestro views using file system commands or a graphical file manager. This can result in loss of data from previous simulation runs.

You can use Library Manager to copy maestro views. Use one of the methods described in Table 12-1 when you:

- Copy a maestro view
- Copy a cell containing a maestro view
- Copy a library in which any cell has a maestro view

Note: This section describes only the procedures for copying a maestro view. The procedures for copying libraries or cells containing maestro views are similar. For more information about copying libraries and cells, see the *Cadence Library Manager User Guide*.

Table 12-2 Methods of Copying maestro **Views**

Method	Purpose
Copy only the simulation setup, but not the results database, run log files, and simulation results for the history items.	Use this method if you only want to reuse the simulation setup existing in a maestro view, and do not want to view the simulation results, plot waveforms, or do other tasks that need access to simulation results in the new maestro view. For more information, see Copying only the Simulation Setup in a maestro View. **Note:** If the results database exists within the maestro view that you are copying, you can view the simulation results in the new maestro view. However, you cannot plot waveforms or do other tasks that need access to simulation results in the new maestro view. For more information about the location of the results database, see Specifying Results Database Location.

© 2023 Cadence Design Systems, Inc.

Method	Purpose
Copy the simulation setup and the results database	Use this method if you want to reuse the simulation setup and view the simulation results, but do not want to plot waveforms or do other tasks that need access to simulation results in the new maestro view. For more information, see Copying the Simulation Setup and the Results Database in a maestro Cellview on page 521.
Copy everything (the simulation setup, results database, run log files and the simulation results) in the maestro view	Use this method if you want to make a complete copy of an maestro view, or create a backup of an maestro view. For more information, see Copying Everything Contained in a maestro Cellview on page 521.

Copying only the Simulation Setup in a maestro View

To copy only the simulation setup in a maestro cellview, do the following.

1. (Optional) If you want to run triggers set for the copy operation, set the following environment variables in the `.cdsenv` file:
   - `ddserv.lib enableCopyInDFII`: Specifies that the command to copy cellview is to be run within the DFII session instead of Library Manager. This allows the command to run the triggers registered for it.
   - `cdsLibManager.copyGlobals mpsRadio`: Specifies whether the copy operation uses the copy service of the session, when available, or the local copy engine in Library Manager. This also allows the *Copy* command of Library Manager to run the triggers.
2. Start the Virtuoso design environment.
3. In the CIW, choose *Tools – Library Manager*.

   The Library Manager form appears.
4. In the *Library* column, select the library in which the maestro view exists.
5. In the *Cell* column, select the cell in which the maestro view exists.
6. In the *View* column, select the maestro view.
7. Choose *Edit – Copy*.

© 2023 Cadence Design Systems, Inc.

The Copy View form appears.

8. In the *To* group box, type specify the destination library, cell and view name.
9. Click *OK* to copy the maestro view.

Copying the Simulation Setup and the Results Database in a maestro Cellview

To copy the simulation setup and the results database in an maestro view, do the following:

1. Perform steps 3 to 8 described in Copying only the Simulation Setup in a maestro View.
2. Select the *Update Instances* check box.

   The drop-down list becomes active.
3. From the drop-down list, select *Of New Copies Only*.
4. Click *OK* to copy the maestro view.

See also:

- How is the Results Database Copied for a maestro Cellview?

Copying Everything Contained in a maestro Cellview

To copy everything that is contained in a maestro cellview, do the following:

1. If you have used the `asimenv.startup projectDir` environment variable to specify the location for storing simulation results, ensure that the variable is set in your `.cdsenv` file.

   For more information about specifying the location for storing simulation results, see the following topics:

   - Specifying Options for Saving Simulation Results
   - saveResDir
2. Set the `adexl.cpupdtr copyResultsData` environment variable in the `.cdsenv` file.

   Note: The `copyResultsData` variable is used by both Virtuoso and a supporting utility. Therefore, it is recommended to set it in the ~/`.cdsenv` file that is read by both. It is not sufficient to set the variable in `.cdsinit`.

© 2023 Cadence Design Systems, Inc.

3. Launch Virtuoso.
4. Copy the maestro view using the procedure described in Copying the Simulation Setup and the Results Database in a maestro Cellview.

See also:

- How is the Results Database Copied for a maestro Cellview?
- How are Simulation Results Copied for a maestro Cellview?

How is the Results Database Copied for a maestro Cellview?

You can copy the results database contained in an maestro cellview using one of the following procedures:

- Copying the Simulation Setup and the Results Database in a maestro Cellview
- Copying Everything Contained in a maestro Cellview

The results database can be located inside or outside the maestro view. For more information about the location of the results database, see Specifying Results Database Location.

- If the results database is located inside the maestro view, it is copied into the new maestro view.
- If the results database is located outside the maestro view, it will not be copied into the new maestro view. Instead, a copy of the results database will be created in the specified results database location. For example, if the results database for the `maestro` view in the `myCell` cell of the `myLib` library is located outside the `adexl` view at:

  ```
  $HOME/myResultsDatabase/myLib/myCell/maestro/results/data
  ```

  and you copy the `maestro` view to a new view called `maestro_copy`, the results database for the `maestro_copy` view will be located at:

  ```
  $HOME/myResultsDatabase/myLib/myCell/maestro_copy/results/data
  ```

How are Simulation Results Copied for a maestro Cellview?

Simulation results are always located outside the maestro view. For more information about the location of simulation results, see Specifying Options for Saving Simulation Results.

When an maestro view is copied using the procedure described in Copying Everything Contained in a maestro Cellview, simulation results will not be copied into the new maestro view. Instead, a copy of the simulation results is created in the specified simulation results

© 2023 Cadence Design Systems, Inc.

location. For example, if the simulation results for the `maestro` view in the `myCell` cell of the `myLib` library is located outside the `maestro` view at:

`$HOME/mySimulationResults/myLib/myCell/maestro/results/data/<history_item>`

and you copy the `maestro` view to a new view called `maestro_copy`, the simulation results for the `maestro_copy` view will be located at:

`$HOME/myResultsDatabase/myLib/myCell/maestro_copy/results/data/<history_item>`

Deleting a maestro View

To delete an maestro view, do the following:

1. In the CIW, choose *Tools – Library Manager*.

   The Library Manager form appears.
2. In the *Library* column, select the library in which the maestro view exists.
3. In the *Cell* column, select the cell in which the maestro view exists.
4. In the *View* column, select the maestro view.
5. Choose *Edit – Delete*.

   The Delete Cell Views form appears.
6. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Working with Read-Only or Editable maestro Views

Tip

Read a blog related to this feature at Virtuosity: Read Mode Done Right.

When working with a read-only maestro cellview, you can modify the setup and run simulations, but you cannot save the changes at the same locations as you can do for an editable cellview.

The following sections describe how you can work with read-only maestro views:

- Opening maestro Views in Read-Only Mode
- Opening maestro Views in Edit mode
- Opening maestro Views for which You Do Not Have Write Permissions
- Running Simulations from Read-Only maestro Views
- Saving Setup Changes in Read-Only maestro Views

Opening maestro Views in Read-Only Mode

There are two ways to open a maestro view in read-only mode.

- Choose *File—Make Read Only*.

  Note the following:

  - This option is available only for ADE Assembler.
  - While switching from edit mode to read-only mode, if you have unsaved changes in your cellview, an error message is displayed asking you to confirm whether you want to save or discard the changes made in the active cellview.

- In Library Manager, perform the following steps.

  a. Use the *Library*, *Cell*, and *View* fields to select your maestro view.

  b. In the *View* field, right-click the maestro view and choose *Open (Read-Only)*.

  If the cellview contains one test, it is opened in read-only mode in ADE Explorer. If it contains multiple tests, it is opened in read-only mode in ADE Assembler. The title bar of the window displays the text *Reading* to indicate that the cellview is opened in read-only mode.

© 2023 Cadence Design Systems, Inc.

Note: In addition to the ADE Explorer or ADE Assembler license, you need a VSE L/XL license to open a maestro view in read-only mode.

Opening maestro Views in Edit mode

By default, a maestro view is opened in edit mode, but if you want to make a read-only maestro view editable,

➥ Choose *File—Make Editable*.

While switching from a read-only maestro view to an editable maestro view,

- If you have another editable session open in the background, you will receive an error message prompting you to choose between the following options:
  - Work in the open session.
  - Close the open session and then change the cellview in the current session to edit mode.
- If that cellview has changed on disk while the read-only session was open, you will receive a warning message prompting you to choose between the following options:
  - Overwrite the on-disk changes and switch the mode.
  - Keep the session in read-only mode and possibly save those changes in a different cellview later.

If a maestro cellview is DM-managed, the following case are possible:

- If the cellview status is set to *CheckedIn*, a dialog box is displayed prompting you to choose between the following options:
  - *Yes*: The cellview is checked out on the basis of the check out preferences specified through the Auto Checkout Preferences form. If the cellview gets checked out successfully, you can switch to edit mode.
  - *No*: An error message gets displayed in the CIW.
- If the cellview is checked out by some other user, an error message is displayed in the CIW stating that the cellview cannot be switched to edit mode.

Note: In edit mode, the *Save* button is enabled whenever there are any unsaved changes. However, in certain design management systems, you might observe that *Save* is not enabled. In such scenarios, you can set the alwaysEnableSaveButton environment variable to `t` to always keep the *Save* button enabled in edit mode.

© 2023 Cadence Design Systems, Inc.

Viewing the DM Status of a Cellview

If a cellview is managed by a Design Management (DM) system, you can view the status and version number of the cellview in the title bar.

There are three possible values:

- *CheckedIn*: specifies that a cellview is available for check out to a workarea.
- *CheckedOut*: specifies that a cellview has been checked out to a workarea.

  Note: If you try to open a cellview that is already checked out by another user, a local copy of the cellview is displayed with the status *CheckedOut* (*`userID`, where `userID` represents the user ID of the user who checked out the cellview.
- *UNMANAGED*: specifies that a cellview is not managed by a DM system.

Note: The behavior of this feature can vary according to the used DM system.

Opening maestro Views for which You Do Not Have Write Permissions

When you open a maestro view for which you do not have write permissions, the following message appears:

```
Could not open <library_name> <cell_name> <view_name> for edit (no data or lock
available). Would you like to open it for read?
```

Click *Yes* to open the view in read-only mode.

For more information about opening maestro views, see Opening an Existing Setup.

Important Points to Note

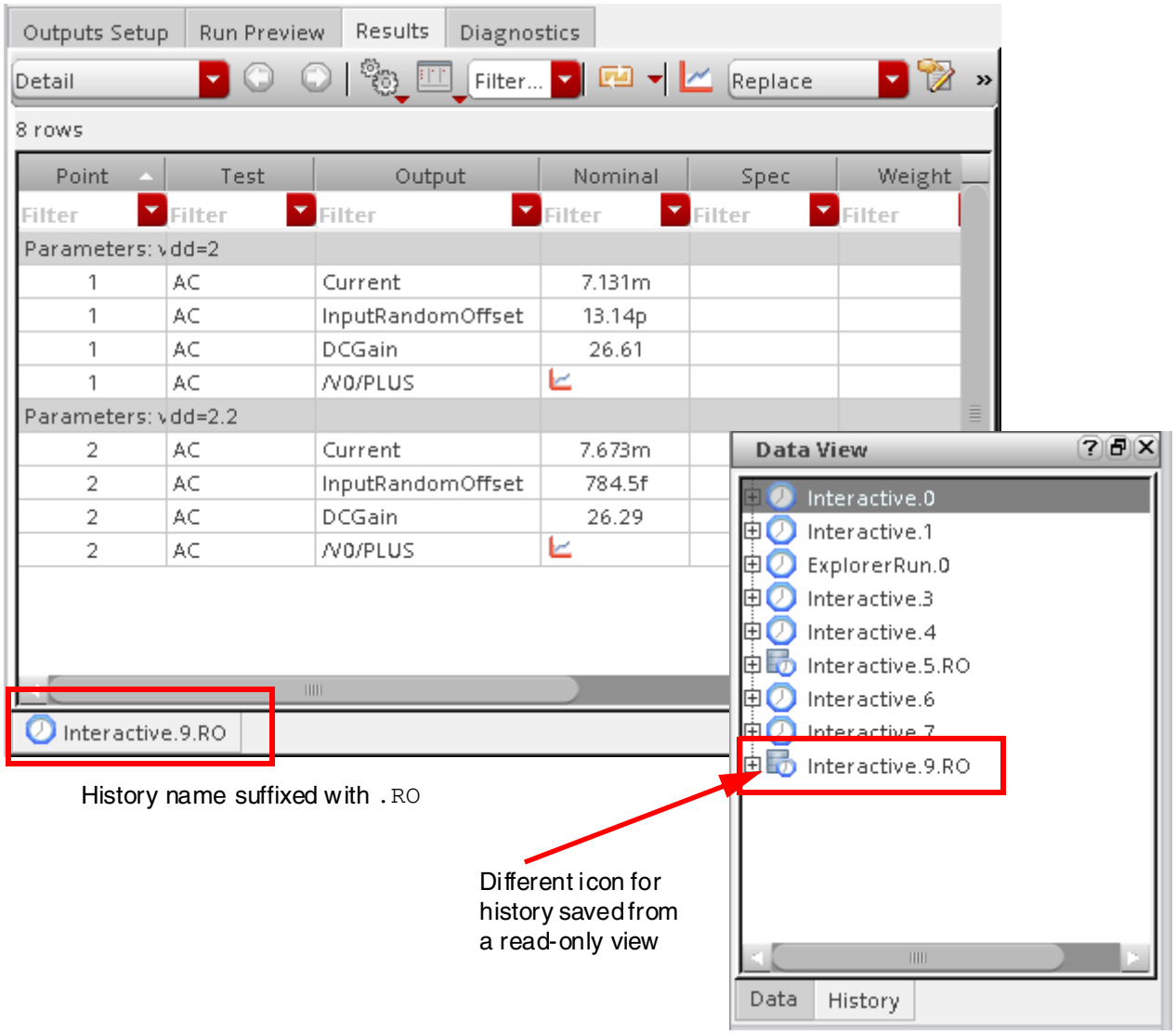
- Multiple instances of a maestro cellview can be opened simultaneously in read-only mode within a Virtuoso session.
- Across multiple Virtuoso sessions, a maestro cellview can be opened simultaneously in the read-only mode.
- For editing, a maestro cellview can be opened in edit mode only in one session. If a cellview is already being edited and you try to open it again in edit mode in the same or different ADE Explorer or ADE Assembler session, an error message is displayed.

© 2023 Cadence Design Systems, Inc.

Running Simulations from Read-Only maestro Views

When you open a cellview in read-only mode, you can make changes in the setup and run simulations. The simulation results use the updated setup information, but you cannot save the changes in the original maestro cellview. If it is important to save the changes, you need to use another cellview name or make the current cellview editable before saving the changes.

When a simulation is run for a read-only cellview in ADE Assembler, a history checkpoint is saved with a `.RO` suffix in the name. Also, on the *History* tab, a different icon is used for such a history, as shown below.



© 2023 Cadence Design Systems, Inc.

Use the *View Results* or *Load Setup to Active history* commands to view the results or the setup used for the history with the `.RO` suffix.

To save the setup changes from a `.RO` history into a maestro view, open the view in the edit mode, load the read-only history setup to active, and save the cellview. You can also convert a read-only history into an editable history by using the *Move to View History* command from the context-sensitive menu of a history item. For more details, refer to Saving a Read-Only History Inside the maestro View.

Important Points to Note

- When a simulation is run for a read-only cellview in ADE Explorer, the results are saved in the `ExplorerRORun.0.RO` file at the following location:

  `<simulationDir>/<lib>/<cell>/<view>/results/maestro/ExplorerRORun.0.RO/`

  The same `ExplorerRORun.0.RO` file is overwritten for subsequent runs from this or another read-only instances of this cellview in ADE Explorer.

  For more details, refer to Saving Results in Explorer in *Virtuoso ADE Explorer User Guide*.

- If multiple read-only instances of a cellview are open, you can make changes in their setup and run simulations independently. However, when you run a simulation, during the time of history creation, the cellview is temporarily locked and at that time, other instances cannot create a history.

- If a read-only instance of a cellview is trying to run a simulation while the setup is locked by another instance, by default, ADE Assembler waits for `300` seconds to acquire the lock. If the previous lock is released during this waiting time, the lock is obtained for the next task to be done. Otherwise, the task is not completed. You can change this default waiting time to acquire a lock on the read-only cellview by using the waitToAcquireROLock environment variable.

- Similarly, the setup database of the read-only cellview is temporarily locked when you rename or delete a `.RO` history. If the setup database is already locked and you try to rename or delete a `.RO` history, ADE Assembler shows an error reporting that the database is locked and unavailable for editing.

- If multiple instances of a maestro cellview are opened in read-only mode, a `.RO` history saved from one instance is visible in all other read-only or editable instances for that cellview. Similarly, when you rename or delete a `.RO` history from one instance, the change is reflected in all the instances.

© 2023 Cadence Design Systems, Inc.

- If the view that you opened in read-only mode has the *Overwrite History* field on the Save Options form set to an existing history name without the `.RO` suffix, it is ignored. Instead, the field is set to `Next History Run` and a new history is saved with the name as `<history-prefix>.RO`. If you want to overwrite the results to a specific read-only history, you can set this field to the name of a read-only history available in the drop-down list.

The following sections provide more details about the histories saved from a read-only maestro view:

- Location of Simulation Results and Results Database Information for Read-Only maestro Views
- Saving a Read-Only History Inside the maestro View
- Number of History Entries Saved for Read Only maestro Views

Important

You can disable running of simulations in a read-only maestro view using the `adexl.gui disableRunInReadOnly` environment variable.

Location of Simulation Results and Results Database Information for Read-Only maestro Views

By default, ADE Assembler writes the simulation results and setup information for a checkpoint run from a read-only maestro view to the *libraryName*/*cellName*/viewName/results/data/<*history_item*> path in the location specified using the `asimenv.startup projectDir` environment variable. The default setting for this environment variable is `$HOME/simulation`.

The results database information is also saved to `libraryName/cellName/viewName/results/data` in the same location specified using the `asimenv.startup projectDir` environment variable.

Important

Since ADE Assembler needs to write data while in the path set by `asimenv.startup projectDir`, it is important that you have write permission for this location.

You can specify a different location for storing simulation results using the procedure described in Specifying Options for Saving Simulation Results.

© 2023 Cadence Design Systems, Inc.

Saving a Read-Only History Inside the maestro View

By default, the histories of simulations run from a read-only cellview are saved. If you do not want to save the histories, set the `readOnlyHistorySupport` environment variable to `nil`.

```
envSetVal("adexl.simulation" "readOnlyHistorySupport" 'boolean nil)
```

To save the history as a part of the maestro view, do this:

1. Open the maestro cellview in the edit mode.
2. Right-click the history name on the *History* tab of the Data View pane and choose *Move to Cellview (Make Writable)*.

   The setup information and history results are moved from the project directory to the location where the maestro view results are saved, that is, to *libraryName*/*cellName*/*viewName*/results/maestro/<*history_item*>.

You can revert this change by using the *Move to Project Directory (Make Read Only) History* command in the context-sensitive menu for a history on the *History* tab.

Number of History Entries Saved for Read Only maestro Views

You can set the maximum number of histories to be saved in read-only mode by setting the `saveLastNROHistoryEntries` environment variable. The default value of this variable is `10`.

```
envSetVal("adexl.simulation" "saveLastNROHistoryEntries" 'int 10)
```

Saving Setup Changes in Read-Only maestro Views

If you make any changes to the setup in a read-only maestro view, the Save As form appears when you save or close the maestro view. You can save the changes to a new maestro view in a location where you have write permissions.

Alternatively, you can perform the following steps to save the setup changes without creating a new cellview:

1. Make changes in the setup of a read-only cellview.
2. Run a simulation. A history is saved with the `.RO` suffix.
3. Open the cellview in edit mode.
4. Right-click the `.RO` history and choose the *Load Setup to Active* command to load the setup from the read-only history.

© 2023 Cadence Design Systems, Inc.

5. Save the setup. The changes are saved in the original maestro cellview.

For information about using the Save As form, see Saving the Simulation Setup.

© 2023 Cadence Design Systems, Inc.

Importing the Simulation Setup

You can import the complete simulation setup or only specific simulation settings

Tip

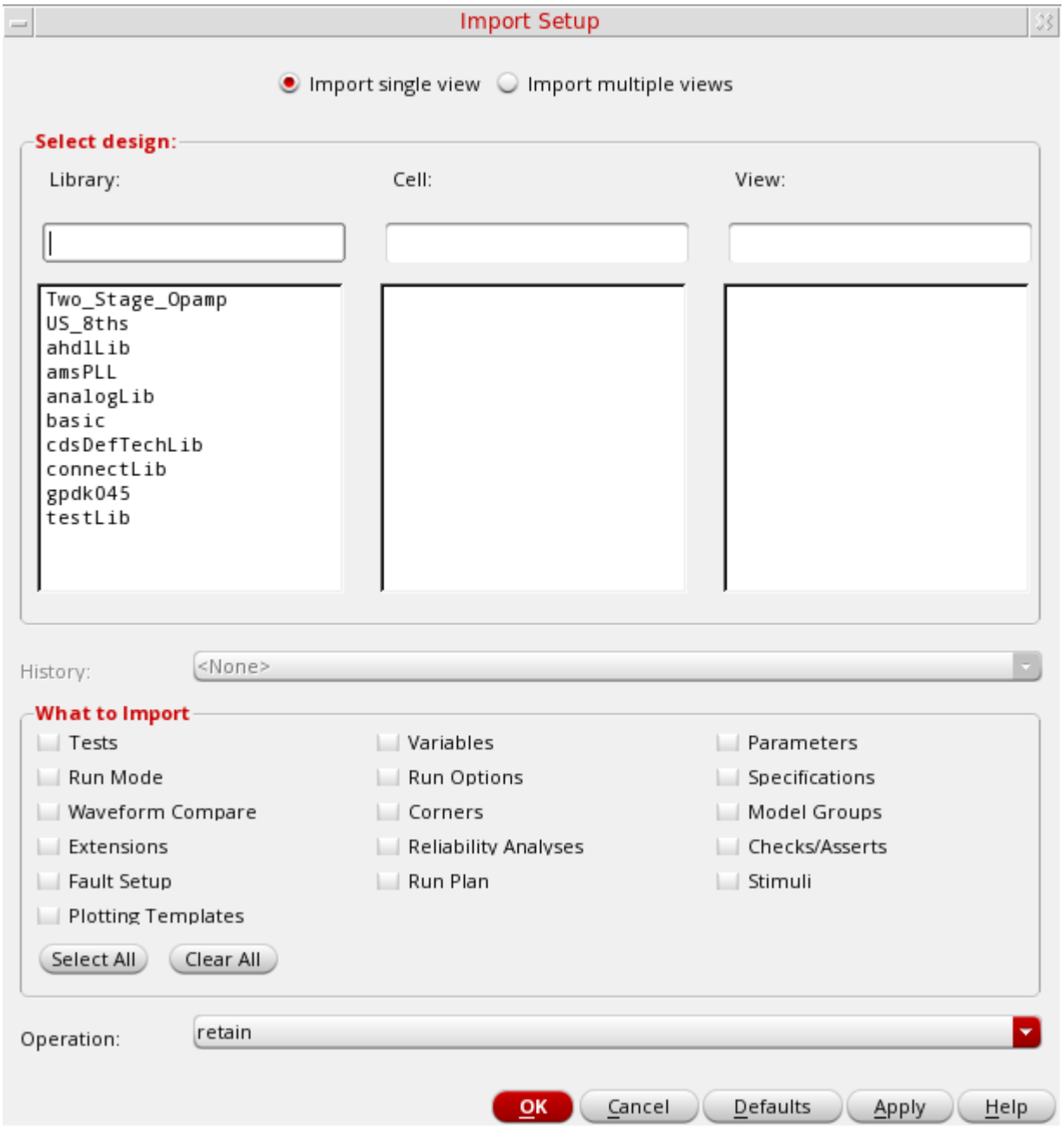
In a SKILL script, you can use thes `maeConvertAndCombineMultiADELToAssembler` function to import setup.

To import the simulation setup from one or more existing adexl or maestro cellviews to the current maestro view, do the following:

1. Choose *File – Import*.

© 2023 Cadence Design Systems, Inc.

The Import Setup form appears.



Select appropriate options on this form to specify the names of cellviews and the settings to be imported.

2. By default, the *Import single view* option is selected. This indicates that setup can be imported from a single cellview selected on the form. If you want to import details from multiple cellviews, select the *Import multiple views* option.

© 2023 Cadence Design Systems, Inc.

3. In the *Select design* section, select the name of a library and cell that contain the cellviews from which you need to import setup.
4. Select the names of one or more views in the *View* list. These are the cellviews from which you need to import the setup details.

   Note: You can import setup details from one or more ADE L, ADE XL, or ADE Assembler cellviews.
5. In the *History* drop-down list, select the history item from which you want to import the setup. By default, the components from the active simulation setup of the selected view are imported.
6. In the *What to Import* section, select the check box next to the settings you want to import.

   Note: Only the check boxes corresponding to the settings available in the setup to be imported are enabled. If any check box is not enabled, it implies that the corresponding setting is not available in the selected history or active setup of the view.

   By default, all the enabled settings are selected for import. You can click *Clear All* to clear all the selections or click *Select All* to select all the enabled check boxes for import.

   The following table describes the use of all the available check boxes in this section:

Select	To
Tests	Import the test definitions in the Data View pane. The simulation state of each test is also saved in the subdirectories.
Variables	Import the global variable definitions, including their enabled status and values, in the Data View pane.
Parameters	Import the definitions of parameters, including their enabled status and values, in the Data View pane.
Run Mode	Import the active run mode and set it on the Run toolbar.
Run Options	Import the following settings: ■ Options in the Run Options form. ■ Simulation options for all the customized run modes. For example, if you changed the default simulation options of the Monte Carlo, Global Optimization, and Local Optimization run modes before saving the state, the tool will import the settings for all the three run modes.

© 2023 Cadence Design Systems, Inc.

Specifications	Import the specifications defined in the Outputs pane. This includes the details of the specification type and their target values.
Corners	Import the corner definitions defined in the Corners Setup form.
Model Groups	Import the model groups defined in the Corners Setup form.
Extensions	Import the parasitic details in the Parasitic Setup form.
Reliability Analysis	Import the settings for reliability analysis.
Checks/Asserts	Import the settings for Checks/Asserts analysis.
Fault Setup	Import the settings for the fault setup assistant.
Run Plan	Import the run plan setup.
Stimuli	Import the setup for the Stimuli Assignment form for the tests that match
Plotting Templates	Import plotting templates

7. In the *Operation* drop-down list, select the import mode.

The following table describes the use of each import mode:

Select	To
retain	Retain the current setup information and append other setup information for each setting selected in the *What to Import* group box. For more information, see How the Simulation Setup is Updated When You Load Setup States on page 544.
merge	Overwrite the current setup information for items that have the same name and append other setup information for each setting selected in the *What to Import* group box. For more information, see How the Simulation Setup is Updated When You Load Setup States on page 544.
overwrite	Overwrite the current setup information for each setting selected in the *What to Import* group box. For more information, see How the Simulation Setup is Updated When You Load Setup States on page 544.

© 2023 Cadence Design Systems, Inc.

8. Click *OK* to import the setup into the current maestro view.

Working with Setup States

ADE Assembler supports different run modes like Single Run, Sweeps and Corners, Monte Carlo Sampling, Local Optimization, Global Optimization, and so on. For more information about run modes, see Specifying the Run Mode on page 83.

For each run mode, you may need a different simulation setup. For example, you may need different parameter specifications for running Monte Carlo analysis, optimization and sensitivity analysis. Similarly, you may need a different set of global variable values for running local and global optimization. Even for the same run mode, you may need a different simulation setup. For example, you may want to run Single Run, Sweeps and Corners with a different set of global variables.

ADE Assembler allows you to create setup states that contain all or part of the simulation setup. You can later restore the simulation setup from the setup state by loading all or part of the settings in the setup state. This allows you to avoid modifying the simulation setup every time you need to run simulation using a different setup. The following examples describe the use of setup states:

Example 12-1 Different Setup States for Different Run Modes

To have a different setup for Single Run, Sweeps and Corners runs and Monte Carlo analysis, do the following:

1. Define the settings required for Single Run, Sweeps and Corners runs and save the settings in a setup state named `SweepsCornersSetup`.

   For more information about creating a setup state, see Creating or Updating a Setup State on page 538.

2. Modify the settings as required for Monte Carlo analysis, and save the settings in a setup state named `MonteCarloSetup`.

3. Do the following:

   - Before you run Single Run, Sweeps and Corners, load the `SweepsCornersSetup` setup state using the *Overwrite* option.

     For more information about loading a setup state, see Loading a Setup State on page 540.

   - Before you run Monte Carlo analysis, load the `MonteCarloSetup` setup state using the *Overwrite* option.

© 2023 Cadence Design Systems, Inc.

Example 12-2 Different Setup States for Different Set of Global Variables

To run Single Run, Sweeps and Corners with a different set of global variables, do the following:

1. Create the first set of global variables and save the variables in a setup state named `SetupGlobalVars_1`.

   For more information about creating global variables, see Creating a Global Variable on page 250. For more information about creating a setup state, see Creating or Updating a Setup State on page 538.

2. Delete the first set of global variables.

3. Create the second set of global variables and save the variables in a setup state named `SetupGlobalVars_2`.

4. Do one of following before you run Single Run, Sweeps and Corners:

   - To run Single Run, Sweeps and Corners using the global variables in the setup state named `SetupGlobalVars_1`, load the `SetupGlobalVars_1` setup state using the *Overwrite* option.

     For more information about loading a setup state, see Loading a Setup State on page 540.

   - To run Single Run, Sweeps and Corners using the global variables in the setup state named `SetupGlobalVars_2`, load the `SetupGlobalVars_2` setup state using the *Overwrite* option.

See the following topics for more information:

- Creating or Updating a Setup State on page 538
- Loading a Setup State on page 540
- Renaming a Setup State on page 542
- Deleting a Setup State on page 542
- Running a Simulation Using a Setup State on page 547

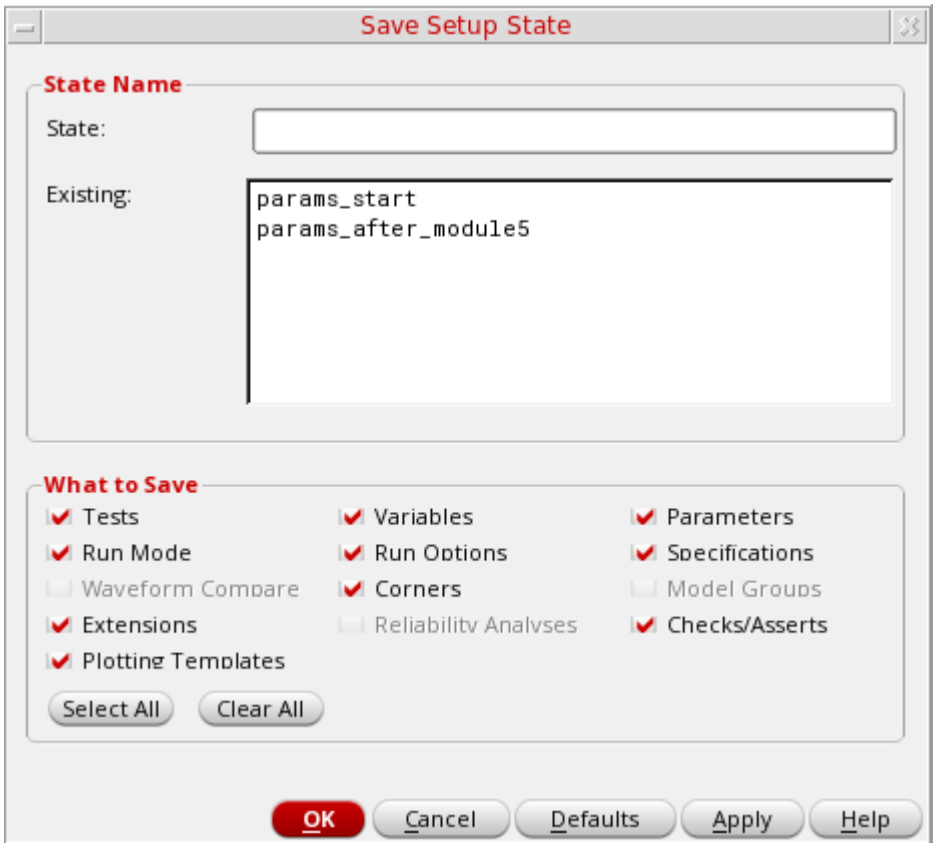
© 2023 Cadence Design Systems, Inc.

Creating or Updating a Setup State

To create a new setup state, or update an existing setup state, do the following:

1. Choose *File – Save Setup State*.

   The Save Setup State form appears.



2. Do one of the following:

   - To create a new setup state, enter the name of the setup state in the *State* field.
   - To update an existing setup state, select an existing state in the *Existing* list.

3. In the *What to Save* group box, select the check box next to the settings you want to save.

   By default, all the settings existing in the maestro view are selected for saving. You can click *Select All* to select all the settings for saving, or click *Clear All* to clear all the selections.

© 2023 Cadence Design Systems, Inc.

Select	To
Tests	Save the test definitions in the Data View pane. The simulation state of each test is also saved in the subdirectories.
Variables	Save the global variable definitions, including their enabled status and values, in the Data View pane.
Parameters	Save the definitions of parameters, including their enabled status and values, in the Data View pane.
Run Mode	Save the active run mode that is currently selected on the Run toolbar.
Run Options	Save the following settings: ■ Options from the Run Options form. ■ Simulation options for all the customized run modes. For example, if you have changed the default settings of the Monte Carlo and Local Optimization run modes, but the currently active run mode is Global Optimization, the tool will save the settings for all three run modes.
Specifications	Save the specifications defined in the Outputs pane. This includes the details of the specification type and their target values.
Waveform Compare	Save the wave specs added in the Outputs pane.
Corners	Save the corner definitions defined in the Corners Setup form.
Model Groups	Save the model groups defined in the Corners Setup form.
Extensions	Save the parasitic details set by using the Parasitic Setup form.
Reliability Analysis	Save the details set by using the Reliability form.
Checks/Asserts	Save the enabled or disabled status of the *Checks/Asserts* check box specified in the Data View assistant pane. It also saves the enabled or disabled status of this feature for each test. **Note:** Outputs of type `violation` are saved in test state.

4. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Important

By default, the maestro state is saved in XML file format, which saves fewer files as compared to the number of files saved in an ADE state.

Loading a Setup State

To load a setup state, do the following:

1. Choose *File – Load Setup State*.

   The Load Setup State form appears.

2. In the *State Name* group box, select the name of the state you want to load.
3. In the *What to Load* group box, select the check box next to the settings you want to restore to the current simulation setup.

© 2023 Cadence Design Systems, Inc.

By default, all the settings existing in the setup state are selected for restore. You can click *Select All* to select all the settings for restoring, or click *Clear All* to clear all the selections.

Select	To
Tests	Restore the test definitions in the Data View pane. The simulation state of each test is also saved in the subdirectories.
Variables	Restore the global variable definitions, including their enabled status and values, in the Data View pane.
Parameters	Restore the deifinitions of parameters, including their enabled status and values, in the Data View pane.
Run Mode	Restore the active run mode and set it on the Run toolbar.
Run Options	Restore the following settings: ■ Options in the Run Options form. ■ Simulation options for all the customized run modes. For example, if you changed the default simulation options of the Monte Carlo, Global Optimization, and Local Optimization run modes before saving the state, the tool will restore the settings for all the three run modes.
Specifications	Restore the specifications defined in the Outputs pane. This includes the details of the specification type and their target values.
Waveform Compare	Restore the wave specs to the outputs in the Outputs pane.
Corners	Restore the corner definitions and their enable status defined in the Corners Setup form.
Model Groups	Restore the model groups defined in the Corners Setup form.
Extensions	Restore the parasitic details set by using the Parasitic Setup form.
Reliability Analysis	Restore the details set by using the Reliability form.
Checks/Asserts	Restore the enabled or disabled status of the *Checks/Asserts* check box specified in the Data View assistant pane, and the enabled or disabled status of this feature for each test. **Note:** Outputs of type `violation` are restored from test state.

© 2023 Cadence Design Systems, Inc.

4. In the *Operation* drop-down list, select the restore mode.

Select	To
retain	Retain the current setup information and append other setup information for each setting selected in the *What to Load* group box. For more information, see How the Simulation Setup is Updated When You Load Setup States on page 544.
merge	Overwrite the current setup information for items that have the same name and append other setup information for each setting selected in the *What to Load* group box. For more information, see How the Simulation Setup is Updated When You Load Setup States on page 544.
overwrite	Overwrite the current setup information for each setting selected in the *What to Load* group box. For more information, see How the Simulation Setup is Updated When You Load Setup States on page 544.

5. Click *OK* to load the selected setup state.

Renaming a Setup State

To rename a setup state, do the following:

1. Expand the *Setup States* tree in the Data View assistant.
2. Double-click the setup state name that you want to change.

   The setup state name is enabled for change.
3. Type a new name and press `Enter`.

   The setup state is renamed.

Deleting a Setup State

To delete a setup state, do the following:

1. Choose *File – Remove Setup State*.

© 2023 Cadence Design Systems, Inc.

The Remove Setup State form appears.

2. In the *State Name* group box, select the name of the setup state you want to delete.
3. Click *OK* to delete the setup state.

© 2023 Cadence Design Systems, Inc.

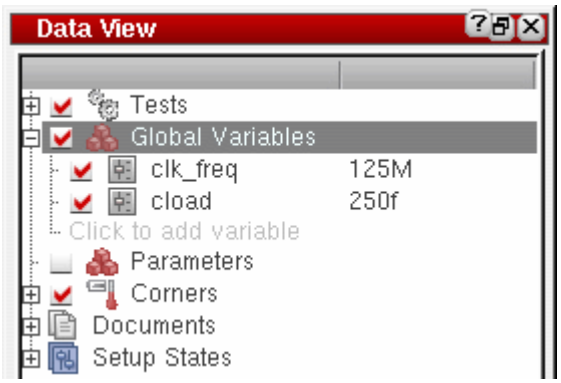
How the Simulation Setup is Updated When You Load Setup States

When you load a setup state, you can specify the type of settings to be loaded in the What to Load group box.

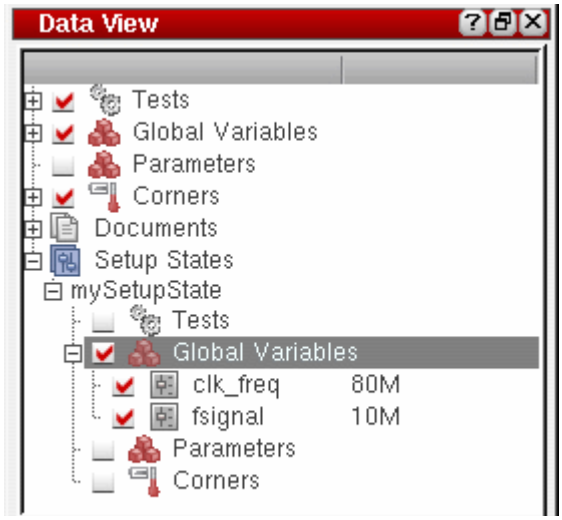
You can also choose the operating mode to be set while loading the setup state information. The operating mode describes how the current simulation setup is updated when you load a setup state in `retain`, `merge` or `overwrite` mode.

The following example shows how the different operating modes load the information:

The current simulation setup in a maestro cellview has the following global variables:



Choose to load variables from a setup state that has the following global variables:



See the following sections to understand how the current simulation setup is updated when you load the setup state in retain, merge or overwrite mode:

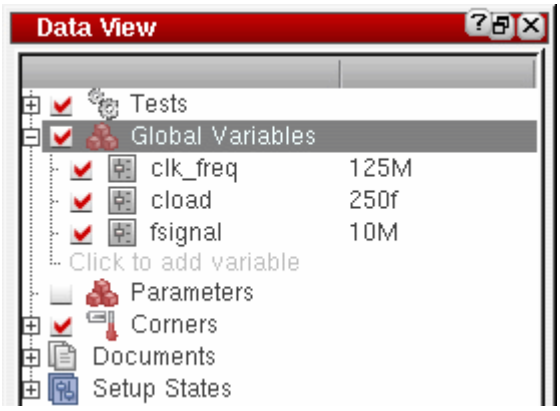
- Results of Loading a Setup State in Retain Mode on page 545

© 2023 Cadence Design Systems, Inc.

- Results of Loading a Setup State in Merge Mode on page 546
- Results of Loading a Setup State in Overwrite Mode on page 546

Results of Loading a Setup State in Retain Mode

The updated simulation setup after loading the setup state in *retain* mode will have the following global variables:



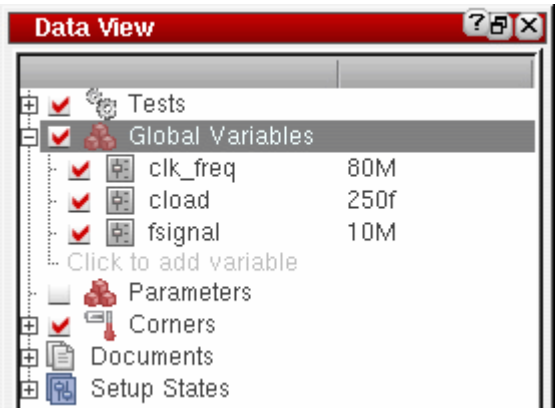
Note the following:

- The settings for the `clk_freq` variable are retained because the setup state has a variable with the same name.
- The settings for the `cload` variable are retained
- The `fsignal` variable is added because the current simulation setup does not have a variable with the same name

© 2023 Cadence Design Systems, Inc.

Results of Loading a Setup State in Merge Mode

The updated simulation setup after loading the setup state in *merge* mode will be:

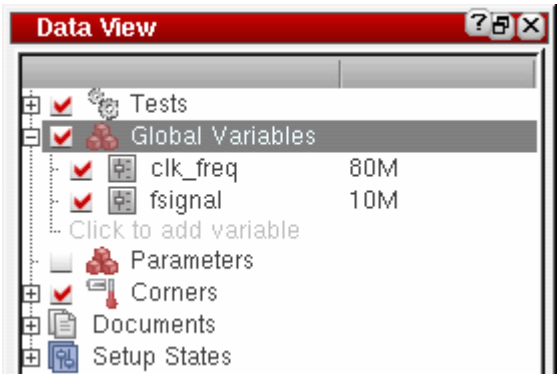


Note the following:

- The settings for the `clk_freq` variable are overwritten because the setup state has a variable with the same name.
- The settings for the `cload` variable are retained because the setup state does not have a variable with the same name.
- The `fsignal` variable is added because the current simulation setup does not have a variable with the same name.

Results of Loading a Setup State in Overwrite Mode

The updated simulation setup after loading the setup state in *overwrite* mode will be:



All the global variables in the current simulation setup are deleted and the global variables in the setup state are added when you load the setup state in *overwrite* mode.

© 2023 Cadence Design Systems, Inc.

Running a Simulation Using a Setup State

You can run a simulation based on the settings in a setup state.

To run a simulation using a setup state, do the following:

➤ Right-click the setup state in the expanded *Setup States* tree on the Data View pane and choose *Run*.

Note: The history item for the simulation run will have a name starting with the name of the setup state. For example, if the name of the setup state is `myCorners`, the history item will have the name `myCorners.`*`seqNum`*, where *`seqNum`* is `0` (zero) for the first history item, then 1+(the largest existing *`seqNum`* for that setup state). For more information about history items, see Chapter 20, "Working with History Checkpoints."

© 2023 Cadence Design Systems, Inc.

Creating a Plan Using Setup States

A plan is a sequence of steps required to complete a particular task. For example, characterizing a design might involve running several tests or sweeps. These tasks can be grouped together to form a characterization plan.

Tip

The Run Plan assistant in ADE Assembler provides a convenient way to set up runs, apply conditions between runs, and save regression scripts. Use that assistant to create run plans that involve a common set of tests and their setup. If your run plan needs to use a different set of tests or different setups of a particular test, create a plan using setup states. However, such plans will be run from OCEAN and will create OCEAN histories.

To create a plan by using setup states, you need a separate setup state for each task in the plan that you link, in the order they need to be run, to an OCEAN script file. When you run the OCEAN script file, ADE Assembler does the following for each setup state, in the order in which the setup states are linked to the OCEAN script file:

1. Loads the current simulation setup to memory.
2. Loads the setup state to update the simulation setup in memory.
3. Runs simulation using the updated simulation setup in memory.
4. Clears the simulation setup in memory.

For example, if you have two tasks in a plan, create two setup states—say, `myTask1` and `mytask2`—that contain the simulation settings required to run the simulations corresponding to each task in the plan. Then link the setup states to an OCEAN script file named `myPlan.ocn`, in the order in which they need to be run. When you run the `myPlan.ocn` OCEAN script file, ADE Assembler does the following:

1. Loads the current simulation setup to memory.
2. Loads the `myTask1` setup state to update the simulation setup in memory.
3. Runs simulations using the updated simulation setup in memory.
4. Clears the simulation setup in memory.
5. Loads the current simulation setup to memory.
6. Loads the `myTask2` setup state to update the simulation setup in memory.
7. Runs simulations using the updated simulation setup in memory.

© 2023 Cadence Design Systems, Inc.

8. Clears the simulation setup in memory.

To create a plan, do the following:

1. Create setup states that contain the simulation settings required to run the simulations corresponding to each task in the plan. For more information, see Creating or Updating a Setup State on page 538.
2. Right-click a setup state in the expanded *Setup States* tree on the Data View pane and choose *Link to OCEAN XL Script*.

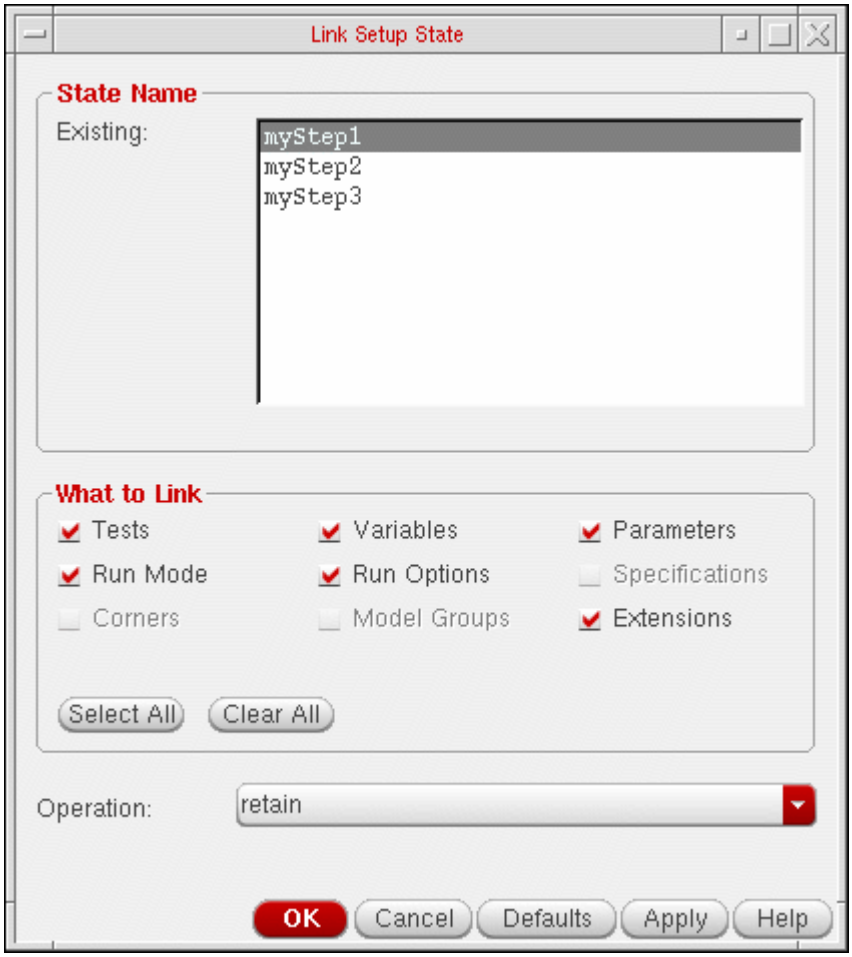
   The Choose an OCEAN XL File form appears.
3. Do one of the following:
   - To link a setup state to a new OCEAN script file, type the name of the OCEAN script file in the *File Name* field, and click *Open*.
   - To link a setup state to an existing OCEAN script file, select the OCEAN script file, and click *Open.* The following message appears:

     ```
     File <filename> already exists. How do you want to link to it.
     ```

     Do one of the following:
     - Click *Append* to add the OCEAN XL commands required to link the setup state in the end of the file.
     - Click *Overwrite* to delete the contents of the file and add only the OCEAN XL commands required to link the setup state.

© 2023 Cadence Design Systems, Inc.

The Link Setup State form appears.



4. In the *State Name* group box, select the name of the setup state you want to link to the OCEAN XL script file.
5. In the *What to Link* group box, select the check box next to the settings you want to be restored to the simulation setup in memory.

   By default, all the settings existing in the setup state are selected. You can click *Select All* to select all the settings, or click *Clear All* to clear all the selections.

Select	**To**
Tests	Restore tests
Run Mode	Restore run mode settings

© 2023 Cadence Design Systems, Inc.

Corners	Restore corners
Variables	Restore global variables
Run Options	Restore run options
Model Groups	Restore model groups
Parameters	Restore parameters
Specifications	Restore specifications
Extensions	Restore extensions

6. In the *Operation* drop-down list, select the restore mode.

Select	To
retain	Retain the setup information in memory and append other setup information for each setting selected in the *What to Link* group box. For more information, see How the Simulation Setup is Updated When You Load Setup States on page 544.
merge	Overwrite the setup information in memory for items that have the same name and append other setup information for each setting selected in the *What to Link* group box. For more information, see How the Simulation Setup is Updated When You Load Setup States on page 544.
overwrite	Overwrite the setup information in memory for each setting selected in the *What to Link* group box. For more information, see How the Simulation Setup is Updated When You Load Setup States on page 544.

7. Click *OK* to link the setup state to the OCEAN script file.

The OCEAN script file is opened in a text editor. The OCEAN script file contains the OCEAN XL commands required to required to link the setup state.

```
;----------------- Setup State "myStep1" ---------------
ocnSetXLMode(Assembler)
ocnxlTargetCellView( "workshopLib" "MSPS_DUT" "maestro" )
ocnxlLoadSetupState( "myStep1" 'retain ?tests t ?vars t ?parameters t
?currentMode t
    ?runOptions t ?extensions t ?specs nil ?corners nil
    ?modelGroups nil )
ocnxlRun()
```


© 2023 Cadence Design Systems, Inc.

```
ocnxlOutputSummary()
ocnxlEndXLMode(Assembler)
```

8. Repeat steps 2 to 7 to link more setup states to the OCEAN script file. The OCEAN XL commands required to required to link the setup states are appended in the order in which they are linked to the OCEAN script file.

The following example shows how three setup states named `myTask1`, `mytask2` and `myTask3` are linked to an OCEAN script file.

```
;----------------- Setup State "myStep1" ---------------
ocnSetXLMode()
ocnxlTargetCellView( "workshopLib" "MSPS_DUT" "maestro" )
ocnxlLoadSetupState( "myStep1" 'retain ?tests t ?vars t ?parameters t
?currentMode t
    ?runOptions t ?extensions t ?specs nil ?corners nil
    ?modelGroups nil )
ocnxlRun()
ocnxlOutputSummary()
ocnxlEndXLMode()
;----------------- Setup State "myStep2" ---------------
ocnSetXLMode()
ocnxlTargetCellView( "workshopLib" "MSPS_DUT" "maestro" )
ocnxlLoadSetupState( "myStep2" 'retain ?tests t ?vars t ?parameters t
?currentMode t
    ?runOptions t ?extensions t ?specs nil ?corners nil
    ?modelGroups nil )
ocnxlRun()
ocnxlOutputSummary()
ocnxlEndXLMode()
;----------------- Setup State "myStep3" ---------------
ocnSetXLMode()
ocnxlTargetCellView( "workshopLib" "MSPS_DUT" "maestro" )
ocnxlLoadSetupState( "myStep3" 'overwrite ?tests t ?vars t ?parameters t
?currentMode t
    ?runOptions t ?extensions t ?specs nil ?corners nil
    ?modelGroups nil )
ocnxlRun()
ocnxlOutputSummary()
ocnxlEndXLMode()
```

9. Load and run your OCEAN script file in the Command Interpreter Window (CIW).

For more information, see the OCEAN Reference.

Important

You must not load and run your OCEAN XL script in the CIW while ADE Assembler is still running for the same cellview.

When you run the OCEAN XL script, the program reports the following information:

- Sweep parameters and their values

© 2023 Cadence Design Systems, Inc.

- Number of tests, sweep points, and corners
- Points completed and job status information
- Results location to the output area of the CIW

For example:

```
1/1 completed.
*Info*   The result of this OCEAN XL run are saved in "Interactive.3" in library
"workshopLib", cell "MSPS_DUT", view "adexl".
```

The results location corresponds to the lib/cell/view specified in the `ocnxlTargetCellView` call, such as

```
ocnxlTargetCellView( "workshopLib" "MSPS_DUT" "maestro" )
```

See "OCEAN Commands in XL Mode" in the *OCEAN Reference* for information about OCEAN script commands for ADE Assembler.

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

13

Performing Circuit and Device Checks

ADE Assembler provides an interactive user interface where you can define circuit or device checks that are performed using the simulation results. The violations of these checks are highlighted on the `Checks/Asserts` view on the Results tab. You can view and analyze the violations by using the various features of this view, such as, filtering violations using the built-in or custom filters, finding the critical instances or nets on the schematic, or highlighting the violation results on the waveforms. Next, waive or resolve the violations as per your requirements, and rerun simulations to verify the results.

Important

The Monte Carlo Sampling, Improve Yield, and High Yield Estimation run modes do not consider circuit and device checks. Therefore, you need to disable *Checks/Asserts* on the Data View assistant pane before running simulations for these run modes.

This chapter describes how to use the tools and features in ADE Assembler to perform circuit and device checks, and to resolve violations:

- An Introduction to Checks and Asserts
- Adding Checks and Asserts
- Filtering Violations
- Running Simulations and Viewing Violations

© 2023 Cadence Design Systems, Inc.

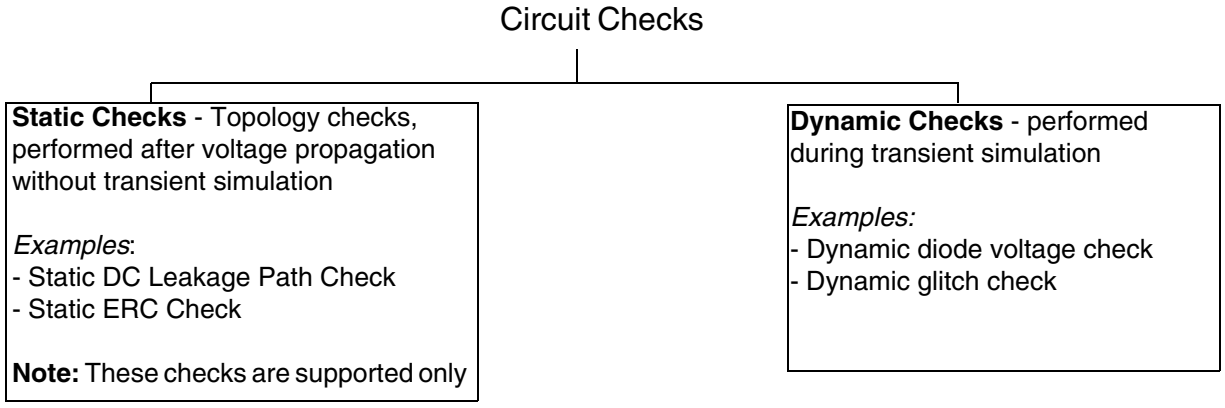
An Introduction to Checks and Asserts

This section provides an introduction to the types of checks supported by ADE Assembler. It also introduces the *Checks/Asserts* tree in the Data View assistant pane.

Checks

Checks (or circuit checks) enable you to analyze typical design problems, such as high impedance nodes, leakage paths between power supplies, setup and hold timing errors, power issues, connectivity problems, or extreme rise and fall times.

Circuit checks are of two types shown below.



For more details on the circuit checks that can be performed by the Spectre simulator, refer to Circuit Checks in *Spectre Circuit Simulator and Accelerated Parallel Simulator User Guide*.

Asserts

Asserts (or device checks) are used to perform checks on design parameters, node voltages, element currents, model parameters, operating point parameters, and expressions. In Spectre, the `assert` and `checklimit` statements are used to define and run device checks.

For details and examples on the device checks that can be performed by the Spectre simulator, refer to Device Checks in *Spectre Circuit Simulator and Accelerated Parallel Simulator User Guide*.

© 2023 Cadence Design Systems, Inc.

Dependency on Simulators

In ADE Assembler, you can perform all the circuit and device checks that are supported by the MMSIM 14.1 ISR11 or later releases. The checks should be defined using the required syntax and keywords, and can include wildcard characters, subcircuits, and instance scoping. Spectre writes the results of these checks into the SQLite database in XML format. ADE Assembler reads that data and displays it in tabular format on the *Results* tab.

Note: The static checks are supported only in APS and XPS modes of Spectre.

© 2023 Cadence Design Systems, Inc.

Setting up Checks and Asserts Analysis

To set up checks and asserts:

1. Select and expand *Checks/Asserts* in the Data View assistant of ADE Assembler.

   All tests defined in the current maestro cellview are displayed under this node. By default, the check boxes for all tests are deselected. Tests that are disabled in the *Tests* node are disabled in this tree as well.

   Note: If the `enableDochecklimitFromDataView` environment variable is set to `t`, the tool automatically updates the *dochecklimit* check box on the *Checks* tab of the Simulator Options form for the enabled tests under the *Checks/Asserts* node in the Data View assistant.

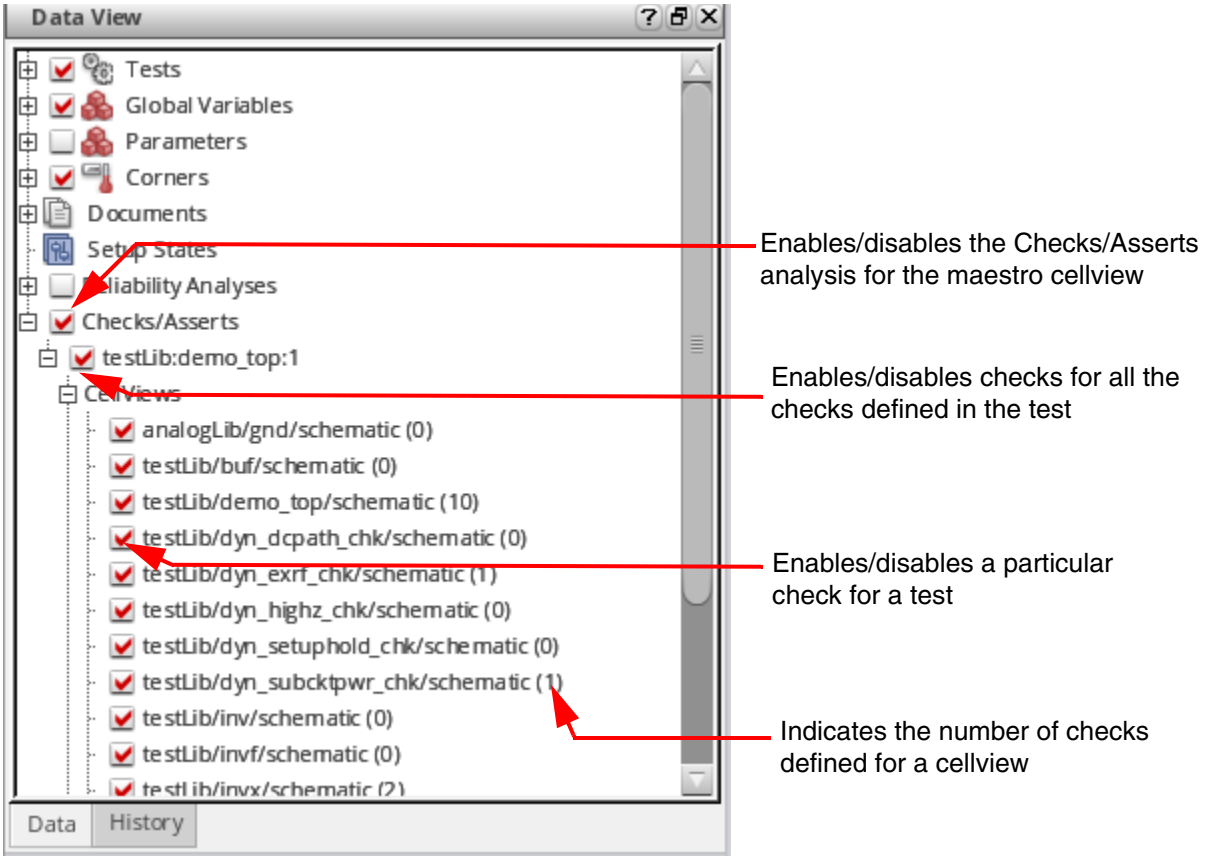
2. In the *Checks/Asserts* node, select the check boxes next to the tests for which you want to enable all the defined checks.

3. Expand the tree for a selected test.

4. Expand the *Cellviews* node under that test node to view the names of the schematic cellviews in that tests.

© 2023 Cadence Design Systems, Inc.

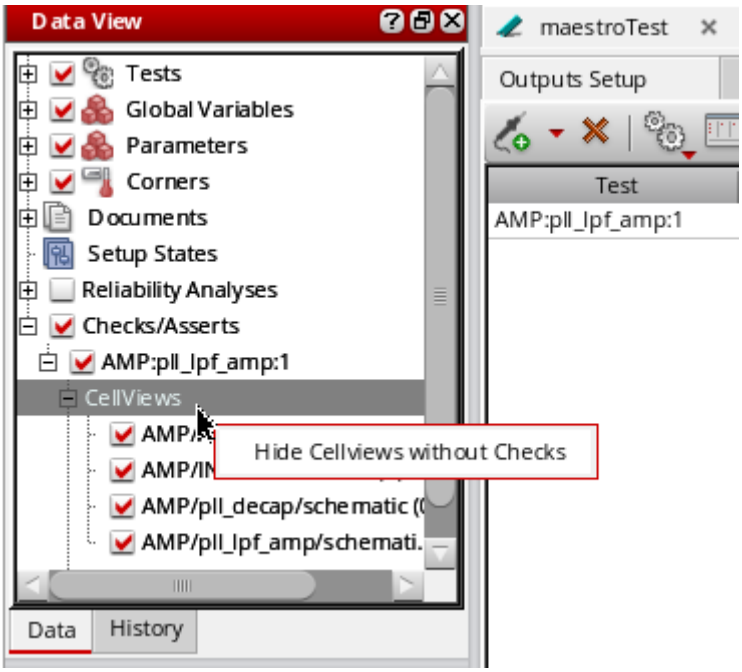
For each schematic cellview, the tool displays the total count of checks and asserts defined in that schematic.



5. If the number of cellviews in your design is large, use the commands in the context menu of the *Cellviews* node to show or hide only those cellviews that contain checks:
   - *Show Cellviews without Checks*

© 2023 Cadence Design Systems, Inc.

- *Hide Cellviews without Checks*



6. Select or clear the check boxes next to the schematic name for which you want to include the checks in the analysis.

You can define checks for cellviews in the Checks/Asserts assistant. For more details, see Adding Checks and Asserts.

Adding Checks and Asserts

Checks can be added in the following two ways:

- By using the Checks/Asserts assistant in the Virtuoso Schematic Editor window
- By copying the Checks/Asserts from One Cellview to Another
- By importing asserts from included text files
- By defining global device checks using the Device Check Options form

While you are adding checks and asserts in the Checks/Asserts assistant, you can also verify how they will be added to the netlist to be used in simulations. For details, see Viewing Netlists for Checks and Asserts.

© 2023 Cadence Design Systems, Inc.

Important

The *Checks/Asserts* check box in the Data View assistant can enable or disable only the checks that are created using the Checks/Asserts assistant in the Schematic Editor, not the checks imported from the text files included in the Assembler setup.

Adding Checks using the Checks/Asserts Assistant

To add checks for a test, perform the following steps:

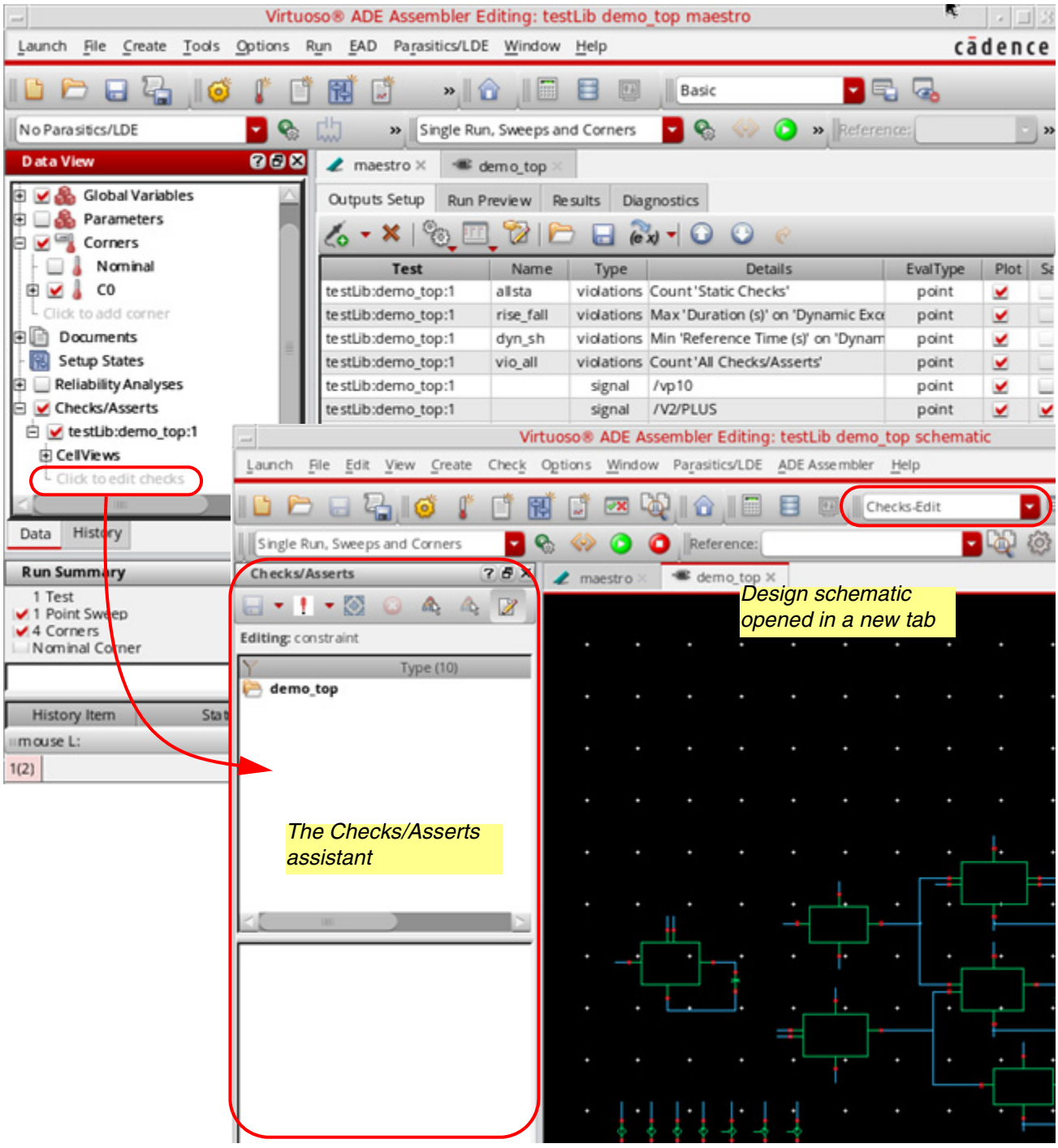
1. In the Data View pane, expand the *Checks/Asserts* tree.

   The names of all the tests defined in the current view are listed in this tree.

2. Expand the test tree for which you want to define checks.
3. Click *Click to edit checks*.

© 2023 Cadence Design Systems, Inc.

The schematic view of that cell is opened in a new tab.



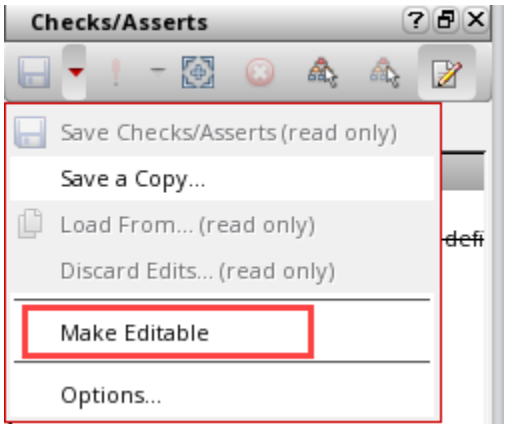
By default, the `Checks-Edit` workspace is opened. This workspace shows the *Checks/Asserts* assistant pane on the left.

© 2023 Cadence Design Systems, Inc.

To add new checks for this test, perform the following steps:

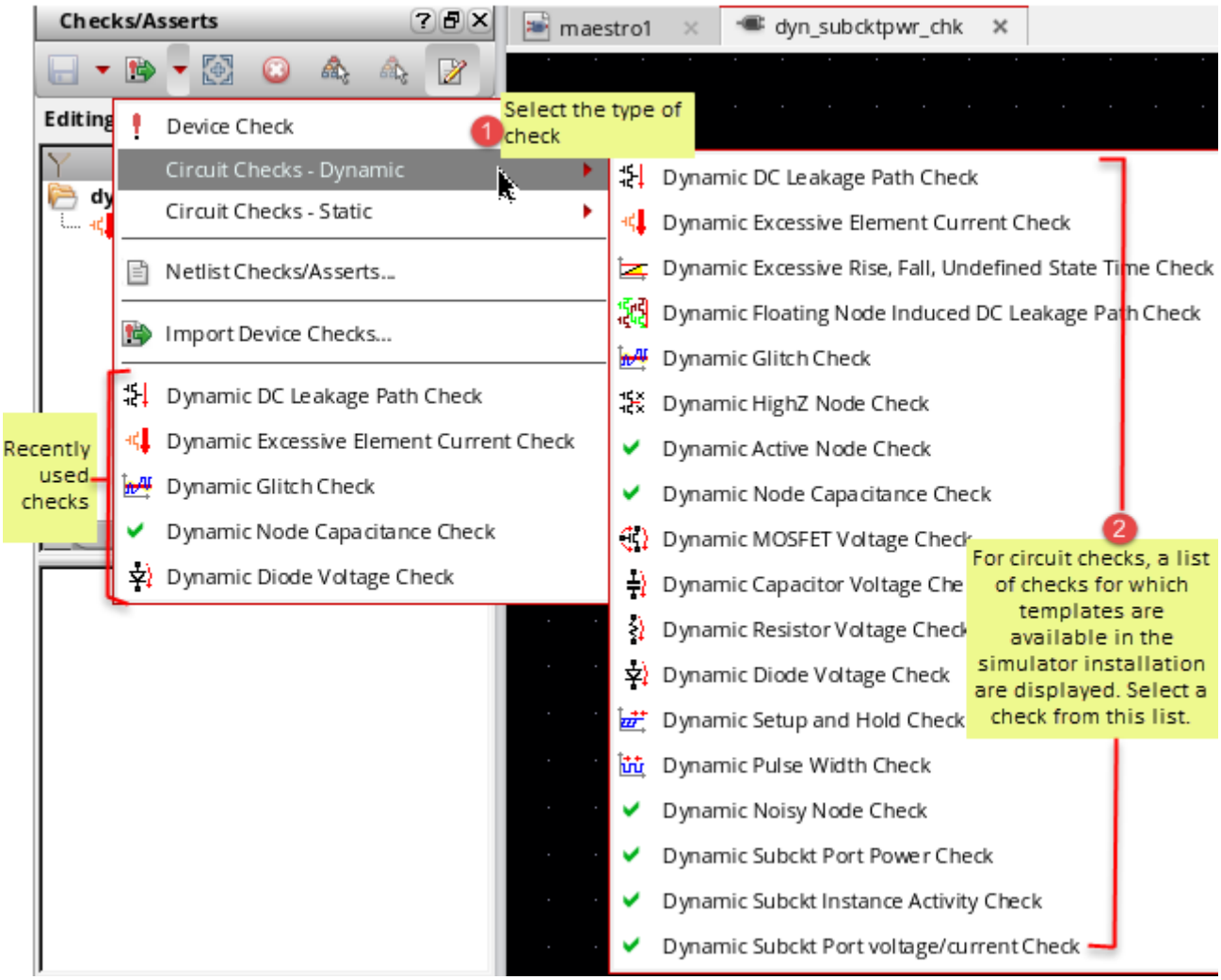
1. In the Checks/Asserts assistant, click the checks drop-down list, and choose the type of check you need to create. For example, *Circuit Checks – Dynamic*.

   Note: If the drop-down list is disabled, ensure that the constraints view is editable. If not, use the Make Editable command in the Checks/Asserts assistant, as shown below.



© 2023 Cadence Design Systems, Inc.

The list of dynamic checks for which templates are available in Spectre are shown in the submenu.



The check is added to the list of constraints in the *Checks/Asserts* assistant. Its corresponding parameters are displayed in the constraint parameters pane.

There is no list of checks for *Device Check*. For this type of checks, you can directly provide parameters in the Editor pane.

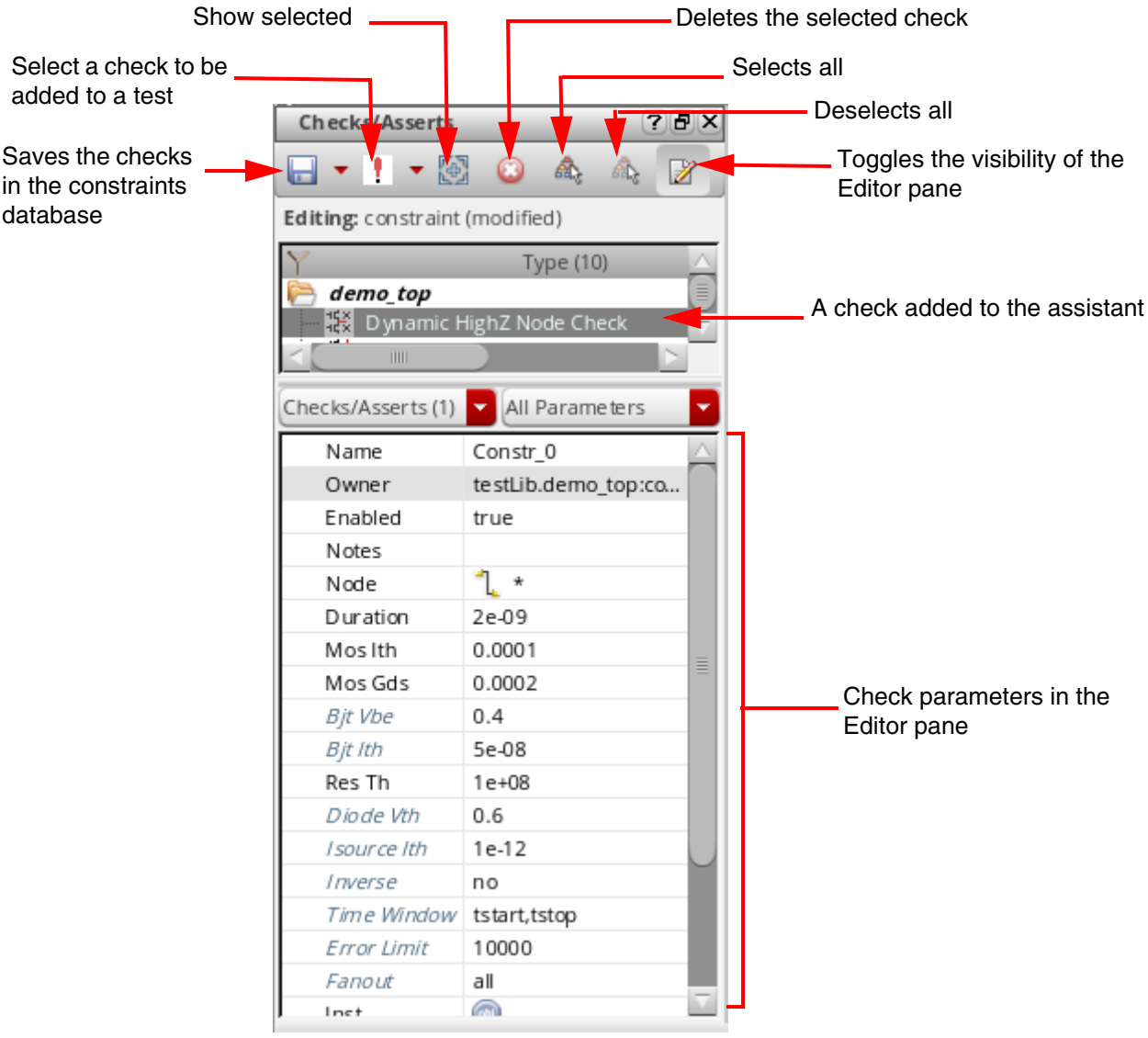
Note: Any custom or default system constraint created in the Constraint Manager assistant can be propagated automatically to the Checks/Asserts assistant. For information about how to enable the automatic propagation, see Propagating Constraints to the Checks/Asserts Assistant in *Virtuoso Unified Custom Constraints Configuration Guide*.

© 2023 Cadence Design Systems, Inc.

2. Move the pointer over a check name in the top area of the assistant to view its definition and parameters in a tooltip.

Tip

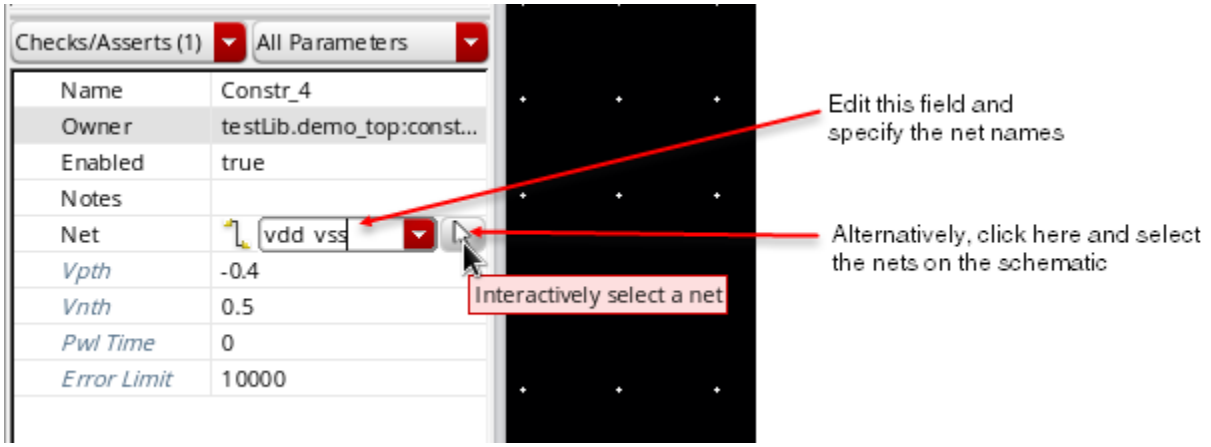
If the Editor pane is not visible, click *Checks/Asserts Editor* ( ) on the assistant toolbar.



3. Edit the check parameters as required. You can define the scope for a check by setting the *Node*, *Device*, and *Inst* fields.

© 2023 Cadence Design Systems, Inc.

Either type the names of nodes, devices, and instances, or directly select these from the schematic.



To apply a check for all the nodes or instances, enter `*` in the respective field.

If a field value for a parameter is unchanged, it will use the default values for that parameter as defined in the simulator file. The default values can be found in two ways:

- By using the `spectre -help <checkName>` command, as shown below.

  ```
  spectre -help assert
  spectre -help dyn_highz
  ```

- By moving the pointer over any check parameter in the Editor pane. The definition and default value for a parameter is displayed in the tooltip.

4. Repeat steps 1 to 3 to add more checks.
5. Click *Save* in the toolbar to save the checks.

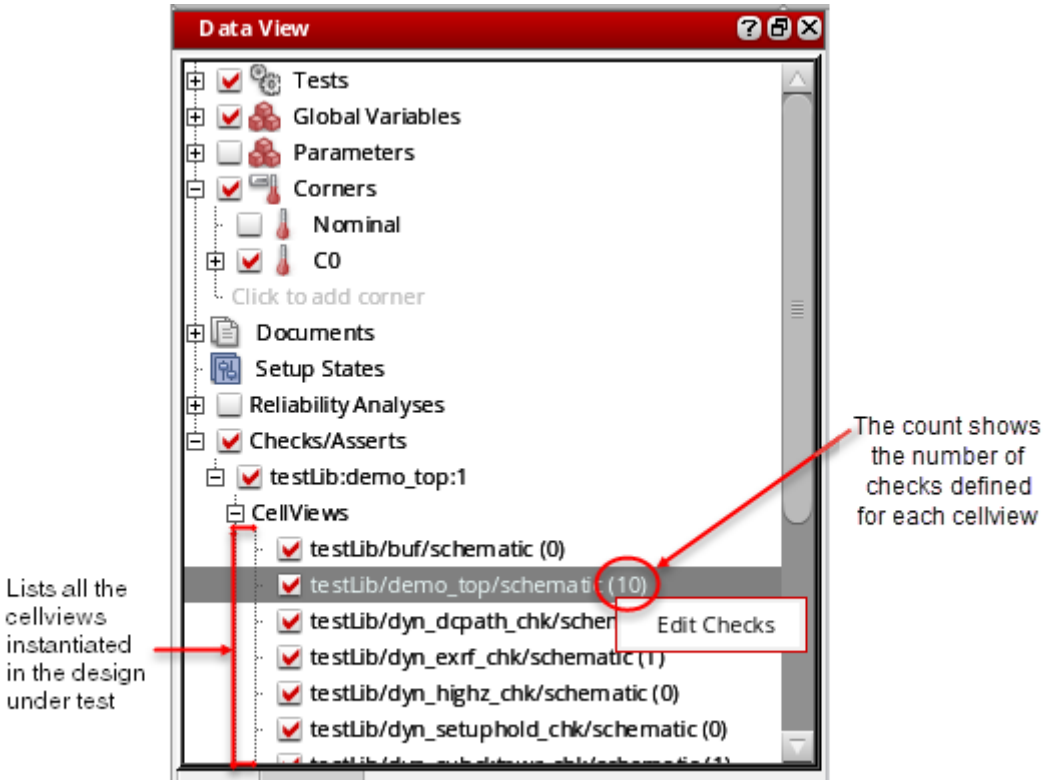
Important

It is required to save the checks to ensure they are netlisted correctly. If you run simulation without saving changes in the constraints view, ADE Assembler displays a message and prompts you to save changes.

© 2023 Cadence Design Systems, Inc.

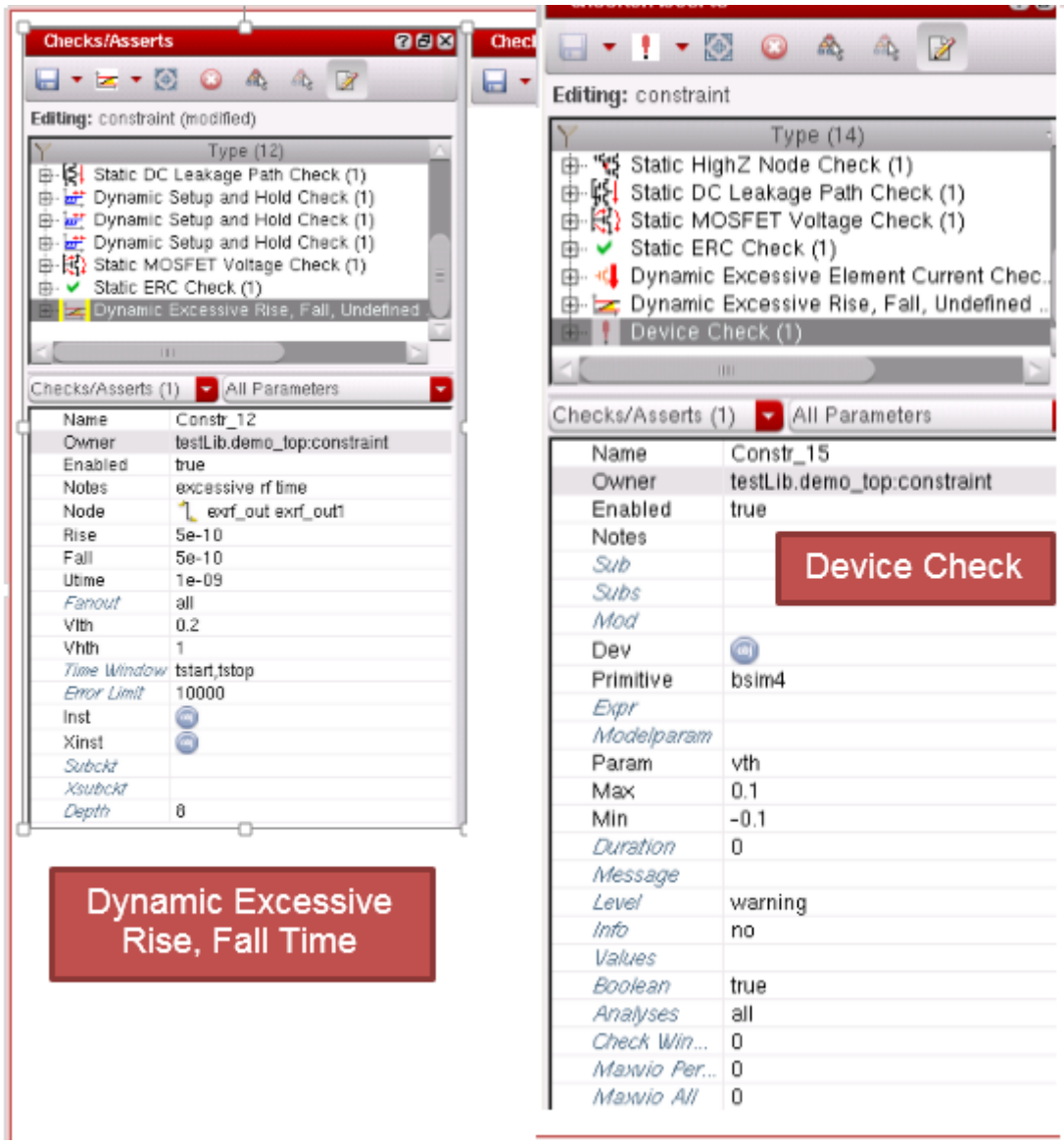
Tip

The *Checks/Asserts — <testName> — Cellviews* tree lists all the cellviews contained in the design under test. The numerical value next to each cellview shows the count of checks defined for each. To edit the checks for a cellview, right-click it and choose *Edit Checks*. The cellview is opened in the schematic view where you can add or edit checks in the Checks/Asserts assistant.



© 2023 Cadence Design Systems, Inc.

An example displaying a circuit check and a device check is shown below.



Important

The checks and asserts defined for a cell using the *Checks/Asserts* assistant are stored in a special partition of the constraint view for the cell. These checks will be netlisted using the subcircuit parameter to keep the scope of the check to the cellview in which it is created. These checks are included in the simulation whenever that cell is instantiated in a design.

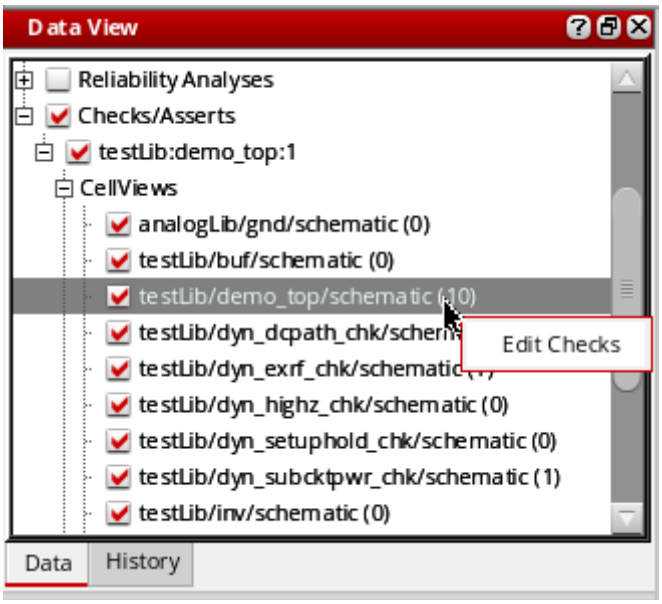
© 2023 Cadence Design Systems, Inc.

Copying Checks or Asserts

If the check or assert you need to create for a cellview already exists for another cellview instantiated in the design, you can copy that and customize it, if required.

To copy a check or assert, perform the following steps:

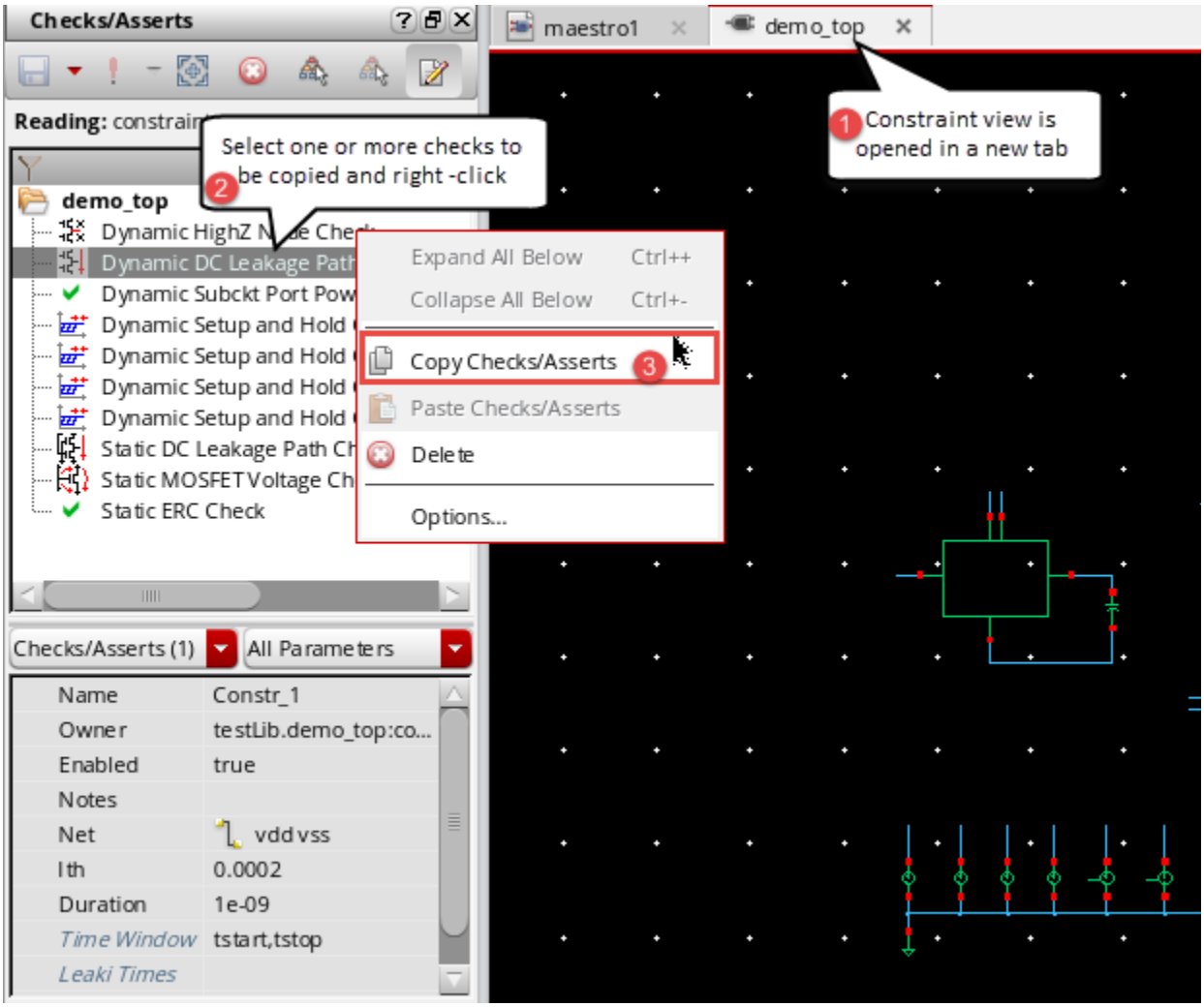
1. Right-click the source cellview from which you need to copy the check or assert, and choose *Edit Checks*.



The constraint view of the cellview is opened in a new tab and the checks/Asserts assistant is displayed.

© 2023 Cadence Design Systems, Inc.

2. Right-click one or more checks and asserts to be copied and choose *Copy Checks/Asserts* from the context-sensitive menu.

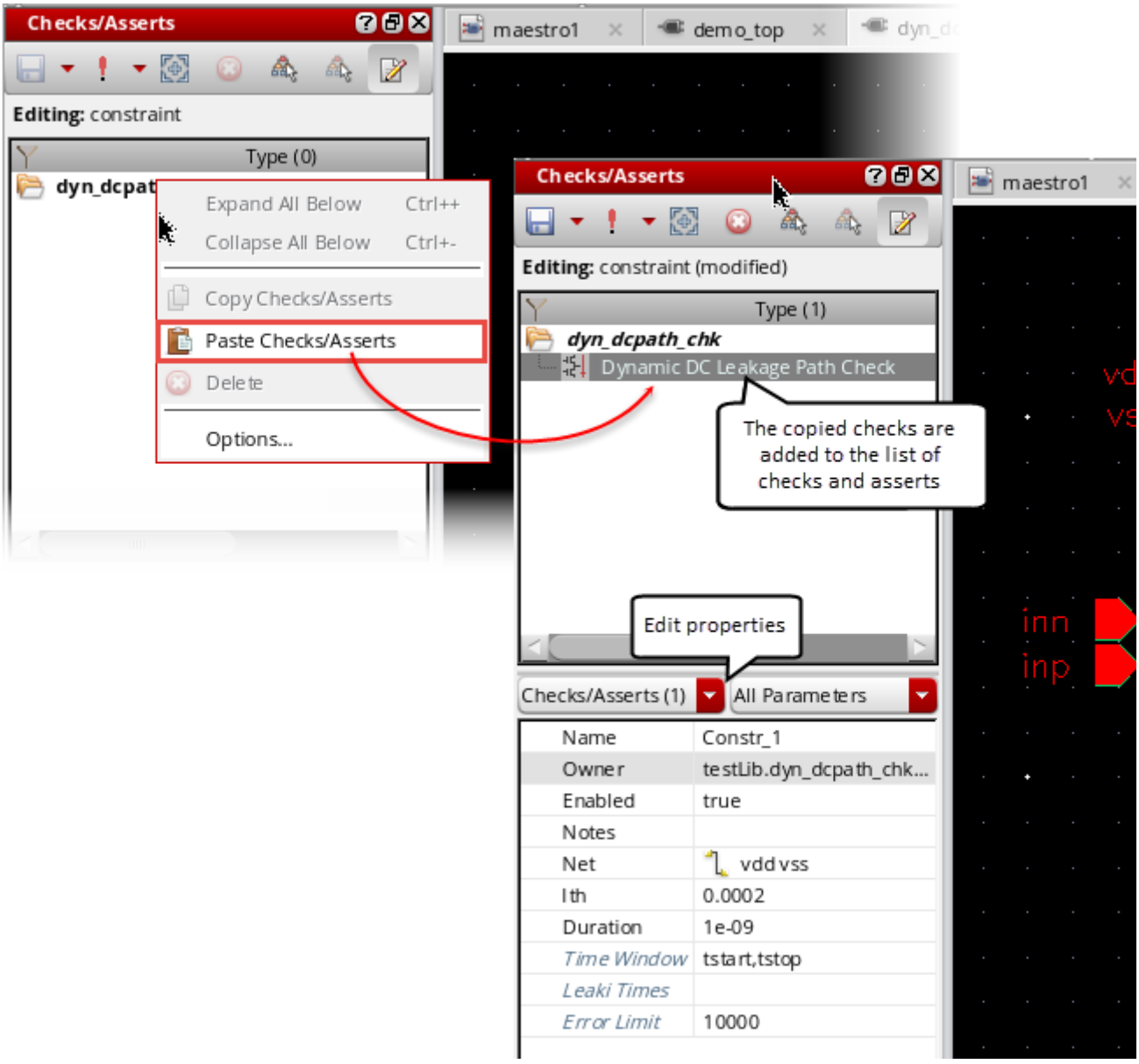


The copied check or assert is saved in the memory.

3. Reopen the maestro cellview tab.
4. Right-click the cellview to which you need to copy or reuse the copied check or assert, and choose *Edit Checks*.
5. The constraint view of the target cellview is also opened in a new tab and the Checks/Asserts assistant is displayed
6. Ensure that the constraint view is editable. If not, click *Save Checks/Asserts – Make Editable*.
7. Right-click anywhere in the checks/asserts pane and choose *Paste Checks/Asserts*.

© 2023 Cadence Design Systems, Inc.

The copied check or assert is added to the cellview. You can view and edit its parameters in the parameter editor pane of the Checks/Asserts assistant.



Importing Asserts from Text Files Included in the Setup

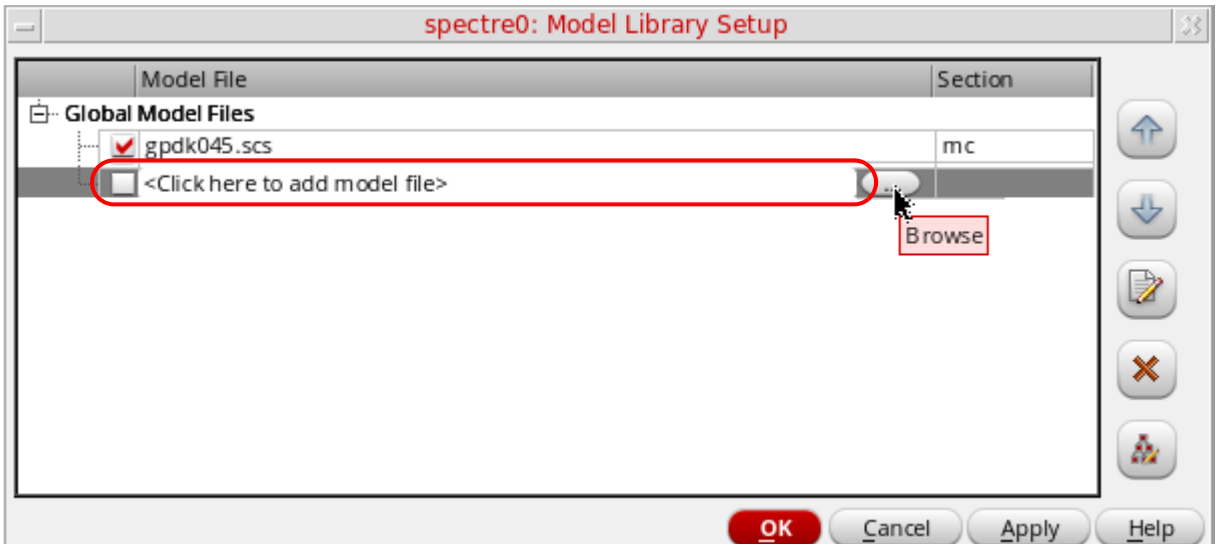
If you have defined device checks or Spectre asserts in model files, you can include those files for a test.

To import device checks, perform the following steps:

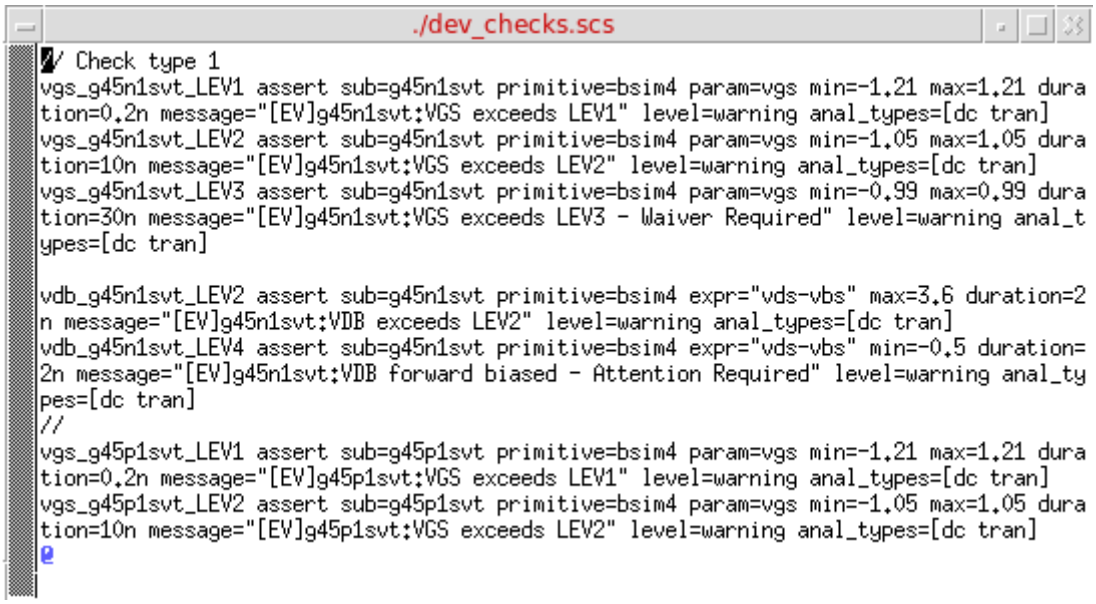
© 2023 Cadence Design Systems, Inc.

1. In the Data View pane, expand the *Tests* tree.
2. Right-click a test and choose *Model Libraries*.

   The Model Library Setup form is displayed.
3. Click `<Click here to add model file>` to view the *Browse* button.
4. Click *Browse* and select a model file that contains the definitions of device checks.



5. (Optional) If you want to view or edit the asserts, select the name of the model file and click *Edit/View Selected Files*.



© 2023 Cadence Design Systems, Inc.

6. Click *OK* to apply the changes and close the form.

The asserts included from the model file will now be included in the netlist and used in simulation runs.

Defining Global Device Checks

Using the Device Checking Options form, you can specify the global device check options, such as start and stop times for a transient analysis and default severity for various Spectre analyses.

To open the Device Checking Options form:

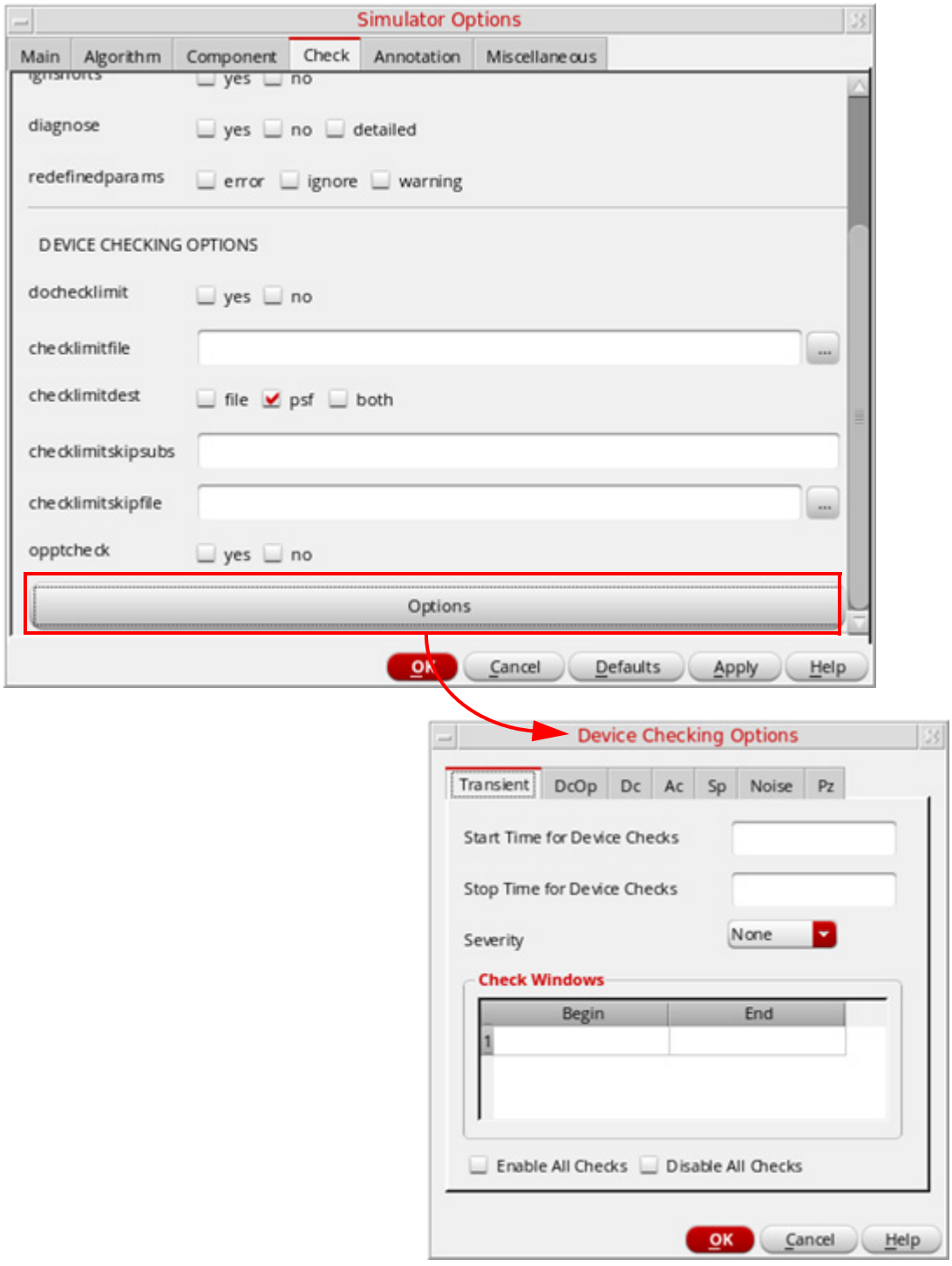
1. Right-click a test and choose *Options – Analog* from the context menu.

   The Simulator Options form appears. For more information about how to use this form, see Setting Spectre Options.

2. Open the *Check* tab.
3. Select the *yes* check box for *dochecklimit* to enable device checks.
4. Click the *Options* button available at the bottom of the Check tab.

© 2023 Cadence Design Systems, Inc.

The Device Checking Options form appears, as shown in the figure below.



When you open this form, the *Transient* tab is open by default. You can also use this form to specify the global device check options for various analyses.

© 2023 Cadence Design Systems, Inc.

The following fields are displayed on the Transient tab:

- *Start Time for Device Checks*–Specify the beginning time at which device checks are to be enabled or disabled for transient analyses.
- *Stop Time for Device Checks*–Specify the end time at which device checks are to be enabled or disabled for transient analyses.
- *Severity*–Select the default severity level for the message that is displayed when Spectre reports violations for the device checks that are enabled for transient analyses. The available options are: None, Error, Warning, Notice, and Fatal.

  The specified severity level overrides the severity level specified for individual checks in the Checks and Asserts assistant.

  Note: If you set the severity to None, the severity level specified for individual checks in the Checks and Asserts assistant will be used.
- *Check Windows*–Specify the time windows when device checks will be enabled. In the Begin and End columns, specify the begin and end times for each time window, one per line in the table.
- *Enable All Checks*–Select this check box to enable all the device checks for transient analyses.

  Note: If this check box is not selected, only the device checks that are enabled for transient analyses in the Checks and Asserts assistant are used for transient analyses.
- *Disable All Checks*–Select this check box to disable all the device checks for transient analyses.

© 2023 Cadence Design Systems, Inc.

The remaining tabs, *DcOp*, *Dc*, *Ac*, *Sp*, *Noise*, and *Pz*, include only three fields as shown in the figure below:

© 2023 Cadence Design Systems, Inc.

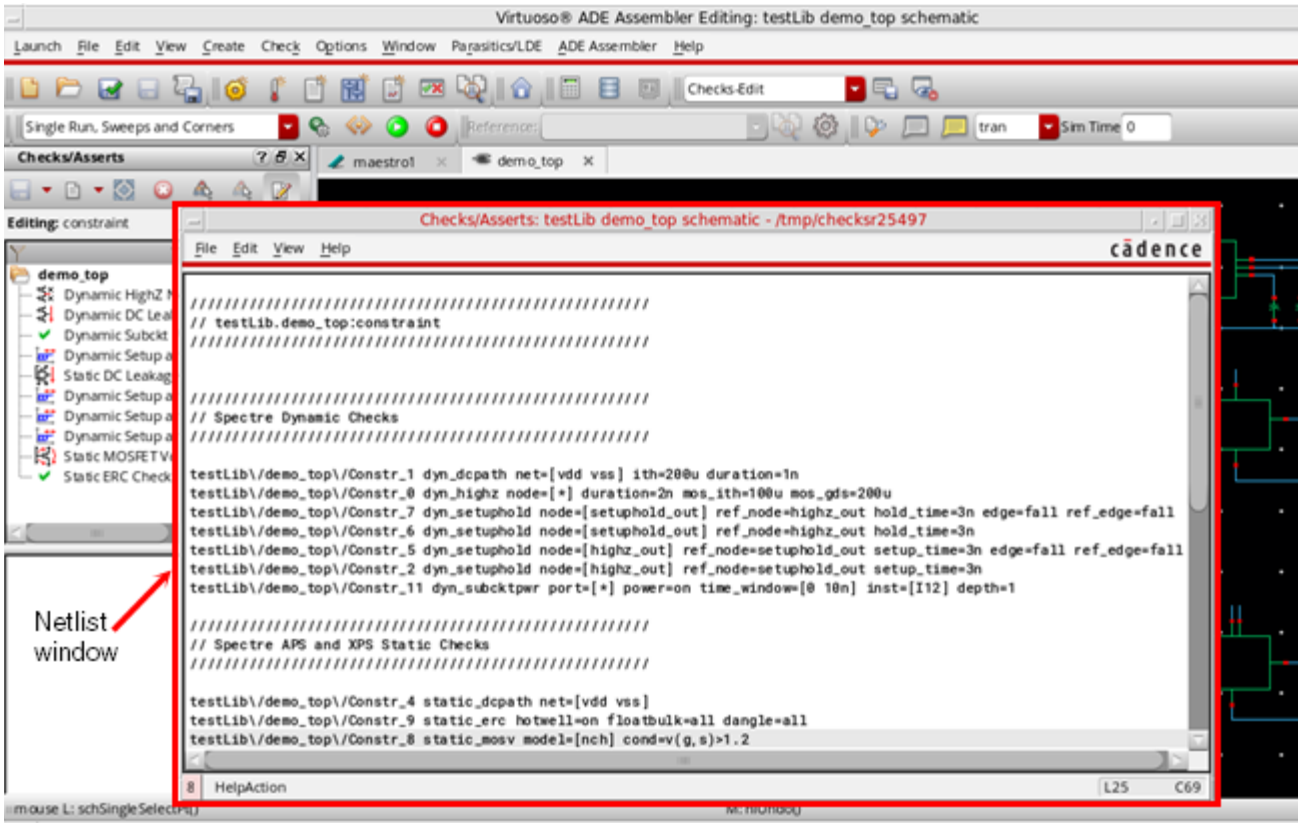
Viewing Netlists for Checks and Asserts

While you are creating or editing checks and asserts in the Checks/Asserts assistant, you can also view how they will be added to the netlist for simulation. You can do this in two ways:

- Viewing netlist statements for all the checks/asserts in a text window
- Viewing the netlist preview for an individual check in the tooltip

Viewing netlist statements for all the checks/asserts in a text window

You can generate the netlist for all the checks and asserts defined at the current design hierarchy level by using the *Netlist Checks/Asserts* command in the *Device Check* drop-down menu on the Checks/Asserts assistant toolbar. This command opens a netlist window that shows all the checks defined at the current level.



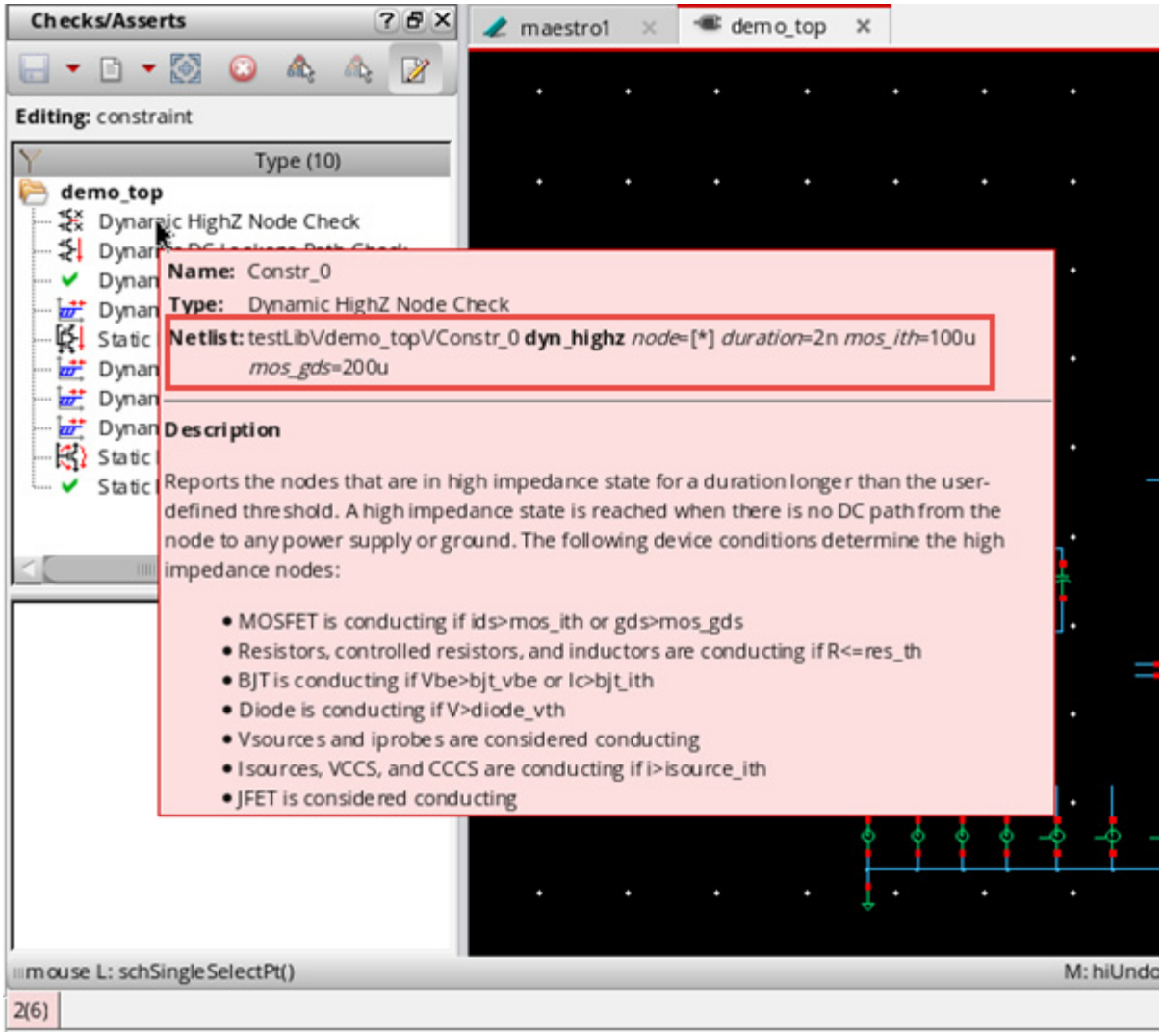
For each design level, you need to run this command separately to view the netlist at that level.

Note: If any check or assert has been disabled in the Checks/Asserts assistant, it will not be netlisted.

© 2023 Cadence Design Systems, Inc.

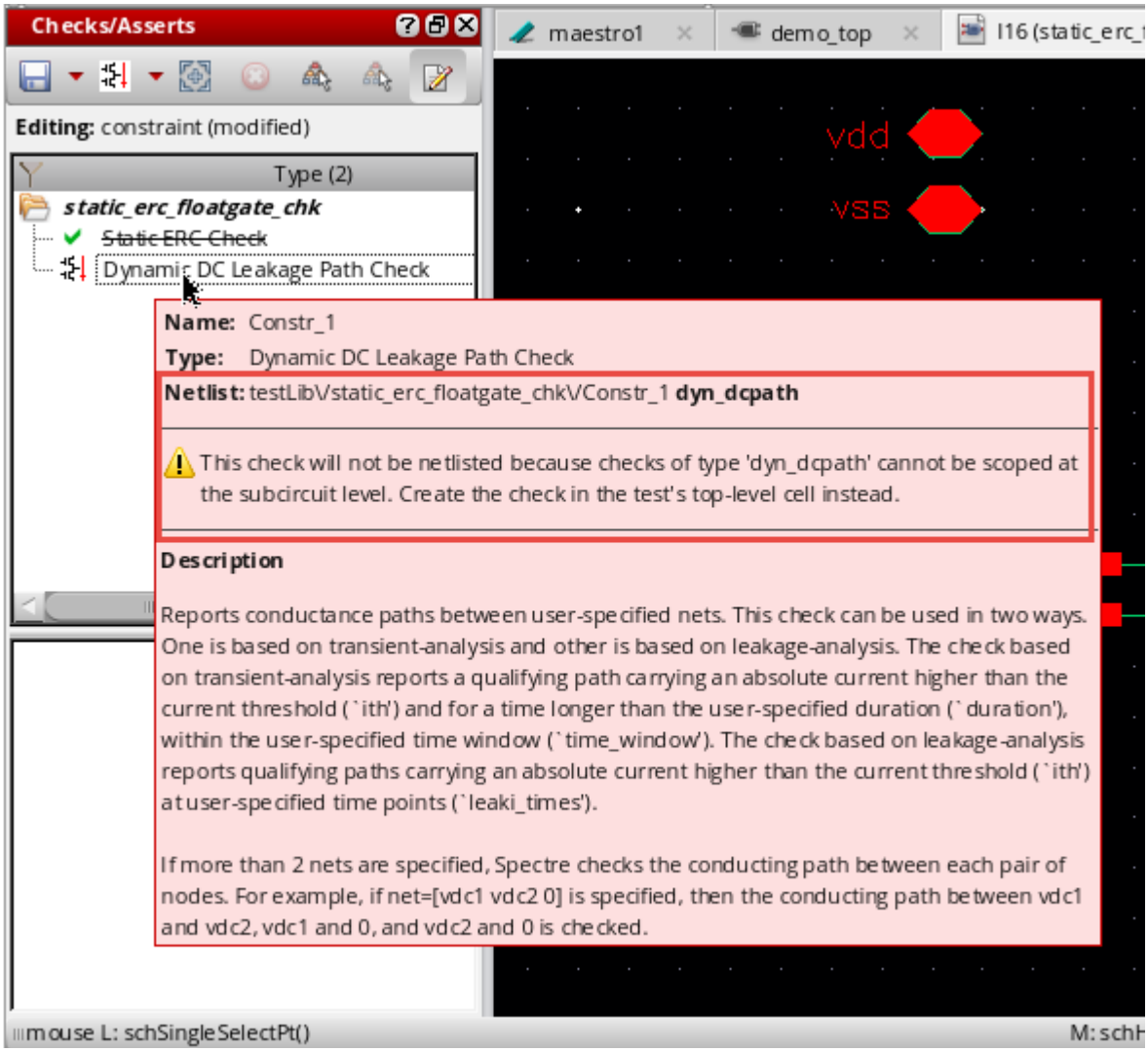
Viewing the netlist preview for an individual check in the tooltip

To preview the netlist for a single check, move the mouse over the check name in the Checks/Asserts assistant. A tooltip will be displayed where you can view the netlist statement.



© 2023 Cadence Design Systems, Inc.

If the netlist for a particular check at the subcircuit level is not valid, appropriate warnings are displayed, as shown below.



Next Steps

After creating circuit checks, the next steps are to:

- Define violation filters and specifications on the Outputs Setup tab
- Run simulations
- View violations on the Results tab

© 2023 Cadence Design Systems, Inc.

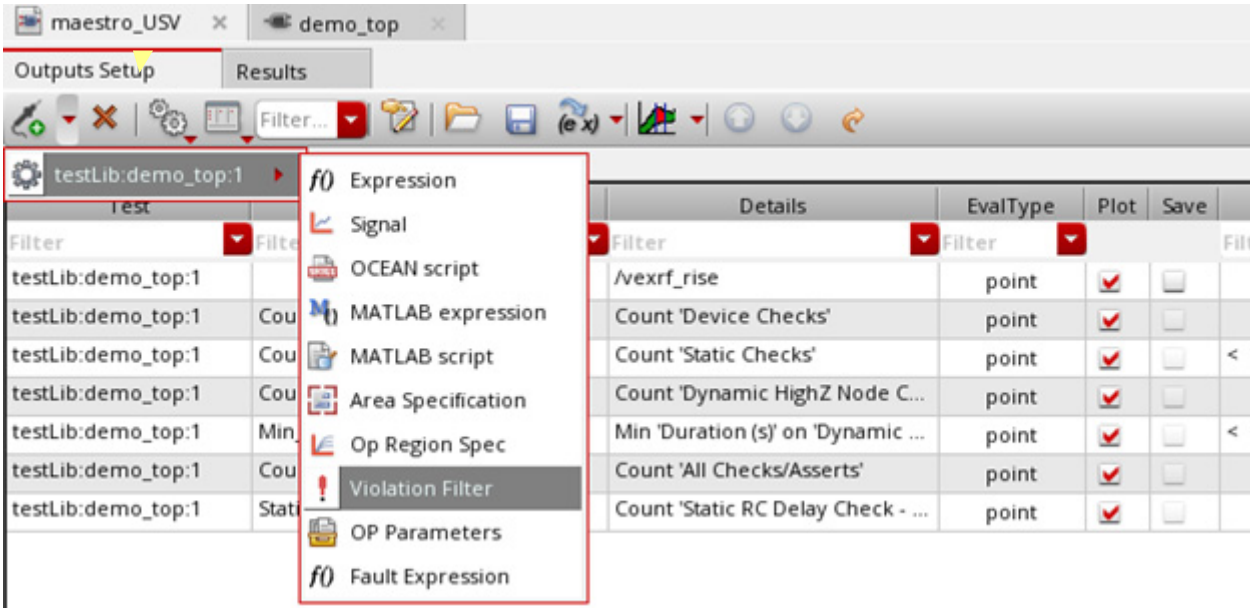
Filtering Violations

Note: Read a blog related to this at Playing with Violation Filters.

Violation filters provide a way to visualize the different metrics for checks, such as the number, duration, or any user-specified categories of violations. Each violations filter can have its own specification and is treated like any other output.

To create a violation filter, perform the following steps:

1. On the Outputs Setup tab, click *Add new output* — `<testName>`— *Violation Filter* to add a new output to show violation filters for checks or asserts.

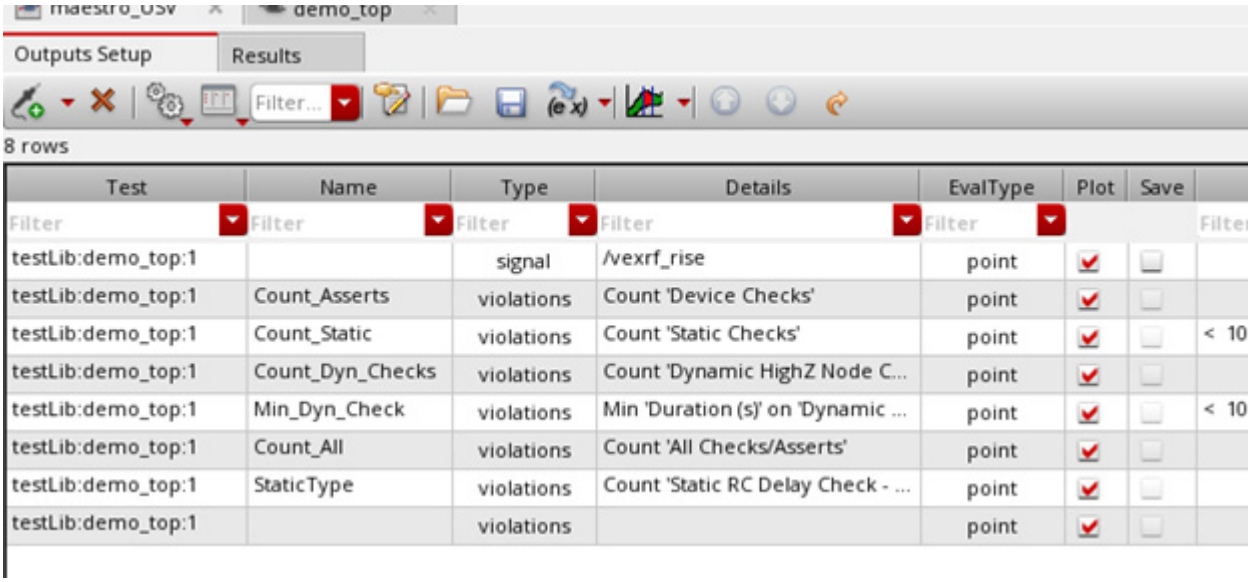


   A new row is created with its type set as `violations`.

2. Enter a name in the *Name* column for the violation output.

© 2023 Cadence Design Systems, Inc.

3. Double-click the *Details* column for this row.

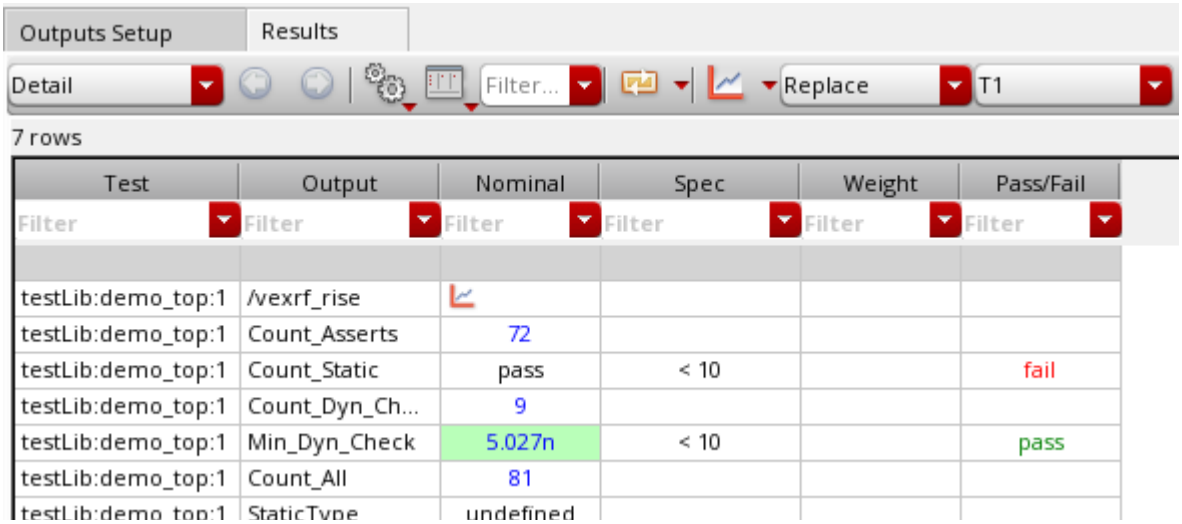


The following three drop-down lists are displayed:

- The first drop-down list is the type of operation you want to perform. For example, you can select `Count` to count the number of violations. Other operations available in this drop-down list are `Avg`, `Min`, `Max`, or `Sum`.
- The second drop-down list specifies the field you want to access within the violation result. For example, for the Avg such as Duration or PeakValue. If the first value is `Count`, the second value defaults to `*`. Otherwise, the choices in the second list vary depending on the item select in the third list.
- The third drop-down list shows the list of checks defined for the test. Choose a check on which you want to run the operation selected in the first drop-down list. You can select an individual violation type, for example, `Dynamic HighZ Node Check` or

© 2023 Cadence Design Systems, Inc.

a grouped category, such as `Dynamic Checks` or `All Checks/Asserts`. The second list of fields is updated depending on the selection in this list.



4. Select values from the drop-down lists to build an expression for the violation filter.

   Example expressions:

   - `Count 'Device Checks'`
   - `Max 'RMS(W)' on 'Dynamic Subckt Port Power Check - Port Power Report'`

5. Add a specification for the violation in the *Spec* column for the output.

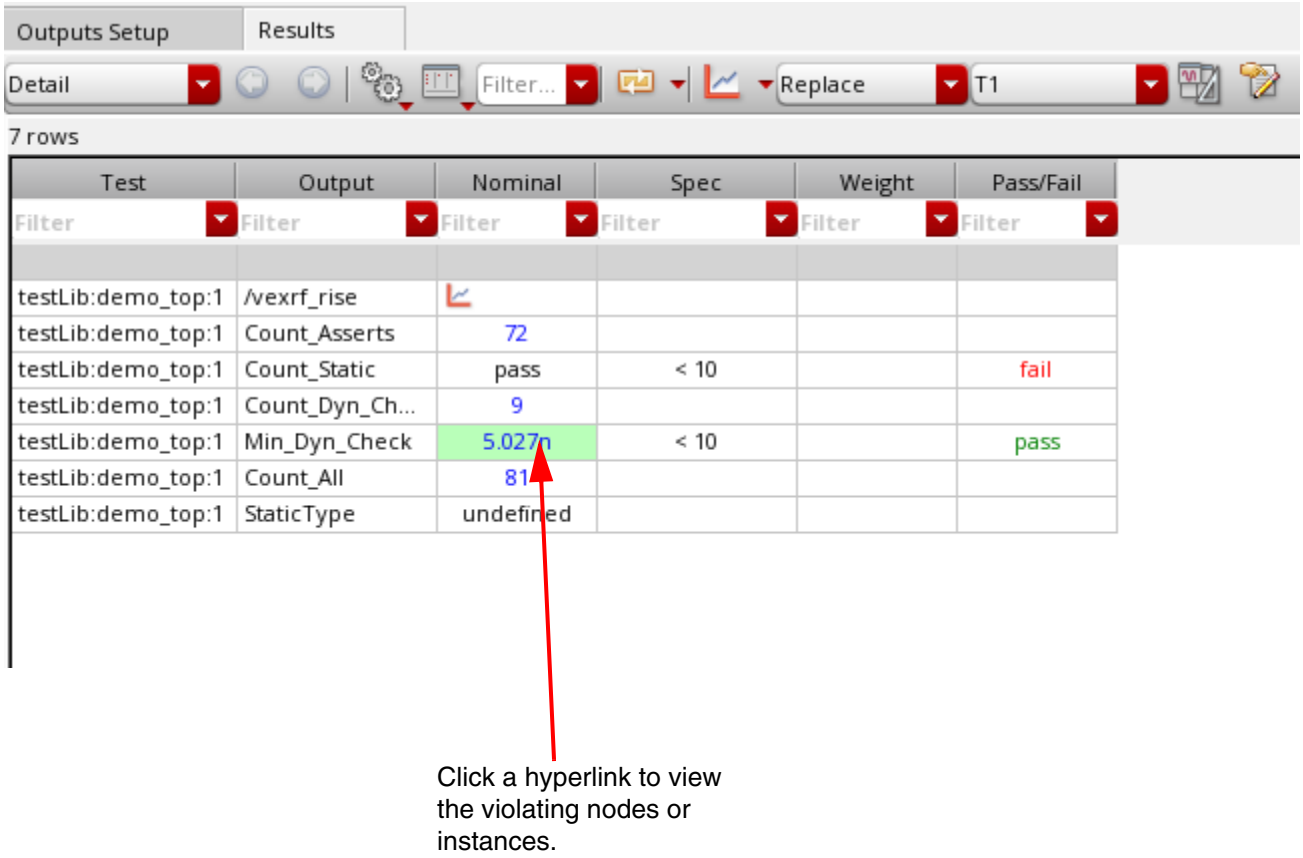
Note: If you have a set of read-only violation filters defined in one or more XML, provide a colon-separated list of XML files using the checksAssertsFiltersPath environment variable. Filters specified using this environment variable are shown in italics in the user interface and are given preference over the customized filters defined in the local `.cadence` directory.

© 2023 Cadence Design Systems, Inc.

Running Simulations and Viewing Violations

After adding checks in the Checks/Asserts assistant and defining violation filters in the Outputs Setup pane, you can run a simulation to run the checks on the simulation results. After the simulations of all points and corners are complete, the `Detail` results view shows a consolidated report of the violation filter expressions defined in the Outputs Setup tab. For example, `Count` reports the number of violations of the specified type.

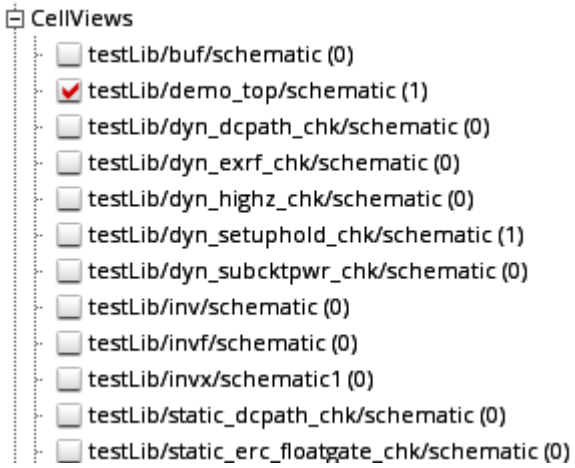
If no violations are found, the result for the violation filter shows `No violations`. When violations are reported, the status is either `pass` or `fail`.



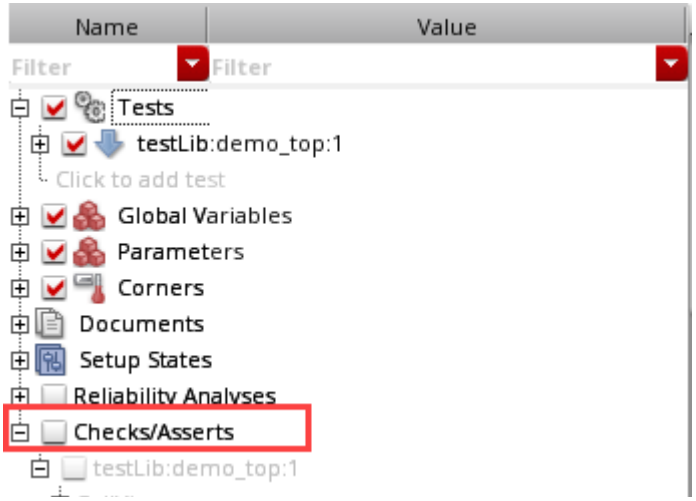
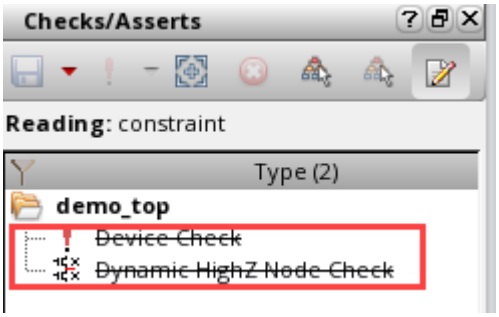
The following table lists the possible result values for the `violations` type of outputs:

Result Value	Description
`<count of violations>`	Indicates that the simulation was run and found a number of violations for this output.

© 2023 Cadence Design Systems, Inc.

`No violations`	Indicates that the *docheckklimit* device checking option on the *Checks* tab of the Simulator Options form is disabled or The check or is disabled in the constraint view
`undefined`	The given type of check is not defined or is disabled in the maestro cellview. For example, if the static type of checks are not defined in the maestro cellview, as shown below.  In this case, the `count 'Static Checks'` output shows the result value as `undefined`.
`Canceled`	If a simulation is canceled before completion, ADE Assembler looks for violations in the results database. If found, violations are displayed on the Results tab. Otherwise, it displays `Canceled`. In this case, you can click the drop-down button next to the command on the *Results* tab and select *Partial Data* to re-evaluate results using the partially complete simulation data. If partial data is available, Assembler loads the violations from that data. It also displays a warning icon next to the values to indicate that they are based on partial data.

© 2023 Cadence Design Systems, Inc.

`disabled`	Indicates that no checks or asserts was run, and no SQL database was generated. This could be because: The Checks/Asserts check box in the Data View assistant is disabled  or an error was found in the checker definition 

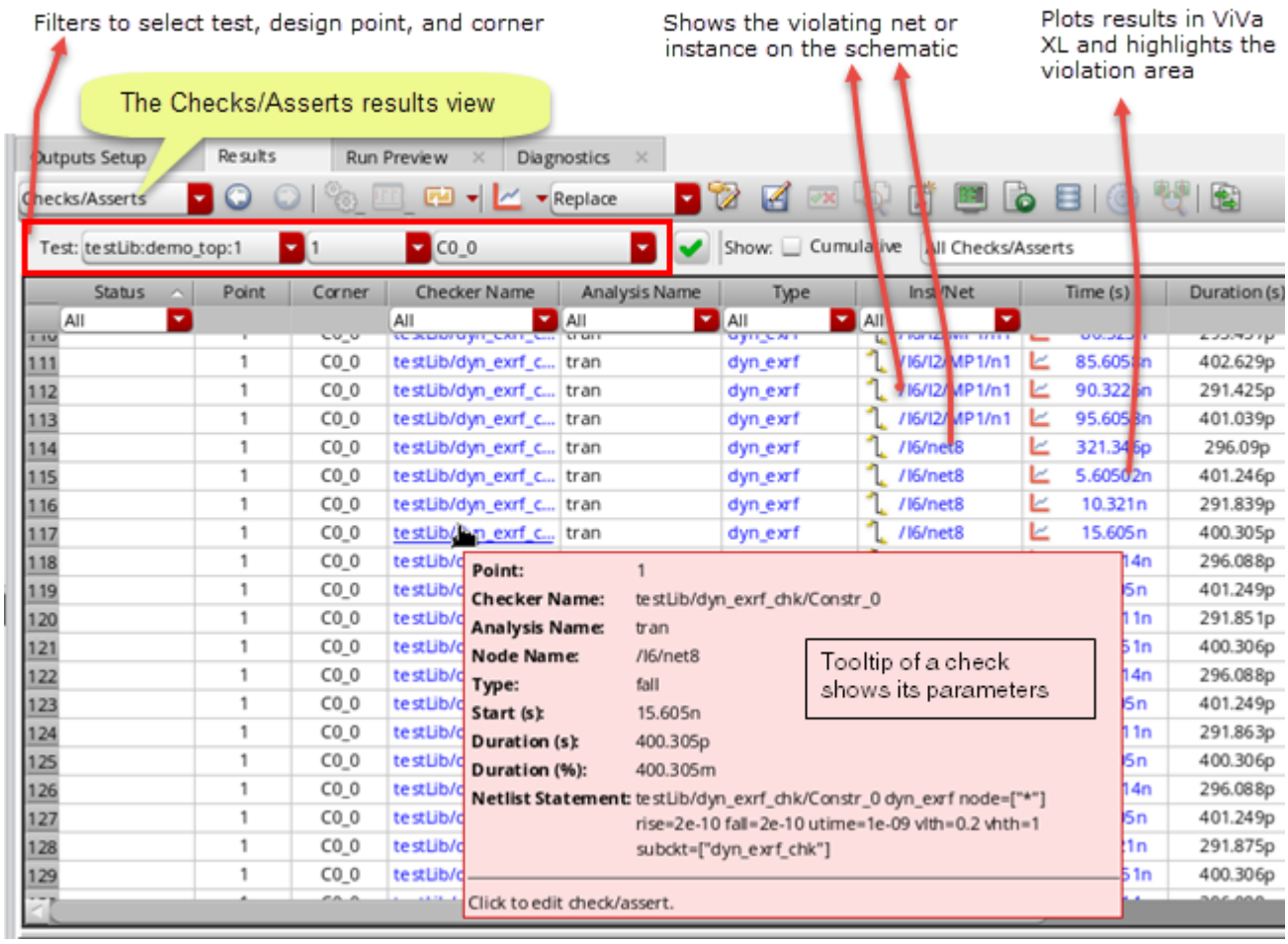
Video

To understand the details of each type of status, you can view the Checks/Asserts Summary Table and Status Display video demonstration.

The violation count value for each corner is a hyperlink. Click any value with hyperlinks to display the detailed results that contributed to this value. The net- or instance-wise details are opened in `Checks/Asserts` results view. With an aim to resolve violations, you can use the hyperlinks in that detailed view, find the violating nodes or instances, waive violations, or add comments.

© 2023 Cadence Design Systems, Inc.

The `Checks/Asserts` results view provides a lot of controls to customize the display of violation details, as shown below.



To check or edit the details of any device check directly from this view, click on the corresponding instance name in the *Inst/Net* column of the results table. The schematic of the instance is opened in a new design tab within the ADE Assembler window. The `Checks-Debug` workspace of the schematic view shows the Checks/Asserts assistant where you can view or edit the details of a device check.

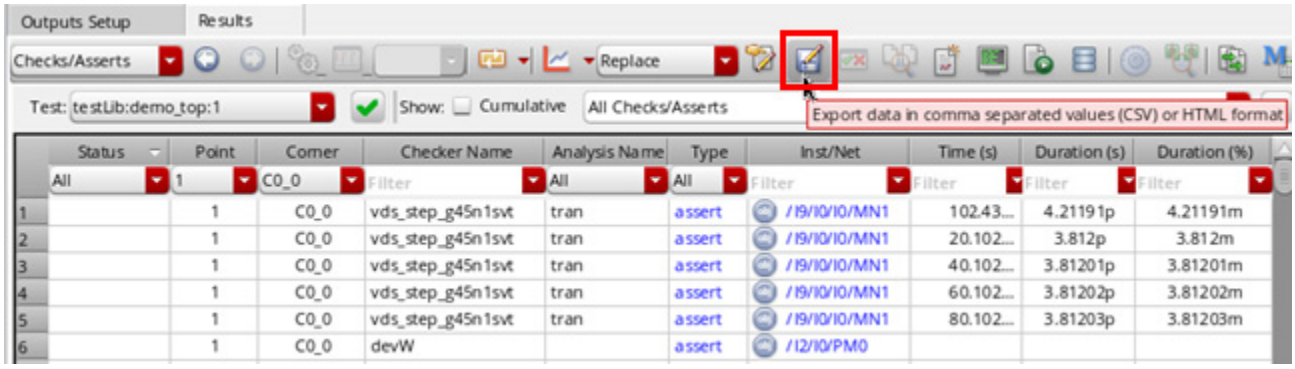
Related Topics

refreshViolationPerCorner

© 2023 Cadence Design Systems, Inc.

Exporting Checks/Asserts Results

You can export the results displayed in the `Checks/Asserts` results view into a CSV file. For this, click *Export Outputs to CSV File* on the toolbar. The Export Results form is displayed. Specify a file name and choose a file type as `.csv` or `.html`. Click *OK* to export the results.



For Checks/Asserts results, the tool exports the data in batch mode with the default set of `10000` rows in a batch. You can change the batch size by setting the `exportBatch` environment variable. While exporting data, the tool shows a progress bar at the bottom of the Results tab. After the data is successfully exported, a message is displayed to report the number of rows exported to the CSV file.

Related Topic

exportBatch

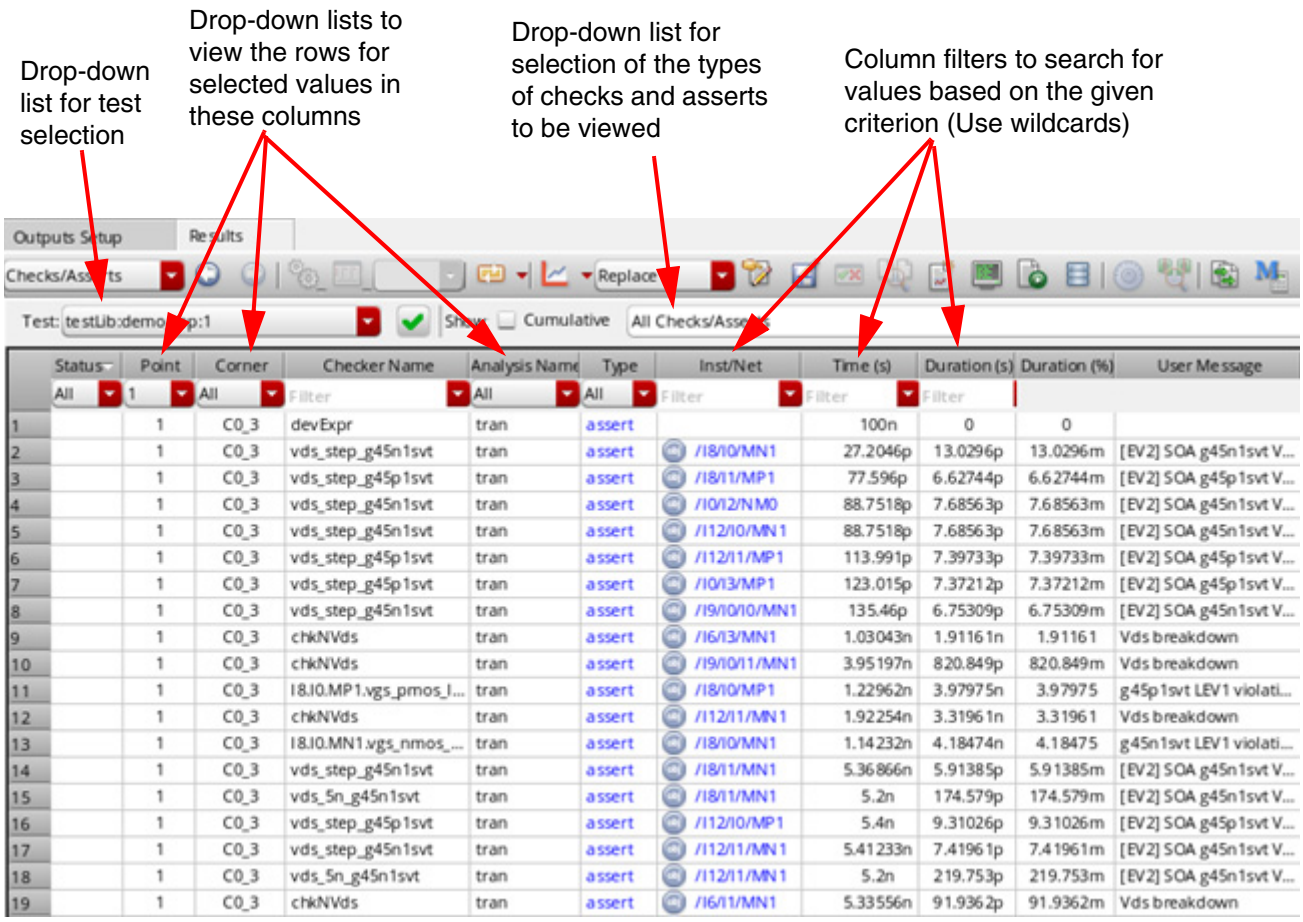
The following sections provide details about how to customize the violations display:

- View itemized violations details
- Waive violations
- View cumulative violation details
- View summary of violations
- View instances with violations on schematic
- Create a custom violation filter
- View violation in ViVA XL graphs
- View violation reports in Web browser
- Viewing results of device checks

© 2023 Cadence Design Systems, Inc.

Viewing Itemized Violation Details

In the itemized view, the details of each violating net or instance for a given test/design point/corner combination are displayed in the `Checks/Asserts` results view. The names of the test for which the results are displayed appears at the top. You can select a different test from the *Test* drop-down list.



Customizing the Display of Violations in the Checks/Asserts Table

You can customize the view of the Checks/Asserts table in the following ways:

- **Hide the columns you do not want to see.** For this, use the *Hide Column* command from the context-sensitive menu of the Checks/Asserts table.

  You can use the *Show All Columns* command to show the hidden columns.

- **Change the order in which columns are displayed.** For this, select the column you want to move, drag it to the location where you want to place it.

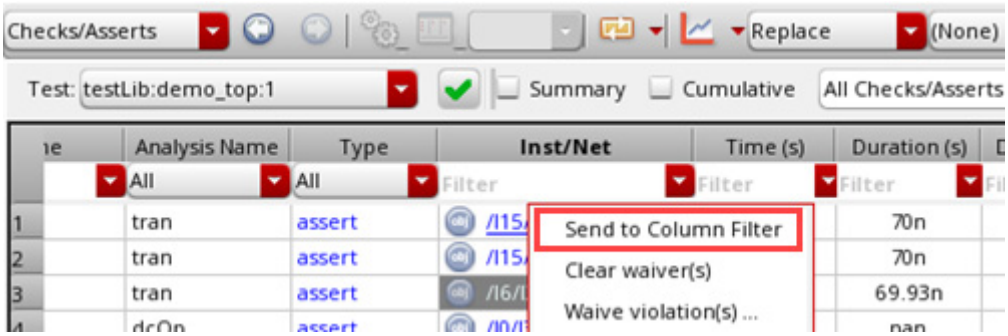
© 2023 Cadence Design Systems, Inc.

- **Apply a field or column filters to view the selected violation data.** Columns such as *Point*, *Corner*, or *Analysis Name* have drop-down filters that you can use to select a specific value to filter data.

  Columns such as *Checker Name*, *Inst/Net*, *Time (s)*, *Duration (s)*, and *Comments* provide enhanced filters where you can type in the filter criteria using specific values and wildcard characters. Specify a criterion and press `Enter` to view the filtered rows.

Tip

To use a particular value from a column as the filter criterion for that column, right-click the value and choose *Send to Column Filter*. The selected value appears in the *Filter* cell of the column, and only the rows containing that value are shown in the outputs table.



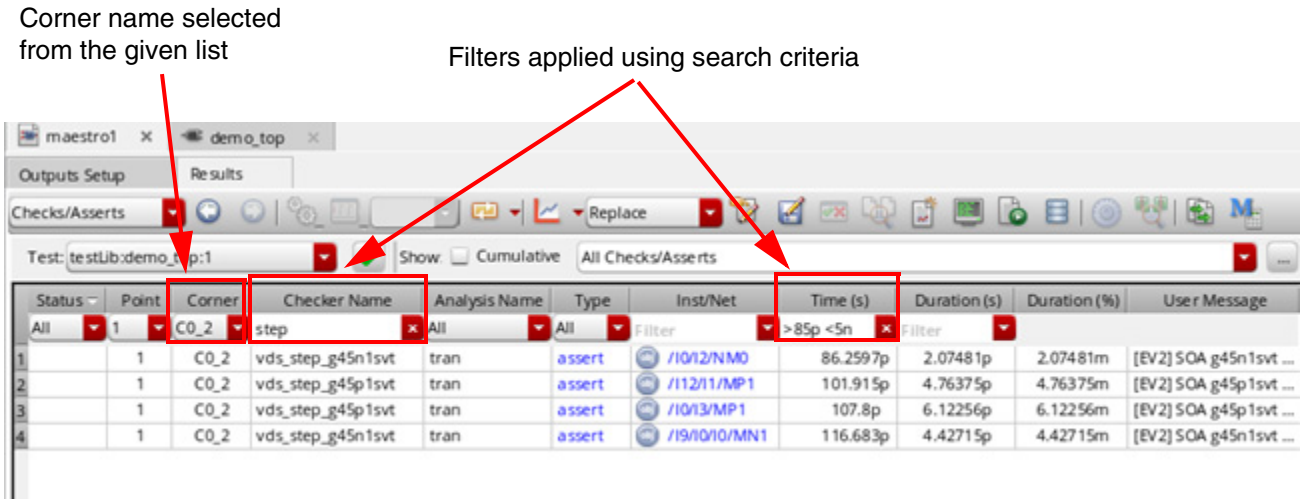
Note the following important points about the column filters:

- **You can apply filters in multiple columns together**.

  The following example shows how column filters have been used to view all the violations for corner `C0_2` where *Checker Name* contains `step` and *Time (s)* is between `85p` and `5n`.

© 2023 Cadence Design Systems, Inc.

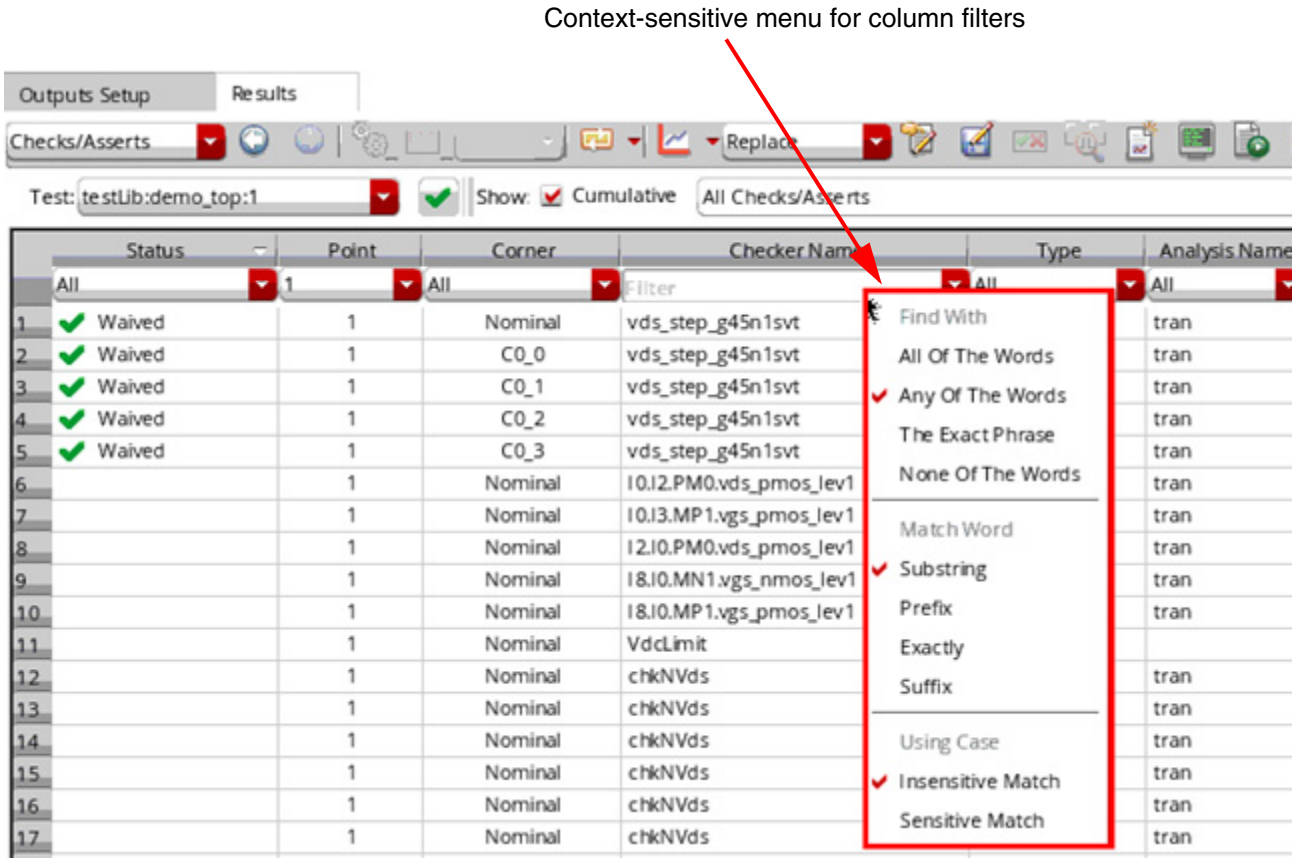
If you are not using comparison operators, the numerical values are compared like strings.



- **You can use numerical comparison operators**, such as > or <, as shown for the *Time (s)* column in the above example.

© 2023 Cadence Design Systems, Inc.

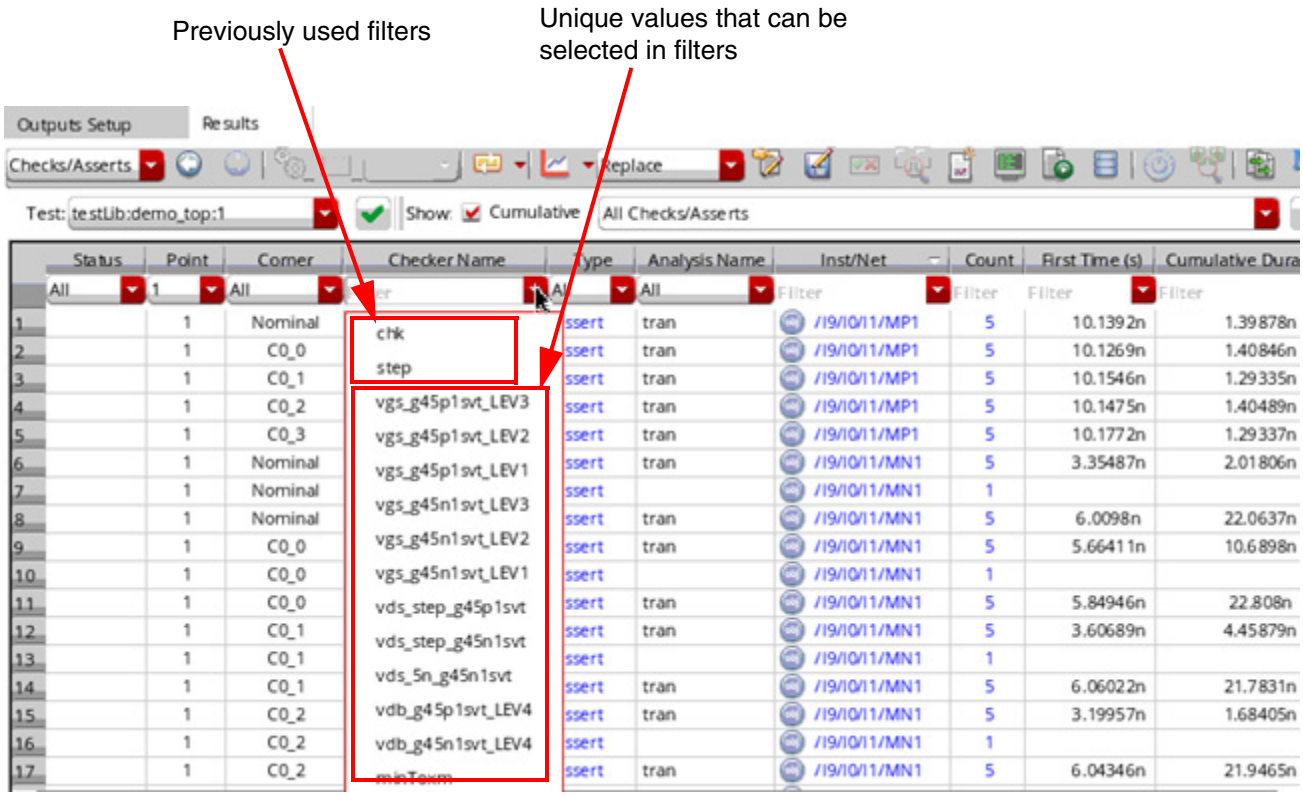
- **You can use filtering options from the context-sensitive menu** of the cell in which you applied a filter. These options are same as the filtering options in other result views.



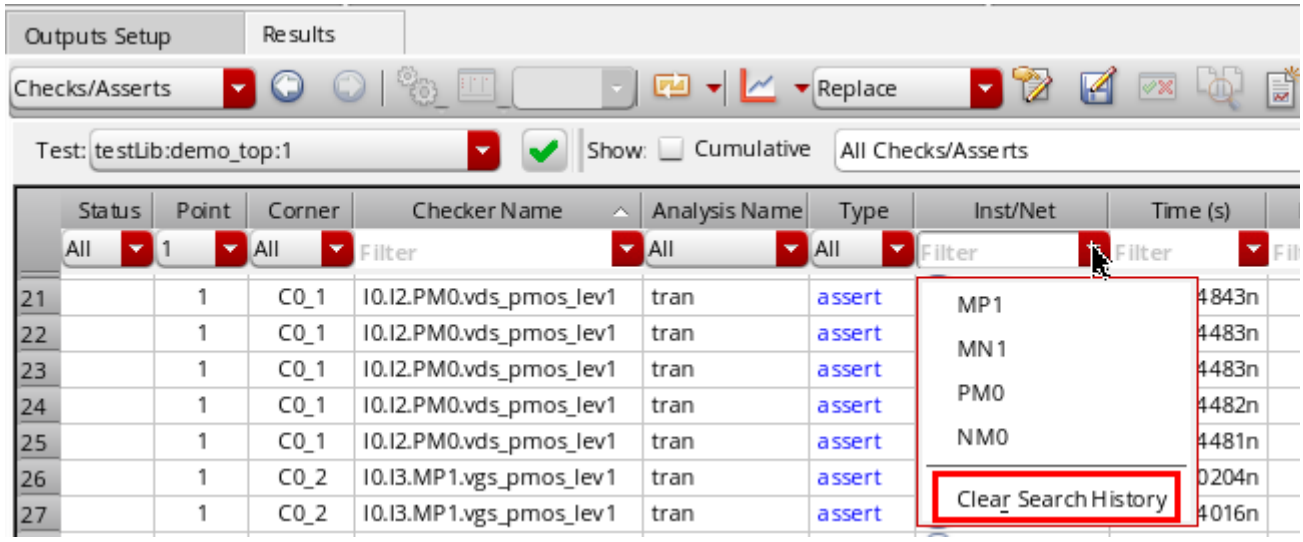
Note: In the *Message*, *User Message*, *Comment*, and *Expression* fields, you can use the *Any of the words* filtering option to filter the rows that contain any of the multiple strings specified in the *Filter* cell.

© 2023 Cadence Design Systems, Inc.

- **You can choose filters from the drop-down list**. As shown in the example given below, the list shows available values including the previously used filters and the unique values in that column.

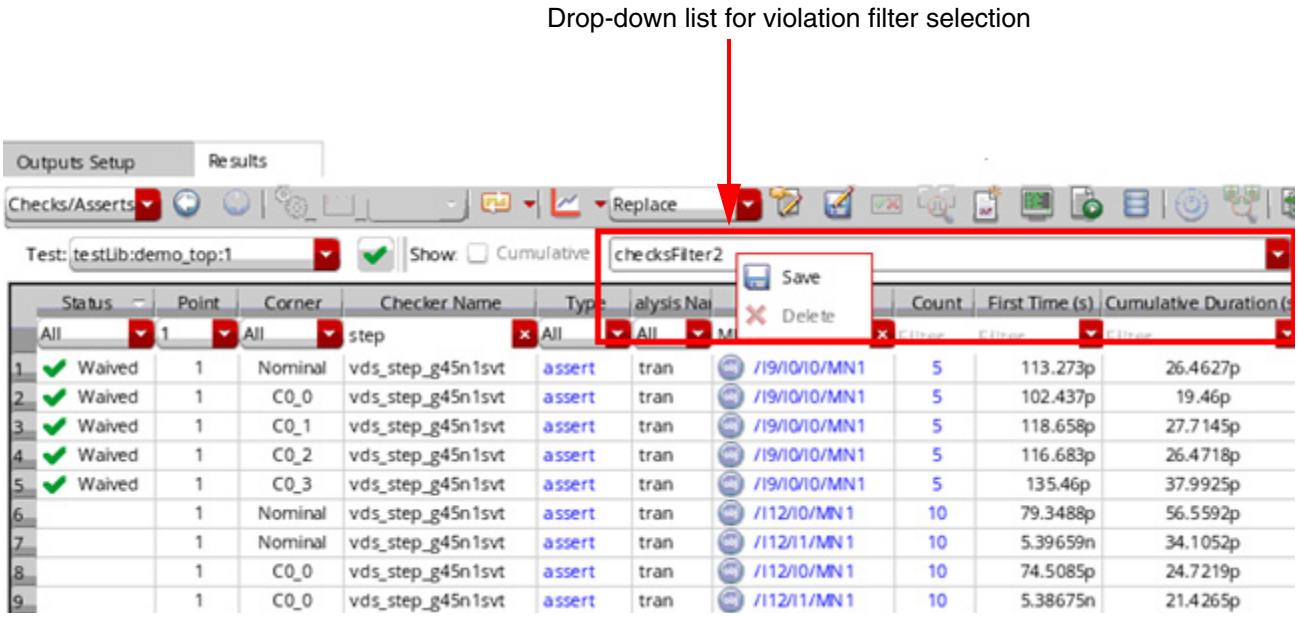


- **You can clear the previously used filters for a column by using the *Clear Search History* command** in the context-sensitive menu of that column.



© 2023 Cadence Design Systems, Inc.

- **You can save the current set of filters** in the Checks/Asserts table with a name. For this, specify filters in all the required columns, type a name in the drop-down list for violation filter selection. Right-click the drop-down list and choose *Save*.



The filters are saved in the `.cadence/dfII/maestro/check-filters` directory and are available for use when you reopen the cellview later.

Later, you can reapply a saved filter by selecting it from the list of violation filters. If not required, you can delete a saved filter by using the *Delete* command in the context-sensitive menu of the drop-down list.

The saved filters are also available in the Customize Violation Filters form, where you can modify them further.

- **You can hide the filters** by using the *Hide Filters* command from the context-sensitive menu of the column headers in the Checks/Asserts table.

Tip

If required, you can choose to waive a few violations by using the *Waive Violation(s)* command in the context-sensitive menu of the violation rows. While you mark a violation as waived, you can also add a comment providing the reason or other information about that. These comments are visible in the the *Comment* column (the right-most column in the Checks/Asserts table).

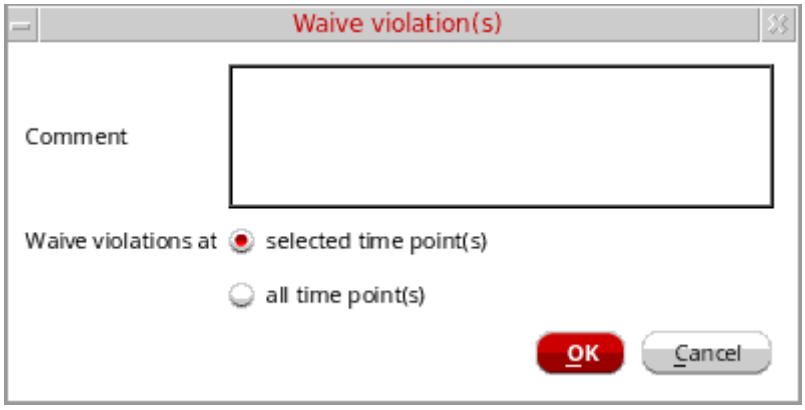
© 2023 Cadence Design Systems, Inc.

Waiving Violations

When you waive a violation, all the relevant violations that occur under the same check/assert, on the same design object and in the same analysis are waived. If there are certain violations that can be ignored, you can choose to wave those off. For example, you can choose to waive all the violations that have the start time before `5n`.

To waive a violation, perform the following steps:

1. Click the header of the *StartTime (s)* field to sort the values in ascending order.
2. Hold the Shift key and select all the rows that have values less than 5n. Right-click anywhere in the selected rows and choose *Waive Violation(s)*. The Waive Violation(s) form is displayed.



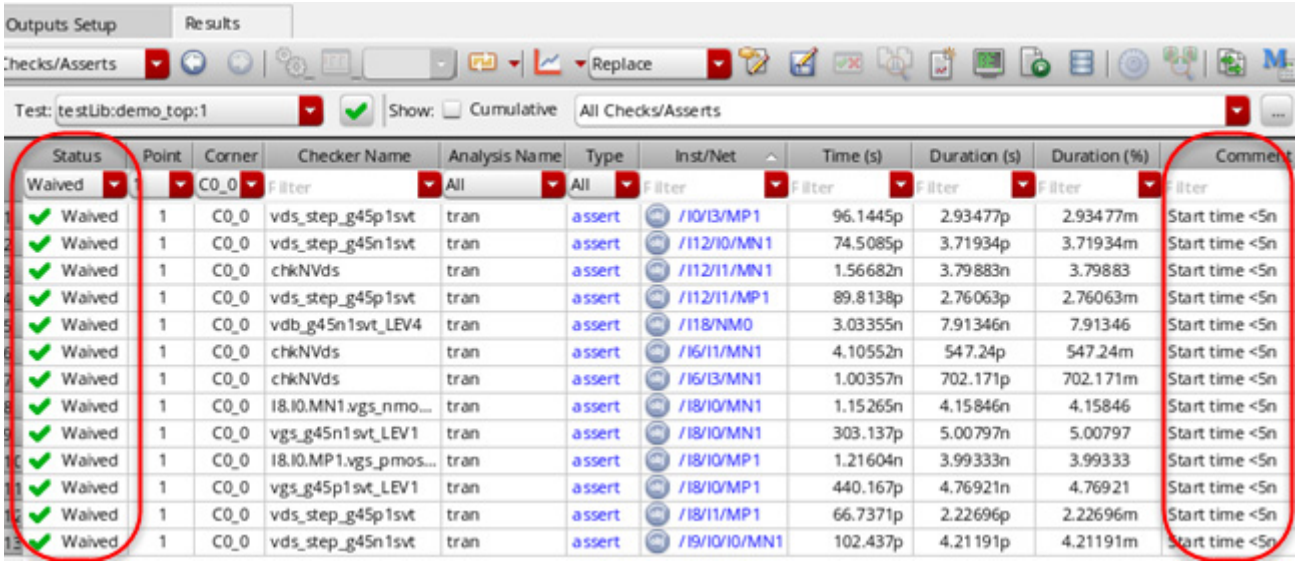
3. Add an appropriate comment in the *Comment* field.
4. You can choose to waive:
   a. The violations under the selected time points by selecting the *selected time point(s)* option.
   b. The violations across all time points by selecting the all *time point(s)* option.

   For this example, ensure that the *selected time point(s)* option is selected.

   Additionally, you can use the environment variable waiveViolationsByCorner to waive only the violations under the same design point and corner.
5. Click *OK*.

© 2023 Cadence Design Systems, Inc.

For the waived violations, the *Status* field shows the status as `Waived`. and the *Comment* field shows the comments you added on the Waive Violation(s) form.



	Status	Point	Corner	Checker Name	Analysis Name	Type	Inst/Net	Time (s)	Duration (s)	Duration (%)	Comment
	Waived		C0_0	Filter	All	All	Filter	Filter	Filter	Filter	Filter
1	Waived	1	C0_0	vds_step_g45p1svt	tran	assert	/I0/I3/MP1	96.1445p	2.93477p	2.93477m	Start time <5n
2	Waived	1	C0_0	vds_step_g45n1svt	tran	assert	/I12/I0/MN1	74.5085p	3.71934p	3.71934m	Start time <5n
3	Waived	1	C0_0	chkNVds	tran	assert	/I12/I1/MN1	1.56682n	3.79883n	3.79883	Start time <5n
4	Waived	1	C0_0	vds_step_g45p1svt	tran	assert	/I12/I1/MP1	89.8138p	2.76063p	2.76063m	Start time <5n
5	Waived	1	C0_0	vdb_g45n1svt_LEV4	tran	assert	/I18/NM0	3.03355n	7.91346n	7.91346	Start time <5n
6	Waived	1	C0_0	chkNVds	tran	assert	/I6/I1/MN1	4.10552n	547.24p	547.24m	Start time <5n
7	Waived	1	C0_0	chkNVds	tran	assert	/I6/I3/MN1	1.00357n	702.171p	702.171m	Start time <5n
8	Waived	1	C0_0	I8.I0.MN1.vgs_nmo...	tran	assert	/I8/I0/MN1	1.15265n	4.15846n	4.15846	Start time <5n
9	Waived	1	C0_0	vgs_g45n1svt_LEV1	tran	assert	/I8/I0/MN1	303.137p	5.00797n	5.00797	Start time <5n
10	Waived	1	C0_0	I8.I0.MP1.vgs_pmos...	tran	assert	/I8/I0/MP1	1.21604n	3.99333n	3.99333	Start time <5n
11	Waived	1	C0_0	vgs_g45p1svt_LEV1	tran	assert	/I8/I0/MP1	440.167p	4.76921n	4.76921	Start time <5n
12	Waived	1	C0_0	vds_step_g45p1svt	tran	assert	/I8/I1/MP1	66.7371p	2.22696p	2.22696m	Start time <5n
13	Waived	1	C0_0	vds_step_g45n1svt	tran	assert	/I9/I0/I0/MN1	102.437p	4.21191p	4.21191m	Start time <5n

You can later review and clear these waivers by using the *Clear Waiver(s)* command in the context-sensitive menu of the violation rows.

Important

The waivers are applied on all the design points and corners, even if they are not currently being displayed. You can import and reapply the waiver to the same violations for other combinations as well, as described below.

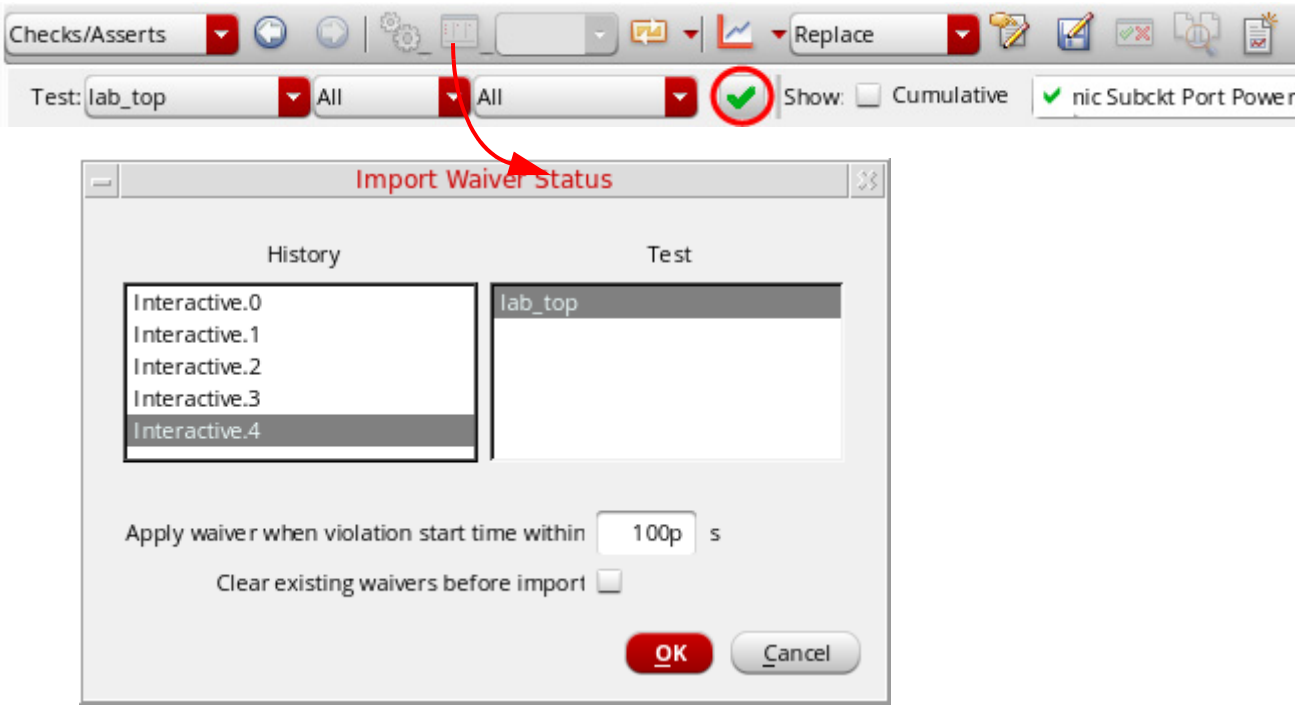
Importing Waiver Status

To import the waiver status for violations from some other combination of test, design point, and corner, perform the following steps:

1. Click *Apply waiver status from another test and/or history* on the toolbar.

© 2023 Cadence Design Systems, Inc.

The Import Waiver Status form is displayed.

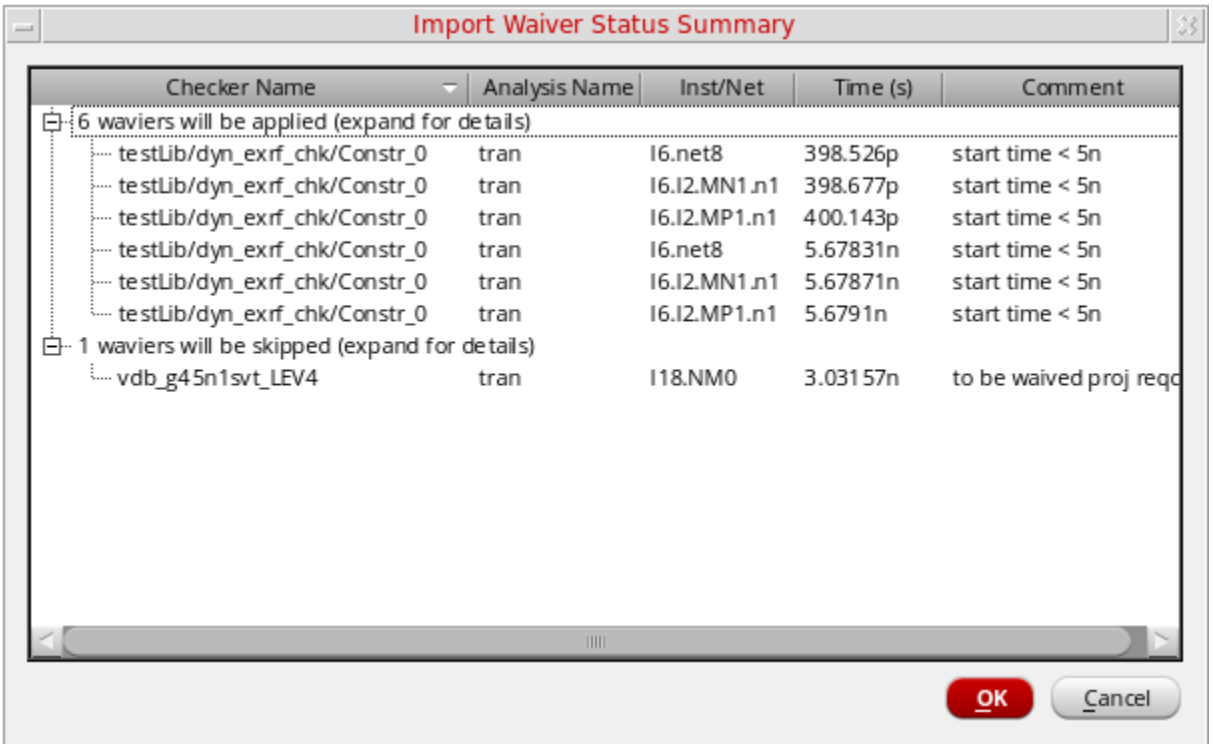


2. From the *History* and *Test* lists, select the names of the history and test from which you need to import waivers.
3. (Optional) Change the criteria for the start time value defined in the *Apply waiver when violation start time within* field.

   This field controls whether a waiver will be applied or not by comparing the violation start time in the current results and the violation start time in the imported waiver. If the difference in the two values is less than the value specified in this field, the waiver is applied, otherwise it is skipped. This is useful where a violation occurs across multiple corners/points or runs, but the start time varies slightly due to differing circuit conditions.
4. Click *OK*.

© 2023 Cadence Design Systems, Inc.

ADE Assembler identifies the violations for which waivers can be applied and shows a summary in the Import Waiver Status Summary window, as shown below.



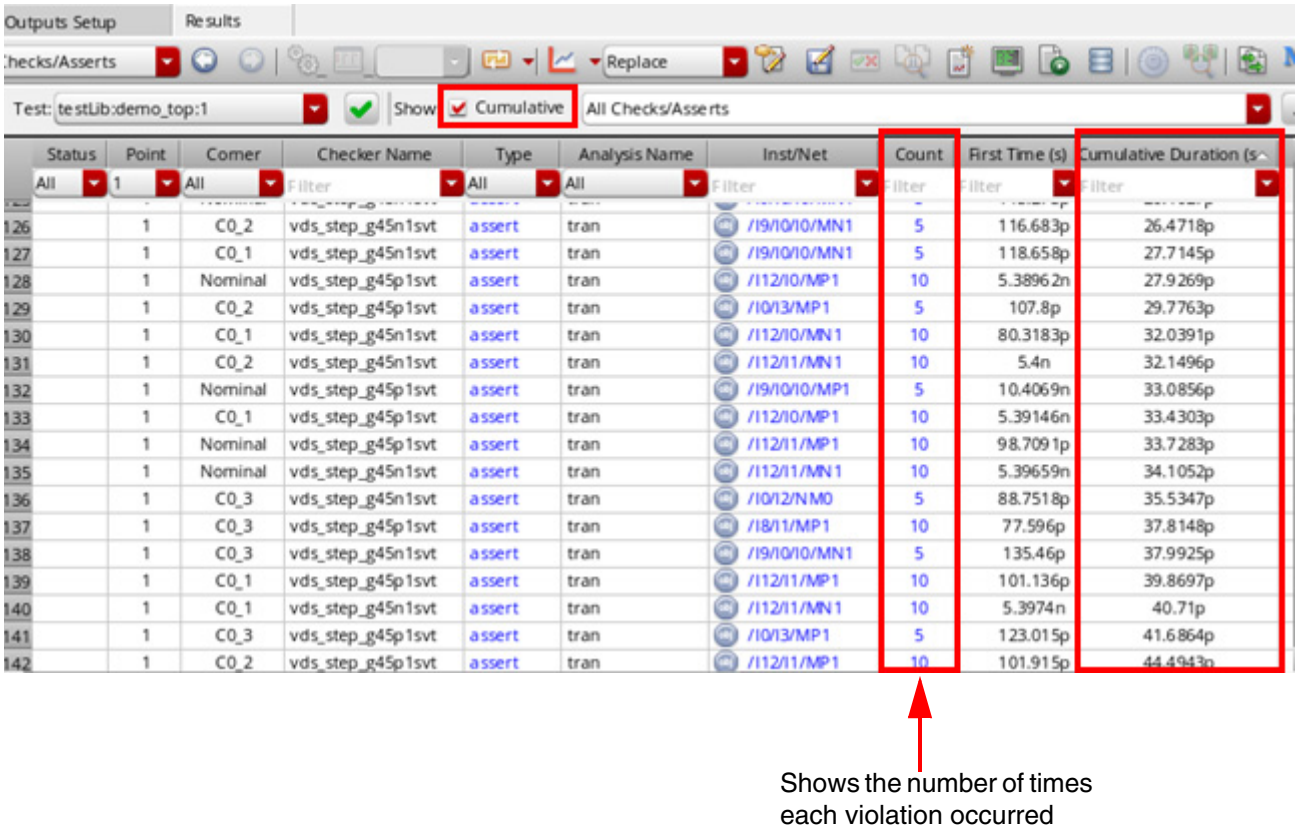
5. Click *OK* to import the violations.

   The waivers are imported and applied to the same violations in the current test, corner, and design point combination. The *Status* and *Comment* fields are also updated.

© 2023 Cadence Design Systems, Inc.

View Cumulative Violation Details

Selecting the *Cumulative* check box shows the cumulative results across time where a row is shown for each unique check. The *Count* column for each check shows the number of times for which a violation occurred for a particular net or instance.



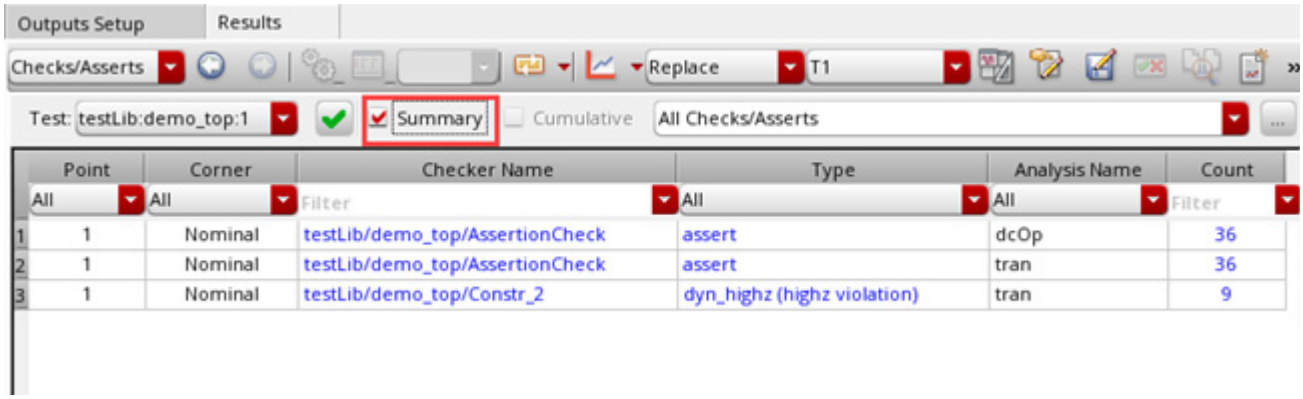
	Status	Point	Corner	Checker Name	Type	Analysis Name	Inst/Net	Count	First Time (s)	Cumulative Duration (s
	All	1	All	Filter	All	All	Filter	Filter	Filter	Filter
126		1	C0_2	vds_step_g45n1svt	assert	tran	/I9/I0/I0/MN1	5	116.683p	26.4718p
127		1	C0_1	vds_step_g45n1svt	assert	tran	/I9/I0/I0/MN1	5	118.658p	27.7145p
128		1	Nominal	vds_step_g45p1svt	assert	tran	/I12/I0/MP1	10	5.38962n	27.9269p
129		1	C0_2	vds_step_g45p1svt	assert	tran	/I0/I3/MP1	5	107.8p	29.7763p
130		1	C0_1	vds_step_g45n1svt	assert	tran	/I12/I0/MN1	10	80.3183p	32.0391p
131		1	C0_2	vds_step_g45n1svt	assert	tran	/I12/I1/MN1	10	5.4n	32.1496p
132		1	Nominal	vds_step_g45p1svt	assert	tran	/I9/I0/I0/MP1	5	10.4069n	33.0856p
133		1	C0_1	vds_step_g45p1svt	assert	tran	/I12/I0/MP1	10	5.39146n	33.4303p
134		1	Nominal	vds_step_g45p1svt	assert	tran	/I12/I1/MP1	10	98.7091p	33.7283p
135		1	Nominal	vds_step_g45n1svt	assert	tran	/I12/I1/MN1	10	5.39659n	34.1052p
136		1	C0_3	vds_step_g45n1svt	assert	tran	/I0/I2/NM0	5	88.7518p	35.5347p
137		1	C0_3	vds_step_g45p1svt	assert	tran	/I8/I1/MP1	10	77.596p	37.8148p
138		1	C0_3	vds_step_g45n1svt	assert	tran	/I9/I0/I0/MN1	5	135.46p	37.9925p
139		1	C0_1	vds_step_g45p1svt	assert	tran	/I12/I1/MP1	10	101.136p	39.8697p
140		1	C0_1	vds_step_g45n1svt	assert	tran	/I12/I1/MN1	10	5.3974n	40.71p
141		1	C0_3	vds_step_g45p1svt	assert	tran	/I0/I3/MP1	5	123.015p	41.6864p
142		1	C0_2	vds_step_g45p1svt	assert	tran	/I12/I1/MP1	10	101.915p	44.4943p

If you click the hyperlinks in the *Count* column, a table is displayed at the bottom of the `Checks/Asserts` view that lists all the time points when the violation occurs and other details of the check that differ at each time point.

© 2023 Cadence Design Systems, Inc.

View Summary Violation Details

Selecting the *Summary* check box shows the count for each check or assert violation across all instances for each combination of corner and analysis name.

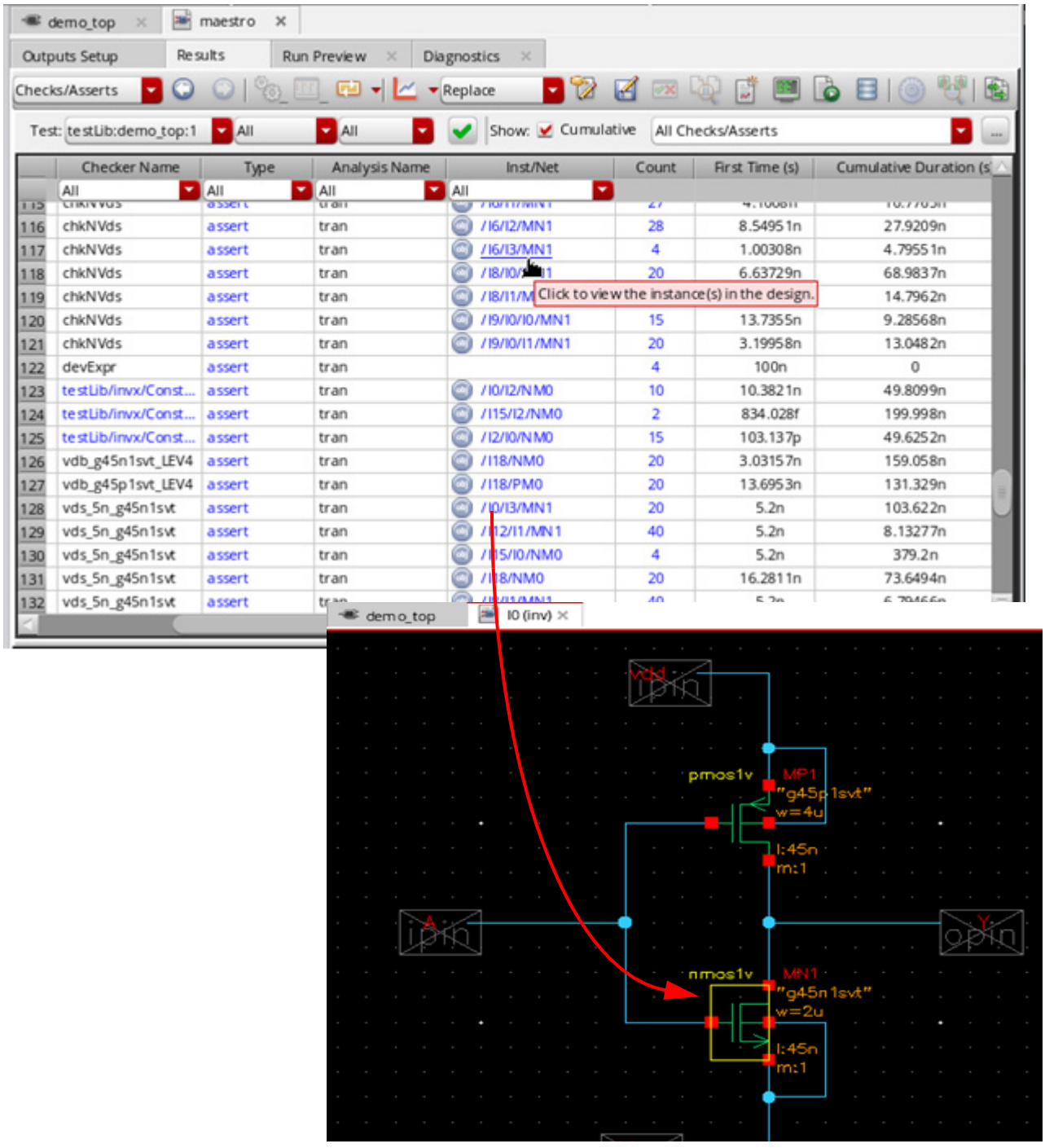


	Point	Corner	Checker Name	Type	Analysis Name	Count
	All	All	Filter	All	All	Filter
1	1	Nominal	testLib/demo_top/AssertionCheck	assert	dcOp	36
2	1	Nominal	testLib/demo_top/AssertionCheck	assert	tran	36
3	1	Nominal	testLib/demo_top/Constr_2	dyn_highz (highz violation)	tran	9

© 2023 Cadence Design Systems, Inc.

View Instances with Violations on Schematic

If you want to view which nets or instances in the schematic view are violating a check, click an instance or a net name in the *Instance* column. The schematic view is opened in a new tab within the ADE Assembler environment, as shown below.



© 2023 Cadence Design Systems, Inc.

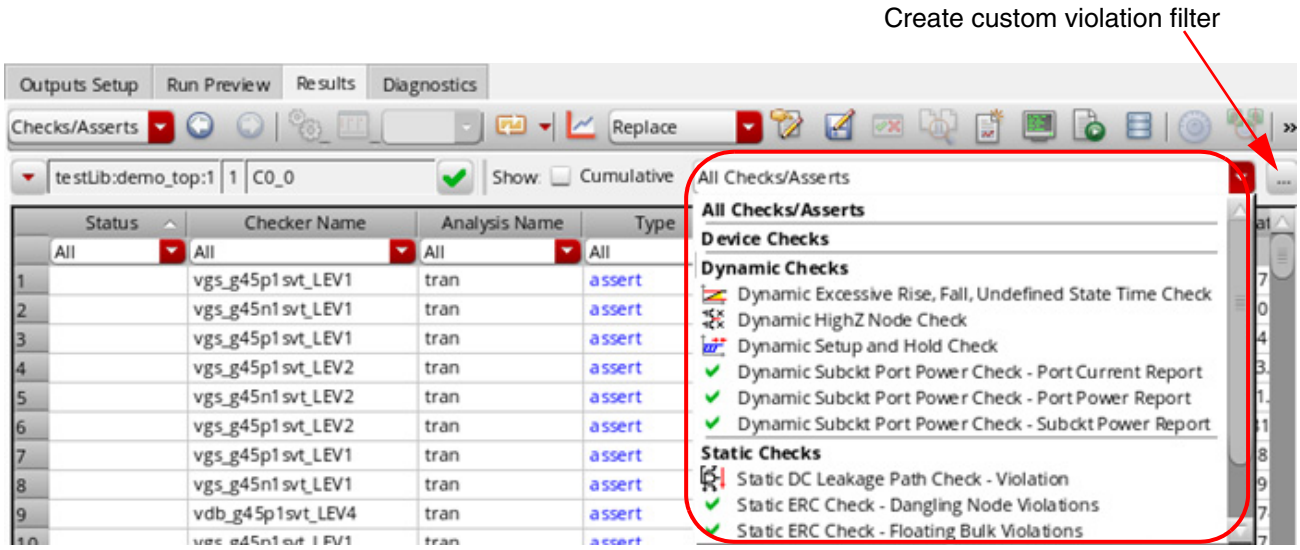
The violating net or instance is highlighted and zoomed into. The zoom scale is controlled by the autoZoomScale environment variable. You can disable the automatic zooming by setting the adexl.gui zoomToProbedInstOrNet environment variable to `nil`.

The schematic view shows the lower-level subcircuit that contains the selected net or instance.

Also see: adexl.gui descendIntoSubcktForShowingInstOrNet

Creating Custom Violation Filter

By default, the drop-down list to select a violation filter shows the three main types of checks, `Device Checks`, `Dynamic Checks`, and `Static Checks`; and the checks for which violations exist are listed under their appropriate types, as shown below.



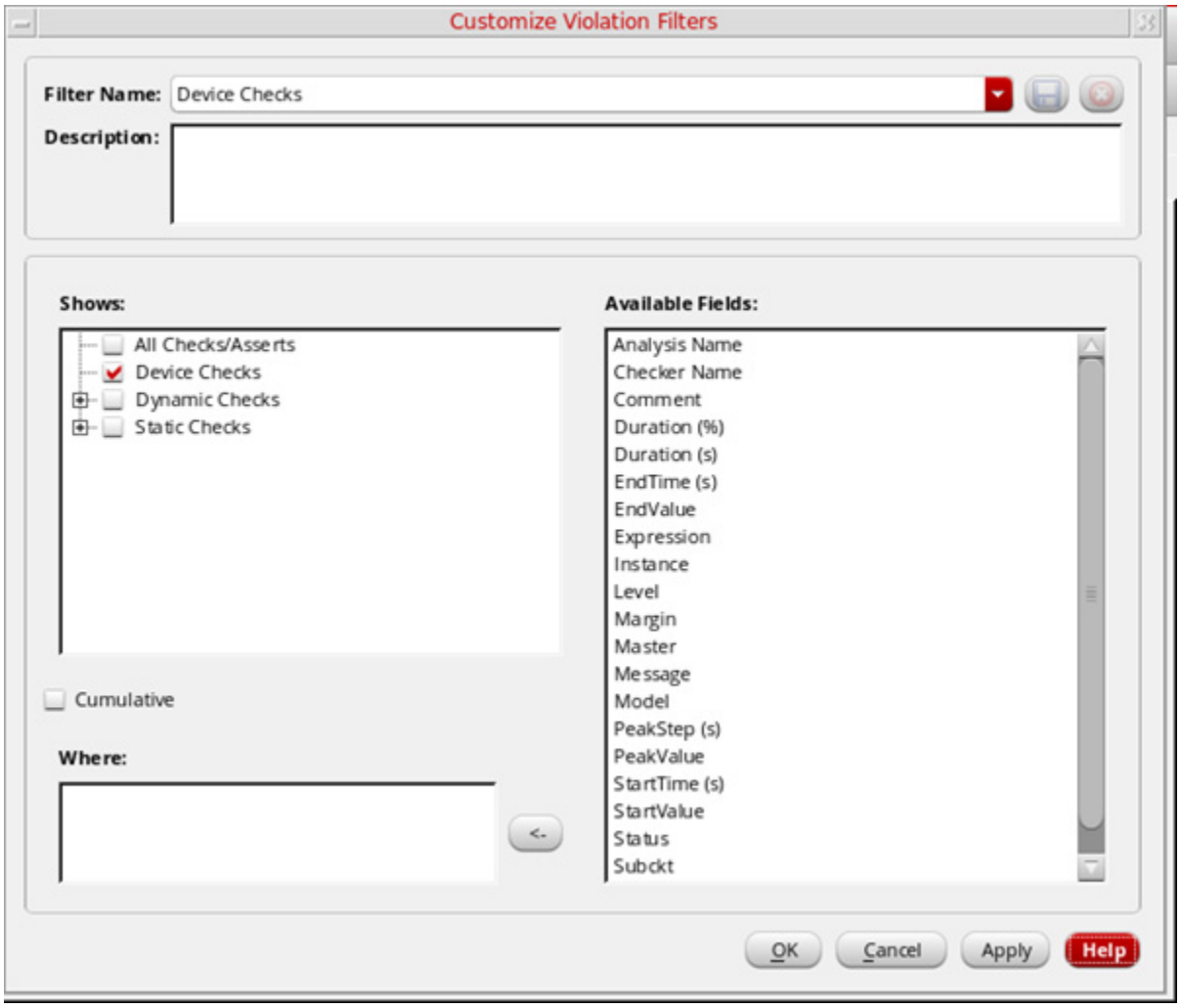
If required, you can create your own custom violation filters with different criteria and save them so that they can be reused for the same or different results.

Note: For custom filters, a new category, `Custom Filters`, is added at the top of this drop-down list. All the custom filters are listed under this category.

To define and save a custom filter, perform the following steps:

© 2023 Cadence Design Systems, Inc.

1. Click *Edit or Create custom violation filter* (...) on the toolbar to open the Customize Violation Filters form.



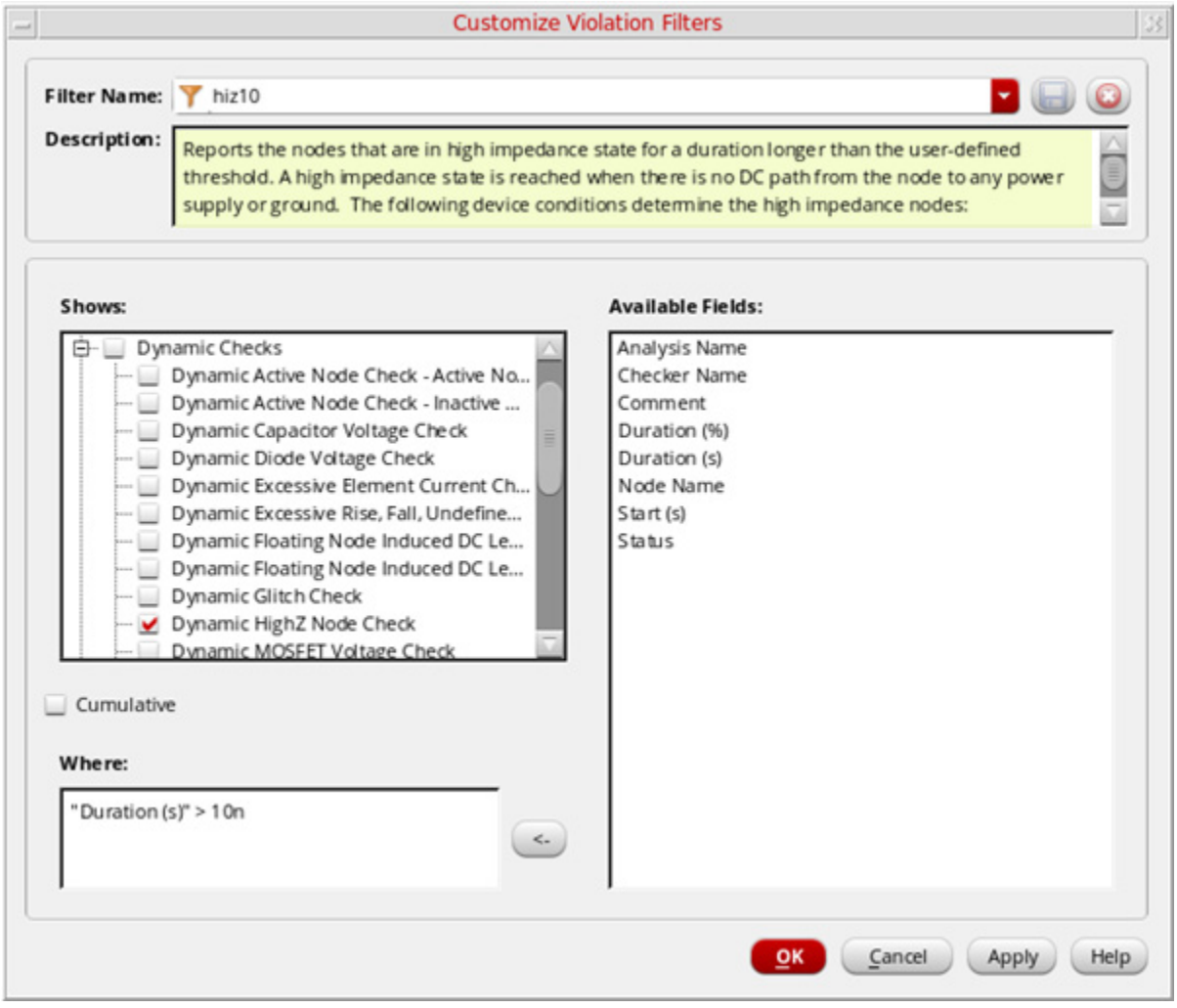
2. Specify values in the following fields to create a custom filter:

Form Field	Description
Filter Name	The name of the filter
Description	The description for the custom filter
Shows	The type of checks to be displayed by this filter.
Available Fields	The list of fields or columns available for the selected check type. This list varies for different types of checks.

© 2023 Cadence Design Systems, Inc.

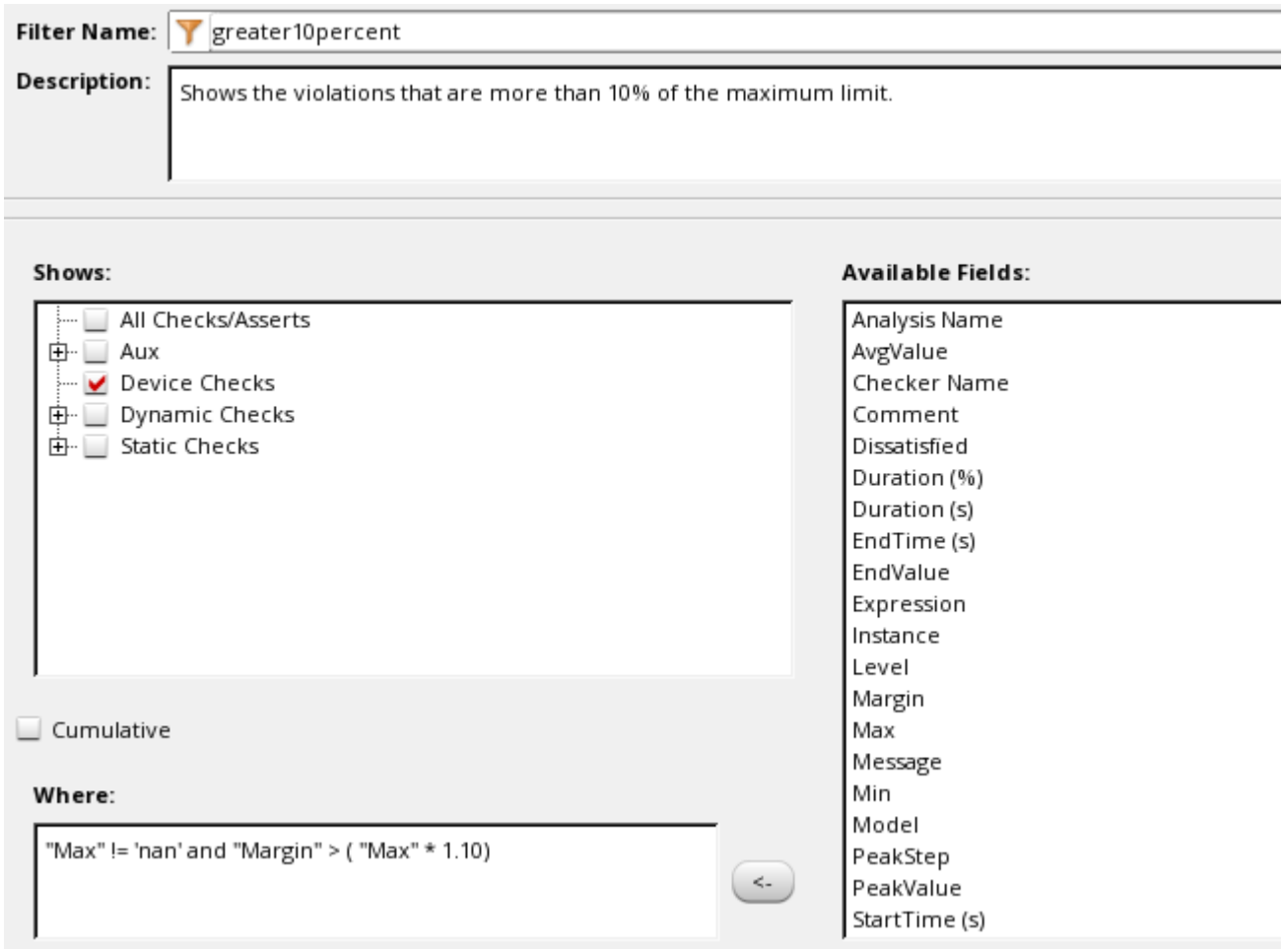
Form Field	Description
Cumulative	Specifies if the filter results should be cumulative.
Where	Specifies the filter criteria. To define this criteria based on the values in a field or column, click the column name in the *Available Field* list and click the left arrow. The field name is added to the *Where* field. You can now build an expression to define the filter criteria.

An example of a custom filter that checks the duration of a violation is shown below.



© 2023 Cadence Design Systems, Inc.

Another example of a custom filter that filters the device checks where violations are more than 10% above the maximum limit is shown below.



3. Click *Save* on this form to save these filter settings in a custom filter.

Note: The custom filters are saved in the `.cadence` directory under the `dfII/maestro/check-filters` subdirectory. You can configure Cadence Setup Search File mechanism (CSF) to specify additional locations from where filters can be read. This allows you to create a library of filters that can be shared amongst multiple users.

Important Points to Consider for Custom Filter Syntax

- Instance and net names must use the simulator namespace, not the schematic namespace. For example, `I8.I0.MP1`is correct, whereas `/I8/I0/MP1` is incorrect.
- Query operators are case insensitive
- Strings can be in single or double quotes

© 2023 Cadence Design Systems, Inc.

- If the field names contain spaces or special characters, they must be enclosed in double quotes.

Query Operators for Custom Filters

Custom filters support all SQLite operators. Some of the commonly used operators are listed in the table given below.

Operator	Definition	Example
`=, ==, !=, <>`	Equal, not equal	`"Inst/Net" = 'I18.NM0'`
`IS, IS NOT, NOT`	Equality operators. Use this to query the Status column	`"Status" IS NOT "Waived"`
`>, <, >=, <=`	Greater than, less than, greater than or equal, less than or equal	`"Duration (s)" > 20n`
`BETWEEN`	Between an inclusive range	`"Time (s)" BETWEEN 10n AND 30n`
`IN`	List of possible values	`"Checker Name" IN ('vgs_nch', 'vgs_pch')`
`LIKE`	Searches for a case insensitive pattern. % is used to define wildcards before and after the pattern.	`"User Message" LIKE '%exceeds%'`
`GLOB, MATCHES`	Case sensitive string matching that uses * and ? wildcards	`"User Message" GLOB '*g45?1svt*VGS*'` `"User Message" MATCHES '*g45?1svt*VGS*'`
`CONTAINS`	Shortcut for `GLOB '*foo*'`	`"User Message" CONTAINS 'exceeds'`
`STARTSWITH, ENDSWITH`	Shortcut for `GLOB 'foo*', GLOB '*foo'`	`"Inst/Net" STARTSWITH 'I18'` `"Inst/Net" ENDSWITH 'NM0'`
`AND, OR`	Combines multiple search expressions	`"Status" IS NOT "Waived" AND "Time (s)"`

© 2023 Cadence Design Systems, Inc.

Viewing Violations in ViVA XL

For violations that occur on nets, ADE Assembler lets you:

- Click the hyperlinks in the *Time(s)* column of the Checks/Asserts results view to plot the time and duration for which a particular result was violating.
- Click the hyperlinks in the *StartTime(s)* column of the results view for device checks to plot the values given in the *Message* column.

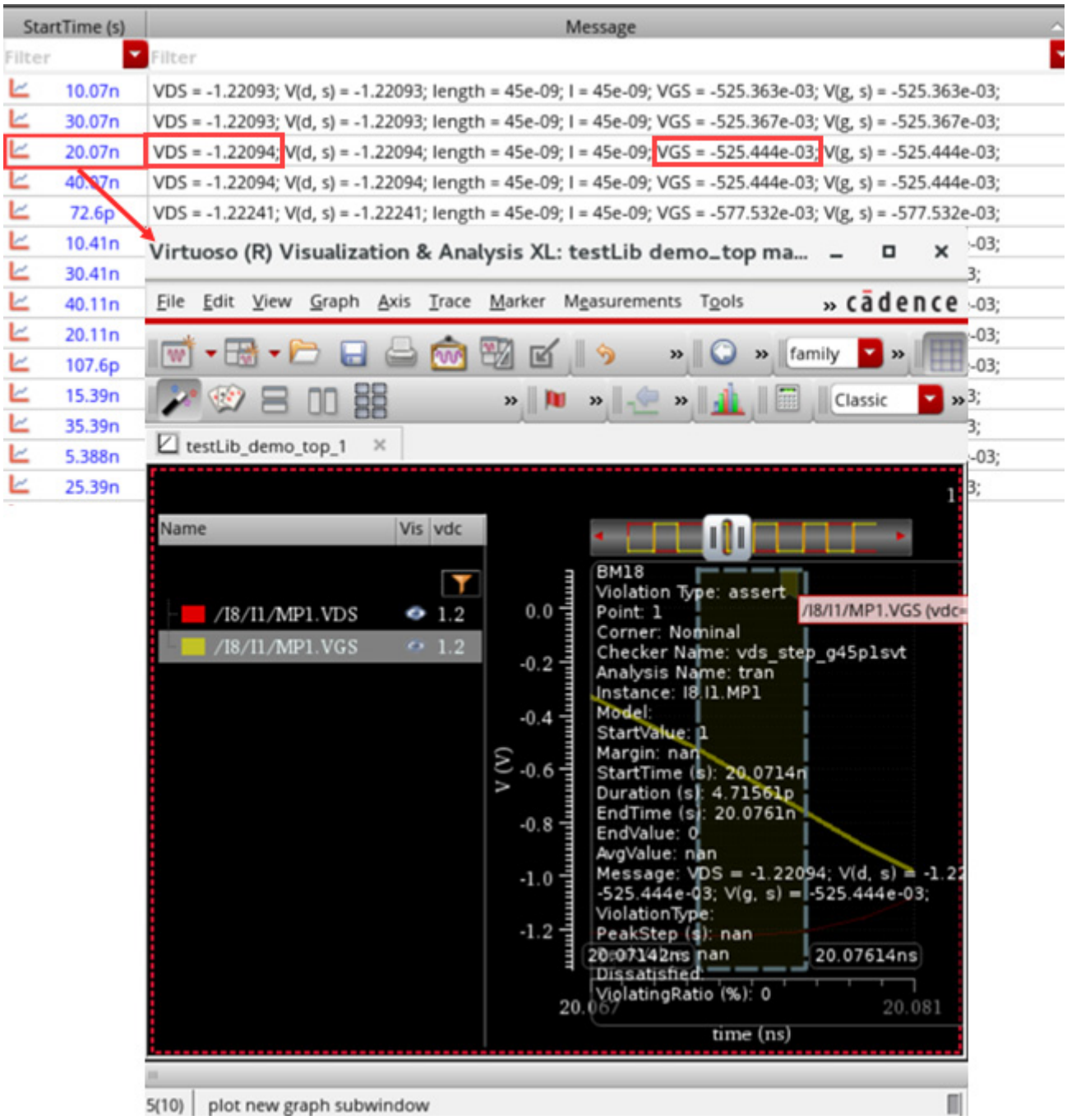
  To enable this feature:

  - Set the option `wfdebug=assert` in the *Additional Arguments* field, in the *Miscellaneous* tab of the Simulator Options form. This also ensures that the signals to be plotted are automatically saved in the outputs setup.
  - Ensure that you use Spectre 21.1 ISR15 or higher release versions.

The transient analysis result for that net is displayed in the Virtuoso Visualization & Analysis XL window. The start and end times for the violation are marked with vertical markers and the

© 2023 Cadence Design Systems, Inc.

violation area is highlighted. You can also see how the given value varies with time. A bookmark also shows the check details, as shown the figure given below.

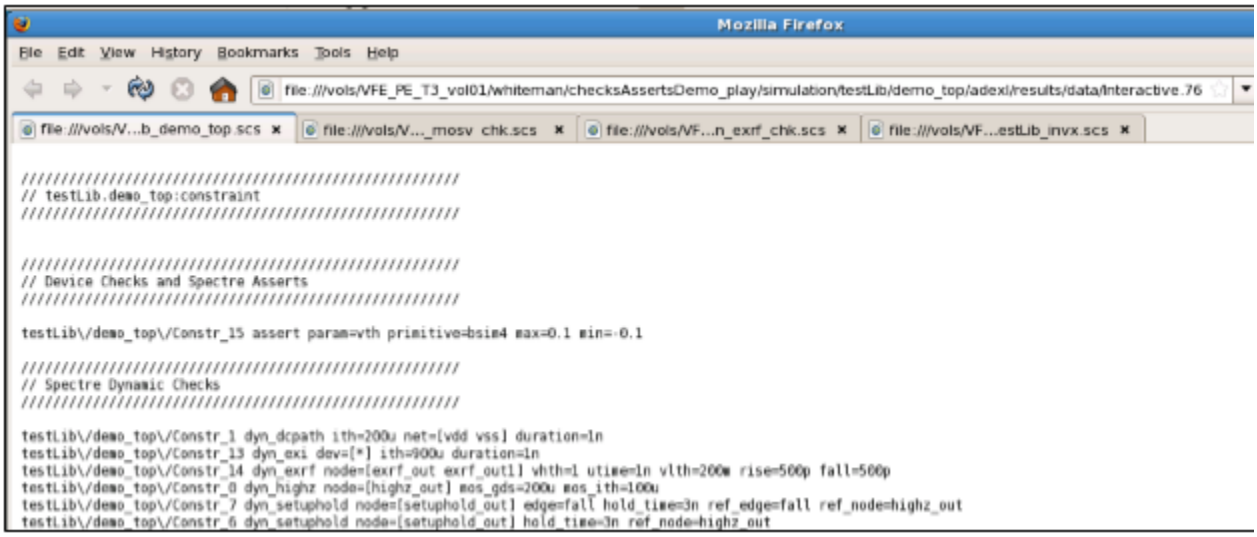


© 2023 Cadence Design Systems, Inc.

View Violations in Browser

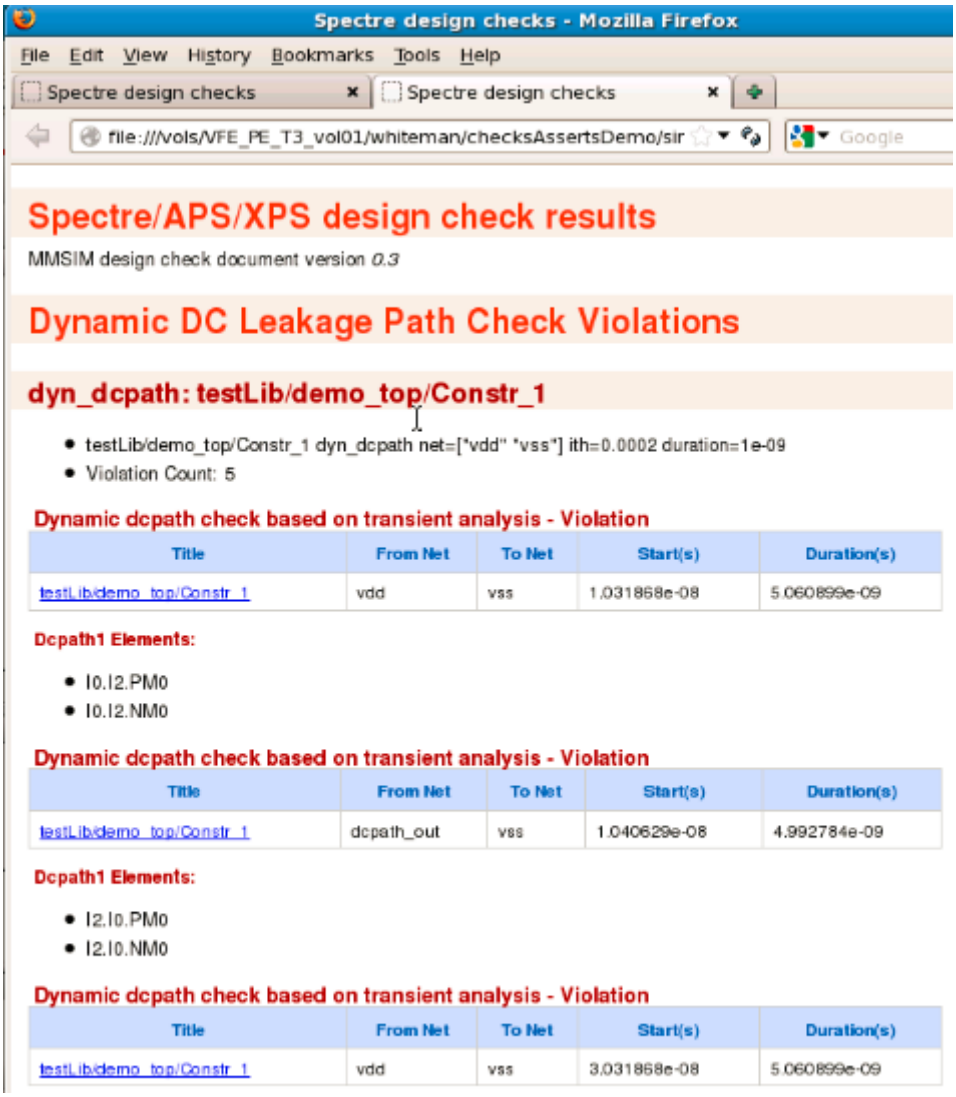
In the `Detail` results view, right-click the result of a violation filter and choose any one of the following commands to view results in a browser:

- *View Netlisted Checks/Asserts*: Opens a browser to display the checks and asserts as they are netlisted. Checks for each cellview are displayed in a separate tab, as shown below.



© 2023 Cadence Design Systems, Inc.

- *View Dynamic/Static Violation Report*: Opens a browser to display the XML results for static and dynamic circuit checks in separate tabs, as shown below.



Note: By default, these reports are displayed in firefox. Set the checksAssertsViewTool environment variable to specify any other browser in which you want to view these reports:

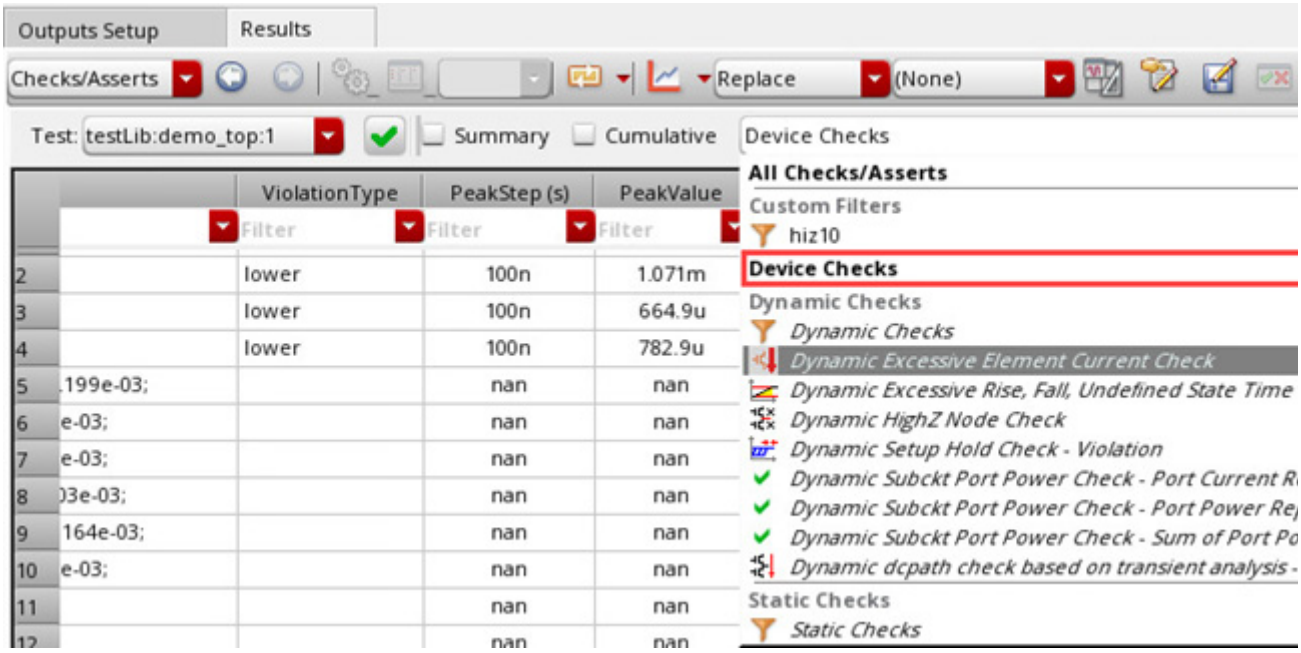
```
envSetVal("adexl.results" "checksAssertsViewTool" 'string "firefox")
```


© 2023 Cadence Design Systems, Inc.

Viewing Only Device Checks

Asserts or device checks are used to perform checks on design parameters, node voltages, element currents, model parameters, operating point parameters, and expressions.

To view the results of the performed device checks, select *Device Checks* from the drop-down menu.



The following fields appear:

Field Name	Description
Status	Waiver status for violations.
Point	Number of sweep points.
Corner	Name of the corner.
Checker Name	Name of the assert.
Analysis Name	Name of the analysis.
Instance	Name of the instance.
Model	Model or subcircuit name related to the device check.
Count	The number of times for which a violation occurred for a particular net or instance.

© 2023 Cadence Design Systems, Inc.

First Time(s)	Start time of the first violation.
Cumulative Duration (s)	Sum total of the time duration of the violations for a given net or instance.
Max Peak Value	Maximum peak value
Max Margin	Margin of the maximum peak value violation.
Max Violating Ratio	Violation ratio of the maximum peak value violation.
Max Peak Time	Start time of the maximum peak value violation.
Max Peak Duration (s)	Time duration of the maximum peak value violation.
Model	Model name of the device.
StartValue	Start value of the violation.
Margin	Difference between the violation start value and the minimum or maximum parameter values.
StartTime (s)	Start time of the violation.
Duration (s)	Duration of the violation.
Duration (%)	Duration percentage of the violation, which is calculated as follows: $\frac{Duration}{Transient\ Analysis\ Length} \times 100$ Here, `Duration` is the duration of the violation and the `Transient Analysis Length` is the stop time specified for the transient analysis.
EndTime (s)	End time of the violation.
EndValue	End value of the violation.
AvgValue	Average value of the violation.
Message	Message that Spectre writes to the log file for the start of the violation.
Max Message	Message that Spectre writes to the log file for the maximum peak value violation.
Max Dissatisfied	The part of a boolean assert expression that was not satisfied for the maximum peak value violation.

© 2023 Cadence Design Systems, Inc.

ViolationType	■ `upper` when maximum value is violated. ■ `lower` when minimum value is violated.
PeakStep (s)	Time step at peak value.
PeakValue	Peak value. **Note:** In an assert check of Boolean type, the peak value is not printed by default. Use `extreme=yes` in the assert statement to enable printing of peak value in this column.
Dissatisfied	The part of a boolean assert expression that was not satisfied.
ViolatingRatio	Percentage of violation detected above or below the specified minimum or maximum value in the checker definition. $ViolatingRatio\ (\%) = \frac{PeakValue - Max\ or\ Min}{Max\ or\ Min} \times 100$ To view this field, set the following Spectre option: `sqldb_violating_ratio=yes` **Note:** This field can also be viewed on selecting a dynamic mosv violation filter.
Min	Minimum value in the checker definition.
Max	Maximum value in the checker definition.
Expression	MDL expression that is checked.
Level	Severity level of the message if the check fails. It can be one of the following values: ■ `notice` or `warning`: The simulation continues after the message is displayed. ■ `error`: Spectre aborts the analysis when the first error-level violation occurs. ■ `fatal`: Spectre aborts the simulation when the first fatal violation occurs. Default value: `warning`
User Message	Message extracted from the `message=` parameter. Any hypertext links given in this field are displayed as hyperlinks.

© 2023 Cadence Design Systems, Inc.

Comment	Comments added to the *Waive Violation(s)* form.

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

14

Running Fault Simulations

As a part of the Legato™ Reliability Solution, fault simulations in ADE Assembler provides the capability to run defect-oriented tests on analog IC designs. These tests allow you to evaluate the ability to eliminate a die with manufacturing defects and resulting test escapes that cause field failures. It can also be used to optimize wafer test, reducing the number of tests required to achieve the target defect coverage by eliminating over-testing and potentially reducing the number of tests.

A typical fault analysis consists of fault injection, fault analysis, and test optimization for better fault coverage. Analog fault analysis provides the capability to estimate fault coverage by generating a detection matrix. In addition, it can be used as part of functional safety analysis.

Some faults are more likely to occur than others; therefore, a test that detects the more likely faults will have a higher coverage than the one that detects the less likely faults. You can calculate the defect coverage using the likelihood-weighted sum of the detected faults divided by the likelihood-weighted sum of all simulated faults.

The defect detection matrix can be obtained by comparing the faulty simulation results with nominal (faultless) data. You can further optimize the test to improve the defect coverage based on the defect detection matrix.

In functional safety applications, fault detection is tracked for both functional and diagnostic (checker) modules but reported separately. Functional safety report can be generated based on assert violations of fault analysis results.

The following sections in this chapter provides details on how to use ADE Assembler for these purposes:

- Prerequisites
- Preparing the Setup for Fault Simulation
- Injecting Faults in a Design
- Creating Fault Groups
- Previewing Faults in a Fault Group

© 2023 Cadence Design Systems, Inc.

- Setting up Run Options for a Fault Simulation
- Running a Fault Simulation
- Viewing Results of Fault Simulation
- Automatic Fault Dropping Using a Run Plan
- Merging Results from Multiple Fault Simulations
- Running Functional Safety Checks

Video

You can view at Performing Fault Simulation in Virtuoso ADE Assembler using Legato Reliability Solution.

© 2023 Cadence Design Systems, Inc.

Prerequisites

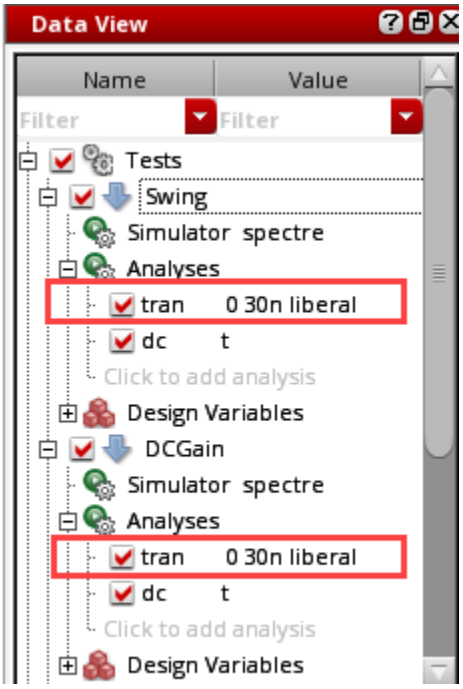
Ensure that you have set the path to the required simulator as indicated by the table given below.

Design	Analysis Type	Simulator
`Analog`	Direct Fault Analysis	Spectre 19.1 ISR7 or higher versions
	Transient Fault Analysis	
`Mixed-Signal`	Direct Fault Analysis	AMS Flex flow with a combination of Xcelium 19.03 s006 and Spectre 19.1 ISR7, or higher versions
	Transient Fault Analysis	

Preparing the Setup for Fault Simulation

Before running a fault simulation, do the following in the ADE Assembler setup:

- **Ensure that the transient analysis is enabled** for the tests in ADE Assembler.

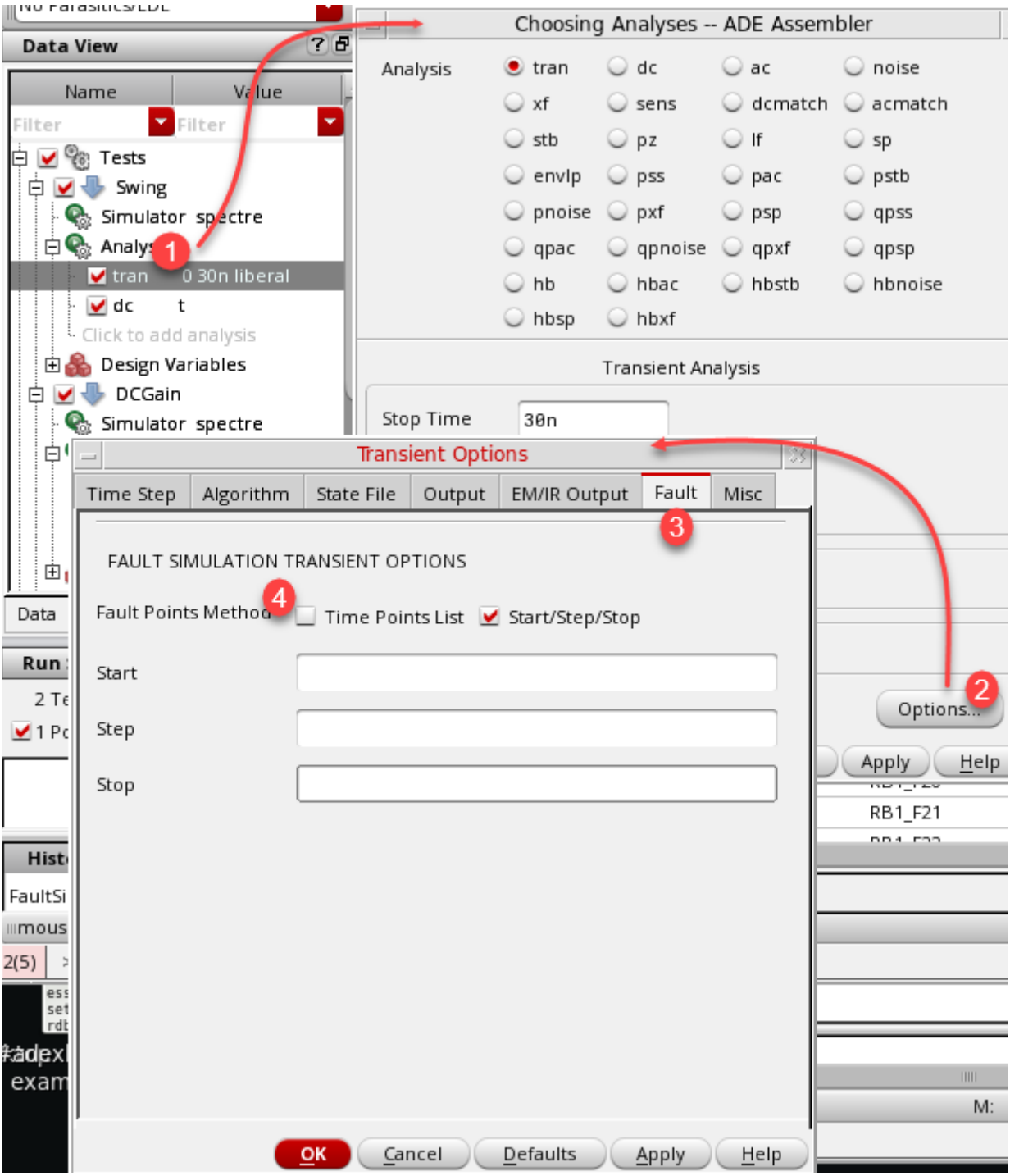


- **Specify time steps for the transient analysis**.

© 2023 Cadence Design Systems, Inc.

On the Choosing Analysis form, click *Options* for the *tran* analysis.

The Transient Options form is displayed. On the *Fault* tab of this form, specify the fault points as a list of specific time points or as a sweep set with start-step-stop time point values.



© 2023 Cadence Design Systems, Inc.

The fault time points specified in this form are applied to the test for which you are editing the analysis options. If you do not specify fault times at the test level, the values specified on the run options form for Fault Simulation are used.

- **Create fault expressions in the Outputs Setup tab** of the maestro view by setting the output type of an expression as `faultexpr`.

  If required, you can change the type of expressions from `faultexpr` to `expr` at any time.

  Note: Expressions identified as fault expressions are not restricted to be evaluated only for the results without faults. These expressions can be evaluated for any result with or without fault.

- **Set the run mode as Fault Simulation**.

- **Set up the faults to be checked using the Fault Setup assistant**. For more details, refer to Setting up Faults.

Setting up Faults

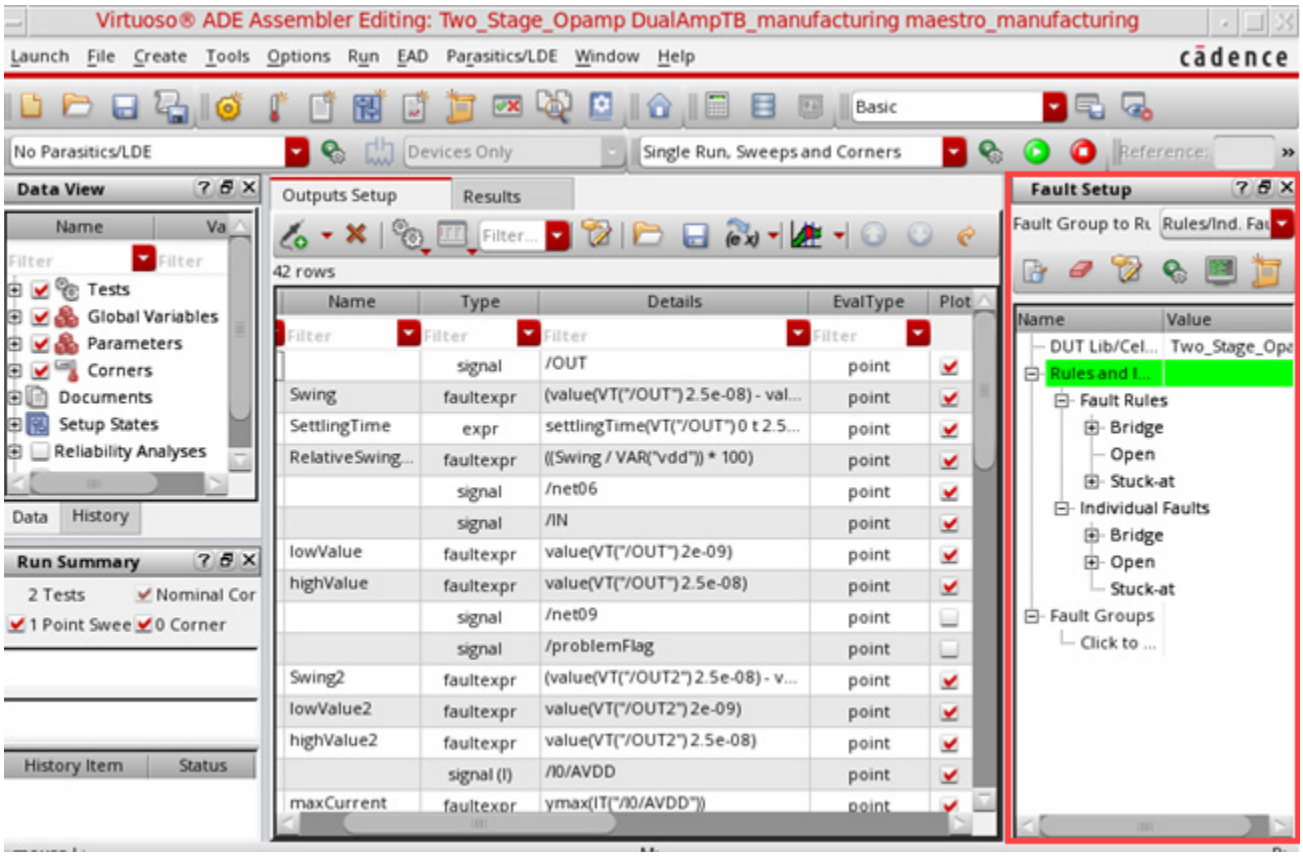
You can set up faults by using the Fault Setup assistant in ADE Assembler.

To open the fault setup, do any one of the following:

- Select the `Fault Simulation` workspace from the Workspace Configuration drop-down list.
- Choose *Window — Assistants — Fault Setup*.

© 2023 Cadence Design Systems, Inc.

The *Fault Setup* Assistant appears to the right of the *Outputs Setup* pane.



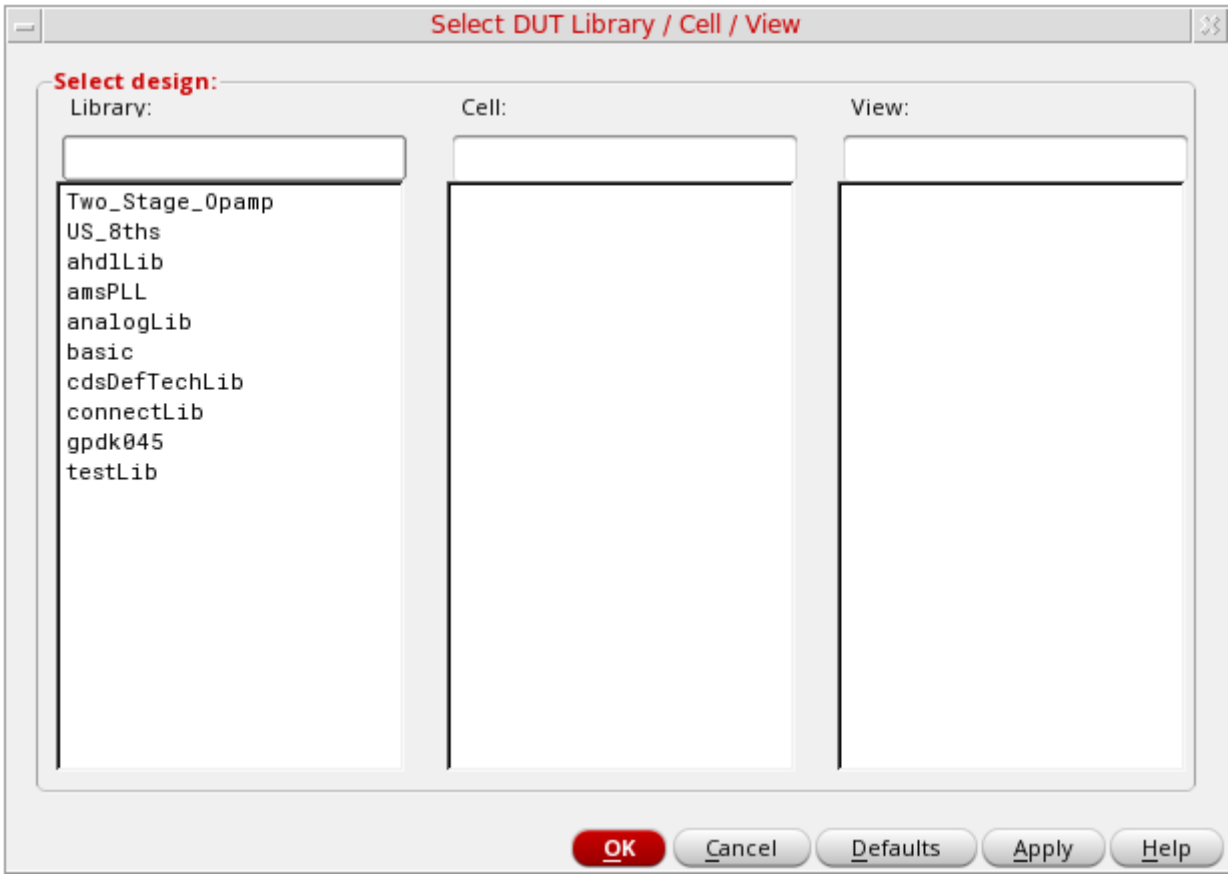
To prepare the setup in the Fault Setup assistant:

1. Specify the DUT library, cell, and view for which you plan to run fault analysis simulation.

   To do this, in the Fault Setup assistant, double-click the value cell for *DUT Library/Cell/View*.

© 2023 Cadence Design Systems, Inc.

The Select DUT Library / Cell / View form is displayed.



Select the library, cell, and view to which this fault setup has to be applied.

Important Points to Note:

- Other fields on the assistant remain disabled until you select the DUT.
- Ensure that the selected DUT is not a top-level design.
- Ensure that the selected DUT is used by all tests enabled for the simulation run.
- If you change the DUT later after configuring the faults, you can retraverse the design to reflect the changes in the Fault Setup assistant.
- To simplify the fault simulation setup, ADE Assembler assumes that multiple tests in a maestro view are used only for one DUT. Even if you use multiple tests for simulating the design, they share the fault setup and the results also show a combined view for all tests. If you want to work on different DUTs, it is recommended to create multiple maestro cellviews.

© 2023 Cadence Design Systems, Inc.

- If your test setup contains config sweeps, all swept cellviews use the common fault list created for the design specified in the *DUT Library/Cell/View* field of the Fault Setup assistant. The fault list is created using the base configuration, which includes the design setup assuming no config sweep is defined. You need to ensure that the swept cellviews use the same device under test so that the fault list remains accurate for all runs. Depending on the changes in the config sweep, you might find that some faults are not simulated for a swept cellview. For example, if you replace a block by a behavioral block, in that case, all faults inside that block become invalid.

2. Ensure that the local variables for the enabled test do not contain sweep values. Local sweeps are currently not supported by fault simulation.
3. Specify faults to be injected into the design, which is the DUT selected in **step 1**.

   There are three ways in which you can specify the faults to be injected in the DUT:

   - Using Fault Rules
   - Creating Individual Faults
   - Using Fault Files

4. (Optional) Create fault groups to group the faults that you would like to use together in a simulation.

   Note: By default, all rules and individual faults defined in the *Rules and Individual Faults* tree in the Fault Setup assistant are used. To run any other group in the simulation run, select the name of that group from the *Fault Group to Run* drop-down list on the top of the Fault Setup assistant.

Note: The rules format has changed from IC 6.1.8 ISR8. If your maestro cellview contains fault rules created before this release, remove those manually and recreate new faults.

© 2023 Cadence Design Systems, Inc.

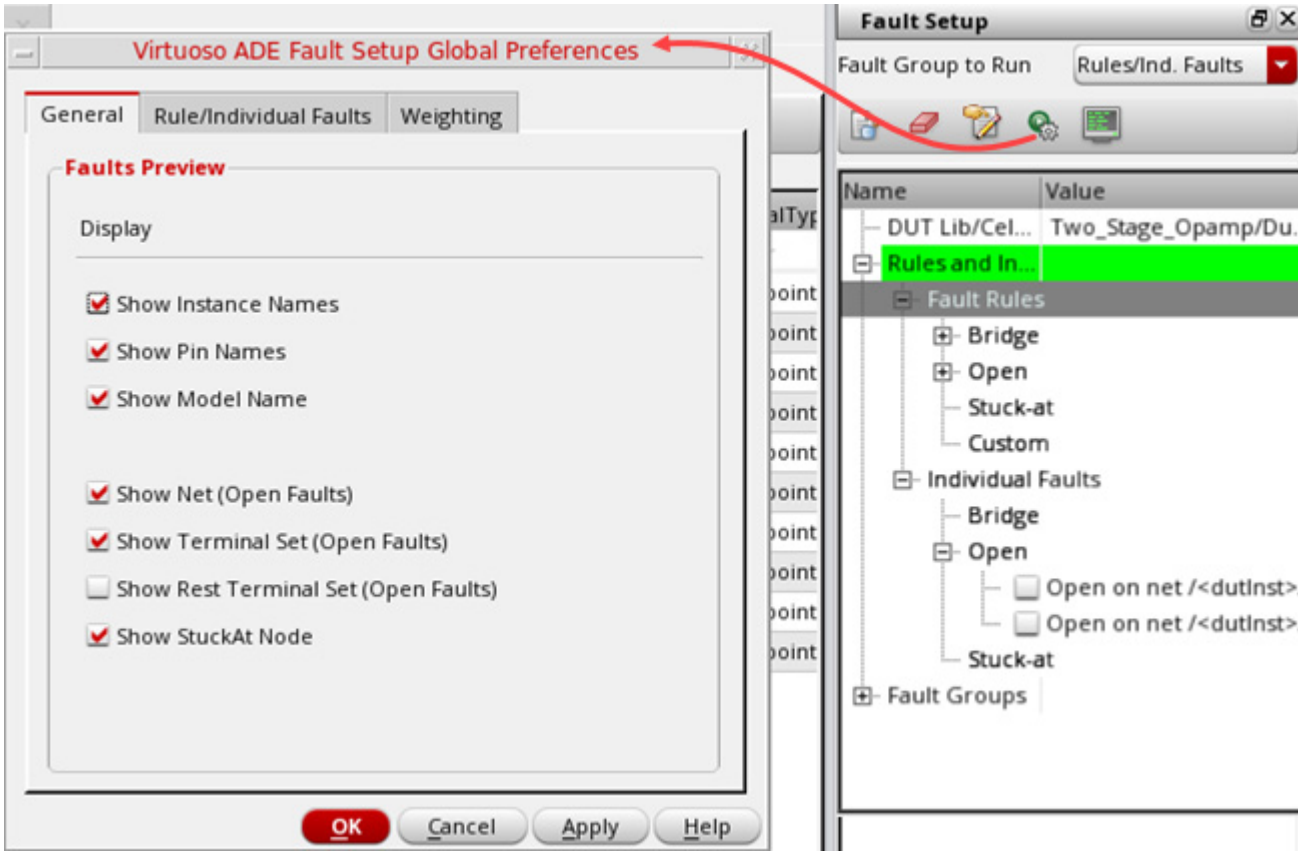
Specifying Global Preferences for Fault Setup

Before defining a fault to be injected, verify that the global or default options to be used for different types of faults have the required values. These global preferences can be specified on the Virtuoso ADE Fault Setup Global Preferences form.

To open the Virtuoso ADE Fault Setup Global Preferences form:

➡ Click ( ) on the toolbar of the Fault Setup assistant.

The Virtuoso ADE Fault Setup Global Preferences form is displayed.



Configure your preferences on the different tabs of this form:

- *General*: Provides options to configure general display options for fault preview that is displayed on the Fault Rules form.
- *Rule/Individual Faults*: Provides options to the default resistance value for each type of fault, and the names of the ground and supply net to which a fault will be stuck.

© 2023 Cadence Design Systems, Inc.

- *Weighting*: Provides options to specify the following weight options for fault rules. Spectre calculates the weighted likelihood for each fault based on its ratio to the total weights within the block. This tab contains the following options:
  - *Default Weight Expression*: An expression to define the default fault weighting function to be used for a fault rule. This is the default value that is displayed in the *Weight Expression* field on the Fault Rule Setup form.

    The default value is 1, which indicates that all faults have the same weight.
  - *Default Weight Factor*: A value that specifies the likelihood for all faults in a given fault block. This is the default value that is displayed in the *Weight Factor* field on the Fault Rule Setup form.

    The default value is 1.
  - *Default Extra options*: Any other Spectre option to specify the weight of a fault block.
  - *Individual Fault–Default Weights*: The default weight to be assigned to an individual fault.
  - *Apply Fault Collapsing to*: Specifies the level at which fault collapsing can be enabled. You can choose one of the following:
    - *All fault rules*: Enables or disables fault collapsing globally for all fault rules.
    - *Rule Specific*: Allows you to control fault collapsing at rule level. This enables the *Enable Fault Collapse* check box on the Fault Rules form.
    - *None*: Disables fault collapsing globally for all fault rules.

© 2023 Cadence Design Systems, Inc.

Injecting Faults in a Design

You can inject either `Bridge`, `Open` or `Stuck-at` faults into the design. The Fault Setup Assistant provides the following methods to specify the faults to be injected:

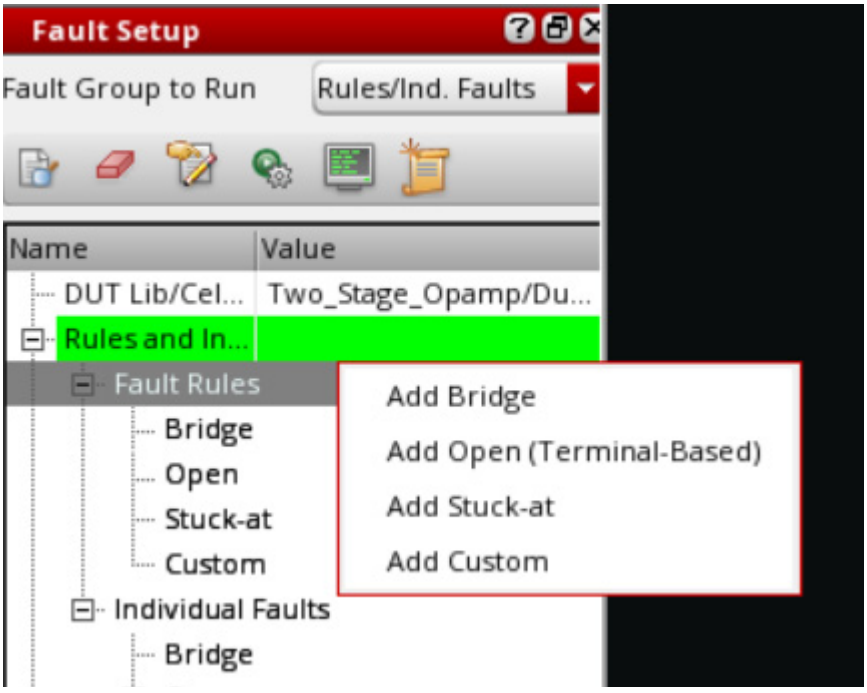
- Using Fault Rules
- Creating Individual Faults
- Using Fault Files
- Importing Fault Rules from Other Cellviews

Using Fault Rules

Fault rules provide a quick way to create faults for multiple instances or nets that meet the given criteria. To define the rules for fault creation, use the Fault Rules form.

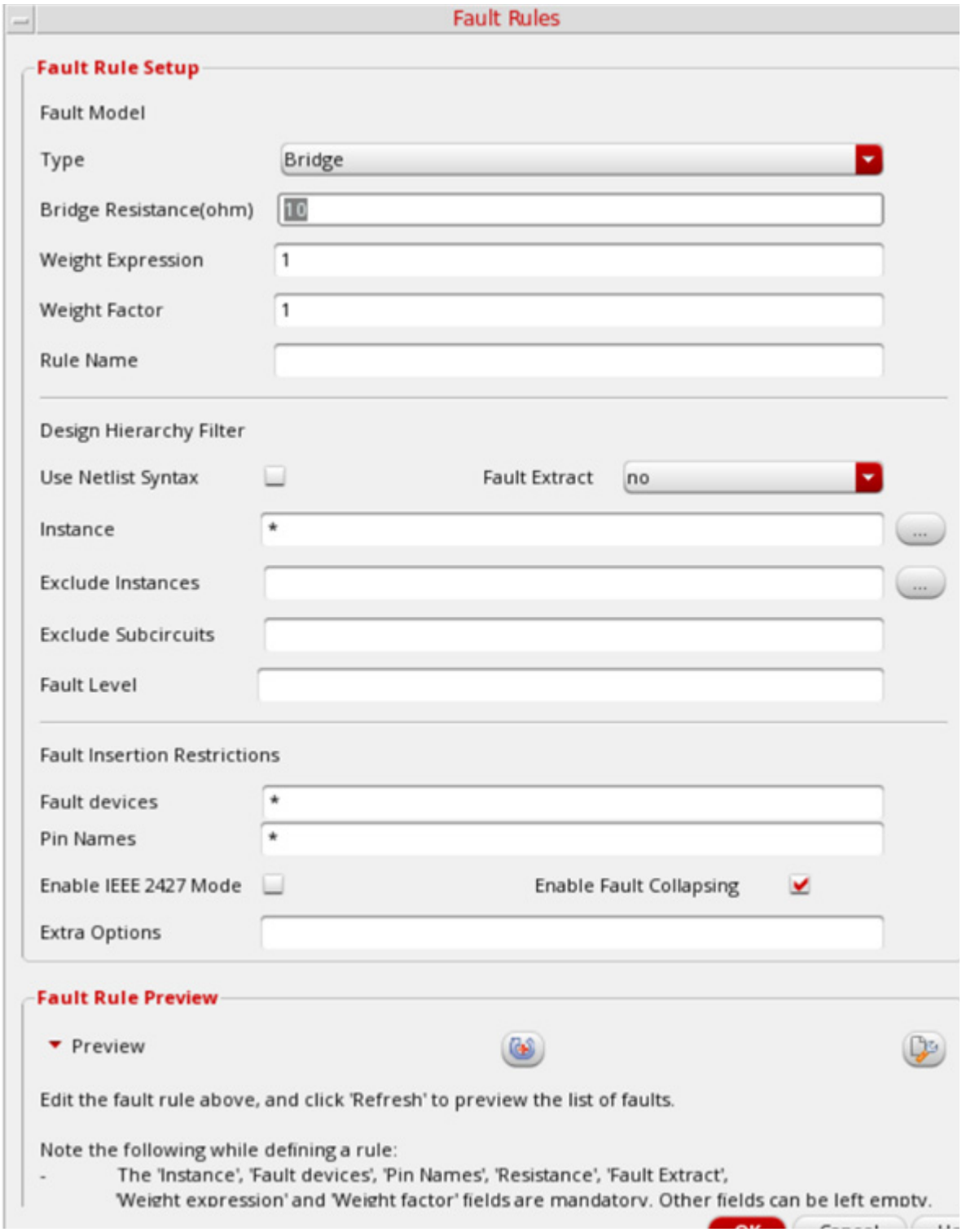
To create faults based on fault rules, perform the following steps:

1. In the Fault Setup assistant, expand the *Rules and individual faults* tree.
2. Right-click *Fault Rules,* and choose the *Add Bridge*, *Add Open*, *Add Stuck-at*, or *Add Custom* command depending on the type of fault for which you are defining the rule.



© 2023 Cadence Design Systems, Inc.

The Fault Rules form is displayed.



© 2023 Cadence Design Systems, Inc.

By default, the rule is defined by using the * wildcard in the *Instances* and *Pin Names* fields.

3. Modify the values in the *Fault Rule Setup* section to define the rules or criteria for fault creation. For example, you can specify the names of instances as `/I3 /I3/I4`, and pin names as `D S`.

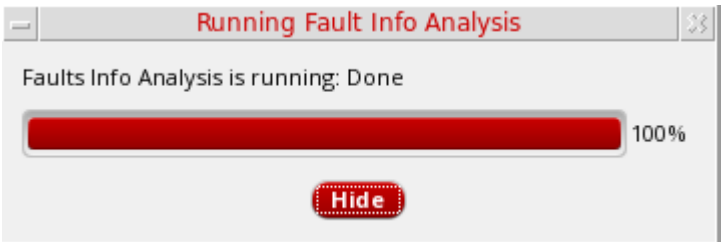
   See Fault Rules Form for mode information.

Tip

The values displayed in the *Weight Expression* and *Weight Factor* fields are taken from the Weight tab of the Virtuoso ADE Fault Setup Global Preferences form

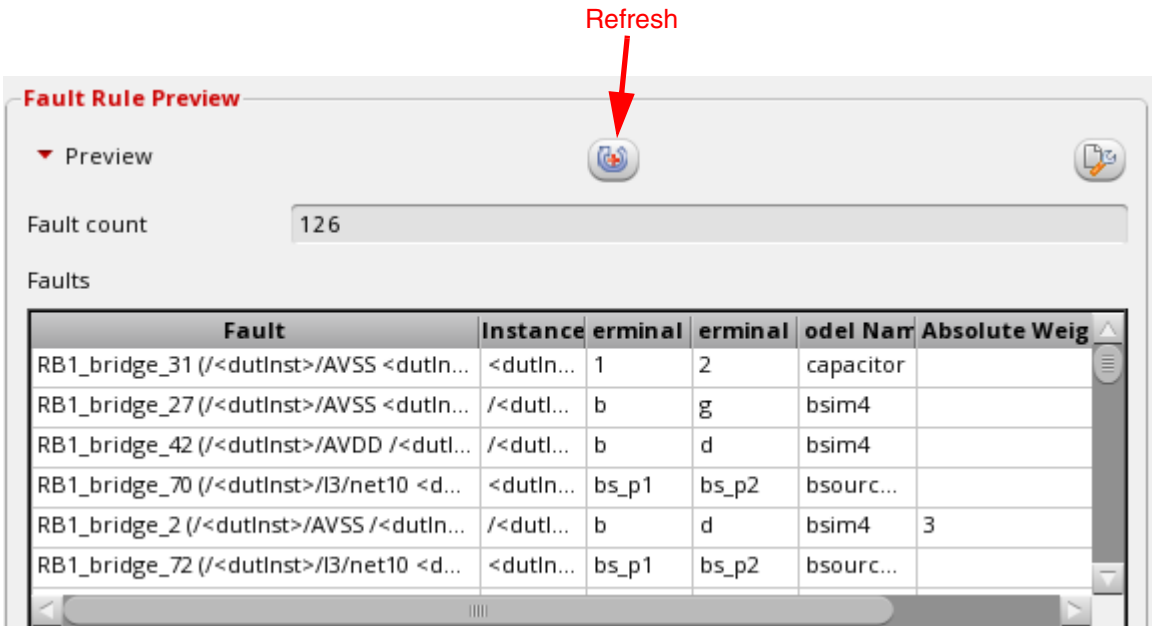
4. In the *Fault Rule Preview* section, click *Refresh* to view the list of possible faults that can be created using the rule defined in this form.

   When you click *Refresh*, ADE Assembler runs Spectre info analysis that generates a fault list based on the instances, devices, and pins specified by the criteria defined by the given rules. A progress bar indicates the completion status of the info analysis.



© 2023 Cadence Design Systems, Inc.

When the info analysis is complete, the resulting list of faults is shown in the *Faults* table at the bottom of this form. The total count of faults is displayed in the *Fault Count* field.

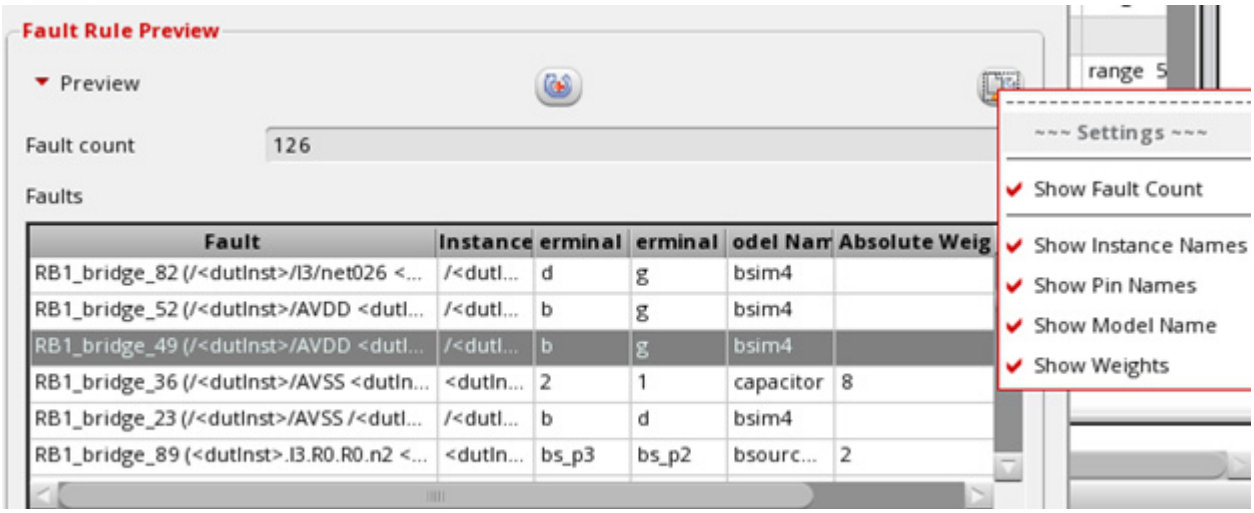


Note: While identifying all faults based on the specified rule, Spectre info analysis uses fault collapsing or the weight expression to calculate the absolute fault, which is the likelihood count of a fault. This absolute weight value is displayed in the *Absolute Weight* field in the preview table. ADE Assembler uses this value later for fault normalization while generating fault preview before simulation.

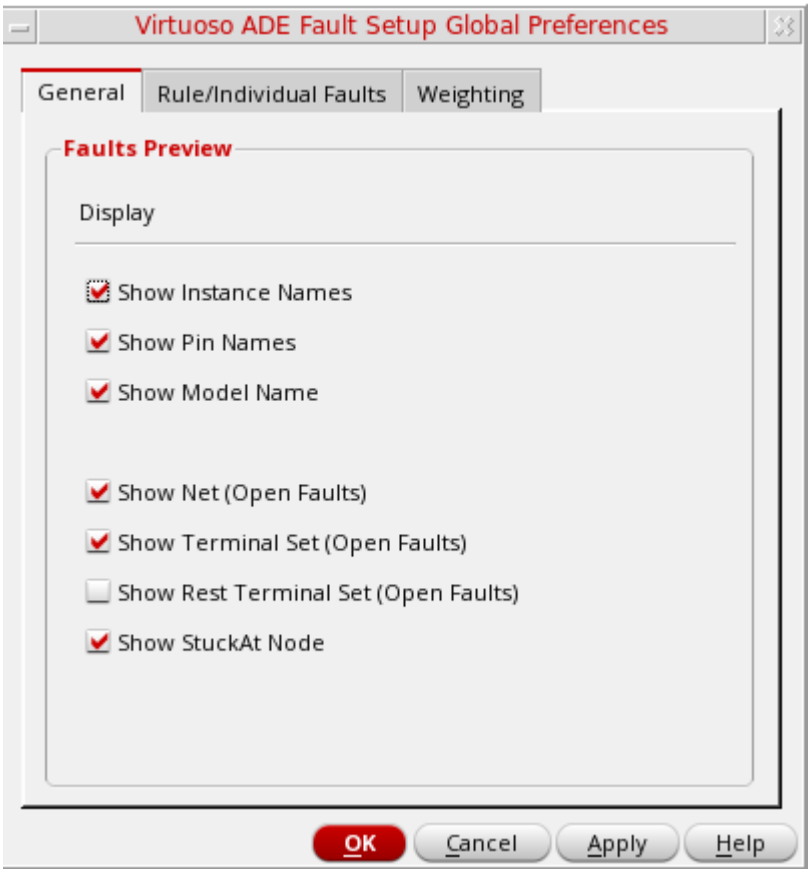
5. Review the list of faults. If required, edit the rules again and click *Refresh* ( ) in the *Preview* section to review the list of possible faults.

© 2023 Cadence Design Systems, Inc.

You can configure the type of information to be displayed in the *Faults* table by clicking *Configure preview settings*, and selecting or deselecting an option.



Note: You can also configure the columns to be displayed by using the options on the *General* tab of the Virtuoso ADE Fault Setup Global Preferences form shown below.



© 2023 Cadence Design Systems, Inc.

6. When the fault rule setup and the resulting faults are final, click *OK* to close the form.

   The rule is added to the `<fault-type>` tree within the *Rules and individual faults* tree.

Tip

A rule can be edited or deleted later by using the *Edit* and *Delete* commands from the context-sensitive menu of that rule. To delete multiple faults at a time, press the `Ctrl` key and select the faults to be deleted, right-click and choose *Delete*.

© 2023 Cadence Design Systems, Inc.

Fault Rules Form

Form Field	Description
Type	Type of fault to be injected. Either `Bridge`, `Open (Terminal-Based)`, `Stuck-at`, or `Custom`.
Bridge Resistance (ohm)	Resistance between two nodes of a bridge fault. Default value: `100`
Weight Expression	An expression that defines the likelihood of faults for a block during fault generation. The likelihood of a fault occurrence can be based on the circuit element area, proximity of the adjacent interconnect, length of the interconnect, or other factors. In ADE Assembler, you can define a weight expression using device parameters. For example, `w*l`. Default value: `1` When the parameters are not specified or contain multiple types of devices or models, weight expressions may not be evaluated for some of the models. In such situations, the expression can result into 0. To avoid this, you can use the `fmin` function in weight expressions. Example: `fmin(w*l, 1e-12)` In this example, `fmin` prevents the weight from becoming zero if the fault instance does not have parameters `w` or `l`. If `w*l` cannot be evaluated, `1e-12` will be assigned as the weight value. Similarly, `fmax` limits the values of weight expressions. For example, `fmax(0.1, 0.001*r)`. In this example, `fmax` clamps the maximum resistance which is used as weighting function. For information about fault weighting and its usefulness in the calculation of coverage report, see Weighted Likelihood Calculation in *Spectre Classic Simulator, Spectre APS, Spectre X, and Spectre XPS User Guide*.

© 2023 Cadence Design Systems, Inc.

Weight Factor	A value that defines the weighting factor for a given block or rule. Weight factor is used for the normalization of faults across fault bocks or rules when faults in a particular fault block have very high or very low likelihood as compared to the faults in other fault blocks. Default value: `1`
Rule Name	Name to be assigned to the fault rule. This rule name prefixed with the default prefix for each rule type is shown in the rule summary shown in the Fault Setup assistant. For example: `RB_myRule(/I4 * where * on * with res = 100)` Default value: ""
Open Resistance (ohm)	Resistance between two nodes of an open fault. Default value: `10k`
Stuck At Net	Name of the net to be used to short the terminals of the specified devices. This field is available only for stuck-at faults.
Custom Type	Type of custom fault to be inserted. These custom faults can be available in text files or in schematics. Possible values are: ■ *Custom Insert*: Generates custom faults of fault subcircuit to be inserted at specific terminals of the given devices. For example, a custom fault of subcircuit to be inserted at the drain, gate, and source terminals of bsim4 devices. ■ *Custom Open*: Generates custom open faults of fault subcircuit to be inserted at specific terminals of the given devices. For example, a custom open fault of subcircuit to be inserted at the drain, gate, and source terminals of bsim4 devices. ■ *Custom Replace*: Generates custom faults of fault subcircuit to replace specific devices. For example, a custom fault of subcircuit to replace bsim4 devices.
Fault Custom	Name of the subcircuit to be used to insert custom faults.
Design Hierarchy Filter	This section contains settings that are used to filter devices on which faults are to be injected.

© 2023 Cadence Design Systems, Inc.

Use Netlist Syntax

Specifies whether to use the netlist format for instance names. For example, `I0.I2`

By default, this check box is not selected and you can specify the instance names in the schematic syntax.

When using the netlist syntax, you can specify only one instance in the *Instance* field. When using the schematic syntax, you can specify multiple instance names.

Fault Extract

Specifies where to inject layout-based faults.

Possible Values:

- `spf`: Generates a fault list that is compatible with the post-layout netlist.
- `sch`: Generates a fault list that is compatible with the schematic netlist.
- `no`: Layout-based fault list is not generated. This is the default value.

If you choose `spf` or `sch`, ensure that you have specified at least one DSPF file in the Simulation Files Setup form for the test.

Note: When this field is set to sch or spf, an additional field *Fault Layer* is displayed in the *Design Hierarchy Filter* section of this form.

Instance

Name of an instance below which you want to inject the faults. Either specify the name of an instance or use the `*` wildcard to inject faults for all instances in the design hierarchy for which the other rules specified on this form are also met.

When the *Use Netlist Syntax* check box is deselected, you can select an instance from the schematic by clicking (...) to the right of this field. You can also use the wildcards to specify instance names in netlist syntax.

© 2023 Cadence Design Systems, Inc.

Exclude Instances
A space-separated list of instance paths that you want to exclude while injecting faults in the given design hierarchy.

You cannot use the `*` wildcard to specify multiple instance paths.

When the *Use Netlist Syntax* check box is deselected, you can select one or more instances from the schematic by clicking (...) to the right of this field. You can also use the wildcards to specify instance names in netlist syntax.

Exclude Subcircuits
A space-separated list of subcircuit names that you want to exclude while injecting faults in the given design hierarchy.

You can use all wildcards in subcircuit names.

Minimum Capacitance
Specifies the minimum capacitance an extracted parasitic capacitor must have to be identified as a short defect.

Note: This field is visible only when the *Type* field is set to `Bridge` or `Stuck-at` and *Fault Extract* is `sch` or `spf`.

Minimum Resistance
Specifies the minimum resistance an extracted segment resistor must have to be identified as an open defect.

Note: This field is visible only when the *Type* field is set to `Open` and *Fault Extract* is `sch` or `spf`.

Fault Level
Defines the hierarchical level of the design to be considered during the fault list generation. By default, Spectre generates faults in subcircuits at all levels of the design.

Fault Layer
Specifies the layer number of the layers on which layout-based faults are to be identified.

Possible values are:

- `0`: Identifies faults on all layers including the connectivity between layers.
- `[x y ...]`:Identifies faults on the specified layers
- `[(x,y)]`: Identifies faults for all the layers in the given range excluding the connectivity between layers in both cases.

Note: This field is visible only when the *Fault Extract* field is set to `sch` or `spf`.

Fault Insertion Restrictions

© 2023 Cadence Design Systems, Inc.

	This section contains settings that define restrictions to be applied while faults.
Fault Devices	A space-separated list of fault devices on which you want to inject faults. You can use the * wildcard to specify multiple device names matching the given criteria. By default the field is set to `*`, which means that the faults are to be injected on all the devices in the specified design hierarchy. It is mandatory to provide a value in this field.
Pin Names	A space-separated list of pin names on which you want to inject faults. If you want to inject faults on all pins of the specified instances, specify `*` in this field. It is mandatory to provide a value in this field. **Note:** This field is not available when custom faults with type *Custom Replace* are inserted.
Enable IEEE 2427 Mode	Enables the IEEE 2427 mode for Spectre `info` analysis.
Enable Fault Collapse	Enables fault collapsing to remove equivalent faults. When you enable fault collapsing, Spectre collapses each set of equivalent faults within a fault rule into one and assigns an absolute weight, which is a summation of the absolute weight assigned to each equivalent fault collapsed into it. For more details on fault collapsing, refer to Fault Collapsing and Weighting Function. This check box is enabled only when the *Apply fault collapsing to* setting on the Virtuoso ADE Fault Setup Global Preferences form is set to *Rule Specific*. That means you can enable or disable fault collapsing at a rule level only if allowed at the global level.
Extra Options	Additional options to be used by the Spectre `info` analysis that is run to inject faults. Ensure that the options are written in a format supported by Spectre.

© 2023 Cadence Design Systems, Inc.

Creating Individual Faults

If you want to create a fault with unique settings, instead of using fault rules, you can create individual faults of type bridge, open, or stuck-at.

- Creating Individual Bridge Faults
- Creating Individual Open Faults
- Creating Individual Stuck-at Faults
- Creating Custom Faults

Creating Individual Bridge Faults

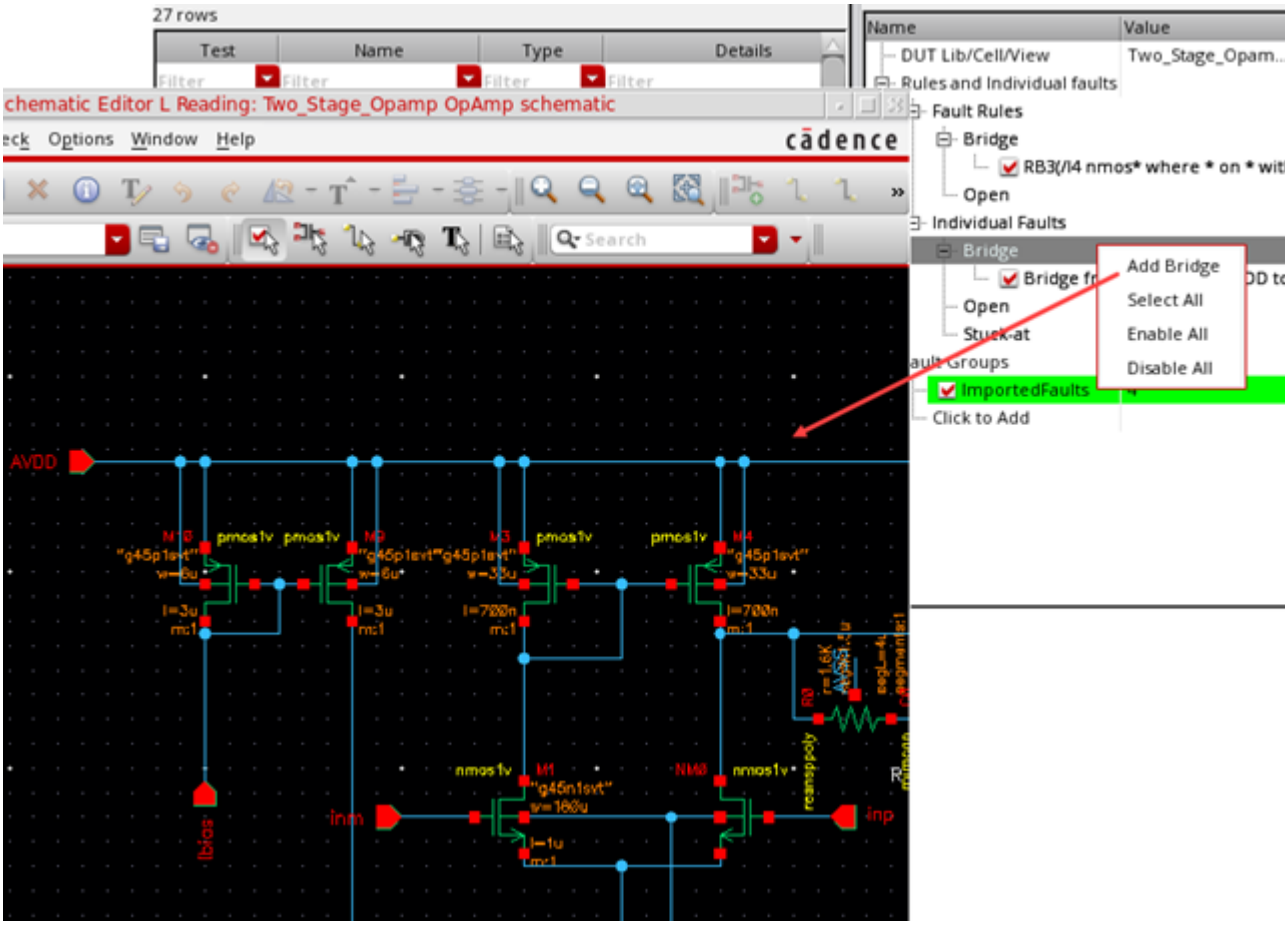
To create an individual bridge fault:

1. Expand *Individual Faults* in the Fault Setup assistant.
2. Right-click *Bridge* and choose *Add Bridge*.

   Alternatively, right-click *Individual Faults* and choose *Add Bridge*.

© 2023 Cadence Design Systems, Inc.

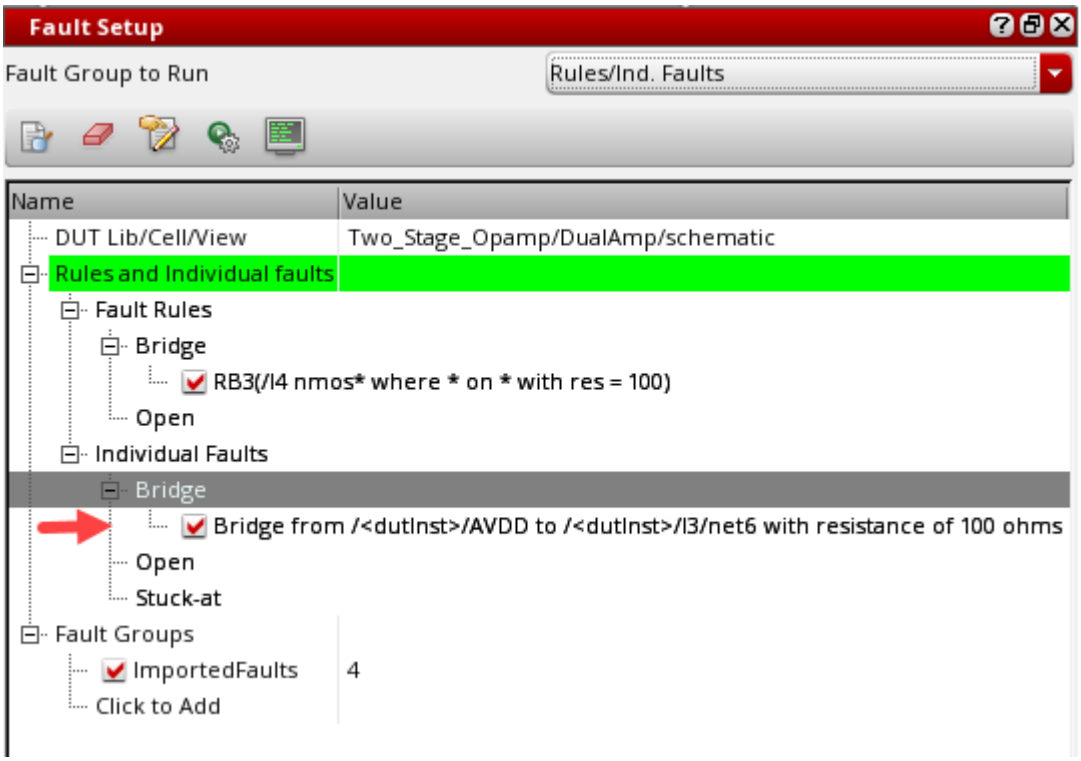
The schematic of the design is displayed.



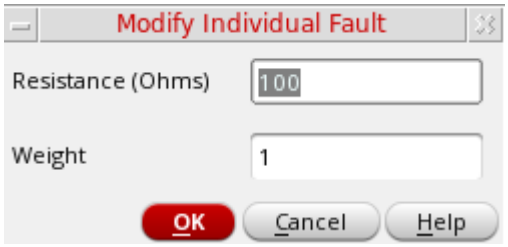
3. Select two nets for the bridge.

   A bridge fault is defined to be added between the selected nets.

© 2023 Cadence Design Systems, Inc.



Note: By default, the fault uses the default resistance value and weight factor specified on the Virtuoso ADE Fault Setup Global Preferences form. You can modify the resistance value for a fault rule by using the *Modify Individual Fault* command in the context-sensitive menu of a fault. This command opens the Modify Individual Fault form that you can use to modify the resistance value or the weight of the selected fault.



Creating Individual Open Faults

To create an individual open fault:

1. Expand *Individual Faults* in the Fault Setup assistant.
2. Right-click *Open* and choose *Add Open*.

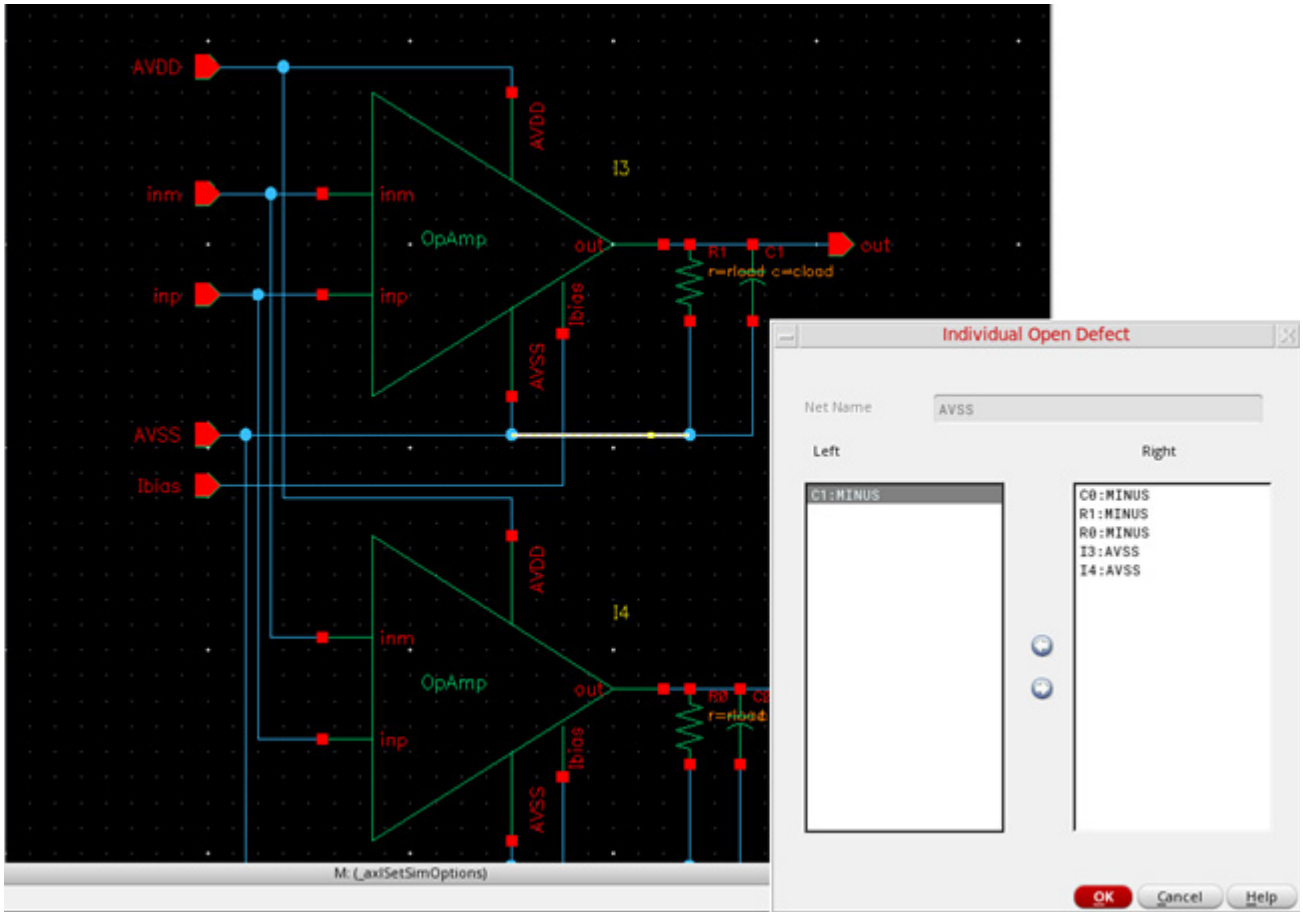
© 2023 Cadence Design Systems, Inc.

Alternatively, right-click *Individual Faults* and choose *Add Open*.

The schematic of the design is displayed.

3. Click a net on the schematic.

   The Individual Open Defect form is displayed. This form displays a list of pins through which the selected net is connected to the instances in the design.



4. Using the arrows, move the terminals to the *Left* or *Right* list to define the placement of the open fault.

   The fault will be injected in such a way that the terminals listed in the *Left* list are on the left and the terminals listed in the *Right* list are on the right of the open resistor is injected on the selected net. The Individual Open Defect form lets you customize the cut set of the net.

© 2023 Cadence Design Systems, Inc.

Creating Individual Stuck-at Faults

To create an individual stuck-at fault:

1. Expand *Individual Faults* in the Fault Setup assistant.
2. Right-click *Stuck-at* and choose *Add Open*.

   Alternatively, right-click *Individual Faults* and choose *Add Stuck-at*.

   The schematic of the design is displayed.
3. Click a net that you want to use for the stuck-at fault.

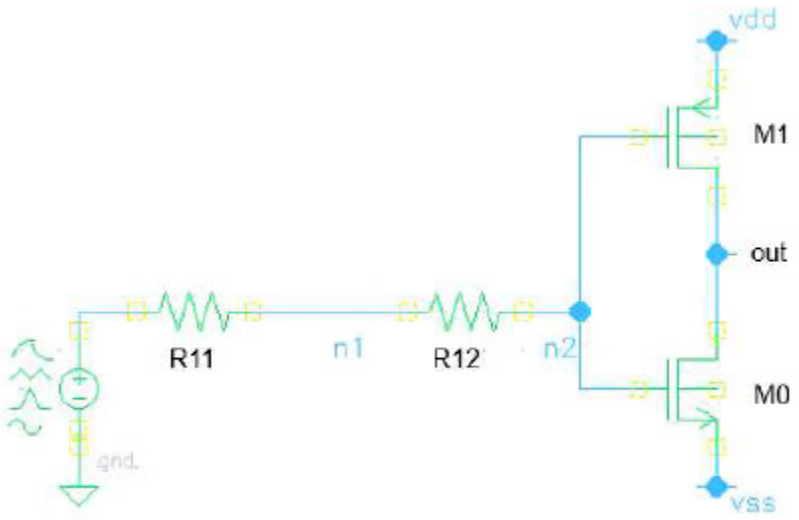
   A new fault is created by using the selected net. The resistance value used for this fault is taken from the Virtuoso ADE Fault Setup Global Preferences form.

© 2023 Cadence Design Systems, Inc.

Creating Custom Faults

You can create the following types of custom faults:

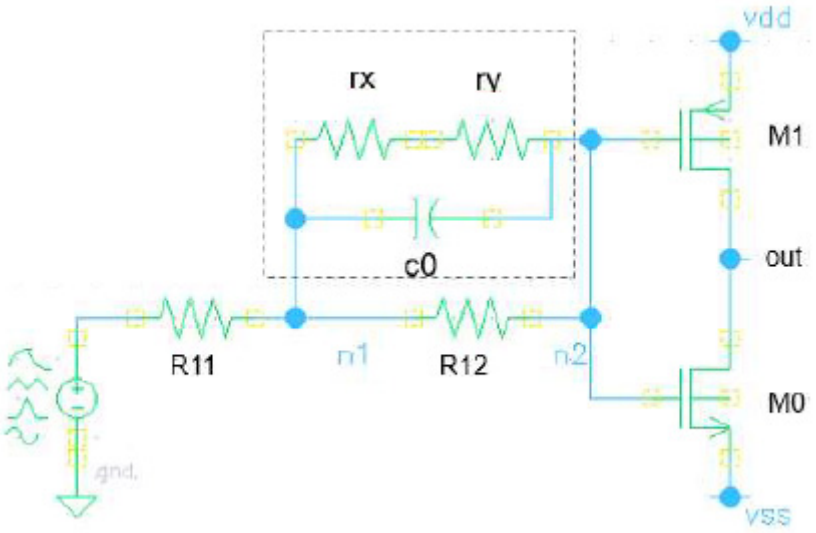
- Custom insertion: In this type, a customized subcircuit instance is connected to the nodes specified in the fault setup. The following example shows that a fault instance is inserted into the nominal circuit by not connecting `n1` and `n2`.



- Custom replacement: In this type, a specific device or subcircuit instance is replaced by a custom subcircuit. The order and number of terminals should match in the two.

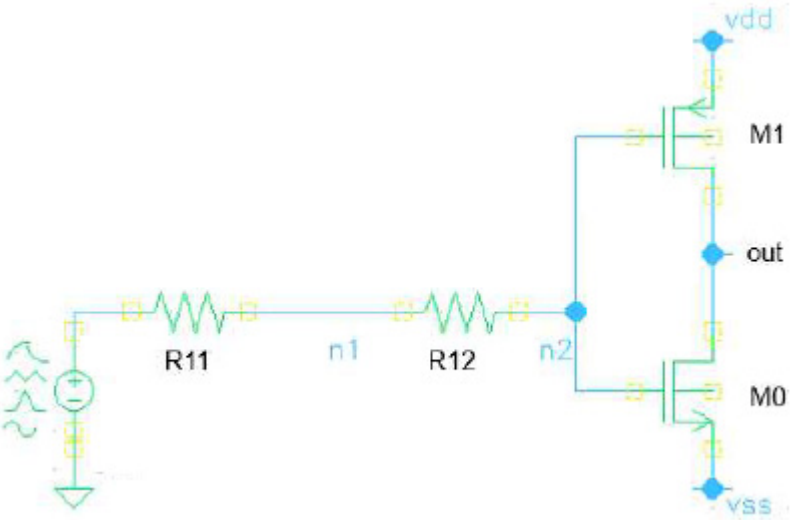
© 2023 Cadence Design Systems, Inc.

In the following example, an instance is replaced by the highlighted fault subcircuit.



- Custom open: In this type, a node is split into two nodes and a custom subcircuit fault is inserted. The custom open subcircuit must have two terminals. While inserting the custom subcircuit, the first terminal of the custom subcircuit is connected to the original node while the second terminal is connected to the new node.

  In the following example, an instance with two terminals is inserted between the original node `n1` and the new node `X`.



© 2023 Cadence Design Systems, Inc.

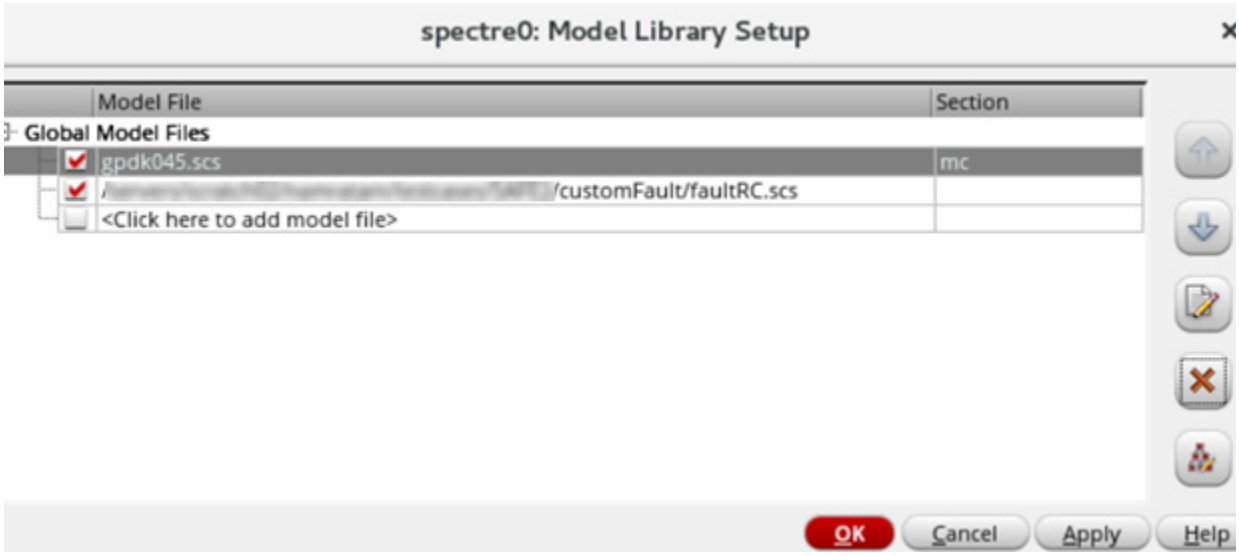
Subcircuits for Custom Faults

You can specify subcircuits to be used in custom faults in the following two ways:

- In a model file, as shown in the following example:

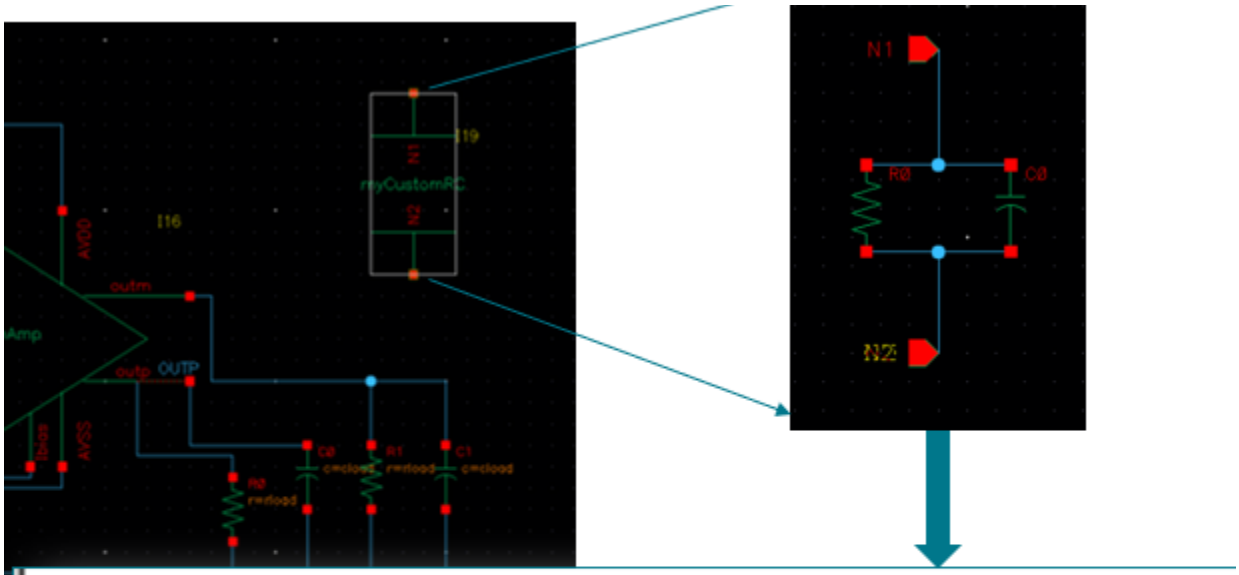
```
subckt faultRC n1 n2
r1 (n1 n2) resistor r=100
c1 (n1 n2) capacitor c=100f
ends faultRC
```

  To use the custom faults from model files, you can add those model files in the Model Libraries Setup form.



© 2023 Cadence Design Systems, Inc.

- A subcircuit in the schematic as shown in the following example:




```
// Library name: Two_Stage_Opamp
// Cell name: myCustomRC
// View name: schematic
// Inherited view list: spectre cmos_sch cmos.sch schematic veriloga
subckt myCustomRC N1 N2
    R0 (N1 N2) resistor r=1K
    C0 (N1 N2) capacitor c=1p
ends myCustomRC
// End of subcircuit definition.
```

Fault Setup for Custom Faults

Similar to the other types of faults, specify fault injection rules on the Fault Rules form. The following form fields are specific to custom faults:

- *Custom Type*: Type of the custom fault.
- *Fault Custom*: Name of the subcircuit to be used to insert custom faults.

Related Topics

The Fault Rules form

© 2023 Cadence Design Systems, Inc.

Using Fault Files

You can import the faults defined in a text file directly into the fault setup. All faults imported from the fault file are added to a group in the setup. For more details, refer to Importing Faults from a File into a Group.

Importing Fault Rules from Other Cellviews

If the fault rules defined in another cellview are relevant for the testbench in the current cellview, you can import the rules from those cellviews by using the Import Setup form.

If the destination cellview does not contain any fault rule, the complete set of fault rules are imported from the source cellview. However, if the destination cellview contains fault rules, the value of the *Operation* field on the Import Setup form defines how the rules from the source and destination cellviews are used.

The following rules describe the import mechanism.

Value	Rules
Retain	■ If the names of imported rules do not match the rules already defined in the destination cellview, they are copied. ■ If the names of imported rules conflict with the rules already defined in the destination cellview, the rules from the source are not copied.
Merge	■ All fault rules from the source cellview are copied to the destination cellview. ■ If the names of imported rules conflict with the rules already defined in the destination cellview, the imported rules are used. The existing rules are overwritten. ■ All unique existing fault rules are retained in the destination.
Overwrite	■ All fault rules are removed from the destination cellview. ■ All fault rules from the source cellview are copied to the destination cellview. ■ The fault rules in the destination cellview and the source cellview are the same.

© 2023 Cadence Design Systems, Inc.

Setting Up Fault Rules for Layout-based Faults

Layout-based fault rules are useful when you want to inject faults that consider parasitics from DSPF files. To create rules for layout-based faults:

1. In the Simulation Files Setup form, specify the paths to the DSPF files extracted from the layout.
2. In the Fault Rules form, choose a fault type in the *Type* field.
3. Enter specifications for the fault rule in the *Fault Model* section.
4. From the *Fault Extract* drop-down list, select one of the following values:
   - `sch`: Injects faults that are compatible with the schematic-based netlist
   - `spf`: Injects faults that are compatible with the layout-based netlist

   Depending on the fault type, a new field is displayed in the *Design Hierarchy Filter* section. When the *Type* field is set to `Bridge` or `Stuck-at`, the field is named *Minimum Capacitance*. When the *Type* field is set to `Open`, the field is named *Minimum Resistance*.
5. Enter a valid value in the *Minimum Capacitance* or *Minimum Resistance* field.
6. Complete the rule setup and close the form.

   A new fault rule is created and added to the Fault Setup assistant.

If required, create more fault rules for other fault types. If you want to use the rules injected by multiple rules in a simulation, create a new group and add the fault rules to that. The tool calculates a unique set of faults and shows the count in the Add to Fault Groups form. Choose a group name and click *OK*.

Note: You cannot include rules for layout-based faults and schematic-based faults in the same group. Fault sampling too does not work for layout-based faults.

Related Topics

Injecting Faults in a Design

Creating Fault Groups

Fault Rules Form

© 2023 Cadence Design Systems, Inc.

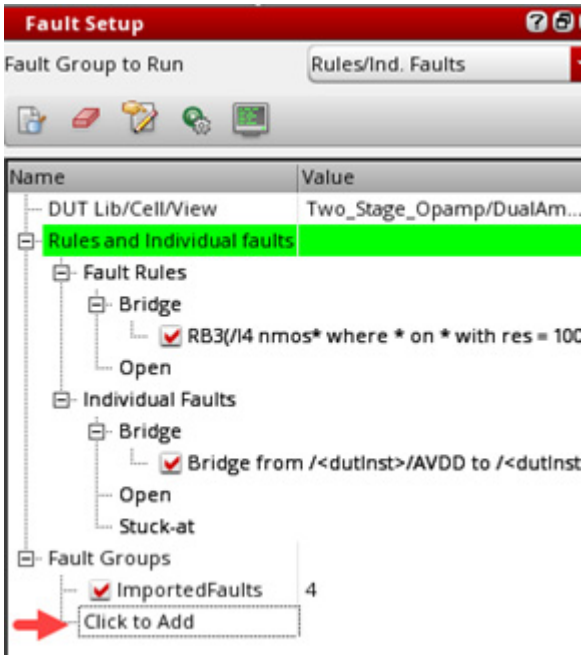
Creating Fault Groups

If you want to use some fault rules together on a design hierarchy, you can bundle them together in fault groups.

By default, the fault simulation is run for the faults specified in the *Rules and Individual Faults* tree in the Fault Setup assistant. This is indicated by the green highlight in this assistant. If you want to consider any other fault group for the simulation run, select the name of that fault group from the *Fault Group to Run* drop-down list.

To create a fault group:

1. In the Fault Setup assistant, expand the *Fault Groups* tree and click *Click to add*.



   A new item is added to the tree and the group name field becomes editable.

2. Type a name to be used for the group, and press `Enter`.

   A new group is created with the given name and its fault count is set to `0`. This group is also added to the *Fault Group to Run* drop-down list.

Related Topics

Adding Faults to Fault Groups

© 2023 Cadence Design Systems, Inc.

Creating a Fault Group by Copying An Existing Group

You can also create a fault group by copying an existing group. The newly created group gets the faults from the source group. You can modify it by adding or removing faults.

To create a copy of a fault group:

➡ Right-click the fault group and choose *Create Fault Group Copy*.

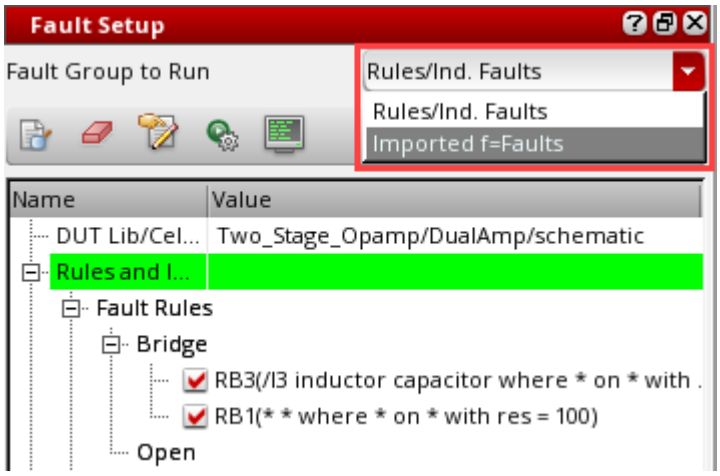
A copy of the group is saved with the default name in the format *source-group-name*_copy_*incremental-num*. You can rename it as required.

The status of the new fault group is same as that of the source group name.

© 2023 Cadence Design Systems, Inc.

Adding Faults to Fault Groups

At a given point of time, there can be many faults and fault rules defined in the Fault Setup assistant. You can include a similar set of faults in a group to manage them together or to run simulation for all faults grouped together.



You can add faults to a fault group in two ways:

- From the fault defined in the Fault Setup assistant
- From the results of a fault simulation

To add faults defined in the Fault Setup assistant to a fault group, perform the following steps:

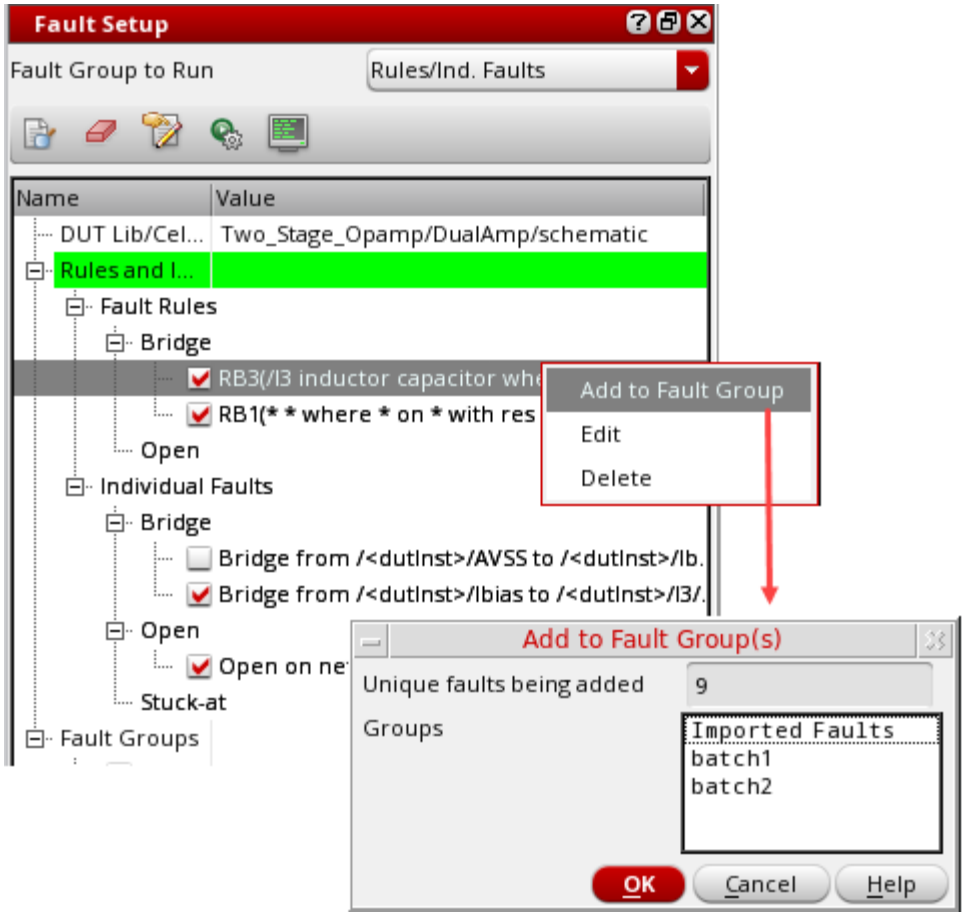
1. Select one or more fault rules or individual faults to be added to the group.
2. Right-click and choose *Add to Fault Group*.

   If no fault list is generated for the selected fault rules or their fault list is not up to date, the tool shows a message to prompt you whether you want to generate the fault list before adding the rule to the fault group. Click *Yes* if you want to continue.

   If you are sure that you want to always generate fault list in such cases, set the suppressAddToFaultGroupsPrompt environment variable to `t`.

© 2023 Cadence Design Systems, Inc.

The Add to Fault Group(s) form is displayed.



In the *Unique faults being added* field, the form displays the total count of unique faults resulting from the rules in different groups.

3. Hold the `Ctrl` key and select the names of groups in the *Groups* list.
4. Click OK.

   The faults are added to the selected groups.

   Note: If you have rules for layout-based faults, you cannot group them together with schematic-based faults in a fault group.

Tip

If you have created fault groups, ensure that in the *Fault Group to Run* drop-down list, you select the group that you want to use for the simulation run. The selected group is highlighted in green. You can also preview the list of faults included in a fault group. For more details, refer to Previewing Faults in a Fault Group.

© 2023 Cadence Design Systems, Inc.

Importing Faults from a File into a Group

You can import the faults defined in a text file directly into the fault setup. All faults imported from the fault file are added to a group in the setup.

To import the faults from a file, perform the steps given below.

1. In the Fault Setup assistant, expand the *Fault Groups* tree.
2. Click *Click to add* and specify a name for the fault group that will contain the faults imported from the fault file.

   The faults imported from the fault file are added to this fault group.
3. Right-click on the newly added fault group and choose *Import Fault File* from the context-sensitive menu.
4. In the Choose Fault Files form, browse the required CSV file path, and click *Open*.
5. The specified CSV file containing the faults is added to the setup.

   Note: The faults given in the fault file are read and their count is reflected in the fault group. The fault group is also added to the *Fault Group to Run* drop-down list.

The fault file is required to be written in a specific CSV format that is read by ADE Assembler. The following example shows the format to be used:

```
Name,Type,From,To,OpenTerminals,Net,Resistance
B1,Bridge,gnd!,/<dutInst>/I10/vdd!,,,100
OT1,Open,,,/<dutInst>/I27:B,/<dutInst>/net21,10K
ON1,Open,,,/<dutInst>/I5/M3:G,/<dutInst>/clk,10K
S1,StuckAt,/<dutInst>/net44,/<dutInst>/I2/vddd!,,,100
```


© 2023 Cadence Design Systems, Inc.

Deleting Fault Rules and Fault Groups

To delete fault rules,

1. Select one or more fault rules.
2. Right-click and choose *Delete*.

   The tool displays a message with a list of selected fault rules seeking confirmation for deletion.
3. Click *OK* to proceed with deletion.

Similarly, to delete fault groups, select one or more fault groups, right-click, and choose *Delete Fault Group* from the context menu.

© 2023 Cadence Design Systems, Inc.

Specifying Fault Sampling Options and Auto Stop Criteria

When the fault universe consists of a large number of faults, it is practically impossible to run simulations for all the faults. You can choose to run simulation for a subset of faults or apply a stopping criteria to automatically stop simulations after a specific coverage criteria is met.

Specifying Fault Sampling Options

Sampling helps in selecting a subset with fewer faults that can represent the whole fault universe. This helps in estimating the overall coverage with very less simulations.

In ADE Assembler, you can use the following sampling methods supported by Spectre:

- random
- randomuniform
- randomweighted
- weightsorted

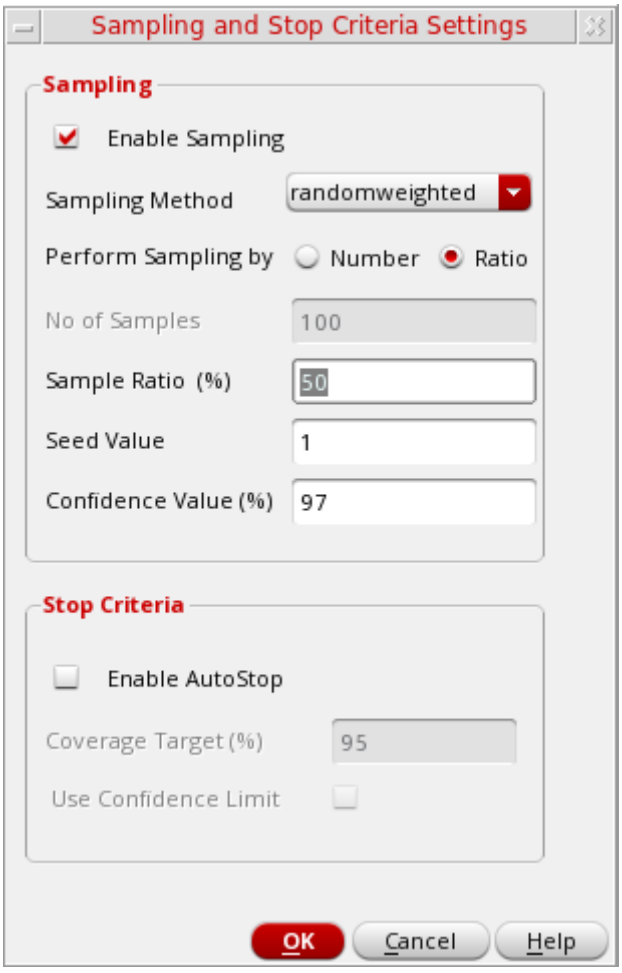
For details and guidance on these sampling methods, see Fault Selection and Sampling in Spectre Classic Simulator, Spectre APS, Spectre X, and Spectre XPS User Guide.

To specify sampling options for fault analysis:

1. Click *Sample the faults* on the Fault Setup assistant toolbar.

© 2023 Cadence Design Systems, Inc.

The Sampling and Stop Criteria Settings form is displayed.



2. Select *Enable Sampling*.
3. Choose a sampling method from the *Sampling Method* drop-down list.
4. In the *Perform Sampling By* field, choose one of the following options to specify how you want to choose samples:

   - *Number*: Uses a specific number of faults. This is selected by default.

     If you choose to perform sampling by number, specify the number of random samples to be selected in the *Number of Samples* field.

   - *Ratio*: Uses a percentage or ratio of the possible samples.

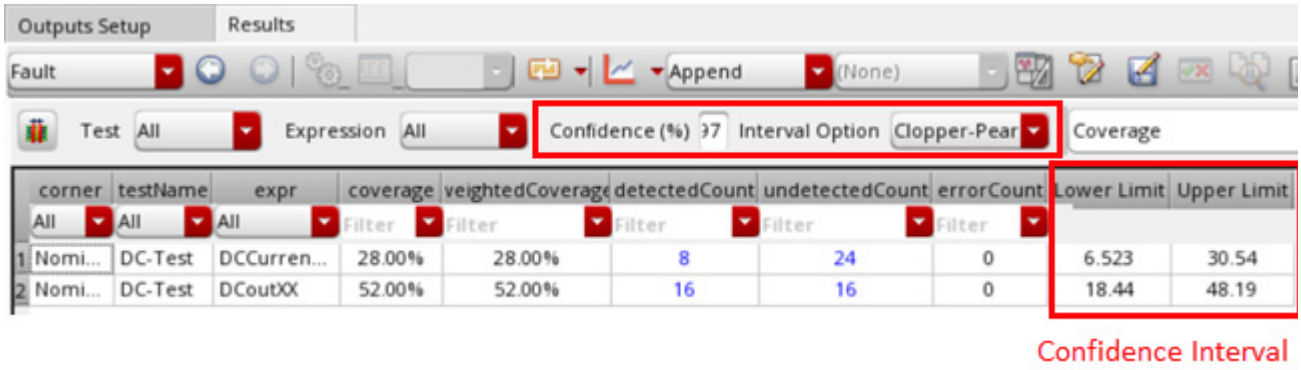
     If you choose to perform sampling by ratio, the *Sample Ratio (%)* field is enabled. Specify the percentage of the samples to be selected.

© 2023 Cadence Design Systems, Inc.

5. In the *Seed Value* field, specify the seed value to be used as the starting point for random sample generation.
6. If the sampling method is *randomuniform* or *randomweighted*, sampling is performed by replacement, which means that a selected fault is retained in the sampling pool and may be sampled again. With this method, the number of faults simulated can be fewer than the number of samples selected. Therefore, fault detection is done at a specific confidence level. The default confidence level is 97%, but you can change it in the *Confidence Value (%)* field.
7. Click *OK* to save the sampling options and close the form.

The sampling options specified on this form are used to generate samples of faults used for simulation. The Fault Group Preview form also shows the sampling information and an effective list of faults.

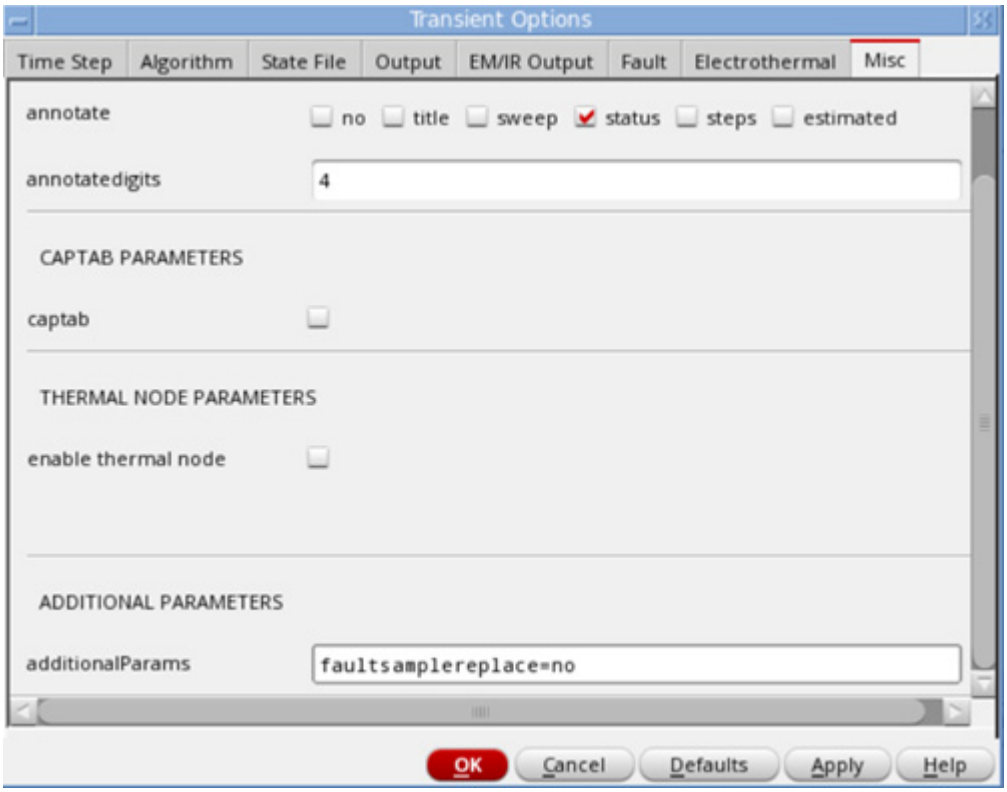
For the *randomuniform* or *randomweighted* sampling methods, Spectre internally uses the Clopper-Pearson formula to estimate the confidence interval when the sampling based simulation is finished. After the results are displayed in the *Faults* results view, you can change the value in the *Confidence (%)* field or the confidence interval calculation formula in the *Interval Option* drop-down list on the toolbar and check the confidence interval.



Spectre supports two approaches for confidence interval calculation with the given confidence level:

- Clopper-Pearson formula: This formula is recommended when fault sampling with replacement is performed. Therefore, when using this formula, use the default value yes for the `faultsamplereplace` option.
- Wald formula: This formula is recommended when fault sampling is done without replacement. When using this formula, specify `faultsamplereplace=no` in the *additionalParams* field on the *Misc* tab of the Transient Options form, as shown below.

© 2023 Cadence Design Systems, Inc.



For more details about these approaches, see Calculating Confidence Interval for Different Sampling Methods.

Specifying Automatic Stopping Criteria

To specify auto stop criteria:

1. In the Sampling and Stop Criteria Settings form, select *Enable AutoStop*.
2. Specify a percentage value in the *Target Coverage (%)* field. When the given percentage of total faults are run, ADE Assembler stops simulating the remaining points.
3. (Optional) Select *Use Confidence Limit* to use the confidence limit instead of coverage target as a criteria for automatic stopping. Use this option when you selected the `randomuniform` or `randomweighted` sampling method.
4. Click *OK* to save the options and close the form.

In the next simulation run, ADE Assembler considers the settings specified on this form and automatically stops the simulation when the specified criteria is met. For example, if you specify the target coverage percentage as 60%, ADE Assembler stops the simulation after

© 2023 Cadence Design Systems, Inc.

60% points are run. For the remaining points, the results are shown as canceled in the `Detail` or `Detail-Transpose` view.

Note: Sometimes, auto stopping might lead to a slightly higher percentage in the results. For example, you set the coverage target as 75%, the final coverage might be shown as 77%. This can happen because by the time calculations are done and the simulation actually stops, more faults get executed in parallel resulting into a higher coverage.

© 2023 Cadence Design Systems, Inc.

Previewing Faults in a Fault Group

Fault preview helps in viewing the final list of faults the tool will use to run simulation for the selected fault group.

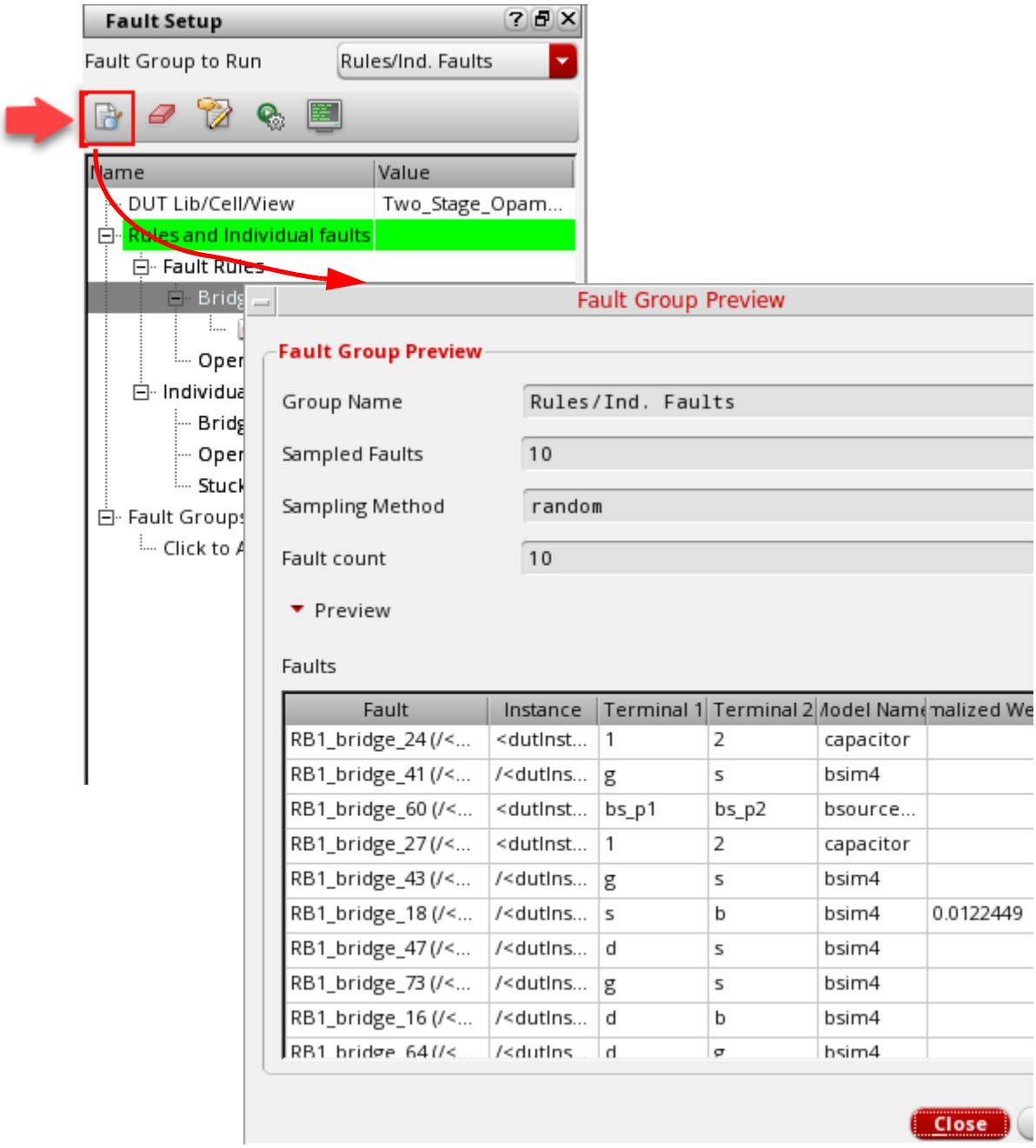
To preview the list of faults included in a fault group, do this:

1. Select the name of the fault group in the *Fault Group to Run* drop-down list on top of the Fault Setup assistant.
2. Click *Preview all faults for fault group selected to run* ( ) on the Faults Setup assistant.

   ADE Assembler runs the Spectre info analysis to identify the fault list and if enabled, applies the sampling options to arrive at the final list of fault samples in the *Faults* table.

© 2023 Cadence Design Systems, Inc.

It then displays the list in the Fault Group Preview window.



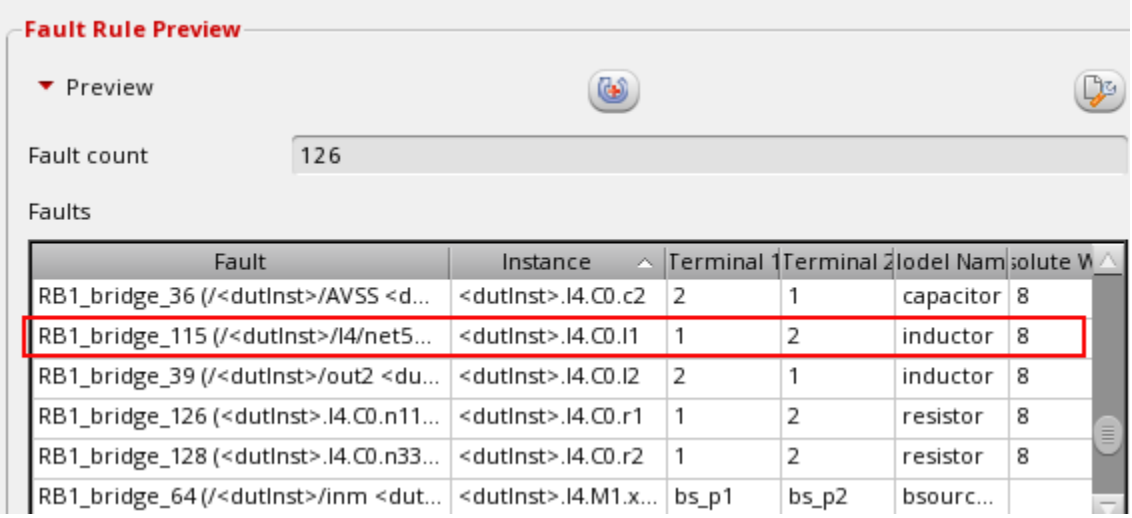
Key Points to Observe

- If the select group is `Rules/Ind. Faults`, the *Faults* table shows a combination of all the faults defined by the rules and the individual faults defined in the assistant.

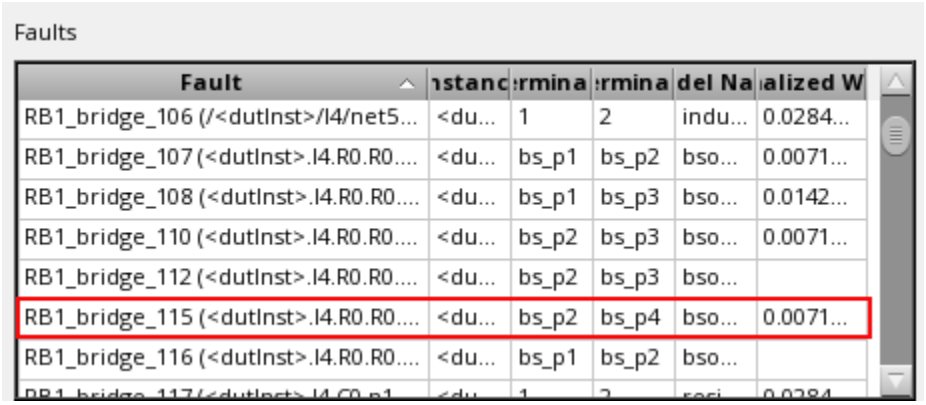
© 2023 Cadence Design Systems, Inc.

- When multiple fault rules are enabled, the weights of faults are normalized across all fault rules to ensure that the relative weight of a fault looks reasonable when compared to other faults from other rules. In the example given below, the fault `RB1_brige_115` has an absolute weight of 8 in the Fault Rule Setup form, where its weight changes to 0.00771after normalization.

Fault Rule Setup



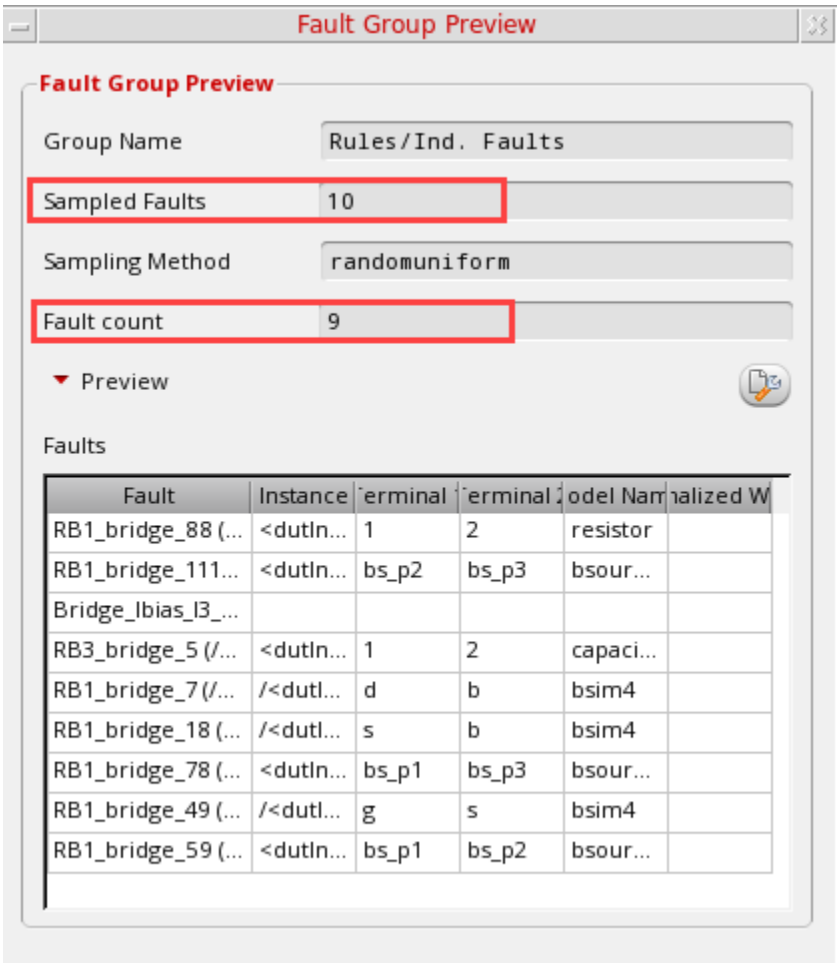
Fault Rules Preview



- When the `randomuniform` or `randomweighted` sampling methods are used, you may observe that the number of faults in the list are less than the sampled faults. This is because in the replacement method, it is possible to choose a sample more than once,

© 2023 Cadence Design Systems, Inc.

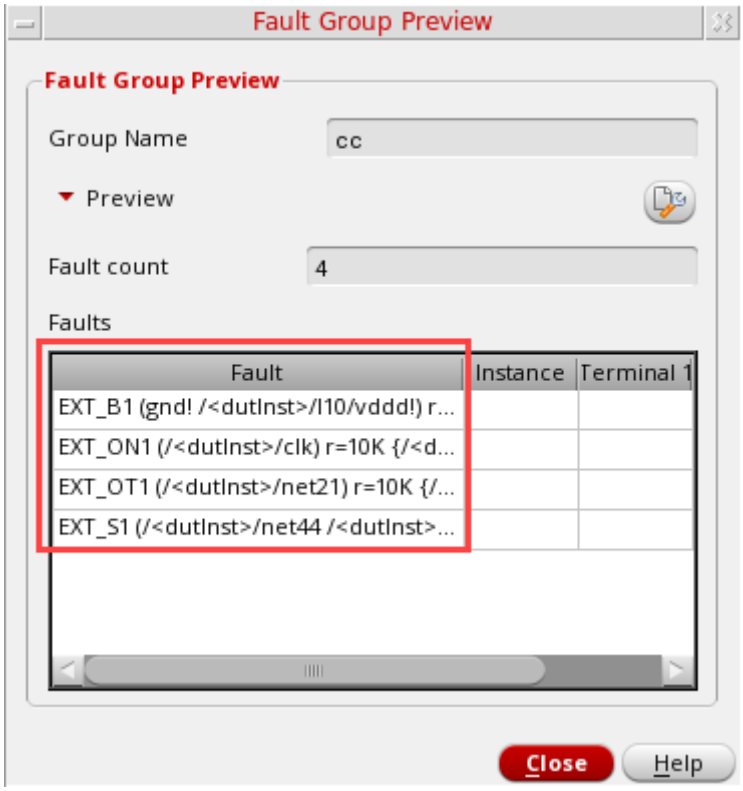
but it is not simulated multiple times. In this case, the fault preview shows a mismatch in the sampled faults and fault count, as shown below.



- When the `randomuniform` sampling method is used, the *Normalized Weight* column in the *Faults* table is blank. This is because Spectre does not consider weighted likelihood with this sampling method. For more details, refer to Fault Selection and Sampling in *Spectre Classic Simulator, Spectre APS, Spectre X, and Spectre XPS User Guide*.
- If the faults are imported from a fault file, they are prefixed with `EXT_` to indicate that these are imported from an external source, and all details for each fault are displayed in

© 2023 Cadence Design Systems, Inc.

the *Fault* column. Other columns remain blank. An example of the imported faults is shown below.



- You can configure the list of columns to be displayed in the *Faults* table, click and customize the display settings according to your preferences.

Probing Faults in a Schematic

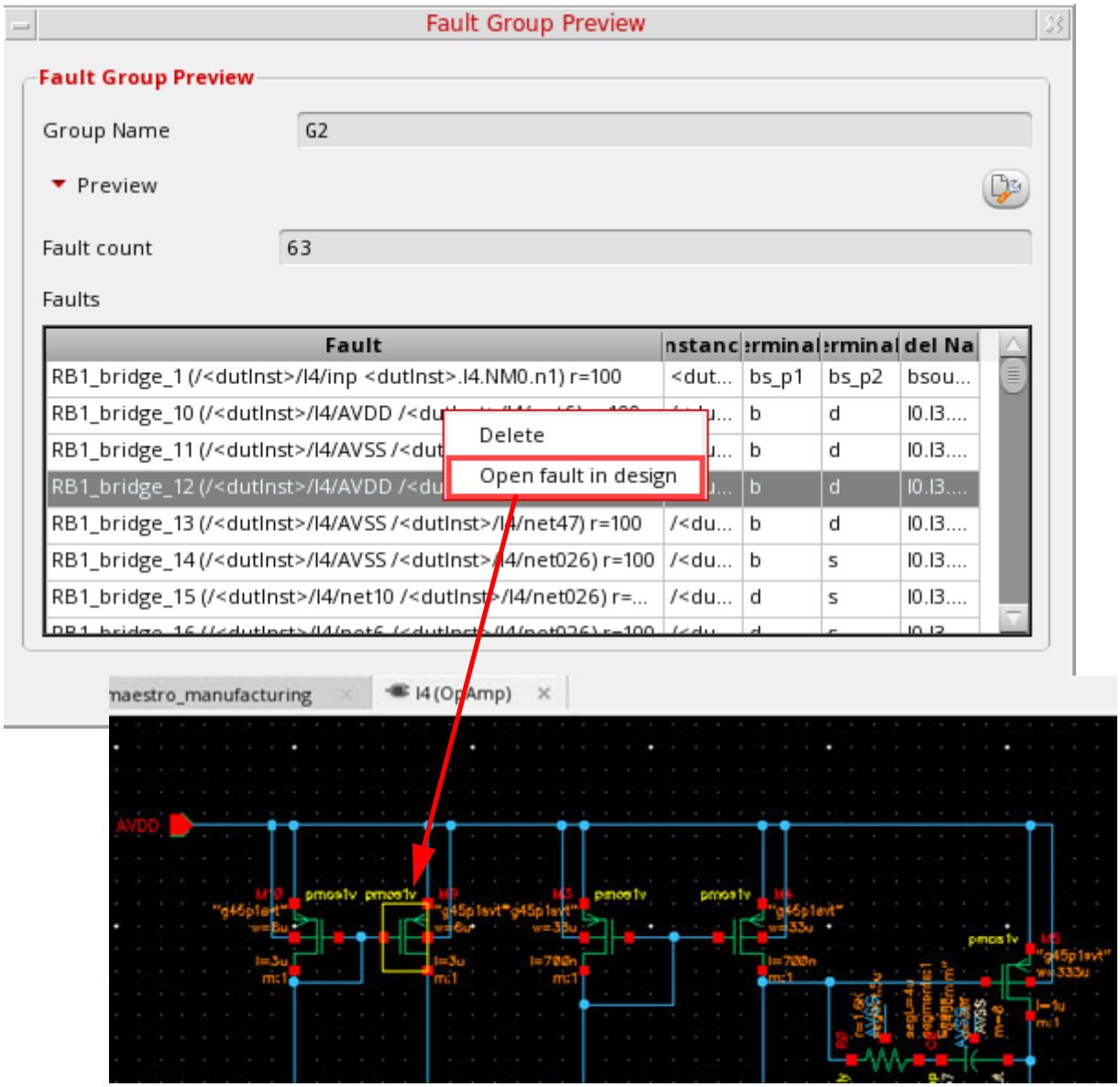
While previewing faults in the Fault Group Preview form, you can also probe them in the schematic view of the design.

To probe a fault in the schematic:

1. Preview the fault list for the group.
2. In the Fault Group Preview form, right-click a fault and choose *Open fault in design*.

© 2023 Cadence Design Systems, Inc.

ADE Assembler opens the schematic in a new tab and highlights the device with the fault.



Clearing Fault Cache

After modifying the fault rules or individual faults, you need to re-evaluate the fault universe by running the rules again. To clear the fault cache and re-validate the faults before generating the fault preview, click on the toolbar of the Fault Setup assistant. After clearing the cache, you can run the *Preview Faults* command to regenerate the fault list.

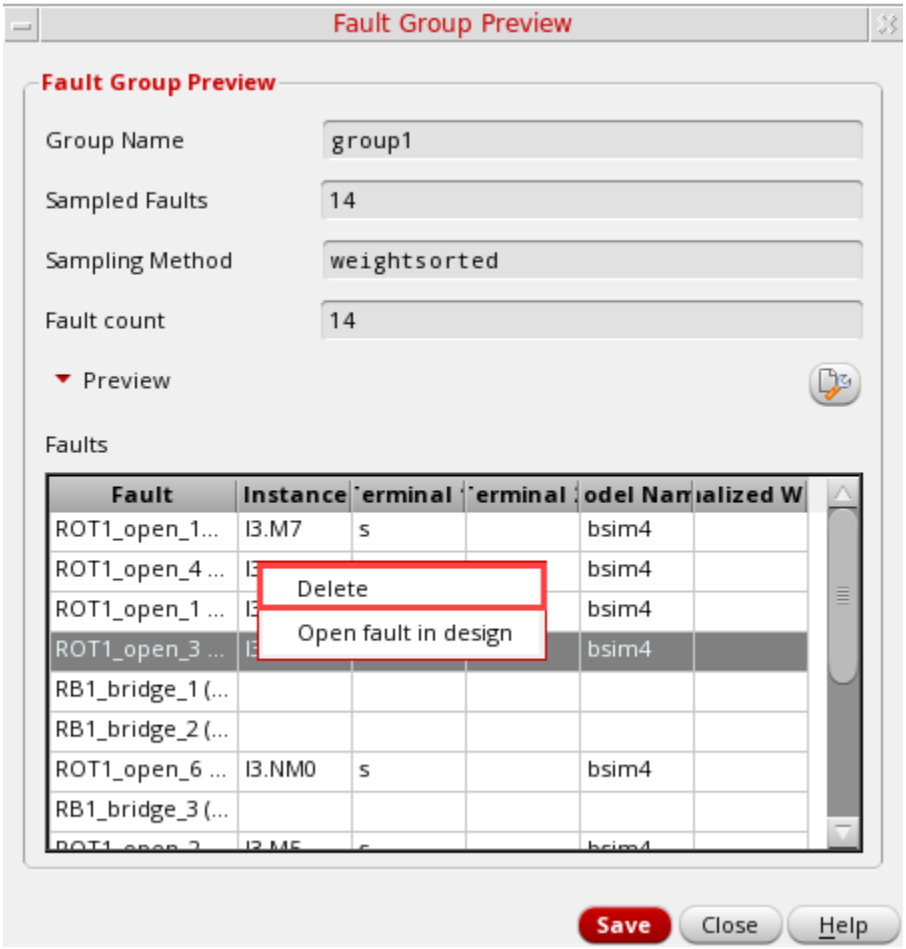
© 2023 Cadence Design Systems, Inc.

Modifying Fault Groups

You can modify a fault group by removing the faults not required for that group. To remove faults from a fault group:

1. Preview the fault list for the group.
2. In the Fault Group Preview form, select the faults to be removed.

   You can hold the *Ctrl* key and click faults for multiple selection.
3. Right-click and choose *Delete*.



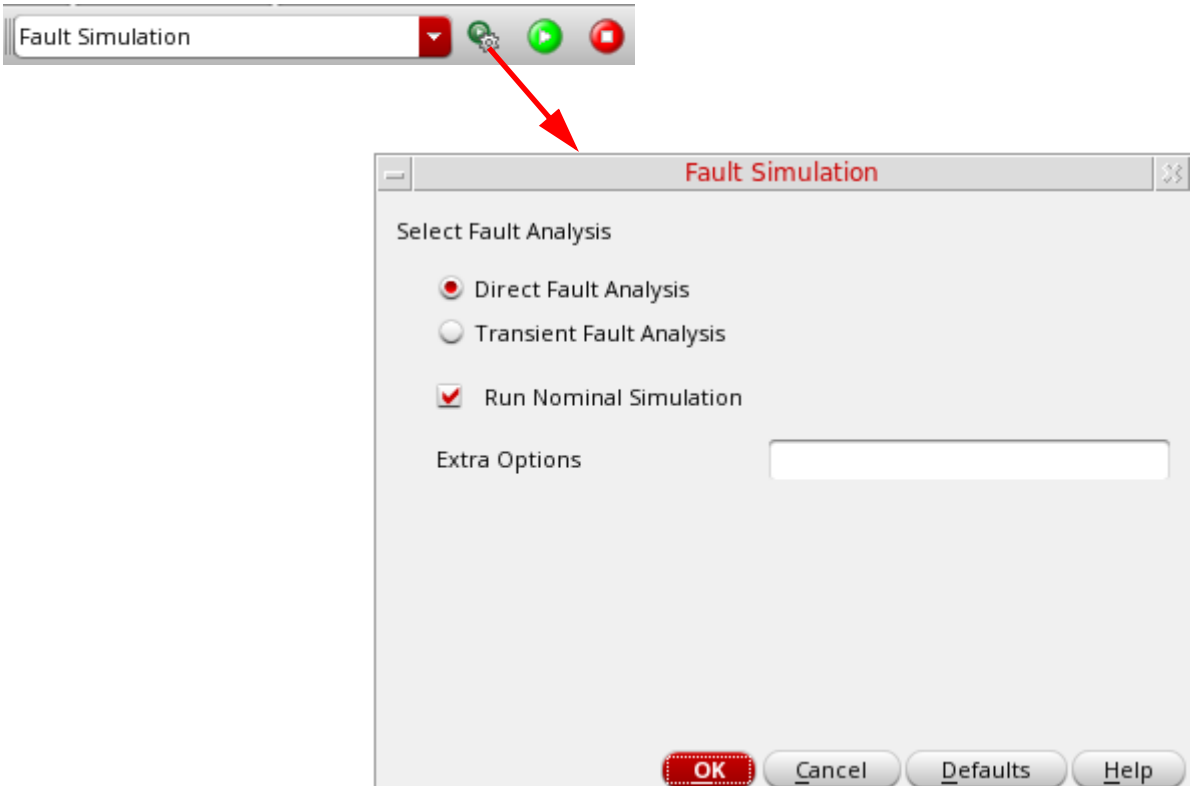
   The selected faults from removed from the preview.
4. Click *Save* to save the changes in the fault group.

   The fault count for this fault group is updated on this form as well as in the Fault Setup assistant.

© 2023 Cadence Design Systems, Inc.

Setting up Run Options for a Fault Simulation

Before running a fault simulation, spefcify the run options to be used. For this, click Simulation Options in the ADE Assembler toolbar.



You can select from the two methods given on the Fault Simulation run options form. These methods use two different fault analysis engines provided by Spectre–direct fault analysis and transient fault analysis. These engines provide a trade-off between accuracy and performance.

Direct Fault Analysis

In the direct fault analysis, faults are injected at the beginning of each analysis type, such as dc, ac, or transient. This mode provides the most accurate results. However, when there is a large number of faults in the design hierarchy, the simulations might take long to complete.

© 2023 Cadence Design Systems, Inc.

Transient Fault Analysis

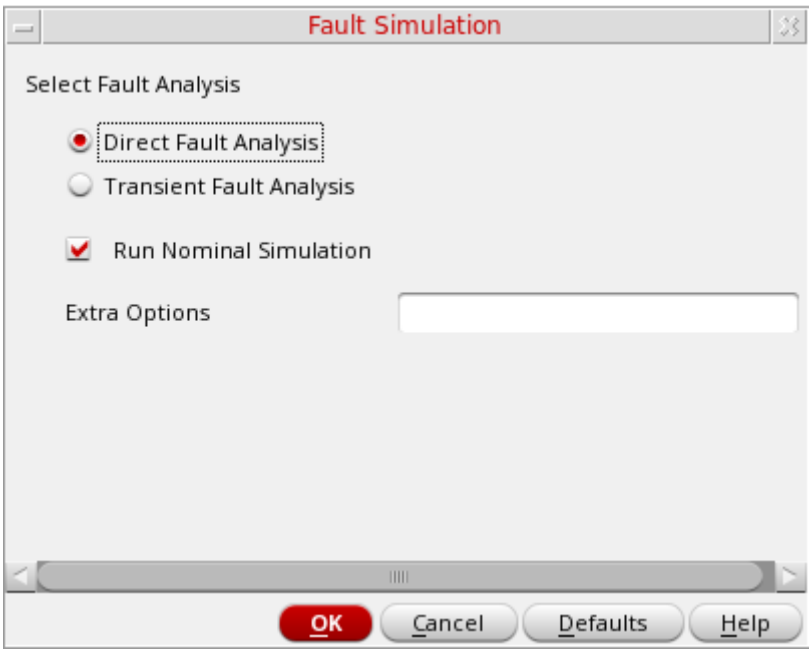
In the transient fault analysis, a circuit with fault is simulated together with the test points specified on the time axis. Spectre starts regular transient analysis till the time reaches the test point. Then, at each test point, the following actions are performed:

- Fault-free transient state is saved
- A fault is injected into the circuit
- With fault-free state used as the initial condition, transient fault simulation is continued with regular time step control and accuracy settings
- The result is saved and the database is cleaned when all fault solutions are found

Fault-free transient simulation is restored and continues till the next test time point.

Run Options for Direct Fault Analysis

As explained earlier, this Direct Fault Analysis is used to inject faults in the beginning of each analysis type and considers them for the entire run. It provides most accurate results, but should be avoided for large designs.



Run Nominal Simulation — Specifies whether to perform a nominal (fault free) analysis before fault analysis.

© 2023 Cadence Design Systems, Inc.

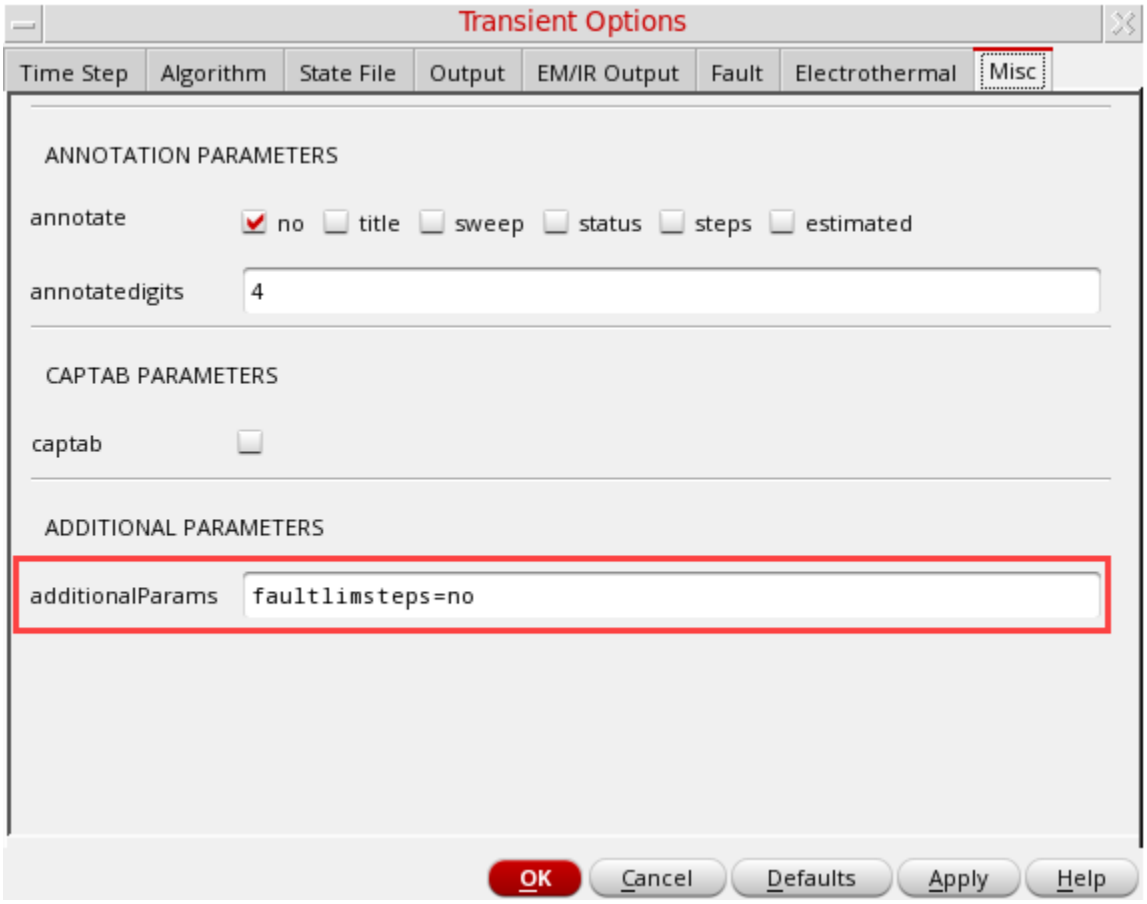
Extra Options — Specifies additional options for the selected fault analysis. You can specify a space-separated list of `option-value` pair of options for that are currently not available in ADE Assembler, but are available in Spectre.

The options specified in the *Extra Options* field are added to the sweep statement for the direct fault analysis. These options are not validated or processed by ADE Assembler. They are directly added to the netlist. Ensure that these are specified in the correct format.

To use any extra options for the transient analysis statement:

1. Expand *Analyses* for the test in the Data View assistant.
2. Double-click *tran* to open the Choosing Analysis form.

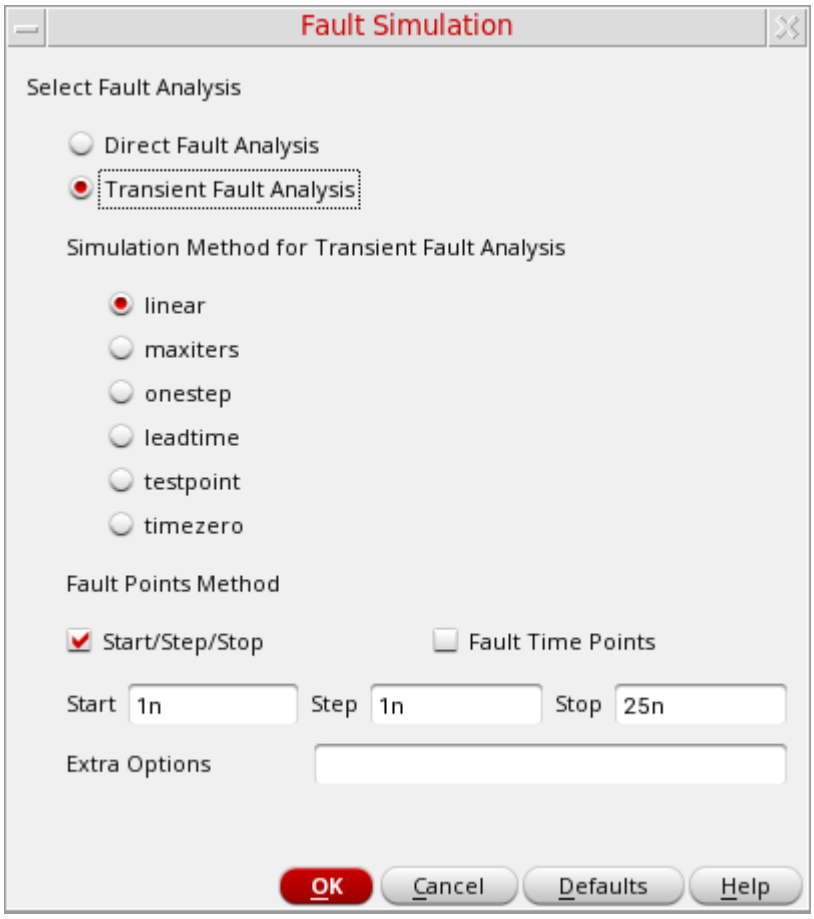
   The *tran* analysis is selected and the form shows options for this analysis.
3. Click *Options* to open the Transient Options form.
4. Specify additional options in the *additionalParams* field on the *Misc* tab. An example is shown below.



© 2023 Cadence Design Systems, Inc.

Run Options for Transient Fault Analysis

Transient fault analysis injects and considers faults only at specific test points and given lead times on the time axis. For the other time values, a fault-free state of circuit is used. This method produces results very fast, but with limited accuracy.



The description of each field is given below.

Section: *Simulation Method for Transient Fault Analysis*

- *linear* — Fastest method with limited accuracy as it simulates one time step at the test point and runs only one Newton iteration.
- *maxiters* — Integrates the time step at the test point taking multiple Newton iterations until the results converge to a commonality or the specified number of maximum iterations are covered.
- *onestep* — Works like *maxiters*, except that it steps back one time step if the simulation results do not converge and continues the Newton iterations until a solution is found.

© 2023 Cadence Design Systems, Inc.

- *leadtime* — An accurate method in which the faults are injected prior to each test point with the lead time, which is specified in the field *Lead Time*.
- *testpoint* — Performs Transient Fault Analysis for every time step till the next test point by injecting faults after the previous test point.
- *timezero* — The most accurate method with the longest simulation run time as it injects faults at time zero and performs transient fault analysis for every time step till the next test point.

Section: *Fault Points Method*

- *Start/Step/Stop* — Displays and enables the fields to set the start, step and stop time for transient fault analysis.
  - *Start* – The first test point or start time
  - *Step* – The time interval between test points
  - *Stop* – The last test point or stop time
- *Fault Time Points* — Displays and enables the *Fault Time Points* field in which you can specify a space-separated list of time points at which the transient fault analysis is to be performed.

Important

These fault time points specified in the run options form for fault simulation are used for all tests in the setup. However, if a particular test contains specific time points in the transient analysis options, the test-specific values take precedence over the global values in the run options.

- *Extra Options* — Specifies additional options for the Transient Fault Analysis. You can specify options that are currently not supported by ADE Assembler, but are supported by Spectre. You can specify a space-separated list of `option=value` pairs.

  Note: Extra options are not validated or processed by ADE Assembler. Ensure that these are specified in the correct format.

Running a Fault Simulation

After injecting the faults and setting the run options, you can run the Fault Simulation by clicking *Run Simulation* on ADE Assembler toolbar.

ADE runs Spectre info analysis before every fault simulation run to resolve the fault universe. In this process, Spectre removes all duplicate faults added by different rules.

© 2023 Cadence Design Systems, Inc.

Also see: warnWhenSimsExceed

After the simulation is complete, the *Results* tab shows the results in *Detail* view. It is a consolidated report of the fault expressions defined in the tests. For a detailed analysis of results for each fault, open the *Fault* result view and apply filters as explained in the Viewing Results of Fault Simulation section.

If there are failures in the simulation results, you can review the netlist or the setup. For more details, refer to the Debugging Results and Troubleshooting Errors section.

Important

To improve the performance of the fault analysis, you can distribute the run to multiple jobs by configuring the job policy.

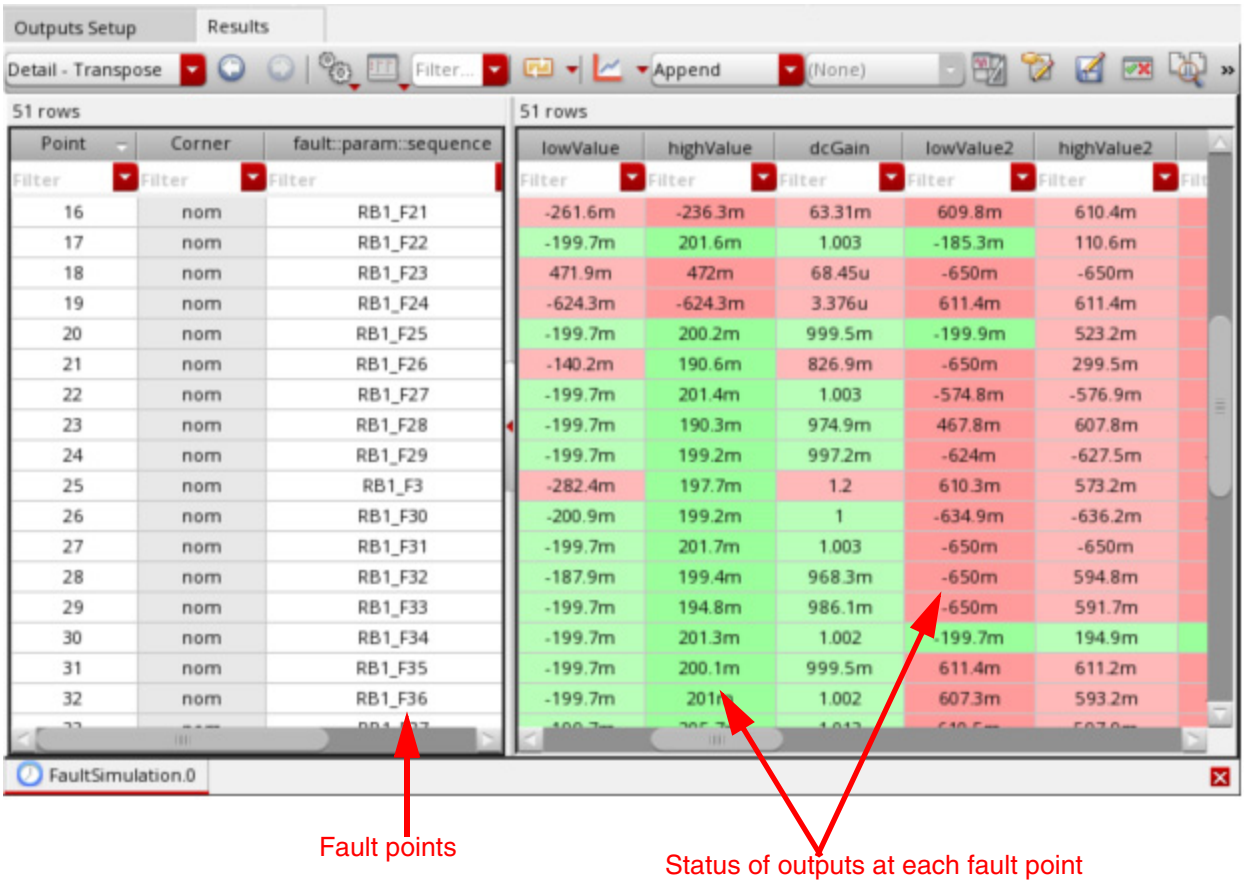
Related Topic: Running Simulations Using Selected Faults from Results

Viewing Results of Fault Simulation

The simulation tries to detect the injected faults for each output of type `faultexpr`, a new output type in ADE Assembler. At each fault point, the simulator evaluates the outputs and

© 2023 Cadence Design Systems, Inc.

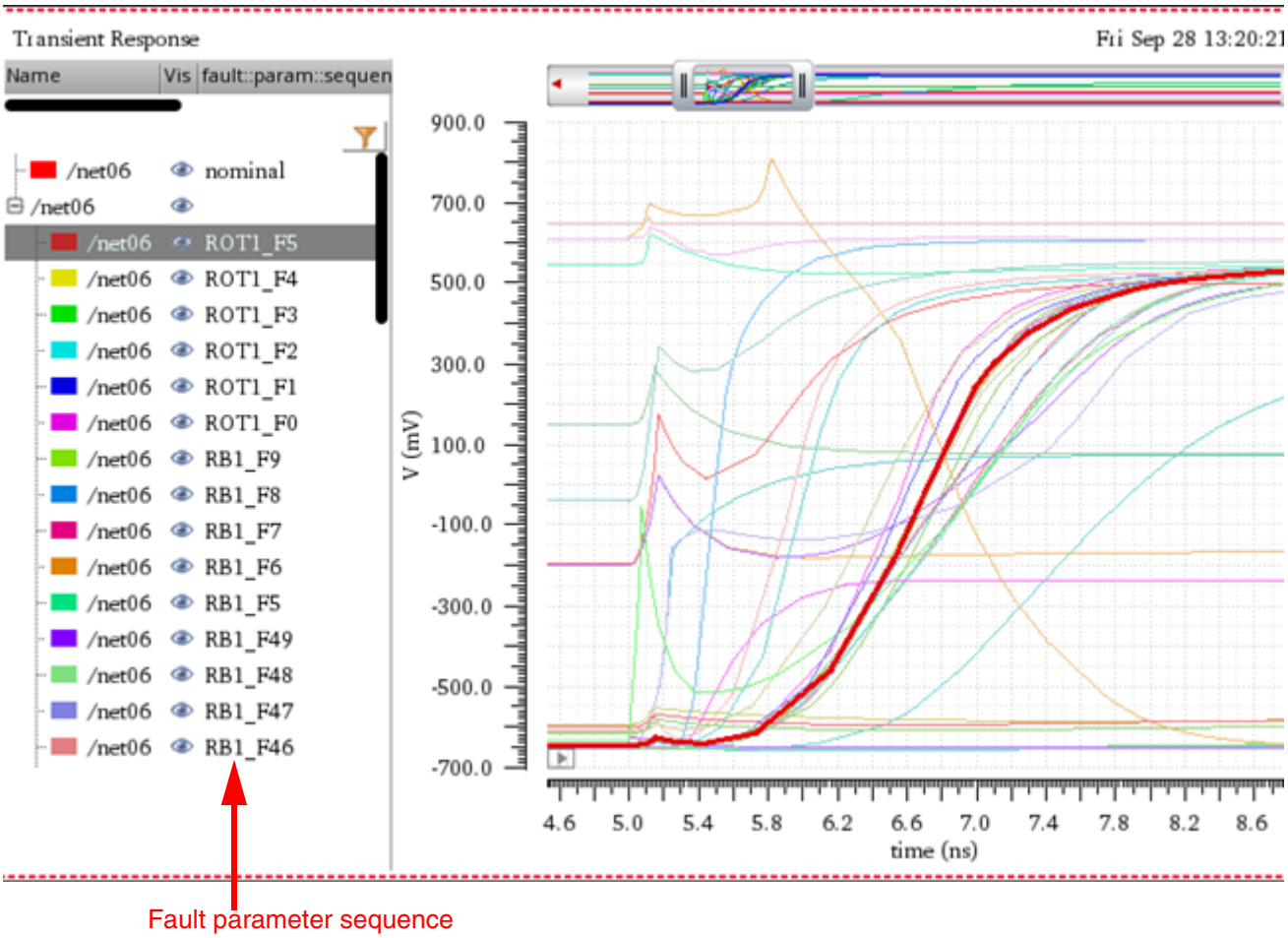
calculates the count of detected and undetected faults, as shown in the snapshot of the `Detail-Transpose` results view given below.



The red and green colors of the cells in the Detail and Detail-Transpose views indicate detected and undetected faults, respectively. Though for a fault simulation, it is considered good when a fault is detected, this color schema has been kept similar to the one used for the results of specifications where red indicates a failure in meeting the defined specification for an output.

© 2023 Cadence Design Systems, Inc.

In the `Detail` view, you can plot graphs for outputs across all design points, as shown below.



The following topics explain how to explore the results of a fault simulation:

- Viewing Summary of Fault Simulation Results
- Customizing the Display of Faults in the Fault View
- Merging Results from Multiple Fault Simulations
- Running Simulations Using Selected Faults from Results

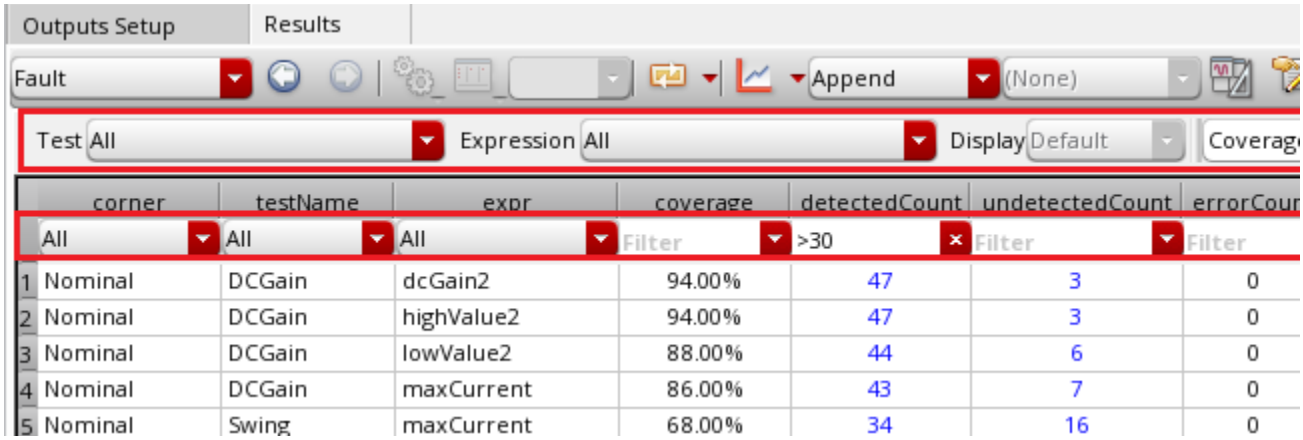
© 2023 Cadence Design Systems, Inc.

Viewing Summary of Fault Simulation Results

You can view the summary of fault results in the `Fault` results view. By default, this view shows the count of detected or undetected faults found while evaluating the expressions for each test.

By default, all simulation errors and evaluation errors are considered as undetected faults. You can use the handleErrorsAs environment variable to ignore the errors or consider them as detected faults. When the errors are ignored, they are not included in the count of detected or undetected errors.

You can use the various filters or hyperlinks to focus on specific test, expression, or output, or to view faults in the design schematic.



When the setup has multiple tests, you can use the coverage report to identify the useful tests that you can keep and eliminate others.

For this, select `Cumulative` in the *expr* column. The cumulative results for all the expressions evaluated for each test are shown. The *detectedCount* and *undetectedCount* columns show the total number of faults detected or undetected for all expressions evaluated for that test.

The example given below shows that test DCGain detects a total of 47 out of 50 faults (with 94% coverage). Therefore, it gives a better coverage, but at the same time, you need to analyze the undetected faults for this test. It helps to ensure that the undetected fault is not a must item to be detected. If it is, you need to modify the setup to ensure that the fault is detected.

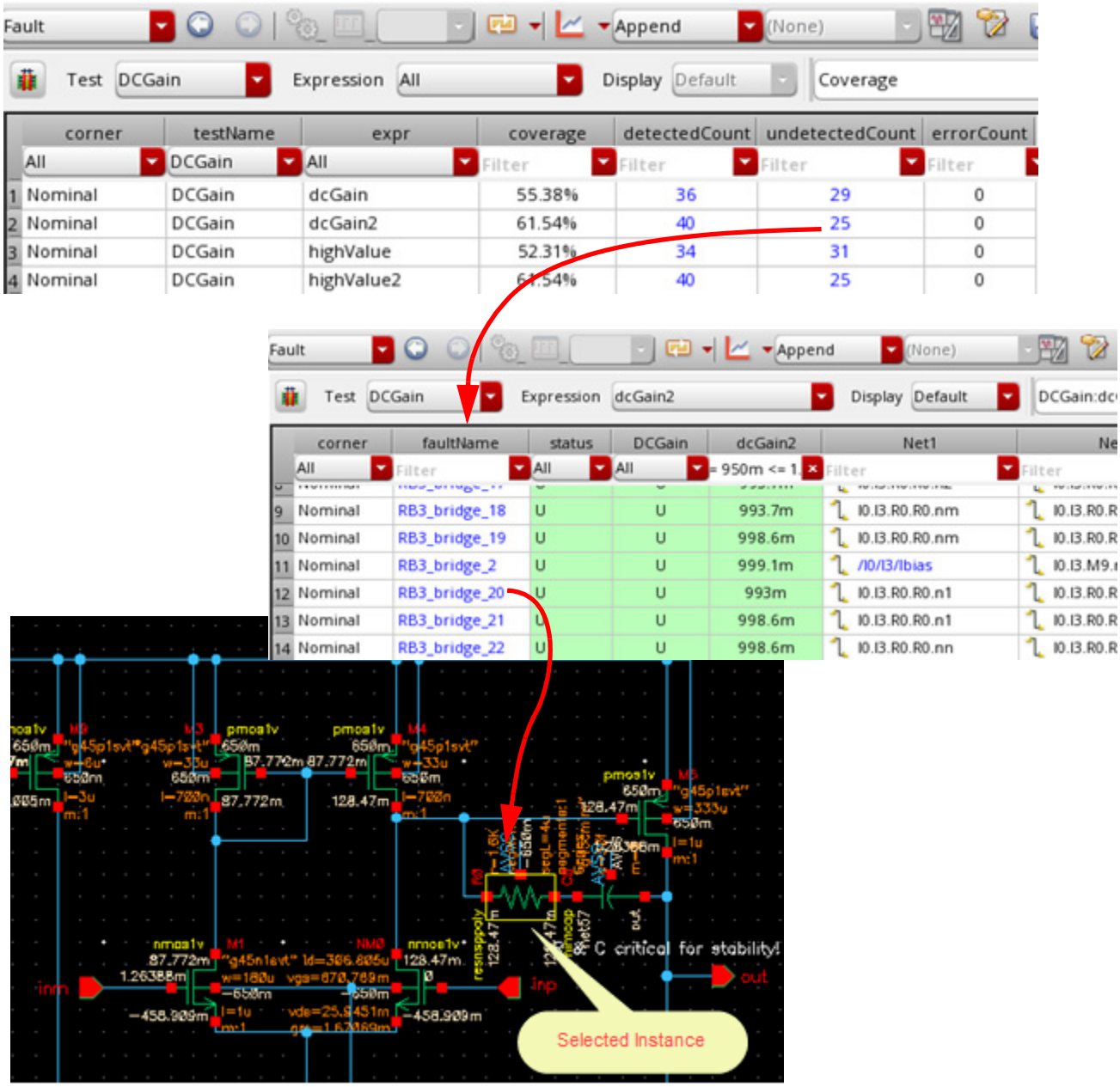
© 2023 Cadence Design Systems, Inc.

Exploring Instance that Contain Faults

The values in the `fault` view are hyperlinked to further details which are helpful in exploring the detected or undetected faults at the schematic level. For example, the count of detected or undetected faults are hyperlinks to the detailed view of these faults.

In the detailed view, you can click a fault name to view the design instance where the fault is reported, as shown in the following example:

© 2023 Cadence Design Systems, Inc.



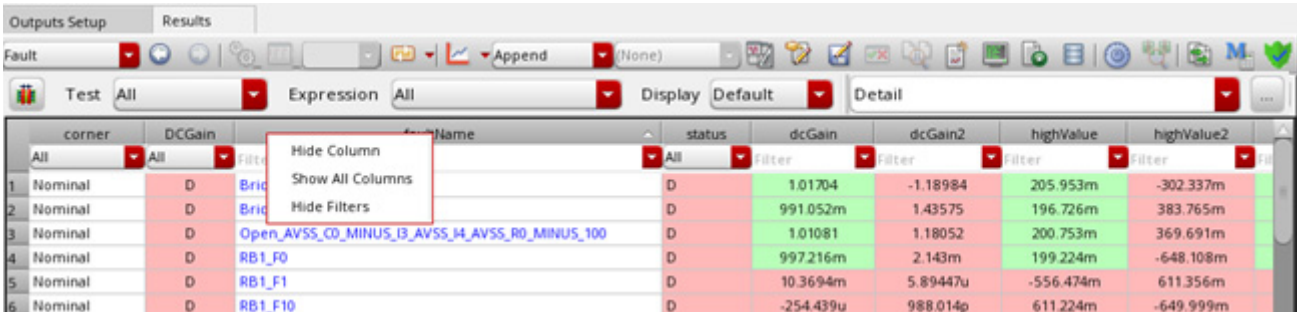
Customizing the Display of Faults in the Fault View

You can customize the view of the fault table in the following ways:

- **Hide the columns you do not want to see.** For this, use the Hide Column command from the context-sensitive menu of the Fault table.

© 2023 Cadence Design Systems, Inc.

You can use the Show All Columns command to view the hidden columns.



- **Change the order in which columns are displayed.** For this, select the column you want to move, drag it to the location where you want to place it, and drop it.
- **Apply the field or column filters to view the selected fault data.** Columns such as Corner, status, or DC Gain have drop-down filters that you can use to choose a specific value to filter data.

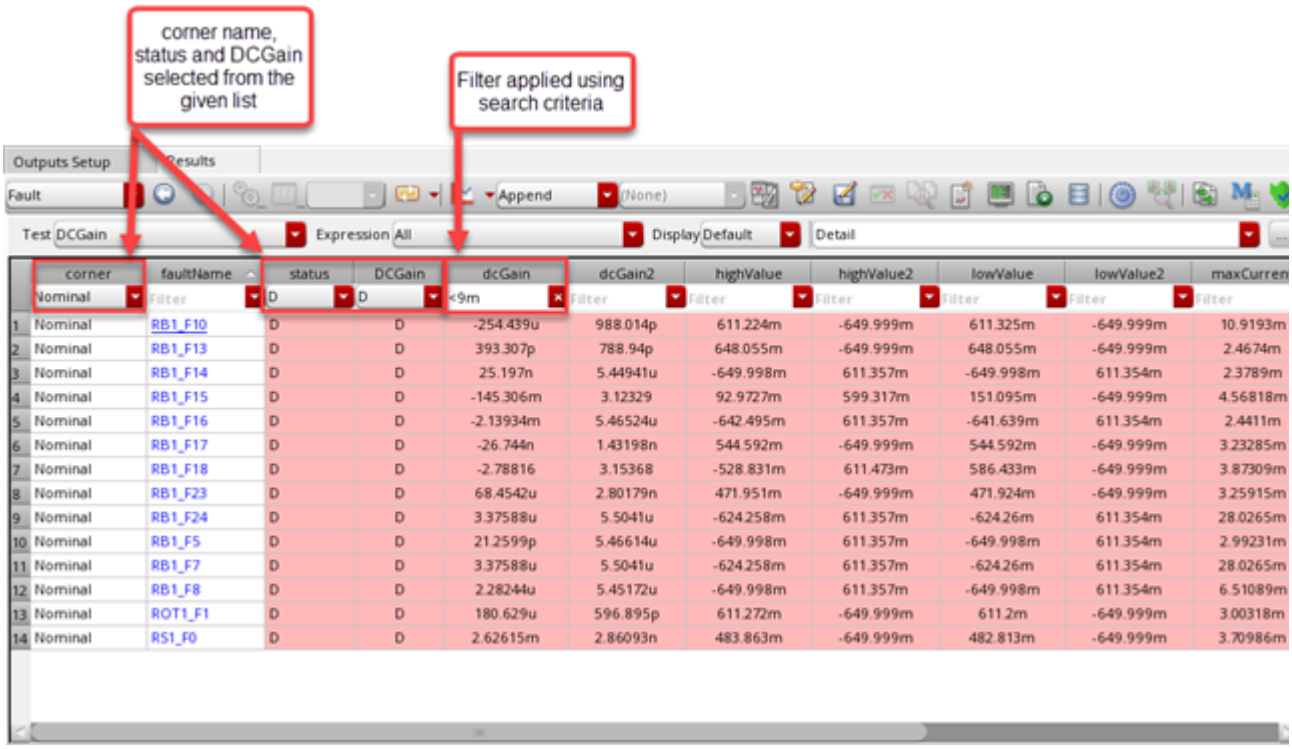
  Columns such as faultName, dcGain, dcGain2, highValue, and so on, provide enhanced filters where you can type in the filter criteria using specific values and wildcard characters. Specify a criteria and press Enter to view the filtered rows.

  Note the following important points about the column filters:

  - **You can apply filters in multiple columns together.**

© 2023 Cadence Design Systems, Inc.

The following example shows how column filters have been used to view all the faults for corner `Nominal` where status is `D`, DCGain is `D` and dcGain is less than 9m.

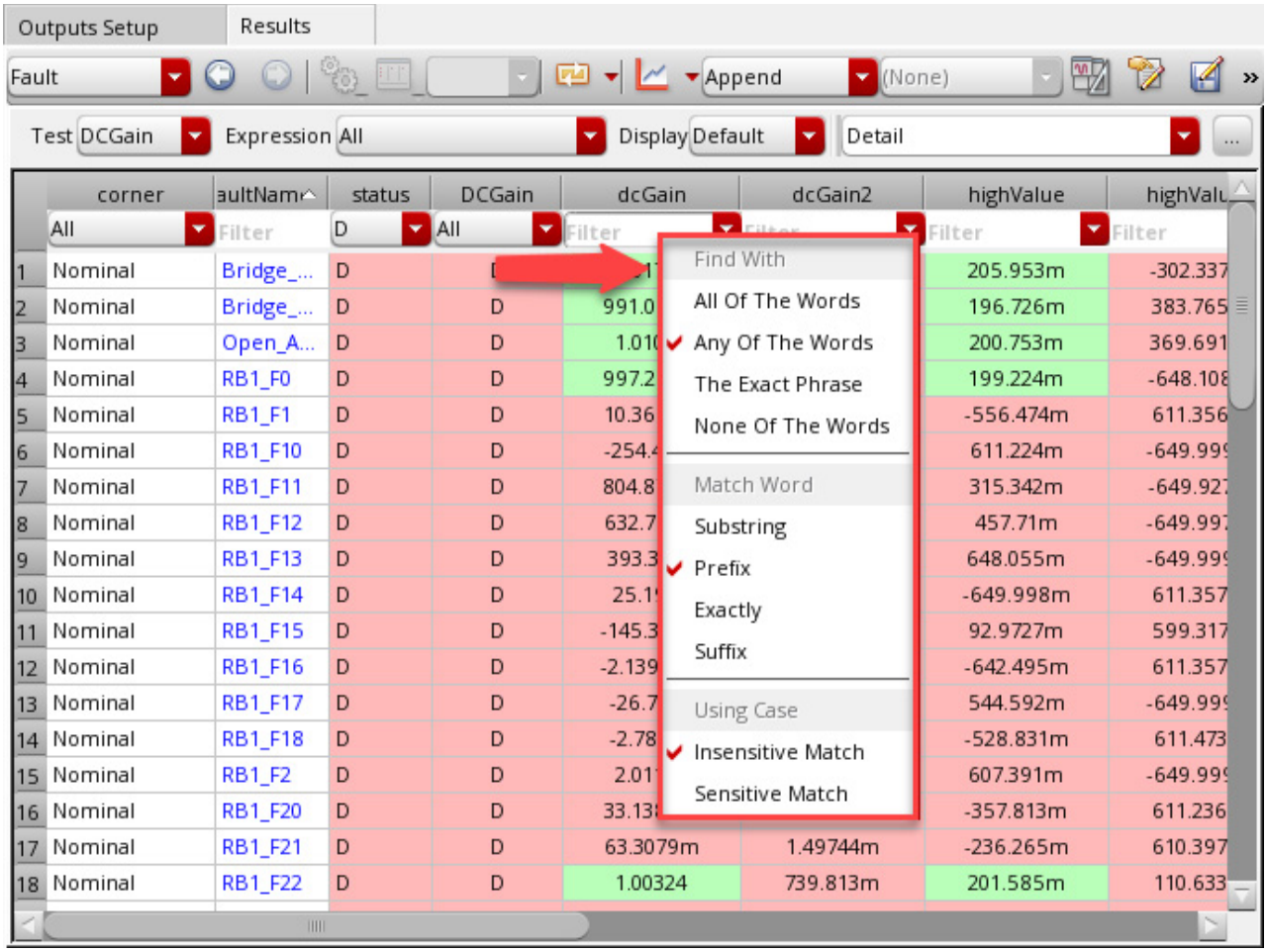


	corner	faultName	status	DCGain	dcGain	dcGain2	highValue	highValue2	lowValue	lowValue2	maxCurren
	Nominal	Filter	D	D	<9m	Filter	Filter	Filter	Filter	Filter	Filter
1	Nominal	RB1_F10	D	D	-254.439u	988.014p	611.224m	-649.999m	611.325m	-649.999m	10.9193m
2	Nominal	RB1_F13	D	D	393.307p	788.94p	648.055m	-649.999m	648.055m	-649.999m	2.4674m
3	Nominal	RB1_F14	D	D	25.197n	5.44941u	-649.998m	611.357m	-649.998m	611.354m	2.3789m
4	Nominal	RB1_F15	D	D	-145.306m	3.12329	92.9727m	599.317m	151.095m	-649.999m	4.56818m
5	Nominal	RB1_F16	D	D	-2.13934m	5.46524u	-642.495m	611.357m	-641.639m	611.354m	2.4411m
6	Nominal	RB1_F17	D	D	-26.744n	1.43198n	544.592m	-649.999m	544.592m	-649.999m	3.23285m
7	Nominal	RB1_F18	D	D	-2.78816	3.15368	-528.831m	611.473m	586.433m	-649.999m	3.87309m
8	Nominal	RB1_F23	D	D	68.4542u	2.80179n	471.951m	-649.999m	471.924m	-649.999m	3.25915m
9	Nominal	RB1_F24	D	D	3.37588u	5.5041u	-624.258m	611.357m	-624.26m	611.354m	28.0265m
10	Nominal	RB1_F5	D	D	21.2599p	5.46614u	-649.998m	611.357m	-649.998m	611.354m	2.99231m
11	Nominal	RB1_F7	D	D	3.37588u	5.5041u	-624.258m	611.357m	-624.26m	611.354m	28.0265m
12	Nominal	RB1_F8	D	D	2.28244u	5.45172u	-649.998m	611.357m	-649.998m	611.354m	6.51089m
13	Nominal	ROT1_F1	D	D	180.629u	596.895p	611.272m	-649.999m	611.2m	-649.999m	3.00318m
14	Nominal	RS1_F0	D	D	2.62615m	2.86093n	483.863m	-649.999m	482.813m	-649.999m	3.70986m

- **You can use numerical comparison operators**, such as > or <.

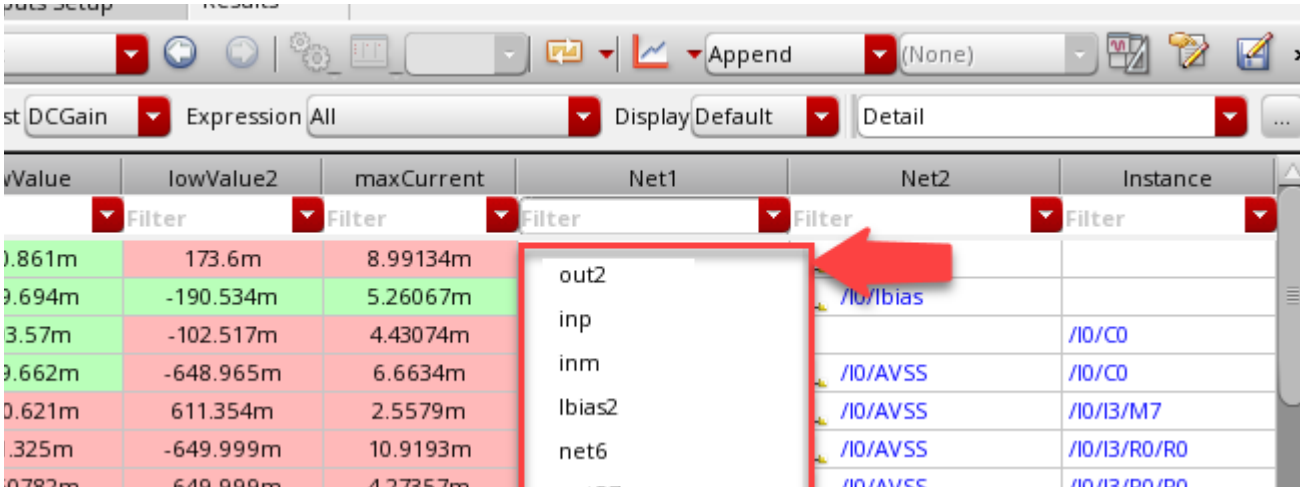
© 2023 Cadence Design Systems, Inc.

- **You can use filtering options from the context-sensitive menu** of the cell in which you applied a filter. These options are same as the filtering options in other result views.

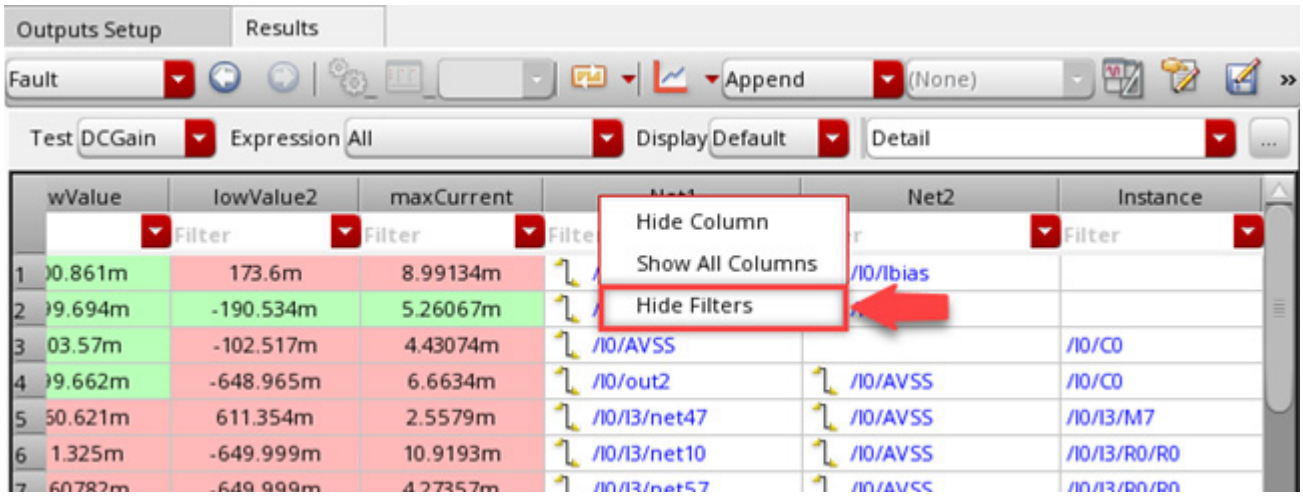


© 2023 Cadence Design Systems, Inc.

- **You can choose filters from the drop-down list.** As shown in the example given below, the list shows available values including the previously used filters and the unique values in that column.



- **You can clear the previously used filters for a column by using the Clear Search History command** in the context-sensitive menu of that column.
- **You can hide the filters** by using the Hide Filters command from the context-sensitive menu of the column headers.



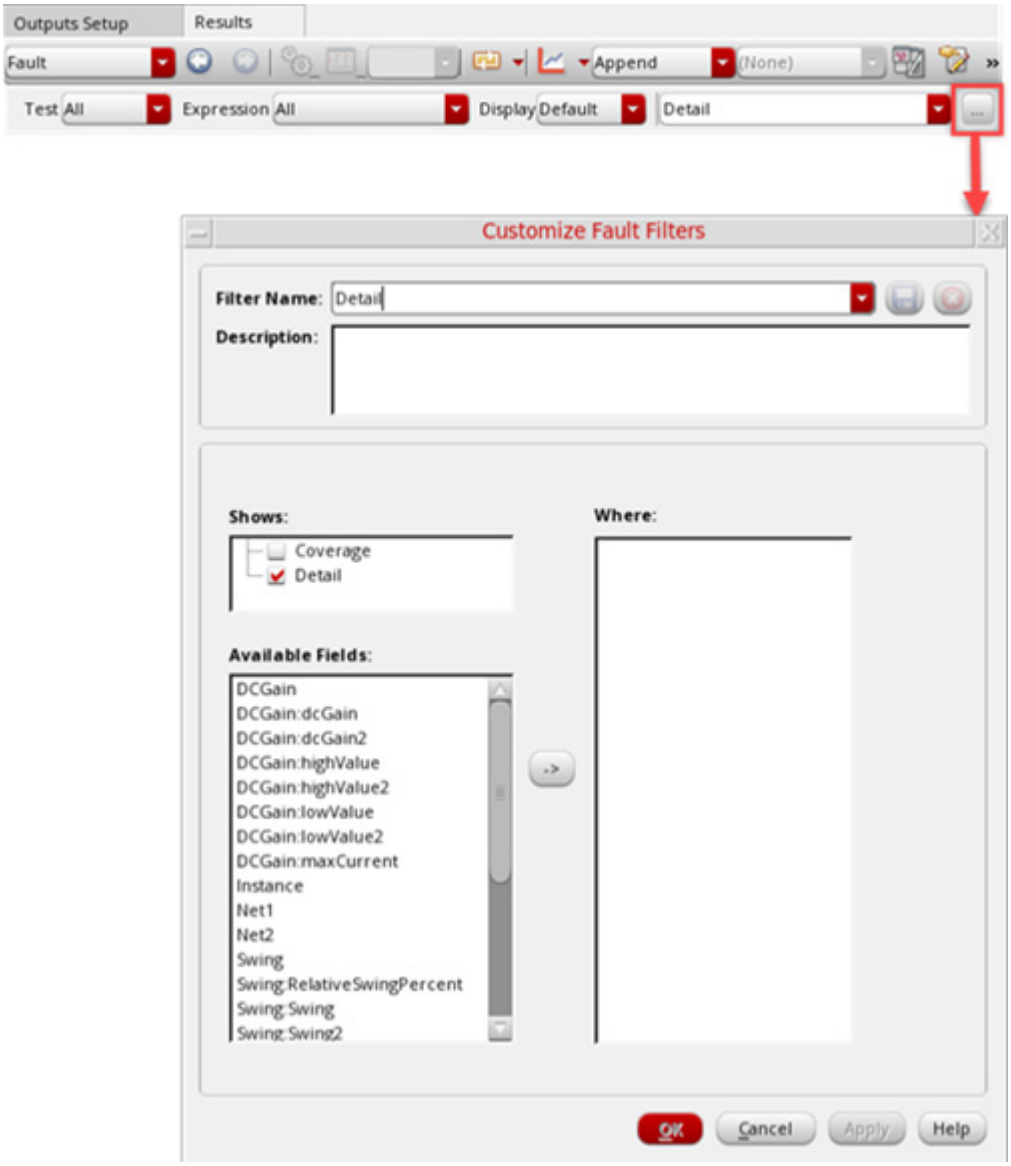
Creating a Custom Fault Filter

By default, the drop-down list to select a fault filter shows the two main types, `Coverage` and `Detail`. If required, you can create your own custom fault filters with different criteria and save them so that they can be reused for the same or different results.

© 2023 Cadence Design Systems, Inc.

To define and save a custom fault filter, perform the following steps:

1. Click *Edit or create custom violation filter* on the toolbar of the `Fault` results view to open Customize Fault Filters form.



2. Specify appropriate values in the following fields of this form.

Field	Description
`Filter Name`	The name of the custom filter

© 2023 Cadence Design Systems, Inc.

`Description`	The description for the custom filter
`Shows`	The type of report you want to view
`Available Fields`	The list of fields or columns available in the Fault result view
`Where`	Specifies the filter criteria. To define this criteria based on the values in a field or column, click the column name in the Available Field list and click the left arrow. The field name is added to the Where field. You can now build an expression to define the filter criteria.

3. Click `Save` on this form to save these filter settings in a custom file.

Important Points to Consider for Custom Filter Syntax:

- Instance and net names must use the simulator namespace, not the schematic namespace. For example, `I8.I0.MP1`is correct, whereas `/I8/I0/MP1` is incorrect.
- Query operators are case insensitive
- Strings can be in single or double quotes
- If the field names contain spaces or special characters, they must be enclosed in double quotes.
- Query operators for custom filters support the same SQLite operators that you can use on the Customize Violation Filters form for Check and Asserts.

© 2023 Cadence Design Systems, Inc.

Merging Results from Multiple Fault Simulations

You can merge the results of multiple fault simulation histories to view the consolidated results in one history.

This feature saves the time required to rerun simulations in the following scenarios:

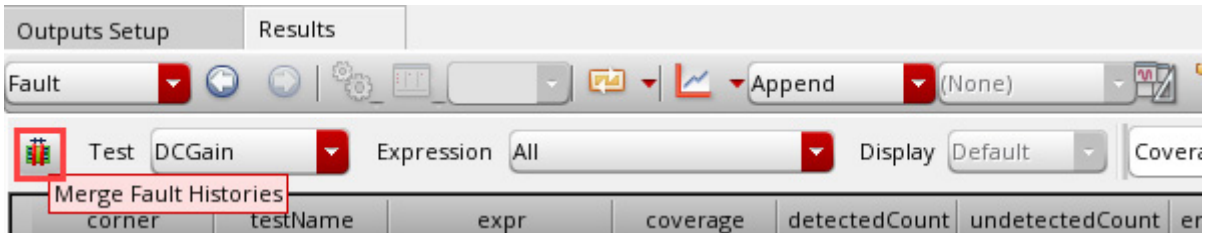
- After running a fault simulation, you modify the fault rules or the test setup, or make any other change in the active setup, and run simulation. You can merge the results saved before and after making these changes and view a consolidated report.
- You want to check the fault detection rate by dropping faults in subsequent runs, and then want to merge results of all runs. For more details, refer to Checking Fault Detection Rate.

Video

You can view a video demonstration of this feature at Merging Fault Simulation Results.

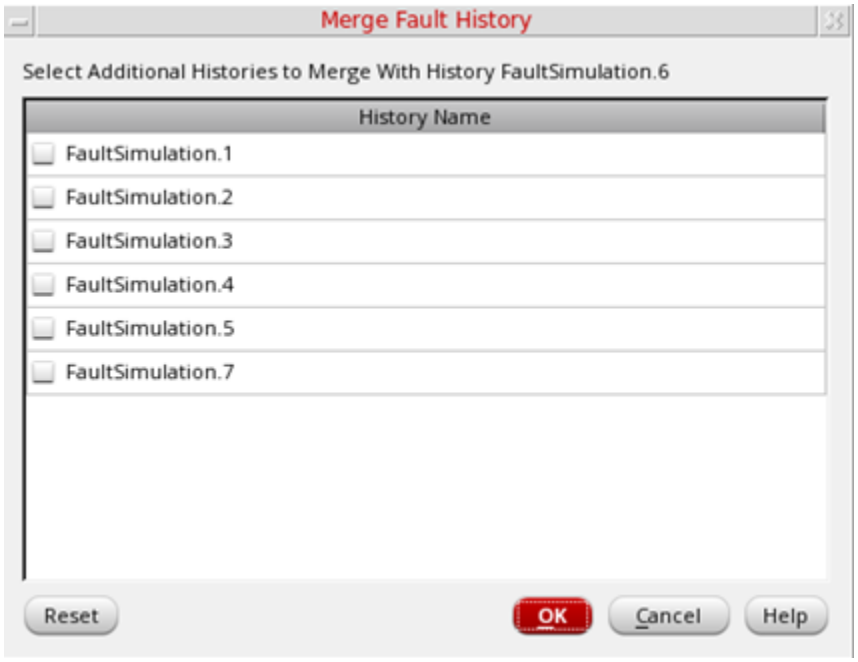
To merge results of multiple fault simulations:

1. Open the history of the fault simulation that you want to update by merging results from other histories into it.
2. Open the *Fault* results view.
3. Click *Merge Fault Histories* on the toolbar of the *Results* tab.



© 2023 Cadence Design Systems, Inc.

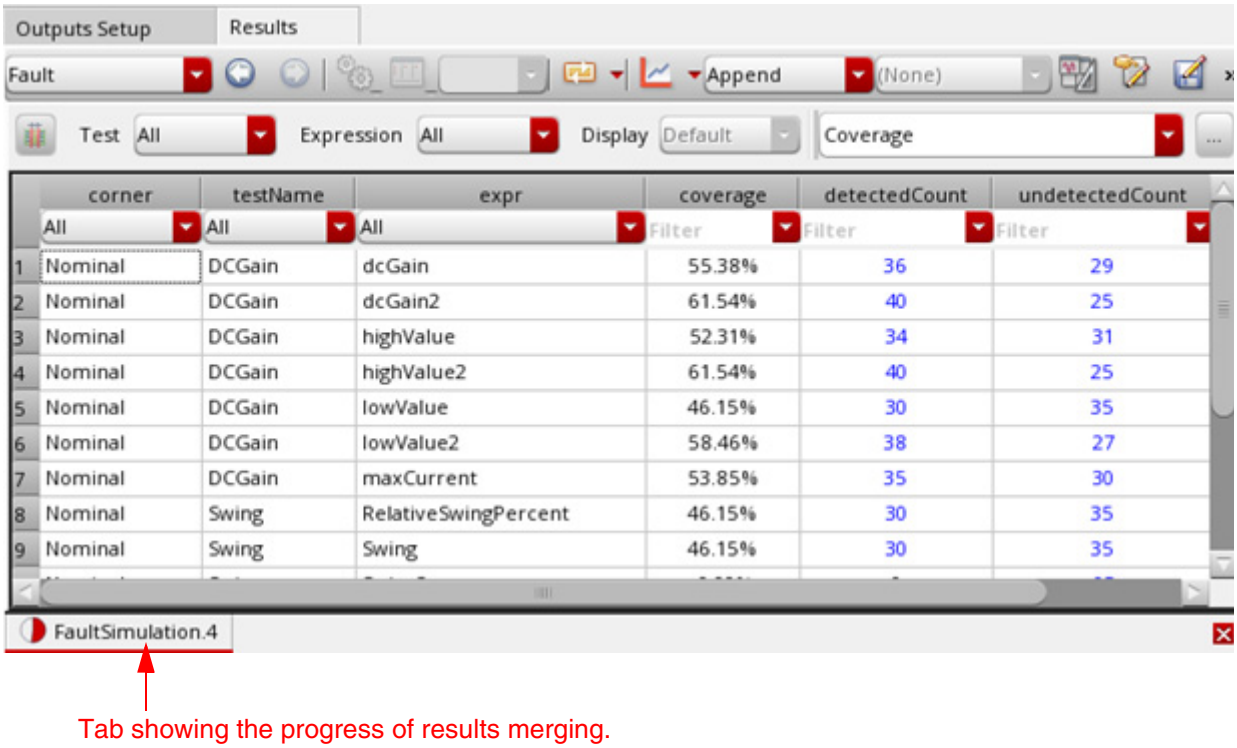
4. The Merge Fault History form is displayed. It shows the list of all available histories of fault simulation run for this testbench, as shown in the following example.



5. Select one or more histories from which you need to merge the results.
6. Click *OK*.

© 2023 Cadence Design Systems, Inc.

7. The results of all selected histories are merged in the history for which you displayed results. The history name on the *Results* tab shows the progress of merging.

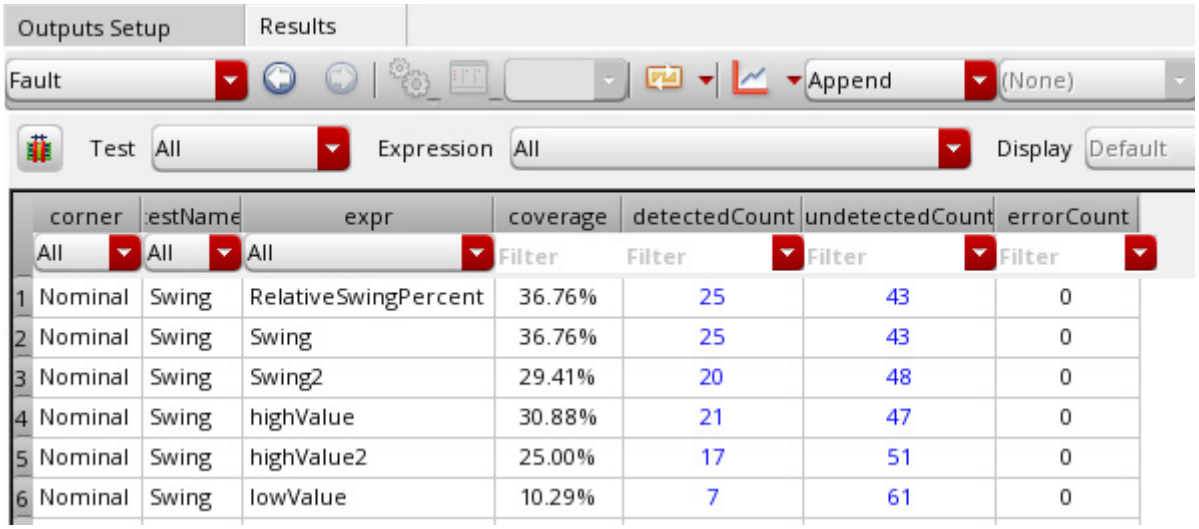


   After the merge is successful, the count of detected and undetected faults are updated to reflect the merged data.

The following example shows the benefit of merging fault simulation results saved for two different tests separately:

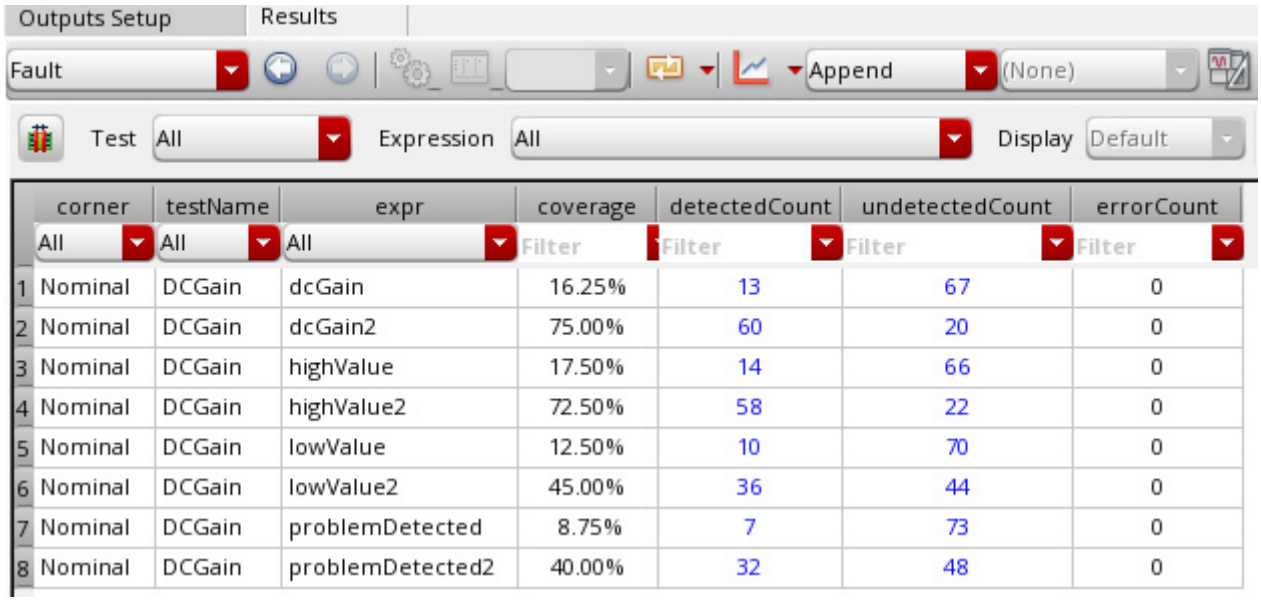
© 2023 Cadence Design Systems, Inc.

Original Results of `FaultSimulation.0`:



	corner	testName	expr	coverage	detectedCount	undetectedCount	errorCount
	All	All	All	Filter	Filter	Filter	Filter
1	Nominal	Swing	RelativeSwingPercent	36.76%	25	43	0
2	Nominal	Swing	Swing	36.76%	25	43	0
3	Nominal	Swing	Swing2	29.41%	20	48	0
4	Nominal	Swing	highValue	30.88%	21	47	0
5	Nominal	Swing	highValue2	25.00%	17	51	0
6	Nominal	Swing	lowValue	10.29%	7	61	0

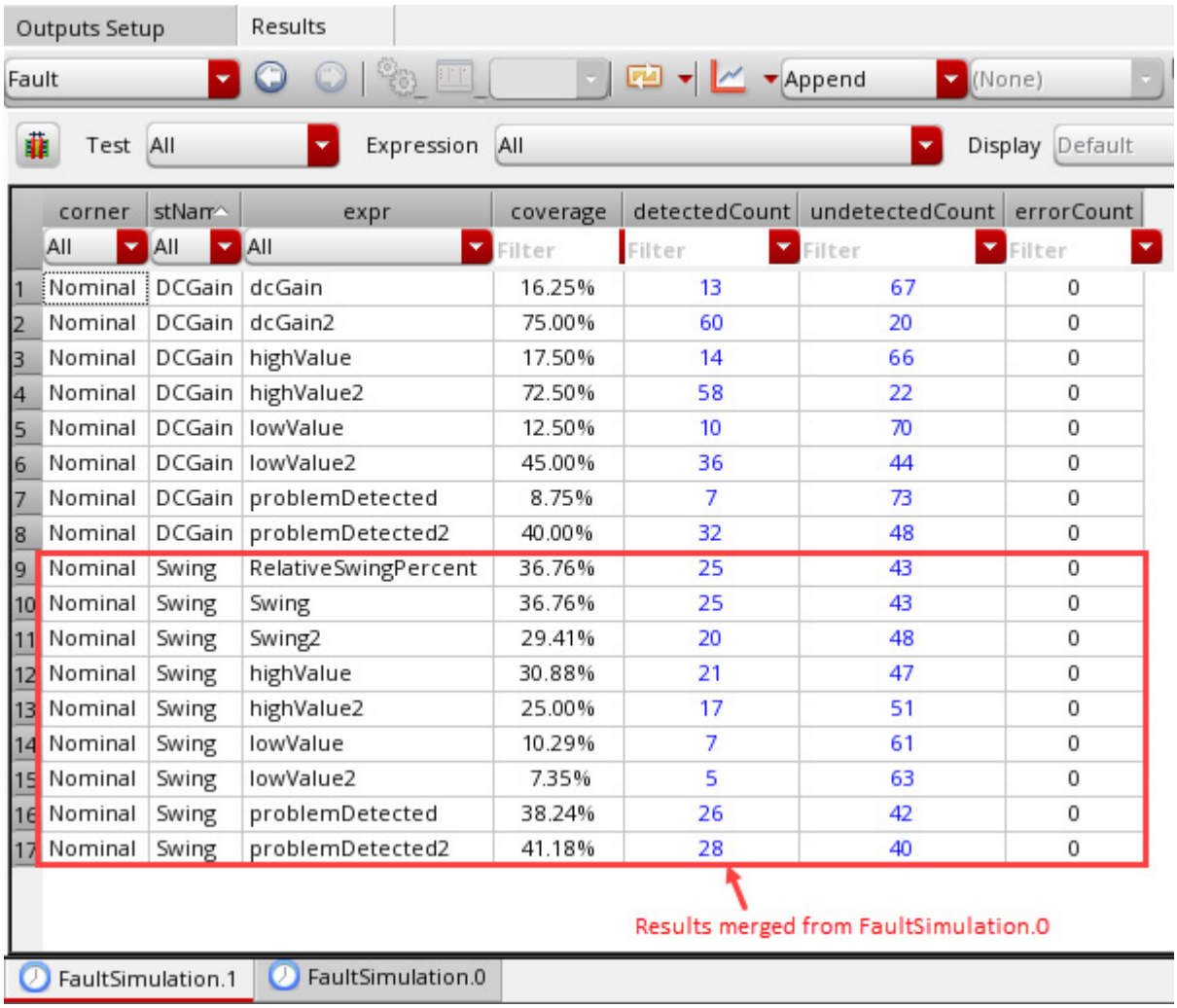
Results of `FaultSimulation.1`:



	corner	testName	expr	coverage	detectedCount	undetectedCount	errorCount
	All	All	All	Filter	Filter	Filter	Filter
1	Nominal	DCGain	dcGain	16.25%	13	67	0
2	Nominal	DCGain	dcGain2	75.00%	60	20	0
3	Nominal	DCGain	highValue	17.50%	14	66	0
4	Nominal	DCGain	highValue2	72.50%	58	22	0
5	Nominal	DCGain	lowValue	12.50%	10	70	0
6	Nominal	DCGain	lowValue2	45.00%	36	44	0
7	Nominal	DCGain	problemDetected	8.75%	7	73	0
8	Nominal	DCGain	problemDetected2	40.00%	32	48	0

© 2023 Cadence Design Systems, Inc.

Results of `FaultSimulation.1` after `FaultSimulation.0` is merged with it:



Important Points to Note

- After the results are merged in a history, they are saved in the results database of the history. Next time, when you open the same history again, it shows the merged data.
- To check whether a history contains merged data, open its results and click *Merge Results* on the toolbar of the *Results* tab. If the results are already merged, the history names are shown as selected.
- To rollback to the original version of a history, that is, to remove the merged data, open the results and click *Merge Results* on the toolbar of the *Results* tab. Click *Reset* on the Merge Fault History form to deselect all selected histories. ADE Assembler removes the results that were merged from other histories.

© 2023 Cadence Design Systems, Inc.

- It is possible to remove the merged data for only a few selected histories instead of all histories. In that case, you need to individually deselect the history names on the Merge Fault History form instead of clicking *Reset*.

Related Topics

dropUndetectedSimErrorsInFaultDroppingFlow

handleErrorsAs

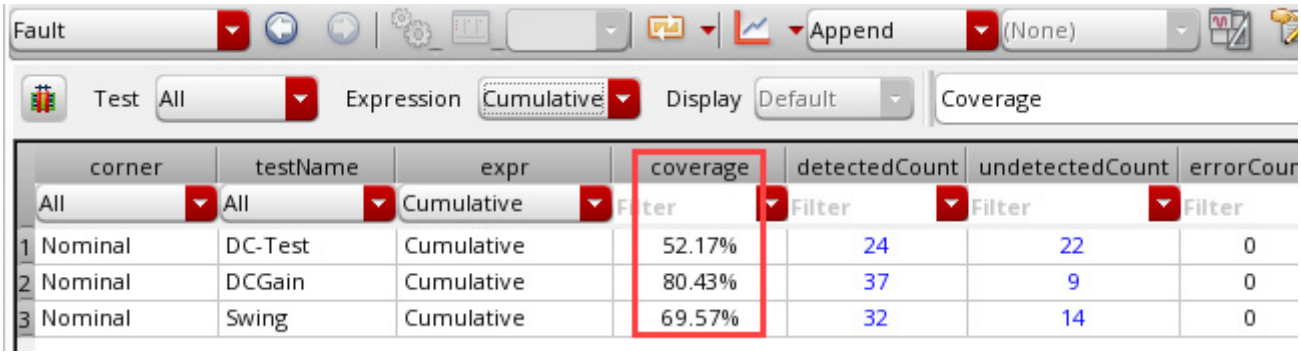
handleSimulationErrorsAs

Checking Fault Detection Rate

Fault simulation runs are generally time consuming because of a large number of faults in the design. Ensuring an effective fault coverage would help in saving time.

In a multi-test testbench, you can check the fault detection rate for each test by running simulation for each test individually and dropping the faults detected for one test in the fault setup for other tests. You can then merge the results of all tests and review the cumulative report to calculate the effective overall coverage.

Consider an example testbench with three tests: DCGain, DC-Test, and Swing. The following figure shows the cumulative coverage of faults when a simulation is run with all three tests enabled in this example:



	corner	testName	expr	coverage	detectedCount	undetectedCount	errorCour
	All	All	Cumulative	Filter	Filter	Filter	Filter
1	Nominal	DC-Test	Cumulative	52.17%	24	22	0
2	Nominal	DCGain	Cumulative	80.43%	37	9	0
3	Nominal	Swing	Cumulative	69.57%	32	14	0

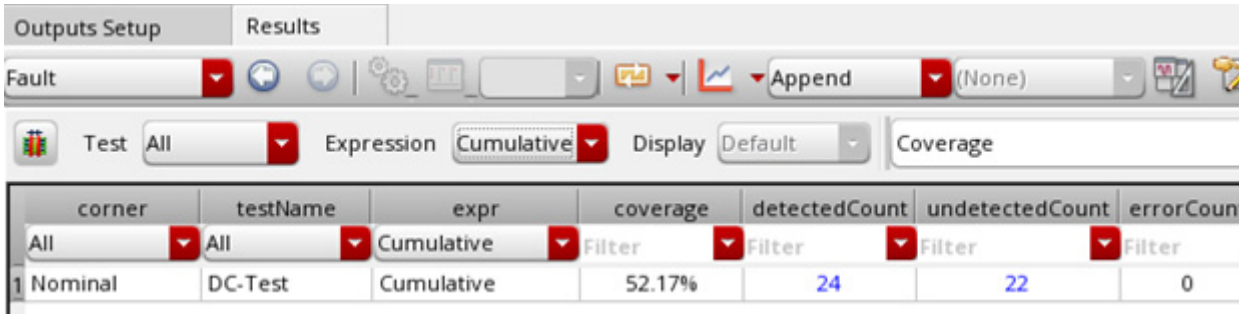
To check the effective coverage of faults:

1. Create a new fault group, say `undetectedFaultGrp`, in the Fault Setup assistant to save all undetected faults.
2. Select a test in the Data View assistant and deselect other tests.

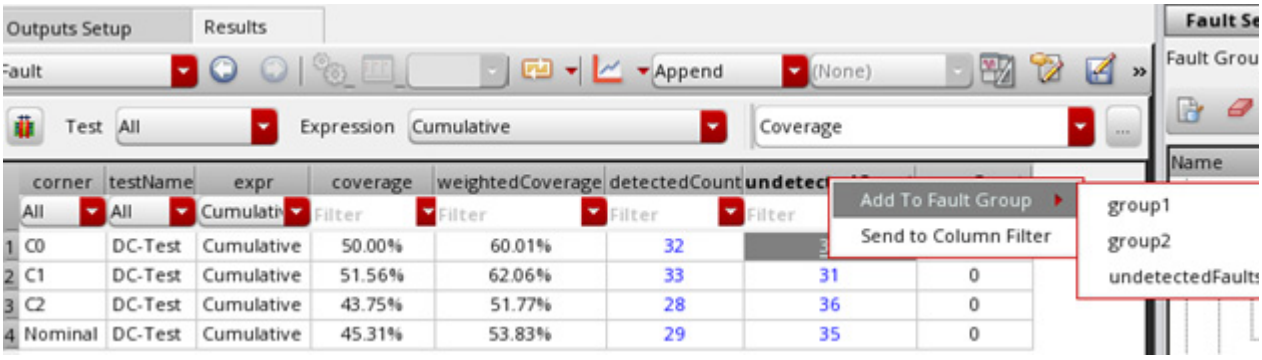
© 2023 Cadence Design Systems, Inc.

In the first run, it is recommended to select a test that has the maximum coverage of faults and minimum run time according to your estimate or observation.

3. Using the complete fault rule setup that was used for simulation with all tests enabled, run a fault simulation.
4. To view the cumulative coverage of faults, open the `Fault` results view and select `Cumulative` from the Filter list for the *expr* column.



5. Right-click the value in the `undetectedCount` column and choose *Add to Fault Group* and then the name of the fault group you created in step 1.



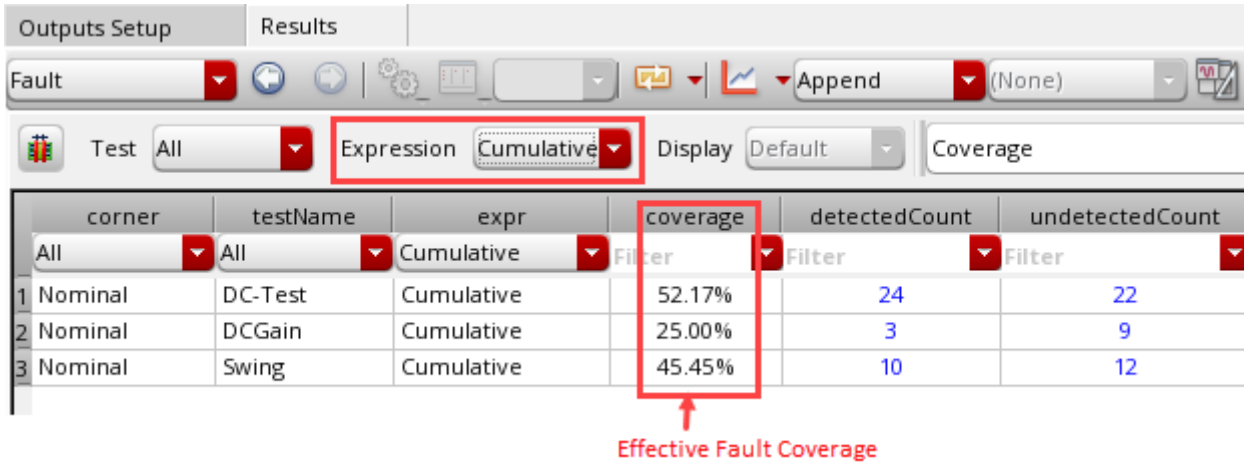
6. Choose the newly created fault group in the *Fault Group to Run* drop-down list in the Fault Setup assistant.
7. Disable the test already used in steps 2 to 5.
8. Select one of the remaining tests and repeat steps 3 to 6.

   In each iteration, you run fault simulations by selecting one test at a time and using the undetected fault list from the previous runs. In every run, you drop the faults already detected in the test used for the previous simulation run. The next run considers only the undetected faults.

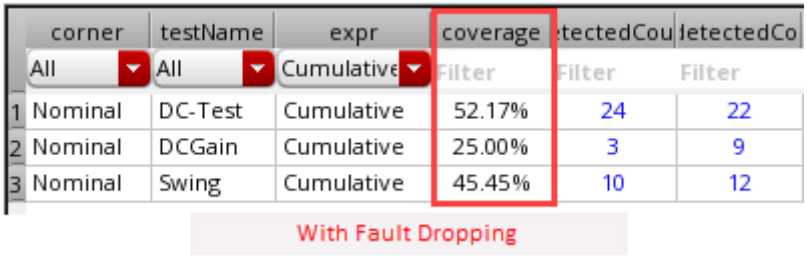
© 2023 Cadence Design Systems, Inc.

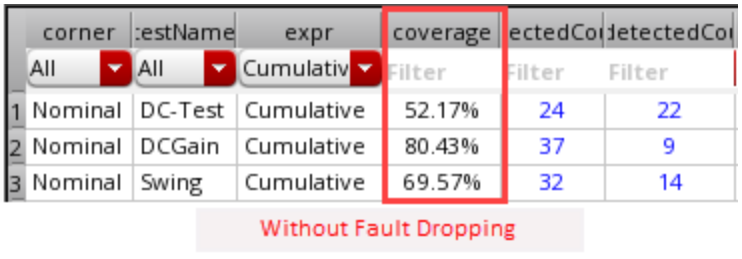
9. After the simulation for all tests are complete, merge the results in one history. For details on how to merge faults, refer to Merging Results from Multiple Fault Simulations.

10. Select `Cumulative` from the *Expression* drop-down list in the Fault results view.



Observe the difference in the fault coverage for each test with the initial coverage reported by the run with three tests.

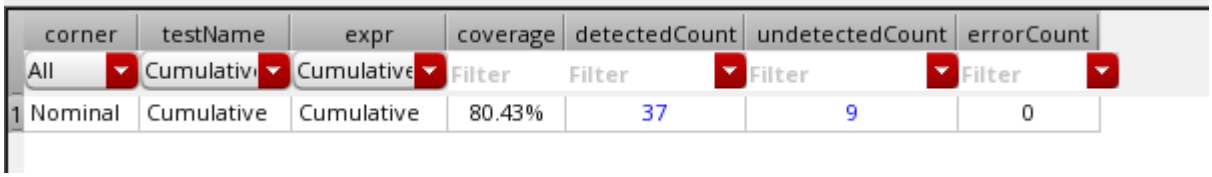




The coverage for tests `DCGain` and `Swing` are different because by default, a fault is counted as detected for all tests if it is detected by any one of them and it is reflected in the coverage for each test. By dropping the faults detected by one test from the simulations for other tests, you create a unique fault set for each test.

© 2023 Cadence Design Systems, Inc.

The fault set created after fault dropping is as effective as the fault set without fault dropping. In the previous example, the cumulative fault coverage for both cases, with or without fault dropping is same, `80.43%`.



	corner	testName	expr	coverage	detectedCount	undetectedCount	errorCount
	All	Cumulativ	Cumulative	Filter	Filter	Filter	Filter
1	Nominal	Cumulative	Cumulative	80.43%	37	9	0

Therefore, it is important to understand that fault dropping is not essential. You can evaluate whether fault dropping improves the run time for your designs or not.

© 2023 Cadence Design Systems, Inc.

Automatic Fault Dropping Using a Run Plan

You can create a run plan using the fault setup to run multiple simulations, one for each test, in a sequence where each subsequent run in the plan uses a reduced list of faults that were not detected in the previous run. This way of dropping faults that are detected by the previous run helps in focusing on fault coverage and also saves time because the simulation is not run for the faults that are already detected.

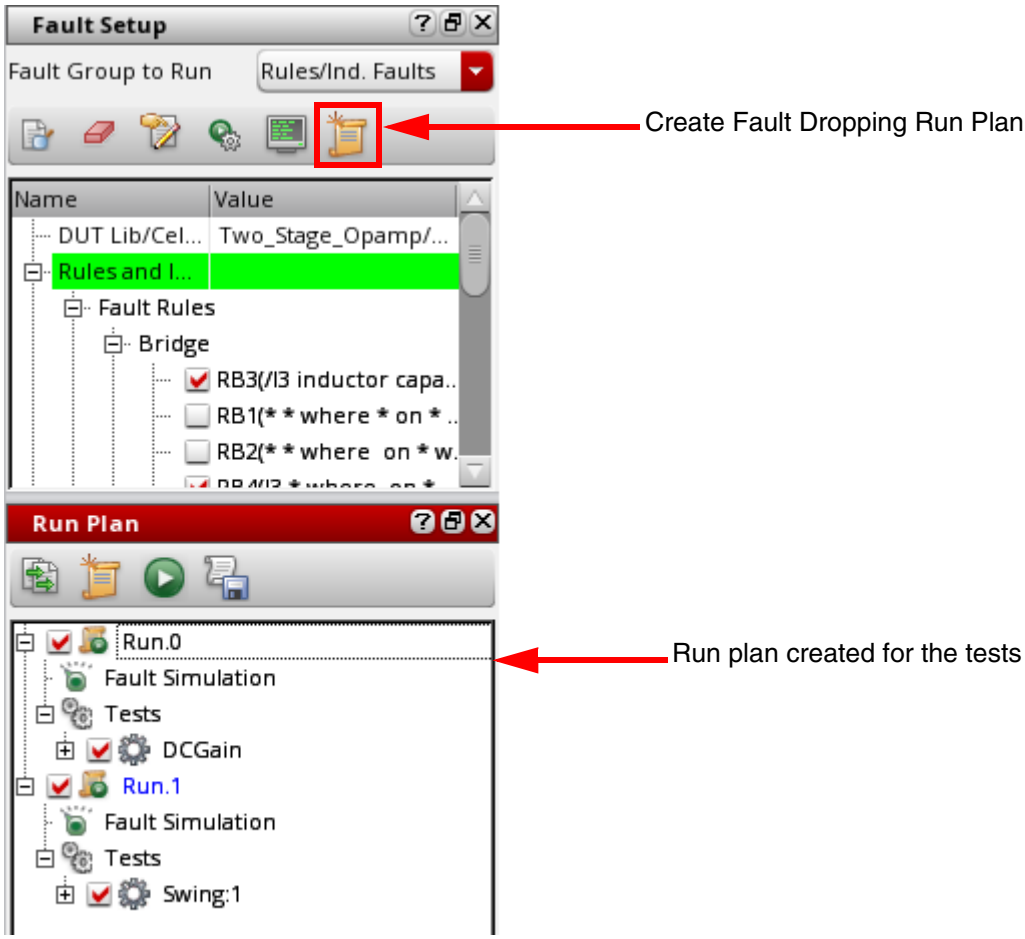
Creating a Run Plan for Fault Dropping

To create a run plan for automatic fault dropping:

1. Ensure that the maestro cellview contains multiple tests that refer to the same design under test. Each test can have a different setup.
2. In the Data View assistant, select the tests for which you want to run fault simulation.
3. Ensure that the selected run mode is *Fault Simulation*.
4. In the Fault Setup assistant, prepare a fault setup using fault rules and individual faults as described in Setting up Faults.
5. Click *Create Fault Dropping Run Plan* on the Fault Setup assistant toolbar.

© 2023 Cadence Design Systems, Inc.

ADE Assembler opens the Run Plan assistant and creates a run plan by using the tests enabled in the Data View assistant.



For each run in the plan, ADE Assembler automatically adds a pre-run script and a post-run script to prepare a fault list for the next run in the plan.

If a run plan already exists in the Run Plan assistant, runs from that run plan are disabled.

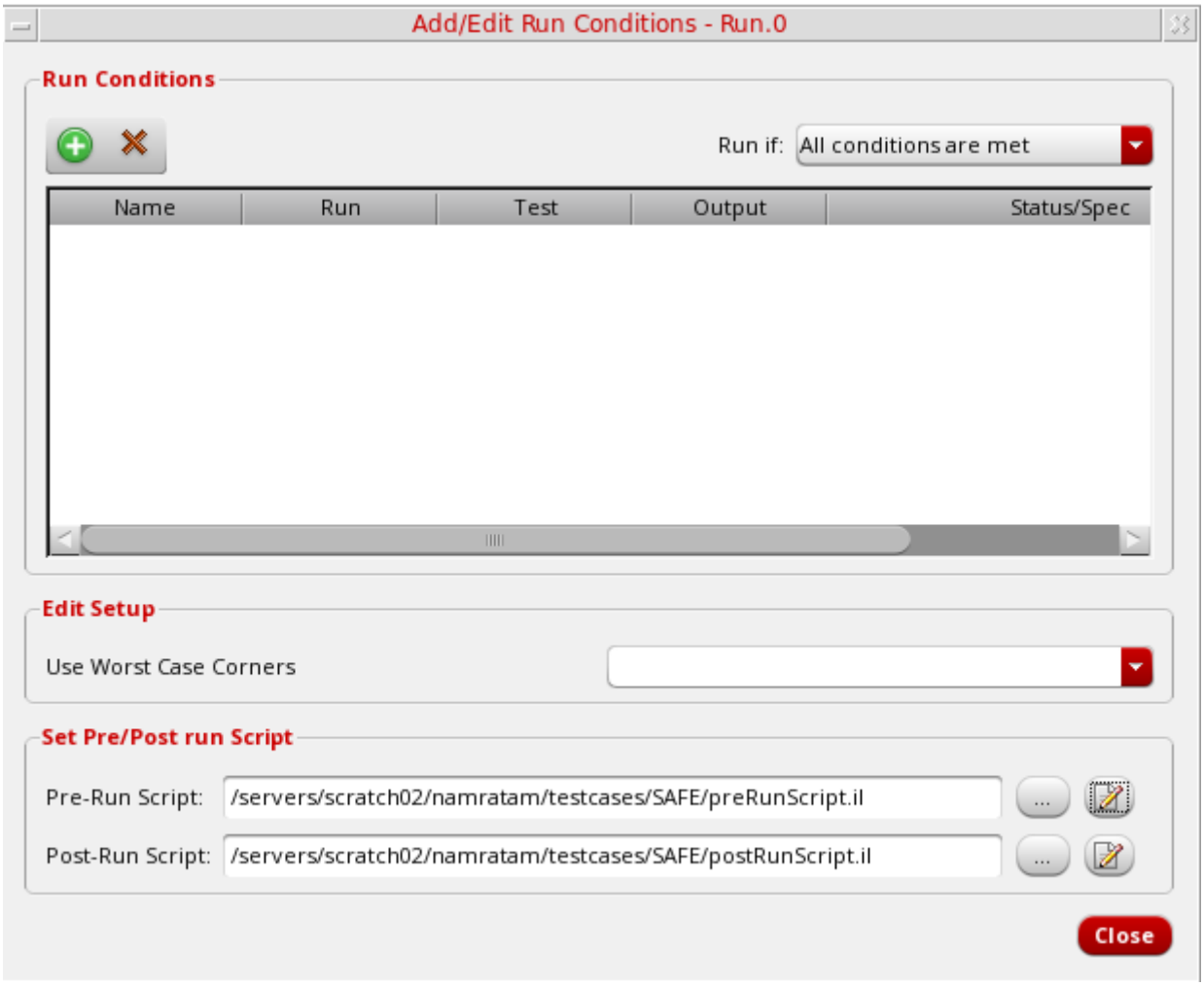
Note: If only one test is enabled in the setup, the run plan is not created and an error message is displayed.

6. (Optional) To review the pre- and post-run scripts, right click on any run in the Run Plan assistant and choose *Edit Run Conditions*.

   The Add/Edit Run Conditions form is displayed for the run. The path to the scripts is shown in the *Pre-Run Script* and *Post-Run Script* fields. You can open the script files

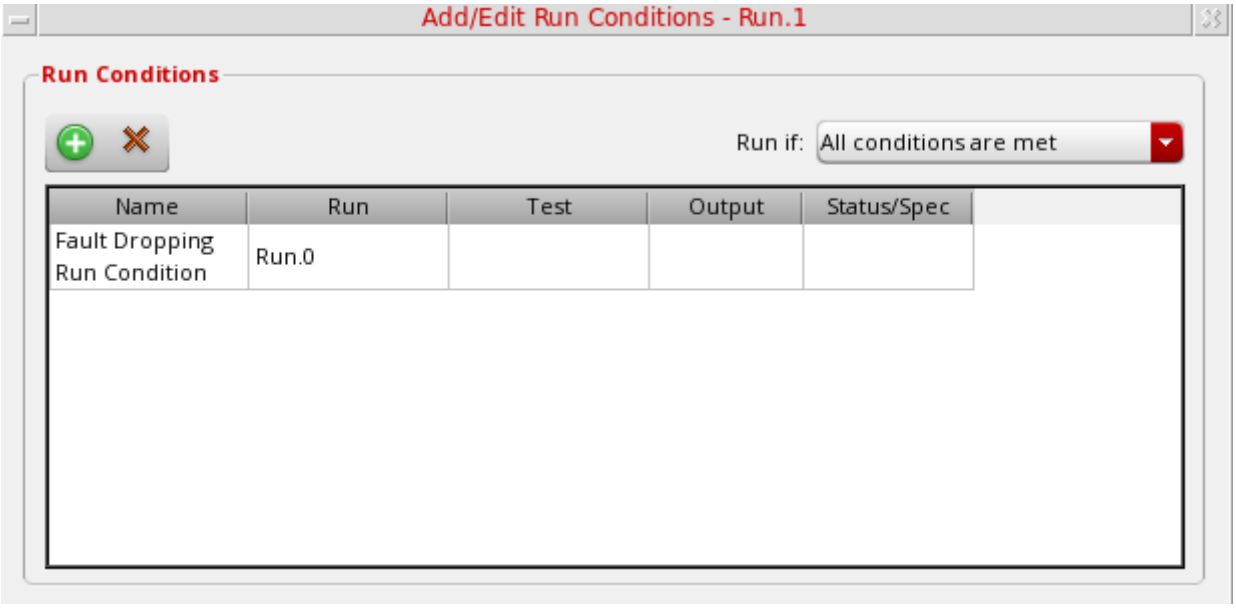
© 2023 Cadence Design Systems, Inc.

to review and if required, modify their contents by using the SKILL functions related to fault simulations. However, it is recommended not to modify these files.



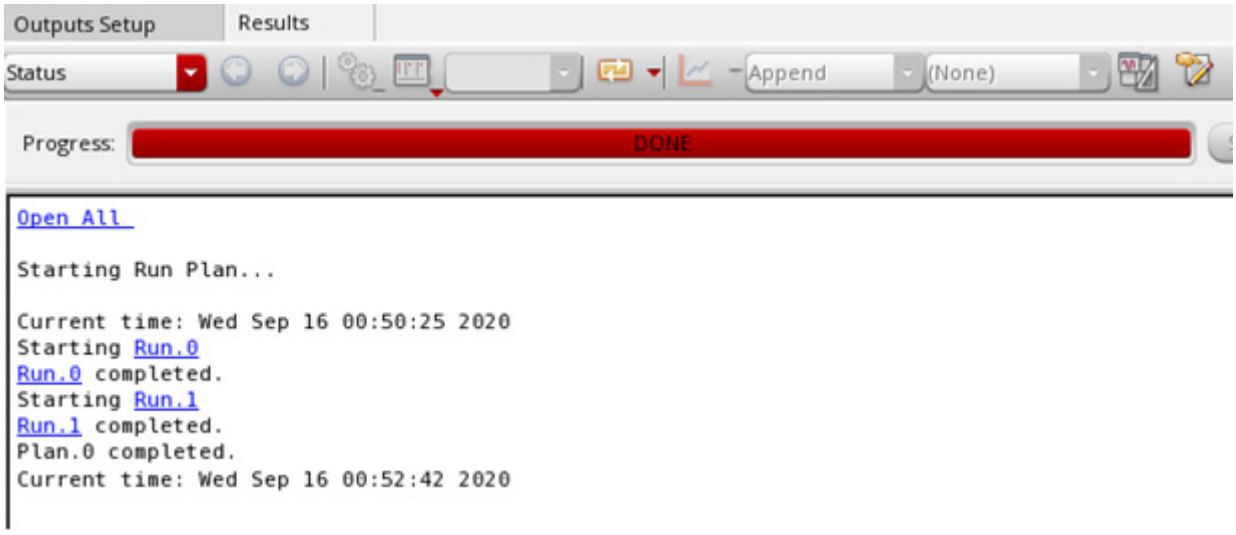
© 2023 Cadence Design Systems, Inc.

For all runs except the first run in the plan, this form also adds a run condition to check the success status of the previous run before starting the next run.



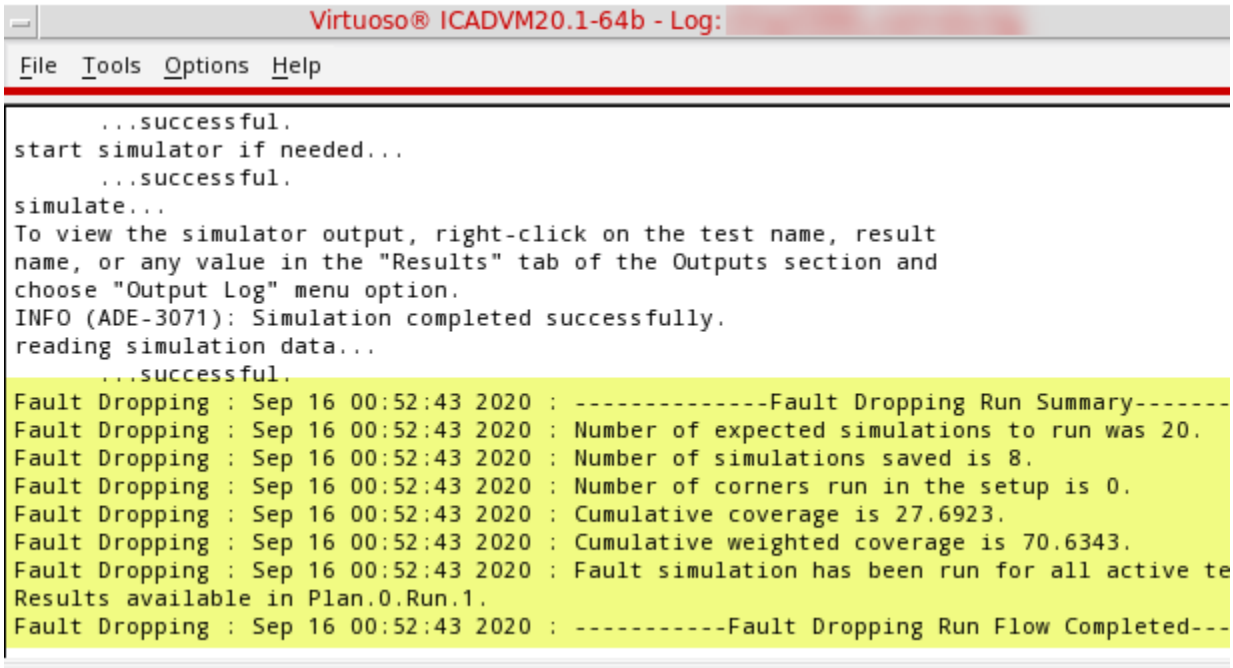
7. Close the Add/Edit Run Conditions form.
8. Click *Execute Run* on the Run Plan assistant toolbar to start a simulation for the run plan.

   The *Results* tab opens the *Status* results view. This view shows the status of each run in the run plan.



© 2023 Cadence Design Systems, Inc.

The CIW also shows the log for the fault dropping flow.



This log indicates the number of simulations saved by dropping faults in the flow. In the above example, the total fault count visible in the Fault Preview form was 10. If all fault points were run for both tests, a total of 10 simulations would have been required. However, the log shows that, fault dropping saved 8 simulations. In addition, this log shows the coverage statistics. These logs are also saved in the `_faultDropping.logs` file in the current work directory.

You can now analyze the detailed results in the *Fault* results view.

© 2023 Cadence Design Systems, Inc.

Running Simulations Using Selected Faults from Results

Based on your analysis of output values, you might need to run a simulation using only the fault points corresponding to a few selected output values. You can create a fault group for those fault points and run simulation for that group.

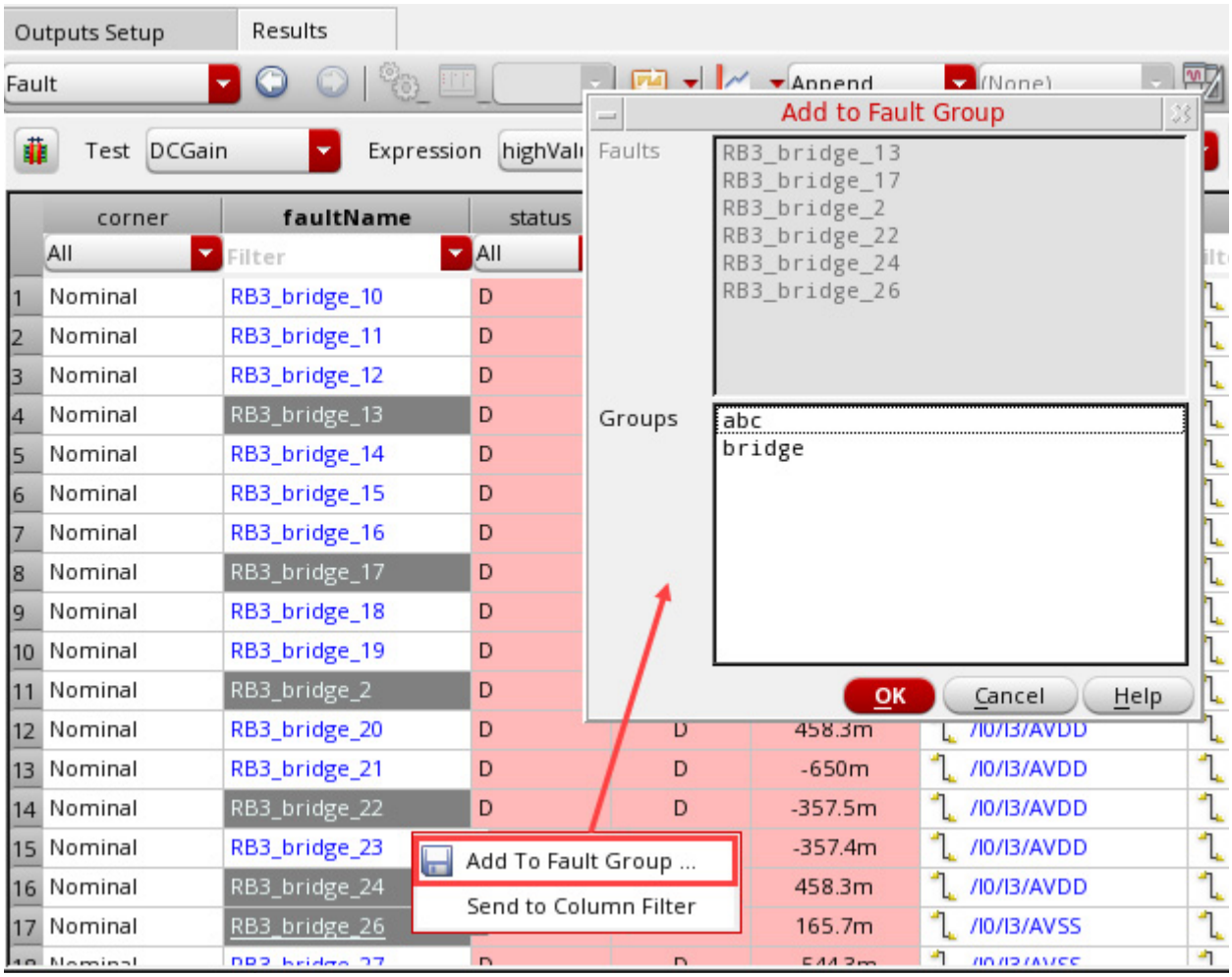
To add faults from simulation results to a fault group:

1. Identify a fault group to which you want to add faults.
2. Open the *Fault* or *Detail–Transpose* results view.
3. Select any cell in the row of the fault that you want to add to a group.

   To add multiple faults, hold the *Ctrl* key and select the cells for those faults.
4. Right-click any one of the selected cells and choose *Add to Fault Group*.

© 2023 Cadence Design Systems, Inc.

The Add to Fault Group form appears. The *Faults* list shows the names of the selected faults, and the *Groups* list shows the names of available fault groups.



5. Select the name of a fault group.
6. Click *OK*.

The faults are added to the group. You can select the group in the *Fault Group to Run* list on the Fault Setup assistant and then run a simulation.

Debugging Results and Troubleshooting Errors

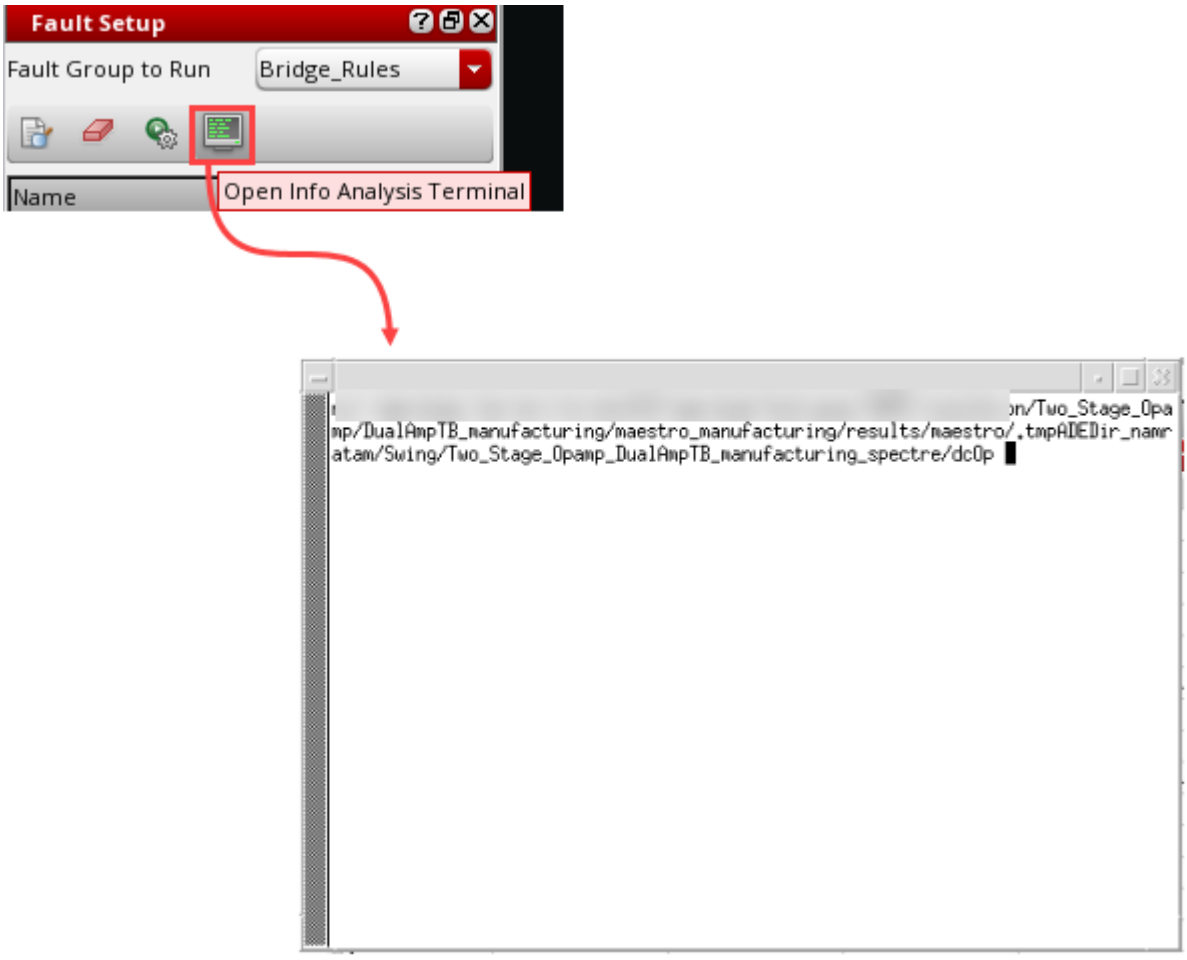
If the design contains dummy devices, Spectre would ignore those because they are not relevant for the functionality of the circuit. However, the devices are still listed in the faults list created by ADE Assembler. This might result into simulation errors. To avoid this, make sure that Spectre does not skip dummy devices.

© 2023 Cadence Design Systems, Inc.

Debugging Results

While debugging simulation results, if you need to access netlist and output log of Spectre info analysis, click *Open Info Analysis Results* on the toolbar of the Fault Setup assistant.

This command opens the directory where the netlist and the `psf` directory of the info analysis is saved.

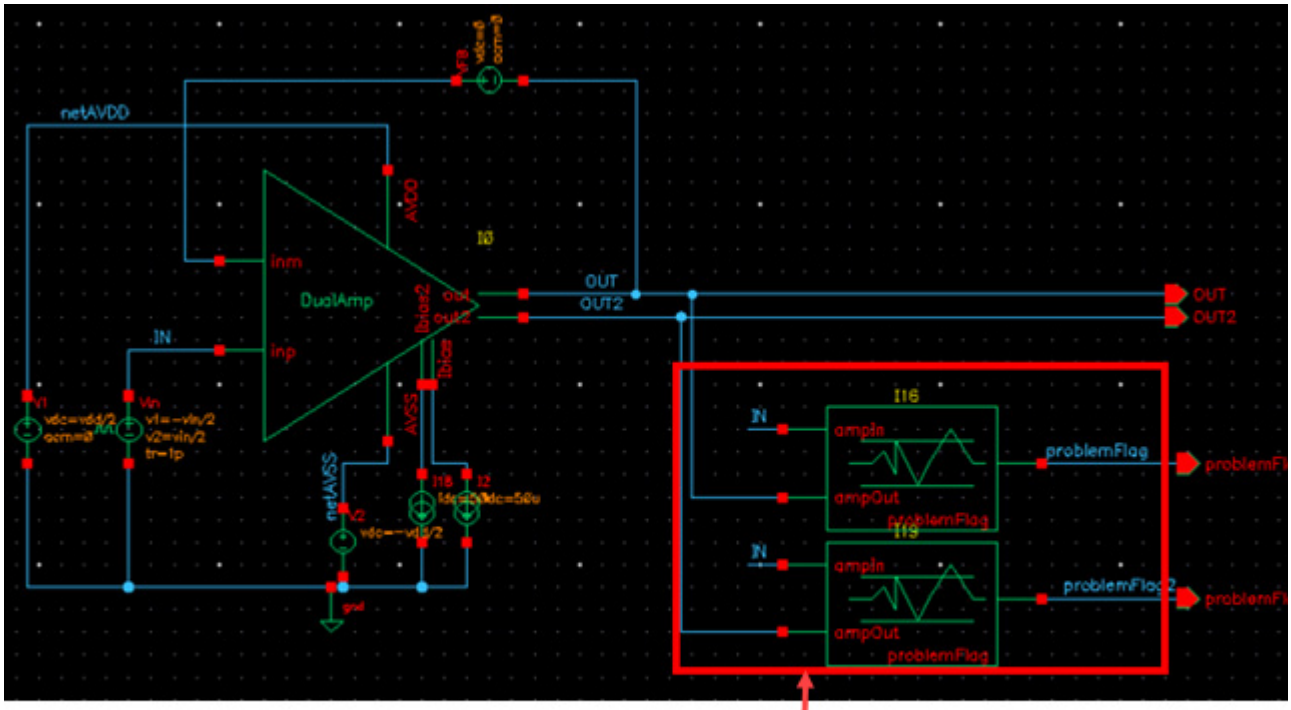


© 2023 Cadence Design Systems, Inc.

Running Functional Safety Checks

An important reliability check for a device is its functional safety. This check mainly focuses on the effect of a failure during the lifetime of a device.

To generate this report, you need to have checker blocks in your design to validate the outputs of the functional block.

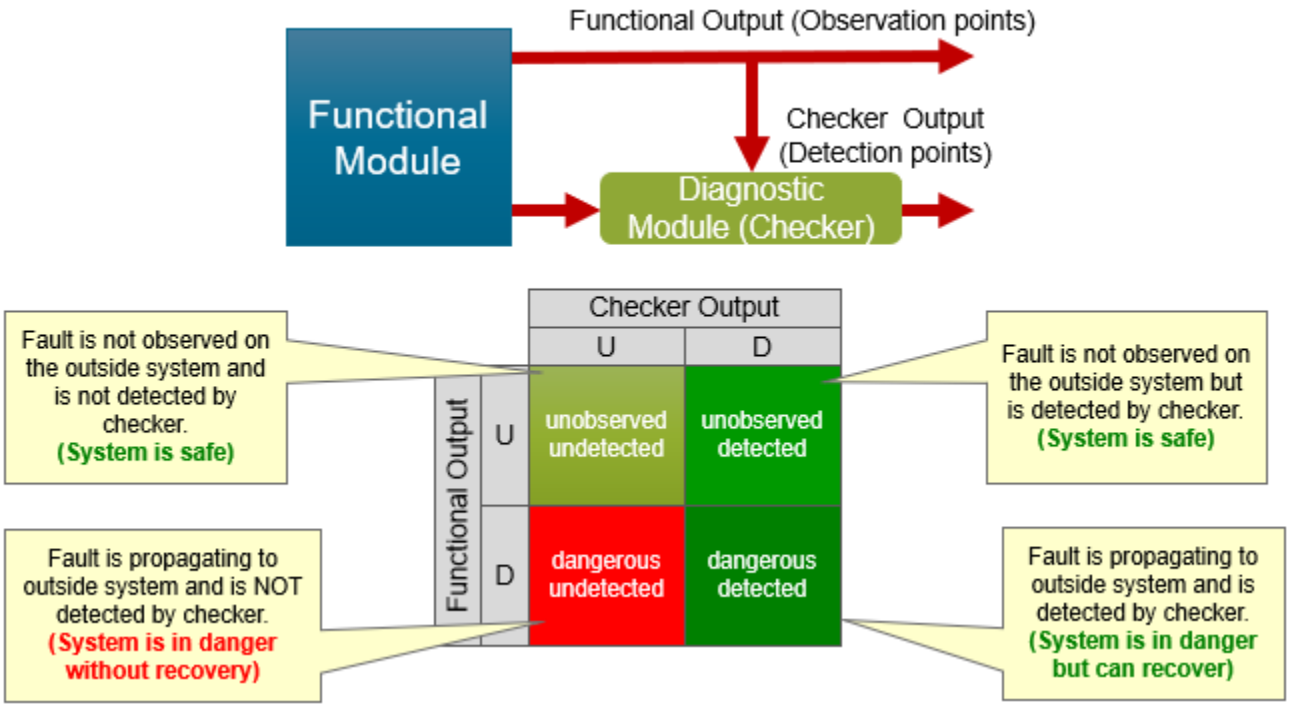


Diagnostic or Checker Blocks

If the fault detection is enabled for both functional and checker blocks, you can generate a functional safety report that is aligned with the safety classification defined in the ISO 26262 standard, which classifies safe or dangerous, and detected or undetected defects.

© 2023 Cadence Design Systems, Inc.

The following diagram describes the various types of functional safety categories defined according to this classification.



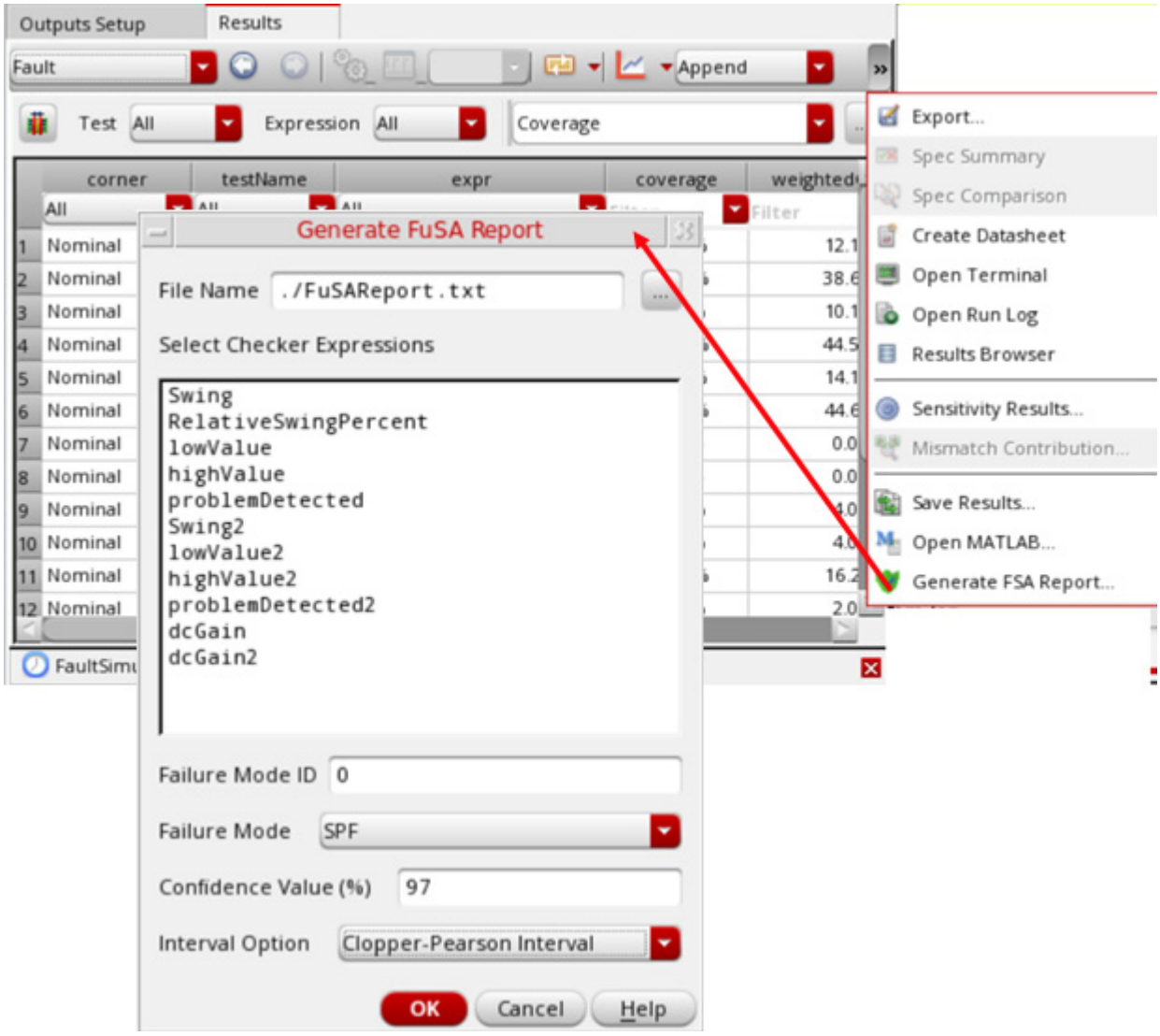
To generate the functional safety report, perform the following steps:

1. Open the results of a fault simulation.
2. Click *Generate FSA Report* ( ) on the toolbar of the *Results* tab.

   The Generate FuSA Report form is displayed.
3. In the *File Name* field, specify the file name and path to be used to save the report.

© 2023 Cadence Design Systems, Inc.

4. In the *Select Checker Expression* list, select the names of the checker expressions you created to check the output of the checker blocks in the design.



5. In the *Failure Mode ID* field, specify an integer ID of the failure mode being addressed with this report.
6. From the *Failure Mode* drop-down list, choose one of the following failure modes:
   - SPF: Single Point Failure
   - LFC: Latent Fault Continuous
   - LFP: Latent Fault Periodic

© 2023 Cadence Design Systems, Inc.

Single point failures are faults in a functional or checker module. In this mode, only a single failure, that is either the functional block or the checker block is considered at a time.

Latent faults are multiple-point faults where the effect of a faulty checker module is evaluated in the context of a faulty functional module. A common method of calculating latent fault results is to apply all relevant failure modes of the functional module to the input of the checker module and then run fault simulation only on the checker module.

Latent faults can be of two types, continuous (LFC) and periodic (LFP).

To identify a latent fault of continuous type, the output of the functional module needs to be monitored continuously by means of a continuous detection in the checker block.

To identify a latent fault of periodic type, the checker module is periodically turned on and off to monitor the output of the functional block.

For more details on these failure modes, refer to Fault Analysis.

Note: The values from the *Failure Mode ID* and *Failure Mode* fields are printed at the top of the functional safety report. These are not used in the calculations done for report generation.

7. In the *Confidence Value* field, specify a confidence level in percentage.

   For the `randomuniform` or `randomweighted` sampling methods, Spectre internally uses the Clopper-Pearson approach to estimate the confidence interval when the sampling based simulation is finished. The confidence level you specify in this field is used to show the corresponding confidence intervals in the fault detection matrix.

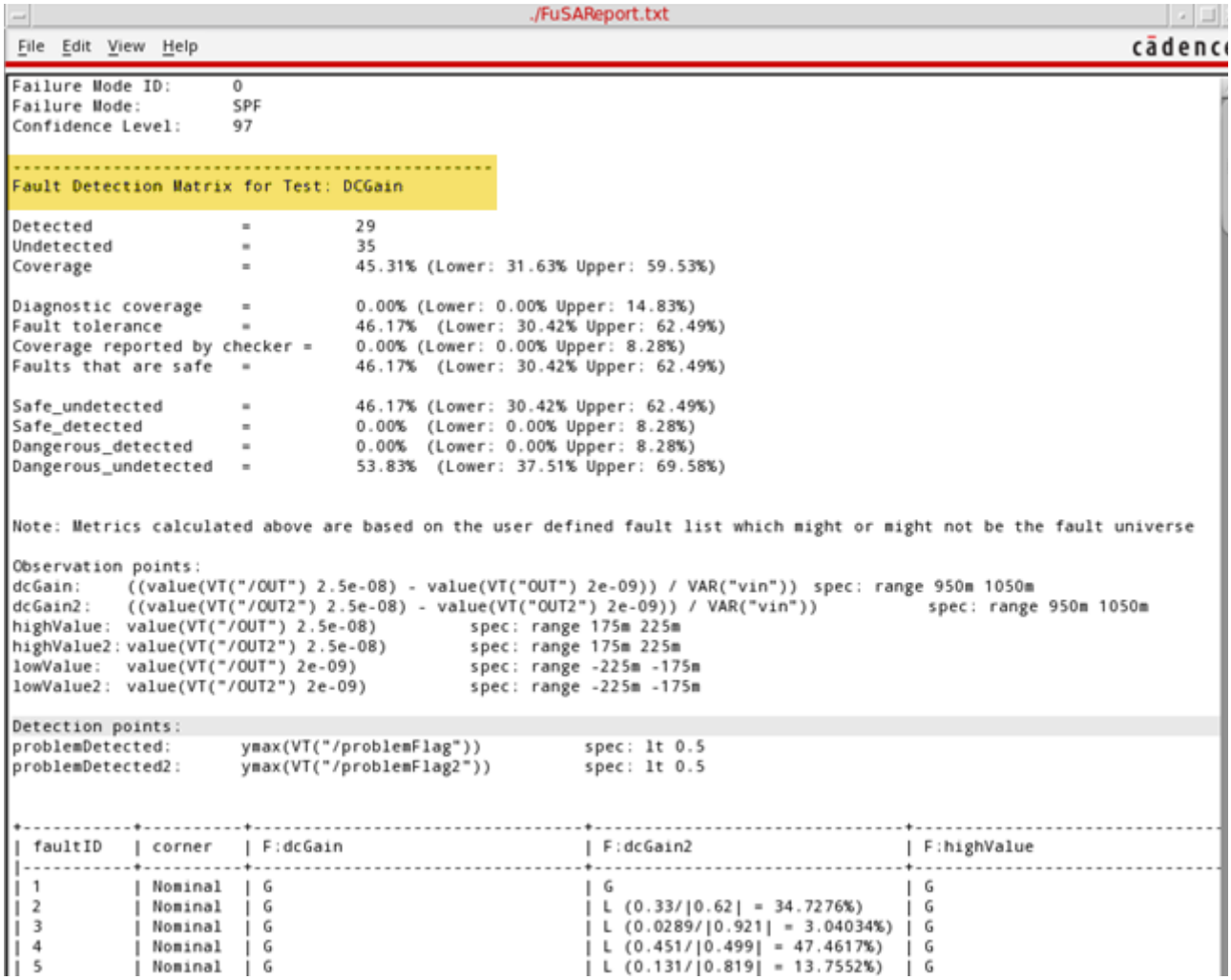
   For more details about how Spectre calculates confidence interval, see Calculating Confidence Interval for Fault Sampling.

8. From the *Interval Option* drop-down list, choose a formula to be used for confidence interval calculation. Both, `Clopper-Pearson Interval` and `Wald Interval` methods are supported.

9. Click *OK*.

© 2023 Cadence Design Systems, Inc.

The resultant report shows the fault detection matrix for each test followed by an aggregate of detected or undetected, safe and unsafe defects.



./FuSAReport.txt

File Edit View Help

```
Failure Mode ID:       0
Failure Mode:          SPF
Confidence Level:      97

------------------------------------------------
Fault Detection Matrix for Test: DCGain

Detected                =          29
Undetected              =          35
Coverage                =          45.31% (Lower: 31.63% Upper: 59.53%)

Diagnostic coverage     =          0.00% (Lower: 0.00% Upper: 14.83%)
Fault tolerance         =          46.17%  (Lower: 30.42% Upper: 62.49%)
Coverage reported by checker =     0.00% (Lower: 0.00% Upper: 8.28%)
Faults that are safe    =          46.17%  (Lower: 30.42% Upper: 62.49%)

Safe_undetected         =          46.17% (Lower: 30.42% Upper: 62.49%)
Safe_detected           =          0.00%  (Lower: 0.00% Upper: 8.28%)
Dangerous_detected      =          0.00%  (Lower: 0.00% Upper: 8.28%)
Dangerous_undetected    =          53.83%  (Lower: 37.51% Upper: 69.58%)


Note: Metrics calculated above are based on the user defined fault list which might or might not be the fault universe

Observation points:
dcGain:     ((value(VT("/OUT") 2.5e-08) - value(VT("OUT") 2e-09)) / VAR("vin"))  spec: range 950m 1050m
dcGain2:    ((value(VT("/OUT2") 2.5e-08) - value(VT("OUT2") 2e-09)) / VAR("vin"))          spec: range 950m 1050m
highValue:  value(VT("/OUT") 2.5e-08)          spec: range 175m 225m
highValue2: value(VT("/OUT2") 2.5e-08)         spec: range 175m 225m
lowValue:   value(VT("/OUT") 2e-09)            spec: range -225m -175m
lowValue2:  value(VT("/OUT2") 2e-09)           spec: range -225m -175m

Detection points:
problemDetected:        ymax(VT("/problemFlag"))          spec: lt 0.5
problemDetected2:       ymax(VT("/problemFlag2"))         spec: lt 0.5


+-----------+----------+----------------------------------+--------------------------------+--------------------------------
| faultID   | corner   | F:dcGain                         | F:dcGain2                      | F:highValue
|-----------+----------+----------------------------------+--------------------------------+--------------------------------
| 1         | Nominal  | G                                | G                              | G
| 2         | Nominal  | G                                | L (0.33/|0.62| = 34.7276%)    | G
| 3         | Nominal  | G                                | L (0.0289/|0.921| = 3.04034%) | G
| 4         | Nominal  | G                                | L (0.451/|0.499| = 47.4617%)  | G
| 5         | Nominal  | G                                | L (0.131/|0.819| = 13.7552%)  | G
```

Note: The functional safety report shows the fault detection matrix for only those tests for which outputs of type `faultexpr` are specified.

For details about this report, refer to Functional Safety Report.

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

15

Working with the Setup Library Assistant

The Setup Library assistant helps you define a project-specific (master) setup for the design blocks in ADE Assembler and then reuse the specified setup to create or modify the current simulation setup in the Data View assistant. The setup in this assistant is generic and can be applied on any design. The Setup Library assistant can be accessed using ADE Assembler and ADE Verifier.

Important

The Setup Library assistant works with the ADE Verifier license. If this license is not available, an error message is displayed.

Following are the main tasks that you can perform using this assistant:

- Specify a master project-specific setup, which includes sweep variables, corners, model files, and so on.
- Save the specified setup in a cellview independent of the `maestro` view, and thus can be conveniently shared with designers working on different blocks.
- Create a verification space. You can use the verification spaces to calculate the coverage percentage in ADE Verifier.

The scope and capabilities of this assistant are defined based on your work requirements and the platforms you are working on:

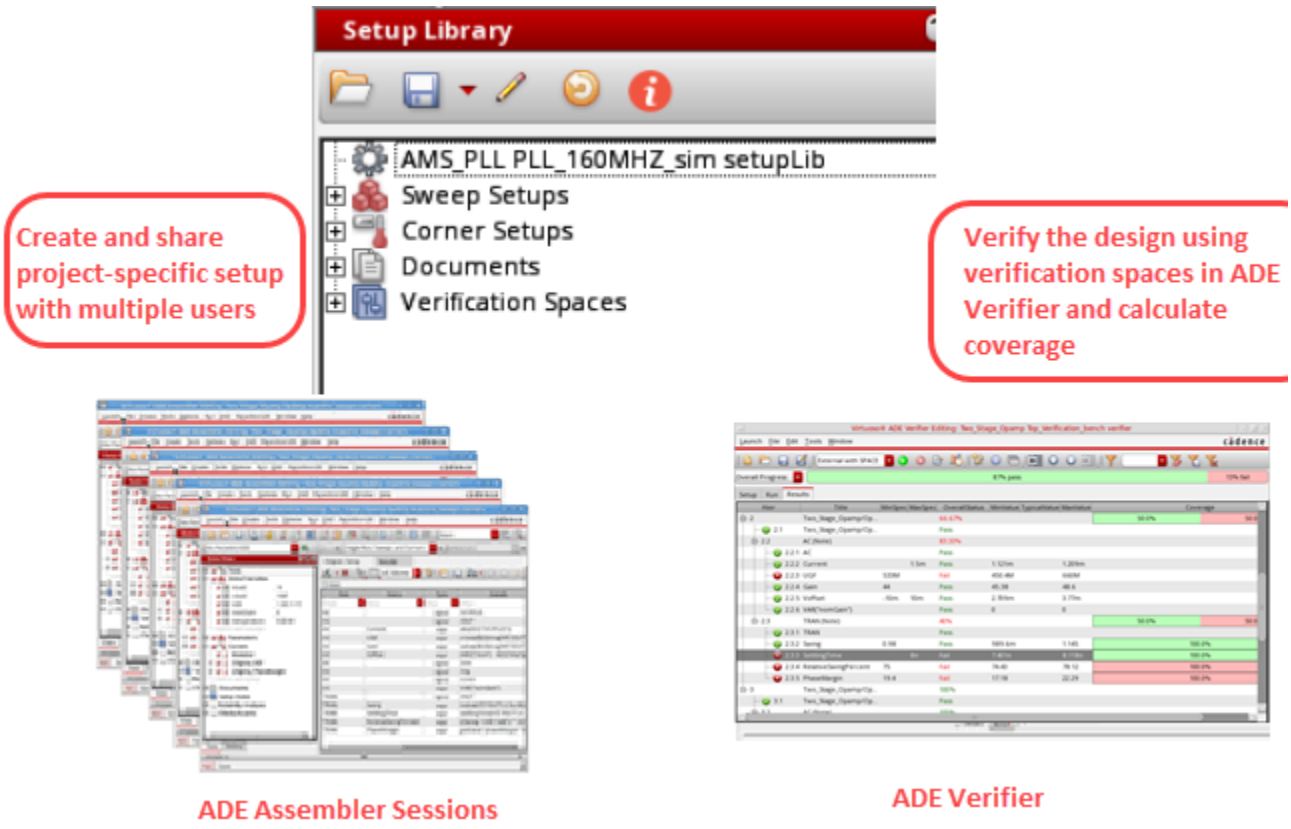
- If you work at the design level (block-level) for analog designing, then you can open this assistant in ADE Assembler to reuse the specified setup to create the setup in the Data View assistant.
- If you work on the top-level design verification (managing the overall project), then you can open this assistant in ADE Verifier to create verification spaces and ensure full coverage of the analog design blocks.

  In ADE Verifier, you can open the Setup Library assistant, load a predefined Setup Library cellview, create verification spaces, and then assign these spaces to different

© 2023 Cadence Design Systems, Inc.

blocks. You can then calculate the coverage percentage and generate a coverage report to check whether the specified setup values in verification spaces are run or not.

To know more about how this assistant works in Virtuoso ADE Verifier, see the Verifying the Design Against the Specified Setup chapter in *Virtuoso ADE Verifier User Guide*.



This chapter includes the following topics:

- Opening the Setup Library Assistant
- Setup Library Assistant Graphical User Interface
- Creating a New Cellview or Loading an Existing Cellview
- Editing the Setup
  - Adding New Sweep Setups
  - Adding New Corner Setups
  - Adding New Simulation Setups
- Reusing the Setup from Data View and Setup Library Assistants

© 2023 Cadence Design Systems, Inc.

- Synchronizing ADE Assembler Setup with Changes in Setup Library
- Creating Verification Spaces
- Managing Documents

Related Blog on Cadence Community

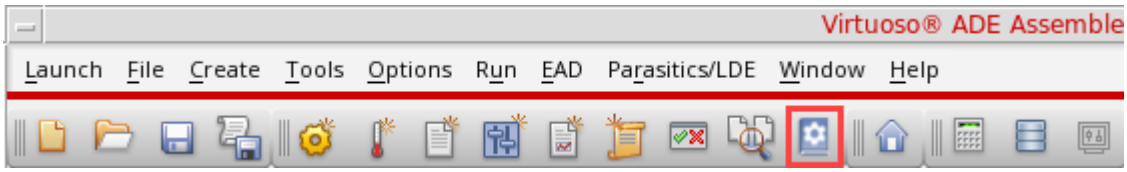
Virtuosity: Simulation Planning and Coverage Environment (SPACE) - Introduction

© 2023 Cadence Design Systems, Inc.

Opening the Setup Library Assistant

Perform one of the following steps to open the Setup Library assistant in ADE Assembler:

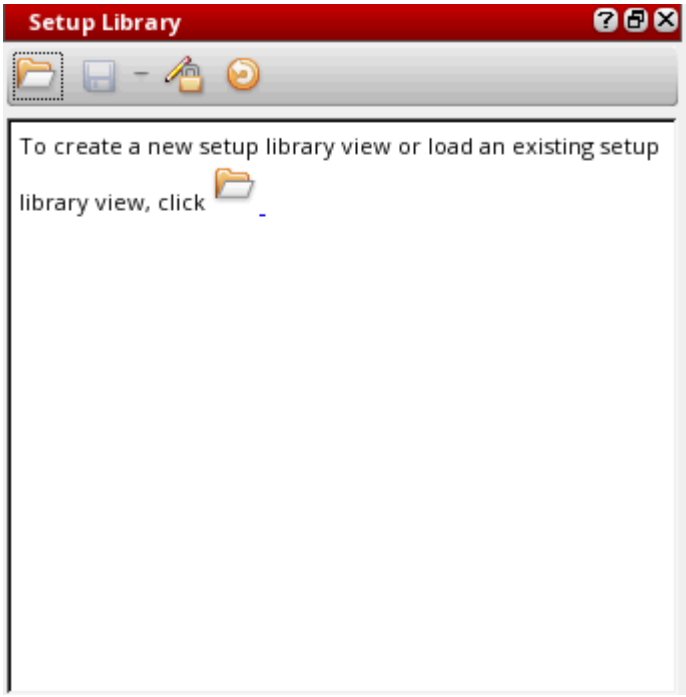
- Click *Open Setup Library* on the toolbar.



  Alternatively, choose *Windows – Assistants – Setup Library*.

- Right-click anywhere on the menu bar and choose *Assistants – Setup Library*.

  The Setup Library assistant appears on the right of the ADE Assembler window, as shown in the figure below:

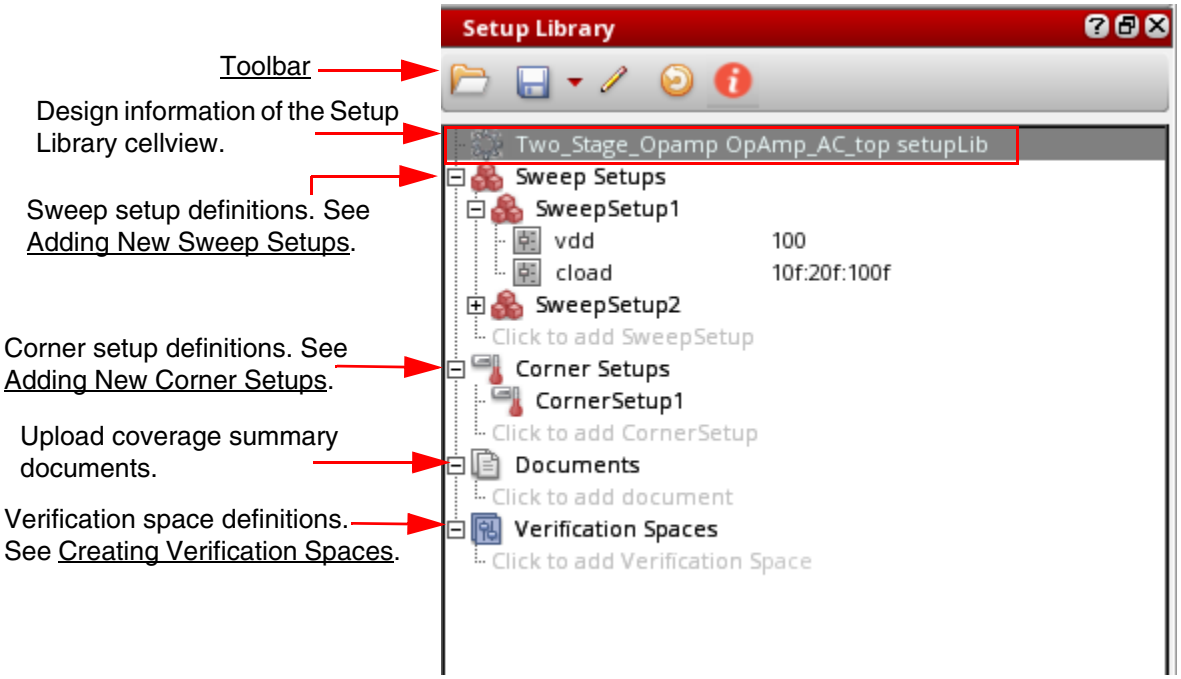


By default, when you open the Setup library assistant for the first time, no setup cellview is loaded in the assistant and the assistant is blank. You can create a new Setup Library cellview or open an existing Setup Library cellview, depending on the type of work you need to perform using this assistant.

© 2023 Cadence Design Systems, Inc.

Setup Library Assistant Graphical User Interface

The Setup Library assistant includes a toolbar and the interface for defining and displaying the setup elements. The figure below explains the graphical user interface of this assistant:



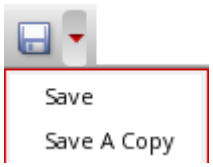
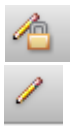
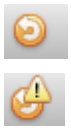
Toolbar

Table 15-1

Command	Description
	Creates a new Setup Library cellview or opens an existing Setup Library cellview in the Setup Library assistant.

© 2023 Cadence Design Systems, Inc.

Table 15-1

Command	Description
	Saves the changes made to the setup elements in the same or a different Setup Library cellview. This drop-down list includes the following two options: ■ *Save* — Saves the changes in the same cellview. ■ *Save A Copy* — Saves the changes in another cellview. When you click this option, the Save As Setup Library CellView form appears in which you can specify the library, cell, and view name to save the setup. **Note:** You need to save the changes made to the `maestro` cellview before you exit the ADE Assembler or ADE Verifier session, otherwise, the changes made to the Setup Library assistant will not be saved.
 *Toggle Edit Mode*	Controls the editing of setup loaded in Setup Library assistant. When the setup is loaded, by default a lock symbol is displayed on this button to indicate that the mode is read-only. If you have the permissions, you can edit the setup. When you click the Toggle Edit Mode button, the lock symbol disappears and the mode is changed to edit. In addition, this lock also prevents you to make changes simultaneously when someone else is editing the Setup Library cellview. If the cellview is being edited by someone, a warning message is displayed to inform you that you cannot edit the setup in the cellview. **Note:** When you switch the Setup Library assistant from edit mode to read-only mode, a message appears that suggests you to save the updates made in the Setup Library.
 *Refresh*	Refreshes the setup with the latest changes. A warning icon on this button indicates that the setup has been modified. When you click the button, the setup is updated to show the latest changes.

© 2023 Cadence Design Systems, Inc.

Table 15-1

Command	Description
	Opens the Task Assistant that provides quick help about the Setup Library assistant.

© 2023 Cadence Design Systems, Inc.

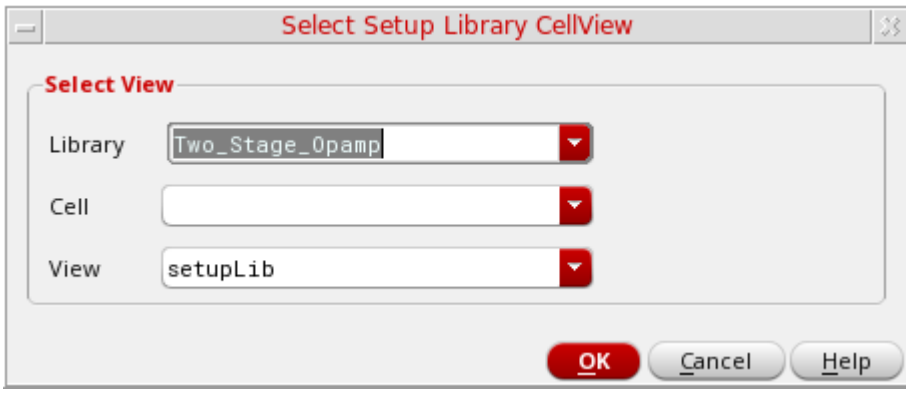
Creating a New Cellview or Loading an Existing Cellview

When you open the Setup Library assistant for the first time, it does not show any data and an information message is displayed that guides you to open a new or an existing Setup Library cellview. You can create a new Setup Library cellview and specify the project-specific setup, or you can open an existing cellview if you want to run or edit the project-specific setup.

To create a new cellview or open an existing view:

➡ Click the folder icon ( ) on the toolbar or in the text displayed in the assistant.

The Select Setup Library CellView form appears, as shown in the figure below:

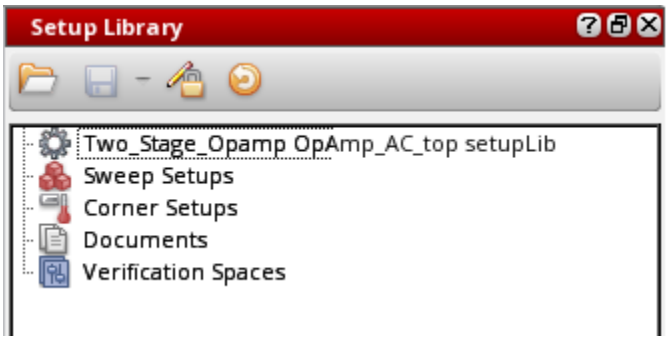


In this form, specify the following fields:

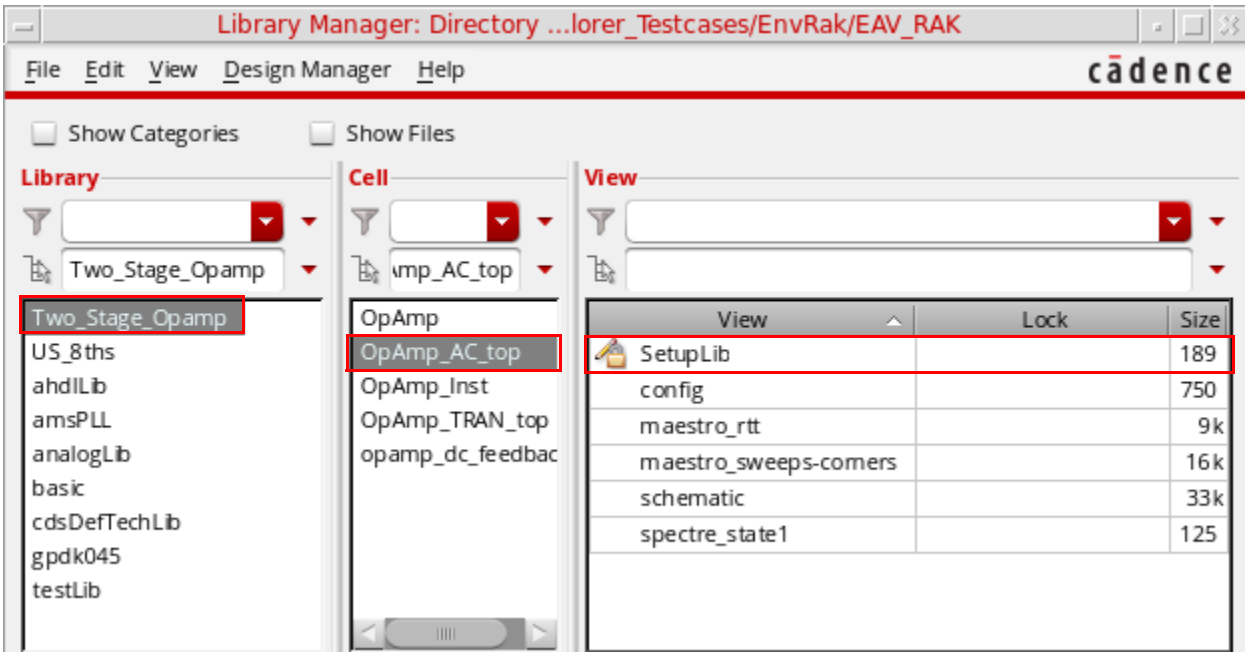
1. In the *Library* and *Cell* drop-down lists, select a library and the corresponding cell.
2. In the *View* drop-down list, all the existing cellviews of type `setupLib` are displayed. Specify a cellview name for the Setup Library view if you want to create a new cellview. By default, the cellview type is displayed as `setupLib` when you open the form. If you want to load an existing cellview, select the cellview name from the drop-down list.
3. Click *OK*.

© 2023 Cadence Design Systems, Inc.

The specified Setup Library cellview is created or loaded in the Setup Library assistant.



The new Setup Library cellview you create is saved and displayed in the *View* section of Library Manager. For example, if you create a new cellview `SetupLib` for library `Two_Stage_Opamp` and cell `OpAmp_AC_top`, this cellview is displayed in the Library Manager window, as shown in the figure below:



© 2023 Cadence Design Systems, Inc.

Editing the Setup

When a new Setup Library cellview is created, it does not include any setup elements. You need to edit the setup to add elements to that cellview. You might also want to modify the an existing setup to add, delete, or modify elements.

Caution

Before you proceed with editing, ensure that the cellview is enabled for editing. The lock symbol on the Toggle Edit Mode button ( ) on the toolbar indicates that the cellview is read-only. Click this button to enable the edit mode.

This section includes the following topics:

- Adding New Sweep Setups
- Adding New Corner Setups
- Adding New Simulation Setups
-

Adding New Sweep Setups

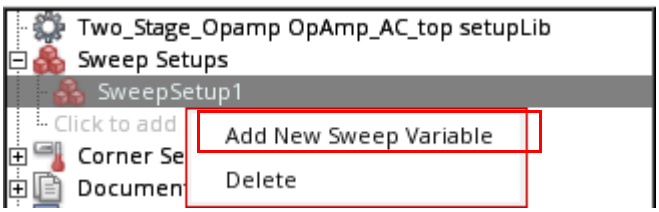
To add a new sweep setup, do one of the following:

- Right-click *Sweep Setups* and choose *Create Sweep Setup*.
- Expand the *Sweep Setups* tree and click the `Click to add SweepSetup` text.

  The sweep setup has been added and displayed in the tree. The default sweep setup name is `SweepSetup1`. You can also rename it if you want.

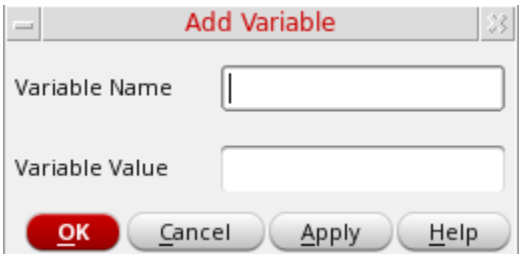
Now, to add a sweep variable in this sweep setup:

➥ Right-click the sweep setup name, such as `SweepSetup1`, and choose *Add New Sweep Variable*.

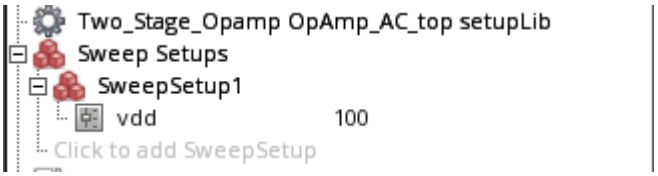


© 2023 Cadence Design Systems, Inc.

The Add Variable form appears in which you can specify the sweep variable name and value.



The specified variable is listed in `SweepSetup1` tree in a new row, as shown below:



Tip

You can double-click the displayed sweep variable value to change it directly in the variable row.

Similarly, you can add more variables to this sweep setup and create more sweep setups. The second sweep setup you create is by default named as `SweepSetup2`.

Alternatively, you can drag the variables from the Data View assistant and add them to the specific sweep setups in the Setup Library assistant. Use the `Ctrl` key to drag multiple variables together.

Note: When you drag and drop a variable from the Data View assistant to the Setup Library assistant, the status of the variable is always set to enabled irrespective of its status in the active setup.

To delete a sweep variable, right-click the variable and choose *Delete*.

To delete the entire sweep setup, right-click the sweep setup tree and choose *Delete*.

To view all the sweep variables in a sweep setup, right-click the sweep setup tree and choose *Expand All*.

To hide all the sweep variables in a sweep setup, right-click the sweep setup tree and choose *Collapse All*.

© 2023 Cadence Design Systems, Inc.

Adding New Corner Setups

To add a new corner setup, do one of the following:

- Right-click *Corner Setups* and choose *Create Corner Setup*.
- Expand the *Corner Setups* tree and click the `Click to add CornerSetup` text.

  The corner setup has been added and displayed in the tree. The default corner setup name is `CornerSetup1`. You can also rename it if you want.

To add a corner to the corner setup:

1. Add a new corner in the ADE Assembler setup or select the already specified corners in the ADE Assembler setup.
2. Drag the corner from the Data View assistant to the corner setup in Setup Library assistant.

   The corner is added in the specified corner setup in the Setup Library assistant.

Note: The only method currently available to add corners to the Setup Library assistant is to drag them from ADE Assembler. After the corners are added in the Setup Library assistant, you can edit them by double-clicking the corner name and corner conditions value. Also note that the status of a corner dragged from the Data View assistant is set as enabled irrespective of its status in the active setup. The tool remembers the status of the corner setup when you drag a corner setup from the Data View to the Setup Library or from the Setup Library to the Data View.

Adding New Simulation Setups

Currently, the Setup Library assistant supports only Monte Carlo simulation setups. Additionally, you can only add a Monte Carlo simulation setup when the currently opened setup library view is editable.

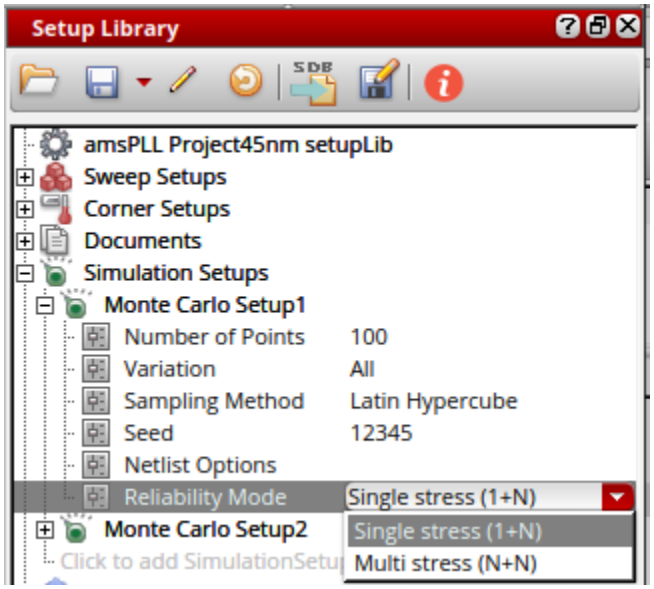
To add a new simulation setup, do one of the following:

- Right-click *Simulation Setups* and choose *Create New Monte Carlo Setup*.
- Expand the *Simulation Setups* tree and click the `Click to add SimulationSetup` text.

  The Monte Carlo simulation setup is added and displayed in the *Simulation Setups* section. The default name of the new simulation setup is `Monte Carlo Setup1`, and

© 2023 Cadence Design Systems, Inc.

it contains the variables `mcnumpoints`, `mcmethods`, `samplingmode`, and `montecarloseed`. You can double-click the setup name and rename it.



To add a variable value in this simulation setup, specify the following:

1. In the `mcnumpoints` variable row, double-click to make the text field editable and specify the number of points for which you want to run the Monte Carlo simulation.
2. From the `mcmethods` drop-down list, select method that you want to use for running Monte Carlo simulations. Possible values are *All*, *Mismatch*, and *Process*.
3. From the `samplingmode` drop-down list, select the mode that you want to use to run Monte Carlo simulations. Possible values are *Random*, *Latin Hypercube*, and *Low-Discrepancy Sequence*.
4. In the `montecarloseed` variable row, double-click to make the text field editable and specify the seed value for which you want to run the Monte Carlo simulation.

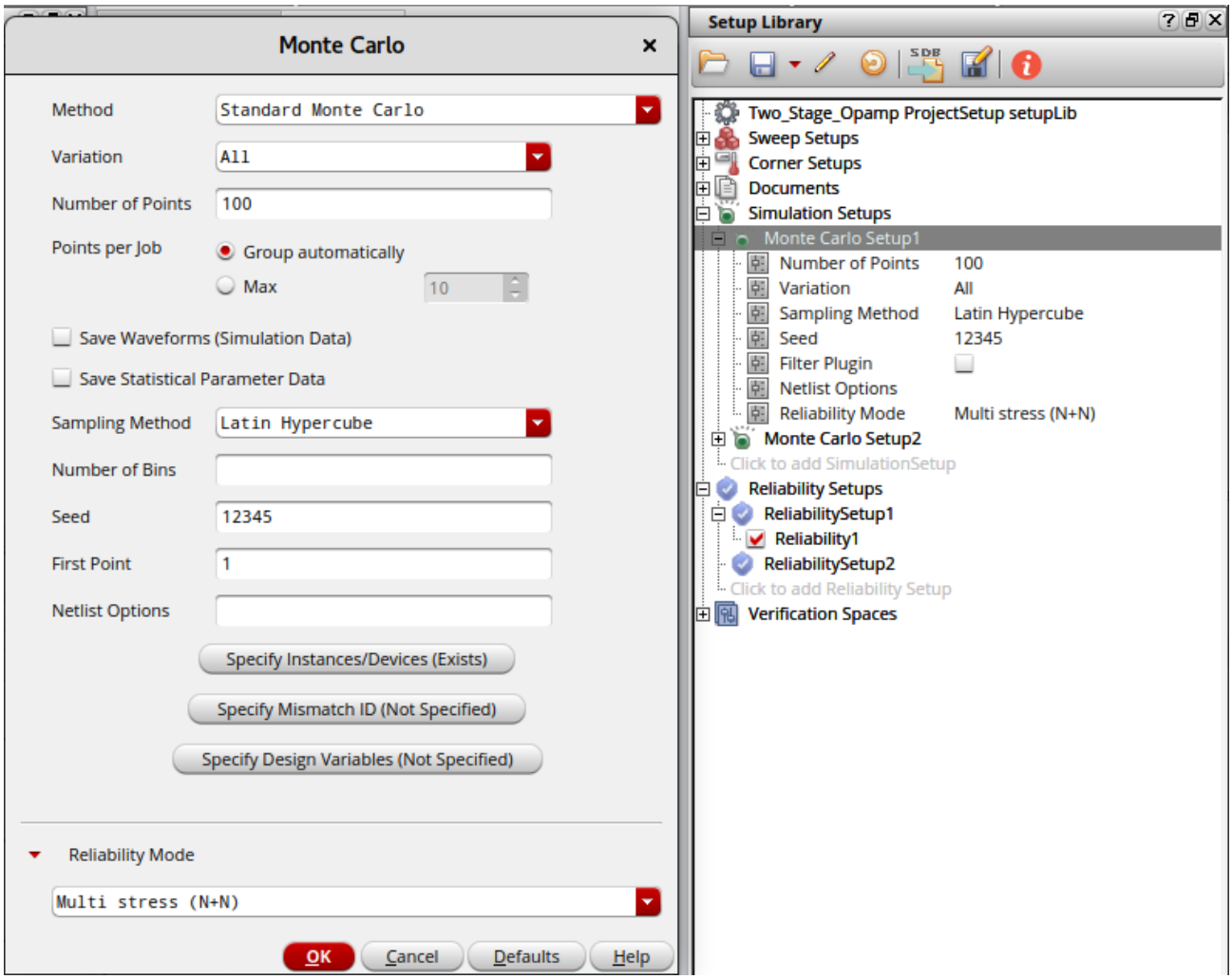
The specified variable values are listed in the `Monte Carlo Setup1` group. You can double-click the displayed variable value to change it directly in the variable row.

You can add more simulation setups in your setup library view. The second simulation setup that you create is by default named `Monte Carlo Setup2`, and so on.

© 2023 Cadence Design Systems, Inc.

Additionally, you can drag the simulation setup from the Setup Library assistant and drop it on the Data View assistant. Use the `Ctrl` key to drag multiple setups together.

When you drag and drop a Monte Carlo simulation setup from the Setup Library assistant to the Data View assistant, ADE Assembler shows the run options in the Monte Carlo options form. The form shows the values that you have specified in `Monte Carlo Setup1`.



To delete a specific simulation setup from the Setup Library assistant, right-click the setup name and choose *Delete*.

To view all the variables in a simulation setup, right-click the simulation setup name and choose *Expand All*.

© 2023 Cadence Design Systems, Inc.

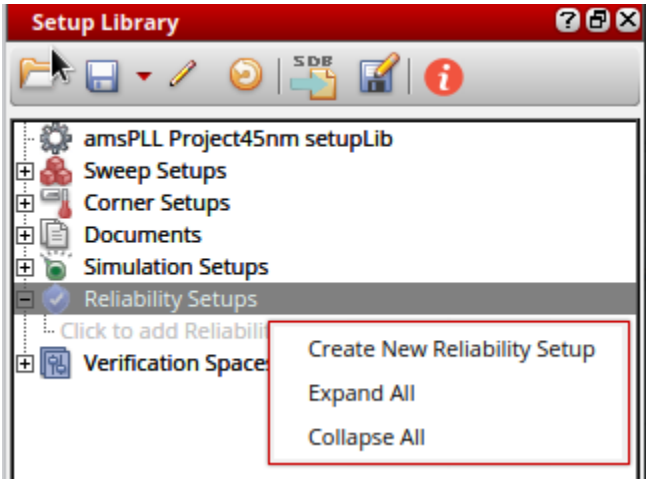
To hide all the variables in a simulation setup, right-click the simulation setup name and choose *Collapse All*.

Adding New Reliability Setups

To add a new reliability setup, ensure that the setup library cellview is editable and do one of the following:

- Right-click *Reliability Setups* and choose *Create New Reliability Setup*.
- Expand the *Reliability Setups* node and click the `Click to add Reliability Setup` text.

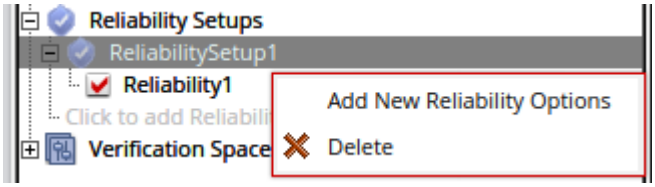
The reliability setup is added and displayed in the *Reliability Setups* section. The default name of the new reliability setup is `ReliabilitySetup1`, which you can rename as required. The second reliability setup that you create is by default named `ReliabilitySetup2`, and so on.



Ensure that the reliability setup name is a non-empty string which is not '.' or '..' and does not contain whitespaces or the characters: `'~\\*/<>'?|\"()=;&$'."`

To add reliability options to a reliability setup in the setup library cellview:

- Right-click the setup name and select *Add New Reliability Options*.



© 2023 Cadence Design Systems, Inc.

The default name of the first set of reliability options is `Reliability1`. You can add more reliability options in your reliability setup.

Alternatively, you can drag the top element of reliability analyses together with one or more reliability options from Data View in ADE Assembler and drop them to any reliability setup or reliability options node in the setup library cellview.

If the *Reliability Setups* group contains multiple reliability options, you can enable the required options and disable the others.

To enable or disable a set of reliability options, do one of the following:

- Right-click the required reliability options node and select *Enable* to enable the reliability options. Alternatively, select the check box for this set of reliability options.
- Right-click the required reliability options node and select *Disable* to disable the reliability options. Alternatively, deselect the check box for this set of reliability options.

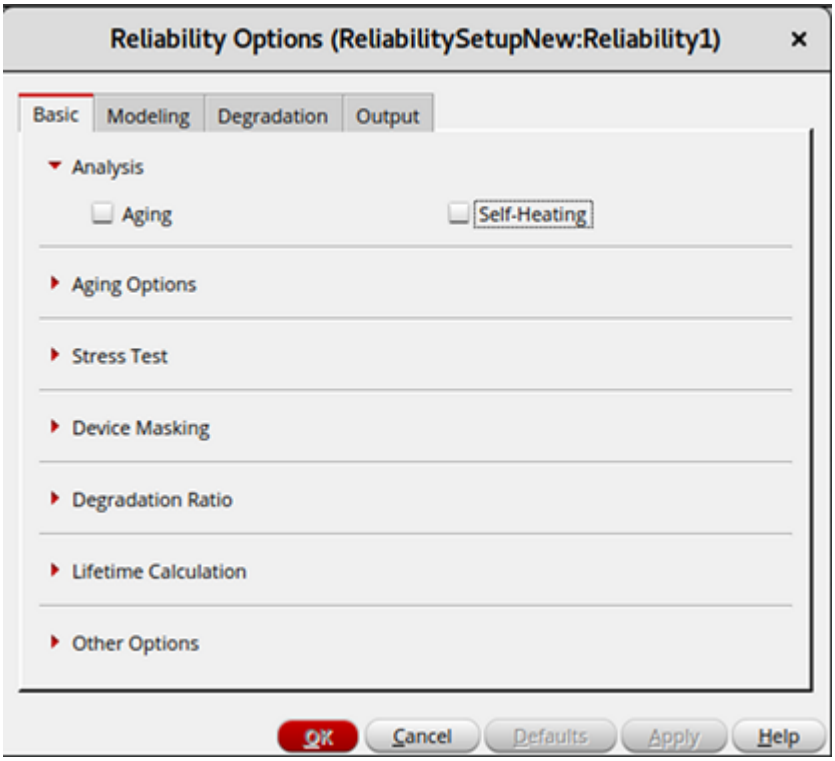
The *Enable* and *Disable* commands are available for multiple selection when the setup library cellview is editable.

To edit the reliability options in a reliability setup:

➥ Right-click the reliability options node and select *Edit Reliablity Options*.

© 2023 Cadence Design Systems, Inc.

The *Edit Reliability Options* command is only available when the setup library cellview is editable.



The Reliability Options form is displayed.

This form is the same as the Reliability Options form that is opened from the Data View in ADE Assembler. The form name is suffixed with the names of the reliability setup and reliability options in the following format.

`Reliability Setup_name:ReliabilityOptions_name`

All options in this form are set to the default values that are read from the `spectre/.cdsenv` file. You can modify the values, if required.

To view the reliability options in the read-only mode:

➡ Right-click the reliability setup name and choose *View Reliability Options*.

This command is available only when the setup library view is in read-only mode. It is not available when you select multiple reliability options.

The Reliability Options form is displayed in view-only mode. The *OK*, *Defaults*, and *Apply* buttons are disabled. This means that even if you modify any values in the form, the settings are not saved in the reliability setup.

© 2023 Cadence Design Systems, Inc.

All changes that you make to a reliability setup are indicated by a yellow exclamation mark on the *Refresh Current Setup Library with latest changes* button on the Setup Library toolbar.

You can also drag one or more reliability options from the Setup Library to the Data View. Some important points to be noted during this process are as follows:

- All reliability options related to devices are ignored
- All test names are set to the first enabled test. You can modify the test name using the Reliability Analysis Editor.

To delete a specific reliability setup from the Setup Library assistant:

➥ Right-click the setup name and choose *Delete*.

To view all the reliability options in a reliability setup:

➥ Right-click the reliability setup name and choose *Expand All*.

To hide all the reliability options in a reliability setup:

➥ Right-click the reliability setup name and choose *Collapse All*.

Reusing the Setup from Data View and Setup Library Assistants

You can load the setup defined in the Setup Library assistant into the Data View assistant and then create your ADE Assembler setup using the specified variables and corners values required for the project.

To do this, you can drag the setup elements from the Setup Library assistant to the Data View assistant. You can drag either a particular sweep or corner value or the entire sweep or corner setups. To drag multiple setup elements, use the `Ctrl` key.

If you drag one or multiple sweep variables or corners together from the Setup Library assistant to Data View, then the variables or corners are added in Data View as a copy. However, if you drag the whole sweep or corner setup, the elements in the dragged setup are displayed with a reference icon. This indicates that the elements are referenced from the Setup Library assistant.

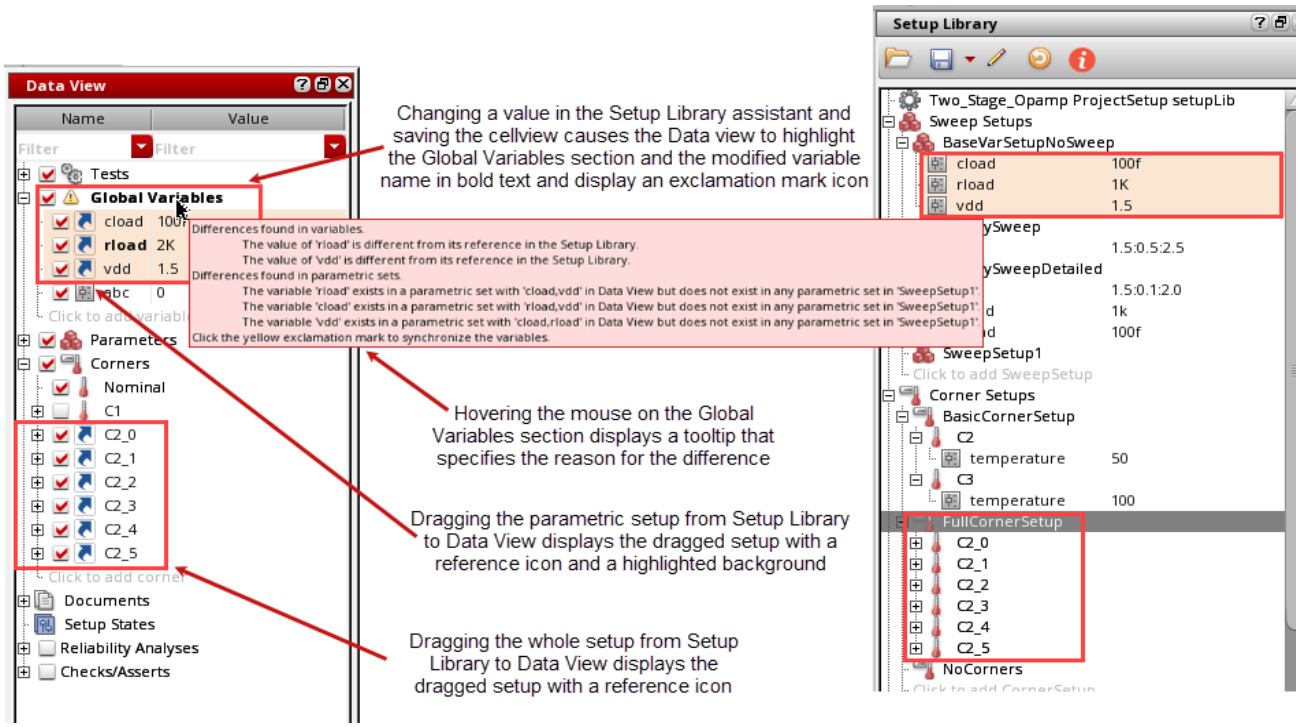
In the case of corners, when you set the overwriteEntireCorner environment variable to `nil`, the corner in the Data View assistant can have additional variables, parameters, model files, or model groups, in addition to the settings from the referenced corner setup in the Setup Library Assistant.

© 2023 Cadence Design Systems, Inc.

If the autoUpdateTagForReference environment variable is set to `t`, the setup dragged from the Setup Library assistant is added as a tag for the variable or corner in the Data View assistant.

Modifications of these referenced elements in either of these assistants are highlighted in the Data View assistant. Consider that the value of a variable, `vdd`, in the Setup Library assistant, is set to `1.5`, and this variable is also present in the Data View assistant as a reference. Now, if you change the value of `vdd` to `1.2` in the Setup Library assistant, or if you change the value of vdd to `2` in the Data View assistant, the Data View assistant displays the variable name in bold text.

Additionally, an exclamation mark icon is also displayed with the Global Variables section, indicating that the variables are not in sync with the Setup Library. When you move your mouse over the section name, a tooltip is displayed to specify the reason for the difference. When you hover the pointer on sweeps and corners in Data View, a tooltip appears to display the name of the referenced setup from Setup Library.



If you want to run simulations for a specific set of sweep combinations for different parameters. you can create a parametric setup in the Setup Library assistant and drag it to the Global Variables section in the Data View assistant. To do this, select one of the following:

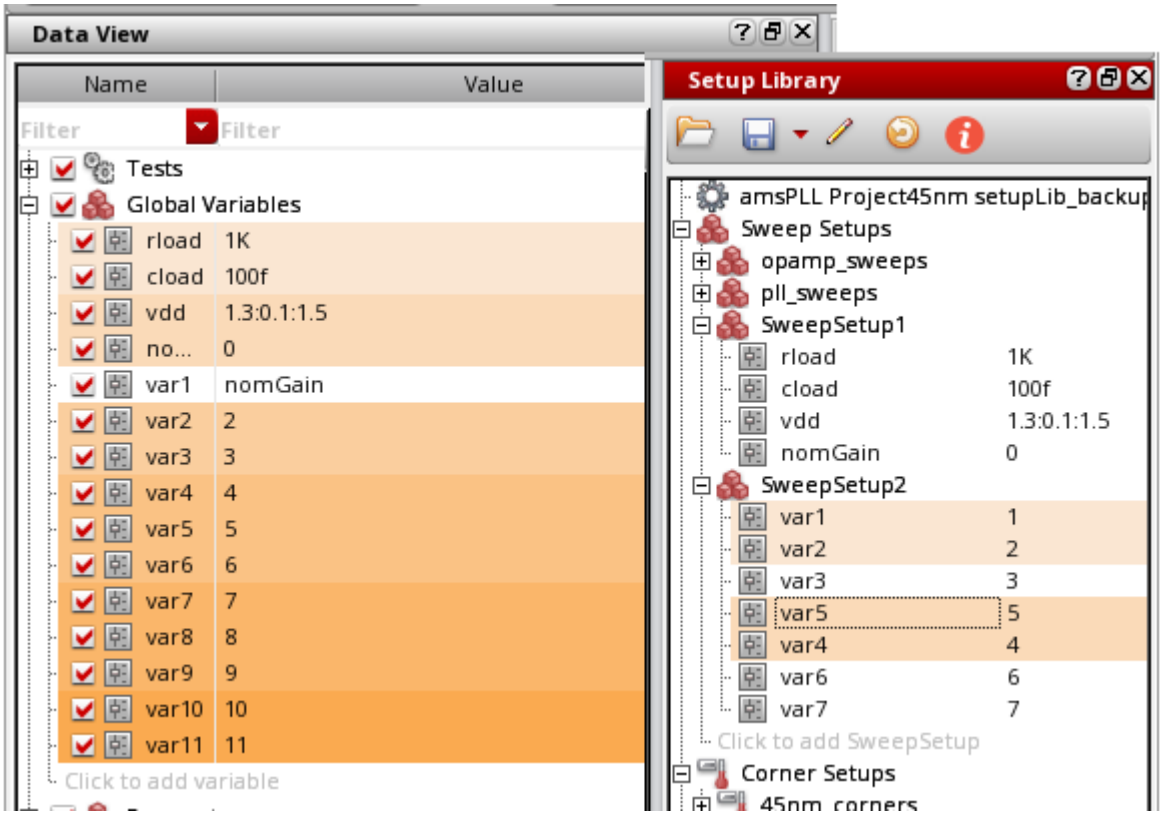
- Select only the variables in the parametric set and drag them to the Data View assistant. This is useful when you only want to add the parametric set, but the sweep setup has other variables, in addition to the parametric set.

© 2023 Cadence Design Systems, Inc.

- Select the sweep setup and drag it to the Global Variables section in the Data View assistant. This is useful when all variables in the sweep setup are grouped as a parametric set, or if you want to add the parametric set as well as all other variables from the sweep setup.

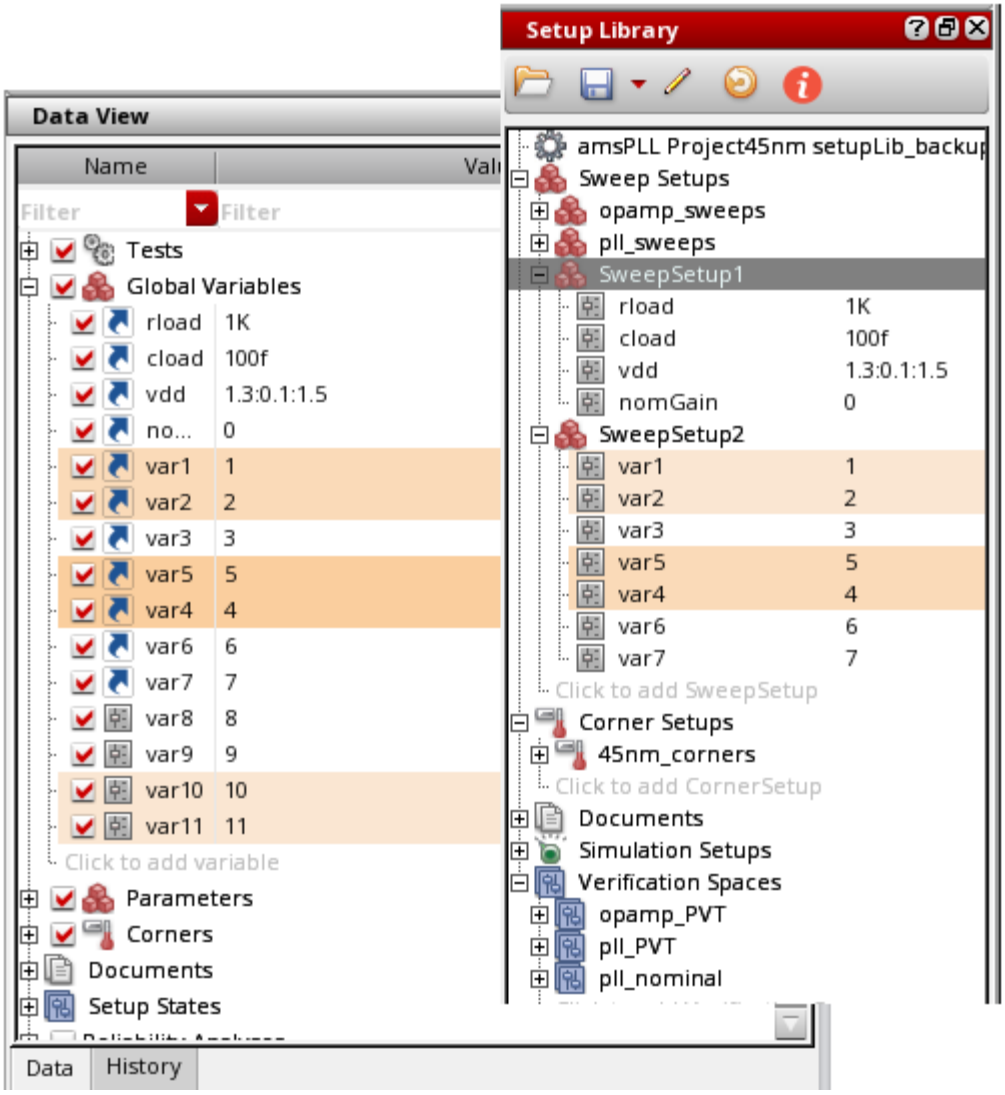
  All variables in the dragged setup are displayed with a reference icon and the variables in the parametric setup are displayed with a highlighted background. If a difference exists in the parametric sets between Global Variables in the Data View and referenced sweep setups in the Setup Library assistant, the Global Variables section and the variable name with the difference is displayed in bold text with an exclamation icon. Additionally, a tooltip displays the differences.

Consider a scenario where the Data View and the Setup Library both contain multiple parametric setups, as shown below. Here, each parametric set is highlighted in a different shade of color.



Now, if you drag `SweepSetup1` and `SweepSetup2` from the Setup Library to the Data View, the setup from the Setup Library takes priority and overwrites the variables and parametric sets in the Global Variables section. The Data View assistant retains a parametric set in the Global Variables section if it contains variables that are not present in the Setup Library. It can be observed that all parametric sets are redefined as follows:

© 2023 Cadence Design Systems, Inc.



You can change the dragged values in Data View assistant; however, it is considered a deviation from the project-specific setup and is not recommended.

You can also drag the setup elements and parametric sets from the Data View assistant to load them into the Setup Library assistant. Here, the parametric setup is displayed with a highlighted background. You can drag one variable or corner at a time or the entire setup in one step. To load multiple setup elements, such as multiple sweeps or corners together, select them using the `Ctrl` key and then drag them.

Note: You can drag and drop setups from Data View assistant to Setup Library only when the Setup Library cellview is opened for editing. Otherwise, a warning message is displayed.

© 2023 Cadence Design Systems, Inc.

Similarly, you can also drag one or more reliability options from the Setup Library to the Data View. Some important points to be noted during this process are as follows:

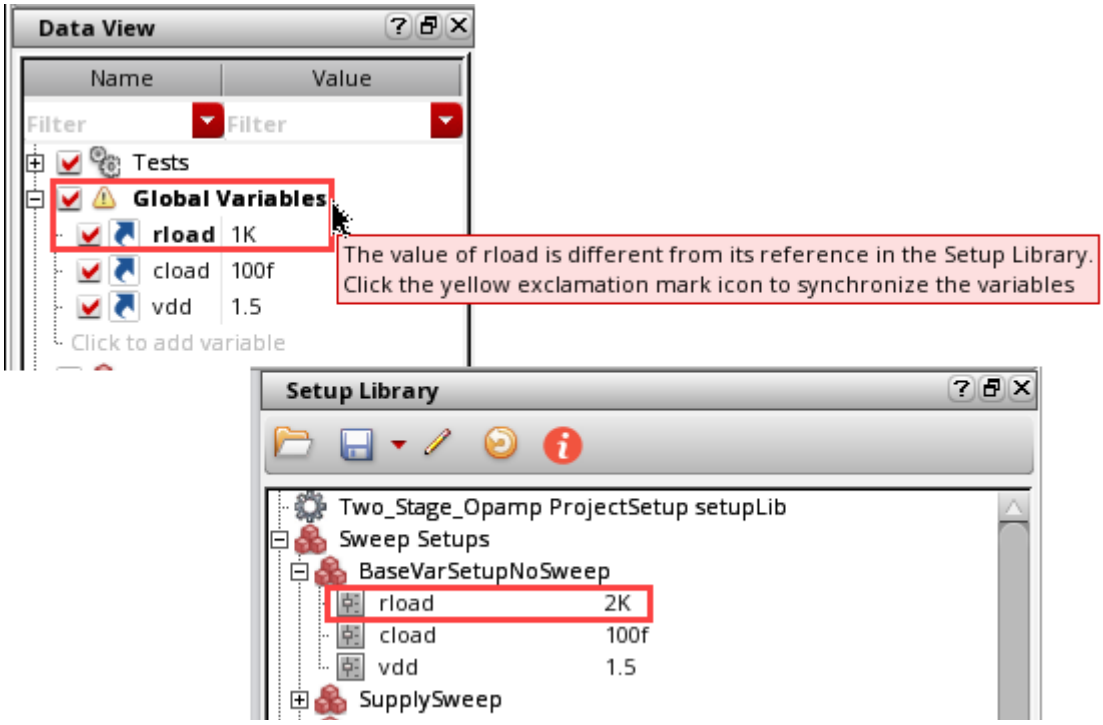
- All reliability options related to devices are ignored
- All test names are set to the first enabled test. You can modify the test name using the Reliability Analysis Editor.
- The reliability options in the Data View assistant are overwritten if the name of the reliability options in Data View is the same as the options in the setup library cellview. However, no changes are made to the original test name and the reliability options related to device.
- A reference icon is displayed on the Data View elements in the reliability setup when you drag the reliability setup from the Setup Library to the Data View. This indicates that the elements are referenced from the Setup Library assistant.
- The name of the Reliability Analyses group and all out-of-sync items in the reliability setup of the Data View assistant are displayed in bold.

Synchronizing ADE Assembler Setup with Changes in Setup Library

If the sweep or corner setup and reliability setup that you reused from the Setup Library assistant is updated in the project-specific setup in Setup Library, the Data View assistant displays the Global Variables section name and variable name, and the Reliability Analyses and reliability elements in bold text.

© 2023 Cadence Design Systems, Inc.

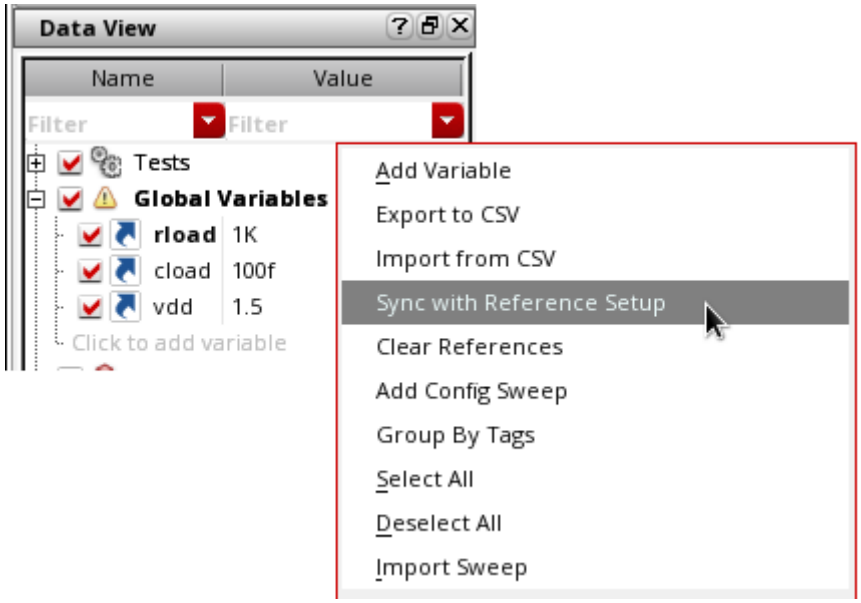
The tooltip also shows a message indicating that the values in the referenced setup have changed, and shows the revised values, as shown below.



You can synchronize the ADE Assembler setup to reflect the latest updates. To do this:

© 2023 Cadence Design Systems, Inc.

➡ Right-click the *Global Variables*, *Corners*, or *Reliability Analyses* section and choose *Sync with Reference Setup*.



The exclamation mark icon on the *Global Variables* and *Reliability Analyses* sections and the bold text disappear and the Data View assistant reflects the updated values from the referenced setup.

Clearing References to Setups in the Setup Library

There can be scenarios when you want to remove any references to the sweep, corner, or reliability setup to meet the following requirements:

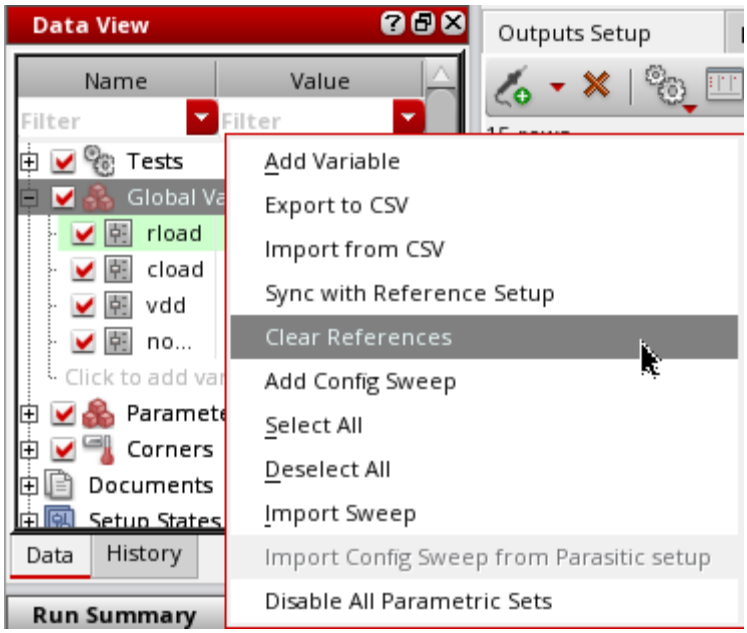
- You want to modify the values in the Data View assistant independent of the Setup Library
- Any changes in setup library do not overwrite the values in Data View assistant

In such cases, you can clear all references to the sweep setups or corners setups.

© 2023 Cadence Design Systems, Inc.

To remove the references:

➡ In the Data View assistant, right-click *Global Variables* or *Corners*, and choose *Clear References*.



ADE Assembler clears the references for all the setups. After this, any changes in the setup library are not reflected in the maestro view.

Once the project-specific setup is defined and reused to create the current simulation setup in Data View, and simulation results are obtained, you can verify the simulation history in ADE Verifier. You can create verification spaces and calculate the design coverage to check whether the entire project-specific setup has been run, and then generate the coverage report in the *Result* tab of ADE Verifier. For more details, refer to Creating Verification Spaces.

Creating Verification Spaces

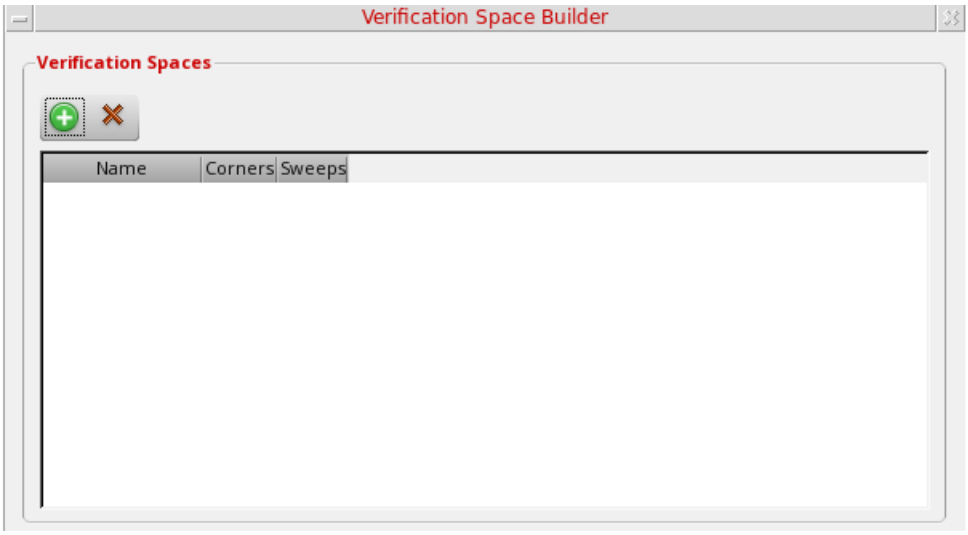
A verification space is a combination of sweep and corner values that you specify in the Setup Library assistant. You can assign these verification spaces to different design blocks and use them to calculate the coverage.

To create verification spaces, do one of the following:

- Right-click the *Verification Spaces* tree and choose *Create Verification Space*.
- Click the `Click to add Verification Space` text displayed under the *Verification Spaces* tree.

© 2023 Cadence Design Systems, Inc.

The Verification Space Builder form appears, as shown in the figure below:



To add a new verification space:

1. Click *Create New Space* ( ).

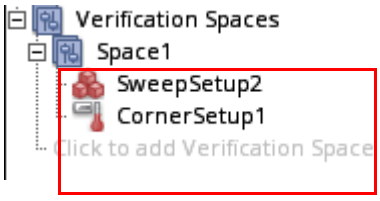
   A new space row is added and named as `Space1`, as shown in the figure below:

Name	Corners	Simulation Setups	Sweeps	Reliability Setup
Space1	CornerSetup1	Monte Carlo Setup1	SweepSetup1	ReliabilitySetup1

   You can double-click the space name and rename it if you want.

2. Select the following, as required:

   - a corner setup from the *Corners* list
   - a sweep setup from the *Sweeps* list
   - a Monte Carlo simulation setup from the *Simulation Setups* list
   - a reliability setup from the *Reliability Setup* list



© 2023 Cadence Design Systems, Inc.

Similarly, you can create more verification spaces and they are by default named as `Space2`, `Space3`, and so on.

To delete a verification space, select the space in the Verification Space Builder form and click Delete ( ).

For more information about how to use the Setup Library assistant to calculate coverage, see the Verifying the Design Against the Specified Setup chapter in *Virtuoso ADE Verifier User Guide*.

Managing Documents

The Setup Library assistant includes a *Documents* section in which you can import a file containing the coverage report for a particular design. For more information about how to export the coverage report into a CSV file, see *Exporting the Coverage Report* in *Virtuoso ADE Verifier User Guide*.

Once the coverage report is exported, you can perform the following step to add it into the *Documents* section of the Setup Library assistant:

➥ Expand the *Documents* tree and click the `Click to add document` text.

The Choose Document to be added form appears in which you can browse and select a document you want to add. The selected file will be added in the *Documents* section.

You can right-click the filename and choose one of the following options to perform various operations:

- *Open*—Select this option to open the CSV file.
- *Delete*—Select this option to delete the CSV file.
- *Save As*—Select this option to save a copy of the CSV file.
- *Refresh*—Select this option to refresh the CSV file to show the latest data. This option is available even if no document is added in the Documents section.

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

16

Performing Reliability Analysis

Reliability analysis can be performed using two modes: Spectre native and RelXpert. The reliability analysis analyzes the effect of time on circuit performance drift and predicts the reliability of designs in terms of performance. In ADE Assembler, you can run the reliability simulation for fresh test (when time is zero), stress test (to generate degradation data), and aged test (at specific intervals, such as 1 yr, 3 yrs, or 10 yrs). In the stress test, extreme environmental conditions are used to stress devices before aging analysis.

Note: The RelXpert Reliability Simulator interface in ADE Assembler is supported with the Spectre Simulator in MMSIM 7.2 or a later version. For more information, see *Virtuoso RelXpert Reliability Simulator User Guide*.

This chapter describes how to set up and run reliability simulation in ADE Assembler.

See the following topics for more information:

- Simulator Modes for Reliability Analysis
- Specifying the Reliability Analysis Setup
- Reliability Options Form
- Creating Outputs for Reliability Analysis
- Running the Reliability Simulation
- Working with Reliability Data
- Viewing Reliability Report
- Annotating Simulation Results to Schematic View

Important Points to Note

- Currently, reliability analysis is run with Sweeps and Corners and Monte Carlo run modes.
- Reliability analysis with Monte Carlo is supported for Spectre and AMS simulators.

© 2023 Cadence Design Systems, Inc.

Simulator Modes for Reliability Analysis

You can run reliability analysis in two simulator modes:

- Spectre native: The Spectre native mode runs simulations using the Spectre simulator. The is the default simulator mode and gives higher performance for large circuits. Starting from MMSIM10.1.1 ISR6, Spectre native reliability analysis supports the BSIM3V3, BSIM4, BSIMSOI and PSP102 models.

  Note: To know more about the Spectre native simulator mode and the models supported by it, refer to the Reliability Simulation Block section in *Virtuoso Spectre Circuit Simulator and Accelerated Parallel Simulator User Guide*.

- RelXpert: The RelXpert mode runs simulations using the RelXpert simulator. This mode accepts agemos parameters for BSIM3v3 and BSIM4 and supports the agemos method for aged `.model` card generation. To enable the RelXpert simulator mode, set the following environment variable in CIW:

  ```
  envSetVal("spectre.relxOpts" "nativeMode" 'string "RelXpert")
  ```

  See also: Specifying Simulator Mode for Reliability Analysis

Specifying the Reliability Analysis Setup

Before you specify the reliability analysis setup, ensure that you do the following:

- In the *Data View* assistant pane, select the tests on which you want to run the reliability simulation (fresh, stress, and aged). This is required because only the tests you select in the Data View assistant pane are visible in the setup form.

  Note: If a local variable is overridden by a global variable, reliability analysis will use the values set for the global variable. Local values will be ignored.

- Ensure that one or more of the following analyses are enabled for the tests in your setup:

  `tran`, `ac`, `sp`, `dc`, `envlp` `hb`, `hbac`, `hbsp`, `hbnoise`, `pss`, `pnoise`, `stb`, `noise`, `pac`, and `pxf`

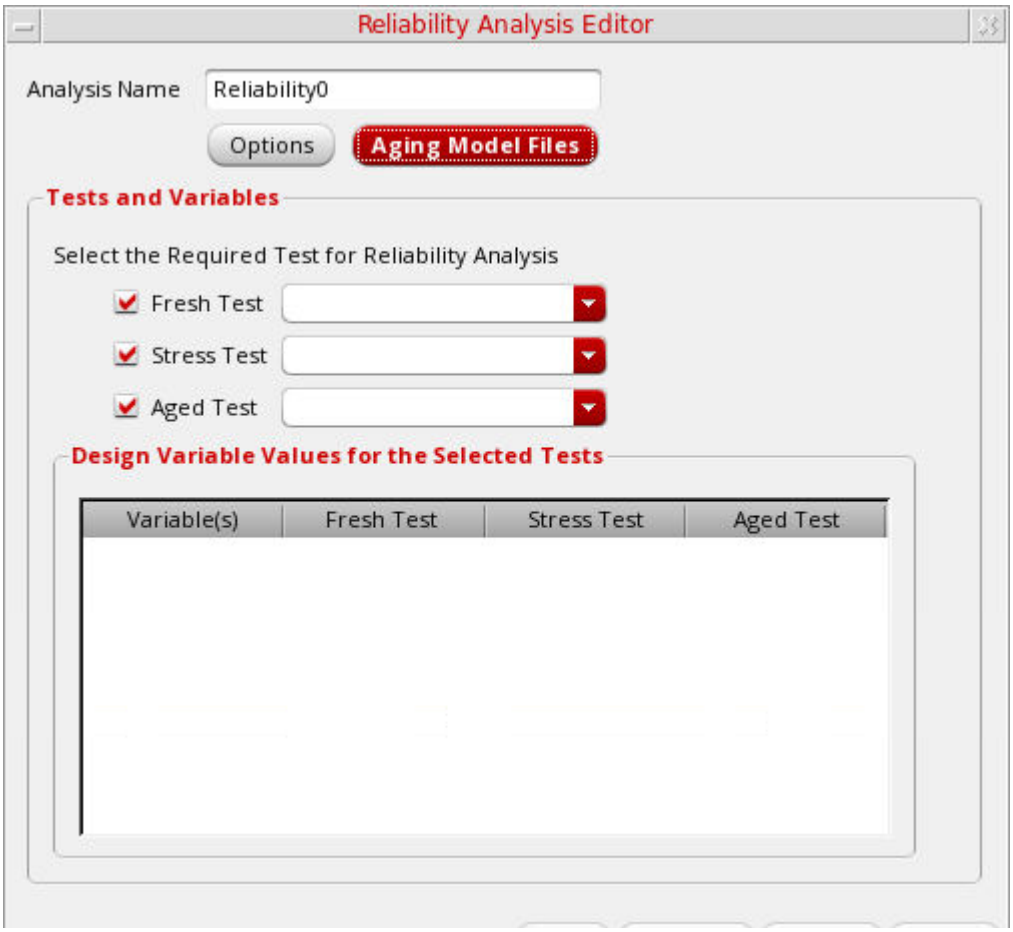
- On the *Run* toolbar, in the *Select a Run Mode* drop-down list, select either *Single Run, Sweeps and Corners* or *Monte Carlo Sampling*. Reliability analysis is supported only for these run modes.

Follow these steps to set up reliability analysis:

1. In the *Data View* assistant pane, expand the *Reliability Analyses* node and click where it says *Click to add reliability analysis setup*.

© 2023 Cadence Design Systems, Inc.

The *Reliability Analysis Editor* form appears.



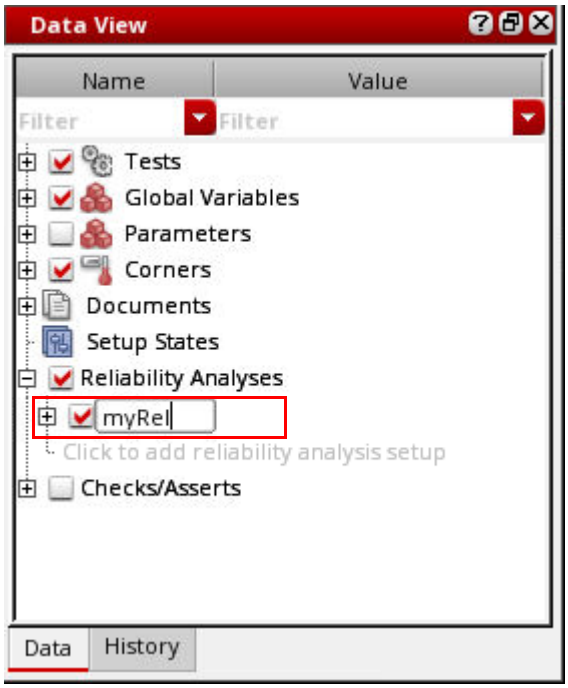
Note: The *Aging Model Files* command is displayed in the *Reliability Analysis Editor* form only if the defaultAgingModelFiles environment variable is set to true. You can click *Aging Model Files* to open the *Aging/Self-Heating Model Setup* form. In this form, you can specify the model files to be used for the stress and aged tests. Paths to the model files are added to the netlist used for simulation.

2. The *Analysis Name* field is a read-only field. The default analysis name is `Reliability0`.

   Note: You can edit the name of a reliability setup only after saving it. To rename a

© 2023 Cadence Design Systems, Inc.

reliability setup, double-click its name in the *Data View* assistant



3. In the *Tests and Variables* section, specify the following:

   a. From the *Fresh Test* drop-down list, select the name of the test you want to use for running fresh simulation.

   b. From the *Stress Test* drop-down list, select the name of the test you want to use for running stress simulation.

      Note: The stress simulation generates a stress file. You can reuse the stress file if you do not want to run the stress test again. See the Reusing the Stress File section for more information.

   c. From the *Aged Test* drop-down list, select the name of the test you want to use for running an aging simulation.

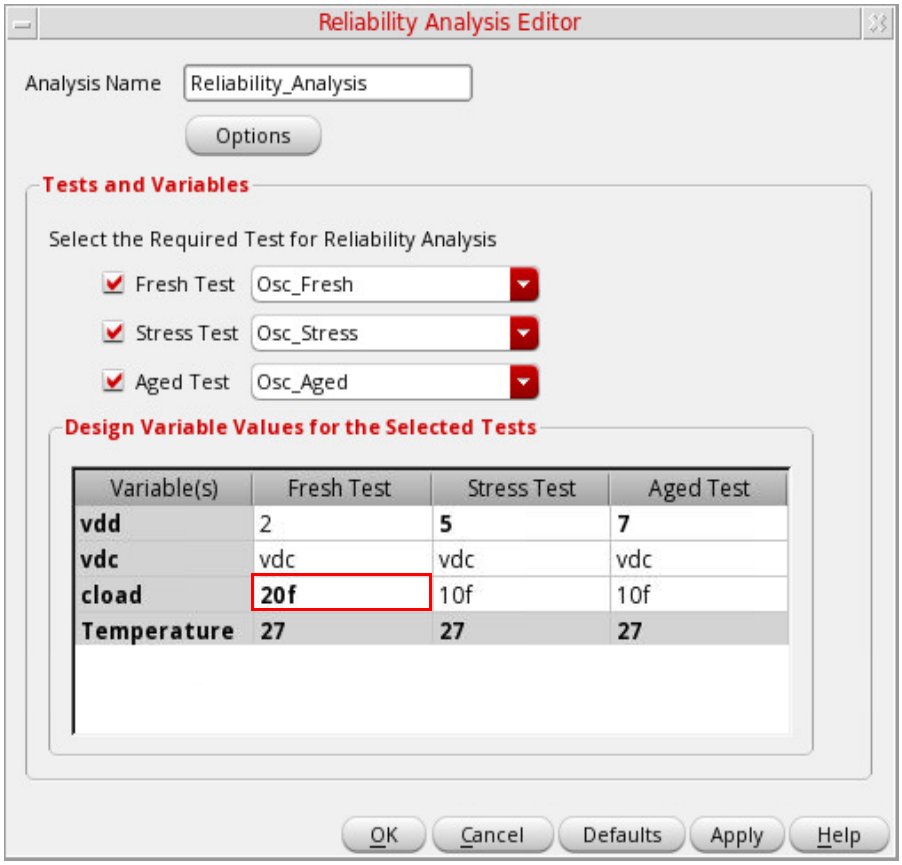
Note the following important points with respect to the selection of tests for the fresh, stress, and aged test:

- It is possible to specify different test names for fresh, stress, and aged tests.
- Simulation for an aged test uses the simulation data generated by the stress test. Therefore, a stress simulation must be run before an aging simulation is run.

The table displayed at the bottom of the form shows the values of global and local variables to be used for fresh, stress, and aged tests.

© 2023 Cadence Design Systems, Inc.

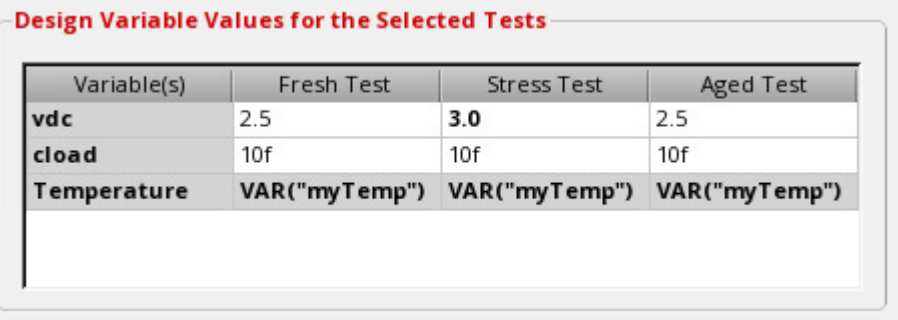
4. If required, you can modify the values of variables in this table. A variable can be set to a static value, as shown in the figure below, or to the name of a sweep variable whose values are substituted at the run time.



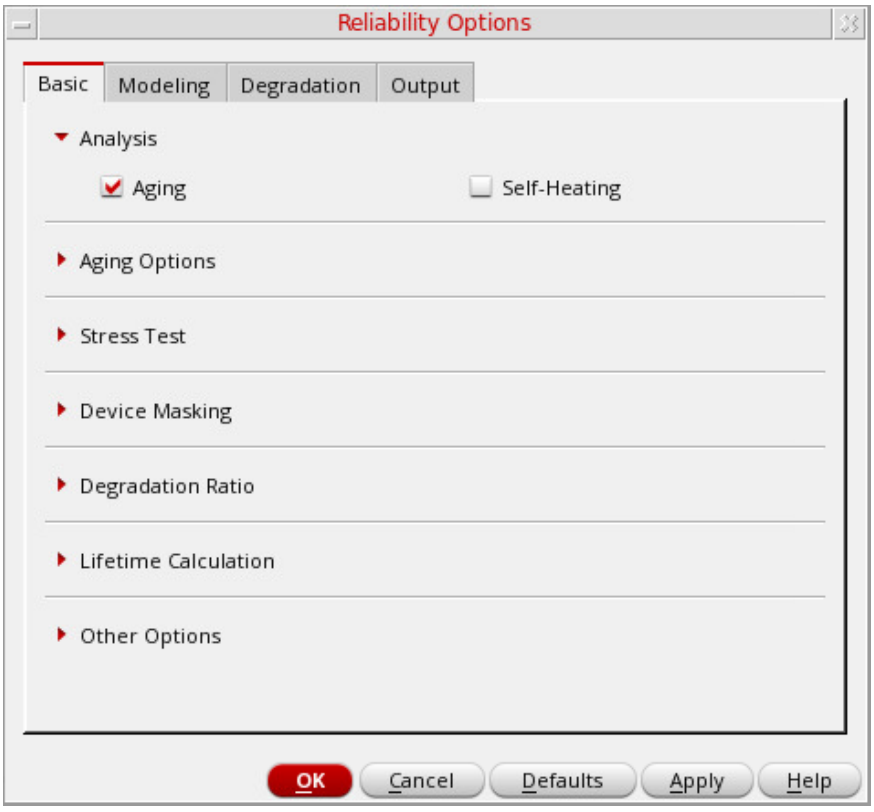
Note: The values of the design variables that are displayed in bold indicates that they have been used in the corner setup or in the testbench. Note that the values set in the reliability setup are used in simulation.

© 2023 Cadence Design Systems, Inc.

You can also use the `VAR()` expression to specify design variables. In the following example, value of the variable `myTemp` is assigned to the design variable `Temperature`.



5. Click *Defaults* to reset design variables to their original values.
6. Click *Options*.
   The *Reliability Options* form appears.



7. In the Reliability form, specify options for the simulation.

   These options for reliability analysis are saved in maestro.sdb setup database.

   For details of the Reliability form, refer to Reliability Options Form.

© 2023 Cadence Design Systems, Inc.

8. After specifying the reliability options, close the Reliability form.
9. Click *OK* to close the form.

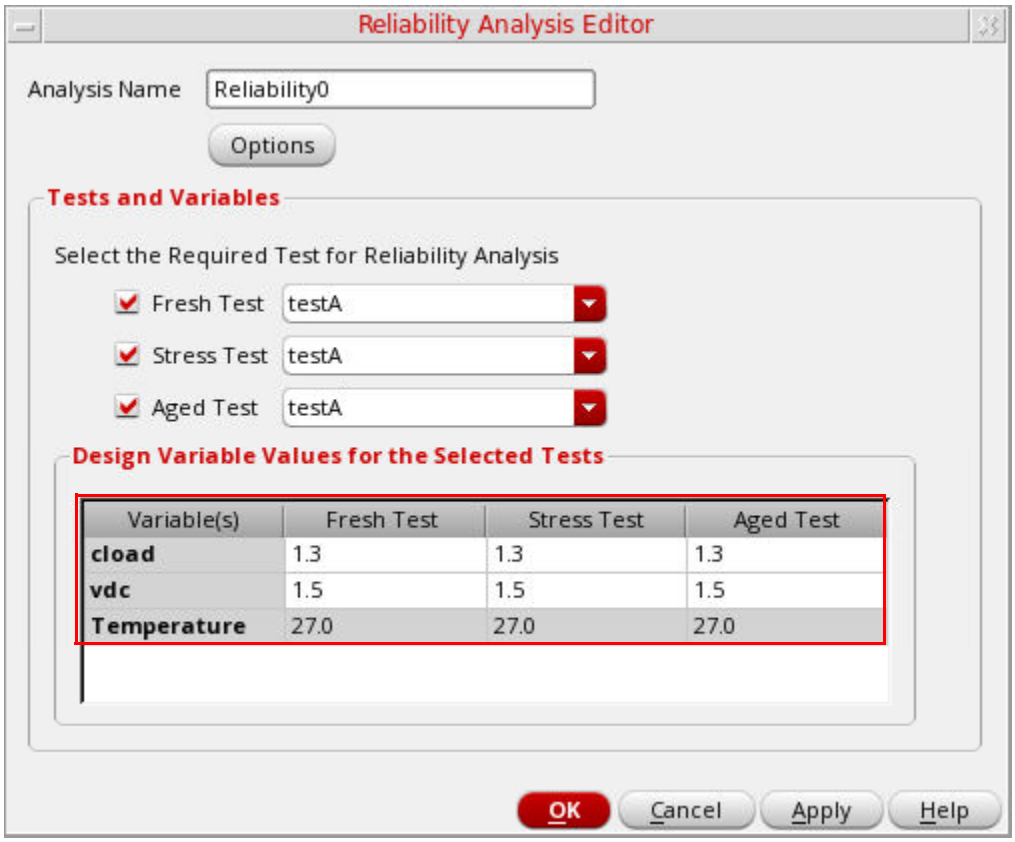
   The reliability setup appears under the *Reliability Analysis* tree on the Data View assistant pane.

   If required, you can add notes to the reliability analysis setup you have created. For this, right-click the reliability analysis setup name on the Data View assistant pane and click *Notes* to open the Add/Edit Notes form. Add notes in the *Notes* field and click *OK*. These notes are displayed in the tooltip for the setup and saved in the setup database.

   For related information, see Adding Notes to a Test.

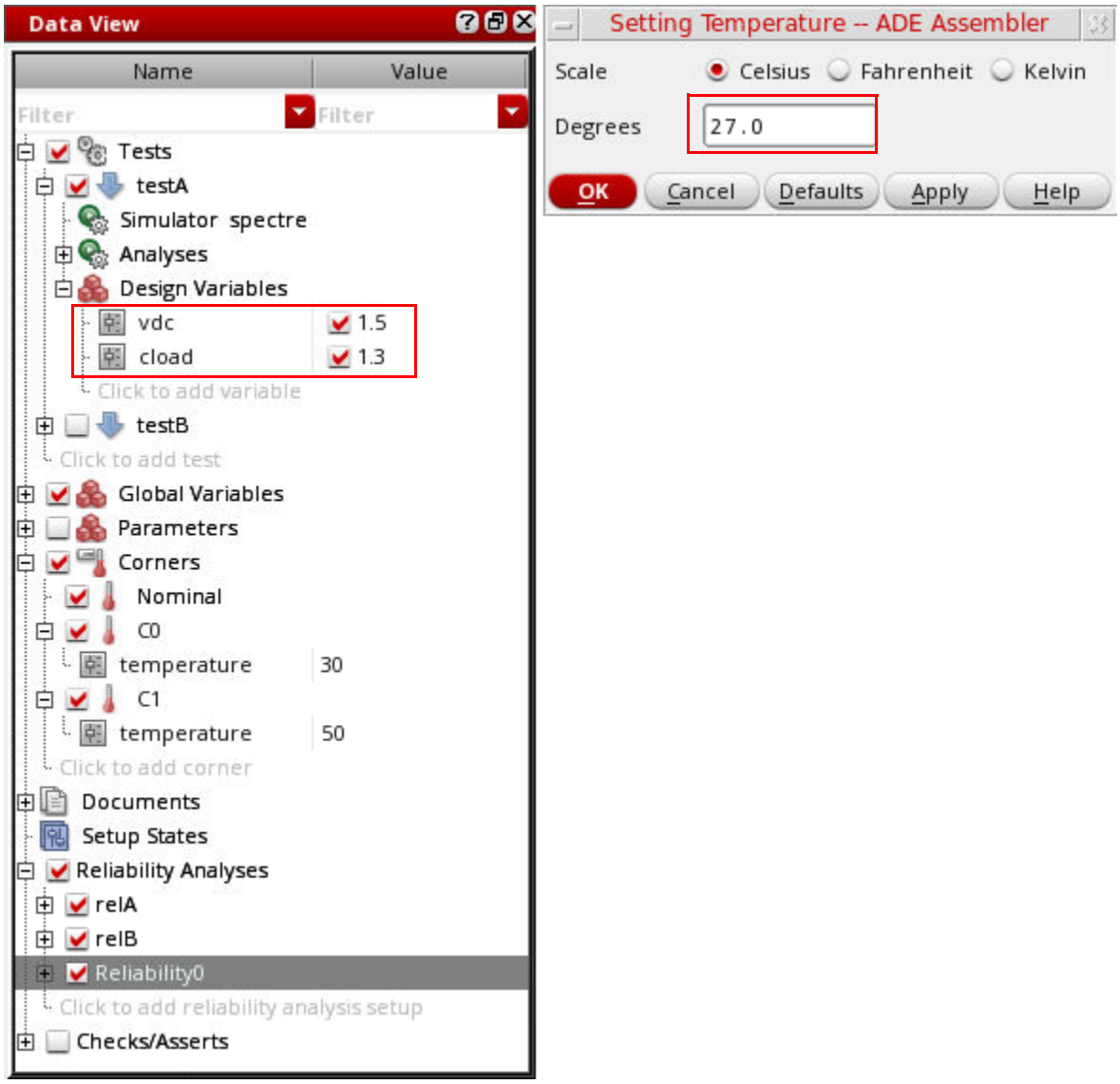
Variables and Temperature in the Reliability Setup

When you select fresh, stress, and aged tests in the Reliability Analysis Editor, the *Design Variable Values for the Selected Tests* section displays the default values of design variables and the simulation temperature for the selected tests.



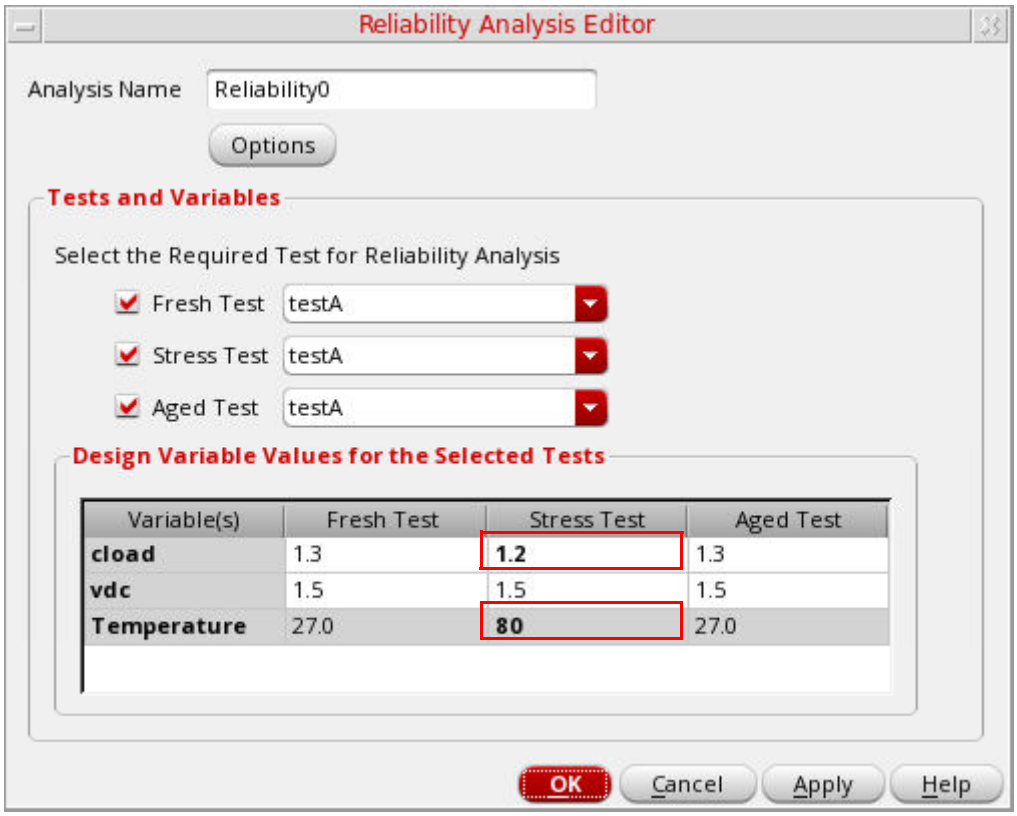
© 2023 Cadence Design Systems, Inc.

These values are the same as defined for these tests in the Data View assistant and in the Setting Temperature form.



© 2023 Cadence Design Systems, Inc.

If you edit the values of the design variables and temperature in the Reliability Analysis Editor, these edited values are displayed in bold. When you hover over an edited value, a tooltip is displayed, indicating that the edited value is overriding the value defined in the test.



The following table shows how the values of `cload` and `temperature` are used in the fresh, stress, and aged simulations.

Corner	Stress Simulation	Fresh and Aged Simulation
Nominal	`temperature = 80, cload = 1.2`	`temperature = 27, cload = 1.3`
C0	`temperature = 80, cload = 1.2`	`temperature = 30, cload = 1.3`
C1	`temperature = 80, cload = 1.2`	`temperature = 50, cload = 1.3`

Note that the values defined and edited on Reliability Analysis Editor take priority over the values defined in the active setup.

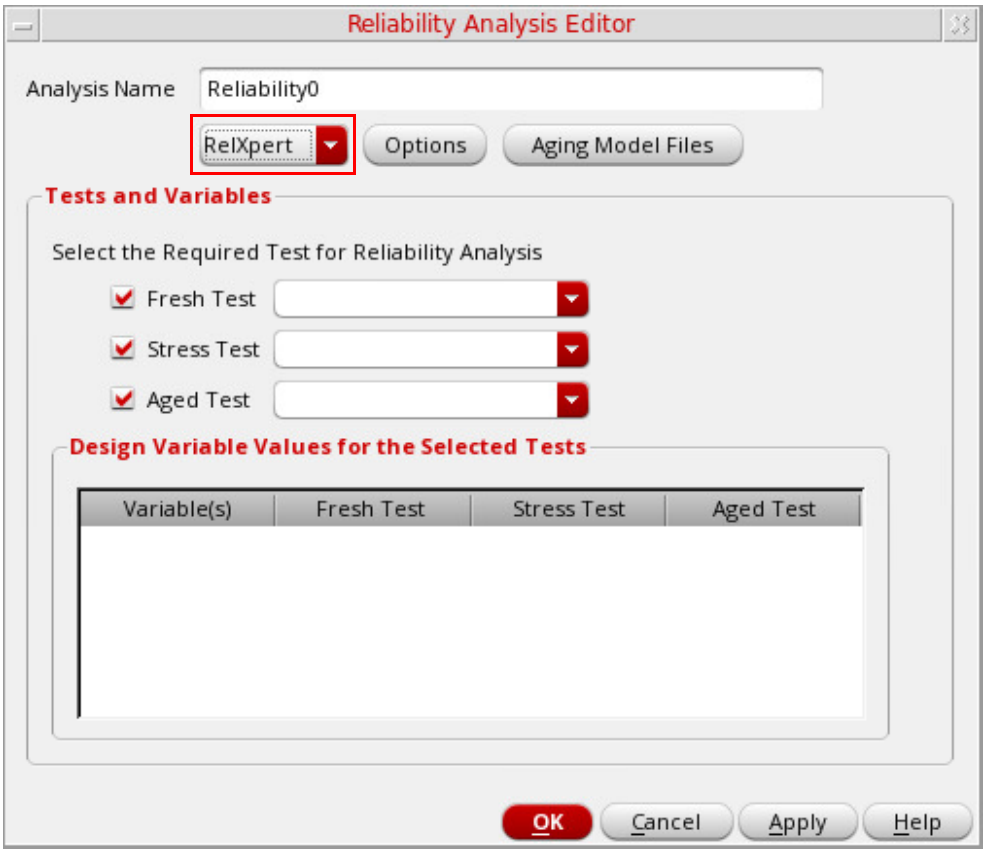
Specifying Simulator Mode for Reliability Analysis

Note: Before specifying the simulator mode, ensure that the following environment is set to

© 2023 Cadence Design Systems, Inc.

```
envSetVal("maestro.gui" "showNativeRelxpertSwitch" 'boolean t)
```

The default simulator mode to perform reliability analysis is Spectre native. However, you can choose RelXpert mode by selecting `RelXpert` from the drop-down list shown in the following figure.



Specifying Aging Model Files

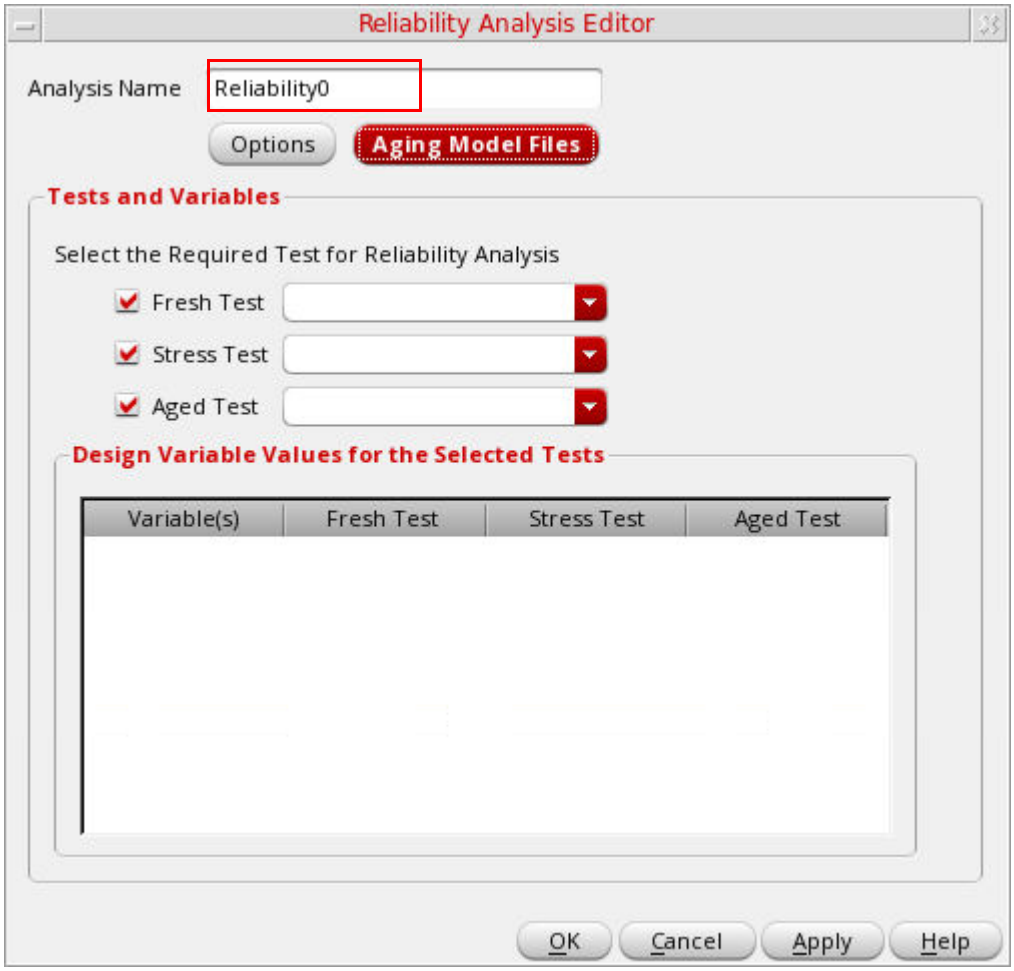
Note: Before specifying the aging model files, ensure that the following environment is set to true.

```
envSetVal("maestro.gui" "displayAgingModelFiles" 'boolean t)
```

Perform the following steps to specify the model files to be used for aging or self-heating analysis for the stress or aged tests:

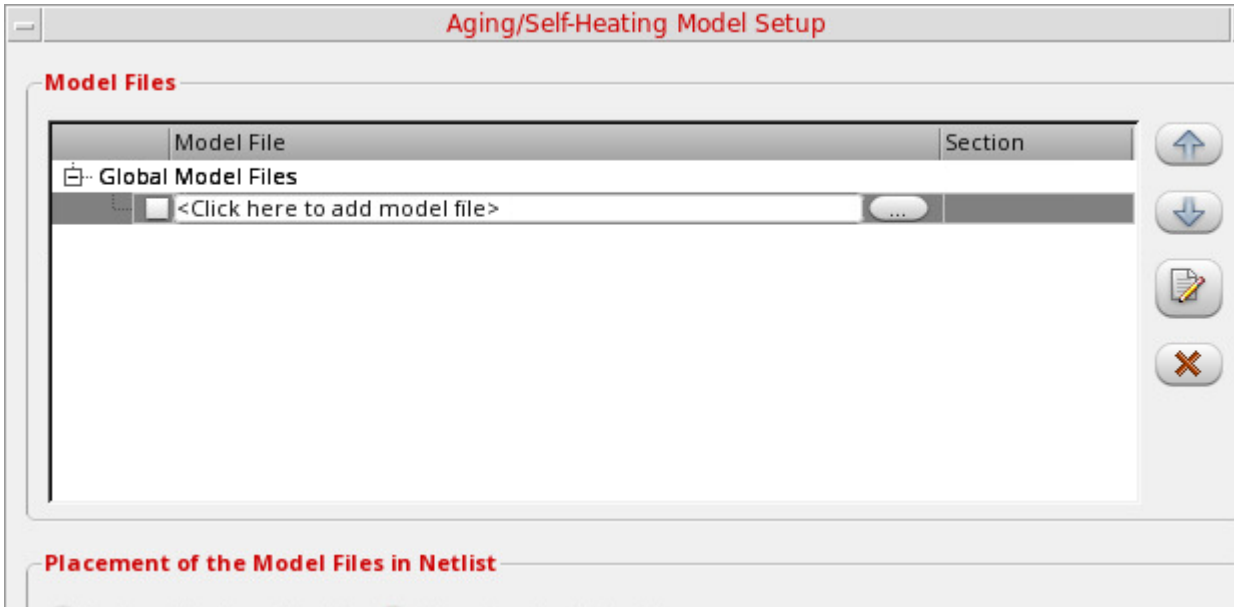
© 2023 Cadence Design Systems, Inc.

1. In the *Reliability Analysis Editor* form, click *Aging Model Files*, as shown in the following figure.



© 2023 Cadence Design Systems, Inc.

The the *Aging/Self-Heating Model Setup* form appears.



2. In the *Model File* column, type the path and file name of the model file you want to use.

   Alternatively, you can click the ellipsis button (…) to open the *Choose Model File* form. Select a valid model file and click *Open*.

3. In the *Section* column, select a section from the drop-down list.

   A model file can have zero or more sections. If a model file contains no sections, there is no drop-down list available. The section you select determines which model definition the simulator uses.

4. In the *Placement of the Model Files in Netlist* section, select one of the following options to define whether you want to print the aging-specific model file before or after the standard model files in the netlist:

   - *Before Standard Model*: Prints the aging model file before the standard model file in the netlist
   - *After Standard Model*: Prints the aging model file after the standard model file in the netlist

   Note: You can also specify these options by setting the defaultAgingPlacementType environment variable.

© 2023 Cadence Design Systems, Inc.

Reusing the Stress File

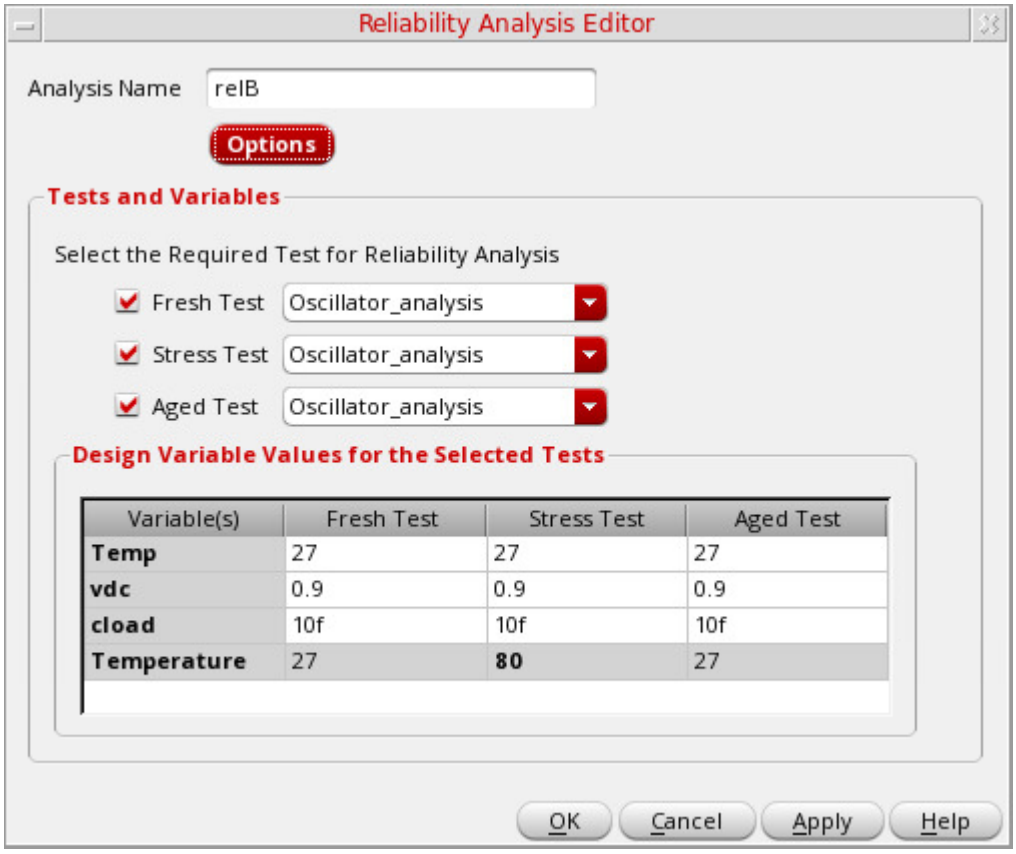
Important

When reusing a stress file between two reliability setups, ensure that *Aging Options* settings are identical in both the setups. If one reliability setup is configured to run a single age point, another setup must also run a single age point. Similarly, if one reliability setup is run with gradual aging configuration, another setup must also run with gradual aging.

To reuse a stress file from one reliability setup to another:

1. From the *Reliability Analyses* node, right-click a setup in which you want to reuse the stress file, and select *Open Setup*.

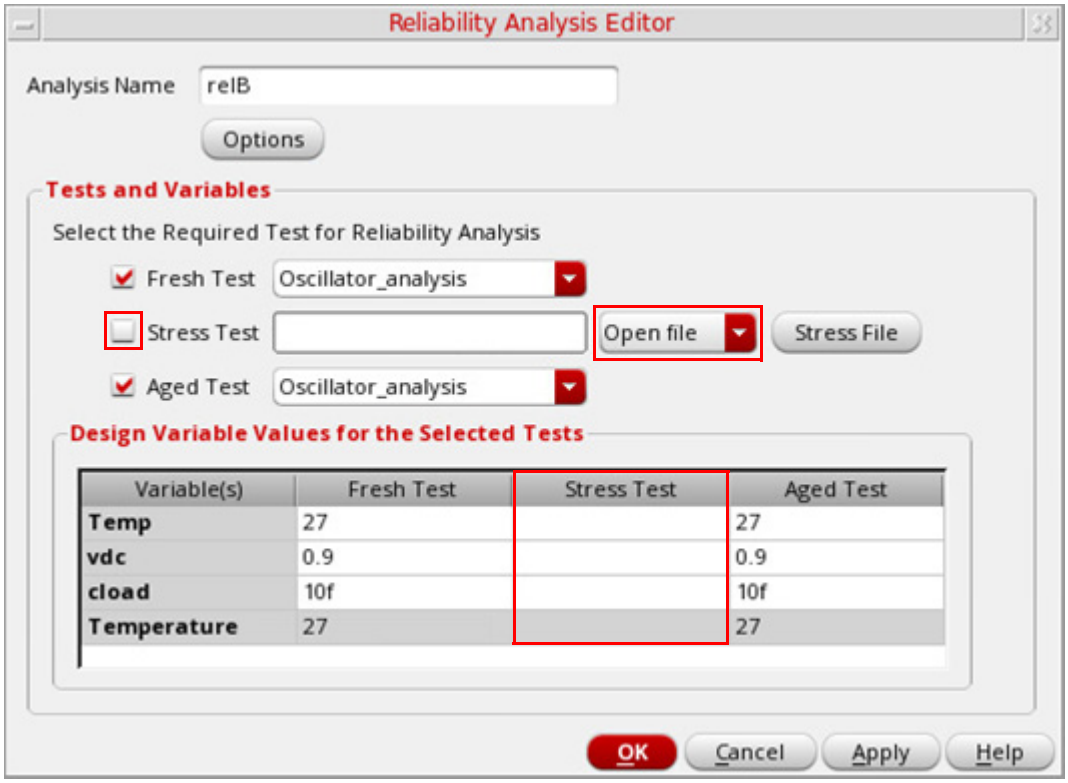
   The Reliability Analysis Editor form opens the selected reliability setup.



2. In the *Reliability Analysis Editor* form, clear the *Stress Test* check box.

© 2023 Cadence Design Systems, Inc.

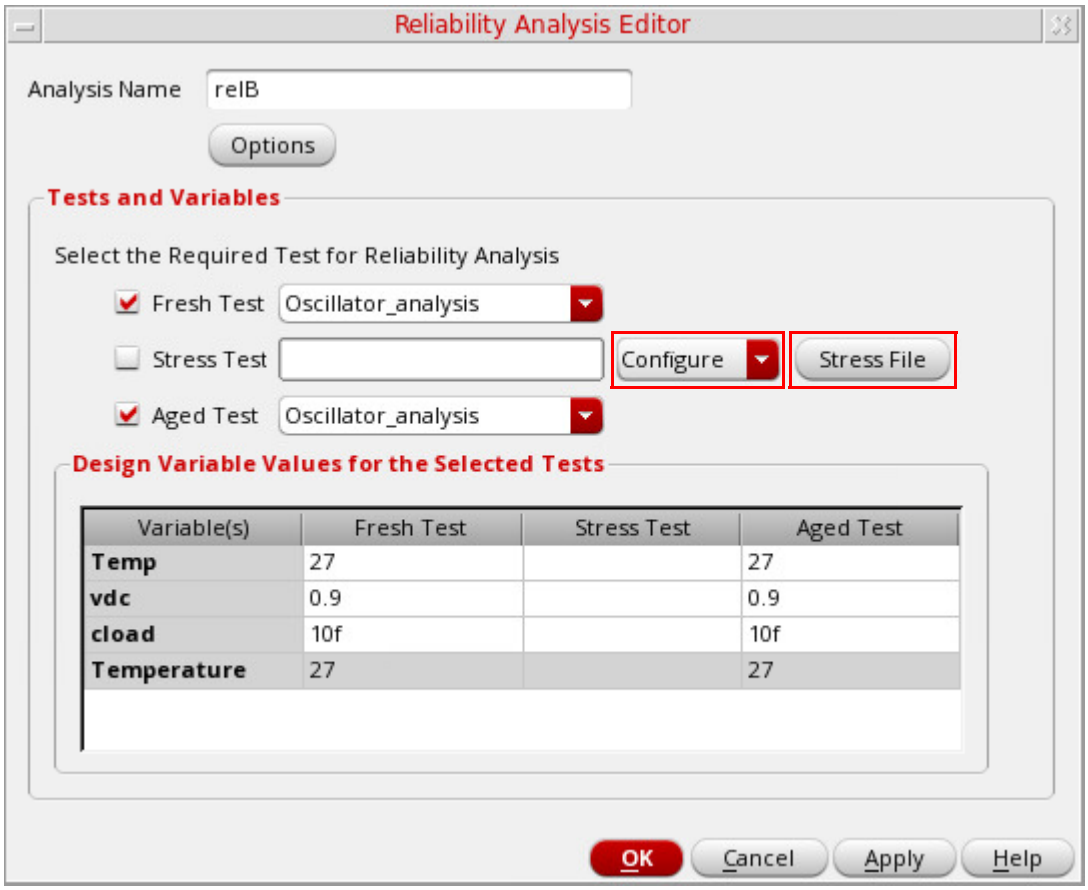
Note that the *Stress Test* column does not display any variables because no stress test is currently selected.



3. To specify the stress file that you want to reuse, do one of the following:
   - Select *Open file* and click *Stress File* to select the stress file.

© 2023 Cadence Design Systems, Inc.

- Select *Configure* and then click *Stress File*.



The *Configure Stress File* form appears.

Specify the following fields to configure the stress file that you want to reuse:

- In the *History Name* field, select a history name.

© 2023 Cadence Design Systems, Inc.

- In the *Reliability Setup Name* field, select the name of the reliability setup from which you want to reuse the stress file.

  You must have at least two reliability setups in the active setup to reuse the stress file. If you have only one reliability setup, an empty string is displayed in the Reliability Setup Name field.

- In the *Run Name* field, select a run name used in the run plan from which you want to reuse the stress file.

- In the *Corner Name* field, select a corner name.

- In the *Match Parameters* field, enter a list of lists containing the name-value pairs of the sweep variables to identify the design point from which you want to reuse the stress file.

  For example, `list(list("cload" "2f") list("vdc" "1.8")).`

  See the `maeGetStressFile` function for more information.

- Click *OK* to close the *Configure Stress File* form.

4. Click *Apply* to save the changes.
5. Click *OK* to close the *Reliability Analysis Editor* form.

Video

You can watch a video demonstration on how to reuse a stress file at Reusing a Stress File in a Reliability Setup. You can also read the related blog at Virtuoso Video Diary: Enhancements in Reliability Analysis.

Reliability Options Form

In the *Reliability Options* form, you can specify the options for reliability analysis. The *Reliability Options* form contains the following four tabs:

- Basic
- Modeling
- Degradation
- Output

Note: *Degradation* and *Output* tabs are disabled if only *Self-Heating* analysis is selected on the *Basic* tab.

© 2023 Cadence Design Systems, Inc.

For detailed information about the fields on all four tabs of the *Reliability Options* form, see Spectre Reliability Analysis in *Spectre Circuit Simulator and Accelerated Parallel Simulator User Guide*.

Video

You can watch a video demonstration on how to use the Reliability Options form at What's New in Reliability Setup. You can also read the related blog at Virtuoso Video Diary: What's New in Reliability Setup.

Basic

Using the *Basic* tab, you can specify analysis type, aging options, start and stop time of reliability analysis, device masking options, and various other options.

The options on the *Basic* tab are grouped in the following sections:

- Analysis
- Aging Options

© 2023 Cadence Design Systems, Inc.

- Stress Test
- Device Masking
- Degradation Ratio
- Lifetime Calculation
- Other Options

Analysis

This section is not available in the RelXpert simulator mode if *AgeMOS* model is selected as the *Aging Model Type* on the *Modeling* tab.

Select the following options to specify the type of analysis to be used for obtaining the reliability values of the devices.The following table describes the options available in the *Analysis* section of the *Basic* tab.

Field	Description
Aging	Select this check box to use the aging flow to obtain the reliability values.
Self-Heating	Select this check box to use the self-heating flow to obtain the reliability values. Environment variable: `enableSelfheating`

Aging Options

This section is available only if the *Aging* analysis is selected in the *Analysis* section of the *Basic* tab.

Field	Description
Single Age Point	Select this option to specify a single age point. Environment variable: `agingPointsType`
Gradual Aging	Select this option to enable gradual aging flow. The gradual aging flow is not supported for self-heating analysis if Spectre native mode is selected. Environment variable: `agingPointsType`

© 2023 Cadence Design Systems, Inc.

Field	Description
Age	Specify the time (age points) at which the device degradation is calculated. For example, `10` If the aging option is set to *Gradual Aging* and mode is set to *Age Points*, you can specify more than one age points in a list of space-separated values. For example, `10 20 30` Environment variables: `mosAgingTime` and `gaAgePoints`
Age Unit	Select the unit for the age points. Possible values are: `Years`, `Days`, `Hours`, `Minutes`, `Seconds` Environment variables: `mosAgingTimeUnit` and `gaAgePointsUnit`
Mode	Select one of the following *Gradual Aging* modes: ■ `Age Points`: Select this option to define the selected age points to perform reliability analysis. ■ `Age Sweep`: Select this option to perform reliability analysis from the start time to the stop time. You then need to define the total number of age steps, start and stop time to perform reliability analysis, unit to measure the start and stop time of each step, and the type of age step. Environment variable: `gaMode`
Dynamic Parameter	Select this check box to set different values of a design variable for different age points. After selecting this check box, double-click in the *Variables* cell to select a design variable, and then set the values in the <Age Points> columns. If you do not select this check box, the same value of the design variable is used for all the age points. This check box is available only when Spectre 18.1 ISR6 or a later version is installed and the `Age Points` mode is selected in the Gradual Aging flow.

© 2023 Cadence Design Systems, Inc.

Field	Description
Save Intermediate Results	■ Select one of the following options to specify the age points for which gradual aging results are to be saved: ■ `All`: Saves the aging results for all the age points specified in the Age field. ■ `Some`: Saves the aging results for the age points based on the index values specified in the text field displayed next to it. After you select this option, specify the index values of the age points for which you want to save the aging results. You can specify any integer value greater than zero as the index. **Note:** The index value for the first age point starts with 1. If you have specified `10 15 20 25 30` in the *Age* field, and want to save the aging results for the intermediate age points `15` and `25`, specify `2 4` in the text field. The aging results for the last age point are always saved by default. Environment variable: `gaSaveSome` ■ `None`: Saves the aging results only for the last age point Environment variable: `gaSaveResults`
Pass Param	Select this check box to pass the URI instance state to the next gradual aging step. This field is applicable only for the gradual aging flow.
Profile	Enables profile gradual aging. In profile gradual aging, the age is calculated independently of the age points because all age points are run with reference to `age = zero`. This means that the result of an age point does not depend upon the results of previous age points. Therefore, the order of the age points is not important in profile gradual aging and the equivalent age is the sum of age values of all age points.

© 2023 Cadence Design Systems, Inc.

Field	Description

Type	Select one of the following options to specify the age step method for the `Age Sweep` mode of the gradual aging flow:
	■ `lin`: Select this option if you want to use the linear method to calculate the time step. If you select this method, the time step is calculated using the following formula:
	`(Stop Time – Start Time) / (Total Steps –1)`
	■ `log`: Select this option if you want to use the logarithm model to calculate the time step. If you select this method, the time step is calculated using the following formula:
	`[log(Stop Time – Start Time)] / (Total Steps –1)`
	Environment variable: `gaAgeStepType`

Unit	Select the unit to measure the start and stop time for each step: Possible values are: `Years`, `Days`, `Hours`, `Minutes`, `Seconds` Environment variable: `gaAgeStepUnit`
Start Time	Specify the start time of the age step in gradual aging flow. Environment variable: `gaAgeStepStart`
Stop Time	Specify the stop time of the age step in gradual aging flow. Environment variable: `gaAgeStepStop`
Total Steps	Specify the total number of steps to perform the reliability analysis. Environment variable: `gaAgeStepTotalStep`
Pass Param	Select this check box to pass the data structure of a URI instance state from a previous aging step to the next gradual aging step. The data structure is copied to a temporary file before it is passed to the next aging step. This option is available when you set up reliability options in the RelXpert mode.

© 2023 Cadence Design Systems, Inc.

Stress Test

The following table describes the options available in the *Stress Test* section of the *Basic* tab.

Field	Description
Transient Time Window for Stress Test	Select this check box if you want to define the start and stop time for reliability analysis during transient simulation. The start and stop time define the transient simulation range. Environment variable: `enableRelXTran`
Single Window	Lets you specify a single time window for the stress test. You can then specify the time window in the *Start Time* and *Stop Time* fields
Start Time	Specify the start time of reliability analysis during transient simulation. **Note:** You can also specify the start time using the `VAR` function. Environment variable: `relxTranStart`
Stop Time	Specify the stop time of reliability analysis during transient simulation. **Note:** You can also specify the stop time using the `VAR` function. Environment variable: `relxTranStop`
Multiple Windows	Lets you specify multiple time windows for the stress test. You must then specify space-separated value pairs for start and stop times in the *Multiple Windows* text field. For example, if you want to specify three time windows in which the start and stop times of the first time window are `10n` and `30n`, those of the second time window are `50n 70n`, and those of the third time window are `90n 110n`, specify the following in this field: `10n 30n 50n 70n 90n 110n`

© 2023 Cadence Design Systems, Inc.

Device Masking

The following table describes the options available in the *Device Masking* section of the *Basic* tab.

Field	Description
Mask Device	Select this check box if you want to include or exclude: ■ models that belong to the subcircuits listed in the subckt list ■ devices that belong to the models listed in the model list ■ devices that are listed in the instance list Environment variable: `mosMaskDev`
Include/Exclude Devices	Select one of the following options: ■ `Include`: Performs reliability analysis only on the specified devices, or the models that belong to the listed subcircuit, or devices that belong to the listed models. ■ `Exclude`: Excludes the listed devices, or the models that belong to the listed sub circuit, or the devices that belong to the specified model from the reliability analysis. Environment variable: `mosMaskDevInclude`
Models List	Specify the name of the models for which the related devices are included or excluded when performing reliability analysis.
Subcircuit In Schematic	Select a subcircuit from the schematic and then click *Add* to add the selected subcircuit to *Subcircuit List*. You can click *Refresh* to get the latest subcircuit list from the schematic. The time taken to refresh the subcircuit list depends upon the size of the schematic.
Subcircuit List	Specify the subcircuits for which the related models are included or excluded when performing the reliability analysis.
Devices List	Select to specify the instances that you want to include or exclude when performing the reliability analysis. Environment variable: `mosMaskDevInstance`

See the Masking Devices section for more information.

© 2023 Cadence Design Systems, Inc.

Degradation Ratio

Note: This section is available in the Spectre native mode only if the *Aging* analysis is selected in the *Analysis* section of the *Basic* tab. This section is not available in the RelXpert mode.

The following table describes the options available in the *Degradation Ratio* section of the *Basic* tab.

Field	Description
Enable Degradation Ratio	Select this check box if you want to specify the HCI, BTI, NBTI, and PBTI weighting ratio in total degradation for the selected devices.
Hot-Carrier Injection (HCI)	Specify the HCI degradation weighting ratio in total degradation. The specified value must be between `0-1`.
Bias Temperature Instability (BTI)	Specify the BTI degradation weighting ratio in total degradation. The specified value must be between `0-1`.
Negative Bias Temperature Instability (NBTI)	Specify the NBTI degradation weighting ratio in total degradation. The specified value must be between `0-1`.
Positive Bias Temperature Instability (PBTI)	Specify the PBTI degradation weighting ratio in total degradation. The specified value must be between `0-1`.

Note: The *Degradation Ratio* options are not available when you set up reliability options in RelXpert mode.

Lifetime Calculation

Note: This section is available only if the *Aging* analysis is selected in the *Analysis* section of the *Basic* tab. This section is not available in RelXpert mode.

© 2023 Cadence Design Systems, Inc.

The following table describes the options available in the *Lifetime Calculation* section of the *Basic* tab.

Field	Description
Calculate Lifetime of Devices	Select this check box if you want to calculate the lifetime of the devices. Environment variable: `mosCalcLifetime`
Degradation Value	Specify the name-value pairs of the device names and threshold degradation values of the electrical properties (`Idsat`, `Vtlin`, `Idlin`) to calculate the lifetime of devices. You can separate the name-value pairs of different devices by using either a comma or a semicolon. For example: If you set the `nmos 0.1, pmos 0.2` value in this field, lifetime will be calculated for: ■ `nmos` devices whose electrical properties have degraded by `10%` or more. ■ `pmos` devices whose electrical properties have degraded by `20%` or more. Environment variable: `mosDeltad`

© 2023 Cadence Design Systems, Inc.

Other Options

The following table describes the options available in the *Other Options* section of the *Basic* tab.

Field	Description

Sub-Blocks Mapping File	Specifies the name of the file to which mapping information between the sub-block and the larger block of a design is saved. Using this mapping information, stress file generated from the stress simulation on the larger block or the full chip can be served as an input for the aging simulation of the sub-block.
	
	The information of the mapping file is also printed in the aged netlist when you run reliability simulation:
	
	`rel reliability {`
	
	`enable_ade_process value=yes`
	
	`age time=[10.0000y]`
	
	`deltad value=0.1`
	
	`accuracy level=1`
	
	`simmode type=aging file="../OSC_TEST/input.bs0" mapping="mapping_file"`
	
	`tran tran stop=10u errpreset=conservative write="spectre.ic" \`
	
	`writefinal="spectre.fc" annotate=status maxiters=5`
	
	`finalTimeOP_aged info what=oppoint where=rawfile`
	
	`}`

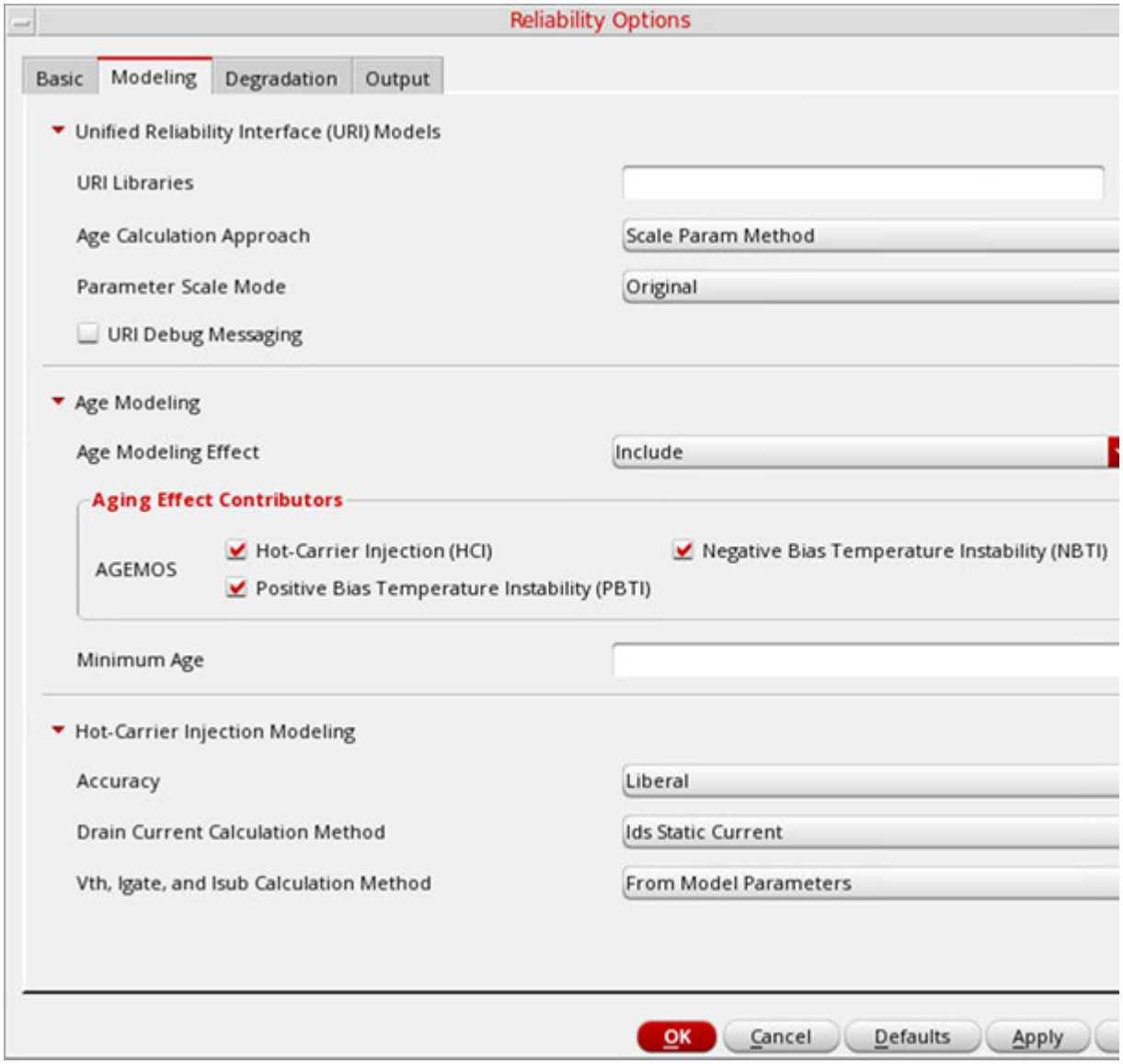
Additional Analysis Arguments	Specify the additional arguments to be considered during reliability simulation. Use either a semicolon or a comma to specify more than one arguments. For example, `.agelevel_only value=[(113 nmos)];.vthmethod spice` or `.agelevel_only value=[(113 nmos)],.vthmethod spice` Environment variable: `otherRXOptions`

© 2023 Cadence Design Systems, Inc.

Field	Description
RelXpert Command Line Option	Specify the command-line options that you want to send to the simulator. This option is available only for RelXpert simulator mode.

Modeling

Using the *Modeling* tab, you can specify the path to the Unified Reliability Interface (URI) libraries and methods to perform aging simulations.



© 2023 Cadence Design Systems, Inc.

The options on the *Modeling* tab are grouped in the following sections:

- Aging Model Type
- Unified Reliability Interface (URI) Models
- Age Modeling
- Hot-Carrier Injection Modeling

Aging Model Type

Important

This section is available only for foundry-specific models.

Unified Reliability Interface (URI) Models

The following table describes the options available in the *Unified Reliability Interface (URI) Models* section of the *Modeling* tab.

Field	Description
URI Libraries	Specify the full path to load the URI libraries or click the *Browse* (...) button to select the libraries. ■ If no URI library was earlier selected, clicking the *Browse* button redirects you to the current working directory. ■ If a URI library was earlier selected, clicking the *Browse* button redirects you to the directory of the previously selected URI library. Environment variable: `uriLibs`
Age Calculation Approach	Select one of the following age calculation methods: ■ *Scale Param Method* ■ *Append Age Method*

© 2023 Cadence Design Systems, Inc.

Field	Description
Append Type	**Note:** This option is available in the RelXpert mode only if the *Append Age Method* is selected as the *Age Calculation Approach*. Select one of the following options to specify the type of circuit elements to which aging results are to be appended: ■ *All Subcircuits*: To append age values to all the subcircuits. However, the age is not appended to the primitive devices. ■ *Standard Subcircuits Only*: To append age values only to the standard subcircuits. ■ *Inline Subcircuits Only*: To append age values only to inline subcircuits. ■ *Primitive Devices*: To append age values only to the primitive devices.
Parameter Scale Mode	Specifies the parameter scaling mode for URI library. Possible values are: ■ *Effective*: Effective parameter is passed to URI for scaling. ■ *Original*: Original parameter is passed to URI for scaling.

Bias Depend Type	Specifies the source on which the aging model is dependent.
	
	Possible values are *All*, *Voltage*, and *None*.
	
	The default value is *All*.
	
	The following code shows an example of the statement printed in the netlist when you set this option to voltage.
	
	`uri_lib_file="/home/user/libURI.so" \`
	`debug=0 scale_mode=effective bias_depend_type=voltage`
	`minage value=1e-30`

URI Debug Messaging	Select this check box to print the debug information. Environment variable: `uriDebugMode`

© 2023 Cadence Design Systems, Inc.

Age Modeling

The following table describes the options available in the *Age Modeling* section of the *Modeling* tab.

Field	Description
Age Modeling Effect	Select one of the following options: ■ *Include*: Includes the effects of the selected AGEMOS aging effect contributors and the specified URI age levels when calculating the device degradation results. ■ *Exclude*: Excludes the effects of the selected AGEMOS aging effect contributors and the specified URI age levels when calculating the device degradation results.

© 2023 Cadence Design Systems, Inc.

Field	Description
Aging Effect Contributors	■ *AGEMOS*: Select the following options to specify the aging effects that you want to include or exclude when calculating the device degradation results: ❑ *Hot-Carrier Injection (HCI)*: To calculate the transistor and circuit performance degradation due to HCI effect. **Note:** If you clear this check box, Hot Carrier Injection Modeling section becomes unavailable on the *Modeling* tab. ❑ *Negative Bias Temperature Instability (NBTI)*: To calculate PMOSFET and circuit performance degradation NBTI effect. ❑ *Positive Bias Temperature Instability (PBTI)*: To calculate NMOSFET and circuit performance degradation due to PBTI effect. Environment variable: `mosMode` ■ *URI Models*: In the *Age Levels* field, enter the age level numbers corresponding to the aging effect defined in the URI models that you want to include or exclude when calculating the device degradation results. The default value is `All`. **Note:** The *URI Models* and *Age Levels* fields are displayed only when you have selected the URI model to be used from the URI Libraries field. See the Different Configurations of AGEMOS and URI Contributors section for more information.
Minimum Age	Specify the smallest Age value for which degraded SPICE model parameters are calculated. Use either decimal notation (`xx.xx`) or engineering notation (`x.xxe+xx`) to specify the value. Environment variable: `mosMinAge`

© 2023 Cadence Design Systems, Inc.

Field	Description
Age Method	**Note:** These options are available only for the RelXpert simulator mode if AgeMOS (Spectre Internal) is selected as the Aging Model Type. Options in this group enables you to specify one of the following methods for calculating degraded SPICE model parameters for aged circuit simulation: ■ AgeMOS method: To select this method, choose `AgeMOS` from the *Method* drop-down list. This is the default method. ■ Method of interpolation from aged files in linear-linear domain: To select this method, choose `Interpolation` from the *Method* drop-down list and `linlin` from the *Scale* drop-down list. ■ Method of interpolation from aged files in linear-log domain: To select this method, choose `Interpolation` from the *Method* drop-down list and `linlog` from the *Scale* drop-down list. ■ Method of interpolation from aged files in log-log domain: To select this method, choose `Interpolation` from the *Method* drop-down list and `loglog` from the *Scale* drop-down list. ■ Method of regression from aged files in linear-linear domain: To select this method, choose `Regression` from the *Method* drop-down list and `linlin` from the *Scale* drop-down list. ■ Method of regression from aged files in linear-log domain: To select this method, choose `Regression` from the *Method* drop-down list and `linlog` from the *Scale* drop-down list. ■ Method of interpolation from aged files in log-log domain: To select this method, choose `Regression` from the *Method* drop-down list and `loglog` from the *Scale* drop-down list.

© 2023 Cadence Design Systems, Inc.

Field	Description
Age Method	■ *Effective Model Calculation*: By default, RelXpert performs aged parameter scaling using the effective parameters. Use this option to enable RelXpert perform scaling of aged parameters without using the effective parameters. Environment variables: `mosAgeDomain` and `mosAgeMethod`
Append Device Age	Select this check box to append age value in the definition statements of all degraded devices in the netlist file. This option is available only for the RelXpert simulator mode if AgeMOS (Spectre Internal) is selected as the Aging Model Type.
Modify Netlist Before Aging Starts	Select this check box to enable the user to review and edit the netlist before the aging simulation is run. This option is available only for the RelXpert simulator mode if AgeMOS (Spectre Internal) is selected as the Aging Model Type.
Enable Maximum Number of Aged Models	Select this check box to specify the maximum number of aged models to be generated. This check box is available only for the RelXpert simulator mode.
Maximum Number of Aged Models	Specify the maximum number of models (excluding the fresh model) to be generated based on the selected grouping method. If the device degraded more than the last aged model, additional models will be generated using extrapolation from the last two aged model. This options appears dimmed and becomes available when the Enable Maximum Number of Aged Models check box is selected. This option is available only for the RelXpert simulator mode if AgeMOS (Spectre Internal) is selected as the Aging Model Type.

© 2023 Cadence Design Systems, Inc.

Field	Description
Grouping Method	**Note:** These options are available only for the RelXpert simulator mode if AgeMOS (Spectre Internal) is selected as the Aging Model Type. Select one of the following methods to group devices with similar degradations. If the number of degraded devices is large, grouping devices with similar degradation reduces the size of degraded netlist file. ■ *Age*: Interpolates the models based on age values in log domain. This is the default. ■ *Degradation*: Interpolates the models based on degradation values using linear scale.
Show Unaged Devices	Select one of the following options to specify whether to list the unaged devices in the output: ■ *First 20*: Only first 20 unaged devices are listed in the ouput ■ *All*: All unaged devices are listed in the output ■ *None*: Unaged devices are not listed in the output **Note:** These options are available only for the RelXpert simulator mode if AgeMOS (Spectre Internal) is selected as the Aging Model Type.
Include Files	Specify the filename whose contents you want to include in the input netlist file. You can also click the ellipsis (…) button to select the file. **Note:** This option is available only for the RelXpert simulator mode if AgeMOS (Spectre Internal) is selected as the Aging Model Type.

© 2023 Cadence Design Systems, Inc.

Field	Description
Compact Aged Netlist	**Note:** These options are available only for the RelXpert simulator mode if AgeMOS (Spectre Internal) is selected as the Aging Model Type. In RelXpert, a subcircuit is expanded if a model name is changed to an aged model name. If the subcircuit definition is large, it causes the netlist size to become huge after expansion and it becomes extremely difficult to open the netlist. You can use the `Analogmodel` feature of Spectre in the aged netlist to change the model name. Therefore, the subcircuit is not expanded, which reduces the netlist size considerably. You can also use the `model-in-model` feature of Spectre to output an aged netlist file in which the shared model parameters are listed only once for devices sharing the same model in the netlist. This also reduces the size of the netlist to a large extent. Select one of the following options to specify which Spectre feature you want to use to reduce the size of aged netlist file: ■ *Models*: If you want to use the `model-in-model` feature of Spectre to output an aged netlist file. ■ *Subcircuit Macromodels*: If you want to use the `Analogmodel` feature of Spectre to change the model name in the netlist. ■ *All*: If you want to use both the Spectre features to reduce the size of netlist file. ■ *None*: If you do not want to use any of the Spectre features to reduce the size of the netlist file.

Different Configurations of AGEMOS and URI Contributors

The following table describes some examples of how you can include or exclude the aging effect contributors of the AGEMOS and URI models when calculating the device degradation results.

© 2023 Cadence Design Systems, Inc.

Age Modeling Effect	Aging Effect Contributors			Description
	AGEMOS		URI Models	
Include	*HCI*	☑	*Age Levels* field is left blank	Includes all the contributors of the AGEMOS and URI models when calculating the device degradation results.
	NBTI	☑		
	PBTI	☑		
Exclude	*HCI*	☑	*Age Levels* field is left blank	■ Includes all the contributors of URI model when calculating the device degradation results. ■ Excludes all the contributors of the AGEMOS model
	NBTI	☑		
	PBTI	☑		
Exclude	*HCI*	☑	103	■ Excludes only the contributor corresponding to the age level `103` defined in the URI model ■ Excludes all the contributors of the AGEMOS model
	NBTI	☑		
	PBTI	☑		

Hot-Carrier Injection Modeling

Note: This section is not available if *Hot-Carrier Injection (HCI)* is not selected as one of the *Aging Effect Contributors* in the *Age Modeling* section.

© 2023 Cadence Design Systems, Inc.

The following table describes the options available in the *Hot-Carrier Injection Modeling* section of the *Modeling* tab.

Field	Description
Accuracy	Select one of the following options to specify the method used in the reliability analysis when performing integration and substrate current calculation. ■ *Liberal*: Select this option if you want to use backward Euler integration. Sets `Isub = 0`, when `Vgs < Vth`. This is the default value. ■ *Conservative*: Select this option if you want to use trapezoidal integration. Calculates `Isub`, when `Vgs > Vth`. Environment variable: `mosHciAccuracy`
Drain Current Calculation Method	Select one of the following options to specify how the simulator obtains the drain current to perform reliability analysis. ■ *Ids Static Current*: Select this option to use the static drain current, which is also called as the channel drain current. This is the default value. ■ *Id Dynamic Current*: Select this option to use the dynamic drain current, which is also called as the AC drain current. ■ *Drain Terminal Static Current*: Select this option to use the terminal static current. Environment variable: `mosHciIdMethod`
Vth, Igate, and Isub Calculation Method	Select one of the following options to specify the method used for obtaining the gate terminal current or substrate terminal current: ■ *From Model Parameters*: Select this option to calculate the substrate or gate terminal current using the model parameters. This is the default value. ■ *From Simulation Results*: Select this option to calculate the substrate or gate terminal current using the simulation results. Environment variable: `mosHciOpMethod`

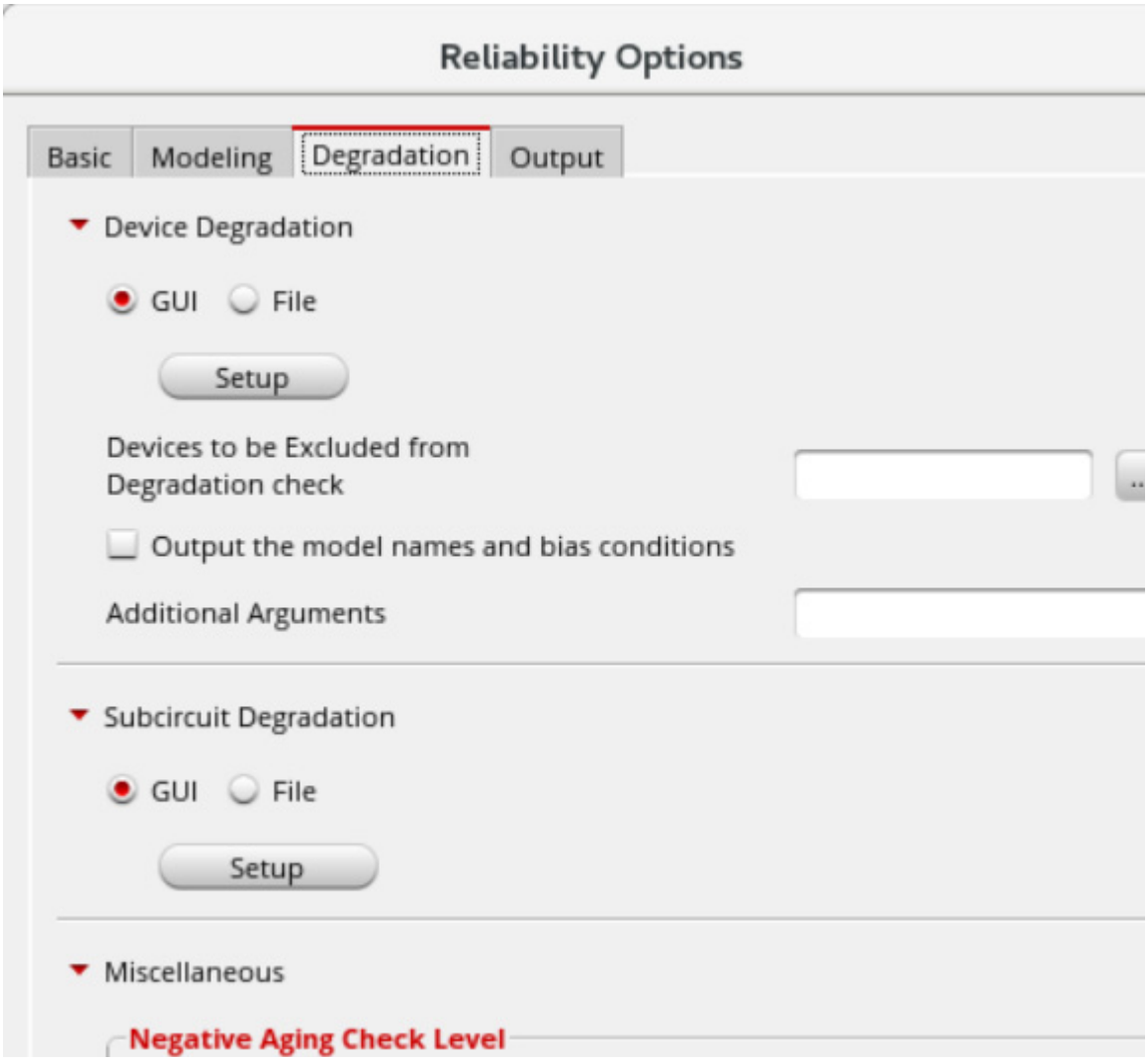
© 2023 Cadence Design Systems, Inc.

Field	Description
Lifetime Parameter Calculation Method	**Note:** This option is available only in the RelXpert simulator mode. Select one of the following options to specify the method used for obtaining lifetime parameters for MOSFET Hot-Carrier simulation: ■ *From Model Parameters*: Select this option to calculate lifetime parameter values using lifetime model equations as described in Hot-Carrier Lifetime model. ■ *From Table File*: Select this option if you want to look up lifetime parameters and interpolate the results from the lifetime parameter file. The filename must contain values for lifetime parameters (that is, h, m, n, hg, and mg under different vgd, vbs, vds, vg/vd, vb/vs, or vd/vs bias conditions). Only one bias condition can be specified in the table file.
Lifetime Parameter File	**Note:** This field is available only when you select the From Table File option from the *Lifetime Parameter Calculation Method* drop-down list. Specify the filename relative to the directory from which the reliability simulator is invoked. You can also specify the full path.

© 2023 Cadence Design Systems, Inc.

Degradation

Using the *Degradation* tab, you can specify the device degradation and subcircuit degradation options.



The options on the *Degradation* tab are grouped in the following sections:

- Device Degradation
- Subcircuit Degradation
- Miscellaneous
- Preset

© 2023 Cadence Design Systems, Inc.

Device Degradation

The following table describes the options available in the *Device Degradation* section of the *Degradation* tab.

Field	Description
GUI, File, Setup, and Setup File	Specify these options to print device degradation (`gm`, `gds`, `Idlin`, `Idsat`, `Vth` degradation) information to a `.bt0` file. Refer to the Specifying Device Degradation Values section for more information. Environment variable: `enableOutputDevDegInFile`
Devices to be Excluded from Degradation Check	Specify the models for which the related devices are included or excluded when performing reliability analysis. Environment variable: `degCheckException`
Output the model names and bias conditions	Select this check box if you want to print the model names and bias conditions in the `input.bt0` file.
Additional Arguments	Specifies additional arguments for output device degradation. You can specify multiple space-separated arguments. For example, `top_deg_num=1 op=[vth idsat idlin gm gds]` Environment variable: `otherODDOptions`

Subcircuit Degradation

The following table describes the options available in the *Subcircuit Degradation* section of the *Degradation* tab.

Field	Description
GUI, File, Setup, and Setup File	Specify these options to print the subcircuit degradation (`Idlin`, `Idsat`, `Vth` degradation) information to a `.bt0` file. Since a subcircuit is treated as a MOSFET device, it should have at least four terminals, which map to drain (D), source (S), gain (G), and bulk (B) of the MOSFET. Refer to the Specifying Subcircuit Degradation Values section for more information.

© 2023 Cadence Design Systems, Inc.

Miscellaneous

The following table describes the options grouped in the *Miscellaneous* section on the *Degradation* tab.

Field	Description
Negative Aging Check Level	Specify the type of message to be generated when negative aging occurs: ■ *Error*: Select this option if you want to generate an error message when negative aging occurs. ■ *Warning*: Select this option if you want to generate a warning message when negative aging occurs. ■ *Ignore*: Select this option to ignore all the errors and warnings when negative aging occurs. Environment variable: `checknegagingtype`
Clamp Degradation Values for Negative Aging	Select this check box if you want Spectre to clamp the degradation values to be the same as fresh values and generates a warning message for negative aging. Environment variable: `checknegagingclamp`
Check degradation positive feedback	Select this check box to print `enable_bias_runaway value=yes` in the netlist.

© 2023 Cadence Design Systems, Inc.

Field	Description
Degradation Calculation Method	Select one of the following options to specify the degradation calculation method to calculate the degradation results for a MOSFET. The degradation results are generated in the bt# file.: ■ *Use Aged Model Card*: To use the aged model card to calculate the degradation results. The results are generated in the `bt#` file. ■ *Calculate HCI/BTI Contributions Separately*: To use the HCI, NBTI, and PBTI reliability model's degradation results separately. The HCI and NBTI degradation results are added to calculate the degradation results for PMOSFET, and the HCI and PBTI degradation results are added to calculate the degradation results for NMOSFET. The results are generated in the `bt#` file. **Note:** These options are available only for RelXpert simulator mode.

Preset

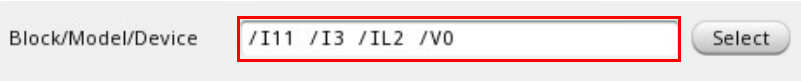
The following table describes the options grouped in the *Miscellaneous* section on the *Degradation* tab.

Field	Description
Preset Age/ Degradation/Lifetime	Select this check box to preset the age, lifetime or degradation value for devices in the netlist file.

© 2023 Cadence Design Systems, Inc.

Field	Description
Preset Type	Select one of the following preset types for a device, model, or block: ■ *Lifetime* ■ *Age* ■ *Degradation* **Note:** If the preset is `age`, the simulator does not perform an age calculation for the specified devices, models, or blocks.If the preset is `lifetime` or `degradation`, the simulator derives age from the preset lifetime or degradation values to calculate degradation or lifetime for the specified devices, models, or blocks.
Preset Value	Specify the value for the selected preset type. ■ For the `lifetime` preset, specify the lifetime value of the device, model, or block. ■ For the `age` preset, specify the age value of the device, model, or block ■ For the `degradation` preset, specify the degradation value of the device, model, or block
Aging Contributor	Specify the age values for the following modeling types: ■ *All* ■ *Hot-Carrier Injection (HCI)* ■ *Negative Bias Temperature Instability (NBTI)* ■ *Positive Bias Temperature Instability*

© 2023 Cadence Design Systems, Inc.

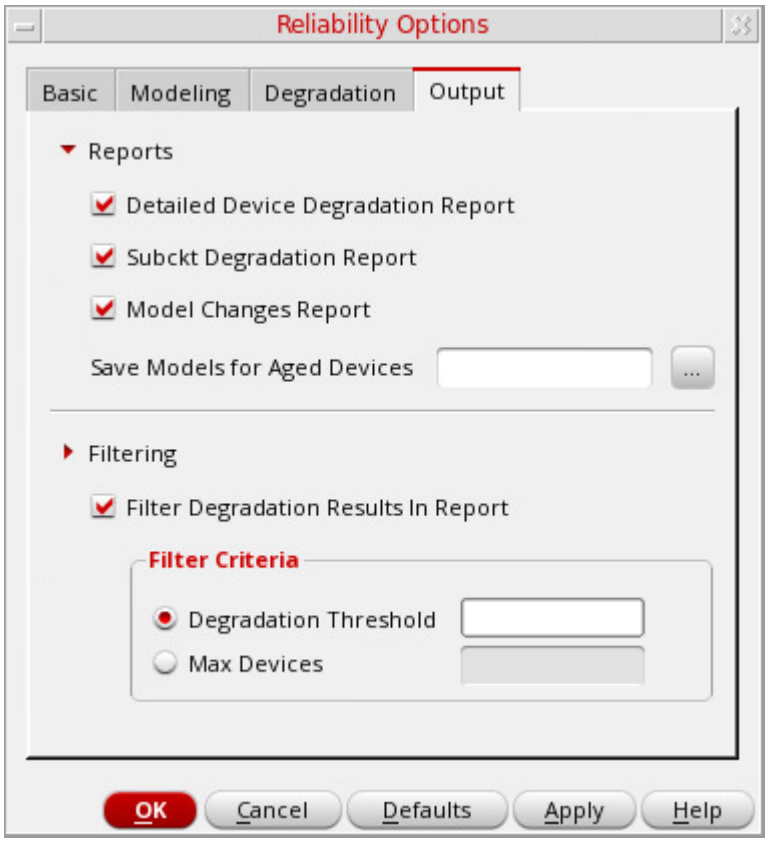
Field	Description
Block/Model/device	Specify the blocks, models, or devices on which the preset values are enabled. Follow these steps to specify the blocks, models, or devices: **1.** Click *Select* to open the schematic. **2.** Select one or more blocks, models, or devices in the schematic. **3.** Press the *Esc* key on the keyboard when you are done. Selected blocks or models are displayed in the *Block/Model/device* field, as shown in the following figure. 

Note: This section is available only for the RelXpert simulator mode.

© 2023 Cadence Design Systems, Inc.

Output

Using the *Output* tab, you can specify the degradation reports that you want to generate and methods to filter degradation results in the reports.



The options on the *Output* tab are grouped in the following sections:

- Reports
- Filtering

Reports

The following table describes the options available in the Reports section on the Output tab.

Field	Description
Detailed Device Degradation Report	Select this check box if you want to view the detailed device degradation report.

© 2023 Cadence Design Systems, Inc.

Field	Description
Subckt Degradation Report	Select this check box if you want to view the subcircuit degradation report. **Note:** This option is not available for the RelXpert simulator mode.
Model Changes Report	Select this check box if you want to print the fresh, effective, and aged parameter values to the `.bm#` file. Environment variable: `reportModelParamChanged`
Save models for Aged Devices	Specify the name of the file in which you want the degraded models to be dumped. Alternatively, you can click the ellipses (...) button to select an existing file. You can also create a new file by specifying its name in the *File name* field on the Choose a File form. Environment variable: `mosAgeModelFilePath`

Filtering

The following table describes the options available in the *Filtering* section on the *Output* tab.

Field	Description
Filter Degradation Results in Report	Select this check box if you want to filter the degradation results in the report. Environment variable: `enableDegSort`

© 2023 Cadence Design Systems, Inc.

Field	Description
Filter Criteria	Select one of the following options to specify the method by which output of all MOS transistors is sorted in the order of descending degradation: ■ *Degradation Threshold:* Prints transistors having degradation values greater than the specified threshold value. Specify the degradation threshold value in the corresponding text field in decimal notation (xx.xx) or in engineering notation (x.xxe+xx). ■ *Max Devices*: Prints only the specified number of transistors that have the highest degradation. Specify the number of maximum devices in the corresponding text field. Environment variables: `degSortType` and `degSortValue`

Specifying Device Degradation Values

The device degradation calculation is based on following fixed bias conditions:

- NMOSFET
  - *Idsat*: `Vds=Vdd`, `Vgs=Vgsat`, if `Vgsat` is specified for the target model, otherwise `Vgs=Vdd`.
  - *Idlin*: `Vds=Vdlin`, if `Vdlin` is specified for the target model, otherwise `Vds=0.05V`; `Vgs=Vglin`, if `Vglin` is specified for the target model, otherwise `Vgs=Vdd`.
  - *gm*: `Vds=Vdlin`, if `Vdlin` is specified for the target model, otherwise `Vds=0.05V`; `Vgs=Vglin`, if `Vglin` is specified for the target model, otherwise `Vgs=Vdd`.
  - *Vth*: `Vds=Vdlin`, if `Vdlin` is specified for the target model, otherwise `Vds=0.05V`; `Vgs=Vglin`, if `Vglin` is specified for the target model, otherwise `Vgs=Vdd`.
- PMOSFET
  - *Idsat*: `Vds=Vdd`, `Vgs=Vgsat`, if `Vgsat` is specified for the target model, otherwise `Vgs=Vdd`.
  - *Idlin*: `Vds=Vdlin`, if `Vdlin` is specified for the target model, otherwise `Vds=-0.05V`; `Vgs=Vglin`, if `Vglin` is specified for the target model, otherwise `Vgs=0.05`.

© 2023 Cadence Design Systems, Inc.

- *gm*: `Vds=Vdlin`, if `Vdlin` is specified for the target model, otherwise `Vds=-0.05V`; `Vgs=Vglin`, if `Vglin` is specified for the target model, otherwise `Vgs=Vdd`.
- *Vth*: `Vds=Vdlin`, if `Vdlin` is specified for the target model, otherwise `Vds=-0.05V`; `Vgs=Vglin`, if `Vglin` is specified for the target model, otherwise `Vgs=0.05`.

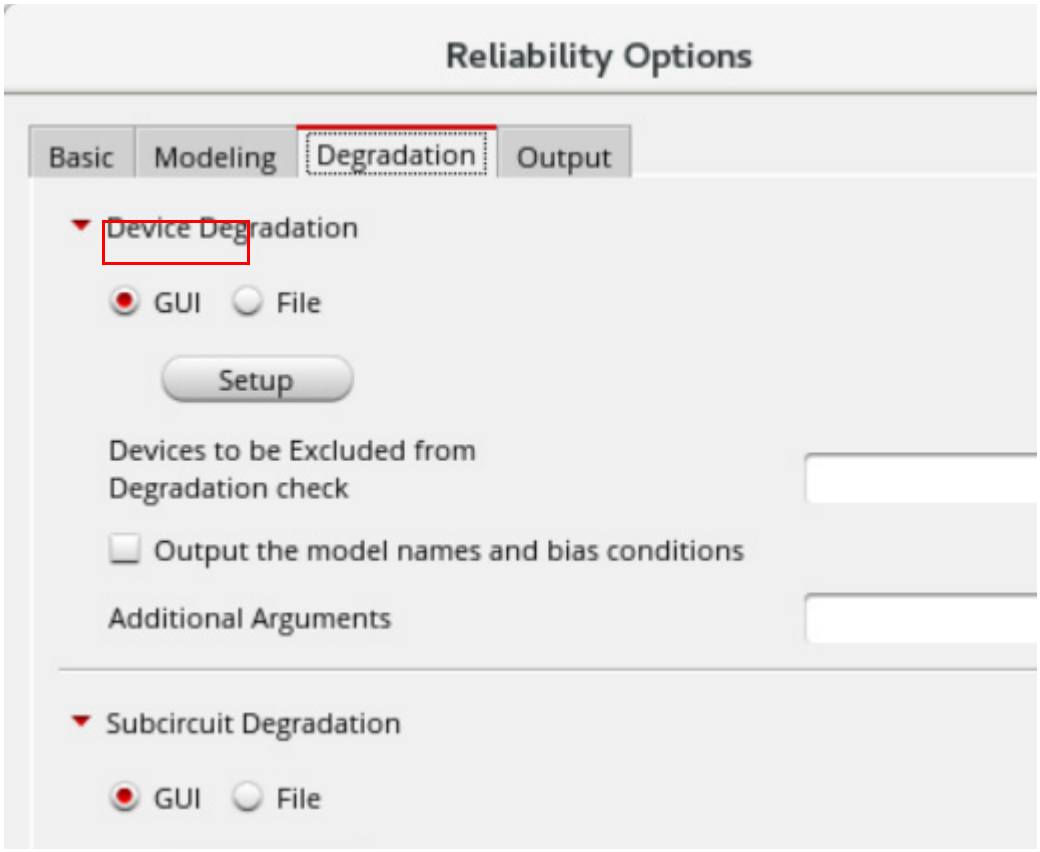
You can specify the device degradation values for the models using one of the following methods:

- Using GUI
- Using Setup File

Using GUI

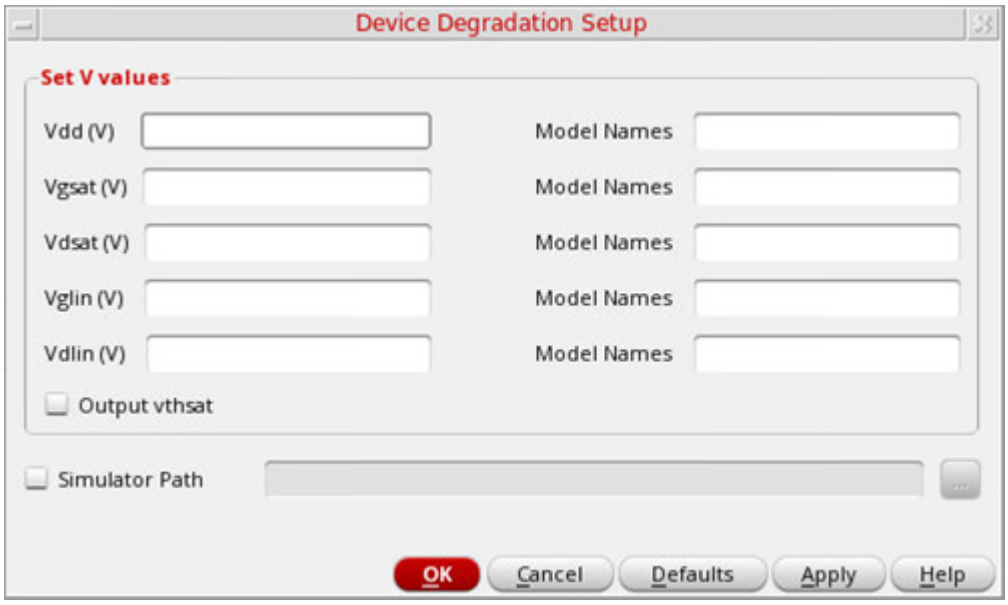
Follow these steps to specify device degradation values through GUI:

1. On the *Degradation* tab, in the *Device Degradation* section, choose *GUI*, and then click *Setup*, as shown in the following figure:



© 2023 Cadence Design Systems, Inc.

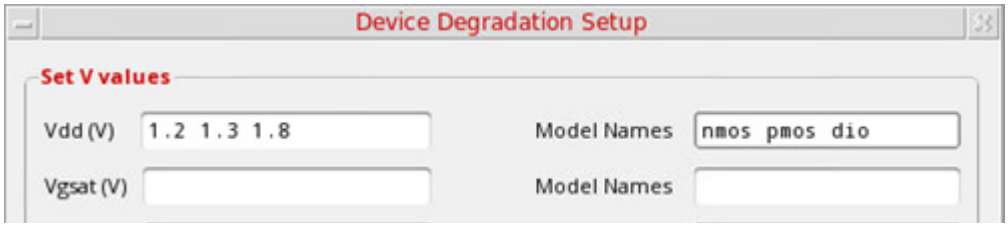
The *Device Degradation Setup* form opens.



2. In the *Set V values* section, do the following:

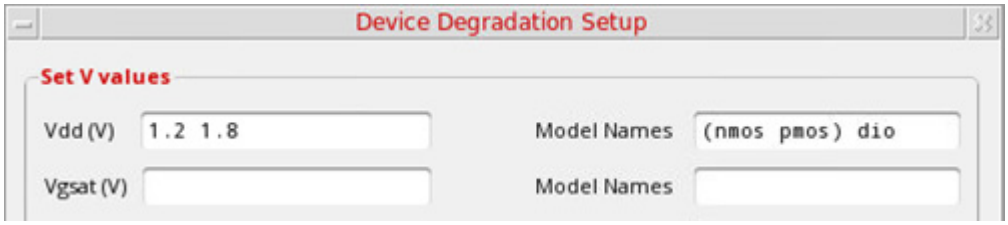
   - Specify the following values for calculating the device degradation values:
     - In the *Vdd (V)* field, specify the bias voltage for the device to obtain `gm`, `gds`, `Idlin`, `Idsat`, and `Vth` degradation.
     - In the *Vgsat (V)* field, specify the `Vgs` value for `Idsat` measurement.
     - In the *Vdsat (V)* field, specify the `Vds` value for `Idsat` measurement.
     - In the *Vglin (V)* field, specify the `Vgs` value for `Idlin/Vt/Gm` measurement.
     - In the *Vdlin (V)* field, specify the `Vds` value for `Idlin/Vt/Gm` measurement
   - For each voltage value, specify the corresponding models in the *Model Names* field. For example, `pmos`, `nmos`, and so on.

   The following example sets the *Vdd* values for `nmos`, `pmos`, and `dio` to `1.2`, `1.3`, and `1.8V` respectively.



© 2023 Cadence Design Systems, Inc.

The following example sets the *Vdd* values for `nmos`, `pmos`, and `dio` to `1.2`, `1.2`, and `1.8V` respectively.



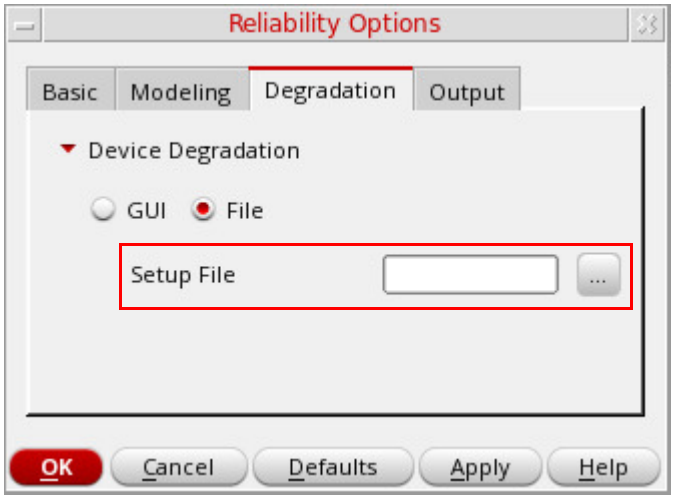
3. Select the *Output vthsat* check box to print `vthsat` information in the `input.bt0` file.
4. Select the *Simulator Path* check box to specify the full path to the simulator that you want to use for aging simulation.

   Note: This field is not available in the Spectre native mode.
5. Click *OK*.

Using Setup File

Follow these steps to specify the device degradation values using a setup file:

1. On the *Degradation* tab, in the *Device Degradation* section, choose *File*. The *Setup File* field is displayed, as shown in the following figure.



2. In the *Setup File* field, specify the path to the setup file containing the device degradation values for the models in use. Alternatively, you can click the ellipsis (…) button to select the setup file.
3. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Specifying Subcircuit Degradation Values

The subcircuit degradation calculation is based on following fixed bias conditions:

- NMOSFET
  - *Idsat*: `Vds=Vdd`, `Vgs=Vgsat` if `Vgsat` is specified for the target subcircuit, otherwise `Vgs=Vdd`.
  - *Idlin*: `Vds=Vdlin` if `Vdlin` is specified for the target subcircuit, otherwise `Vds=0.05V`; `Vgs=Vglin` if `Vglin` is specified for the target subcircuit, otherwise `Vgs=0.05`.
  - *Vth*: `Vds=Vdlin` if `Vdlin` is specified for the target subcircuit, otherwise `Vds=0.05V`; `Vgs=Vglin` if `Vglin` is specified for the target subcircuit, otherwise `Vgs=0.05`.
- PMOSFET
  - *Idsat*: `Vds=Vdd`, `Vgs=Vgsat` if `Vgsat` is specified for the target subcircuit, otherwise `Vgs=Vdd`.
  - *Idlin*: `Vds=Vdlin` if `Vdlin` is specified for the target subcircuit, otherwise `Vds=-0.05V`; `Vgs=Vglin` if `Vglin` is specified for the target subcircuit, otherwise `Vgs=0.05`.
  - *Vth*: `Vds=Vdlin` if `Vdlin` is specified for the target subcircuit, otherwise `Vds=-0.05V`; `Vgs=Vglin` if `Vglin` is specified for the target subcircuit, otherwise `Vgs=0.05`.

You can specify the subcircuit degradation values using one of the following methods:

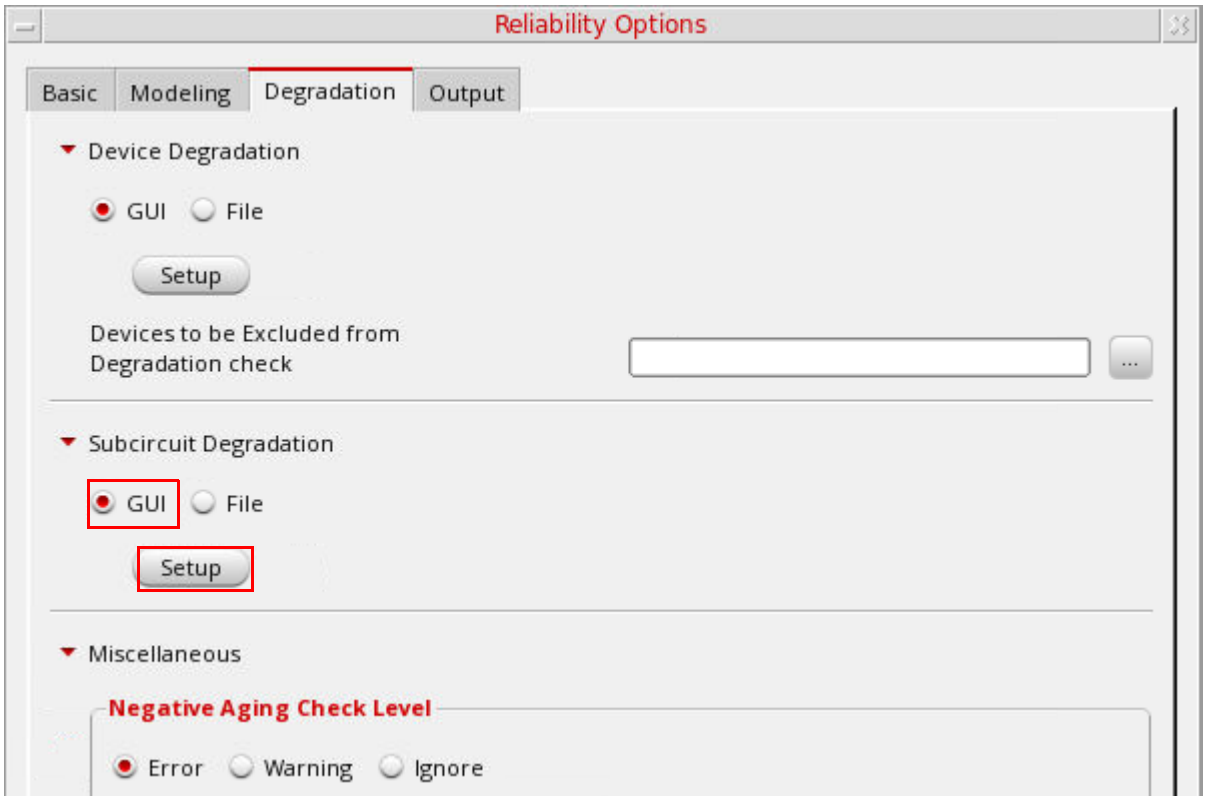
- Using GUI
- Using Setup File

Using GUI

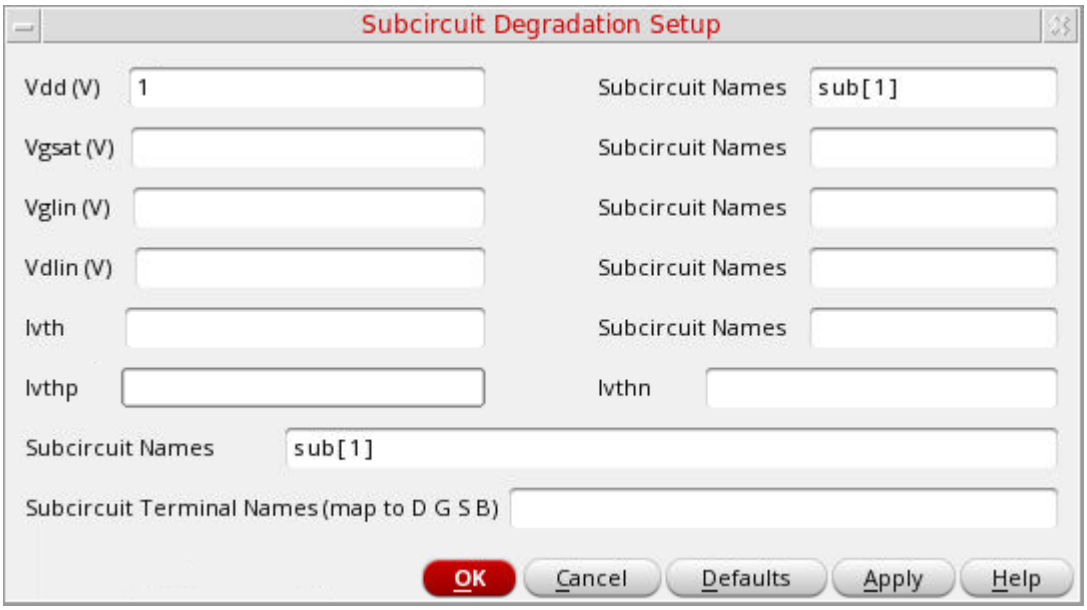
Follow these steps to specify subcircuit degradation values through GUI:

© 2023 Cadence Design Systems, Inc.

1. On the *Degradation* tab, in the *Device Degradation* section, choose *GUI*, and then click *Setup*, as shown in the following figure.



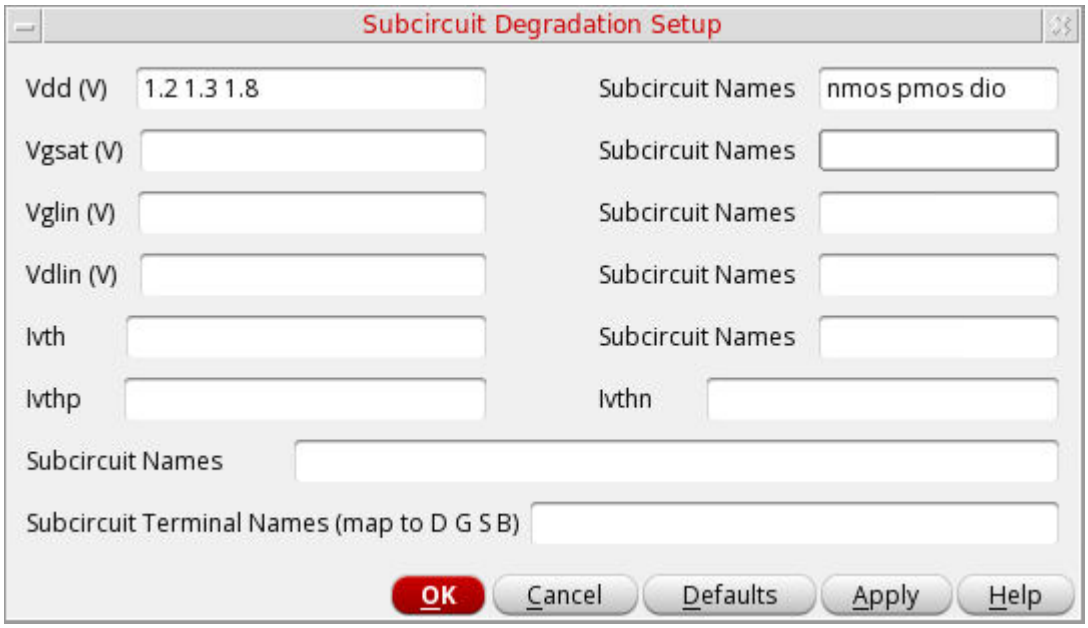
The *Subcircuit Degradation Setup* form opens.



© 2023 Cadence Design Systems, Inc.

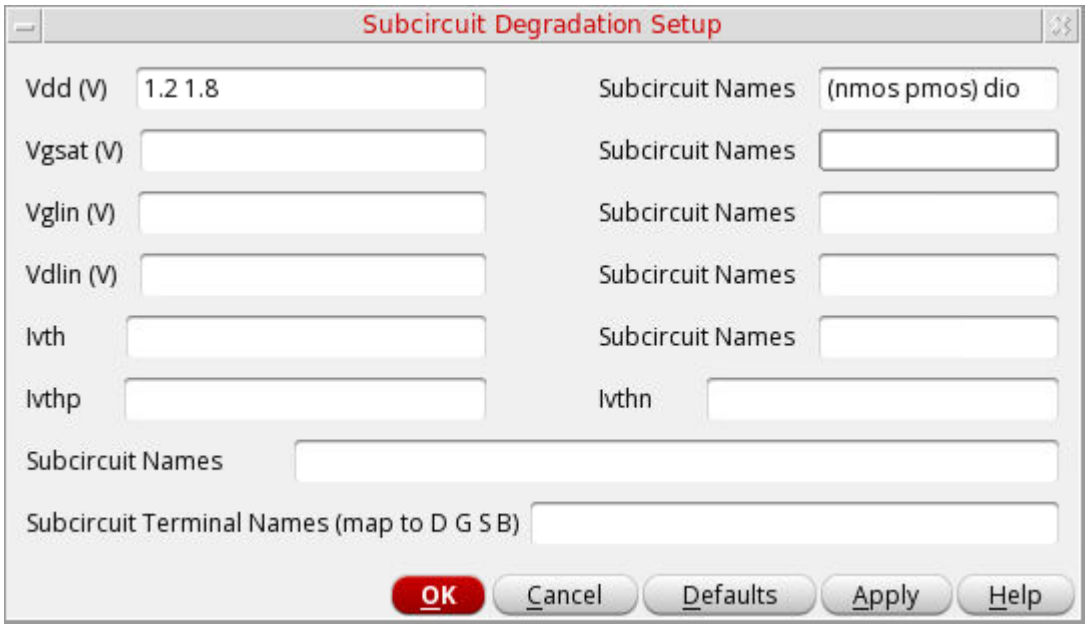
2. In the *Vdd (V)* field, specify the bias voltage for the subcircuit to obtain `gm`, `gds`, `Idlin`, `Idsat`, and `Vth` degradation. This field is mandatory.
3. In the *Vgsat(V)* field, specify the `Vgs` value for `Idsat` measurement.
4. In the *Vglin(V)* field, specify the `Vgs` value for `Idlin/Vt` measurement.
5. In the *Vdlin(V)* field, specify the `Vds` value for `Idlin/Vt` measurement.
6. In the *Ivth* field, specify the constant current value for the specified model.
7. In the corresponding *Subcircuit Names* field, specify the name of the subcircuits. For example, `sub[1]`.

   The following example sets the *Vdd* values for `nmos`, `pmos`, and `dio` to `1.2`, `1.3`, and `1.8V` respectively.



© 2023 Cadence Design Systems, Inc.

The following example sets the *Vdd* values for `nmos`, `pmos`, and `dio` to `1.2`, `1.2`, and `1.8V` respectively.



8. In the *Ivthp* field, specify the constant current value for PMOS.
9. In the *Ivthn* field, specify the constant current value for NMOS.
10. In the *Subcircuit Names* field, specify the list of subcircuits for which degradation needs to be performed.
11. In the Subcircuit Terminal Names field, specify the four-terminal node name in the order `D G S B` for each subcircuit. If a node is not specified, the first four terminals of the subcircuit are taken as `D G S B` by default.
12. Select the *Simulator Path* check box to specify the full path to the simulator that you want to use for aging simulation.

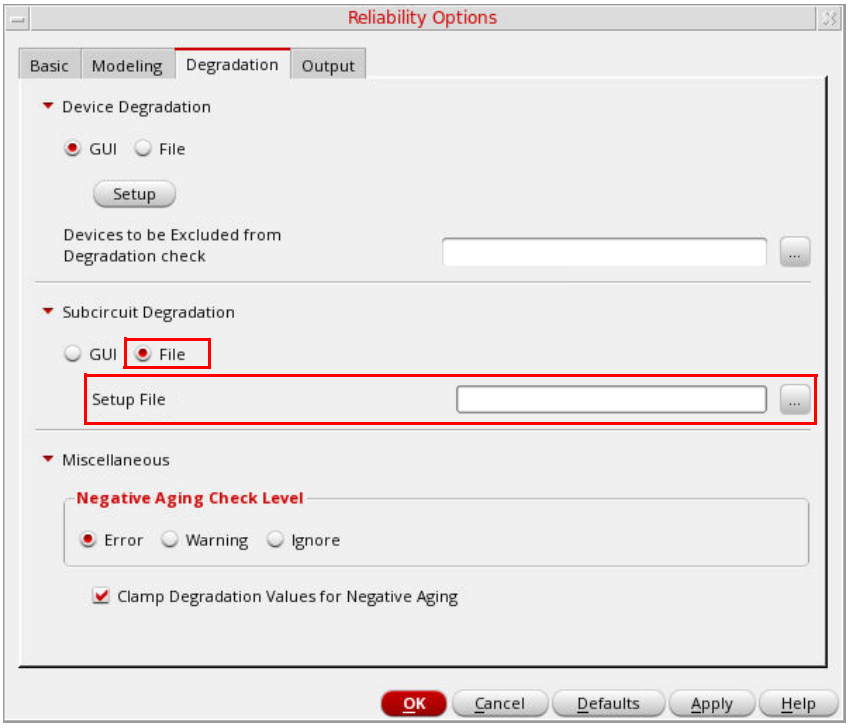
    Note: This field is not available in the Spectre internal mode.
13. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Using Setup File

Follow these steps to specify the device degradation values using a setup file:

1. On the *Degradation* tab, in the *Subcircuit Degradation* section, choose *File*. The *Setup File* field is displayed, as shown in the following figure.



2. In the *Setup File* field, specify the path to the setup file containing the subcircuit degradation values. Alternatively, you can click the ellipsis (…) button to select the setup file.

3. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Creating Outputs for Reliability Analysis

Similar to other analysis, you can create outputs for reliability analysis in the *Outputs Setup* tab. An output can be a signal or an expression.

To create an expression that is based on the output value of another reliability analysis from the same or another simulation history, use the `calcVal` function. Either you can type in the expression in the *Expr* column or use *Expression Builder*. When using the `calcVal` function in an output expression, you must specify different tests for stress and aged test to avoid cyclic dependencies.

Related Topics

calcVal

Adding or Modifying an Output Expression

© 2023 Cadence Design Systems, Inc.

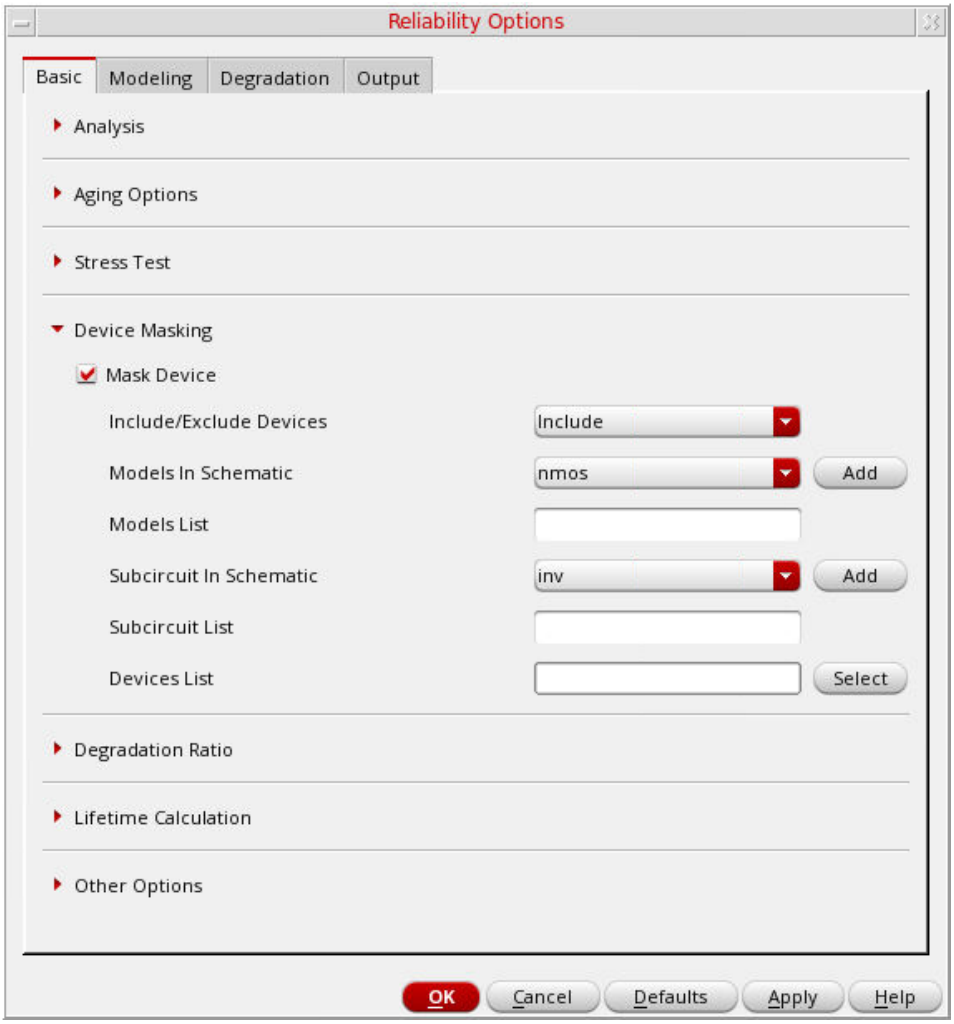
Masking Devices

You can use the options in the *Device Masking* section to specify the devices that you want to include and exclude when calculating device degradation values.

To mask devices:

1. In the *Device Masking* section on the *Basic* tab of the Reliability Options form, select *Mask Devices*.

   The options in the *Device Masking* section are enabled.



2. From the Include/Exclude Devices drop-down list, select one of the following options:

   - *Include*: Device degradation values are calculated for the selected devices.

© 2023 Cadence Design Systems, Inc.

- *Exclude*: Devices degradation values are not calculated for the selected devices.

3. From the *Models In Schematic* drop-down list, select a model, and then click *Add*.

   The selected model is added to the *Models List*. Repeat this step to add another model to the list. Devices in these models are included or excluded when performing reliability analysis, depending on the selection you made in step 2.

4. From the *Subcircuit in Schematic* drop-down list, select a subcircuit, and then click *Add*.

   The selected subcircuit is added the *Subcircuit List*. Repeat this step to add another subcircuit to the list. Devices in these subcircuits are included or excluded when performing reliability analysis, depending on the selection you made in step 2.

5. Click *Select* next to the *Devices List* field to select the instances from the schematic. After selecting the instances, press the *Esc* key on your keyboard.

   The selected instances are added to the *Devices List* field.

6. Click *OK* to close the form.

Copying Reliability Setup

You can reuse an existing reliability setup by creating its copies. You can then make certain modifications and see their effects on the simulation results. This way, you can not only save your time, but also retain the original setup.

Follow these steps to copy a reliability setup:

1. In the *Data View* assistant pane, expand the *Reliability Analyses* node, and then select the reliability setup that you want to copy.

2. Right-click the selected setup and then choose *Create Setup Copy*.

   The settings in the selected setup are copied to a new setup, which is created with the name, *<Name_of_the_existing_reliability_setup>: 1*.

Loading Reliability Settings

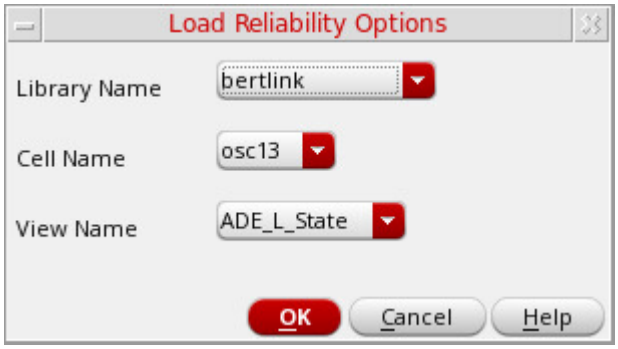
To your existing reliability setup, you can import the reliability settings from another reliability setup that is defined in an ADE L state or a maestro cellview.

Perform the following steps to import the reliability settings:

© 2023 Cadence Design Systems, Inc.

1. In the *Data View* assistant, under *Reliability Analyses* node, select the reliability setup to which you want to import the reliability settings.
2. Right-click the selected setup, and then choose *Load Reliability Settings*.

   The *Load Reliability Options* form appears.



3. From the *Library Name*, *Cell Name*, and *View Name* drop-down lists, select the ADE L state or the maestro cellview where you have defined the reliability setup whose settings you want to import.
4. Click *OK* to import the reliability settings to the selected reliability setup.

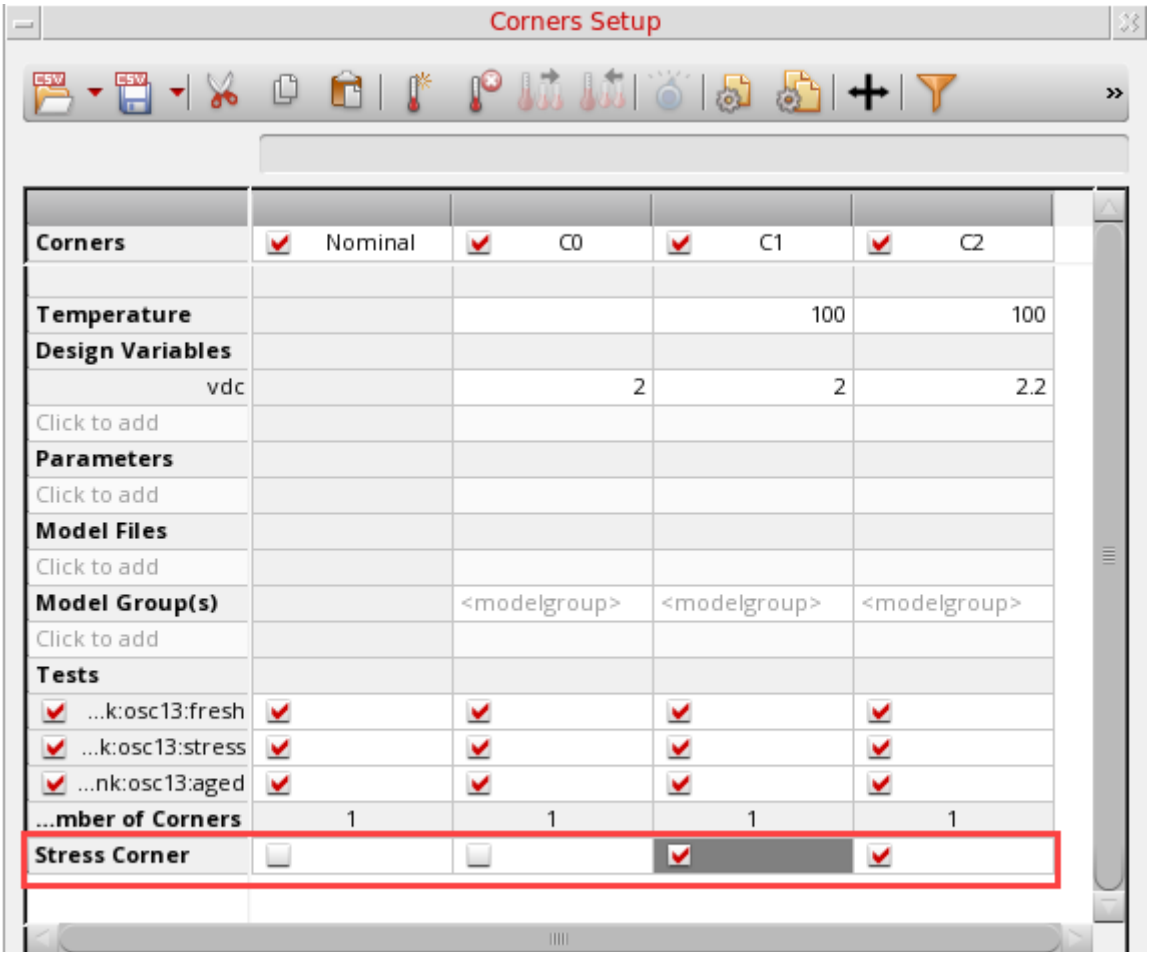
Setting Up Reliability Analysis for Scenarios with Varying Corner Conditions

You can run a reliability simulation with different corner conditions for stress and aged tests. For this, you can create stress corners in the Corners Setup form, and use those corners in the reliability setup for stress corners.

To use this kind of a setup for reliability analysis, you need to change the reliability mode by setting the defaultReliabilityMode environment variable to "`AgingScenarios`". When this setting is enabled, the Corners Setup form shows an additional row named *Stress Corner*

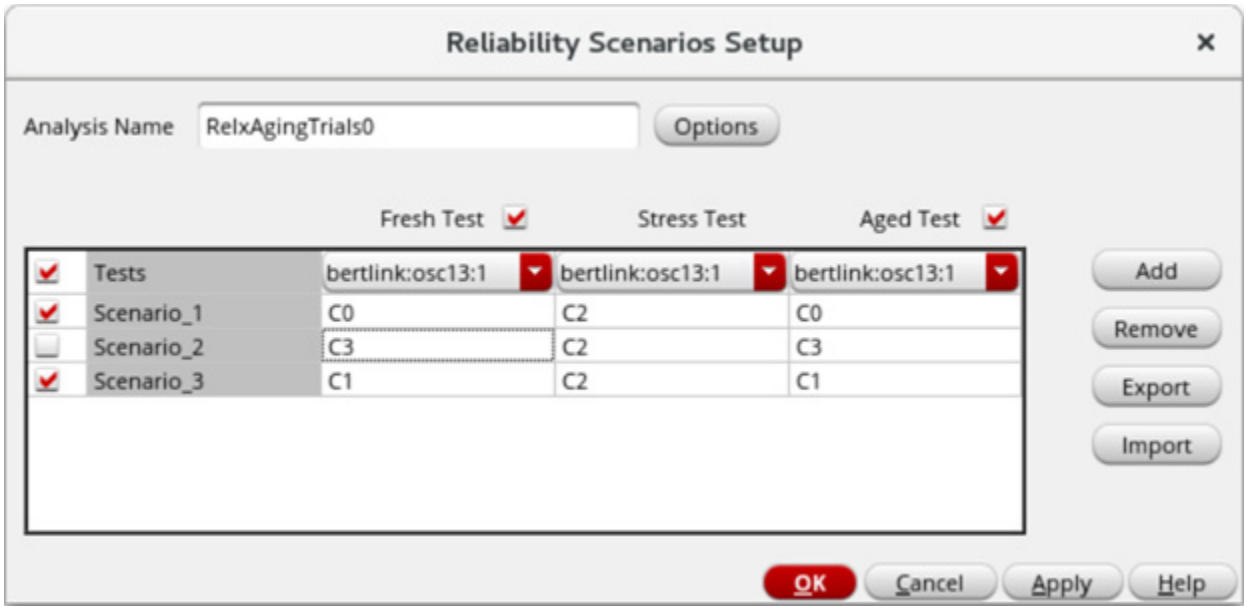
© 2023 Cadence Design Systems, Inc.

at the bottom. You can create special stress conditions for certain corners and select the *Stress Corner* check box to identify those corners as stress corners.



© 2023 Cadence Design Systems, Inc.

In addition, the `"AgingScenarios"` setting also shows additional fields on the *Reliability Scenarios Setup* form to specify the stress corners to be associated with the stress simulation, as shown below.



This kind of setup enables consistent reliability signoff conditions across different setups.

Important Points to Note:

- The default reliability analysis name for scenarios with varying corner conditions is *RelxAgingTrials0*.
- You can select only stress corners for the stress tests. When no stress corner is available in the corners setup, the form displays an error message to indicate the same.

Enabling or Disabling a Reliability Scenario

To enable or disable a reliability scenario in the reliability scenario setup:

1. Select or deselect the check boxes next to the reliability scenarios that you want to enable or disable.
2. Click *Apply* and then click *OK*.

Alternatively:

© 2023 Cadence Design Systems, Inc.

1. Select the reliability scenario row that you want to enable or disable. Hold down the `Ctrl` key to select multiple reliability scenarios.
2. Right-click any of the selected rows and choose one of the following:
   - *Enable*: Selected reliability scenarios are enabled in the setup.
   - *Disable*: Selected reliability scenarios are disabled in the setup.
3. Click *Apply* and then click *OK*.

Only the reliability scenarios that are enabled in the reliability scenario setup are simulated.

© 2023 Cadence Design Systems, Inc.

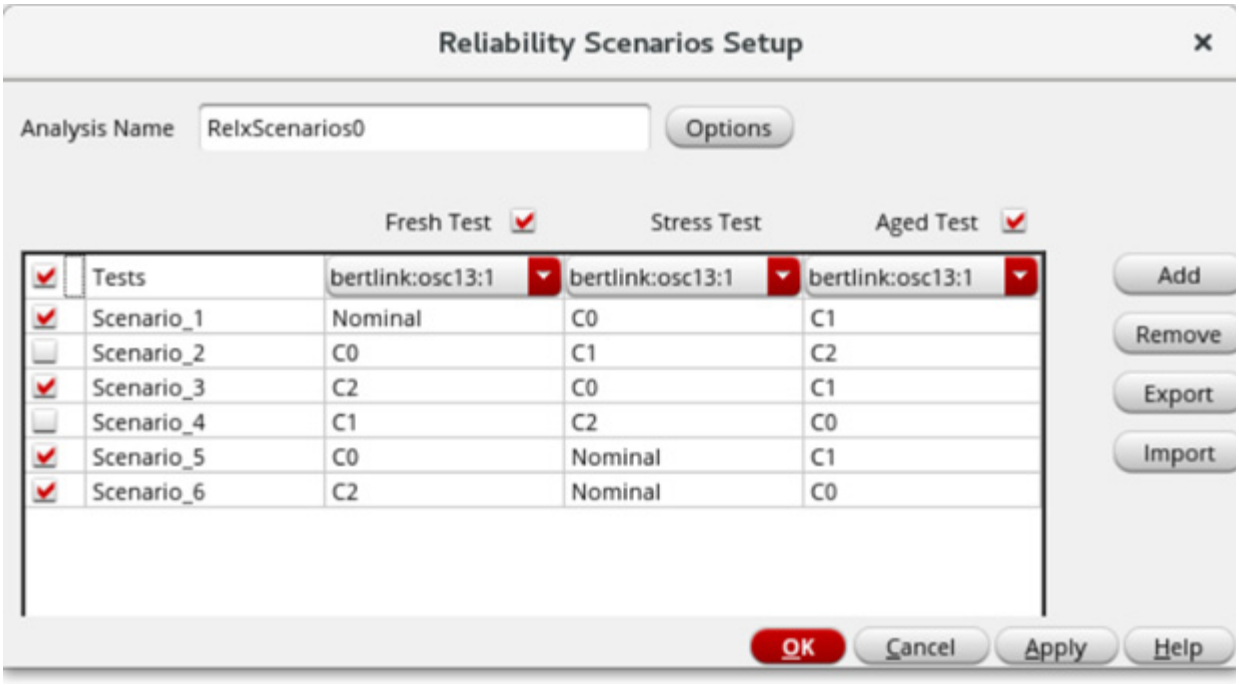
Exporting a Reliability Scenario Setup

You can export a reliability scenario setup to a CSV file. You can later import this CSV file to make a copy of this setup.

To export a reliability scenario setup:

1. In the Data View assistant, click *Reliability Scenarios* to view a list of available reliability scenario setups.
2. Right-click the reliability scenario setup that you want to export and then choose *Open Setup*. Alternatively, double-click the reliability scenario setup.

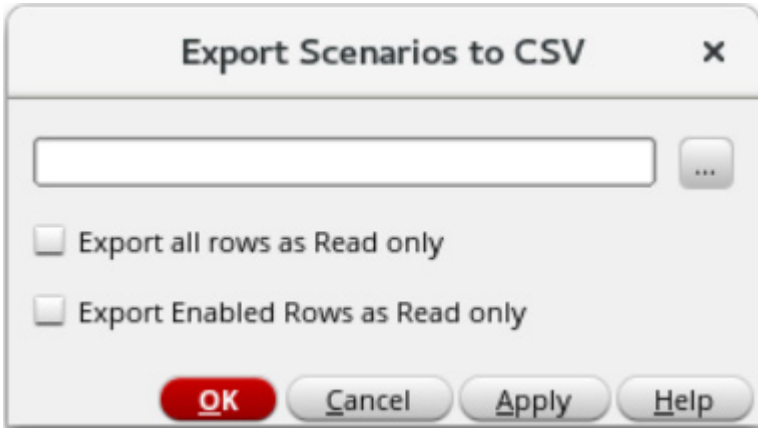
   The Reliability Scenarios Setup form opens.



3. Edit the reliability scenario setup if required. You can add, remove, enable, or disable reliability scenario rows by using the *Add*, *Remove*, *Enable*, or *Disable* commands, respectively.
4. Click *Export* to export the setup to a CSV file.

© 2023 Cadence Design Systems, Inc.

The Export Scenarios to CSV form opens.



5. Specify the path to the CSV file to which you want to export the setup.

   Alternatively, you can click the *Ellipsis* (...) icon to select the CSV file.

6. Select one of the following options:

   - *Export all rows as Read only*: To export all reliability scenario rows in the setup as read-only.
   - *Export Enabled Rows as Read only*: To export only enabled reliability scenario rows as read-only.

7. Click *OK* to export the setup.

   The reliability scenario setup is exported to the specified CSV file.

Related Topics

Importing a Reliability Scenario Setup from the CSV File

Contents of the Exported CSV File

Contents of the Exported CSV File

The contents of the CSV file are as follows:

```
t,Scenario_1,Nominal,C0,C1,r
nil,Scenario_2,C0,C1,C2,e
t,Scenario_3,C2,C0,C1,r
nil,Scenario_4,C1,C2,C0,e
t,Scenario_5,C0,Nominal,C1,r
```


© 2023 Cadence Design Systems, Inc.

```
t,Scenario_6,C2,Nominal,C0,r
```

The CSV file contains entries for each reliability scenarios in separate rows. Each row has six comma-separated values. The following table explains the attributes these values represent.

Attribute No	Description
1	Indicates whether the reliability scenario row is enabled in the setup being exported. Valid values are: ■ `t`: The corresponding reliability scenario row is enabled in the setup. ■ `nil`: The corresponding reliability scenario row is disabled in the setup
2	Name of the reliability scenario. For example: `Scenario_1` and `Scenario_2`.
3	Name of the fresh corner in the corresponding reliability scenario. For example: `Nominal` and `C0`.
4	Name of the stress corner in the corresponding reliability scenario. For example: `Nominal` and `C0`.
5	Name of the aged corner in the corresponding reliability scenario. For example: `Nominal` and `C0`.

© 2023 Cadence Design Systems, Inc.

Attribute No	Description
6	Indicates whether to export or import the corresponding reliability scenario row in read-only mode. Valid values are: ■ `r`: The reliability scenario row is exported or imported in read-only mode. ■ `e`: The reliability scenario row is exported or imported in edit mode. If you selected the *Export all rows as Read only* check box while exporting the setup, all reliability scenario rows have the value `r` for this attribute. If you selected the *Export Enabled Rows as Read only* check box, only the enabled scenario rows will have the value `r` for this attribute, whereas the disabled rows will have the value `e` for this attribute. The reliability scenario rows with `r` values in the CSV file, when imported, are grayed out in the reliability scenario setup. These rows cannot be edited in the Reliability Scenarios Setup form.

Related Topics

Exporting a Reliability Scenario Setup

Importing a Reliability Scenario Setup from the CSV File

© 2023 Cadence Design Systems, Inc.

Importing a Reliability Scenario Setup from the CSV File

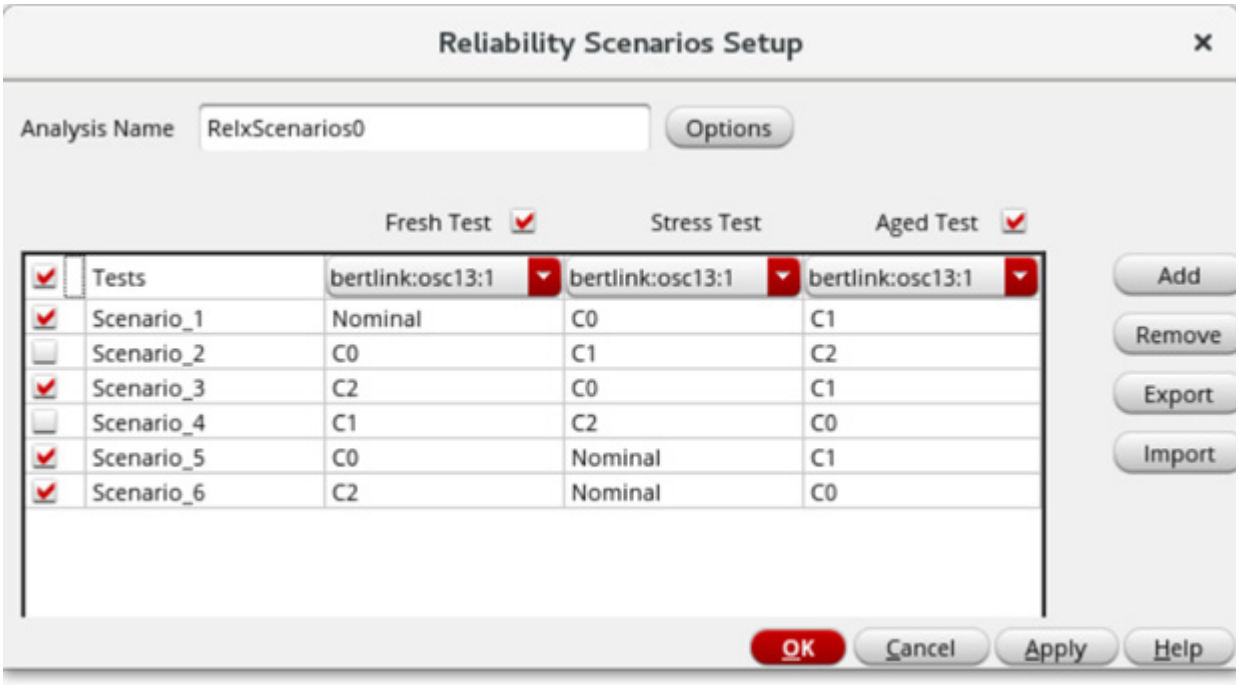
You can either create a new reliability scenario setup or edit an existing reliability scenario setup by importing the setup saved in a CSV file.

To import a reliability scenario setup:

1. In the Data View assistant, click *Reliability Scenarios* to view a list of available reliability scenario setups.
2. Right-click the reliability scenario setup that you want to edit and then choose *Open Setup*. Alternatively, double-click the reliability scenario setup.

   If there are no existing reliability scenarios setups, click *Click to add reliability scenarios setup* to create a new setup.

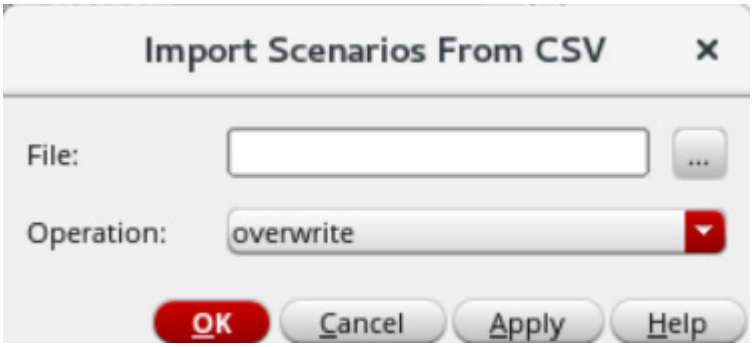
   The Reliability Scenarios Setup form opens.



3. Click *Import* to import the setup from a CSV file.

© 2023 Cadence Design Systems, Inc.

The Import Scenarios From CSV form opens.



4. Specify the path to the CSV file from which you want to import the setup.

   Alternatively, you can click the *Ellipsis* (...) icon to select the CSV file.

5. From the *Operations* drop-down list, select one of the following options:

   - *overwrite*: All existing scenarios are deleted from the Reliability Scenarios Setup form, and scenarios from the CSV file are added to the form. This is the default value.
   - *retain*: If a scenario name being imported from the CSV file already exists in the Reliability Scenarios Setup form, its import is discarded from the CSV file, and the existing scenario in the form is retained as is.

     When you select this option:

     - Scenarios that exist only in the Reliability Scenarios Setup form, but not in the CSV file being imported, are kept as is.
     - Scenarios that exist only in the CSV file, but not in the form, are appended.
   - *merge*: If a scenario name being imported from the CSV file already exists in the Reliability Scenarios Setup form, the existing scenario is deleted from the form, and its details are copied from the CSV to the form.

     When you select this option:

     - Scenarios that exist only in the form, but not in CSV file, are kept as is.
     - Scenarios that exist only in the CSV file, but not in the form, are appended in the form.

6. Click *OK* to import the setup.

© 2023 Cadence Design Systems, Inc.

Related Topics

Exporting a Reliability Scenario Setup

Contents of the Exported CSV File

Running the Reliability Simulation

Before you run reliability simulation, ensure that the following prerequisites are met:

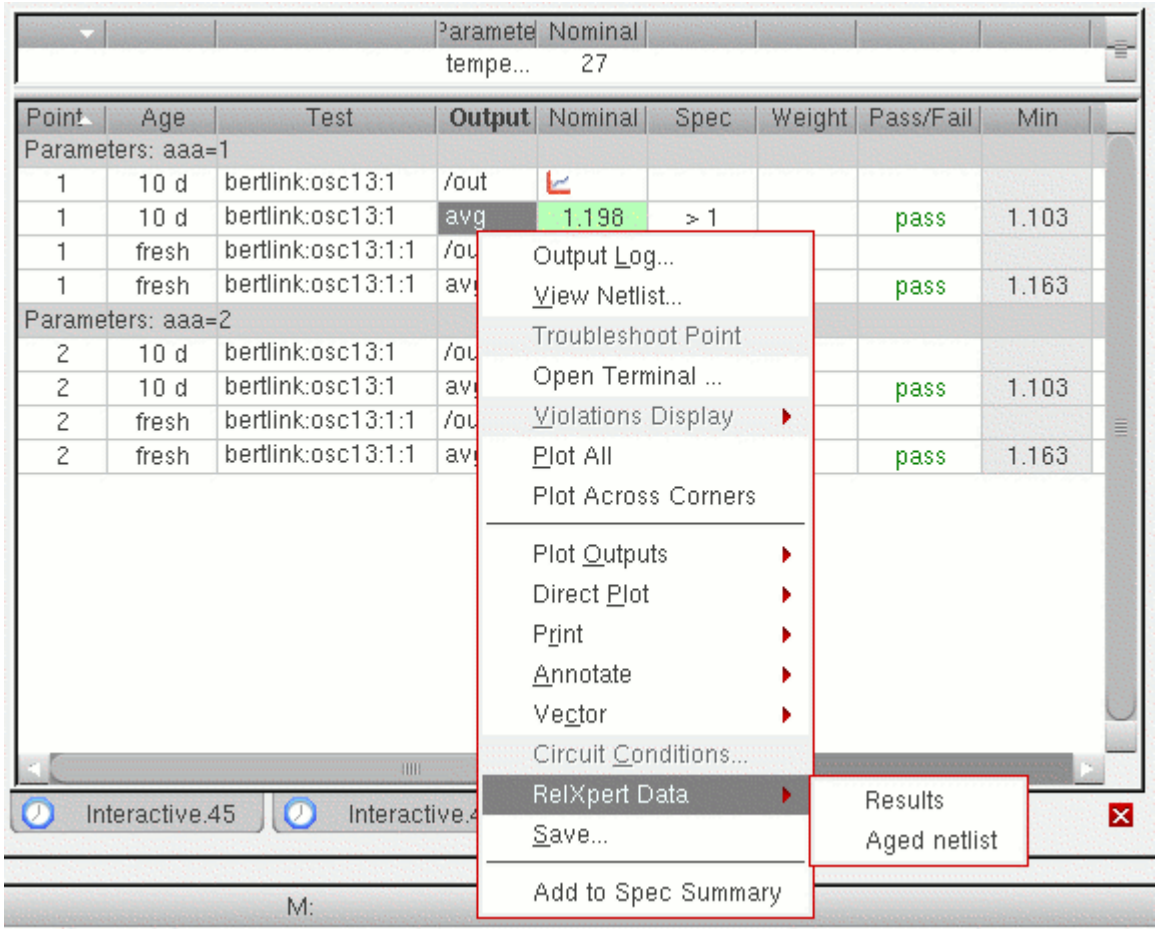
- A Spectre or APS transient analysis is set up and enabled for the stress test. You can set these by using the ADE Assembler Test Editor.
- The *Enable Reliability* check box is selected in the RelXpert Options form.

To run the reliability simulation:

➡ Click the *Run Simulation* button .

© 2023 Cadence Design Systems, Inc.

The fresh, stress, and aging simulations are run for the specified tests. To view the RelXpert simulation result data, right-click the *Results* tab and choose an appropriate option from the menu shown in the following figure.



For more details on how to view RelXpert simulation result data, refer to Working with Reliability Data.

Important Points to Note

- You can also run multiple RelXpert setups simultaneously. When you run multiple setups, an additional column, *Relx*, is visible before the *Age* Column on the *Results* tab. The *Relx* column specifies the name of the corresponding RelXpert setup.

  The number of results sets generated from the reliability analysis is determined by the total number of data points on which the simulations specified in the reliability analysis setup is run. Consider a reliability analysis with the following data that needs to be run for fresh and aged tests (two age points):

© 2023 Cadence Design Systems, Inc.

- 1 variable with 2 sweep points
- 3 corners (2 corners + 1 nominal corner)

In this example, each simulation (one fresh and two aging) is run 6 times. Therefore, the total number of data points for which the reliability analysis is run is 18.

For more information, see the Working with Reliability Data.

- If we have enabled Reliability Analysis in ADE Assembler, simulations are run only for those tests that are part of any of the Reliability Analysis setup. Therefore, if there are any tests that are enabled in ADE Assembler user interface, but are not used as fresh, stress, or aged tests in the Reliability Analysis setup, those tests are ignored. In such a case, the following warning message is displayed in CIW.

  ```
  WARNING (ADEXL-7005): The following tests are not being used in any of the
  enabled Reliability Analysis setups and hence it will be disabled for the
  current run
  ```

- RelXpert Reliability Simulator always runs in batch mode. Therefore, ADE Assembler too automatically runs in batch mode when you enable RelXpert. The following message is displayed in CIW indicating the change in mode:

  ```
  WARNING: The Spectre run mode needs to be 'batch' when running RelXpert.
  Automatically setting the run mode to 'batch'.
  ```

- Consider the case of an ADE Assembler setup in which reliability analysis is not set up. If you modify a testbench in the ADE Assembler Test Editor, and enable reliability analysis, after you close the Test Editor, a default reliability setup named `Default_Relx` is created in the Data View pane.
- The default ADE Assembler reliability setup will be created only for the test for which reliability setup is enabled in the Test Editor. Once a default setup has been created for a test, if you create or modify another test in the Test Editor, no default setup will be created for that test.

© 2023 Cadence Design Systems, Inc.

Difference Between Reliability Mode And Aging Scenarios

The following table lists the differences between reliability mode and aging scenarios.

Reliability Mode	Aging Scenarios
■ This is the default reliability mode. The default value of the environment variable `maestro.gui defaultReliabilityMode` is `Reliability`. ■ Corners and variables defined in the active setup are common to all fresh, stress, and aged tests. ■ Cannot run different corners for fresh, stress, and aged tests. ■ Supports *Monte Carlo Sampling* run mode. ■ Local sweep is not supported. If you want to sweep a temperature or a variable, you must define them in the *Global Variables* section or in the Corners Setup form.	■ You need to set the environment variable to `maestro.gui defaultReliabilityMode` to `AgingScenarios` to enable aging scenarios. ■ A *Stress Corner* must be defined in the Corners Setup form before setting up a reliability scenario. ■ Can run different corners for fresh, stress, and aged tests. ■ Does not support *Monte Carlo Sampling* run mode.

© 2023 Cadence Design Systems, Inc.

Working with Reliability Data

With reliability data, you can do the following:

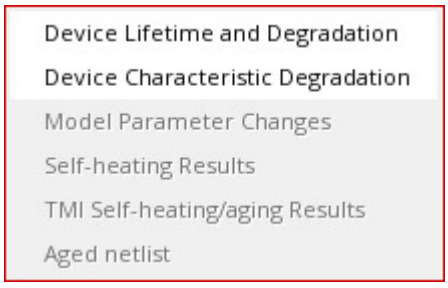
- Viewing Simulation Results
- Viewing Aged Netlist
- Viewing Output Logs for Reliability Simulations
- Plotting Results
- Plotting Results Across Corners
- Printing or Plotting Stress Results for Gradual Aging Run
- Annotating Simulation Results to Schematic View

Viewing Simulation Results

To view simulation results, do the following:

1. On the *Results* tab, select the required test.
2. Right-click the selected test and choose *Reliability Data*.
3. Depending on the MMSIM version used to run the simulation, the following submenu is displayed:
   - If you have used MMSIM 11.1 to run simulation, the following submenu is displayed.

   - If you used MMSIM 12.1 to run simulation, the following submenu is displayed.

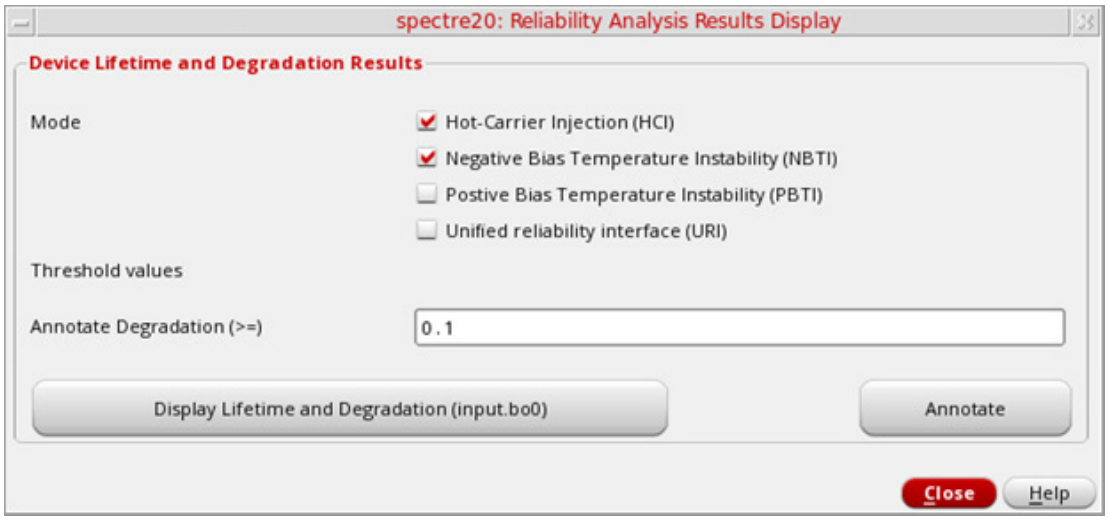
     Note: Depending on the type of test: fresh, stress or aged; the simulator mode:

© 2023 Cadence Design Systems, Inc.

native or RelXpert); and the reliability setup, only those commands are available in this submenu for which the corresponding results are available.

4. Choose the result that you want to view.

   The *Reliability Analysis Results Display* form appears. The options displayed on this form depend upon the type of results that you chose. The following figure shows an example of the form when *Reliability Data > Device Lifetime and Degradation* is selected.



5. In the *Reliability Analysis Results Display* form, do the following:

Click...	To...
Display Lifetime and Degradation	Display the lifetime and degradation values for all degraded devices. See the Displaying Lifetime and Degradation Results section for more information.
Display Device Characteristic Degradation	Display the device degradation for the specified aging time period. See the Displaying Device Characteristic Degradation Results section for more information.
Display Model Parameter Changes	Display fresh and aged parameter information. See the Displaying Model Parameter Change Results section for more information.

© 2023 Cadence Design Systems, Inc.

Display Self-heating	Display the temperature rise for each device. See the Displaying Self-Heating Results section for more information.

© 2023 Cadence Design Systems, Inc.

Displaying Lifetime and Degradation Results

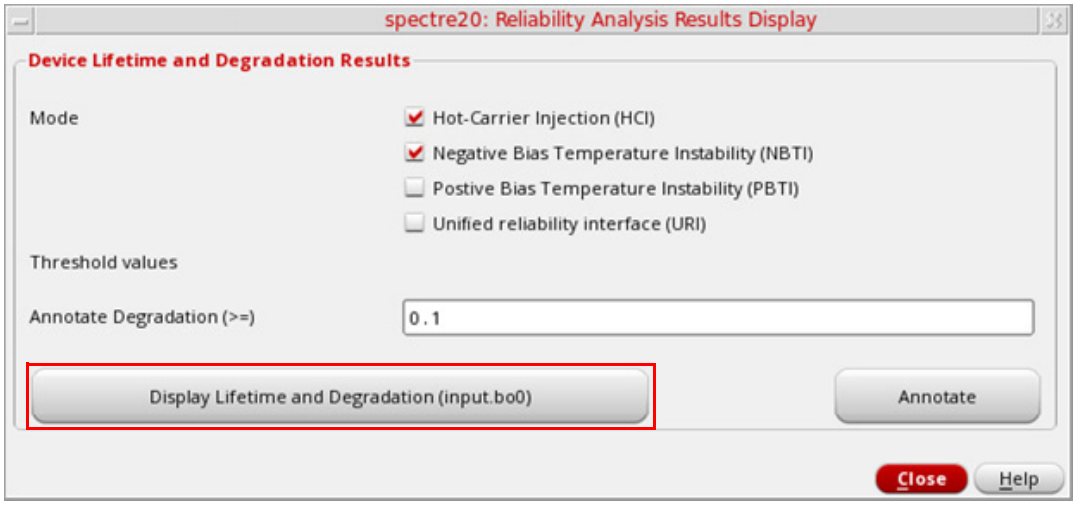
The lifetime and degradation results display a list of devices whose electrical properties have degraded by at least the value specified in the *Degradation Value* field on the *Basic* tab of the *Reliability Options* form.

If you set the *Degradation Value* to `0.1`, the lifetime and degradation results display a list of devices whose electrical properties have degraded by `10%` or more.

Follow these steps to view the lifetime and degradation results of the devices:

1. On the *Results* tab, right-click a result and select *Reliability Data – Device Lifetime and Degradation*.

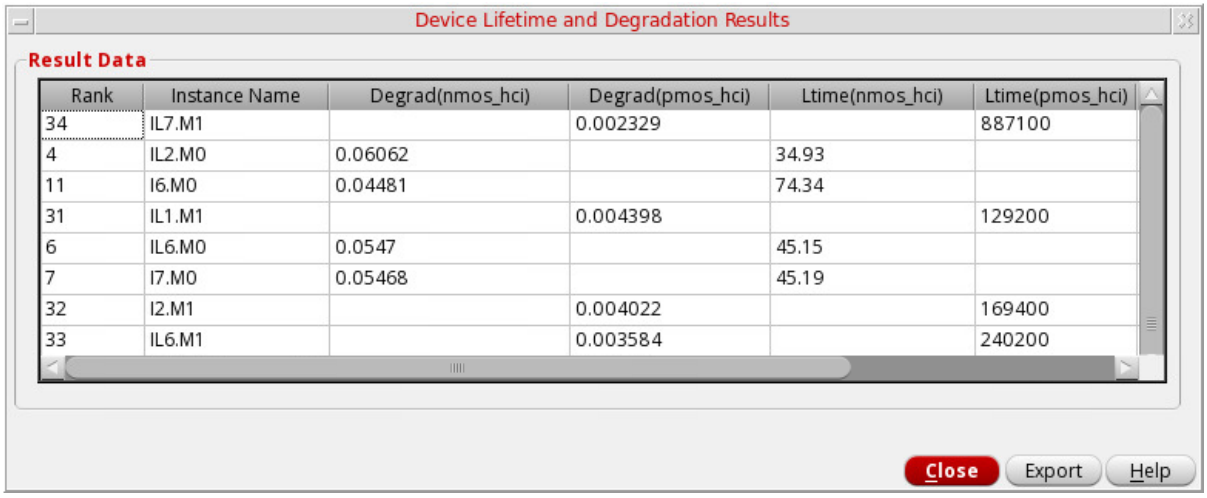
   The *Reliability Analysis Results Display* form appears.



2. Click *Display Lifetime and Degradation* to view the results.

© 2023 Cadence Design Systems, Inc.

The *Device Lifetime and Degradation Results* form appears, displaying the results in a table.



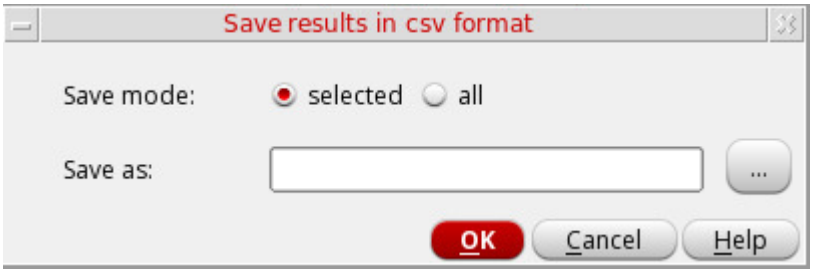
You can click the column headers to sort the values in ascending or descending order.

Exporting the Lifetime and Degradation Results

Follow these steps to export the lifetime and degradation results of the devices:

1. In the *Device Lifetime and Degradation Results* form, select the cells in the table that you want to export to the CSV file, and then click *Export*.

   The *Save results in csv format* form appears.



2. In the *Save mode* field, select one of the following options:
   - *selected*: Exports data for the selected cells
   - *all*: Exports data for all the cells
3. In the *Save as* field, type the name of the CSV file to which you want to export the result data.

© 2023 Cadence Design Systems, Inc.

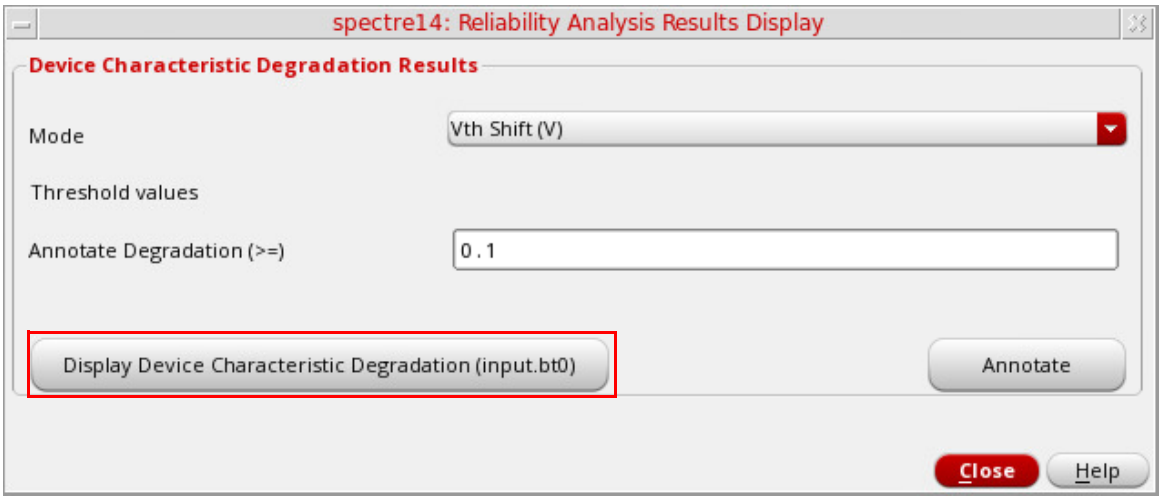
4. Click *OK*.

Displaying Device Characteristic Degradation Results

Follow these steps to view the characteristic degradation results of the devices:

1. On the *Results* tab, right-click a result, and choose *Reliability Data – Device Characteristic Degradation*.

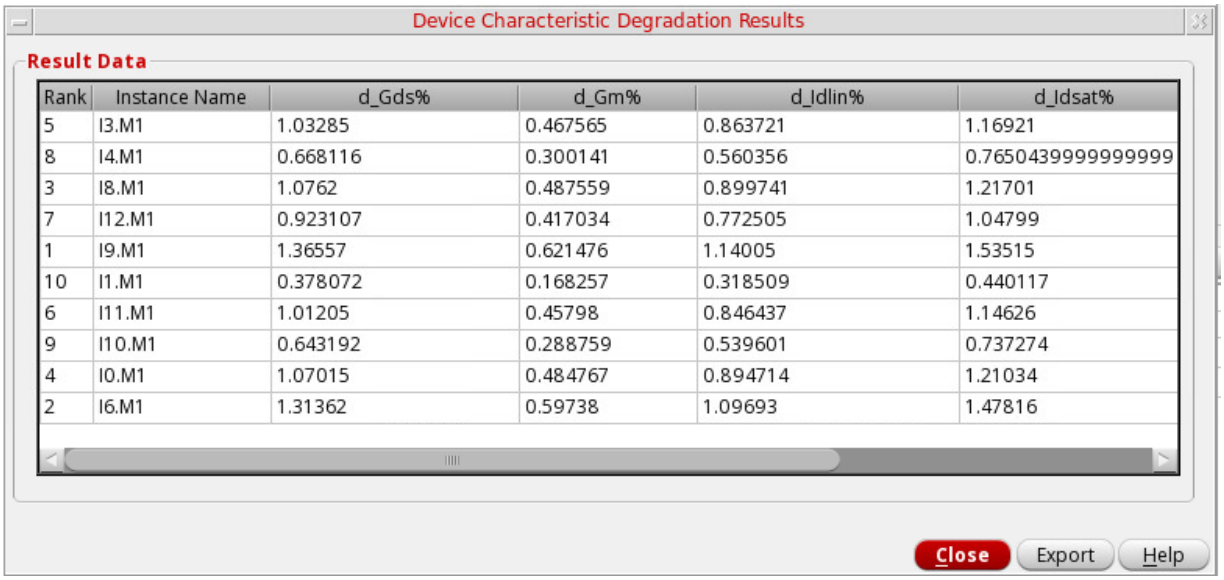
   The *Reliability Analysis Results Display* form appears.



2. Click *Display Device Characteristic Degradation (input.bt0)* to view the results.

© 2023 Cadence Design Systems, Inc.

The *Device Characteristic Degradation Results* form appears, displaying the results in a table.



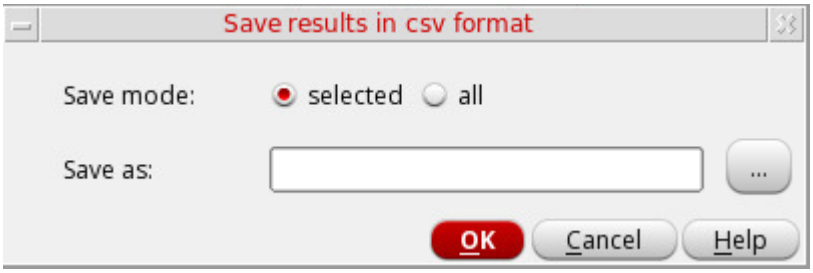
You can click the column headers to sort the values in ascending or descending order.

Exporting the Device Characteristic Degradation Results

Follow these steps to export the device characteristic degradation results:

1. In the *Device Characteristic Degradation Results* form, select the cells in the table that you want to export to the CSV file, and then click *Export*.

   The *Save results in csv format* form appears.



2. In the *Save mode* field, select one of the following options:
   - *selected*: Exports data for the selected cells
   - *all*: Exports data for all the cells

© 2023 Cadence Design Systems, Inc.

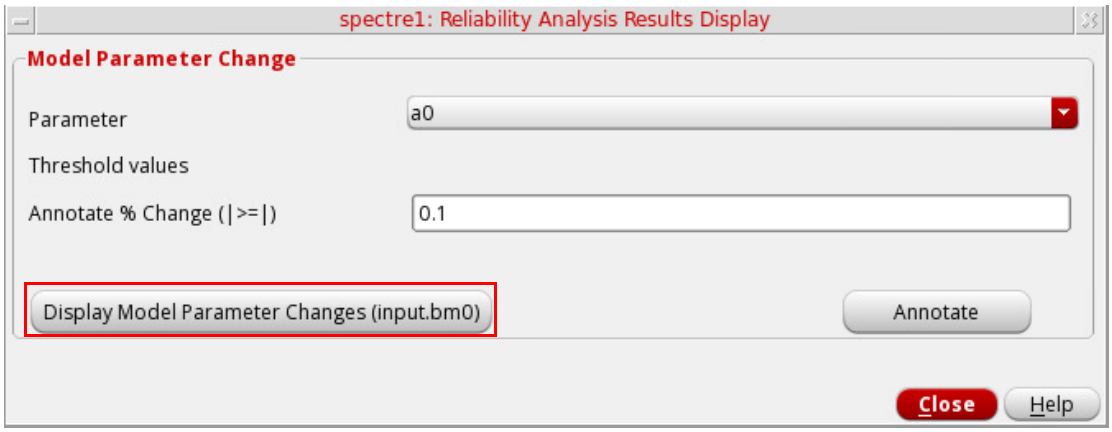
3. In the *Save as* field, type the name of the CSV file to which you want to export the result data.
4. Click *OK*.

Displaying Model Parameter Change Results

Follow these steps to view the model parameter change results:

1. On the *Results* tab, right-click a result, and choose *Reliability Data* > *Model Parameter Changes*.

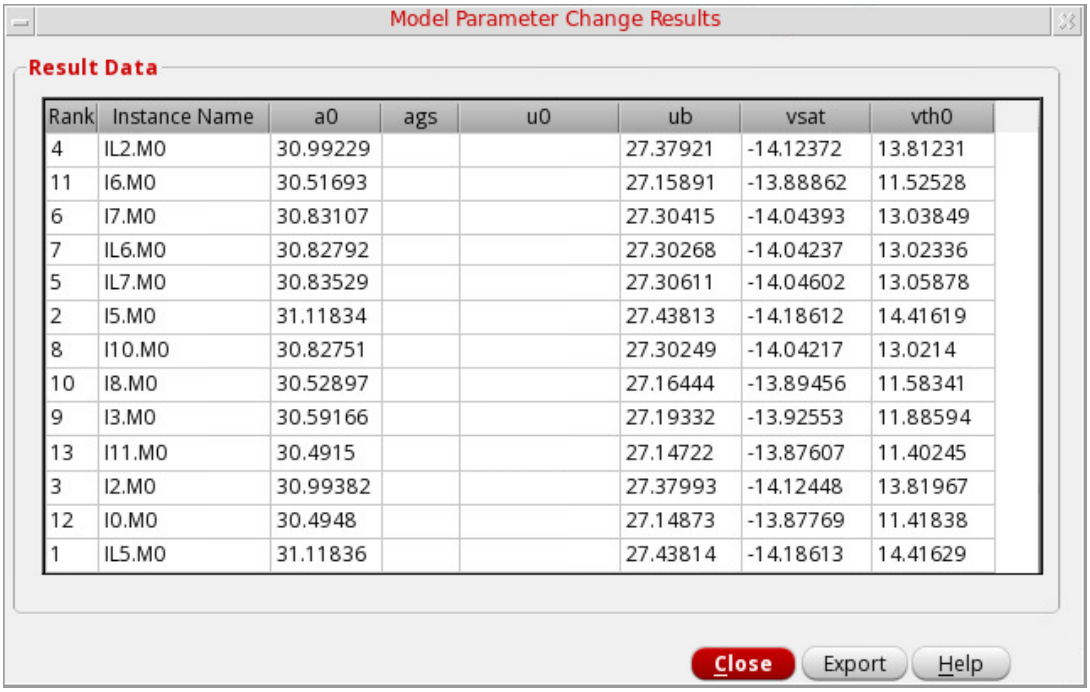
   The *Reliability Analysis Results Display* form appears.



2. Click *Display Model Parameter Changes (input.bm0)* to view the results.

© 2023 Cadence Design Systems, Inc.

The *Model Parameter Change Results* form appears, displaying the results in a table.



Rank	Instance Name	a0	ags	u0	ub	vsat	vth0
4	IL2.M0	30.99229			27.37921	-14.12372	13.81231
11	I6.M0	30.51693			27.15891	-13.88862	11.52528
6	I7.M0	30.83107			27.30415	-14.04393	13.03849
7	IL6.M0	30.82792			27.30268	-14.04237	13.02336
5	IL7.M0	30.83529			27.30611	-14.04602	13.05878
2	I5.M0	31.11834			27.43813	-14.18612	14.41619
8	I10.M0	30.82751			27.30249	-14.04217	13.0214
10	I8.M0	30.52897			27.16444	-13.89456	11.58341
9	I3.M0	30.59166			27.19332	-13.92553	11.88594
13	I11.M0	30.4915			27.14722	-13.87607	11.40245
3	I2.M0	30.99382			27.37993	-14.12448	13.81967
12	I0.M0	30.4948			27.14873	-13.87769	11.41838
1	IL5.M0	31.11836			27.43814	-14.18613	14.41629

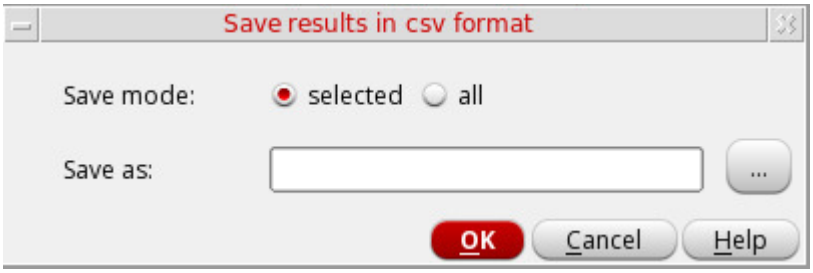
You can click the column headers to sort the values in ascending or descending order.

Exporting the Model Parameter Change Results

Follow these steps to export the model parameter change results:

1. In the *Model Parameter Change Results* form, select the cells in the table that you want to export to the CSV file, and then click *Export*.

   The *Save results in csv format* form appears.



2. In the *Save mode* field, select one of the following options:

   - *selected*: Exports data for the selected cells

© 2023 Cadence Design Systems, Inc.

- *all*: Exports data for all the cells

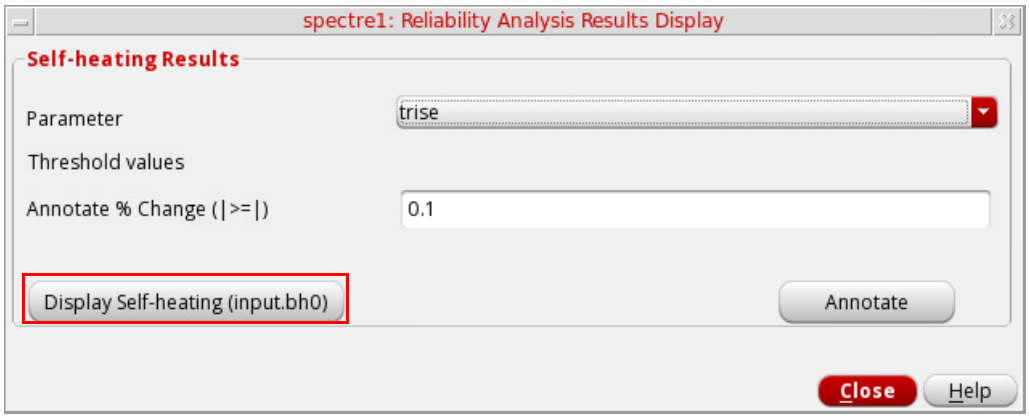
3. In the *Save as* field, type the name of the CSV file to which you want to export the result data.
4. Click *OK*.

Displaying Self-Heating Results

Follow these steps to view the self-heating results:

1. On the *Results* tab, right-click a result, and choose *Reliability Data* > *Self-Heating Results*.

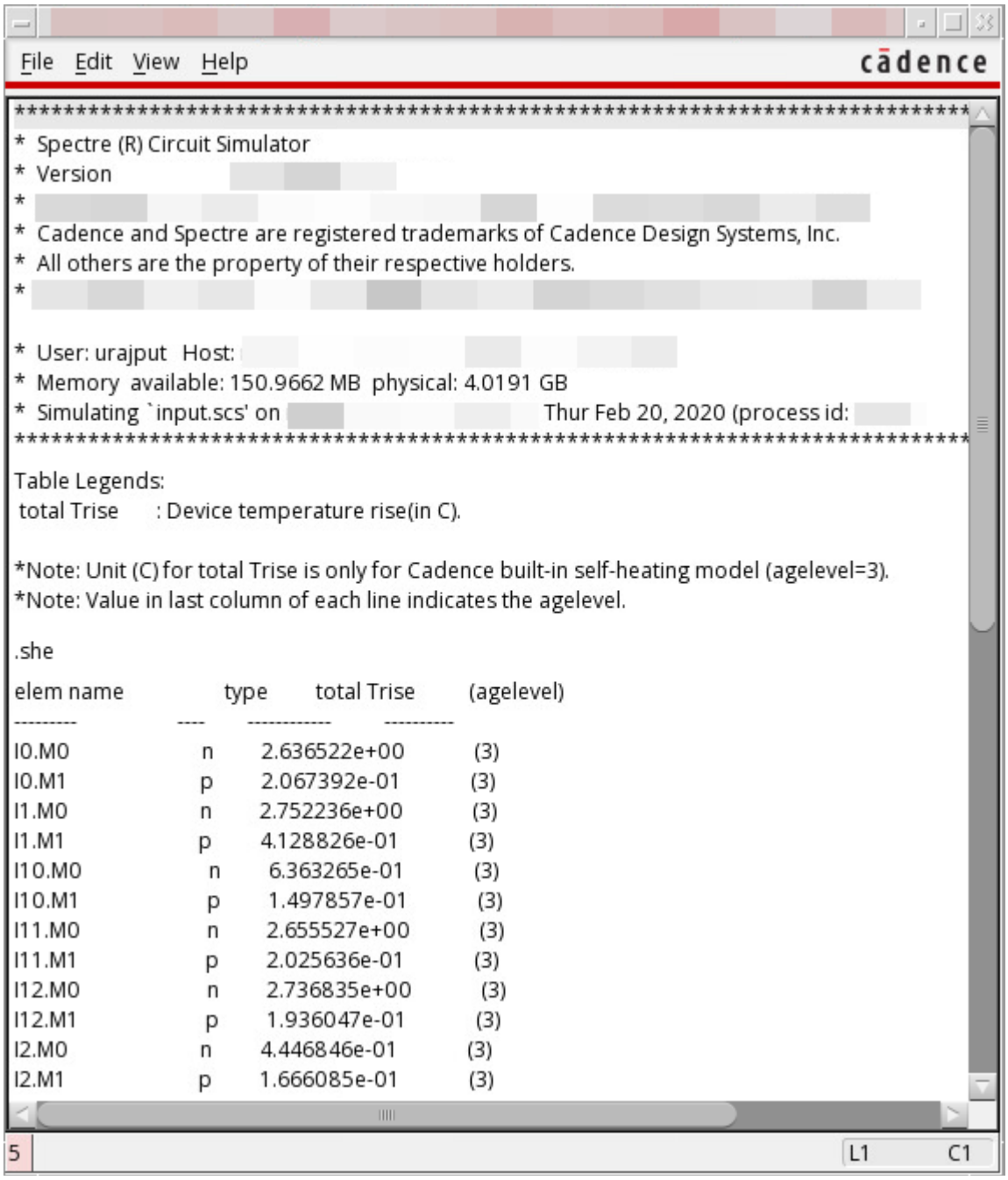
   The *Reliability Analysis Results Display* form appears.



2. Click *Display Self-heating (input.bh0)* to view the results.

© 2023 Cadence Design Systems, Inc.

The *self-heating results are displayed in a new window.*



© 2023 Cadence Design Systems, Inc.

Viewing Aged Netlist

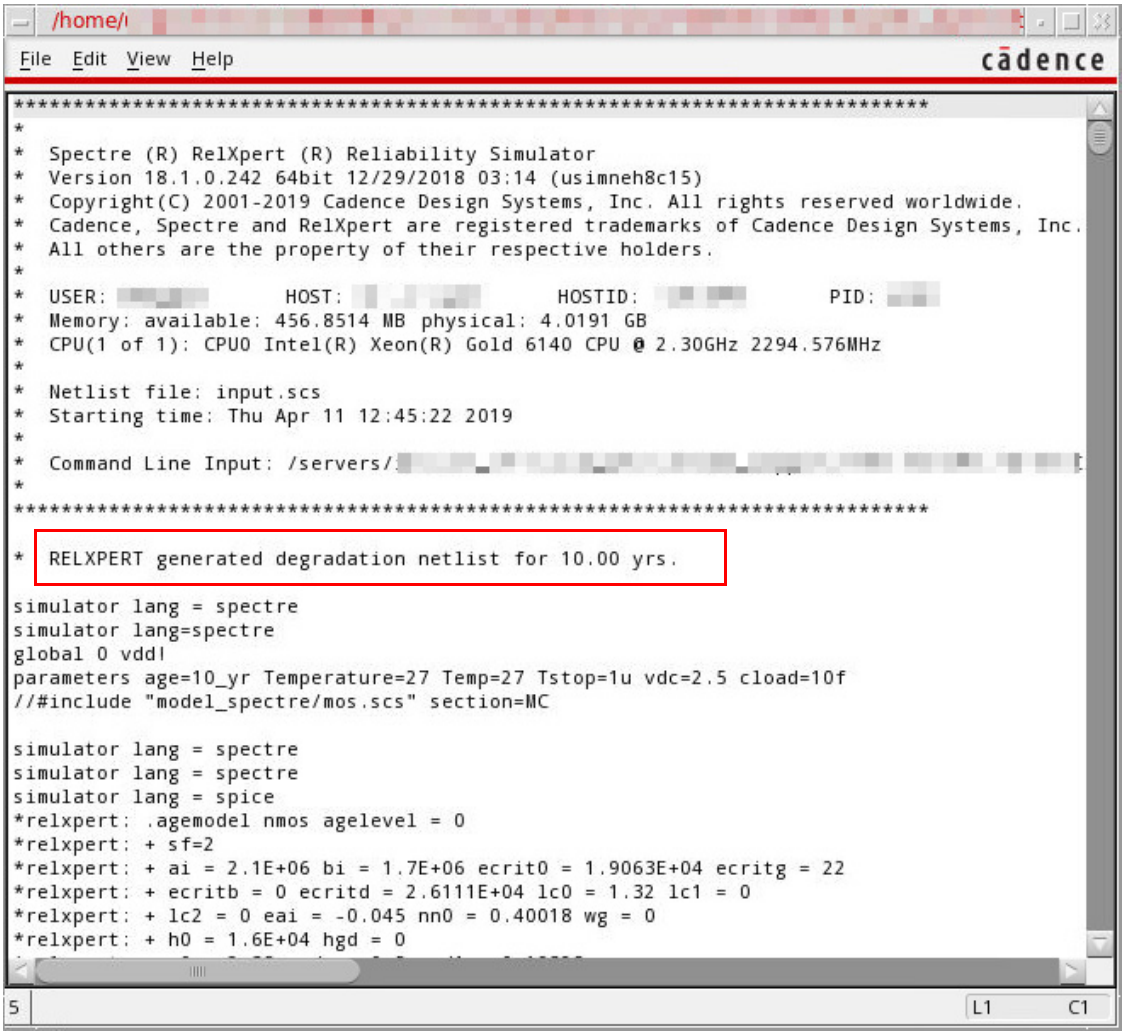
To view the aged netlist for a test:

1. In the Virtuoso ADE Assembler window on the *Results* tab, select the result of an aged test.
2. Right-click the selected test and choose *Reliability Data – Aged netlist*.

   The *Aged netlist* is available only in RelXpert mode. You can switch to RelXpert mode by setting the following environment variable:

   ```
   envSetVal("maestro.gui" "showNativeRelxpertSwitch" 'boolean t)
   ```

   The aged netlist generated by the reliability simulator is displayed in a new window. A sample is shown below.



© 2023 Cadence Design Systems, Inc.

Viewing Output Logs for Reliability Simulations

Perform the following steps to view the output logs for a reliability simulation result:

1. In the *Results* window, select the test for which you want to view the output log.
2. Right-click the selected test and then choose *Output Log*.

   The output log is displayed in a new window.

Plotting Results

You can plot the fresh, stress, or aged results in the Virtuoso Visualization and Analysis XL window.

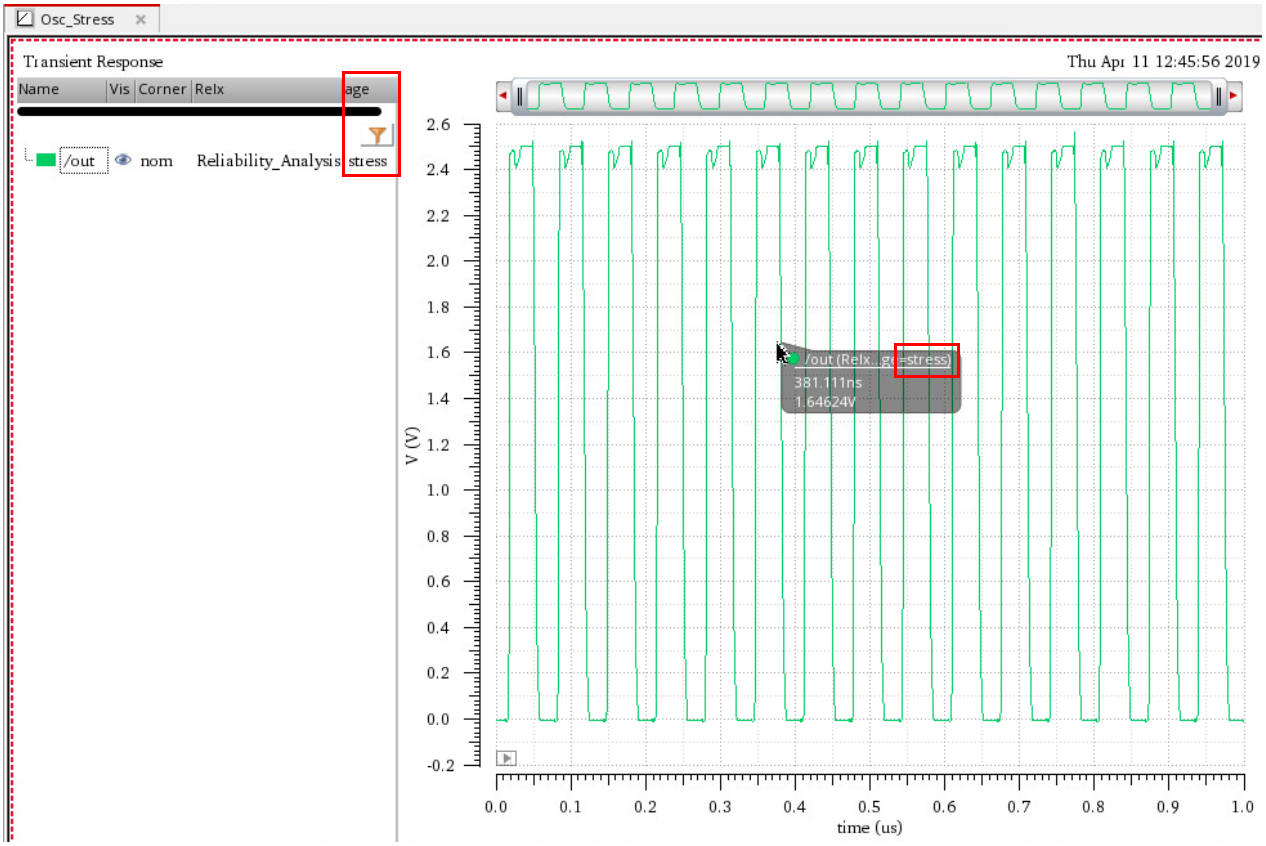
To plot the results:

➡ On the *Results* tab, right-click the fresh, stress, or aged result and choose *Plot*.

The results are plotted in the Virtuoso Visualization and Analysis XL window. In the plot of fresh test, the graph legend shows age value as `0` and for the age test, the legend shows the age in years. However, for the stress result, the trace legend and trace name

© 2023 Cadence Design Systems, Inc.

for the stress plot shows `age=stress` to indicate that the results are plotted for stress results, as shown below.



Plotting Results Across Corners

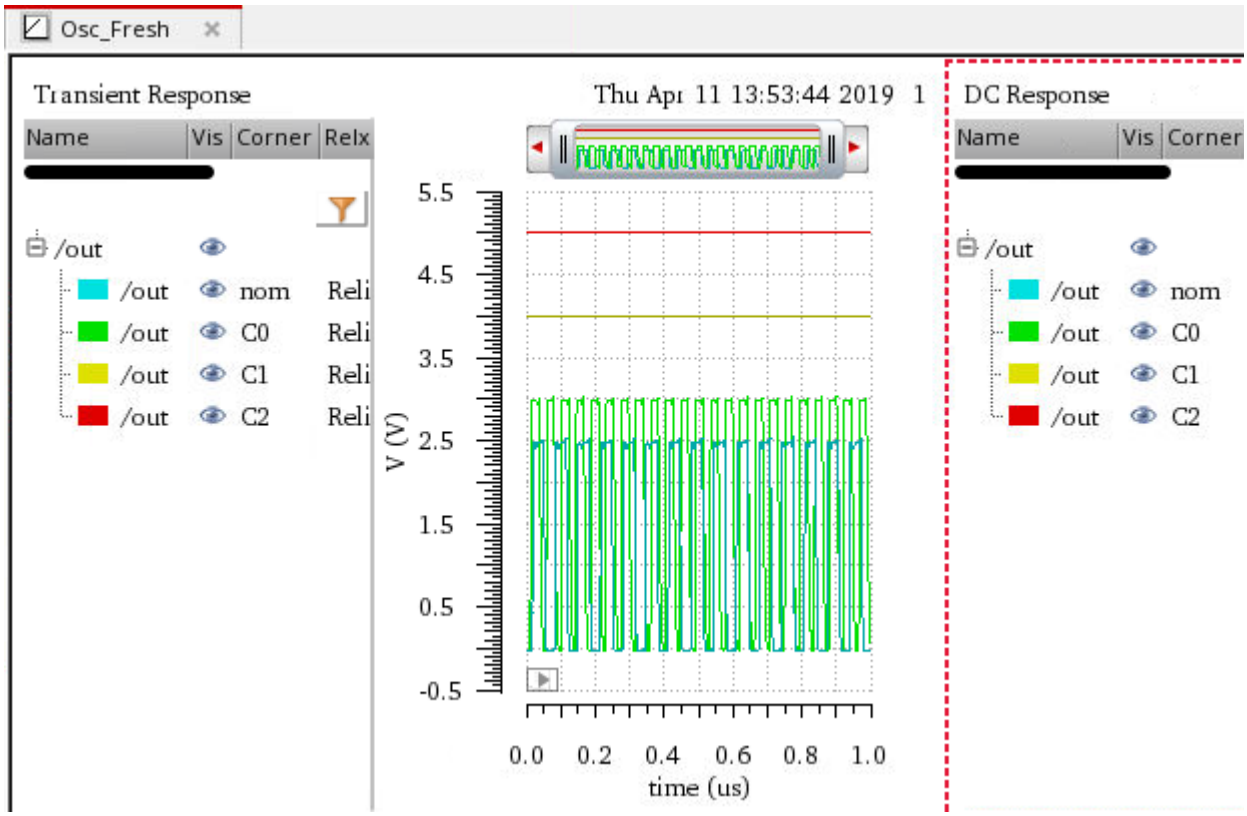
You can plot the fresh and aged results across corners.

To plot results across corners:

➡ On the *Results* tab, right-click the fresh, stress, or aged result and choose *Plot Across Corners*.

© 2023 Cadence Design Systems, Inc.

The results are plotted in the Virtuoso Visualization and Analysis XL window, as shown below.

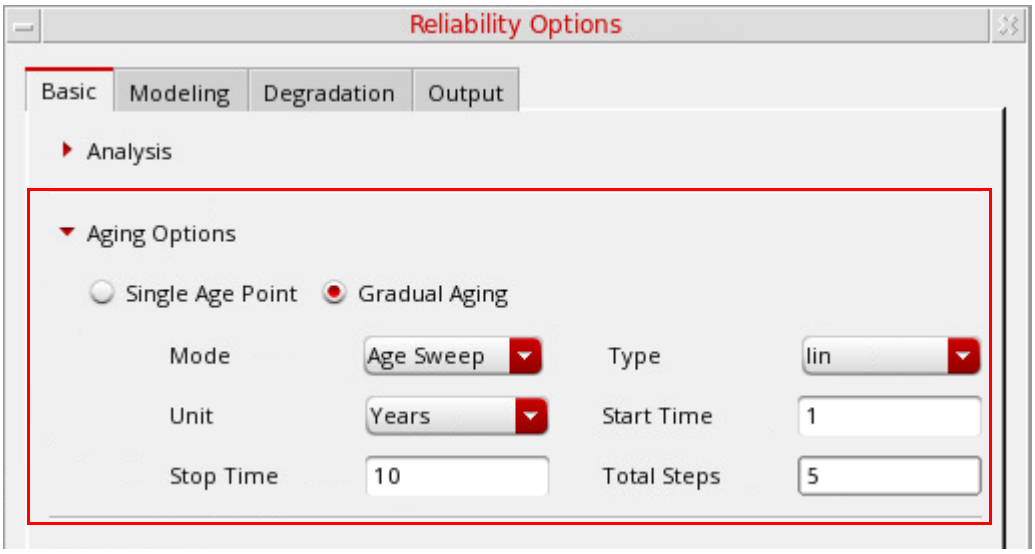


Note: When you plot the results of Reliability analysis across corners, age is plotted on the x-axis.

Printing or Plotting Stress Results for Gradual Aging Run

If you had chosen to save the results data for gradual aging, you can print or plot the stress results for every age point for which data was saved.

© 2023 Cadence Design Systems, Inc.



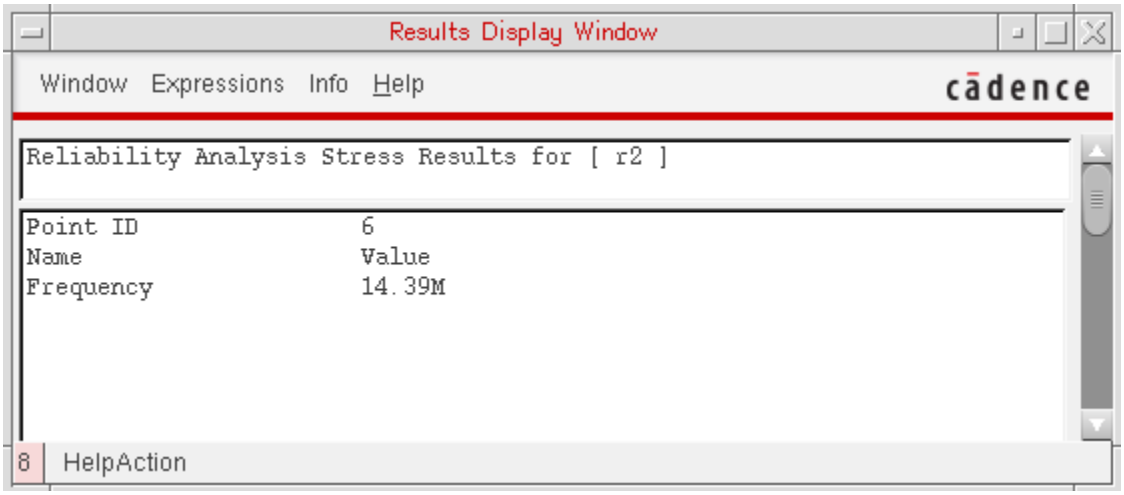
- Printing Stress Results for an Age Point
- Plotting Stress Results for an Age Point

Printing Stress Results for an Age Point

To print stress results for an age point:

1. On the *Results* tab, select the result of that age point.
2. Right-click and choose *Print Stress Results*.

   A text file containing the stress results is displayed, as shown below.



© 2023 Cadence Design Systems, Inc.

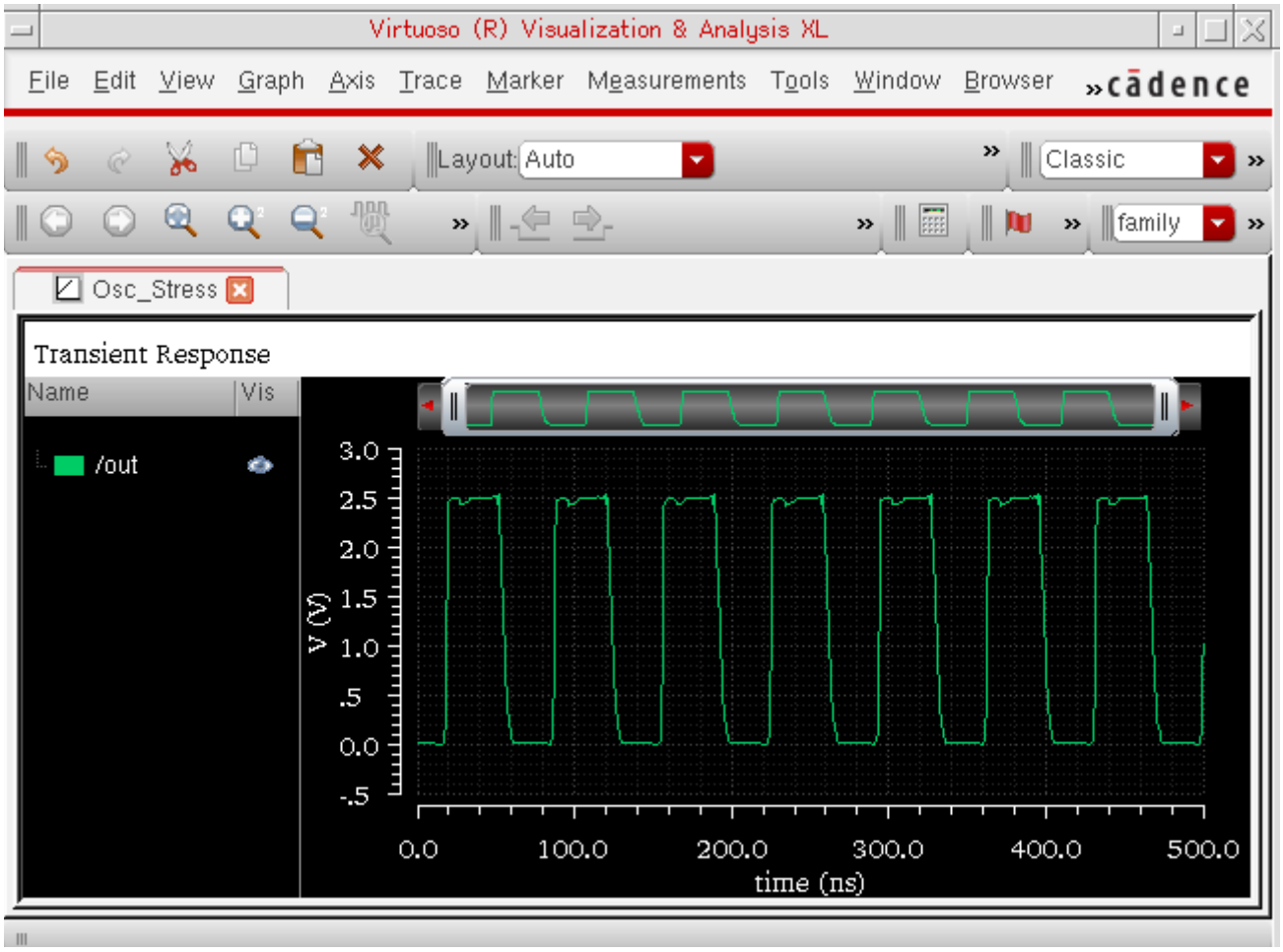
Note: The *Print Stress Results* command is available only for the aging points of the gradual aging run. It is not available for fresh and stress points or age point of non-gradual aging run.

Plotting Stress Results for an Age Point

To plot stress results for an age point:

1. On the *Results* tab, select the result of that age point.
2. Right-click and choose *Plot Stress Results*.

   The results are plotted in the Virtuoso Visualization and Analysis XL window, as shown below.



Note: The *Plot Stress Results* command is available only for the aging points of the gradual aging run. It is not available for fresh and stress points or age point of non-gradual aging run.

© 2023 Cadence Design Systems, Inc.

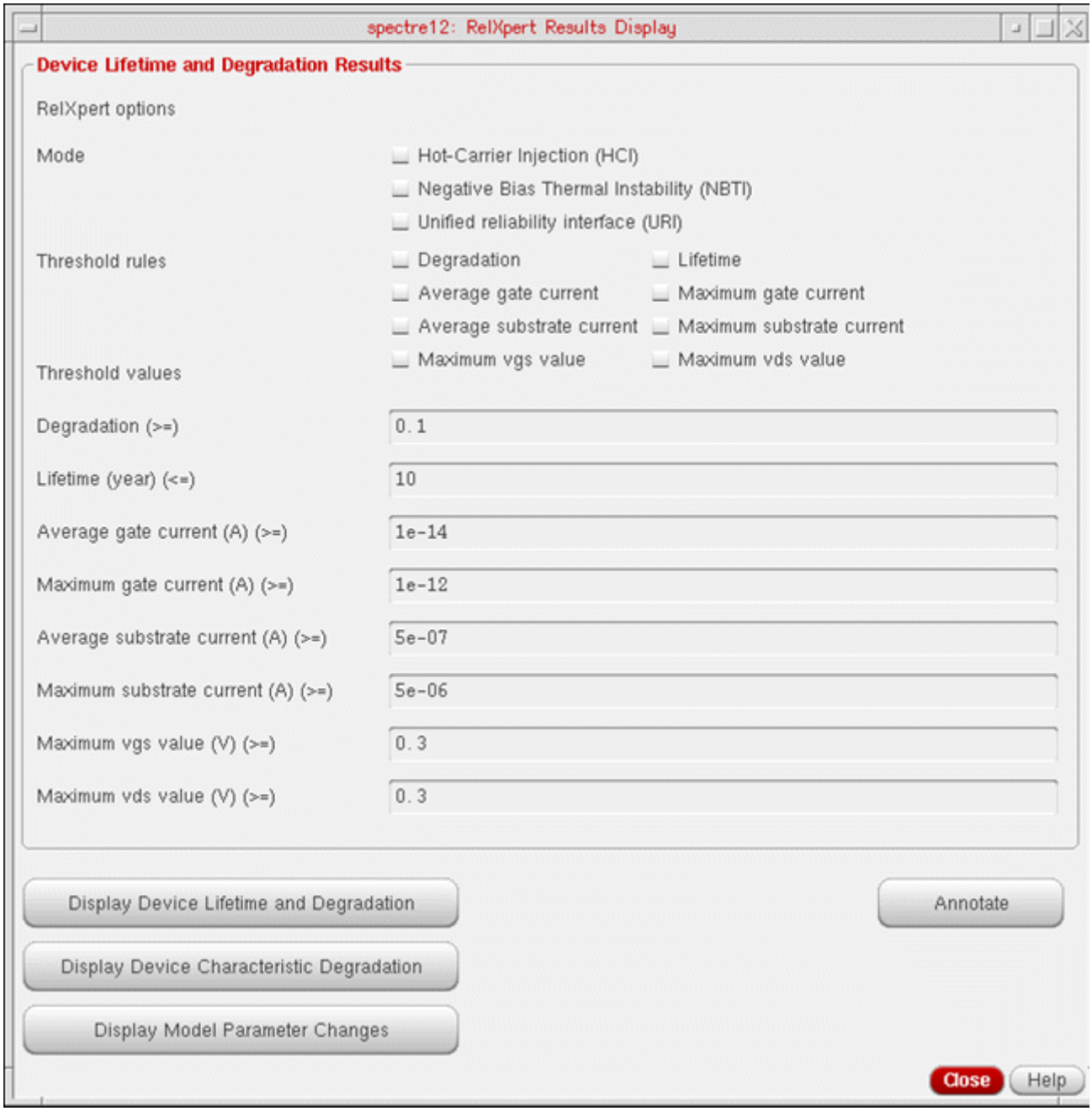
Annotating Simulation Results to Schematic View

To annotate the device lifetime and degradation results to schematic view:

1. In the Virtuoso ADE Assembler window, on the *Results* tab, select the required test.
2. Right-click the selected test and choose *Reliability Data – Results*.

© 2023 Cadence Design Systems, Inc.

The RelXpert Results Display form is displayed.

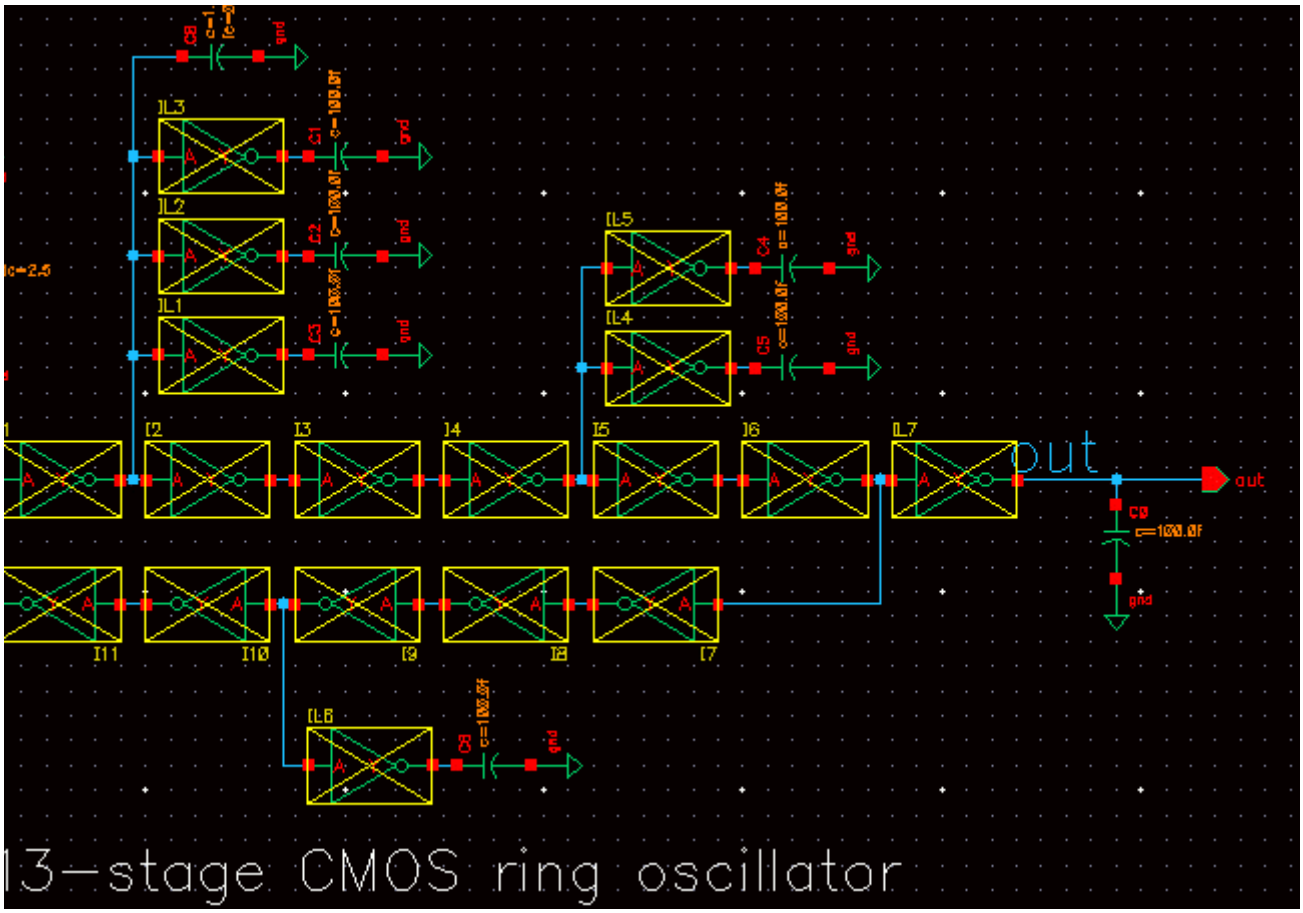


3. Select the RelXpert modes for which you want to backannotate the results.

   Results are backannotated based on the specified threshold values. For example, if the Lifetime (year) (<=) field specifies a threshold value of 10, results are backannotated only to those instances whose lifetime is less than or equal to 10 years.

© 2023 Cadence Design Systems, Inc.

4. Select the threshold rules based on which you want to backannotate the results. The threshold rules corresponding to the selected modes are displayed in the form. The fields containing the default threshold values for the selected rules become available.
5. Click the *Annotate* button. The instances to which the results are backannotated are highlighted in yellow in the schematic.



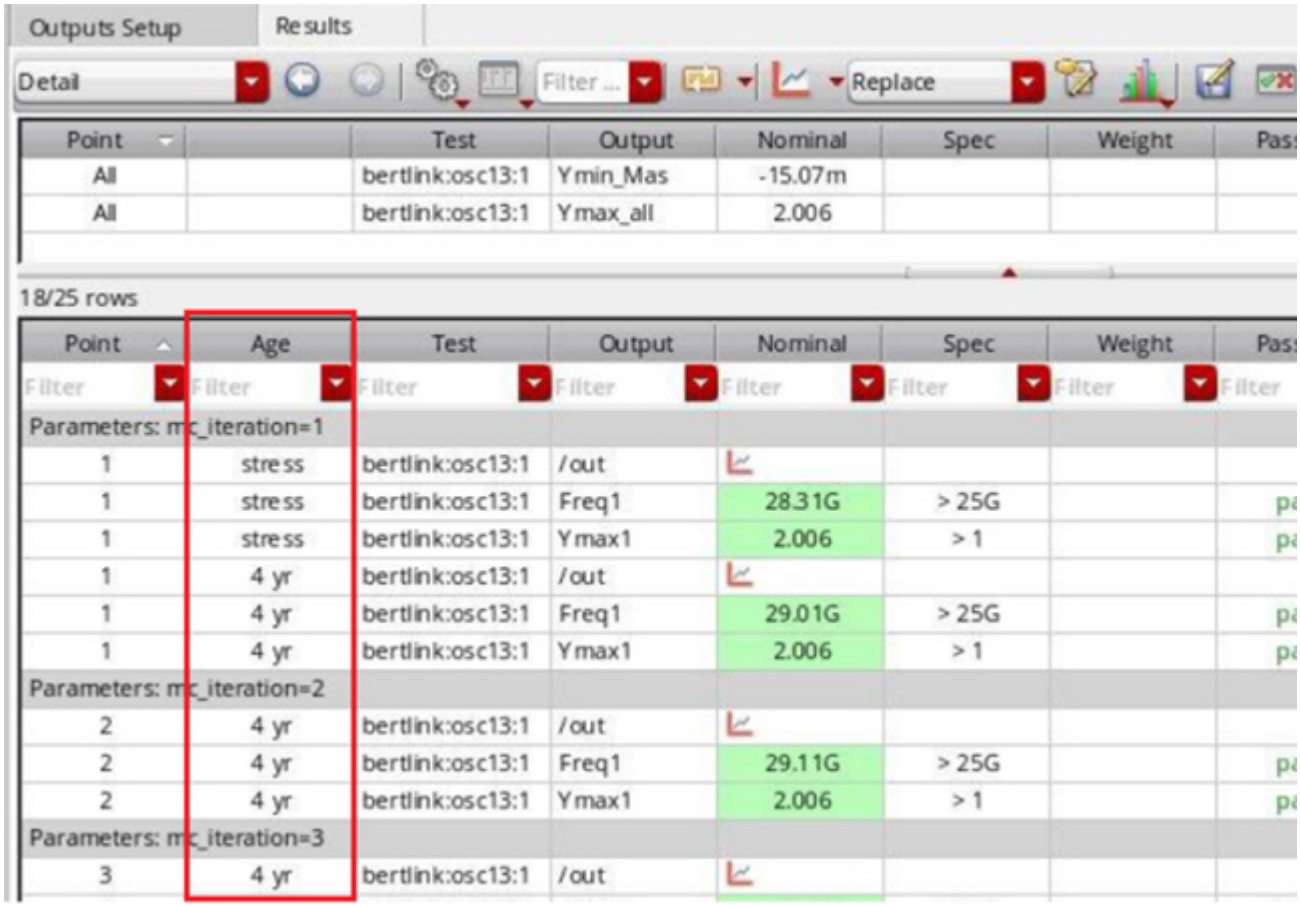
Note: In a hierarchical design, a block is highlighted if results are backannotated to an instance in that block. When you descend into the block, the instances to which the results are backannotated are also highlighted.

Performing Reliability Analysis for Monte Carlo Run Mode

You can perform reliability analysis in the *Monte Carlo Sampling* run mode using the *Standard Monte Carlo* method.

© 2023 Cadence Design Systems, Inc.

After running the simulation, use the Detail view on the Results tab to view the detailed results for the reliability analysis.



In this view, the *Age* column displays the source of simulation results. The possible values are:

- `fresh`: fresh simulation
- `stress`: stress simulation
- *<age point>*: aged simulation
- `she`: self-heating simulation
- *<age point>*`_she`: combination self-heating and aged simulation

Limitations

The following features are not supported with reliability analysis in a Monte Carlo simulation:

© 2023 Cadence Design Systems, Inc.

- Gradual Aging
- Advanced Monte Carlo methods
- *Latin Hypercube* and *Low Discrepancy Sequence* sampling methods
- Job distribution
- Measure outputs for SPICE

Important

When reliability analysis is run with the Monte Carlo run mode, stress simulation is run in 1: N ratio of total iterations. That is, for all N iterations, only one stress simulation is run. The stress simulation is run for nominal point followed by aging simulation for the remaining points.

A stress simulation run for Monte Carlo does not consider statistical variations. Therefore, you may note that the results of a Monte Carlo stress simulation does not match the results of statistical corner simulations.

© 2023 Cadence Design Systems, Inc.

Viewing Reliability Report

Important

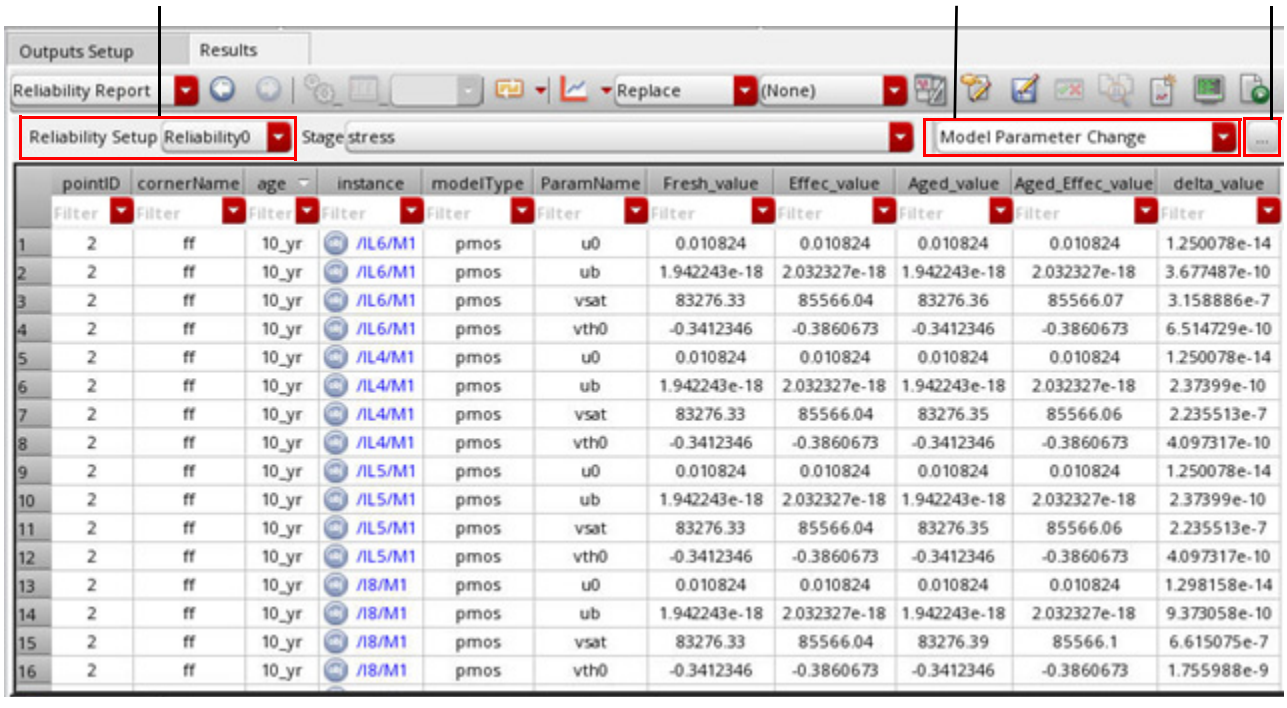
Reliability report works with Spectre version 20.1 ISR10 or later. This feature is not supported for third-party simulators, RelXpert mode, and Reliability Analysis in Monte Carlo run mode.

Reliability report lets you view reliability analysis results in a tabular format. The tabular format makes it easy to view, sort, and filter reliability results data.

To view reliability report, do one of the following:

- In the *Results* tab, from the *Select the results view* drop-down list, select *Reliability Report*.
- In the Results tab, right-click an output value for a stress result and then select *Reliability Data – Switch to Reliability Report*.

The following figure shows an example of how the reliability report shows the results of *Model Parameter Change* for the Reliability Setup, *Reliability0*.

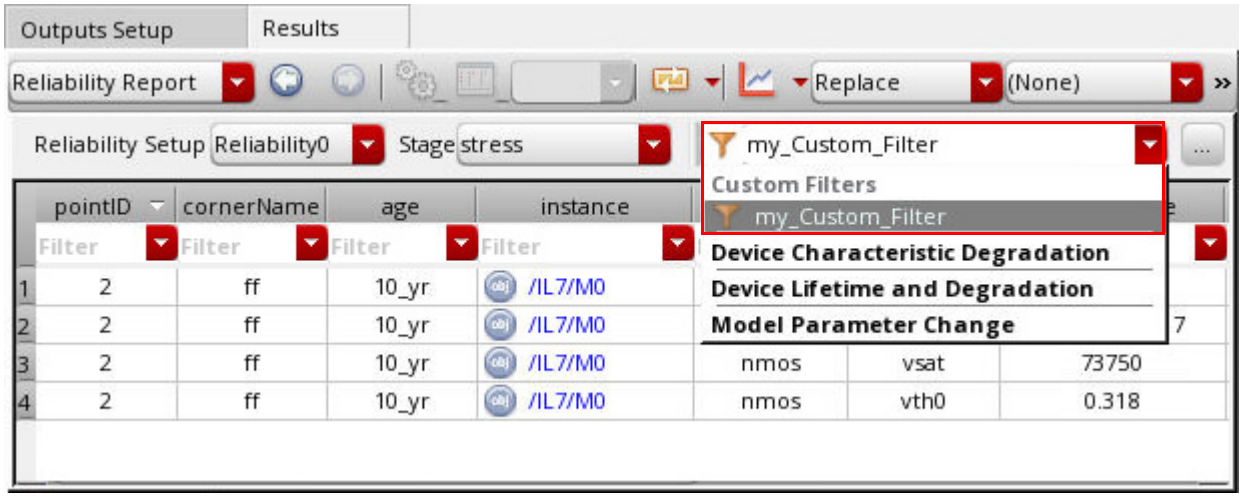


	pointID	cornerName	age	instance	modelType	ParamName	Fresh_value	Effec_value	Aged_value	Aged_Effec_value	delta_value
1	2	ff	10_yr	/IL6/M1	pmos	u0	0.010824	0.010824	0.010824	0.010824	1.250078e-14
2	2	ff	10_yr	/IL6/M1	pmos	ub	1.942243e-18	2.032327e-18	1.942243e-18	2.032327e-18	3.677487e-10
3	2	ff	10_yr	/IL6/M1	pmos	vsat	83276.33	85566.04	83276.36	85566.07	3.158886e-7
4	2	ff	10_yr	/IL6/M1	pmos	vth0	-0.3412346	-0.3860673	-0.3412346	-0.3860673	6.514729e-10
5	2	ff	10_yr	/IL4/M1	pmos	u0	0.010824	0.010824	0.010824	0.010824	1.250078e-14
6	2	ff	10_yr	/IL4/M1	pmos	ub	1.942243e-18	2.032327e-18	1.942243e-18	2.032327e-18	2.37399e-10
7	2	ff	10_yr	/IL4/M1	pmos	vsat	83276.33	85566.04	83276.35	85566.06	2.235513e-7
8	2	ff	10_yr	/IL4/M1	pmos	vth0	-0.3412346	-0.3860673	-0.3412346	-0.3860673	4.097317e-10
9	2	ff	10_yr	/IL5/M1	pmos	u0	0.010824	0.010824	0.010824	0.010824	1.250078e-14
10	2	ff	10_yr	/IL5/M1	pmos	ub	1.942243e-18	2.032327e-18	1.942243e-18	2.032327e-18	2.37399e-10
11	2	ff	10_yr	/IL5/M1	pmos	vsat	83276.33	85566.04	83276.35	85566.06	2.235513e-7
12	2	ff	10_yr	/IL5/M1	pmos	vth0	-0.3412346	-0.3860673	-0.3412346	-0.3860673	4.097317e-10
13	2	ff	10_yr	/I8/M1	pmos	u0	0.010824	0.010824	0.010824	0.010824	1.298158e-14
14	2	ff	10_yr	/I8/M1	pmos	ub	1.942243e-18	2.032327e-18	1.942243e-18	2.032327e-18	9.373058e-10
15	2	ff	10_yr	/I8/M1	pmos	vsat	83276.33	85566.04	83276.39	85566.1	6.615075e-7
16	2	ff	10_yr	/I8/M1	pmos	vth0	-0.3412346	-0.3860673	-0.3412346	-0.3860673	1.755988e-9

© 2023 Cadence Design Systems, Inc.

By default, the drop-down list in the top-right corner in the Reliability Report view of the Results tab displays only the main types of reliability reports, such as *Device Characteristic Degradation*, *Device Lifetime and Degradation*, and *Model Parameter Change*.

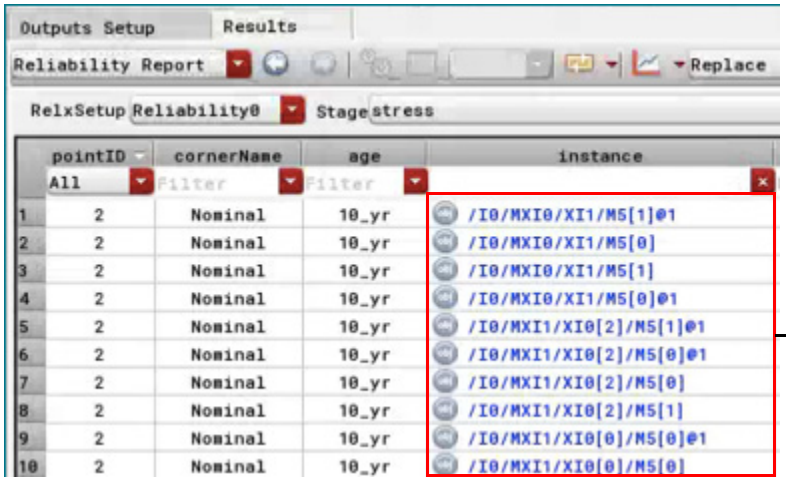
However, a new category, *Custom Filters*, is added at the top of this drop-down list. All the custom filters are listed under this category.



You can also open an instance in the schematic directly from the reliability report by clicking the hyperlink of that instance in the instance column.

In case of post-layout simulations, reliability report also supports annotations for out-of-context devices, which means that when you click the hyperlinks of such instances in the reliability report, it will open these instances in the schematic.

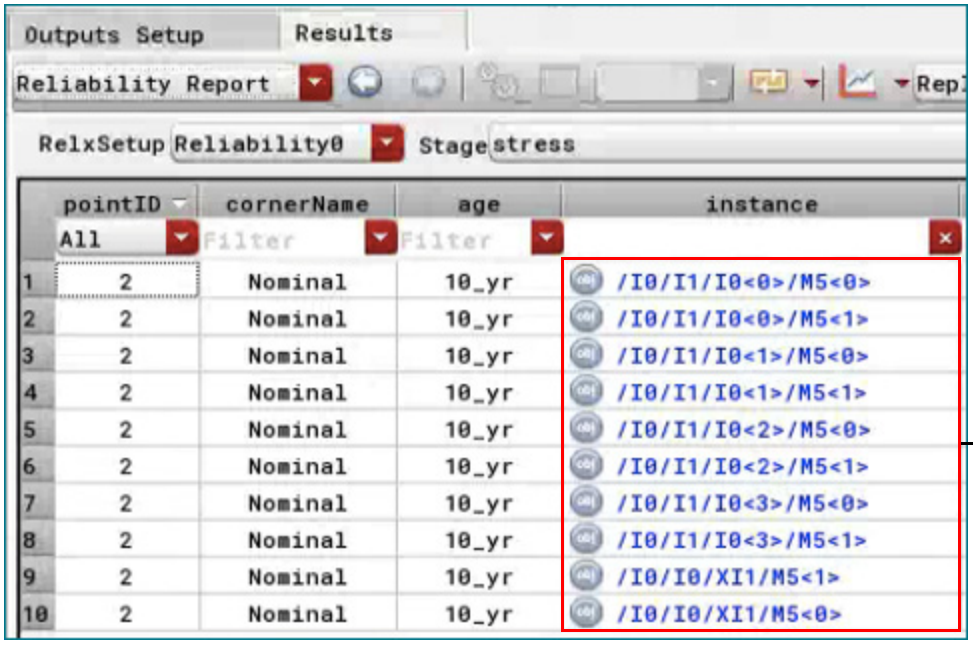
By default, reliability report shows instance names in post-layout format.



Instance names in post-layout format

© 2023 Cadence Design Systems, Inc.

However, you can set the environment variable `maestro.results postLayoutToSchematicInRelReport` to `t` to view instance names in schematic format in the reliability report.



You can also read the related blog at Virtuosity: Reliability Analysis Report-Reliable Results Made Interactive.

Related Topics

Customize Reliability Data Filters Form

Annotating Degradation Parameters on the Schematic

Customizing the Display of the Reliability Report

Applying Filters to Columns

Creating a Custom Reliability Data Filter

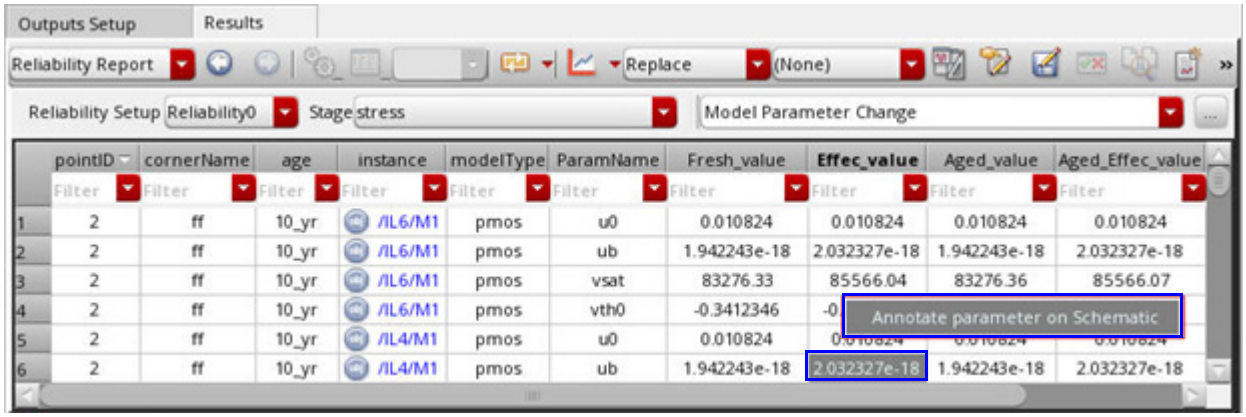
postLayoutToSchematicInRelReport

© 2023 Cadence Design Systems, Inc.

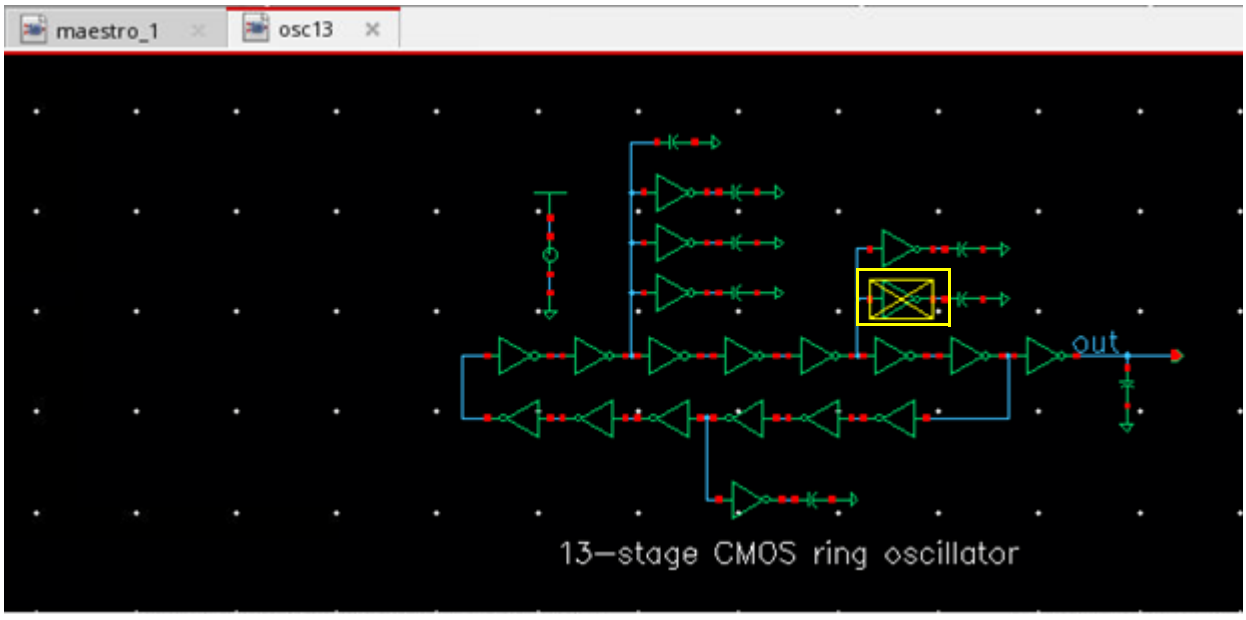
Annotating Degradation Parameters on the Schematic

To annotate a degradation parameter on the schematic:

➡ In the reliability report, right-click a degradation parameter, and choose *Annotate parameter on Schematic*.



The degradation parameter is annotated on the schematic.



Related Topics

Viewing Reliability Report

© 2023 Cadence Design Systems, Inc.

Customizing the Display of the Reliability Report

You can customize the display of the reliability report in the following ways:

- **Hide a column**: Right-click a column header and then select *Hide Column* to hide the column you do not want to see. To view the hidden columns, right-click any column header and select *Show All Columns*.
- **Change the order in which columns are displayed**: Select the column that you want to move and drag it to the location where you want to place it.

Related Topics

Viewing Reliability Report

© 2023 Cadence Design Systems, Inc.

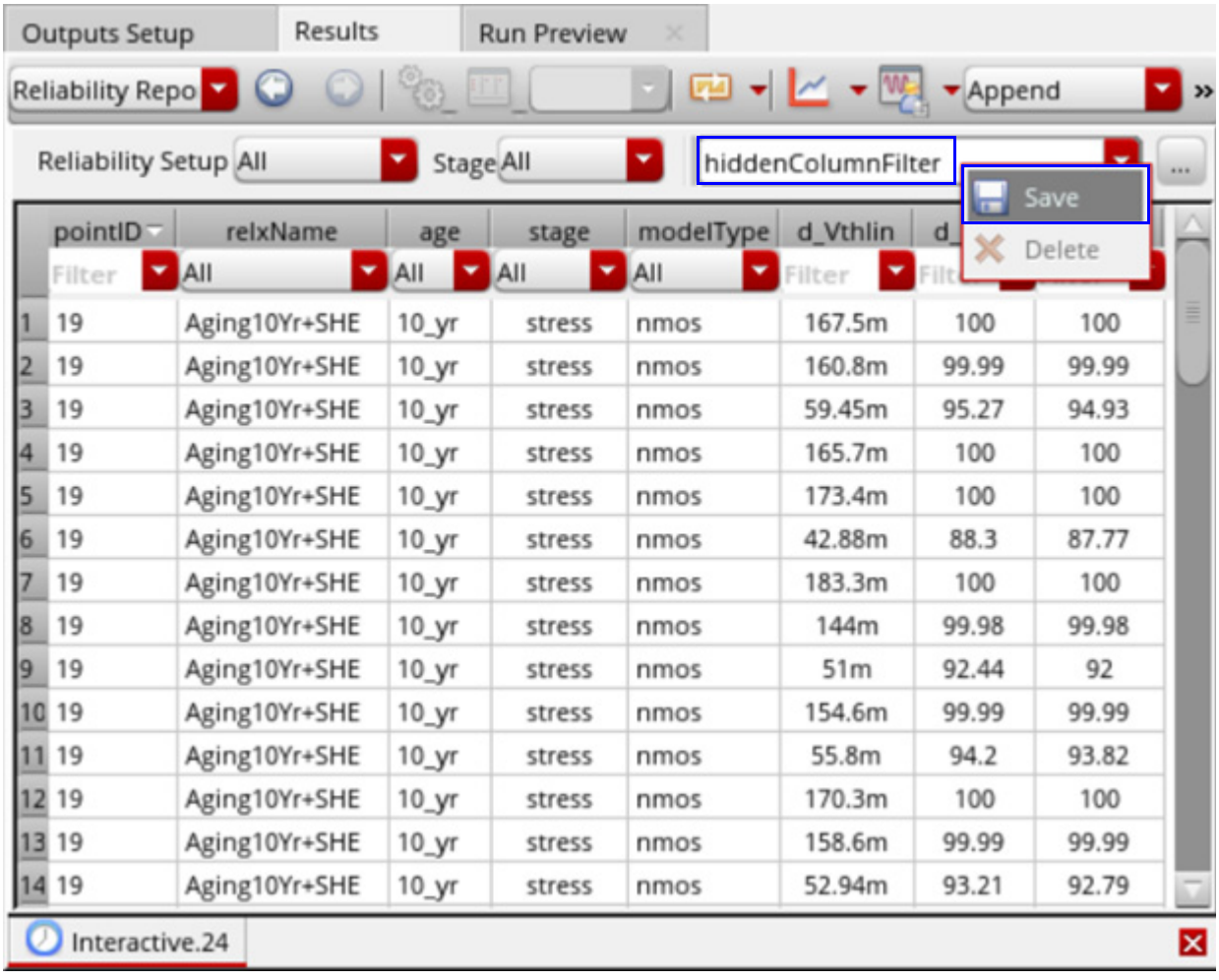
Creating a Data Filter to Hide Selected Columns

To create a data filter to hide the selected columns in the *Reliability Report* view:

1. Right-click the header name of the column that you want to hide, and choose *Hide Column*.
2. Repeat step 1 for other columns you want to hide.

   For example: *cornerName*, *instance*, *d_Vthsat*, *d_Gm*, and *d_Gds*.
3. Type the name of the data filter in the *Select a reliability data filter* field.

   For example: `hiddenColumnFilter`.



4. Right-click anywhere in the field, and click *Save*.

© 2023 Cadence Design Systems, Inc.

The filter is saved with the specified name. When you apply this filter, the columns you selected to hide are not displayed in the *Reliability Report* view.

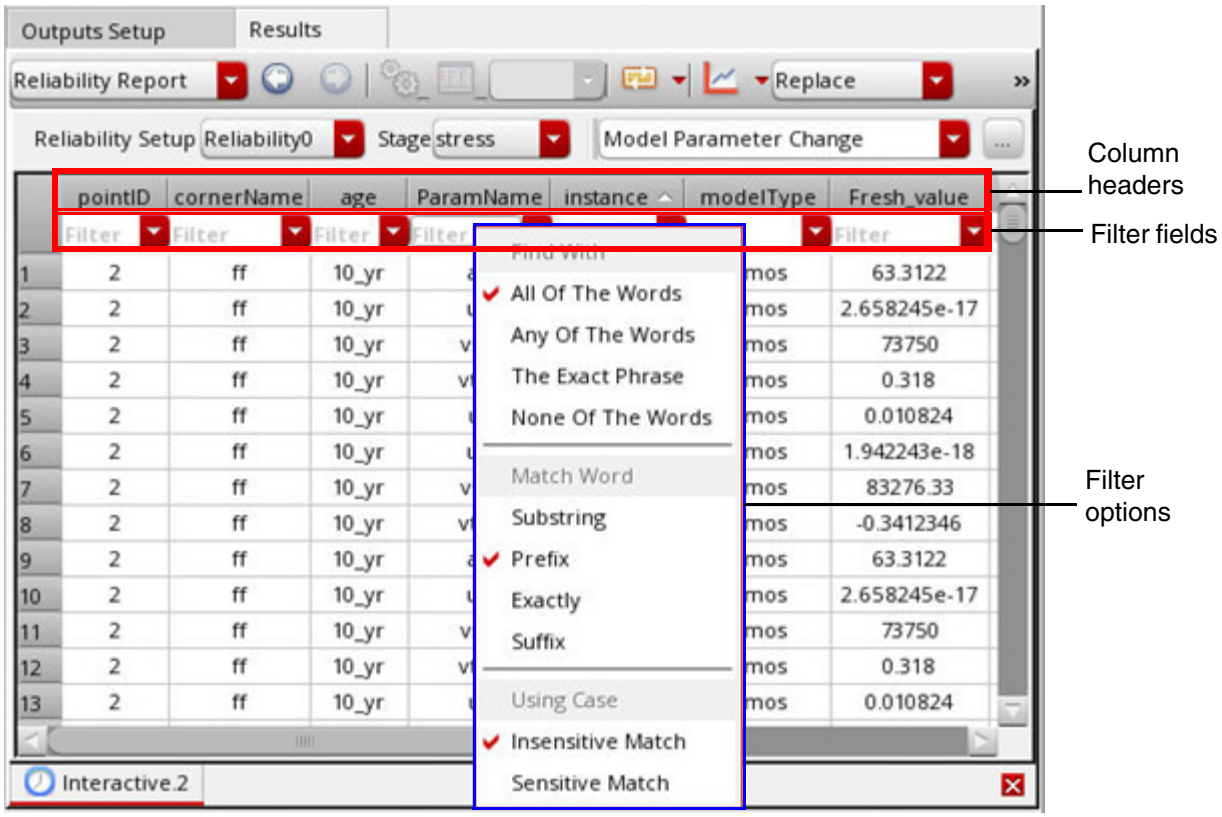
© 2023 Cadence Design Systems, Inc.

Applying Filters to Columns

You can apply filter criteria to column to selectively view the results that match the specified filter criteria.

To apply filter criteria to columns:

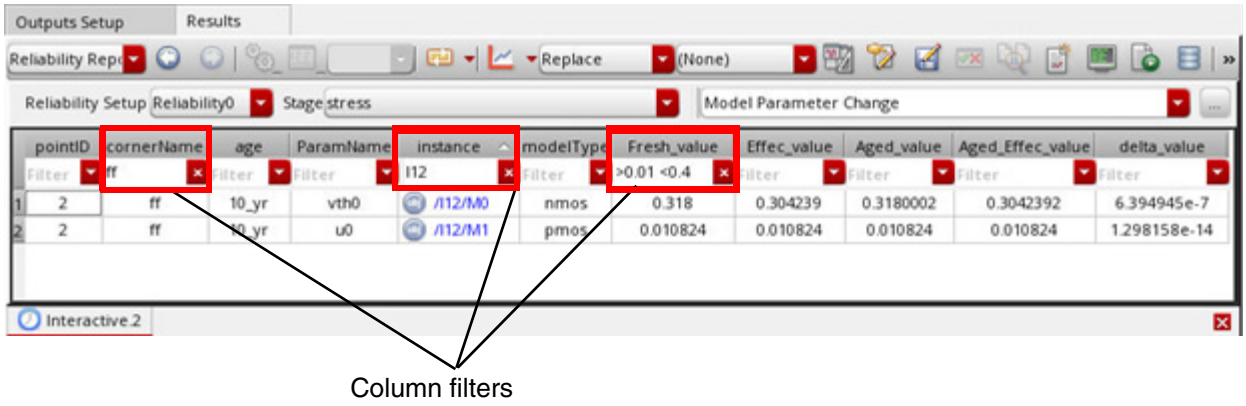
1. Right-click anywhere in the *Filter* field below the column header to which you want to apply the filter, and select the filter options.



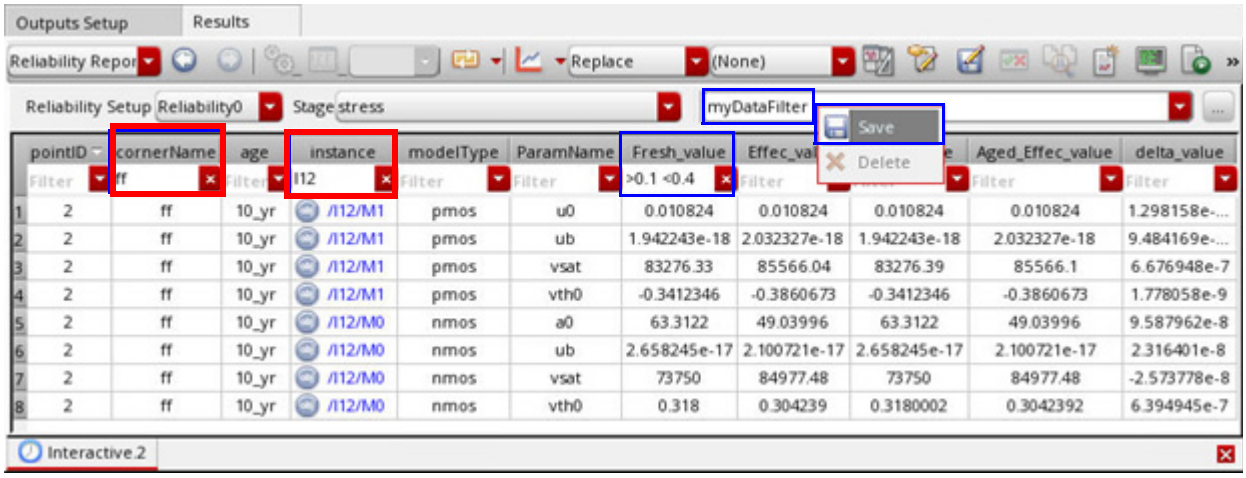
2. Enter the filter criteria in the *Filter* field and click *Enter* to filter the reliability results in the reliability report.

© 2023 Cadence Design Systems, Inc.

The following example shows how multiple column filters have been used to view the reliability results only for the *cornerName* `ff` and *instance* `I12`, where the *Fresh_value* is between `0.01` and `0.4`.



3. In the drop-down list for the custom filter selection, type a name with which you want to save the filter, and right-click the drop-down list and choose *Save*.

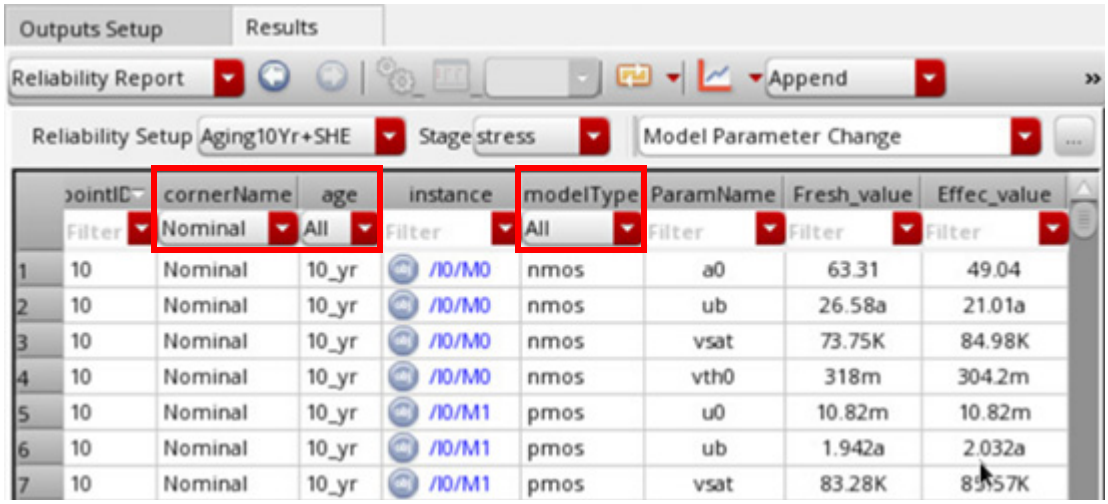


The filters are saved in the `.cadence/dfII/maestro/reliability-filters` directory and are available for use when you reopen the cellview later. You can reapply a saved filter by selecting it from the list of custom filters. If not required, you can delete a saved filter by using the *Delete* command in the context-sensitive menu of the drop-down list.

The saved filters are also available in the Customize Reliability Data Filters form, where you can modify them further.

You can use the drop-down lists below the headers of the cornerName, age, and modelType columns to filter values in these columns.

© 2023 Cadence Design Systems, Inc.



Filter Options

The following table describes various filter options that can be applied on the column headers of the reliability report to filter the results.

Filter Options	Description
Find With	Lets you specify a word-based filter criterion.
All Of The Words	Displays the results that contain all of the words specified in the *Filter* field.
Any Of The Words	Displays the results that contain any of the words specified in the *Filter* field.
The Exact Phrase	Displays the results that contain the exact phrase specified in the *Filter* field.
None Of the Words	Displays the results that do not contain the words specified in the *Filter* field.
Match Word	Lets you specify a string-based filter criterion.
Substring	Displays the results that contain the words specified in the *Filter* field.
Prefix	Displays the results that start with the words specified in the *Filter* field.
Exactly	Displays the results that exactly match the words specified in the *Filter* field.

© 2023 Cadence Design Systems, Inc.

Filter Options	Description
Suffix	Displays the results that end with the words specified in the *Filter* field.
Using Case	Lets you specify a case-based filter criterion.
Insensitive Match	Filters the results regardless of the case specified in the *Filter* field.
Sensitive Match	Filters the results matching the case specified in the *Filter* field.

Related Topics

Viewing Reliability Report

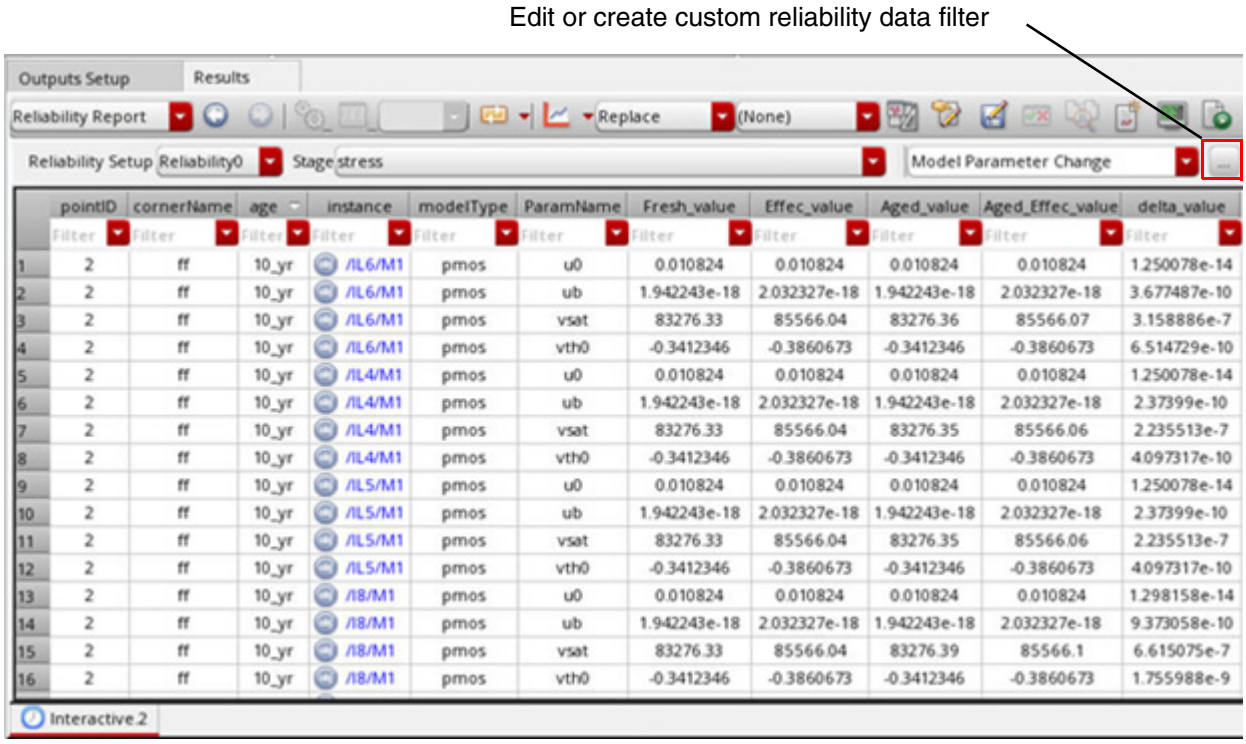
© 2023 Cadence Design Systems, Inc.

Creating a Custom Reliability Data Filter

You can create your own custom reliability data filters with different filter criteria and save these custom filters so that they can be reused for the same or different reliability analysis reports and reliability setups.

To create a custom reliability data filter:

1. In the *Reliability Report* view of the *Results* tab, click the *Edit or create custom reliability data filter* command.



© 2023 Cadence Design Systems, Inc.

The Customize Reliability Data Filters form opens.

2. In the *Filter Name* field, enter the name of the custom reliability data filter that you want to create.
3. In the *Description* field, add a description for the custom reliability data filter.
4. From the *Shows* list, select the type of reliability report to which you want to apply the custom filter.
5. From the *Available Fields* list, select a column name and then click the left arrow (<-) to add it to the *Where* field.
6. In the *Where* field, you can build expressions to define filter criteria using SQLite operators.

© 2023 Cadence Design Systems, Inc.

The following example shows a custom filter named `my_Custom_Filter` that filters the results of the *Model Parameter Change* report in which *delta_value* is greater than `4.645217e-9` for the instance `/IL7/M0`.

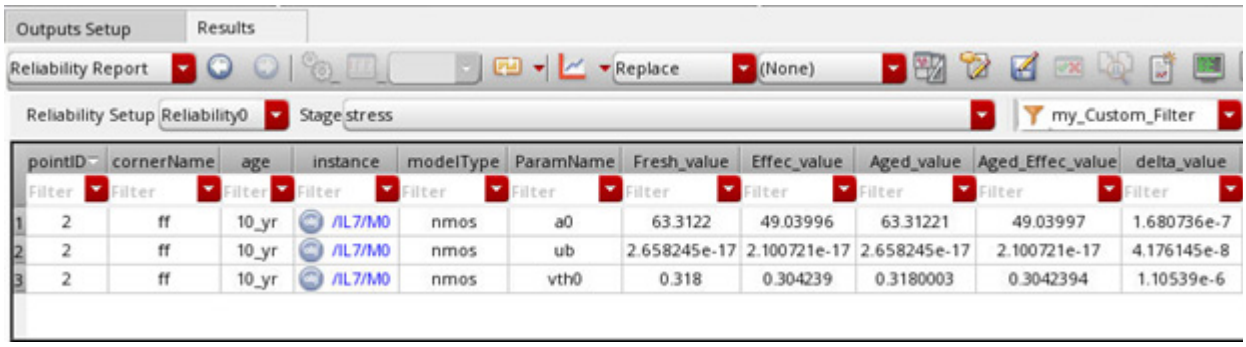
7. Click *Save* on the Customize Reliability Data Filters form to save these settings in the custom filter.

   Note: The custom filters are saved as XML files in the `.cadence` directory under the `dfII/maestro/reliability-filters` subdirectory. You can configure the Cadence Setup Search File mechanism (CSF) to specify additional locations from where filters can be read. This allows you to create a library of filters that can be shared amongst multiple users.

8. Click *OK* to apply the custom filter to the reliability report.

© 2023 Cadence Design Systems, Inc.

9. The results of the reliability report are filtered based on the applied custom filter.



	pointID	cornerName	age	instance	modelType	ParamName	Fresh_value	Effec_value	Aged_value	Aged_Effec_value	delta_value
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	2	ff	10_yr	/IL7/M0	nmos	a0	63.3122	49.03996	63.31221	49.03997	1.680736e-7
2	2	ff	10_yr	/IL7/M0	nmos	ub	2.658245e-17	2.100721e-17	2.658245e-17	2.100721e-17	4.176145e-8
3	2	ff	10_yr	/IL7/M0	nmos	vth0	0.318	0.304239	0.3180003	0.3042394	1.10539e-6

Related Topics

Creating a Custom Reliability Data Filter

Custom Filter Syntax

Query Operators for Custom Filters

Viewing Reliability Report

© 2023 Cadence Design Systems, Inc.

Custom Filter Syntax

- Instance and net names must use the schematic namespace, not the simulator namespace. For example, `/IL7/M0` is correct, whereas `IL7.M0` is incorrect.
- Query operators are case-insensitive.
- Strings can be in single or double quotes.
- If the field names contain spaces or special characters, they must be enclosed in double quotes.
- The borders of the *Where* field on the Custom Reliability Data Filters form are highlighted in red if there are errors in the syntax of the custom filter. A tooltip is also displayed, indicating the location of the syntax error when you hover over the *Where* field.

Related Topics

Creating a Custom Reliability Data Filter

Query Operators for Custom Filters

Viewing Reliability Report

Query Operators for Custom Filters

The custom filters support all SQLite operators. The following table lists some of the commonly used operators.

Operator	Definition	Example
`=, ==, !=, <>`	Equal, not equal	`"delta_value" = 4.645217e-9`
`IS, IS NOT, NOT`	Equality operators. Use this to query the Status column	`"instance" IS NOT "/IL7/M0"`
`>, <, >=, <=`	Greater than, less than, greater than or equal, less than or equal	`"delta_value" > 4.645217e-9`
`BETWEEN`	Between an inclusive range	`"delta_value" BETWEEN 4.645217e-9 AND 5.963147e-9`

© 2023 Cadence Design Systems, Inc.

IN	List of possible values	"modelType" IN ('pmos', 'nmos')
LIKE	Searches for a case insensitive pattern. % is used to define wildcards before and after the pattern.	"User Message" LIKE '%exceeds%'
GLOB, MATCHES	Case sensitive string matching that uses * and ? wildcards	"User Message" GLOB '*g45?1svt*VGS*' "User Message" MATCHES '*g45?1svt*VGS*'
CONTAINS	Shortcut for GLOB '*foo*'	"instance" CONTAINS 'M0'
STARTSWITH, ENDSWITH	Shortcut for GLOB 'foo*', GLOB '*foo'	"instance" STARTSWITH 'I18' "instance" ENDSWITH 'NM0'
AND, OR	Combines multiple search expressions	"delta_value" > 4.645217e-9 AND "instance" IS "/IL7/M0"

Related Topics

Creating a Custom Reliability Data Filter

Custom Filter Syntax

Viewing Reliability Report

© 2023 Cadence Design Systems, Inc.

Customize Reliability Data Filters Form

Use the Customize Reliability Data Filters form to create or edit filter criteria to selectively view the reliability results data.

Field	Description
Filter Name	Name of the custom reliability data filter.
Description	Description of the custom reliability data filter.
Shows	Type of reliability reports to be displayed: ■ *Device Lifetime and Degradation* ■ *Device Characteristic Degradation* ■ *Model Parameter Changes* ■ *Self-Heating Results*
Available Fields	The list of fields or columns available for the selected reliability report type. This list varies with the type of reliability report selected in the *Shows* list.
Where	Builds the expression to create the custom reliability data filter.

Related Topics

Creating a Custom Reliability Data Filter

Custom Filter Syntax

Query Operators for Custom Filters

Viewing Reliability Report

© 2023 Cadence Design Systems, Inc.

17

Running Simulations

Before running a simulation, you need to prepare a setup using job policy to specify the resources to be used for different services supporting the simulation. This chapter provides information about the setup details. In addition, it explains how to manage simulations when you need to suspend or resume a run, or to close a Virtuoso session while a simulation is in progress.

For details, refer to the following topics:

- The Simulation Flow in ADE Assembler
- Using Simulation Manager with LSCS
- Using Convergence Aids
- Specifying a Name for Simulation History
- Starting a Simulation
- Suspending and Resuming Netlisting or Simulation Runs
- Stopping a Simulation
- Canceling Simulations for Selected Tests or Corners
- Checking Run Status on the Progress Bar
- Stopping Jobs and Resubmitting Simulations
- Continuing the In-Process Simulations After ADE Assembler GUI Exits
- Viewing the ADE Assembler Logs After Running Simulations
- Viewing Job Status
- Viewing the Job Log
- Viewing the Simulation Output Log File
- Running an Incremental Simulation

© 2023 Cadence Design Systems, Inc.

- Simulating Only Unfinished or Erroneous Points
- Setting Up Debug Options
- Debugging Points
- Creating and Running an OCEAN Script

© 2023 Cadence Design Systems, Inc.

The Simulation Flow in ADE Assembler

All simulations running in ADE Assembler are distributed according to the job control method specified in the job policy.

ADE Assembler supports the following two job control modes:

- **LSCS**: This is the default job control mode. It is a highly scalable job control mode that can handle thousands of netlisting and simulation jobs in parallel. This mode contains three systems: Virtuoso, Netlist Service, and Simulation Service. These systems work independently and concurrently with the shared queues of messages. The workflow being divided into separate processes means that the resources are consumed only when they are needed. Therefore, this model enables a more efficient exchange of information between these distinct processes.

  For more details about how to configure setup and run simulation for LSCS mode, refer to Running Simulations with LSCS Job Control Mode.

- **ICRP**: This uses the IC Remote Processes that are started by ADE Assembler in the `virtuoso -nograph` mode. These remote processes perform the following tasks:

  - Generate netlists
  - Run simulations
  - Evaluate the results
  - Return the results to the ADE Assembler user interface

  Separate remote processes imply that the above tasks do not block the ADE Assembler user interface and still enable you to continue working when the simulation run is in progress.

  An ICRP process may start a few child processes (clsbd, oaFSLockD, cdsNameServer, cdsServIpc) to perform some special tasks. These child processes may become daemon and continue to exist longer than the ICRP process itself. These processes should not be killed when the original ICRP job has exited because new ICRPs that are dependent on these daemon processes may have been started in the intervening time.

  Note: A daemon is a program or process that runs as a background process.

  For more details about how to configure setup and run simulation for ICRP mode, refer to Running Simulations With ICRP Job Control Mode.

© 2023 Cadence Design Systems, Inc.

Using Simulation Manager with LSCS

With LSCS job control mode, you can use ADE simulation manager, also referred to as the simulation management service, which enables a decentralized and distributed architecture for the management of simulation jobs.

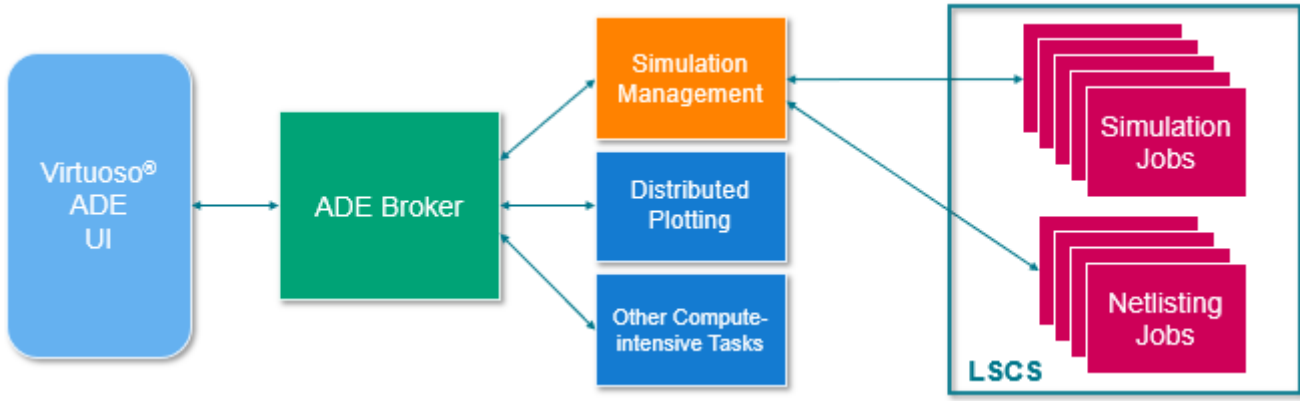
This architecture provides the following benefits:

- Offloads computation-intensive tasks, such as the management of simulation jobs, results plotting, and re-evaluation, from the Virtuoso session and makes the user interface of ADE Assembler more responsive to user interactions.
- Manages the progress of simulation jobs even if you close the host Virtuoso session that started the simulation. When you reopen the cellview, it reconnects the cellview to the simulation process if it is still in progress and you can control the simulations in the same way as you could before exiting Virtuoso.
- Provides a robust infrastructure that lets you reconnect with the running jobs and control them as before if Virtuoso stops abruptly during a simulation.

Using the simulation management service can be beneficial for complex designs and large setups that have a high simulation run time because it allows users to exit Virtuoso to release the memory reserved for other operations that could affect the simulation progress.

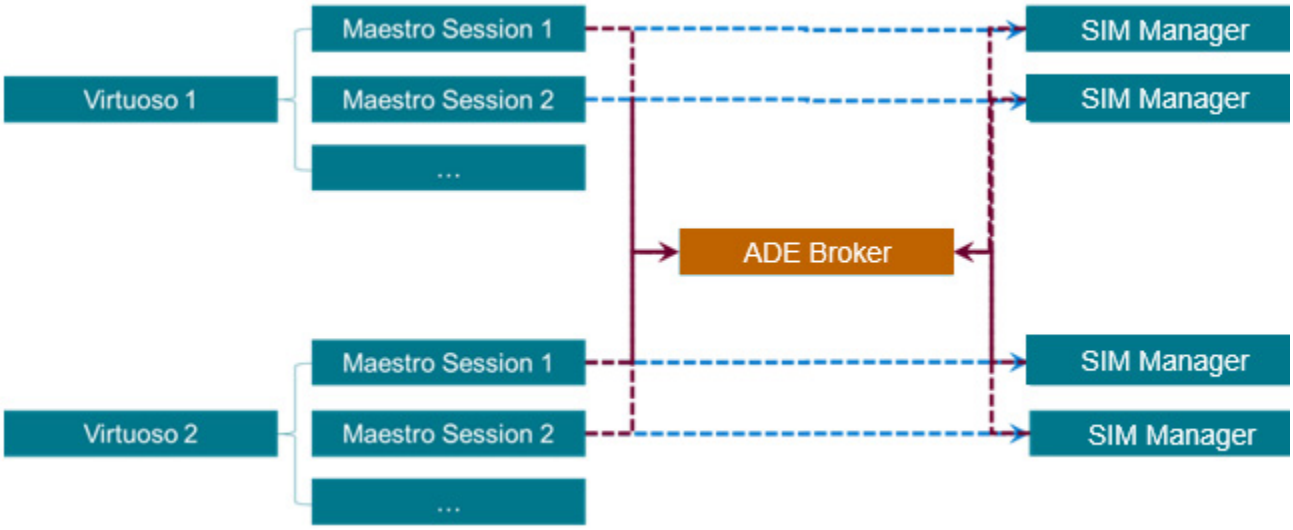
The Simulation Manager Infrastructure

The following diagram shows how the ADE simulation management tasks are offloaded from the ADE Assembler user interface. The simulation management service manages all netlisting and simulation jobs keeping the ADE Assembler UI available for other tasks.



© 2023 Cadence Design Systems, Inc.

This infrastructure is supported by a centralized ADE broker service that keeps a log of simulation manager services for all Virtuoso sessions and maestro cellviews running on a computer. The following diagram describes the architecture that uses this broker service.



The centralized ADE broker service is the key element of this infrastructure. This service can support both multi-user and single-user environments.

Related Topics

Running Simulations in a Multi-User Environment for Simulation Manager

Running Simulations in a Single-User Environment for Simulation Manager

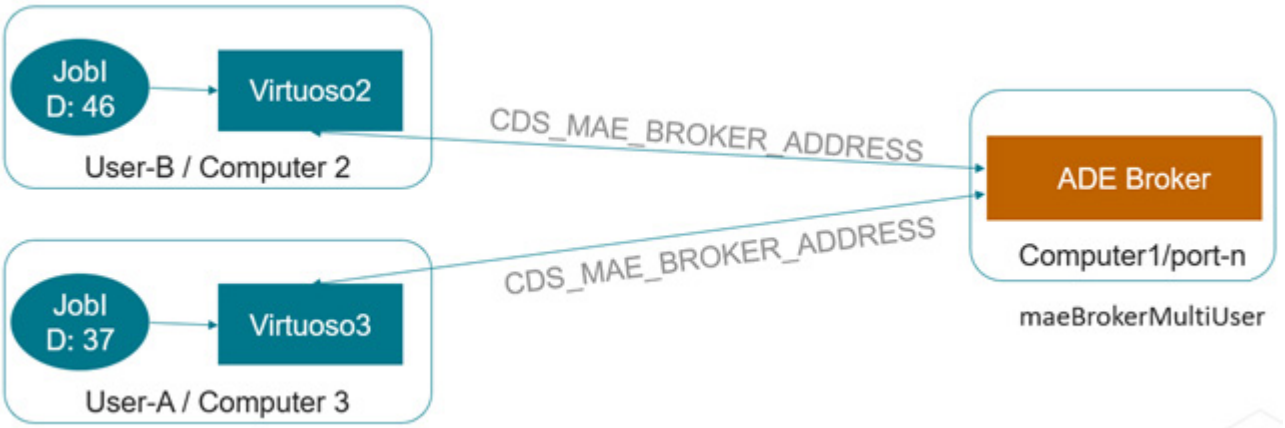
Enabling Simulation Manager

© 2023 Cadence Design Systems, Inc.

Running Simulations in a Multi-User Environment for Simulation Manager

In a multi-user environment, multiple Simulation Manager services and an ADE Broker service can run independently on separate hosts in the network. When launching a Virtuoso Studio instance, you need to specify the address and port information of the ADE broker service to which you want to connect the Virtuoso session.

A multi-user environment lets multiple Virtuoso sessions started by multiple users on different hosts to connect to a common ADE broker.



It is recommended that a CAD team or an administrator starts the ADE Broker service on a remote host by running the following command:

```
maeBrokerMultiUser -start-daemon
; Starts a multi-user ADE broker and assigns an available port. It also returns
; that port number. Users can connect to the ADE broker by specifying the port


maeBrokerMultiUser -start-daemon -port port-number
; Starts a multi-user ADE broker at the given port
```

To connect to this ADE broker, each user must specify the broker address by running the following command before launching Virtuoso Studio:

```
setenv CDS_MAE_BROKER_ADDRESS brokerHost:portNumber
```

For example:

```
setenv CDS_MAE_BROKER_ADDRESS host-lnx02:37431
```

When you run a simulation, Virtuoso Studio uses the `Simulation Manager Default` job policy to start a Simulation Manager service. You can customize the default job policy to use an LSF command to run this service on a remote host. Alternatively, you can use another job policy by setting the `simulationServiceJobPolicy` environment variable.

© 2023 Cadence Design Systems, Inc.

The connection with the Simulation Manager service is released after the simulation is successfully complete. The connection to the ADE broker is released after you close the Virtuoso Studio session.

In the event of a planned or an abrupt exit of Virtuoso Studio session while a simulation is in progress, the services run independently of the Virtuoso Studio session and take the in-progress simulation to completion.

The next time you launch Virtuoso Studio and reopen the maestro cellview for which simulation was started, the broker allows its reconnection to the services it was connected to in the previous Virtuoso Studio session. If the simulation running on a service is complete, it is not essential for the cellview to reconnect to the service. The results are saved in the history and can be viewed on the *Results* tab.

Related Topics

Enabling Simulation Manager

Managing Simulations with Simulation Manager

Exiting and Reconnecting with an In-progress Simulation

simulationServiceJobPolicy

© 2023 Cadence Design Systems, Inc.

Running Simulations in a Single-User Environment for Simulation Manager

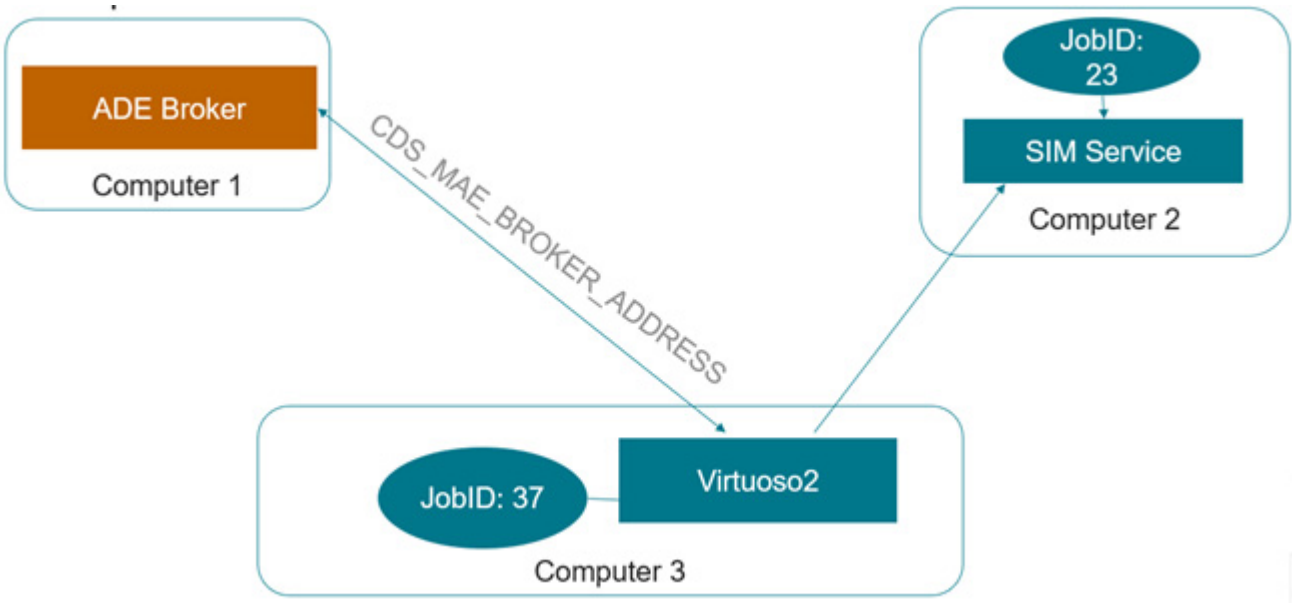
In a single-user environment, the ADE broker process can run on the same or a different host as that used by the Virtuoso Studio session.

You can start an ADE broker service by running the following command:

```
maeBroker -start-daemon
```

This command starts the ADE broker service and returns a port through which you can connect to the service.

When launching a Virtuoso Studio instance, you can specify the address and port information of the ADE broker service to which you want to connect the Virtuoso Studio session.



When you start a simulation run, Virtuoso launches a local or remote Simulation Manager service as per the specified job policy. `Simulation Manager Default` is the default job policy used for Simulation Manager. You can customize its settings or use another job policy by setting the `simulationServiceJobPolicy` environment variable.

In the event of a planned or an abrupt exit of the Virtuoso session while a simulation is in progress, the services take the in-progress simulations to closure.

The next time you launch Virtuoso Studio and reopen the maestro cellview for which simulation was started, the broker automatically reconnects the Virtuoso Session to the Simulation Manager service it was connected to in the previous Virtuoso session. This is

© 2023 Cadence Design Systems, Inc.

because the `maestro.simulation autoConnectBroker` environment variable is set to `t`. If the simulation running on a service is complete, it is not essential for the cellview to reconnect to the service. The results are saved in the history and you can view those on the *Results* tab.

Challenges in a Single-User Environment

Running and managing ADE broker service in a single-user environment has the following challenges:

- Before launching Virtuoso, you need to always ensure that an ADE broker process is running.
- You need to ensure that multiple ADE broker services are not running on the host. If multiple ADE brokers are running, you must use the `CDS_MAE_BROKER_ADDRESS` shell variable to connect to the correct broker.
- There are chances that an ADE broker is underutilized in a scenario when serving a single host.

Related Topics

Enabling Simulation Manager

Managing Simulations with Simulation Manager

Exiting and Reconnecting with an In-progress Simulation

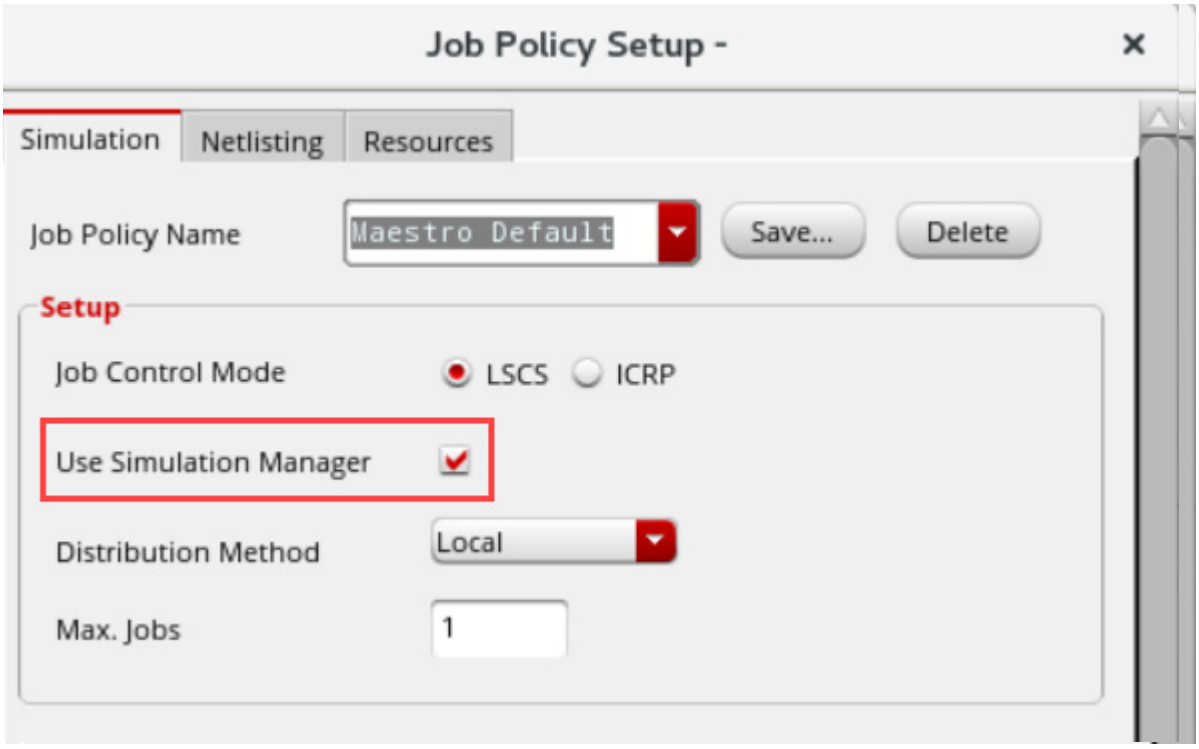
simulationServiceJobPolicy

© 2023 Cadence Design Systems, Inc.

Enabling Simulation Manager

You can enable simulation manager in ADE Assembler by ensuring that *LSCS* is selected in the *Setup* section of the Job Policy setup form.

1. Select *Use Simulation Manager*.



2. Complete the remaining job policy setup as required.

Note: This simulation manager is not supported in ADE Explorer. While a maestro cellview is open, if you transition from ADE Assembler to ADE Explorer, ADE simulation service is disabled. If you run a simulation from ADE Explorer while another simulation is already in progress in ADE Assembler, the tool displays a message seeking confirmation to stop the simulation.

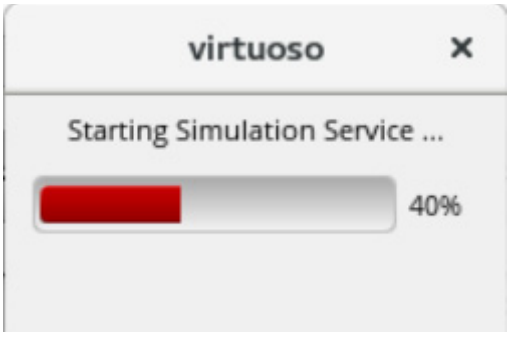
License Requirement

To use this feature, you require the Virtuoso ADE Simulation Manager (95267) license.

© 2023 Cadence Design Systems, Inc.

Managing Simulations with Simulation Manager

When you run the simulation after enabling *Use Simulation Manager*, ADE Assembler automatically initiates the ADE broker and starts the simulation service. In this phase, the following message is displayed in the maestro cellview to indicate the progress of the initialization of simulation manager.



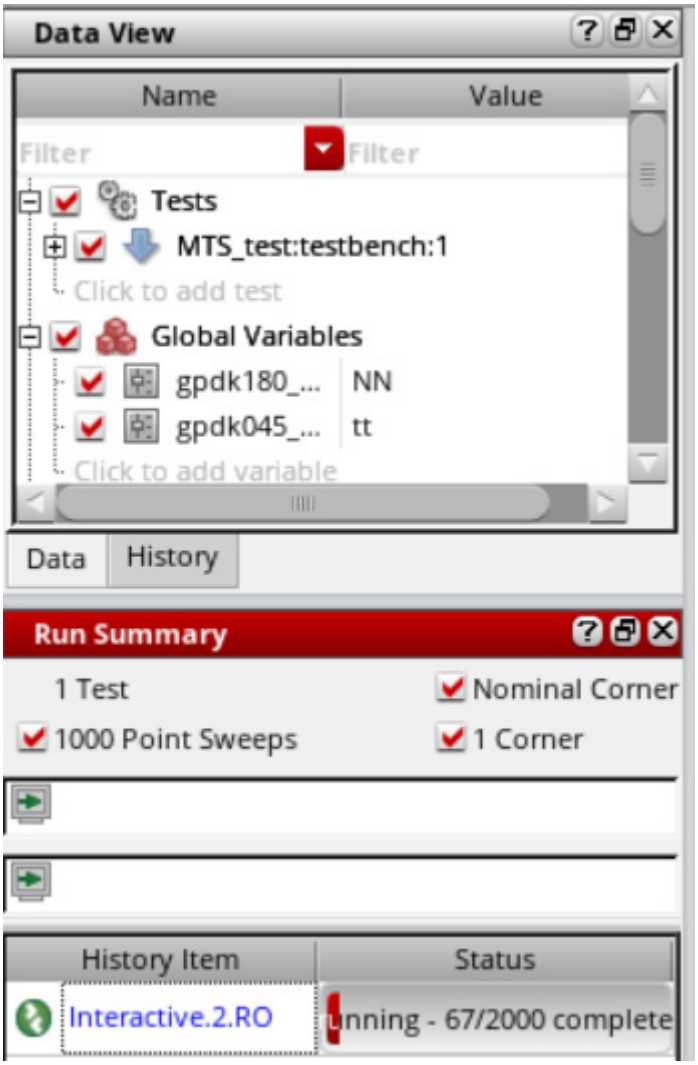
After a connection with the simulation service is established, the simulation is started and a history is created for the run.

The CIW log also shows the status of each step followed to connect to the ADE broker.

```
p
lanager: [2/6] generating job script '
l/SimService1.sh'
lanager: [3/6] starting job, log file at '
/SimService1.stdout'
lanager: [4/6] job script started, waiting for Simulation Manager
lanager: [5/6] Simulation Manager started, trying to connect
lanager: [6/6] connected to '
2:36495'
```


© 2023 Cadence Design Systems, Inc.

In the *Run Summary* assistant, the icon for this history indicates that the simulation is using the simulation manager.



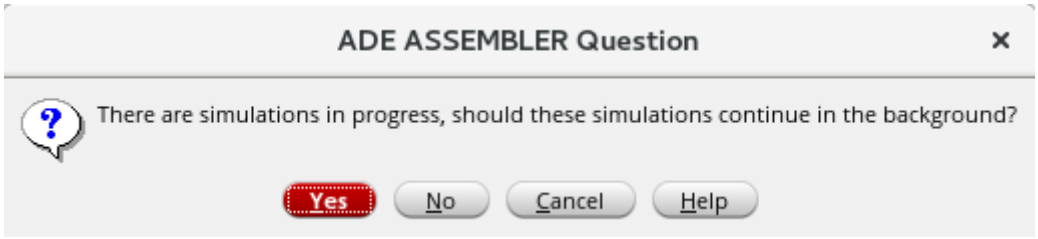
Also note that the history name has a .RO suffix, which indicates that for simulations that use Simulation Manager, the results are saved in read-only histories. These histories are also locked to avoid any accidental modification in another maestro cellview. After the simulation run is complete, the read-only status is removed form the history and it is unlocked.

The simulation continues and the results are saved as usual. While the simulation is running in the background, you can continue to configure the setup or explore results saved in a history.

© 2023 Cadence Design Systems, Inc.

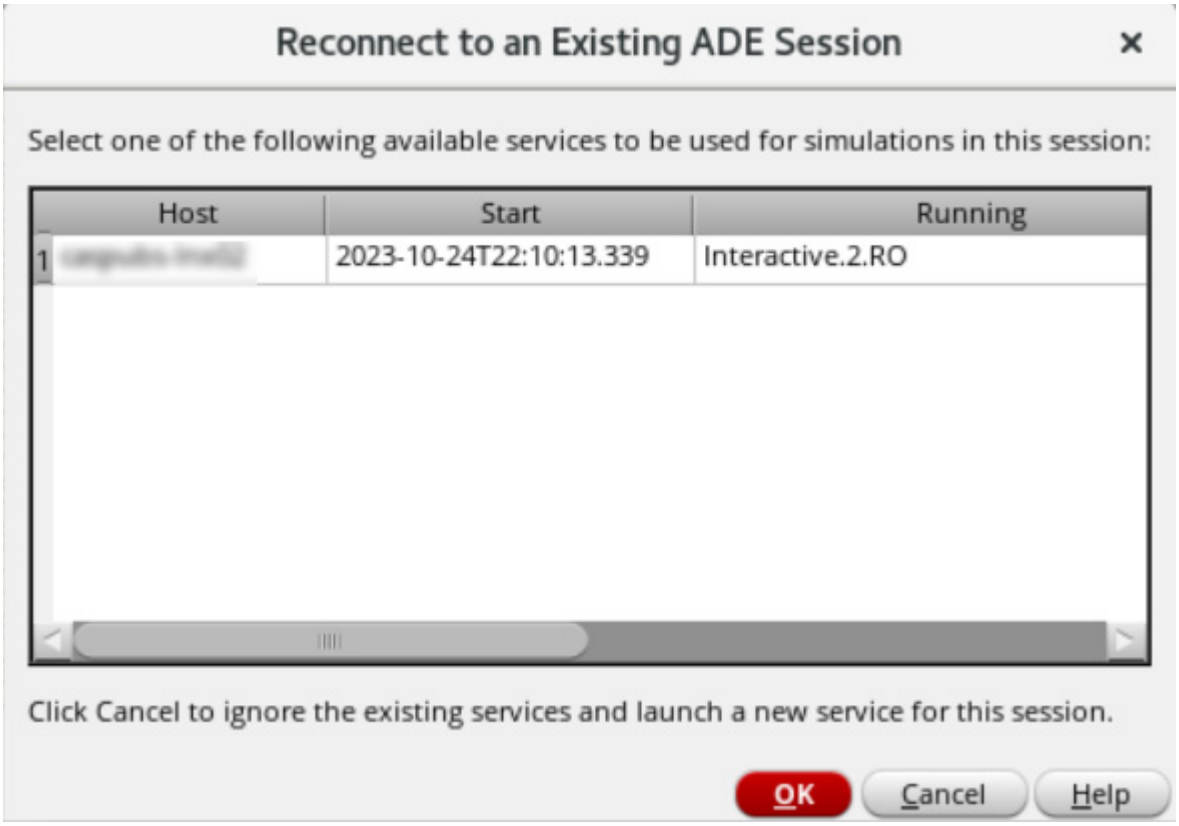
Exiting and Reconnecting with an In-progress Simulation

If you exit the maestro cellview or the Virtuoso session before the simulation is complete, ADE Assembler displays a dialog box, asking you to confirm whether to continue the simulation.



Click *Yes* to continue.

When you reopen the maestro cellview, if the simulation is in progress, Virtuoso displays the Reconnect to an Existing ADE Session form that shows a list of all services available for the cellview.



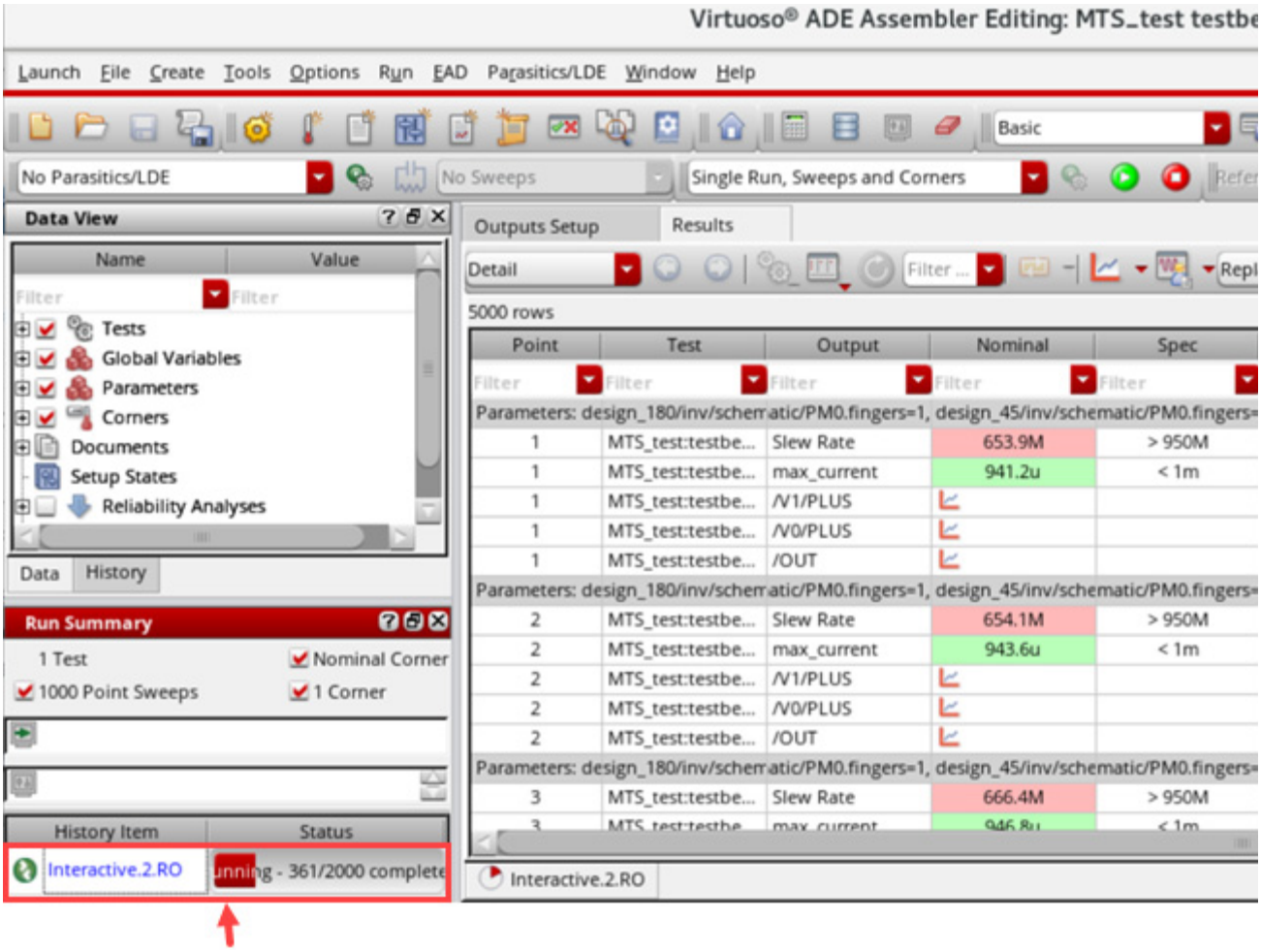
Select a service and click *OK* to reconnect to the same service.

© 2023 Cadence Design Systems, Inc.

The cellview is opened and the history corresponding to the simulation in progress is displayed showing the current status of the run.

If the simulation completes before you reopen the cellview, you can view the results saved in the history.

The following example shows a simulation run that was in progress in the previous maestro session.

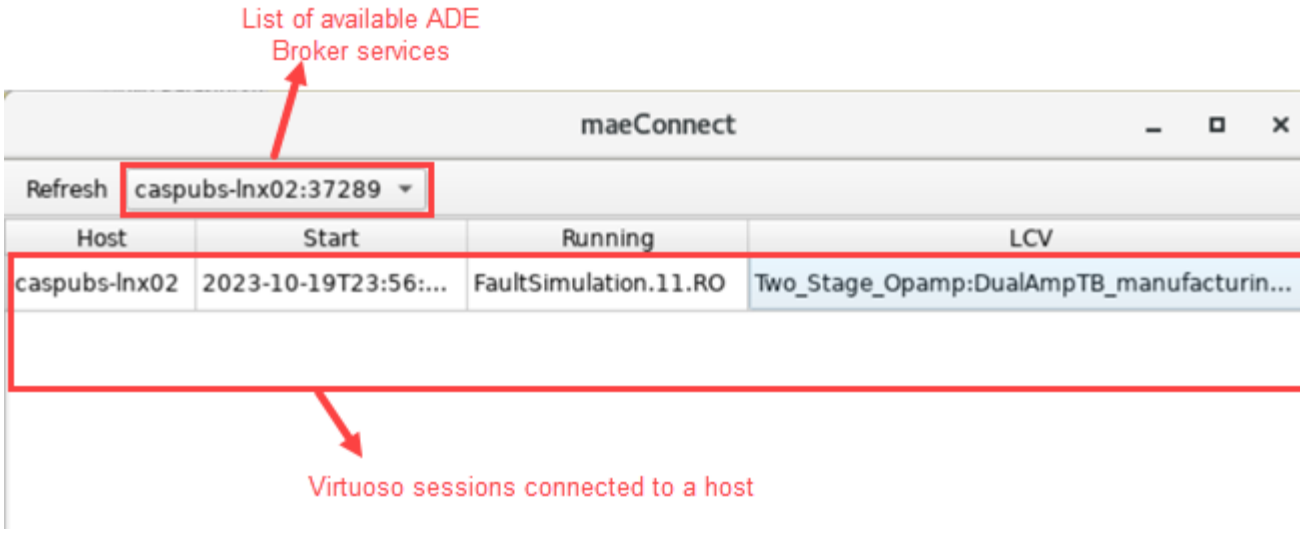


When you reopen the cellview, the same state is maintained for the run. You can suspend, resume, stop, or manage it as required.

© 2023 Cadence Design Systems, Inc.

Explicitly Opening a Virtuoso Session Connected to a Simulation Manager Service

You can use the `maeConnect` command to display all available ADE broker services on the host.



Select an ADE Broker service from the drop-down list of service names. A list of Virtuoso sessions connected to the broker service are listed in the table.

Right-click a row and choose *Open GUI* to launch the Virtuoso session on the same host on which the Simulation Manager service is running and to open the maestro session.

If a particular simulation is not required to be completed or is stuck due to some network issues, right-click and choose *Kill Job* to stop the simulation and close the Virtuoso session.

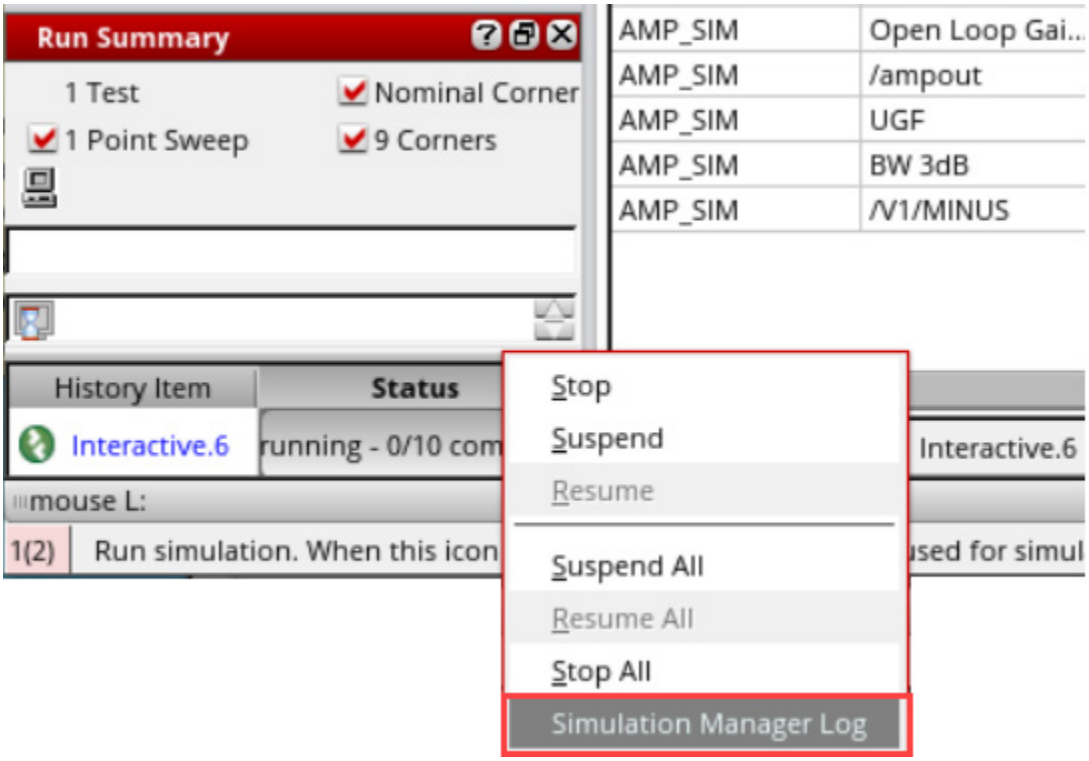
Debugging Simulation Manager

While debugging simulations that use simulation manager, you can check the log of simulation manager in addition to the other logs files for netlist, simulation, or expression evaluation.

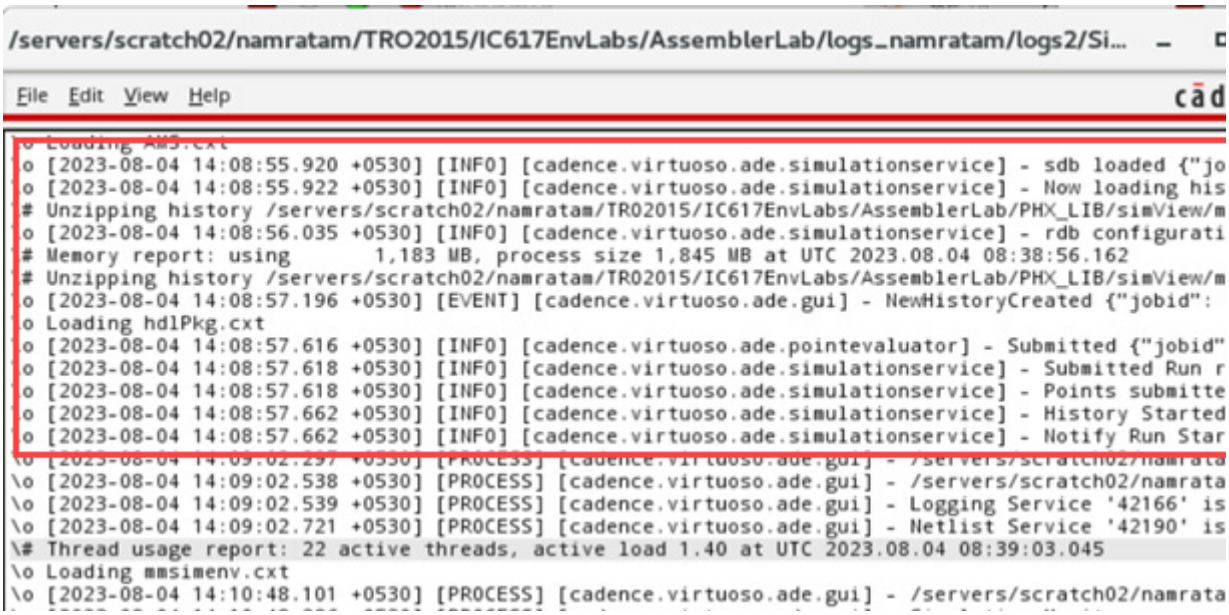
To open the simulation manager log:

© 2023 Cadence Design Systems, Inc.

1. Right-click the history that uses the simulation manager log and choose *Simulation Manager Log*.



2. ADE Assembler opens the `SimServicen.log` file saved for the maestro cellview, as shown below.



© 2023 Cadence Design Systems, Inc.

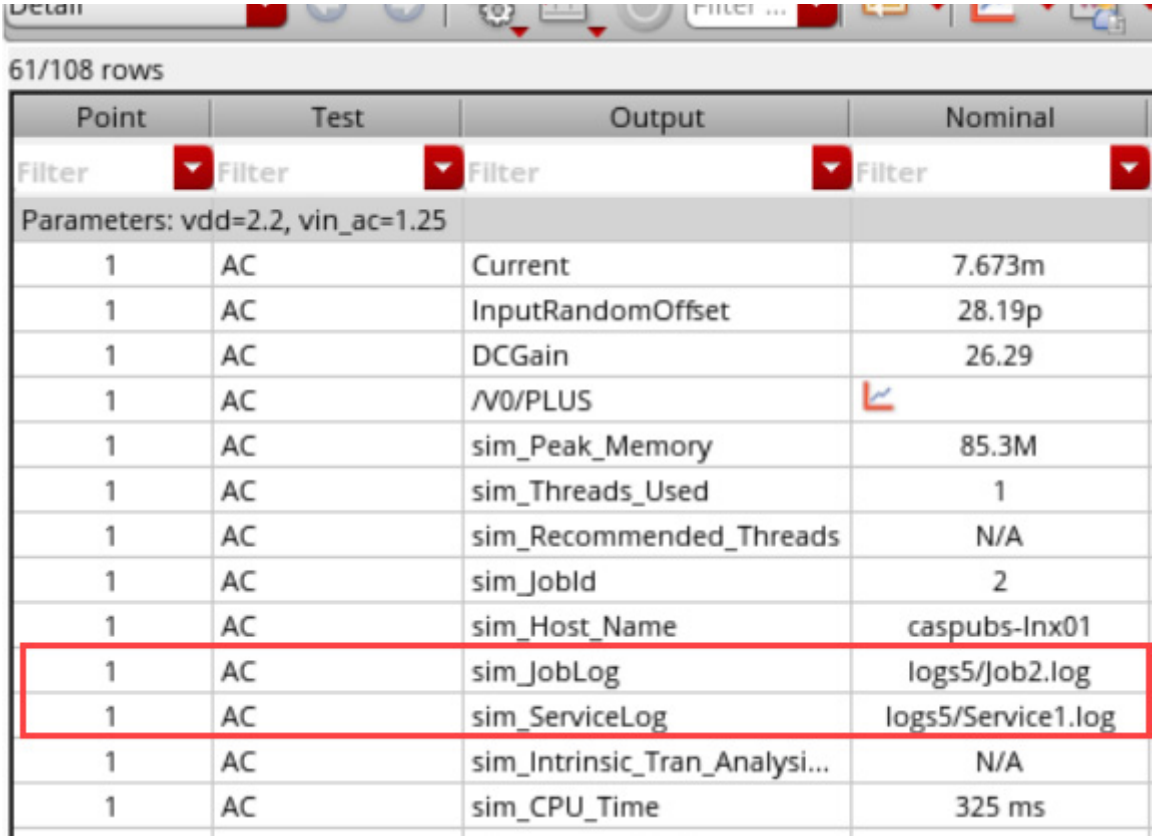
Debugging Points with Simulation Manager

To enable debugging of points in a simulation run using Simulation Manager, you can display the names of log files in the statistical information for each point.

Set the following environment variables:

```
(envSetVal "maestro.results" "defaultRunStatisticItems" 'string
"memory,threads,recommendedThreads,jobId,host,jobLog,serviceLog,tranIntrinsicElaps
edTime,cpuTime,cpuUtil")

(envSetVal "maestro.results" "enableRunStatisticInfoDisplay" 'boolean t)
```

When these variables are set, the statistical information on the *Results* tab shows the names of corresponding log files that you can check while debugging a point.



61/108 rows

Point	Test	Output	Nominal
Filter	Filter	Filter	Filter
Parameters: vdd=2.2, vin_ac=1.25			
1	AC	Current	7.673m
1	AC	InputRandomOffset	28.19p
1	AC	DCGain	26.29
1	AC	/V0/PLUS	
1	AC	sim_Peak_Memory	85.3M
1	AC	sim_Threads_Used	1
1	AC	sim_Recommended_Threads	N/A
1	AC	sim_JobId	2
1	AC	sim_Host_Name	caspubs-lnx01
1	AC	sim_JobLog	logs5/Job2.log
1	AC	sim_ServiceLog	logs5/Service1.log
1	AC	sim_Intrinsic_Tran_Analysi...	N/A
1	AC	sim_CPU_Time	325 ms

Related Topic

Viewing Log Files for a Data Point

Viewing a Consolidated Log for a History

© 2023 Cadence Design Systems, Inc.

Setting Up Job Policies

A job policy specifies the job control mode to be used to run simulations. In addition, it specifies the job distribution method: local, remote, or command; maximum number of jobs to be run at a time; and various job timeout settings.

The default job policy for ADE Assembler is `Maestro Default`. You can also define a custom job policy, save it with a unique name in the `.cadence/jobpolicy` directory and set it as default by using the defaultJobPolicy environment variable.

The order in which the job policy files are searched in the `.cadence/jobpolicy` directory is determined by the Cadence Setup Search File mechanism (CSF). To find this information, CSF uses the `setup.loc` file, which is an ASCII file that specifies the locations to be searched and the order in which they should be searched. For more information about the setup.loc file or how to edit search order, see Cadence Setup Search File: setup.loc in the *Cadence Application Infrastructure User Guide*.

A job policy can be defined for:

- A single test. For more details, refer to Setting Up a Job Policy for a Test.
- A global job policy that is applied to all the tests that do not have their own job policies.

A job policy can be defined by using the Job Policy Setup form. For more details, refer to Setting Up the Default Job Policy.

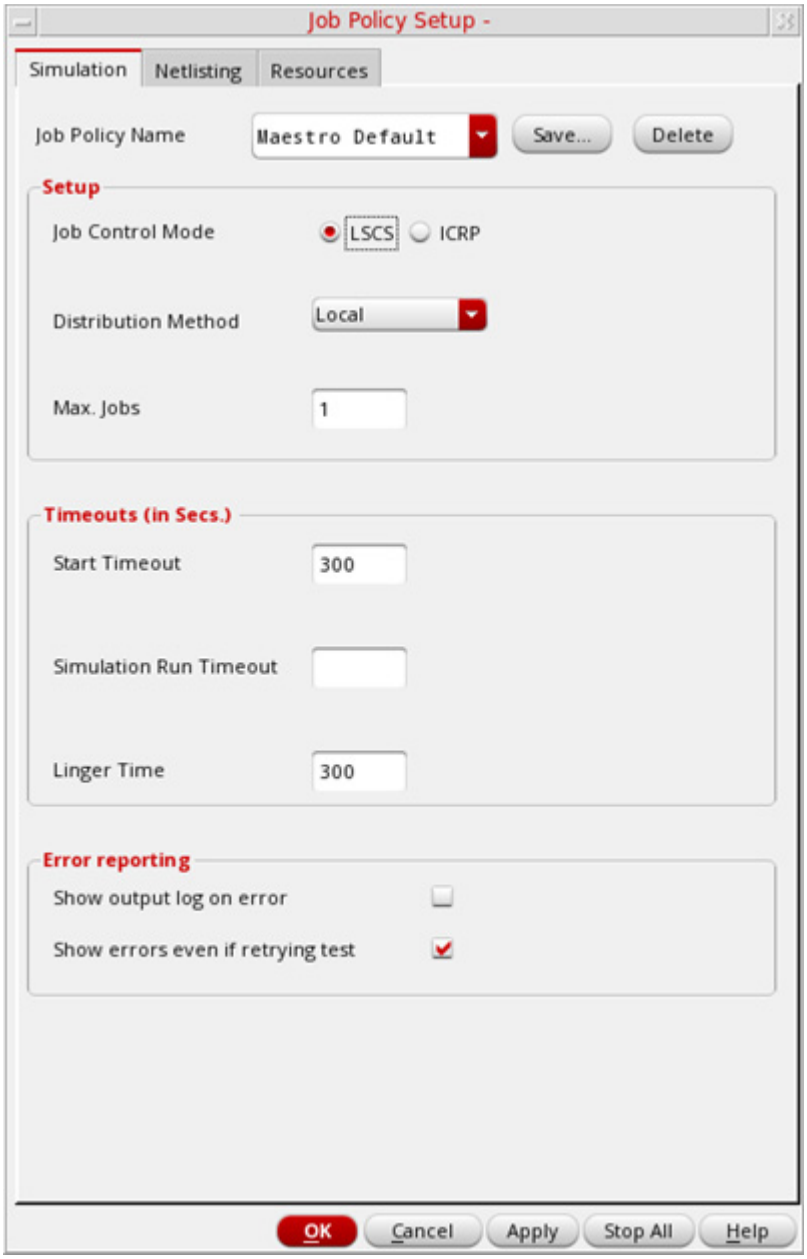
While running a simulation, ADE Assembler looks for a job policy in the order given below and applies the policy that is found first:

- A test-specific job policy is applied to the test by using the *Job Setup* context-sensitive menu command for that test.
- The default job policy specified by using the defaultJobPolicy environment variable in the `.csdenv` file.
- If no custom job policy setting is found, the `Maestro Default` job Policy is used.

© 2023 Cadence Design Systems, Inc.

Viewing the Global Job Policy

To view or modify the global job policy, choose the *Options — Job Setup* command in ADE Assembler.



Note: The *Netlisting* tab is visible only in the *LSCS* job control mode.

© 2023 Cadence Design Systems, Inc.

Setting Up a Job Policy for a Test

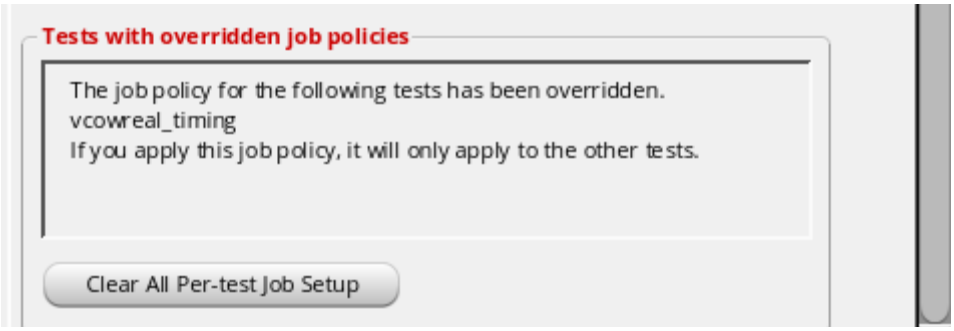
You can set up a job policy for each test. This enables you to use different distributed processing resources or queues for different tests.

➥ To set up a job policy for a test, on the Data View pane, right-click a test and choose *Job Setup*. The Job Policy Setup - <test name> form is displayed. Modify the settings to suit your requirements and save the setup with a new policy name.

➥ To remove the job setup for a test and apply global job policy for it, on the Data View pane, right-click a test and choose *Clear Job Setup*.

Important Points Related to Job Policies

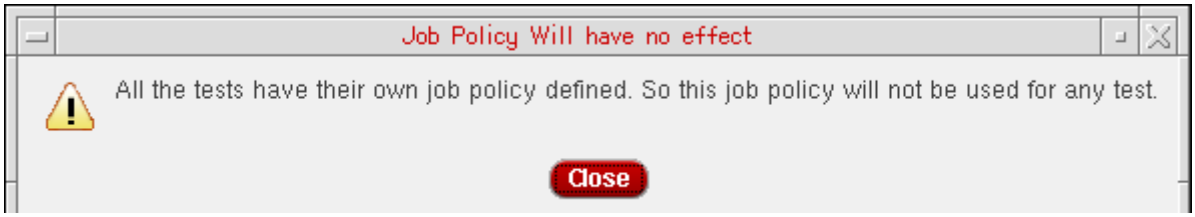
- Name of a test-specific job policy is shown in the test tooltip.
- If you want to use a custom job policy in every new maestro view, set the defaultJobPolicy environment variable to the name of your job policy. If the default policy is not required for a specific view, you can change it later.
- The job policy information is saved in the active setup database of a cellview and is applied in every subsequent use.
- The job policy specified for a test overrides the global job policy specified using the *Options – Job Setup* commands. Information about the tests for which job policy is defined is shown in the global Job Policy Setup form, as shown below.



- To use the global job policy for all tests in the setup and avoid any overrides, click *Clear All Per-test Job Setup*. The local job policies defined for each test is cleared.

© 2023 Cadence Design Systems, Inc.

- If all the tests have test-specific job policy, the global job policy specified through the *Options – Job Setup* command does not have any impact. If you specify a global job policy in this case, a message is displayed stating that the job policy will have no effect.



- If some tests are running with test-specific job policies whereas other are running with global job policy, the *Options – Job Setup – Stop All* command stops the simulations for only those tests that are using the global job policy.
- For every test-specific job policy, a separate ICRP process is started and that is not used for any other test.
- If you change the job policy for a test while an ICRP process was running for that test, ADE Assembler completes the simulation for the running point. After that, the job is terminated and a new ICRP process is started according to the new job policy.
- If you make changes to the job policy for a test, such as change the job policy name or replace the job policy, while an LSCS process was running for that test, ADE Assembler completes the simulation for the running point and terminates the job. Then, all jobs from that test that use this job policy are terminated and new jobs are restarted according to the modified job policy.
- If you modify the *Max Jobs* value in the Job Policy Setup form, ADE Assembler completes the simulation for the running point before the job is terminated, and then rectifies the number of jobs by launching more jobs or terminating extra jobs.

  You can use this functionality to terminate lengthy jobs and run simulation for critical jobs by freeing up the resources. If you have multiple jobs in the queue, you can modify the Max Jobs value and then terminate any unnecessary jobs from the Run Summary assistant.

Setting Up the Default Job Policy

To specify a customized default job policy:

1. In the Job Policy Setup Form:

   a. Specify a job policy name. For example, `myDefaultJobPolicy`.

© 2023 Cadence Design Systems, Inc.

b. Change the values for the fields *Distribution Method*, *Max. Jobs*, and *Linger Time*.

2. Click the *Save* button next to the *Job Policy Name* field.

   The Save Job Policy form is displayed.

3. In this form, select the path for the job policy.
4. Click *OK*.
5. Set this job policy name as the value for the following variable:

   ```
   envSetVal( "adexl.icrpStartup" "defaultJobPolicy" 'string
   "myDefaultJobPolicy" )
   ```

   Note: By default, the job control mode is set to LSCS. Set the following variable to specify the default job policy for ICRP job control mode:

   ```
   envSetVal("maestro.distribute" "forceJobControlModeWhenViewOpen"
   'cyclic "ICRP")
   ```

If you do not specify a job policy, the program applies the following defaults.

- Distribution Method: `Local`
- Max. Jobs: `1`
- Start Timeout: `300 (seconds)`
- Linger Time: `300 (seconds)`
- Show errors even if retrying test: `t`
- Optimize Single Point Run: `t`
- Communication Timeout: `180 (seconds)`
- Do not estimate: `t`
- In the *Default Value* group box of the *Resources* tab:
  - Memory: `1000 MB`
  - CPU: `1 unit(s)`
- In the *Scale Estimated Values* by group box:
  - Memory: `100%`
  - CPU: `100%`

© 2023 Cadence Design Systems, Inc.

Specifying a Job Policy Name

To specify a job policy name, do the following:

1. Choose *Options – Job Setup*.

   The Job Policy Setup form appears.

2. In the *Job Policy Name* field, type a name for your job policy.

   The job policy name must be alphanumeric with no spaces or special characters.

You can specify the default job policy name by setting the defaultJobPolicy environment variable.

Optimizing a Single Point Run

The *Optimize Single Point Run* check box selected by default in the *Setup* section of the Job policy Setup form. ADE Assembler optimizes the run according to the specified job distribution method in the job policy.

- ICRP: Runs the netlist generation, expression evaluation, and simulation tasks inside the ADE Assembler process.
- LSCS: Runs the netlist generation task in the ADE Assembler process. Expression evaluation and simulation are run in a remote process.

This saves the overhead of starting a separate process, thereby reducing the overall processing time.

© 2023 Cadence Design Systems, Inc.

Important

While the netlist generation for a single point run is being done inside the ADE Assembler process, the user interface becomes unavailable for any changes. Therefore, it is recommended to clear the *Optimize Single Point Run* check box for designs and tests that require a large time to generate netlist.

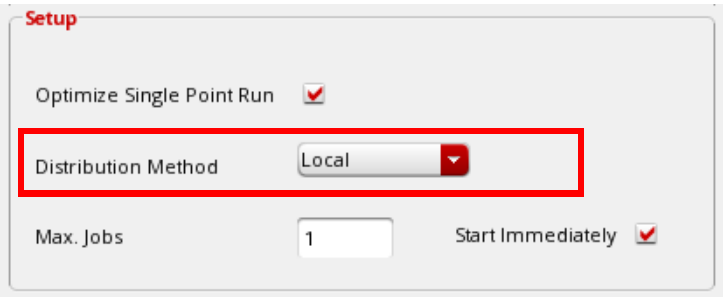
This feature is not supported when:

- Simulating a single point for the EAD flow or for Monte Carlo sampling.
- The distribution method is set to *Command* and the command is non-interactive. An interactive command waits for the job to complete before returning control to the terminal.

Related environment variable: useSameProcess

Specifying a Distribution Method

You can specify a distribution method by using the *Distribution Method* drop-down list.



Depending on your setup, you can choose one of the following methods:

- Local – Runs simulations on the local host.
- Remote-Host – Runs simulations on the remote host that you specify.

  Note: On selecting *Remote-Host* as the distribution method, the *Host* field is displayed. In this field, specify the name of a valid remote host.
- Command – Sends a command to the distributed processing software.

  Note: On selecting, *Command* as the distribution method, the *Command* field is displayed. In this field, specify the command you want to use to start jobs.The command must adhere to these guidelines.

© 2023 Cadence Design Systems, Inc.

Click *Dry Run* to check the network health before running a simulation. To run this check, ADE Assembler submits the number of jobs specified in the *Max. Jobs* field of the Job Policy Setup form without actually running the processes. The maximum limit for the number of jobs that can be submitted is 100.

- *LBS* – Uses Cadence Load Balancing Software.

Note: Both LBS and Command distribution method submit ADE Assembler job to a job distribution engine, for example Sun Grid Engine (SGE). In LBS mode, you can specify a selected set of qsub options whereas with the Command method you can submit a command with any qsub option.

The process owner should be the same user who is running simulations from ADE Assembler.

If the specified job distribution method is invalid, ADE Assembler checks whether a default distribution method is returned by the `maeGetDefaultDistributionMethod` custom SKILL function. You can define this function in the `.cdsinit` file as shown in the example below.

```
procedure(maeGetDefaultDistributionMethod()
"Command"
)
```

Specifying Max Jobs

The *Max Jobs* field specifies the maximum number of jobs that can run in an ADE Assembler session. This value along with the value specified in the *Specify* field on the Run Options form is used by ADE Assembler to assign jobs to the simulations running in a session.

To specify the maximum number of jobs that can run, do the following:

- In the *Max. Jobs* field, type the maximum number of jobs that can run at any time during your ADE Assembler session.

© 2023 Cadence Design Systems, Inc.

Note: If the distribution method specified in your job policy is *Command*, *Local* or *Remote-Host*, the maximum number of jobs that can run at any time cannot exceed the value specified using the maxIPCJobsLimit environment variable. The default value of this environment variable is `100`.

When *Start Immediately* is turned on, the program immediately submits the specified maximum number of jobs, or one job for every test, whichever is less, when you start ADE Assembler. This means that the defined number of processes are already running before you start the simulation. This saves the overhead of staring processes at time of running simulation.

To turn off this feature, do the following:

- Deselect the *Start Immediately* check box.

Important Points to Note

- Even if already started, the life of an idle process is controlled by the *Linger Time* field whose default value is `300` second. This implies that if a user launches a simulation 5 minutes after starting ADE Assembler, these processes would have already been killed and would restart when a simulation is run.
- The *Start Immediately* check box is visible only for the ICRP mode.
- In a scenario where the resource requirements change dynamically, it is recommended not to select the *Start Immediately* check box. The initial process will start based on the previous resource requirement in the job setup, but since the resource requirement change for every new run, the previously started process may not fulfill the new requirements. Hence, the older processes need to be killed and new processes are launched on the computer that meets the resource requirement.

Specifying a Command for DRMS

Consider the following guidelines when creating a command for DRMS:

- It is expected that the process owner is the same user that is running simulations from ADE Assembler.
- When the software submits a job, it runs a separate process. The software prepends the command you type in the *Command* field on the Job Policy Setup form to the job startup command. The program runs the entire command on the local host.

© 2023 Cadence Design Systems, Inc.

Important

The command you type must take the job startup command as its argument.

For example, you might type a DRMS command such as `bsub` for LSF or `qsub` for Sun Grid Engine (SGE) in the *Command* field:

```
bsub -I -q queueName other_bsub_options
```

The program puts the process startup command at the end of the command you type as follows:

```
bsub -I -q queueName other_bsub_options processStartupCommand
```

You are responsible for making sure that this command is valid, that the `bsub` command you type can start a process on some host in the cluster using the specified options.

Tip

You can try customizing your job submissions by developing wrapper scripts and using them with a command job policy to submit your jobs. The program will not validate the correctness of your wrappers/scripts.

- It is always recommended to submit jobs in interactive mode to provide more control to ADE Assembler. The interactive mode provides the following two benefits:
  - ADE Assembler can find the job health status
  - ADE Assembler can provide an exit method to the job

  It is recommended to submit Command jobs as interactive mode jobs.

  The status of the job can be inferred from the local shell "proxy" process that exists in interactive mode. ADE Assembler can also send `SIGTERM`, an interrupt, to the DRMS "proxy" process running on the local host, which is then passed to the remote process. If required, ADE Assembler can also force a job to terminate by using the proxy process.

  Depending on the DRMS you are using, you can use the following commands to submit an interactive job:

  LSF: `bsub with -K, -I, or -Is`

  SGE: `qsh, qlogin, or qrsh`

  Network Computer: `nc run with -I, -Ir, or -Il`

  Note: The `-K` and `-I` options are not compatible with the continueICRPRunOnAbruptGUIExit environment variable. When the variable is set to `t` and the command mode uses `-I` or `-K`, the ICRPs will still be interrupted when the parent process dies. If you want to use continueICRPRunOnAbruptGUIExit avoid using the interactive job mode.

© 2023 Cadence Design Systems, Inc.

A few scenarios where interactive jobs are useful are given below.

Example 1: You run a simulation in batch mode. After some time, you stop the simulation while the job is pending before it got a chance to start. The job will remain pending in the DRMS system until it starts. The job will exit automatically when the DRMS starts. If you would have used the interactive mode instead, the job would have been killed by ADE Assembler immediately when stop the simulation.

Example 2: You run a simulation in batch mode, but you have specified a bad resource requirement (one that cannot be met). You then stop the simulation while the job is pending to correct the resource string. The job will remain pending in the DRMS system unless you kill it manually. If you would have used interactive mode instead, the job would have been automatically killed by ADE Assembler.

Example 3: If the process hangs for some reason, ADE Assembler can force termination when the command job is submitted as an interactive job.

Example 4: Jobs were submitted to the DRMS, but got killed somehow while they were in the pending state. In normal mode, ADE Assembler will keep waiting for these jobs to start until the start timeout has been reached. However, in the interactive mode, ADE Assembler can detect the status and re-submit the job rather than waiting for the start timeout.

Related environment variable: inferCommandICRPStatusFromProxy

Specifying an LBS Job Policy

ADE Assembler supports the following Distributed Resource Management Systems (DRMS):

LBS	Load Balancing System – Simple job distribution system (`cdsqmgr`) for setting up queues (collections of host computers) and hosts per queue; each queue has a job-per-host limit
LSF	Load Sharing Facility – DRMS from Platform Computing
SGE	Sun Grid Engine – Freeware from Sun Microsystems, Inc.
OpenLava	OpenLava workload management system

To specify a load balancing system job policy, do the following:

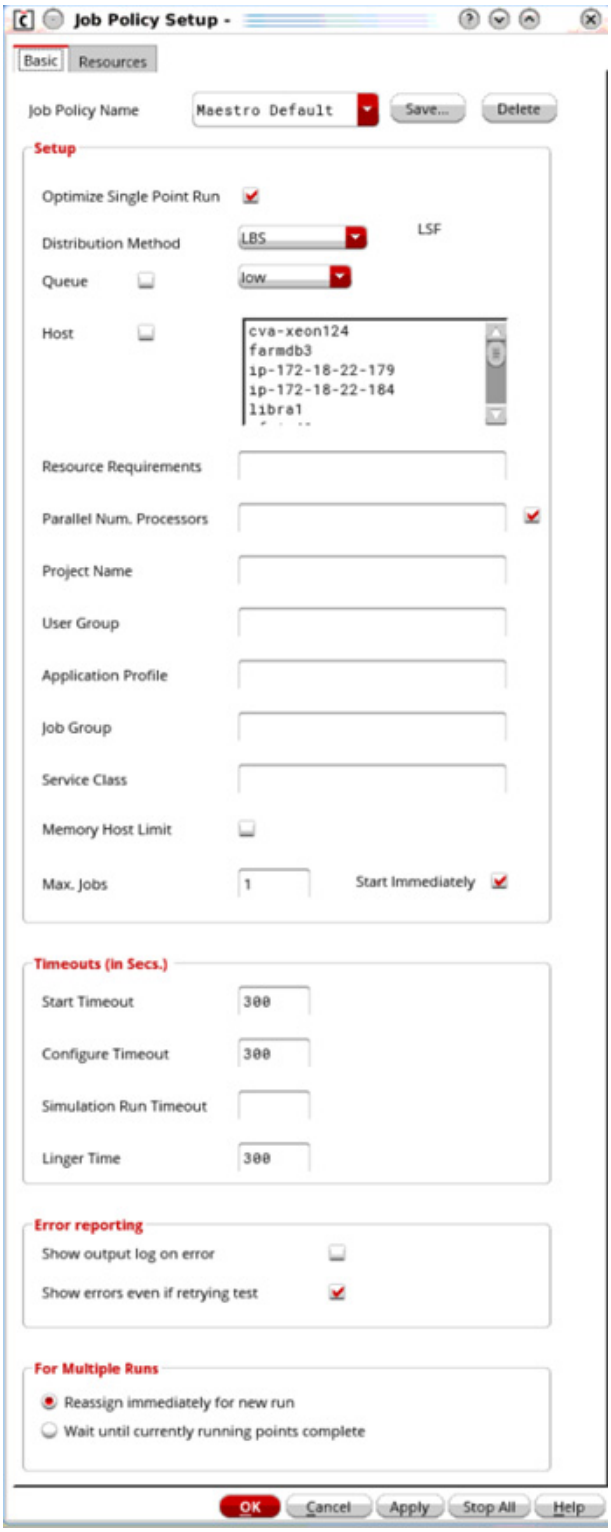
1. Specify a name for your job policy.
2. In the *Distribution Method* drop-down list, select *LBS*.

© 2023 Cadence Design Systems, Inc.

Check boxes, fields, and list boxes related to specifying queues and hosts appear.

© 2023 Cadence Design Systems, Inc.

If you have set up the LBS DRMS, the form shows additional fields.



© 2023 Cadence Design Systems, Inc.

If you have set up the SGE DRMS, the form shows the fields relevant to SGE.

1. (For LSF and OpenLava only) Select the *Queue* check box and select an available queue from the drop-down list.

   Note: Your system administrator determines the list of available queues. See System Administrator Information.

   If you do not select the *Queue* check box, the program uses default system queue.

2. (For LSF and OpenLava only) Select the *Host* check box and select an available host from the list area.

   Note: The list of hosts corresponds to those available in the selected queue (see the previous step).If you do not select the *Host* check box, your DRMS decides which host to use.

3. (For LSF and OpenLava only) In the *Resource Requirements* field, specify any additional resource requirements to submit the job.

   The program uses these resource requirements along with the queue/host you specified to start the job. It is up to the load sharing facility software to resolve any conflicts between the queue/host and the resource requirements.

4. (For LSF and OpenLava only) In the *Parallel Num. Processors* field, specify the number of parallel processors to be used to run a simulation. To use this value, select the check box given to the right of this check box.

5. (For LSF and OpenLava) In the *Job Group* field, specify the name of the job group to which you need to submit the job.

6. (For LSF only) In the *Service Class* field, specify the name of the service class to which you need to submit the job.

7. (For LSF only) Select the *Memory Host Limit* check box to apply the memory limit set for the selected LSF queue in the configuration file for LSF.

8. (For SGE only) In the *Hard Resource Req* field, specify the resources that must be allocated before a job can be started.

   For example, `num_proc=4,mem_total=4G`

9. (For SGE only) In the *Soft Resource Req* field, specify the resources that a job needs but do not have to be allocated before the job can be started. The specified resources will be allocated to the job on an as-available basis.

   For example, `mem_free=2G`

© 2023 Cadence Design Systems, Inc.

10. (For SGE and OpenLava only) In the *Job Priority* field, specify the priority for the job being submitted relative to other pending jobs submitted by you. Default priority of a submitted job is `0`. Users with administrator privileges can set this value from `-1023` to `1024`. Other users can set this value from `-1023` to `0`.

For example, you can set priority for a job as `-500`.

11. (For SGE only) In the *Parallel Environment* field, specify the name of the parallel environment.

12. In the *Parallel Num Processors* field, specify the number of parallel processors to be used to run the submitted job.

13. (Optional) Specify maximum job information.

14. (Optional) Specify job timeouts.

15. (Optional) Specify error reporting options.

16. (Optional) Specify multiple run options.

17. Save your job policy.

For more information, see Saving or Deleting a Job Policy.

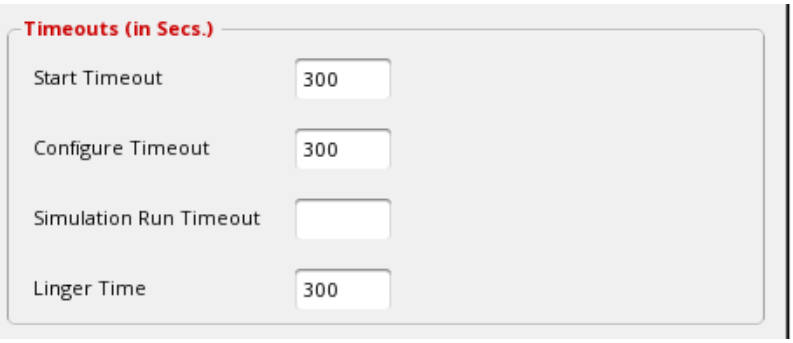
18. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Specifying Job Timeouts for ICRP

To customize job timeout options, specify values in the fields of in the *Timeouts* group box of the *Simulation* tab of the Job Policy Setup form.

All the values in this section are specified in seconds. When a timeout value is set to `0`, -1, or is left blank, the timeout is disabled. This leads to an unlimited wait time.



The following table describes the fields in the *Timeouts* section:

Field	Description
Start Timeout	Specifies the time (in seconds) for which the tool should wait for the ICRP process to report back to ADE Assembler that it has started in an ICRP job. The wait time starts as soon as ADE Assembler submits the job. If no response is received in the given time, ADE Assembler considers the job not started and kills it. If there is another job waiting to start, ADE Assembler starts it. **Note:** If you are using LBS or an interactive bsub or qsub command for job distribution and the jobs take longer than this timeout value to start, ADE Assembler might consider the job as failed. To avoid this, you can set the start timeout to blank or to a larger value, for example, 100000.

© 2023 Cadence Design Systems, Inc.

Configure Timeout	Specifies the time (in seconds) for which the tool should wait for the process to report back to ADE Assembler that it has configured the job. The wait time starts as soon as ADE Assembler sends the job configure command. If the process does not respond with a successful configuration message in the given time, ADE Assembler considers the job configuration unsuccessful and kills it. ADE Assembler looks for the next idle job and tries to start and configure it. **Note:** This option is available only for the ICRP mode.
Simulation Run Timeout	Specifies the number of seconds to wait for the process to report back to ADE Assembler that it has run the job. The wait time starts as soon as ADE Assembler sends the run command for the job. If the process does not respond that the job has completed in the number of seconds you specify, ADE Assembler considers the job run unsuccessful and kills it. ADE Assembler looks for the next idle job and tries to start, configure, and run it.
Linger Time	Specifies the number of seconds to wait for the process to kill a remote job after the simulation is complete. For LSCS job control mode, this linger time value is used by both processes, Simulation monitor and expression evaluator.
Communication Timeout	For ICRP job control mode, specifies the number of seconds to wait for the process to report back to ADE Assembler that the Message Passing System (MPS) communication has been successfully established.

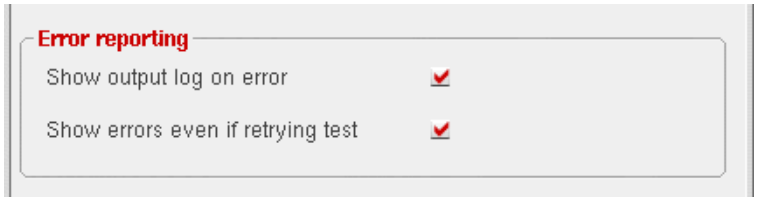
Related Topics

Troubleshooting Jobs that Stay Pending until Timeout

© 2023 Cadence Design Systems, Inc.

Specifying Error Reporting Options

To customize error reporting, you need to specify the options in the *Error Reporting* group box of the *Simulation* tab of the Job Policy Setup form.



1. To view the output log file of the remaining error points, right-click the job status icon on the Run Summary assistant pane.

   By default, if multiple error points exist for a test, the program displays only the output log file of the first error point in the test.

2. To cause the program to display the output log files of all error points in the test, select the *Show output log on error* check box.

To cause the program to display the output log file on the occurrence of an error for a test, even if the ADE Assembler distribution system is retrying the test, select the *Show errors even if retrying test* check box.

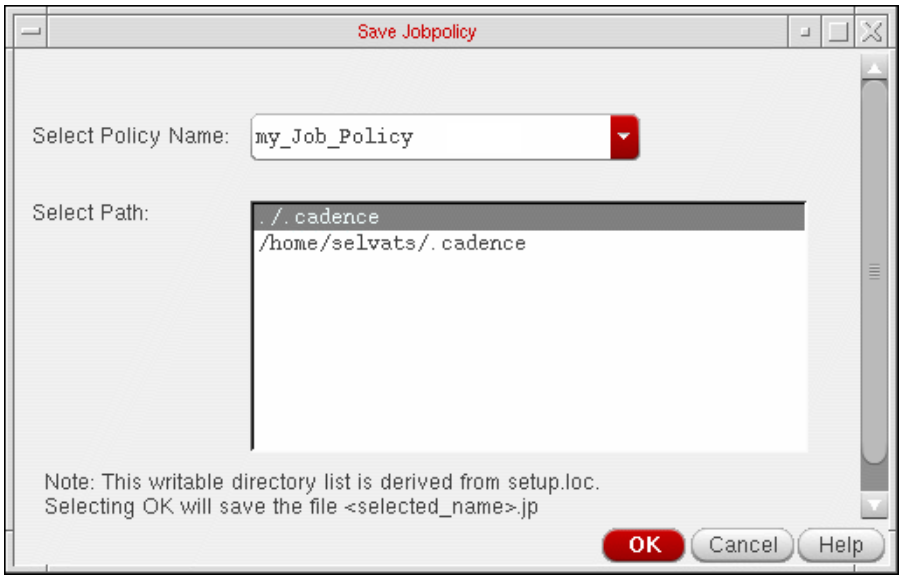
Saving or Deleting a Job Policy

To save a job policy, do the following:

1. In the Job Policy Setup form, click *Save*.

© 2023 Cadence Design Systems, Inc.

The Save Job Policy form appears.



2. In the *Select Path* list, select the directory where you want to save the job policy.

   You can save the job policy in one of the following directories in which you have write permissions or in the paths specified in your `setup.loc` file where you have write permissions:

   - `.cadence` directory in the current directory
   - The `.cadence` directory in the path specified in the `CDS_WORKAREA` environment variable.
   - `$HOME/.cadence` (the `.cadence` directory in your home directory)
   - The `.cadence` directory in the path specified in the `CDS_PROJECT` environment variable.
   - The `.cadence` directory in the path specified in the `CDS_SITE` environment variable.

3. Click *OK*.

   The job policy is saved in the `jobpolicy` directory under the selected directory. The job policy file has the `.jp` extension.

If a job policy file with the same name is found in more than one of the above locations, the first job policy file found in the first of the following locations is used:

- `.cadence` directory in the current directory

© 2023 Cadence Design Systems, Inc.

- The `.cadence` directory in the path specified in the `CDS_WORKAREA` environment variable.
- `$HOME/.cadence` (the `.cadence` directory in your home directory)
- The `.cadence` directory in the path specified in the `CDS_PROJECT` environment variable.
- The `.cadence` directory in the path specified in the `CDS_SITE` environment variable.

To delete a job policy, do the following:

1. Choose *Options – Job Setup*.

   The Job Policy Setup form appears.
2. From the *Job Policy Name* drop-down list, select the job policy you want to delete.
3. Click *Delete*.

© 2023 Cadence Design Systems, Inc.

Running Simulations with LSCS Job Control Mode

Large-Scale Cloud Simulation (LSCS) is a highly scalable job control mode that can handle thousands of netlisting and simulation jobs in parallel. It is a new job control mode that is available in addition to the existing IC Remote Processes (ICRP) mode. LSCS implements a new architecture to run simulations with better resource utilization. It also enables future scalability requirements, mainly cloud simulations.

How is LSCS Different from ICRP?

To understand the benefits provided by LSCS, it would help to first understand the difference in the underlying architecture of the two job control modes, as shown in the following figure:

In the ICRP mode, the netlisting, expression evaluation, and communication with the main GUI process are handled by a single Virtuoso process. It makes the system non-scalable for a large number of jobs because Virtuoso blocks a lot of resources even after netlisting is done and the simulation process is running independently.

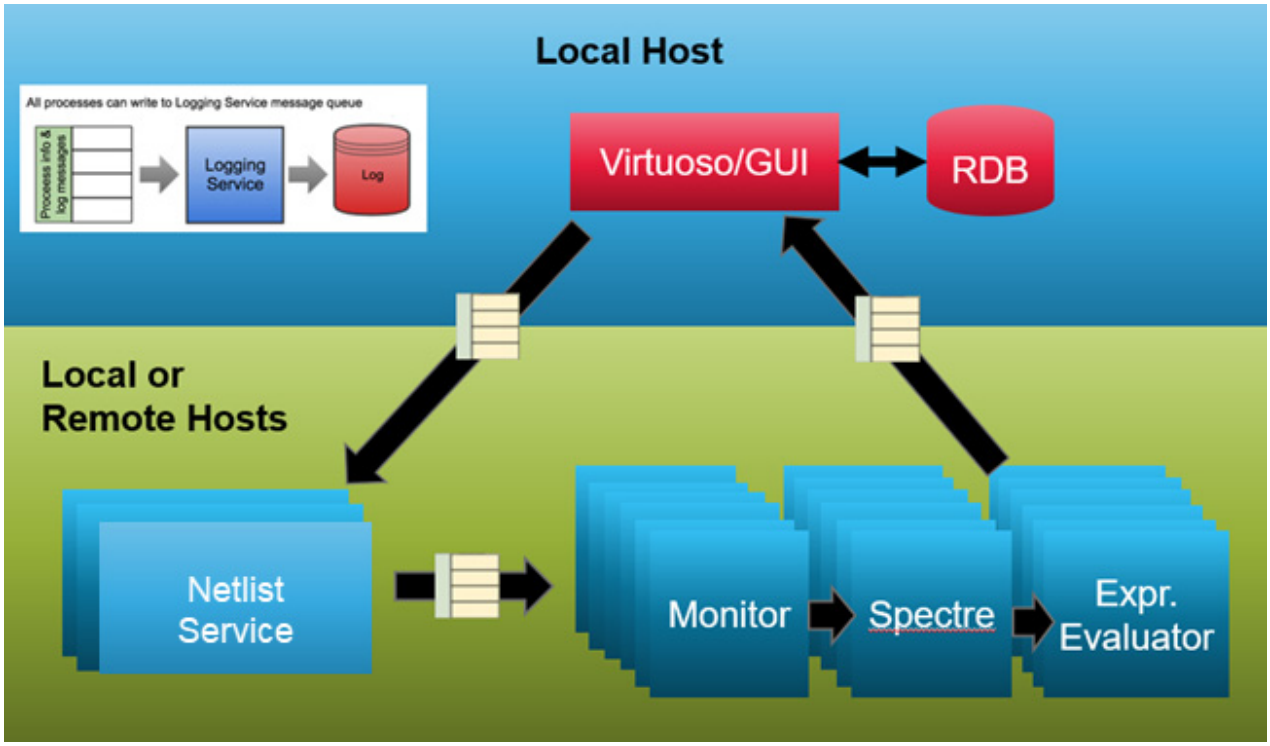
Even though netlisting can be done in the main GUI process using the `ignoreDesignChangesDuringRun` environment variable, it makes the GUI unresponsive until netlisting is completed. Thus, ICRP job control mode is inefficient for very large designs.

© 2023 Cadence Design Systems, Inc.

LSCS Architecture

In LSCS mode, three systems, Virtuoso, Netlist Service, and Simulation Monitor, interact with each other to control different parts of a simulation run. These systems work independently and concurrently with a set of shared queues of messages.

The workflow is divided into separate processes that consume resources only when they are needed. Therefore, this model enables a more efficient exchange of information between these distinct processes.The following figure describes the architecture of LSCS job control mode in detail.



The LSCS job control mode is driven by message queues, which work on producer-consumer model. It is a decoupled structure in which a producer process places a task in a message queue. The consumer process processes the tasks in the message queue whenever it is free from its existing tasks. This architecture supports a scalable model for inter-process communication because no process is waiting for the other process to complete a particular task. This helps in avoiding a lot of situations where resources can reach a blocked state waiting for other processes to complete.

The most important advantage of this implementation is that the producer and the consumer can work at its respective speed. These features of LSCS mode enhance the user experience because the GUI process is free from undesirable interrupts.

© 2023 Cadence Design Systems, Inc.

Note: If you have multiple netlister processes running, each process netlists a different test in parallel until the tests are exhausted and then jobs are used to create a netlist for each point.

Expression Evaluation Process

The expression evaluation process is an independent process that runs outside of Virtuoso to evaluate the expressions. This process does not require any access to the design-related information. Therefore, it is launched by Virtuoso as an independent process by using the vds executable. This process reads the initialization information from the `.vdsinit` file. It does not read `.cdsinit`.

Important

The expression evaluator process is launched only by an internal function of a Virtuoso process. You cannot run this process independently by explicitly running the vds executable from the command line or in any other way. However, you can configure the settings to be used by `vds` in `.vdsinit`.

By default, the expression evaluation process does not call `virtuoso` because it does not need to access the design information. However, if your expressions rely on the design or layout information, you can explicitly switch to use `virtuoso` by setting the `exprEval` environment variable to `"virtuoso"`:

```
envSetVal("maestro.exprEval" "binaryName" 'string "virtuoso")
```

After the above environment variable setting is done, the expression evaluator uses the `virtuoso` executable instead of `vds`.

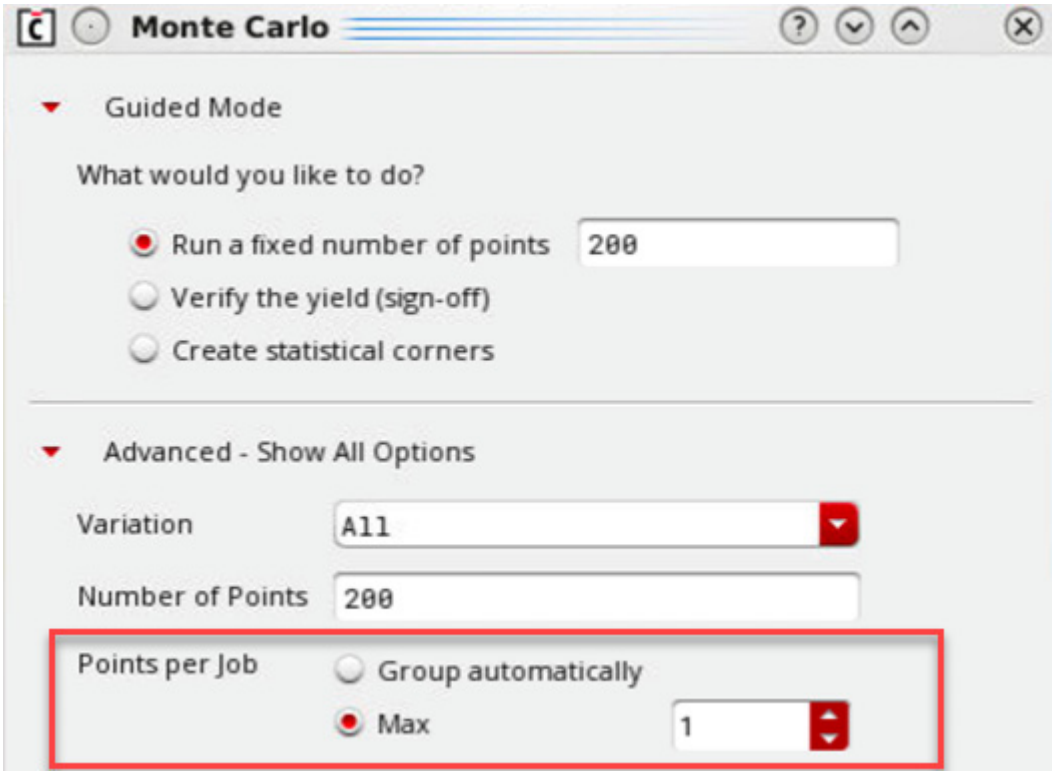
Benefits of LSCS Job Control Mode

The various benefits of LSCS are listed below.

- It is a highly scalable job control mode because each process runs independently without impacting the other processes or making their resources hang. This allows reliable handling of thousands of netlisting and simulation jobs running in parallel.
- It checks if the design for different points is common. For the common designs using config sweep, it reuses the netlist across corners. This allows optimum resource utilization. If the design is same, a netlist can be reused in the following cases:
  - Across corners or variable sweeps for each view
  - Across the same design tests or tests with same top-level testbench

© 2023 Cadence Design Systems, Inc.

- It enables flexible distribution of the respective jobs. The jobs queued up for each process can be distributed to different farm computers locally or remotely. This is enabled by allowing different job policy definitions for simulation and netlisting, respectively. This significantly reduces the netlisting and simulation run time as compared to ICRP job control mode.
- The testbench design can be opened in read-only mode while netlisting is in progress. Therefore, when the simulation is in progress, you can modify the setup. Only design editing is not possible.
- Enables better debugging through improved log recording and introduction of new log views. For more information, refer to the section Viewing Log Files.
- You can control the maximum iterations per simulation in the Monte Carlo Options form.



Benefits of LSCS Over ICRP

The following points describe how LSCS is better than ICRP.

- Due to sequential processing in ICRP, a simulation process has a large waiting time when the netlist generation is in progress. In LSCS, all the processes run independently.

© 2023 Cadence Design Systems, Inc.

- In ICRP, the netlist of the same design is generated repeatedly for different points in a design. However, LSCS reuses the netlist of the same design.
- In ICRP, large number of jobs can cause the GUI to freeze. However, LSCS allows running each job without impacting the other processes or hanging their resources.
- In order to save the netlist for a point from inadvertent changes, when netlisting is done in the GUI process, UI is frozen until all netlists are created. In LSCS, netlists are created in background processes and integrity of each netlist is maintained.
- In ICRP, all the resources used in netlisting, monitoring and expression evaluation are used up when a simulation is in progress. However, LSCS distributes all the processes and queues them at different farm computers. This minimizes the possibility of holding up of any resources for a process.

Limitations of LSCS

The LSCS job control mode has the following limitation:

- It is supported for only Cadence simulators, Spectre and AMS. If you set LSCS and run simulation using any third-party simulator, ADE Assembler prompts you to confirm whether it can change the job control mode to ICRP. Click *OK* to change the control mode to continue the simulation with ICRP, or stop simulation and change the simulator.

Performance of LSCS

For the netlisting phase in LSCS, you might experience a slight performance dip in IC23.1 when compared to the earlier versions. This is caused by an additional check to improved stability in IC23.1. We ensure that all the necessary data is synchronized on the remote machines in the NFS before simulation starts. This may take a few seconds depending on the performance of your NFS system.

Configuring the Job Policy Setup for LSCS

Job distribution is done according to the distribution method set by the *Job Control Mode* option on the Job Policy Setup form in ADE Explorer or ADE Assembler.

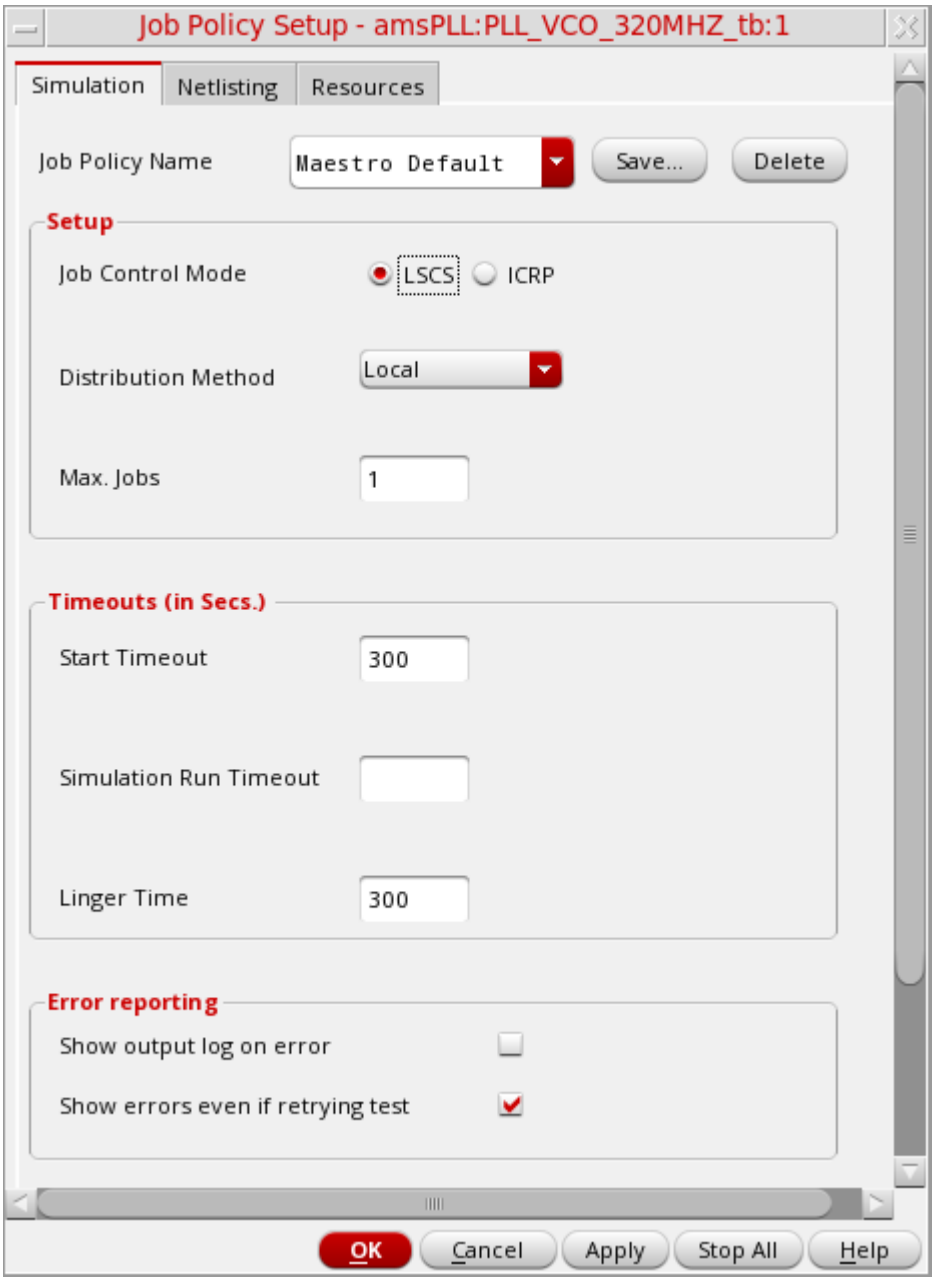
To set up LSCS job control mode:

1. In ADE Explorer, choose *Setup – Job Setup* to open the Job Policy Setup form.

   Note: In ADE Assembler, you can open the form by choosing *Options – Job Setup*.

© 2023 Cadence Design Systems, Inc.

2. On the *Simulation* tab, select *LSCS* job control mode.



Each job can be configured to run one or more simulations. ADE Explorer and ADE Assembler internally uses these jobs to efficiently distribute time-consuming tasks that can be performed in parallel. Settings for these jobs such as how many remote processes to start, where the processes should run, on local or remote computers, or the time for which a remote process should stay active and wait for a simulation to run are set as a job policy.

© 2023 Cadence Design Systems, Inc.

Important Points to Note

- In LSCS mode, jobs for simulation and netlisting run independently. You must set separate job policies to define the type of resources to be used by them.
- Any changes made in a job policy might impact the simulations in other maestro sessions opened in the same Virtuoso session and using the same job policy. To avoid any impact on those sessions, you can save the policy with another name.

The following sections describe the settings on the various tabs of the Job Policy Setup form:

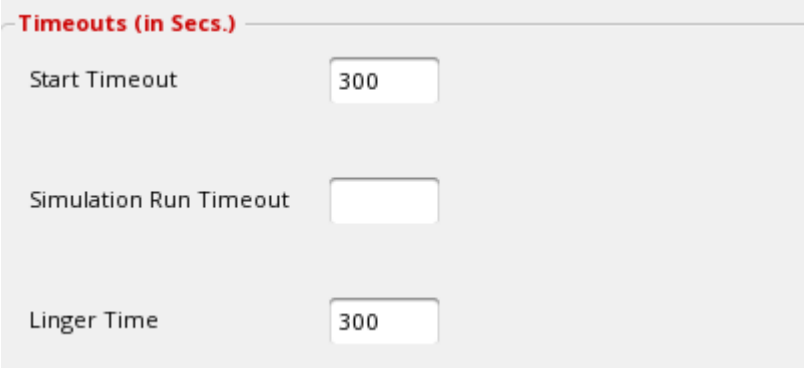
- Specifying a Job Policy Name
- Optimizing a Single Point Run
- Specifying a Distribution Method
- Specifying Max Jobs
- Specifying a Command for DRMS
- Specifying an LBS Job Policy
- Specifying Job Timeouts for ICRP
- Specifying Error Reporting Options
- Setting up the Netlisting Job Policy for LSCS
- Setting up the Resource Estimation Options for LSCS

© 2023 Cadence Design Systems, Inc.

Specifying Job Timeouts for LSCS

To customize job timeout options, specify values in the fields of in the *Timeouts* group box of the *Simulation* tab of the Job Policy Setup form.

All the values in this section are specified in seconds. When a timeout value is set to `0`, -1, or is left blank, the timeout is disabled. This leads to an unlimited wait time.



The following table describes the fields in the *Timeouts* section:

Field	Description
Start Timeout	Specifies the time (in seconds) for which the tool should wait for the simulation process to report back to ADE Assembler that it has started. The wait time starts as soon as ADE Assembler submits the job. If no response is received in the given time, ADE Assembler considers the job as not started and kills it. If there is another job waiting to start, ADE Assembler starts it. This value is also used as the wait time for the expression evaluator jobs. If a job fails to complete for a point, ADE Assembler sets the status of that point as `eval err`. **Note:** If you are using LBS or an interactive bsub or qsub command for job distribution and the jobs take longer than this timeout value to start, ADE Assembler might consider the job as failed. To avoid this, you can set the start timeout to blank or to a larger value, for example, 100000.

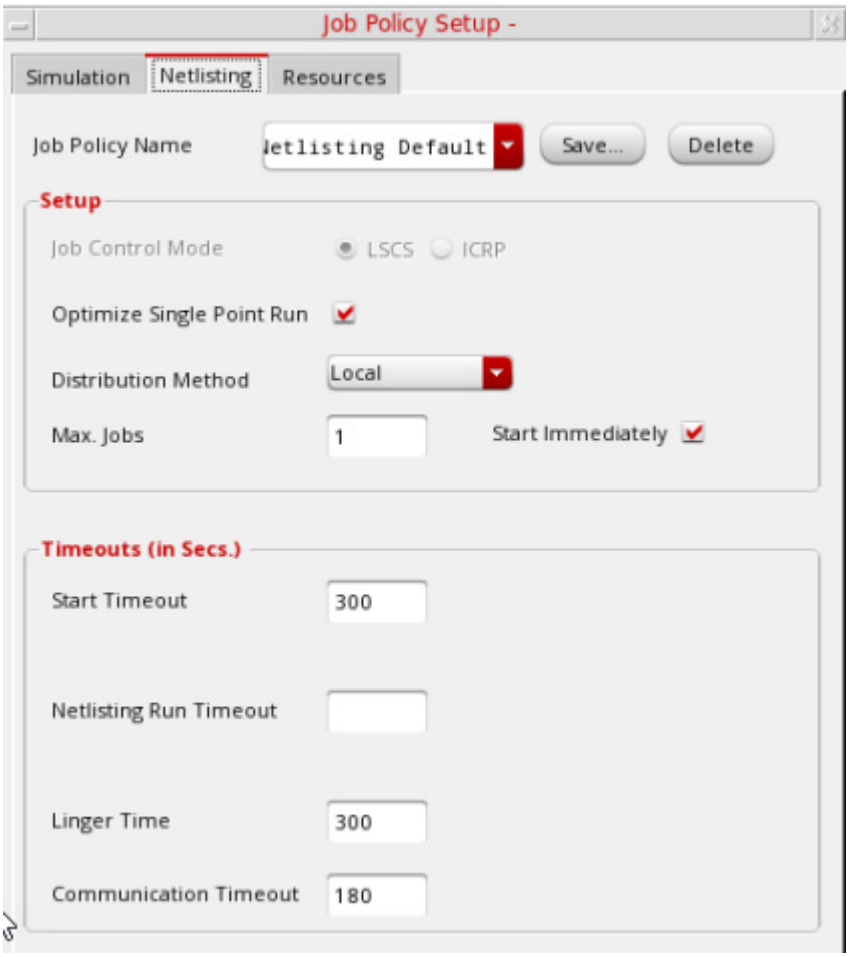
© 2023 Cadence Design Systems, Inc.

Simulation Run Timeout	Specifies the number of seconds to wait for the Spectre monitor process to report back to ADE Assembler that it has run the job. The wait time starts as soon as ADE Assembler sends the run command for the job. If the process does not respond that the job has completed in the number of seconds you specify, ADE Assembler considers the job run unsuccessful and kills it. ADE Assembler looks for the next idle job and tries to start, and run it.
Linger Time	Specifies the number of seconds to wait for the Simulation monitor and expression evaluator processes to kill a remote job after its task is complete.

© 2023 Cadence Design Systems, Inc.

Setting up the Netlisting Job Policy for LSCS

You can set up a job policy for netlisting jobs on the *Netlisting* tab of Job Policy Setup form.



The following table describes all the fields on this tab of the Job Policy Setup form.

Field Name	Description
Optimize Single Point Run	Specifies that an ADE Assembler process must be used for netlist generation to optimize a simulation run. For more information, see Optimizing a Single Point Run.

© 2023 Cadence Design Systems, Inc.

Field Name	Description
Distribution Method	Specifies a distribution method by using the drop-down list. Depending on your setup, you can choose one of the following methods: ■ *Command* – You send a command to the distributed processing software. ■ *Local* – The program creates netlists on the local host. ■ *Remote-Host* – The program creates netlists on the remote host that you specify.
Max Jobs	Specifies the maximum number of netlist services that can run in an ADE Assembler or ADE Explorer session. **Note:** For the local or remote-host distribution method, it is recommended to keep the maximum jobs for netlisting services less than the number specified by the maxNetlistingJobsForLocal environment variable. By default, this variable is set to `50`. When the *Start Immediately* check box is selected, the program immediately submits the specified maximum number of jobs, or one job for every test, whichever is less, when you start ADE Explorer or ADE Assembler. This means that the defined number of LSCS processes are already running before you start the simulation. This saves the overhead of starting LSCS processes at the time of generating the netlist. To turn off this feature, deselect the *Start Immediately* check box.
Start Immediately	Specifies whether to launch the netlister as soon as a maestro view is opened.
Start Timeout	Specifies the time for which the tool should wait for the netlisting process to report back to ADE Assembler that it has started. The wait time starts as soon as ADE submits the job. **Note:** If LSCS does not respond in the given time, ADE Assembler considers the job not started and kills it. If there is another job waiting to start, the tool starts it.

© 2023 Cadence Design Systems, Inc.

Field Name	Description
Netlisting Run Timeout	Specifies the time to wait for a netlisting job to complete. It starts as soon as a netlisting job is initiated by ADE **Note:** If the netlisting job does not respond that the job has completed in the given time, ADE Assembler considers the job run unsuccessful and kills it and looks for the next idle job and tries to start, configure, and run it.
Linger Time	Specifies the number of seconds after which you want the program to kill a remote job after the netlist is generated.
Communication Timeout	Specifies the number of seconds to wait for the process to report back to ADE Assembler that the Message Passing System (MPS) communication has been successfully established. This value is helpful in troubleshooting a pending job.

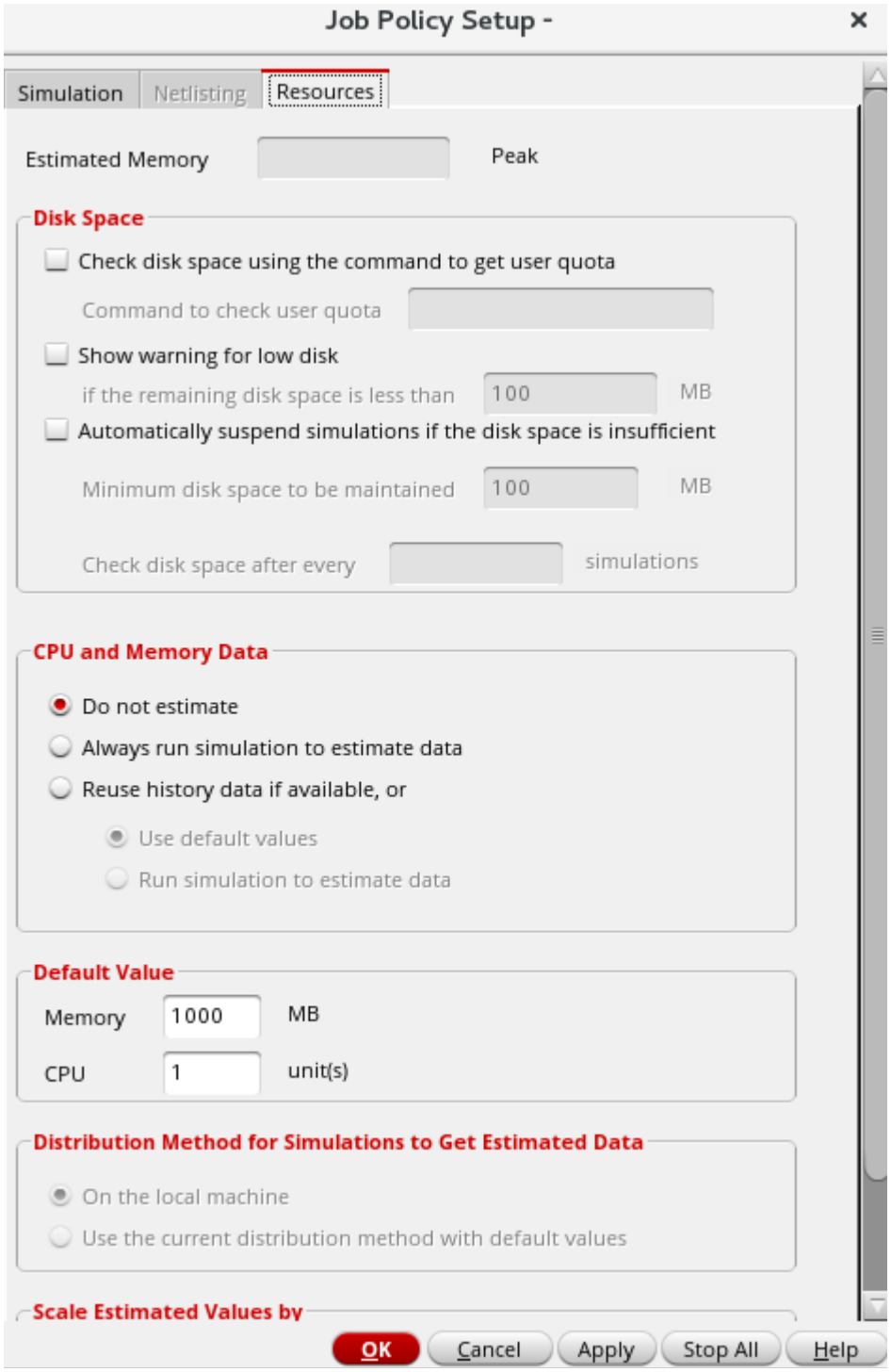
Related Topics

Troubleshooting Jobs that Stay Pending until Timeout

© 2023 Cadence Design Systems, Inc.

Setting up the Resource Estimation Options for LSCS

On the *Resources* tab of Job Policy Setup form you can specify options for disk space usage and for CPU and memory data estimation.



© 2023 Cadence Design Systems, Inc.

The following table describes all the fields on this tab of the Job Policy Setup form.

Form Field	Description
Estimated Memory	Shows the estimated memory requirement to run the current simulation run. This feature is enabled by using the options given in the *CPU and Memory Data* group box.
Disk Space	
- Check disk space using the command to get user quota	Enables checking of the disk space available in user quota. If you want to alert when the available disk space is less than the given space, select this check box to check for user quota of disk space.
- Command to get user quota	Specifies the command to be run to check the user quota. The command must return an output in the format `'available: space'`. For example: `echo 'available: 10000'` you can also specify the name of a script that returns this value. For example: `./getQuota.sh` If you enter an invalid command, Spectre displays a question giving you choice to either continue the simulation run without disk space check based on user quota or to cancel the run and correct the command.
- Show warning for low disk	Enables checking of the available disk space when a simulation run is started.
- Automatically suspend simulations if the disk space is insufficient	Enables automatic suspension of all the simulations in progress whenever the available disk space falls below the minimum threshold specified in the *Minimum disk space to be maintained (in MB)* field. For more details, see Suspending Simulations Automatically When the Disk Space is Low.
CPU and Memory Data	

© 2023 Cadence Design Systems, Inc.

Form Field	Description
- Do not estimate	Disables estimation of processors and memory required for the current simulation. In this case, you can yourself specify the required number of processors to be used to distribute simulations. **Note:** When you disable estimation for LBS DRMS, select the check box to the right of the *Parallel Num. Processors* field on the *Simulation* tab of the Job Policy Setup form and specify a value in it. If you clear the check box given to the right of the *Parallel Num. Processors* field, the value given in the field is also cleared. In such a case, you should select any one of the other two options in the *CPU and Memory Data* field to get an estimate of resources required.
- Always run simulation to estimate data	Enables resource estimation. When this option is selected, ADE Assembler runs the `+query=all` Spectre command before running simulations for the complete setup. This command retrieves the estimated CPU along with the lower and upper bounds and populates the lower bound values in the resource requirements string on the *Simulation* tab. It also retrieves the estimated memory and adds that to the *Estimate Memory* field on the *Resources* tab. If the simulation that estimates resources does not return valid results, ADE Assembler uses the data from a history, if available, or the default values specified in the *Default Value* section on this tab. See also: How ADE Explorer and ADE Assembler Estimate Resources?.

© 2023 Cadence Design Systems, Inc.

Form Field	Description
- Reuse history data if available	Enables resource estimation. When this option is selected, ADE Assembler checks the resources required from the Spectre log file of the most recent history. It uses the maximum memory and processors required across all points. Before reusing the data from a history, ADE Assembler compares the following components in the setup of the history with that of the active setup: ■ Simulator ■ Simulation files ■ Test names ■ Design name ■ Run mode ■ Model files and sections in corners If any mismatch is found, ADE Assembler reruns resource estimation based on the active setup. If no history data is available, the tool can either use the default values specified in the *Default Value* group box or run a simulation to estimate resources.
Default Value	
- Memory	Specifies the default memory requirement in Mbytes Default value: `1000` (in MBytes)
- CPU	Specifies the default number of CPUs required Default value: `1` (number of processors) If you know the resource requirements for the simulations you are running, you can specify the default values and select the *Use Default Values* option in the *Provide CPU and Memory Data* group box.
Distribution Method for Simulations to Get Estimated Data	

© 2023 Cadence Design Systems, Inc.

Form Field	Description
- On the local machine	Specifies that the simulation for resource estimation needs to be run on the local computer irrespective of the distribution methods set on the *Simulation* tab. This option is helpful when you cannot run simulations on a local computer.
- Use the current distribution method with default values	Specifies that the simulation for resource estimation needs to be run using the distribution method specified on the *Simulation* tab. This option helps to calculate an accurate estimate of resource requirement because it uses the same remote computers that are required to run the simulations.
Scale Estimated Values by	
- Memory	Specifies the value by which the estimated value of memory requirement should be scaled. Default value: `100%`
- CPU	Specifies the value by which the estimated value of CPU requirement should be scaled. Default value: `100%`

How ADE Explorer and ADE Assembler Estimate Resources?

To run resource estimation, ensure the following:

- In the High Performance Simulation Options form, the *Manual* option is selected for *Multi-Threading* and the *#Threads* field is set to `lsf`.
- On the *Simulation* tab of the Job Policy Options form, the *Distribution Method* field is set to `LBS` and the resource requirements string contains the `%CPU_EST` and `%MEM_EST` templates.
- On the *Resources* tab of the Job Policy Options form, the *Always run simulation to estimate data* option is selected.

With these settings, ADE Explorer and ADE Assembler runs the `+query=all` Spectre command before running simulations. It reads the estimated CPU and memory information returned by Spectre and substitutes the values in the `%CPU_EST` and `%MEM_EST` templates

© 2023 Cadence Design Systems, Inc.

in the LBS resource string. These resource requirements are used by the distribution method to allocate the required resources for simulation.

When the *Reuse history data if available* option is selected, the tool uses the resource estimation data from a previous history. Similarly, when the *Use default values* option is selected, the tool uses the default values specified in the *Default Value* section of this form. In either case, before starting simulation, the tool prints a message in the CIW indicating the source of resource estimated for a run.

Important Points to Note

- Spectre does not support resource estimation for the following run modes:
  - Improve Yield
  - Sensitivity Analysis
  - Single Run, Sweeps and Corner with statistical corners

  If these run modes are set, by default, ADE Assembler shows an error message and prompts you to select the *Default Values* group on the *Resources* tab of the Job Policy Setup form.

  If you do not want to be prompted when the current run mode is not supported for resource estimation by Spectre, set the `ignoreRunModeCheckInResourceEstimation` environment variable to `t`. When this variable is set to `t`, the tool uses values from an available history. If there is no history available, it uses the default memory and CPU value specified in the *Default Values* group on the *Resources* tab of the Job Policy Setup form.
- Spectre currently supports resource estimation for only hb and tran analyses. If the setup contains an enabled analysis other than these two, the tool automatically reuses values from an available history. If there is no history available, it uses the default memory and CPU value specified on the *Resources* tab of the Job Policy Setup form.

  If you want to be notified when the current analysis is not supported for resource estimation by Spectre, set the `ignoreAnalysisCheckInResourceEstimation` environment variable to `nil`. This variable lets you to decide whether to use the resource estimation values from any other history or the default values from the job policy.
- Resource estimation is done for each test in the setup. If the global job policy is used by all tests, ADE Assembler uses the maximum CPU or memory value among the values estimated for all the tests. If the tests are using their own overridden job policies, resource estimation is run for each test individually. If multiple tests use the same policy, the maximum estimated CPU or memory value is used for all.

© 2023 Cadence Design Systems, Inc.

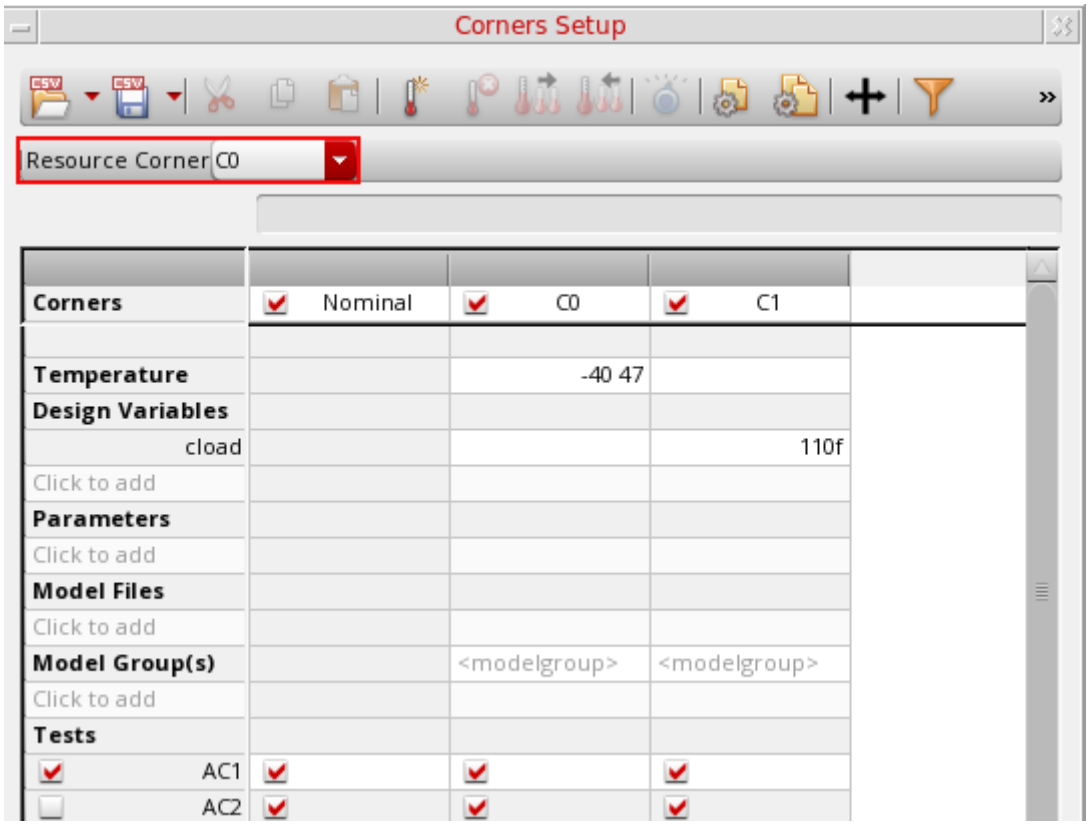
Running Resource Estimation for a Specific Corner

By default, ADE Assembler runs resource estimation on the nominal corner. If your setup contains corners that would require a higher level of resources than nominal, it is a better option to assess the resource requirement for a corner with the highest resource requirements.

You can specify a resource corner at the global level for all tests or override it for a specific test.

To specify a resource corner for all tests, do the following:

1. In the Corners Setup form, add the corner with high resource requirements.
2. Select the corner name in the *Resource Corner* drop-down list below the toolbar on this form.



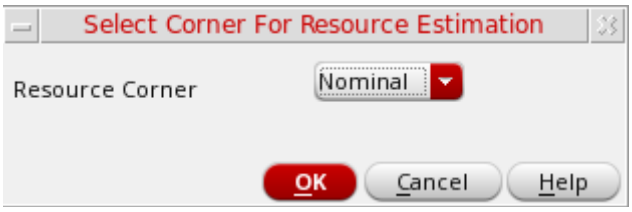
3. Click *OK* to close the corners setup.

To override a resource corner for a specific test, do the following:

1. In the Data View assistant, right-click the test and choose *Resource Corner*.

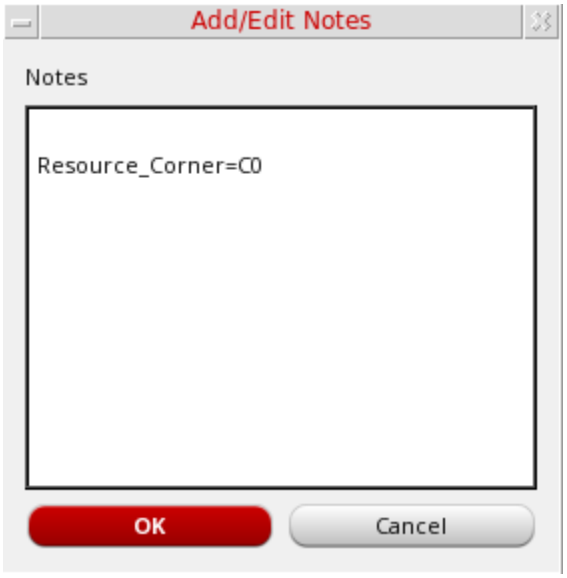
© 2023 Cadence Design Systems, Inc.

The Select Corner for Resource Estimation form is displayed.



2. From the *Resource Corner* drop-down list, select the corner for which you want to run resource estimation for that test.

   The global resource estimation corner is overridden for the test. This test-specific resource corner is added to the notes for the test. You can validate the resource corner in the tooltip for the test or right-click the test name and choose *Notes* to open the Add/Edit Notes form.



If a resource corner is specified, ADE Assembler runs resource estimation as described below.

- If the resource corner specified in the Corners Setup form is enabled for the current simulation run, resources are estimated for that corner. However, if the resource corner is disabled, the nominal corner is used.
- If the resource corner is disabled for a test, ADE Assembler uses the first enabled corner for that test.
- If the resource corner is a group corner, the first corner in that group is used.

© 2023 Cadence Design Systems, Inc.

- If a global resource corner is overridden for a particular test, estimation for that test is done by using the specified corner. For other tests, estimation is done by using the resource corner in the Corners Setup form.

Starting a Simulation

After you have selected a job control mode and completed the setup, run a simulation.

When a simulation is running, the status bar at the bottom of the ADE Explorer or ADE Assembler window shows the different stages of the simulation run. In addition, a progress bar on the bottom right of the window displays the information about the results directory, library, cell, and view.

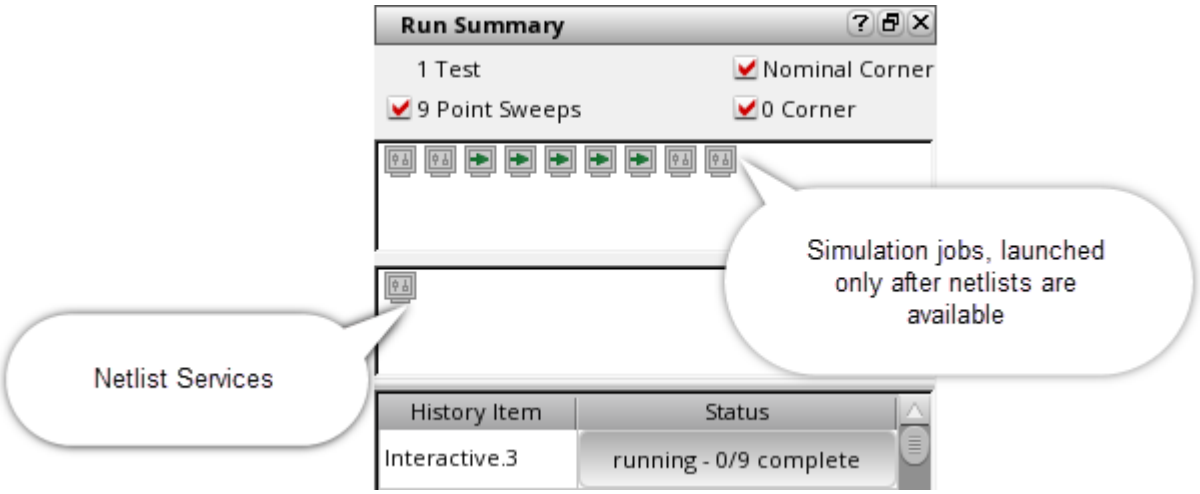
Reviewing Simulation and Netlisting Progress

When you run a simulation, a netlist is generated and the Run Summary assistant appears in ADE Assembler. The Run Summary assistant shows the following information about the simulation in progress:

- The number of tests running
- The number of values varying (Point Sweeps)
- The number of corners
- The nominal corners

© 2023 Cadence Design Systems, Inc.

While ADE Explorer or ADE Assembler is preparing to run a simulation or during a run, you can check its status on the progress bar displayed on the Run Summary assistant pane. The progress bar is displayed to the right of the history item name.



Points to Remember:

- The log files are written to the `logs_<username>/logs<n>` directory.
- The number of jobs launched for each process is driven by the *Max. Jobs* defined in the respective job policy.
- Simulation jobs are launched only after the netlists are available.
- In LSCS, netlisting is done in the beginning at the test level. If netlist creation at the test level fails, the status for all points is shown as *netl err* irrespective of their submission status. Therefore, it is possible that if you cancel any point, its status is shown as *netl err*.

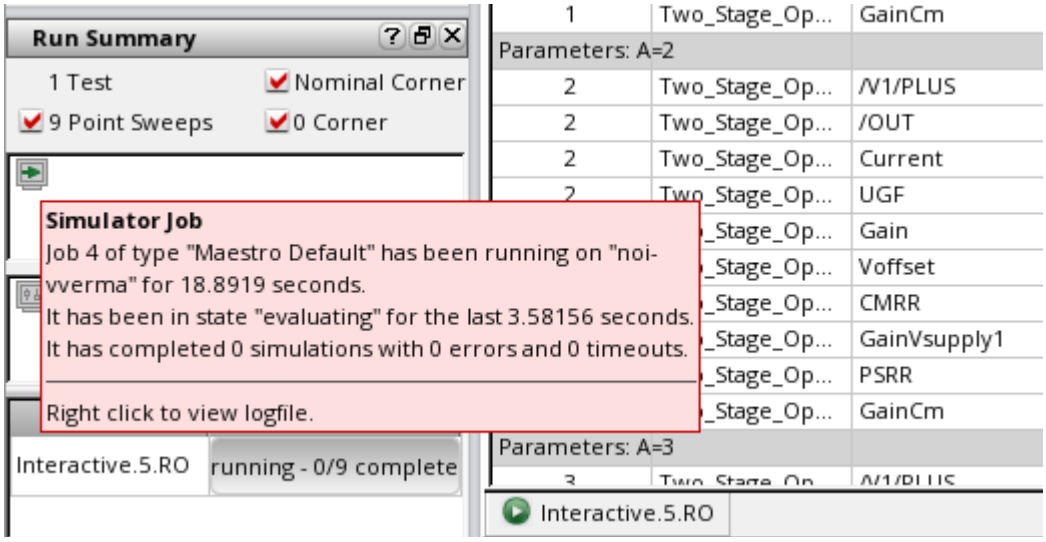
Note: The shows that the respective netlisting or simulation job is waiting.

Managing Jobs During Simulations

During a simulation, the *Run Summary* assistant shows the status of netlisting and simulation jobs that are running. You can check the status of each individual job by placing the pointer over it. A tooltip is displayed giving the details of the current simulation in progress

© 2023 Cadence Design Systems, Inc.

and the total number of simulations completed by the job. An example is shown in the following figure.



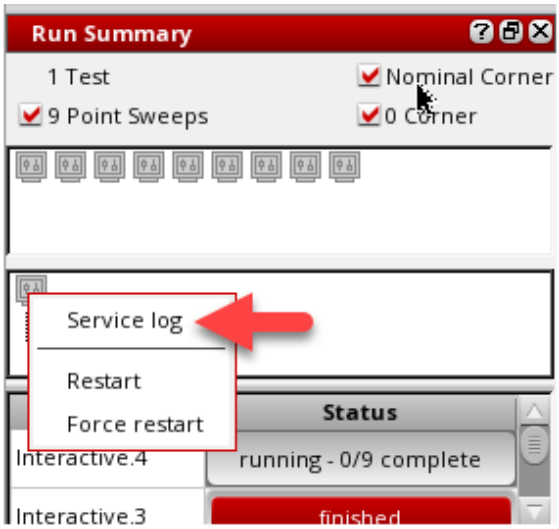
Stopping Jobs and Resubmitting Simulations

Stopping a job is helpful in the following scenarios:

- When a job is submitted to a computer that is known to be slow or with limited memory. This information can be gathered from the tooltip of a job.
- When a particular job has simulated a significantly smaller than average number of simulations.
- When nearing the end of a run, only one or two jobs remain pending and are taking more than the expected time to complete the simulation for the current design points.

© 2023 Cadence Design Systems, Inc.

To stop a job and resubmit the simulation, right-click the job and choose *Stop and Resubmit*.



When you use this command, ADE Explorer or ADE Assembler does the following:

- The icon for the stopped job changes to one with a cross, and then disappears. A new job is created to replace the stopped job.
- The simulation that was running on the stopped job is placed at the beginning of the queue of the pending simulations. The simulation for this point is then submitted to the next available job.
- If no more simulations are pending, a new job may not be created.

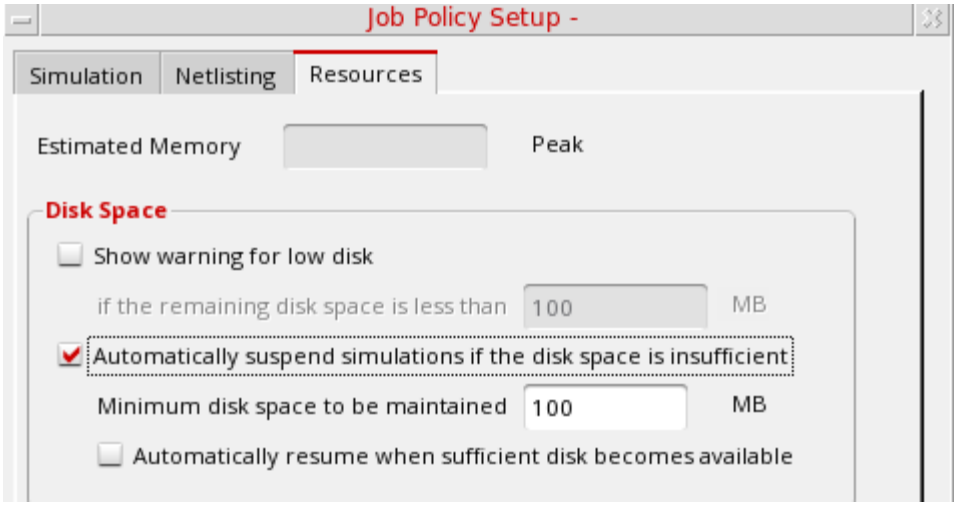
Automatically Suspend Simulations When the Disk Space is Low

You can configure the commands to automatically suspend simulations when the disk space is low, and flag the appropriate information or warning messages.

To enable checking of disk space and automatic suspension of simulations depending on the configured settings:

1. Choose *Options – Job Setup* to open the Job Policy Setup form.
2. Open the *Resources* tab.

© 2023 Cadence Design Systems, Inc.



3. Specify the following options in the *Disk Space* group:

   - *Show warning for low disk*: Select this check box to enable checking of the available disk space when a simulation run is started. ADE Assembler shows a warning in these cases:

     - If the available disk space before running a simulation for the first point is less than the threshold value specified in the *If the remaining disk space is less than _ MB* field, a warning message is flagged to report about the unavailability of the required space and to confirm whether the run is to be continued.

     - For the Single Run, Sweeps, or Corners run mode, after the first design point is simulated, the disk space consumed for that point is used to estimate the total space required to run simulations for all the points. If the **remaining (available – estimated)** disk space falls below the threshold, ADE Assembler shows the following types of messages:

       - **If the remaining (available – estimated) disk space is more than the threshold** specified in the *If the remaining disk space is less than* field, an information message is flagged to report the same.

       - **If the remaining (available – estimated) disk space is less than or equal to the threshold** specified in the *If the remaining disk space is less than* field, a warning message is flagged to report the same. It is recommended to manually suspend the simulation run for the current history, free up the disk space, and resume the run. To manually suspend a job, right-click the progress bar of the history in the Run Summary pane and click *Suspend*.

       Consider an example of a simulation run for three sweep points.
       - The simulation of the first design point used: 15 MB

© 2023 Cadence Design Systems, Inc.

- The estimated disk space for the remaining two points: 30 MB
- The available disk space after the first design point is simulated: 115 MB
- The threshold value for disk space: 100 MB
- The estimated remaining disk space = 115 – 30 = 85 MB, which is less than the threshold value

In this scenario, after the simulation for the first design point is complete, a warning message is displayed to clear disk space.

- *Automatically suspend simulations if the disk space is insufficient*: Select this check box to enable Spectre to automatically suspend all in-progress simulations whenever the available disk space falls below the minimum threshold specified in the *Minimum disk space to be maintained (in MB)* field.

  By default, the suspended jobs remain in that state until you resume those. However, you can select the *Automatically resume simulations when the disk space becomes available* check box to enable Spectre to automatically resume the suspended jobs. When this check box is selected, Spectre internally checks for the availability of disk the disk space ans resumes the suspended simulations.

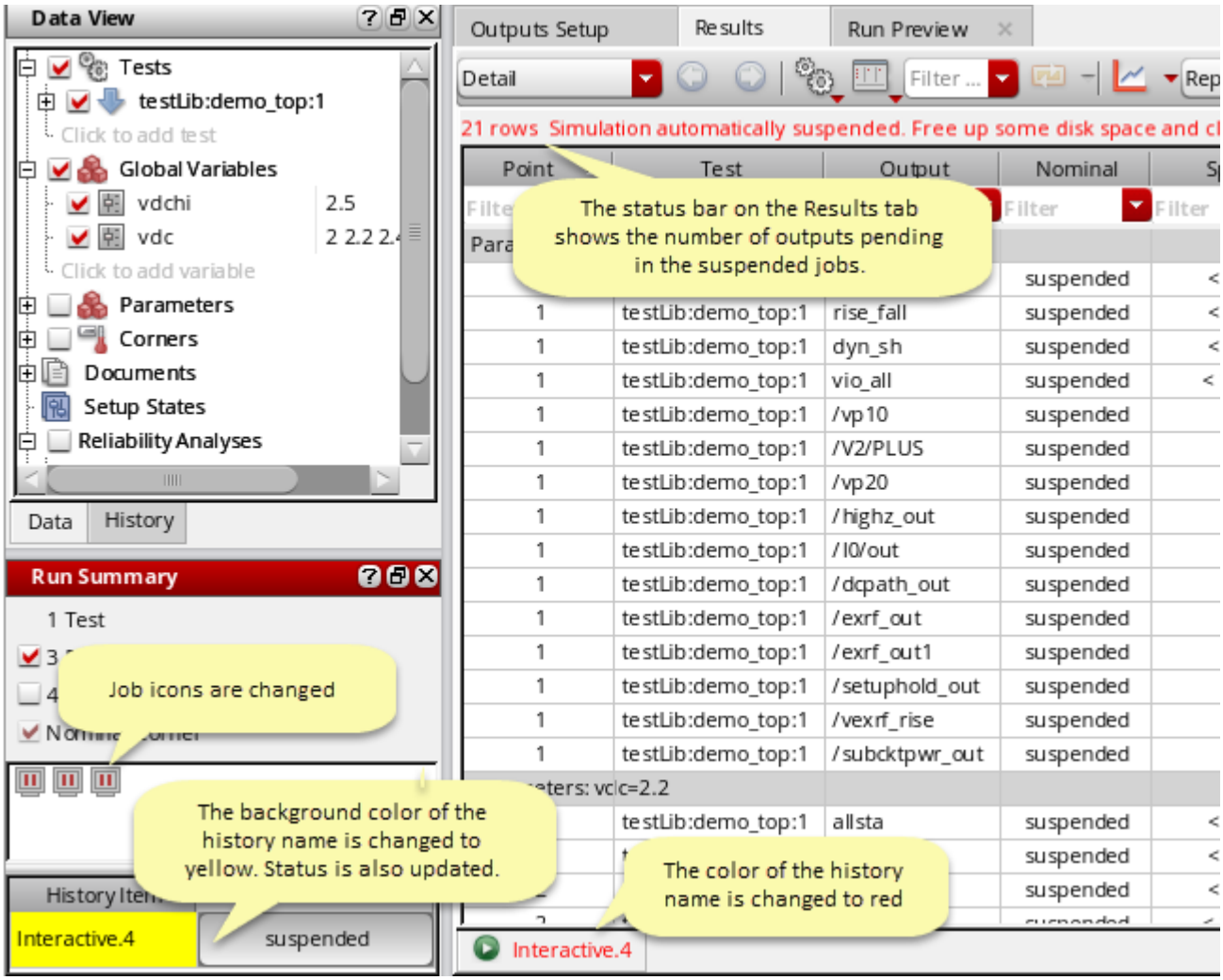
4. Click *OK* to close the form.

   ADE Assembler uses the settings specified on this form to constantly monitor the available and estimated disk space, and displays appropriate messages on the *Results*

© 2023 Cadence Design Systems, Inc.

tab. If a simulation is automatically suspended, the status on the *Run Summary* tab and the *Results* tab is updated, as shown below.

.



Restarting the Jobs

If the tool tip of a job shows the name of the slow computer and you observe that the currently running simulation is taking an unusually long time to complete, you can stop the job and restart it. The queuing mechanism will redistribute the job to another computer that has better resources.

Right-click and use the following commands in the context-sensitive menu of a job to restart it:

- *Restart*: Allows the current activities of a job to complete before stopping and then restarting the job.

© 2023 Cadence Design Systems, Inc.

- *Force Restart*: Forces the current activities of a job to halt immediately and forces the job to restart.

Viewing Log Files

For a simulation run using LSCS run mode, you can view the log files in two ways:

- Viewing Log Files for a Data Point
- Viewing a Consolidated Log for a History

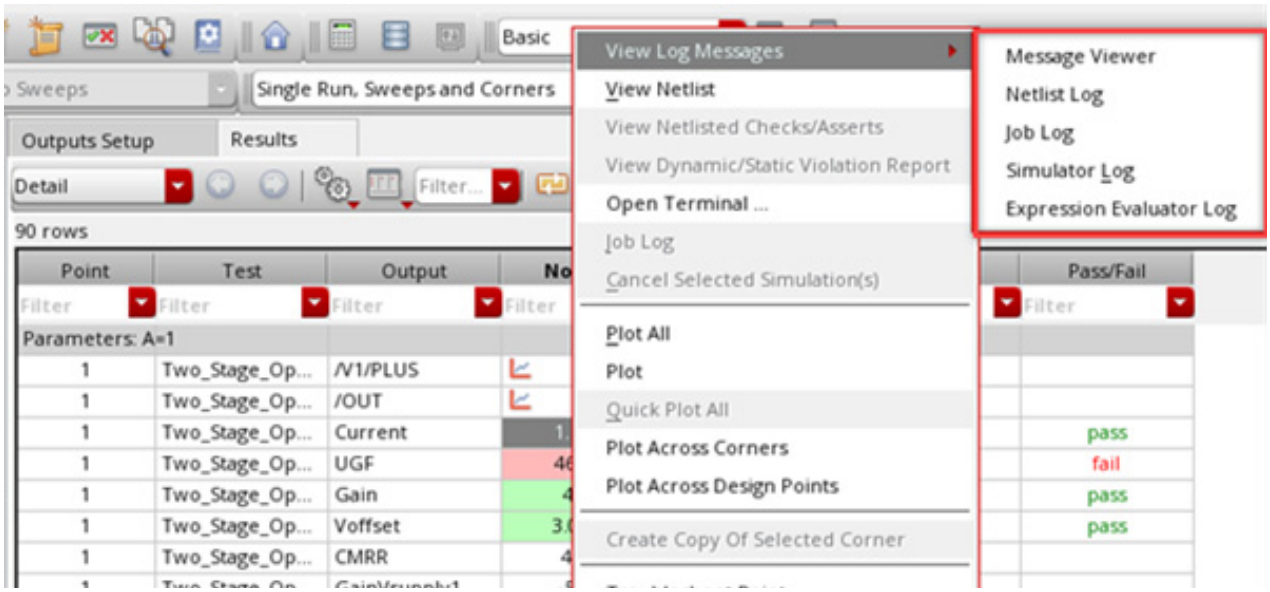
Viewing Log Files for a Data Point

To view the logs for a data point:

1. Right-click a result value in the `Detail` or `Detail-Transpose` results view.
2. Click *View Log Messages* and choose one of the following types of logs you want to view:
   - Message Viewer
   - Netlist Job
   - Job Log
   - Simulator Log

© 2023 Cadence Design Systems, Inc.

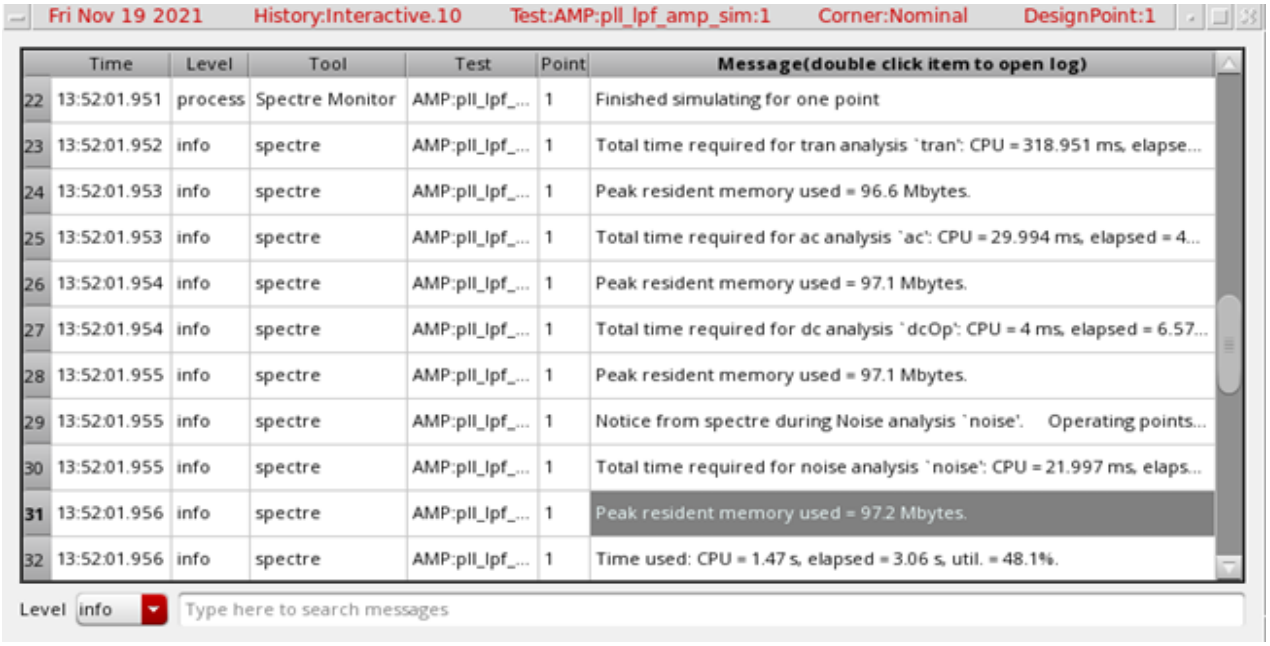
- Expression Evaluator Log



ADE Assembler opens a new window to display the log.

Message Viewer

This log view combines the messages for a specified point from the netlisting, simulation, and expression evaluation jobs that were involved in running a simulation for that point.



© 2023 Cadence Design Systems, Inc.

The tool extracts all the information displayed for a simulation process, for example, the peak resident memory, from the Spectre output that is saved in the Spectre output log.

To view the messages of a specific severity level, choose the level name from the *Level* drop-down list at the bottom of this viewer. The possible levels are: debug, event, process, info, warn, and error.

In addition, you can use the search bar to filter the information based on specific keywords in the *Message* column.

Netlist Job

The netlist log file shows netlisting logs for a specific point. It includes various details, such as testbench details, variables and parameters for netlist, netlist directory, and netlisting errors (if any).

Job Log

The job log contains information about LSCS jobs and all the simulations that were run on that job. It includes various details, such as the Cadence software version number, operating system version, working directory, the testbench details, variables and parameters for every run, simulator details, results location, data directory, netlist directory, and simulation errors (if any).

Simulator Log

This log file shows information about simulation jobs for the specified data point. It includes various details, such as the Cadence software version number, operating system version, working directory, design path, simulator, results location, data directory, start time and end time for the simulation run, the number of points completed, and the simulation errors (if any).

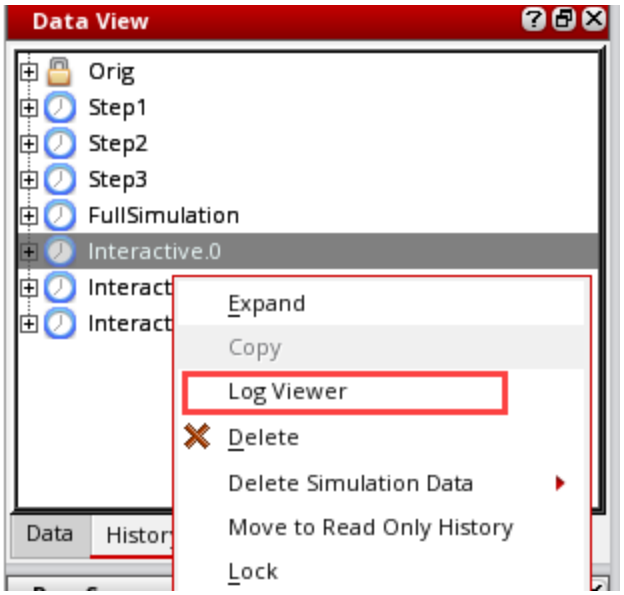
Expression Evaluator Log

This log file contains the information about the configuration setup, working directory, testbench details, variables, and parameters of the expression evaluator process for the specified point.

© 2023 Cadence Design Systems, Inc.

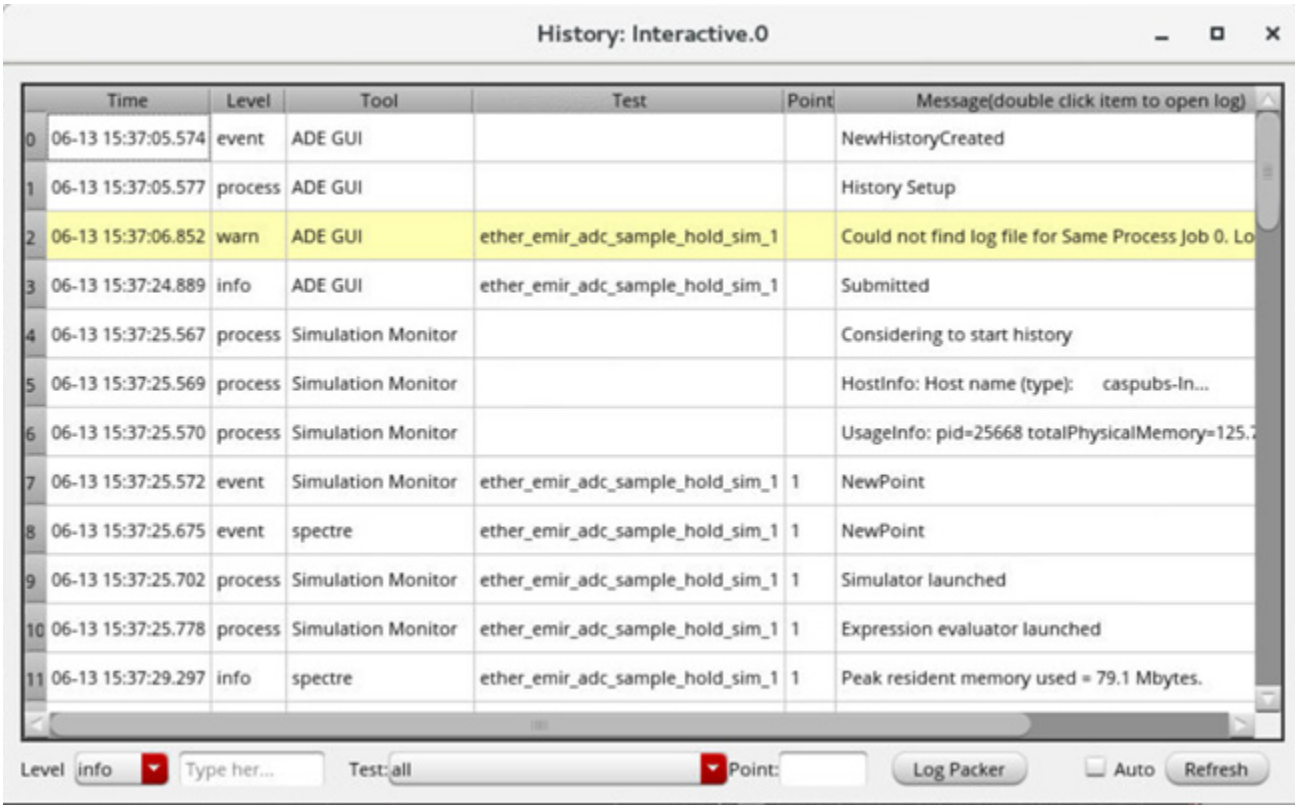
Viewing a Consolidated Log for a History

To view a consolidated report for a simulation run so that you can read the logs from all sources in one log viewer window, right-click a history and choose *Log Viewer*.
i



© 2023 Cadence Design Systems, Inc.

The Log Viewer window for the history is displayed.
i



Note: To open the Log Viewer window in ADE Explorer, choose *Tools – Log Viewer* command from the toolbar.

Observe that the *Tool* column in this table indicates the source of the log information. For example, row 23 is reported by the netlist service, row 24 by the expression evaluator, and row 28 by the Spectre simulator.

You can use the controls at the bottom of the viewer to filter the information displayed in the table. For example, you can choose error from the *Level* drop-down list and test `DCGain` from the *Test* drop-down list to view all errors reported for that test.

Click *Refresh* to view the logs for a latest run job.

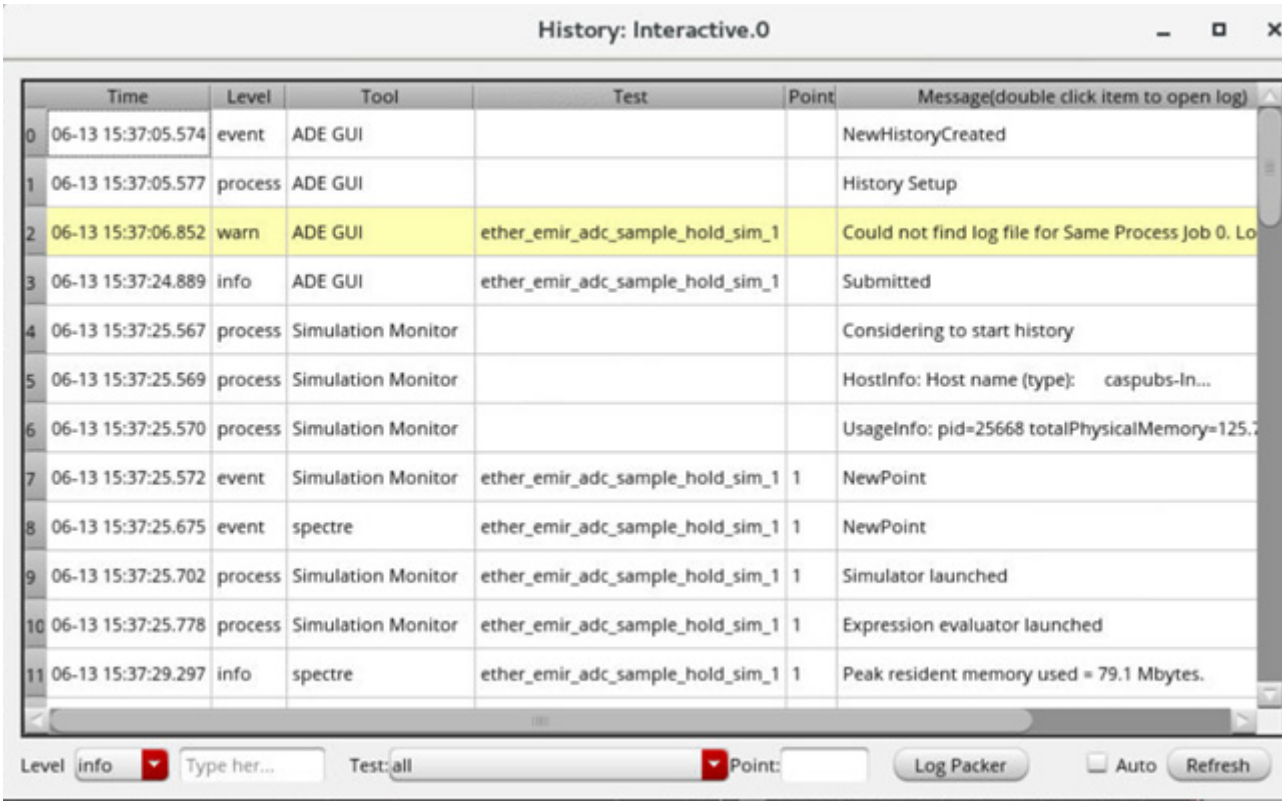
Saving Logs in a Compressed File

In addition to customizing the display of information, you can save the log in a tar file or zip file and share with others for debugging or analysis of the netlisting and simulation run.

To save all logs for a history in a compressed file:

© 2023 Cadence Design Systems, Inc.

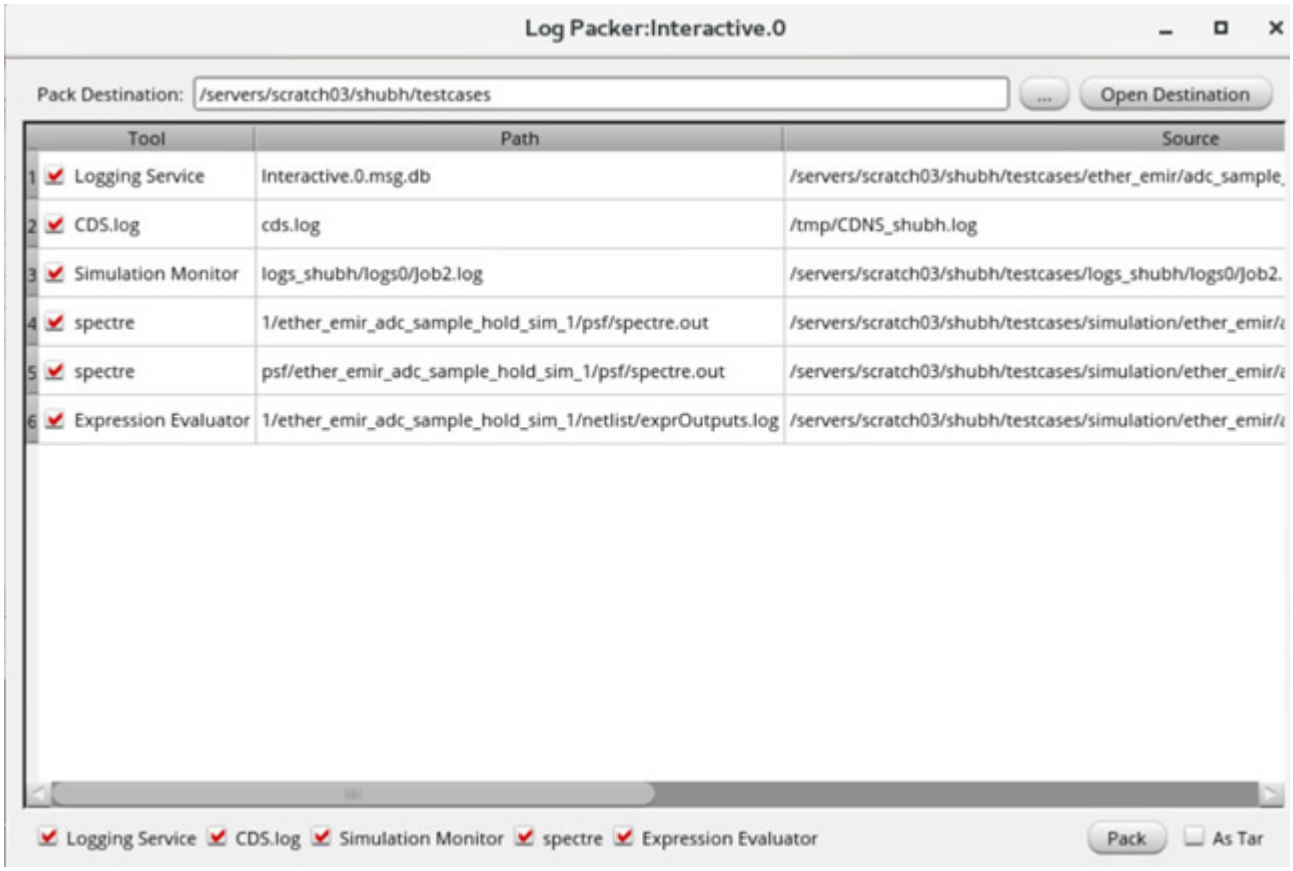
1. Right-click a history in the *History* tab of the Data View assistant and choose *Log Viewer*.
2. The log view that combines the messages for a specified point from all the jobs like netlisting, simulation, and expression evaluation.



3. Click *Log Packer.*

© 2023 Cadence Design Systems, Inc.

The Log Packer window is displayed.



4. Click the browse button to the right of the *Pack Destination* field and choose the location where you want to save the compressed log files.
5. Review the selected tools and if required, deselect the check box corresponding to the tool for which you do not want to save logs.
6. (Optional) To create a tar file, select *As Tar*.
7. Click *Pack* to compress and save the compressed files at the specified location.

The Log Packer utility is available in the log viewer for the complete history. It is not available for an individual data point.

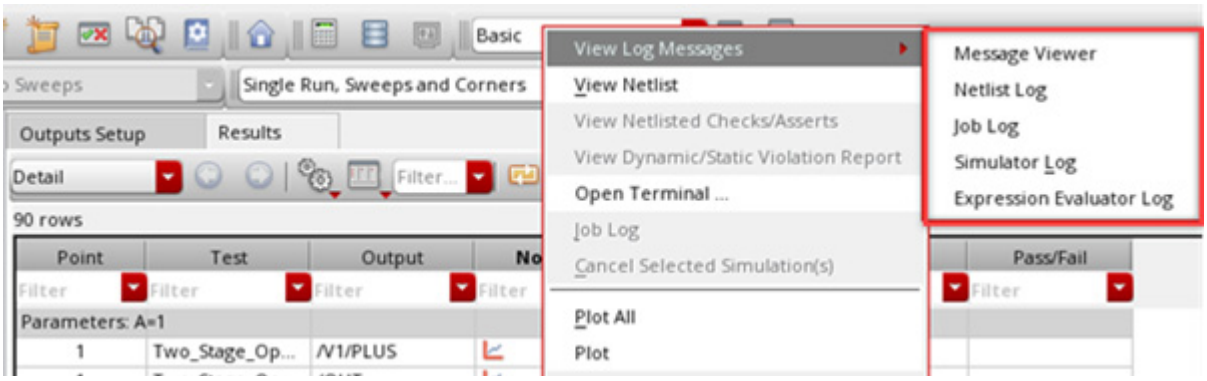
© 2023 Cadence Design Systems, Inc.

Using Job Logs for Debugging

You can use log files to view the debugging information for a simulation.

To view the logs for a data point:

1. Right-click a result value in the `Detail` or `Detail-Transpose` results view.



2. Click *View Log Messages* to view any of the following logs:
   - Netlist Log
   - Job Log
   - Simulator Log
   - Expression Evaluator Log
3. Review the information given in the log files and take appropriate steps to resolve the issues and to obtain the desired results.
4. (Optional) By default, the log files show basic error, warning, or informational messages. If the information available in the log files is not sufficient and you need more details, use the following environment variables to specify a higher debug level in which the tool saves additional details and displays more detailed messages:
   - `maestro.debug netlisting`
   - `maestro.debug simAndEval`
   - `maestro.debug beanstalk`
   - `maestro.debug mainVirtuoso`

© 2023 Cadence Design Systems, Inc.

Related Topics

Log Files for LSCS Job Control Mode

maestro.debug netlisting

maestro.debug simAndEval

maestro.debug beanstalk

maestro.debug mainVirtuoso

© 2023 Cadence Design Systems, Inc.

Log Files for LSCS Job Control Mode

The following table describes the available log files for LSCS job control mode.

Log File	Description
Netlist Log	Displays the netlisting logs for a specific point. It includes various details, such as testbench details, variables and parameters for netlist, netlist directory, and netlisting errors (if any).
Job Log	Displays information on LSCS jobs and all the simulations that were run on that job. It includes various details, such as the Cadence software version number, operating system version, working directory, the testbench details, variables and parameters for every run, simulator details, results location, data directory, netlist directory, and simulation errors (if any).
Simulator Log	Displays information on the simulation jobs for a specified data point. It includes various details, such as the Cadence software version number, operating system version, working directory, design path, simulator, results location, data directory, start time and end time for the simulation run, the number of points completed, and the simulation errors (if any).
Expression Evaluator Log	Displays information on the configuration setup, working directory, testbench details, variables, and parameters of the expression evaluator process for the specified point.

© 2023 Cadence Design Systems, Inc.

Debug Utility Form

The Debug Utility form lets you set up options that help you save information useful for debugging of simulation runs. You can also specify certain next steps to be performed in case of a simulation failure.

Field	Description
Enable verbose logging in job logs and CDS log	Logs of the following details in the job log and CDS log: ■ Debug messages with a timestamp in each message ■ Callback evaluation information ■ Full stack trace if an error is thrown ■ Debug information for Monte Carlo, calcVal, wave spec evaluation or re-evaluation ■ DE integration debug information and all calls to SKILL functions issued by the UI ■ runObjFile and interactive plotting operations ■ Debug information for OCEAN The log files are saved in the current directory.
Job Log Directory	Specifies the location in which to save the job logs. This field is useful when you select the *Enable verbose logging in job logs and CDS log* field and want to write the job log files to a different location so that they can be shared for further debugging.
Dump Shell and cdsenv Variables	Specifies the path to the file where you want to save the values of the shell variables and environment variables as they are set in the current session. For the environment variables, by default the tool saves values for the variables from the asimenv, auCore, adexl and maestro product categories. You can change this product list by using the `envVarCategoriesForDump` environment variable. **Note:** Only the environment variables that have been changed from their default values are written to the specified file. Environment variable: envVarCategoriesForDump

© 2023 Cadence Design Systems, Inc.

Field	Description
Save	Saves the shell and cdsenv variables in the file specified by the *Dump Shell and cdsenv Variables* field.
GUI Process ID	Displays the process ID of the current ADE Explorer or ADE Assembler window.
Host Name	Displays the host name of the computer on which you are running Virtuoso.
Options specifying how to proceed in case of failures	Contains options that you can use to specify how you want to proceed in case of failures.
Confirm to proceed when job launch failure counts exceeds	Specifies the number of attempts of failed job launch after which an error message is displayed and the tool proceeds with the next job. Environment variable: `maxJobFailPerPolicy`
Halt batch processing when job log failure count exceeds	Specifies the number of attempts of failed job launch after which the batch processing is stopped. Environment variable: `maxJobFailPerPolicyInBatch`
Maximum number of attempts to resubmit a failed simulation	Specifies the number of times ADE Explorer or ADE Assembler should try resubmitting a failed simulation. Environment variable: `numRetriesOnError`
LSCS debug options	Contains options that help in gathering information that you can analyze while troubleshooting the simulations run in LSCS job control mode. You can view the results of these LSCS debug options in the job log for a data point. To access the job log, in the context-sensitive menu of a data point, click *View Log Messages* and choose *Job Log*.

© 2023 Cadence Design Systems, Inc.

Field	Description
Run checkSysConf when simulator job starts	Runs the `checkSysConf` command before starting a simulation job on remote computers to provide information whether the configuration of those computer resources meets Cadence requirements. The tool dumps the result of the `checkSysConf` command for each job in its log file that is saved in the *Job Log Directory* specified on the Debug Utility form. Use this option when looking for additional information for jobs that are failing on some computers and you want to check whether the software on those computers is compatible.
Run pstack when simulator job is not responding for __ seconds	Runs the pstack command for the process running a particular simulation job. Use this option to get the stack trace for a process that remains unresponsive for the specified number of seconds. Depending on your requirements, you can modify the time limit for this option.
Enable Dry Run for Command Distribution Method	Submits the number of jobs specified in the *Max. Jobs* field of the Job Policy Setup form without actually running the processes. This lets you check the network health before running a simulation. This maximum limit for the number of jobs is 100. This option is applicable only when the specified job distribution method is `Command`.

© 2023 Cadence Design Systems, Inc.

Running Simulations With ICRP Job Control Mode

The ICRP job control model uses the IC Remote Processes that are started by ADE Assembler in the `virtuoso -nograph` mode. These process in turn, may start a few child processes (clsbd, oaFSLockD, cdsNameServer, cdsServIpc) to perform some special tasks.

ADE Assembler internally uses ICRP jobs to efficiently distribute time-consuming tasks that can be performed in parallel. Settings for these jobs such as how many remote processes to start; where the processes should run, on local or remote computers; or the time for which a remote process should stay active and wait for a simulation to run; are set as a job policy.

For more information on managing the ICRP job control mode, see the following topics:

- Specifying a Job Policy Name
- Optimizing a Single Point Run
- Specifying a Distribution Method
- Specifying Max Jobs
- Specifying a Command for DRMS
- Specifying an LBS Job Policy
- Specifying Job Timeouts for ICRP
- Specifying Error Reporting Options
- Specifying Multiple Run Options
- Setting Up Resource Options

Specifying Multiple Run Options

To customize options to be used when running multiple runs in the same ADE Assembler session, you need to specify the options in the *For Multiple runs* group box of the *Simulation* tab of the Job Policy Setup form.

© 2023 Cadence Design Systems, Inc.

To specify the job option to be used when running multiple runs in the same ADE Assembler session, do one of the following in the *For Multiple Runs* group box.

- To cause the program to reassign a completed job for the current run to a new run, select the *Reassign immediately for new run* option.
- To cause the program to wait until all the jobs for the current run are completed before assigning the jobs to a new run, select the *Wait until currently running points complete* option.

Also see: Setting Up Run Options

Setting Up Resource Options

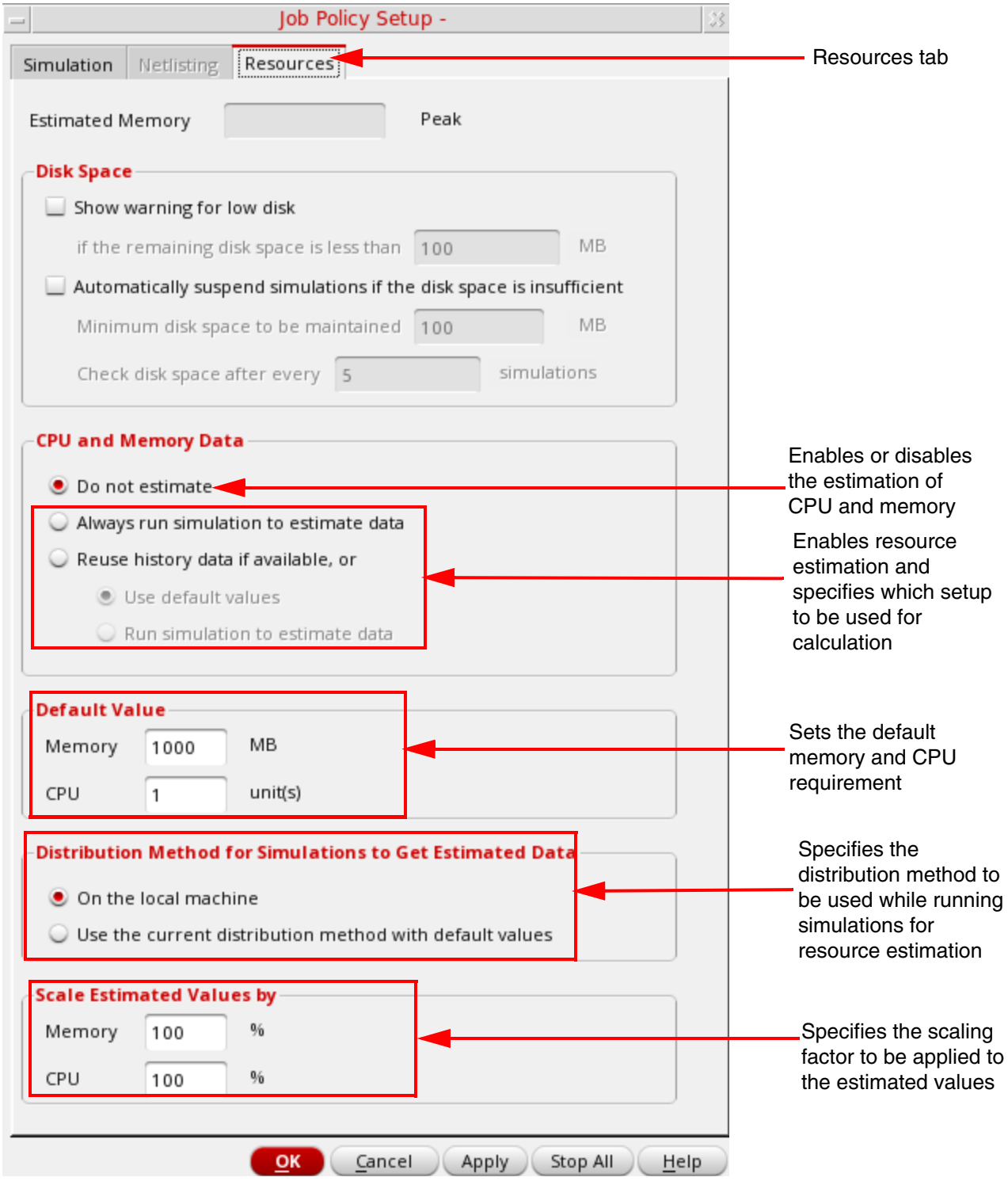
ADE Assembler provides a feature to estimate the memory and CPU resources required to run simulations. After completing the simulation setup, you can use this feature to run resource estimation for the defined tests, and ensure that the required memory and CPUs are available on the local or remote computer as defined by the job policy setup.

Important

This feature can be used only for the **hb** or **tran** analyses, or for **pss** and **qpss** when the time-domain shooting engine is used for these two analyses, that is when the *Engine* property for the **pss** and **qpss** analyses is set to `Shooting` on the Choosing Analyses form. This feature is supported by MMSIM 14.1 ISR8 or later releases.

© 2023 Cadence Design Systems, Inc.

By default, the resource estimation feature is disabled. You can enable and configure it using the *Resources* tab on the Job Policy Setup form shown below.



© 2023 Cadence Design Systems, Inc.

The following table describes the commands given on the Resources tab.

Form Field	Description
Estimate Memory	Shows the estimated memory requirement to run the current simulation run. This feature is enabled using the options given in the *CPU and Memory Data* group box.
CPU and Memory Data	
- Do not estimate	Disables estimation of processors and memory required for the current simulation. In this case, you can yourself specify the required number of processors to be used to distribute simulations. **Note:** When you disable estimation for LBS DRMS, select the check box given to the right of the *Parallel Num. Processors* field on the *Basic* tab of the Job Policy Setup form and specify a value in it. If you clear the check box given to the right of the *Parallel Num. Processors* field, the value given in the field is also cleared. In such a case, you should select any one of the other two options in this field to get an estimate of resources required.
- Always run simulation to estimate data	Enables the feature and runs a simulation before every new simulation to estimate resources. **Note:** If, due to some reason, the simulation that estimates resources does not return a valid result, which is a positive number, ADE Assembler uses the default values specified in the *Default Value* section on this tab.
- Reuse history data if available	Enables the feature and estimates resources based on the data available for the most recent history available in the current ADE Assembler session. If no history data is available, either use the default values specified in the *Default Value* group box, or runs a simulation to estimate values.

© 2023 Cadence Design Systems, Inc.

Form Field	Description
Default Value	
- Memory	Specifies the default memory requirement in Mbytes Default value: `1000 Mbytes`
- CPU	Specifies the default number of CPUs required Default value: `1 processor` If you know the resource requirements for the simulations you are running, you can specify the default values and select the Use Default Values option in the *Provide CPU and Memory Data* group box.
Distribution Method for Simulations to Get Estimated Data	
- On the local machine	Specifies that the simulation for resource estimation needs to be run on the local computer irrespective of the distribution methods set on the *Basic* tab. This option is helpful when you cannot run simulations on a local computer.
- Use the current distribution method with default values	Specifies that the simulation for resource estimation needs to be run using the distribution method specified on the *Basic* tab. This option helps in getting a good estimate of resource requirement because it uses the same remote computers that are required to run simulations.
Scale Estimated Values by	
- Memory	Specifies the value by which the estimated value of memory requirement should be scaled Default value: `100%`
- CPU	Specifies the value by which the estimated value of CPU requirement should be scaled Default value: `100%`

© 2023 Cadence Design Systems, Inc.

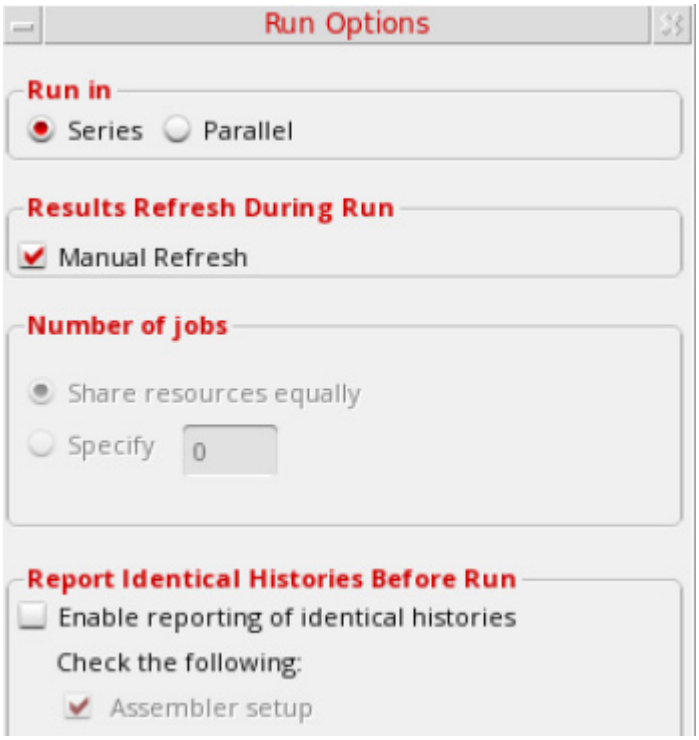
Setting Up Run Options

When you perform multiple ADE Assembler runs simultaneously (that is, an ADE Assembler run is already in progress and you click again to start more runs simultaneously), the options specified on the Run Options form are used to allocate the available ICRPs or jobs. You can change the options on this form to control allocation of the ICRPs among various ADE Assembler runs.

In addition, the form also provides an option to enable checking and reporting of identical histories before running simulation. For more details, refer to Running an Incremental Simulation to Reuse Results.

To set up run options, in the ADE Assembler window, choose *Options – Run Options*.

The Run Options form appears.



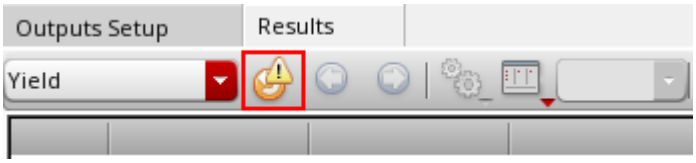
Note: The *Results Refresh During Run* section is hidden by default. You can show it by setting the manualResultsViewRefresh environment variable is set to `t`.

© 2023 Cadence Design Systems, Inc.

The following table describes the commands available on the Run Options form:

Command	Description
Run In group box	Contains the options to specify how more than one ADE Assembler runs are to be performed
Series	Performs multiple ADE Assembler runs in series. In this case, only one run is performed at a particular point of time. If the number of available ICRPs (as specified by *Max Jobs* in the job policy) is greater than the number of points in the current ADE Assembler run, only the ICRPs required to run the points are used. The remaining ICRPs are not allocated to any subsequent run until the current ADE Assembler run is complete.
Parallel	Runs multiple ADE Assembler runs in parallel. If ICRPs are available, multiple ADE Assembler runs are started without waiting for the previous one to complete. The allocation of points or corners to ICRPs depends on the setting in the *Number of jobs* section on this form. **Note:** The default value for *Run In* is set to *Series*. To make *Parallel* as the default value, set the defaultRunInParallel environment variable to `t`.
Results Refresh During Run	Contains the option to control the display of results while a simulation run is in progress **Note:** This section is visible only when the manualResultsViewRefresh environment variable is set to `t`.

© 2023 Cadence Design Systems, Inc.

Command	Description
Manual Refresh	When selected, the *Yield* view of the *Results* tab is not updated while the run is in progress. A command, *Refresh to view results*, is added to the toolbar of the *Results* tab. You can use this command to show or update the results at any time during or after the run.  When you clear the *Manual Refresh* check box, the results are automatically displayed and refreshed on the *Results* tab while the simulation is in progress. You can control the state of the *Manual Refresh* check box by using the `enableManualRefresh` environment variable **Note:** The *Refresh to view results* command is available only on the `Yield` results view.
Number of jobs group box	Contains the options to specify how the ICRPs are to be allocated among multiple ADE Assembler runs.
Share resources equally	When *Run In* is set to *Parallel*, this option shares the number of jobs among all the parallel runs equally. For example, if *Max Jobs* is `10` and the current run is using `4` ICRPs, two additional parallel runs will each get `3` ICRPs.

© 2023 Cadence Design Systems, Inc.

Command	Description
Specify	When *Run In* is set to *Parallel*, this option specifies the maximum number of ICRPs to be allocated to each ADE Assembler run in parallel. **Note:** The value in the *Specify* field should not exceed the value of *Max Jobs* because in that case, the value in *Specify* is ignored. If the value of *Max Jobs* is greater than the value in *Specify*, each ADE Assembler run gets the same number of ICRPs as specified in the *Specify* text box. **Note:** You can change the value of *Specify* between two ADE Assembler runs. For example, *Max Jobs* is set to `10`. Before starting the first run, *Specify* is set to 4, so it gets four ICRPs. After this, if you change the value of *Specify* to 2 and start two more runs, the subsequent runs will get two ICRPs each. Also see: Examples of ICRP Allocation to Parallel Runs Based on Specify and Max Jobs
Report Identical Histories before Run group box	Contains the options to detect changes in the simulation setup before running simulations. For details, refer to Reporting Identical Histories Before Running a Simulation.

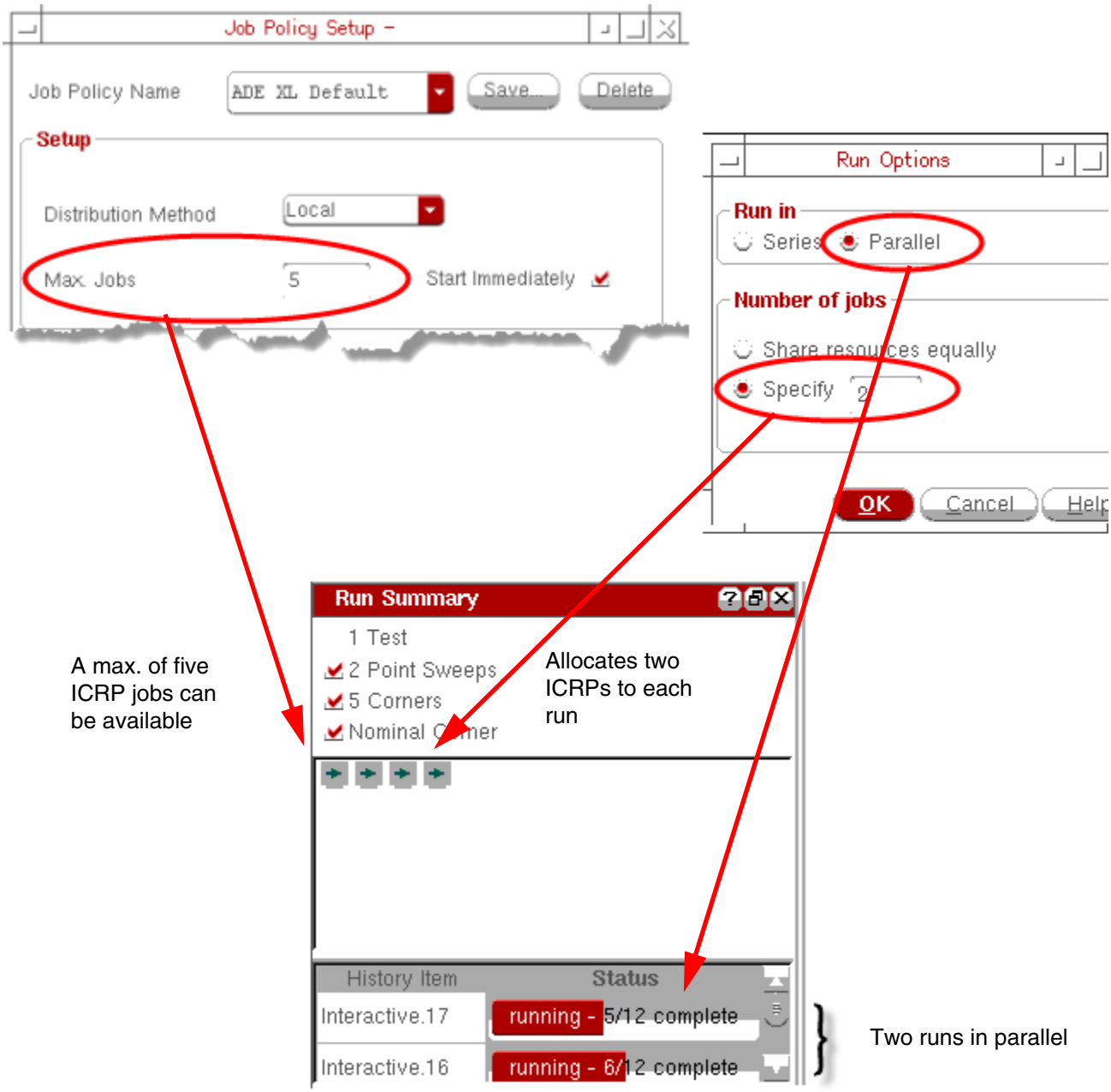
Examples of ICRP Allocation to Parallel Runs Based on Specify and Max Jobs

Example 1

If *Max Jobs*=`5` and *Specify*=`2`, the first ADE Assembler run will get two ICRPs and the subsequent runs will also get two ICRPs allocated to them. If there are only two runs in

© 2023 Cadence Design Systems, Inc.

parallel, one ICRP still remains unallocated, as shown in the figure given below.



Example 2

If *Max Jobs*=6 and *Specify*=4, the first ADE Assembler run will get four ICRPs. The next run initially gets the remaining two ICRPs. Depending on the value set for *Reassign*

© 2023 Cadence Design Systems, Inc.

immediately for new runs job policy setting, ADE Assembler reassigns one of the ICRPs from the first run to the second run so that each run is using three ICRPs.

Example 3

If *Max Jobs*=`24` and *Specify*=`8`, the first run gets eight ICRPs. The next run will get another eight ICRPs. Eight more ICRPs still remain unallocated.

Reporting Identical Histories Before Running a Simulation

Tip

Read a blog related to this at Virtuosity: Preventing Redundant Simulations.

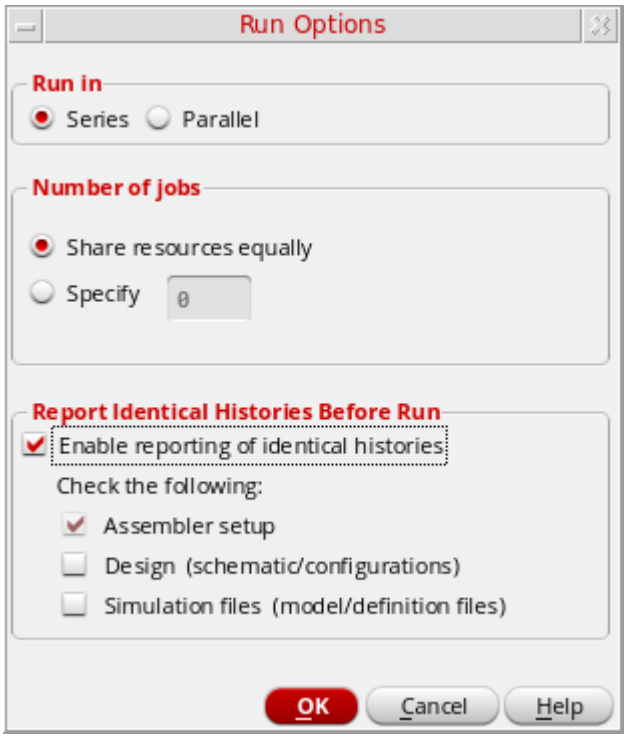
By default, before running a simulation, ADE Assembler does not compare the setup details with any of the existing histories. This can result in running more than one simulation with the same setup. For a resource-intensive simulation, this can block the resources and affect performance. To avoid this, you can use the Run Options form and select the options to compare the active setup with that of existing histories, and run simulation only if the setup is unique. If any identical history is found, it is reported.

To enable reporting of identical histories before running a simulation, perform the following steps:

1. Choose *Options – Run Options*.

© 2023 Cadence Design Systems, Inc.

The Run Options form is displayed.



2. Select the *Enable reporting of identical histories* check box.

   All the check boxes given below this get enabled. These check boxes specify the type of changes you want to identify while comparing the active setup with a history.

   The following table provides more details about the checks.

Type of changes	Checks for
Assembler setup	Changes in setup database, such as test details, corners, device parameter, run mode and run options. This is selected by default. **Note:** While checking for identical histories, ADE Assembler does not compare the settings specified by the reportIdenticalHistoryIgnoreElementsList environment variable. You can modify the value of this variable to specify the type of elements you want to ignore while checking for identical histories.

© 2023 Cadence Design Systems, Inc.

Type of changes	Checks for
Design (schematic/ configurations)	Changes in the schematic or config view of the design. Hierarchical schematic traversal is done to compare the timestamp of each schematic cellview with that of the maestro setup database.
Simulation files	Changes in simulation files, such as definition file and model file. To compare the simulation files, the tool stores the files obtained by parsing spectre.out of the latest run and compares their file hash with the corresponding file hash in the history database. Any addition or removal of model files is detected along with the detection of changes in the setup database.

3. Select the check boxes for the required type of changes you want to identify before running a new simulation. If no change is found, ADE Assembler shows a message that mentions that the setup details in the active setup are same as those of a history and prompts you to confirm that you still want to run a simulation. Other types of changes, for which the corresponding check boxes are not selected, are ignored.
4. Click *OK*.

Important Points to Note

- The settings on the Run Options form are applicable for the current ADE Assembler session.
- This feature is supported only for Single Run, Sweeps, and Corners, Monte Carlo, Global Optimization, and Local Optimization run modes. If any other run mode is selected in the ADE Assembler setup, the *Enable Reporting of Identical Histories* check box on the Run Options form is disabled.
- By default, this feature is not enabled for any run mode. You can enable it for any one of the supported run modes (Single Run, Sweeps and Corners, Monte Carlo Sampling, Global Optimization, Local Optimization, Sensitivity Analysis) by setting the runModesForIdenticalHistoryCheck environment variable.
- This feature is supported for DSPF cellviews.
- This feature is supported for text cellviews stored in HDL files, such as Verilog, VerilogA, VerilogAMS, SystemVerilog, VHDL, VHDLAMS, PSpice, HSPICE, Spectre, and SPICE

© 2023 Cadence Design Systems, Inc.

text files, in Virtuoso libraries. However, note that it compares only the master text cellview, and ignores any changes in the text files included or referenced in the master cellview.

- If you want ADE Assembler to print in the CIW log the differences found in the active setup and the setup of a saved history, set the reportChangesAfterComparingSetupWithHistories environment variable to `t`.

© 2023 Cadence Design Systems, Inc.

Running Pre-run Scripts before Simulation Runs

For each test, you can specify a pre-run script that contains a set of OCEAN commands to be run before the test is simulated. If a pre-run script is specified for an ADE Assembler test, for every point of evaluation, first the pre-run script is run and then the test is simulated for that point.

Using pre-run scripts, you can specify OCEAN commands that read the simulation setup for a test, run pre-simulation analyses for each simulation point and, if required, modify the simulation setup for that point. For example, you can run a script to calculate a calibrated value for a design variable and use that value in the main simulation run. To see an example, refer to Pre-run script Example for Monte Carlo Calibration.

In a pre-run script, you can use any OCEAN command. For example, you can use the `desVar` command to get the value of a design variable in an ADE Assembler test. To see how, refer to Pre-run Script Example. In addition, you use the following OCEAN commands that are specifically meant for use in pre-run scripts:

- ocnxlLoadCurrentEnvironment
- ocnxlSetCalibration
- ocnxlRunCalibration
- ocnxlAddOrUpdateOutput
- ocnxlUpdatePointVariable
- ocnxlGetJobId
- ocnxlGetPointId
- ocnxlMCIterNum
- ocnxlMainSimSession

A pre-run script is run in non-graphical mode. Therefore, in this script, you cannot specify OCEAN commands that plot waveforms.

Note: It is recommended to deselect *Optimize Single Point Run* in job policy setup when a pre-run script is used and the setup has a single test point.

Related Topics

- Adding a Pre-Run Script on page 937
- Modifying a Pre-Run Script on page 940

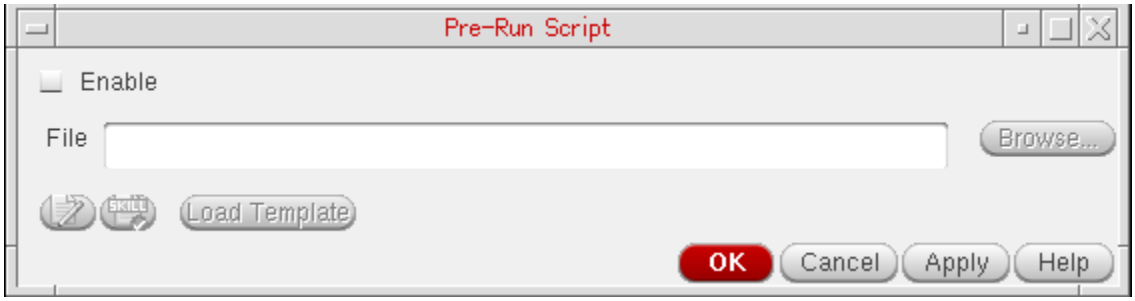
© 2023 Cadence Design Systems, Inc.

Adding a Pre-Run Script

To add a pre-run script for a test, do the following:

1. In the Data View pane, right-click the test and choose *Pre-Run Script*.

   The Pre-Run Script form appears.



2. Click *Enable* to enable the use of a pre-run script for the test.

   All other fields and buttons on the form are enabled.

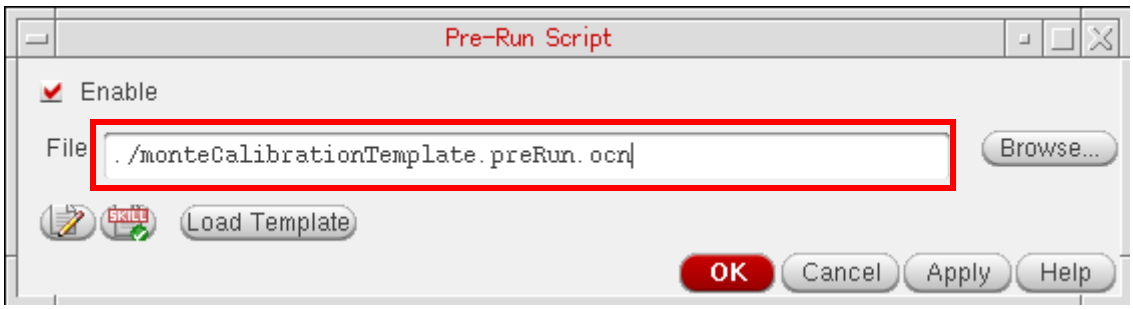
3. Do any of the following to provide a pre-run script:

   - If a script is already saved in a file, specify the path to the file in the *File* field. You can click *Browse* to view the directories and select a file.

     Now, click (*Edit pre-run script*) to open and edit the script in the default text editor for Virtuoso specified by using the `$VISUAL` or `$EDITOR` variables in the `.cshrc` file, in the given order of preference.

     Note: You can also specify the name of your preferred text editor by using the `editor` SKILL variable.

   - To load a script template file and use it to create a new script, click Load Template. The path to the default template, `monteCalibrationTemplate.preRun.ocn`, is displayed in the *File* field.



© 2023 Cadence Design Systems, Inc.

You can now click *Edit pre-run script* to open and edit the script in the default text editor.

4. After editing, save the script and close it.

   If you used a script template, save your script with a new name.

5. (Recommended) Click (*Lint check pre-run script*) to run SKILL Lint rules and check the scripts for errors.

   The script is displayed in the SKILL IDE where you can run lint rules and modify your script.

6. Save the script and close the SKILL IDE.

7. Click *OK* to set the script for your test.

You can also run the following SKILL functions to set a pre-run script:

- axlSetPreRunScriptEnabled
- axlImportPreRunScript
- axlSetPreRunScript
- axlGetPreRunScript

Note: By default, the pre-run script file specified for a test is copied to the maestro view and is saved as a part of the state. To use the script file from its original location and not to copy it to the maestro view, set the copyPreRunScripts environment variable to `nil`.

The following examples show how pre-run scripts can be used.

Example 17-1 Pre-run Script Example

The following pre-run script reads the simulation setup of a test and before every point of evaluation, it runs a transient simulation for a selected set of sweep values of design variable `R1`. Based on the simulation result, it updates the value of another variable `DC1` before running the main simulation for that point.

```
; Read the simulation setup for the test
ocnxlLoadCurrentEnvironment(?noAnalysis t)

; If design variable R1 is swept from 10K to 30K, the following logic will run only
; for the points where R1 > 20K
when(evalstring(desVar(“R1”)) > 20K
    resultsDir(“./newDir”)
    save( 'i "/NM2/D" "/NM0/D" )
        analysis( ‘tran ?stop “10u”)
```


© 2023 Cadence Design Systems, Inc.

```
        run( )
    SimRes=xmax(i("NM2.d" ?result "tran-tran" ?resultsDir "./newDir/psf") 1 ))


; Update value of variable DC1 in the simulation setup for the point based on
; whether R is greater than or lesser than 90u.
    a=110u
    b=120u
    if( SimRes > 90u then
    ocnxlUpdatePointVariable("DCI" sprintf( nil "%L" a))
    else
    ocnxlUpdatePointVariable("DCI" sprintf( nil "%L" b))
    )
)
```

Example 17-2 Pre-run script Example for Monte Carlo Calibration

The following pre-run script customizes the simulation setup of a test and sets up a single iteration Monte Carlo calibration run by inheriting the settings from the main Monte Carlo Sampling run. Next, it calculates a calibrated value that is used in the main simulation.

In the Pre-Run Script form, click *Enable*. Next, click *Open Template*. ADE Assembler loads the *monteCalibrationTemplate.preRun.ocn* template.

Click (*Edit pre-run script*) to open and edit the script in the default text editor for Virtuoso. The following template script is opened.

```
; Read the simulation setup for the test Test1
ocnxlLoadCurrentEnvironment( ?noAnalysis t)


; The following command changes the simulation setup for the calibration run by
; specifying a new analysis for the calibration run
    analysis('ac ?start "1G"  ?stop "3G"  ?step "1M")


; The following command sets up a single iteration Monte Carlo calibration run by
; inheriting the settings from the main Monte Carlo Sampling run
    ocnxlSetCalibration()


; The following logic calculates the calibrated value using the successive
; approximation method. The ocnxlRunCalibration() runs a single iteration Monte
; Carlo simulation.
    for( n 1 noOfBits
        i = noOfBits - n
        BitWord = BitWord + 2**i
        ParamVal = BitWord*8
        desVar( "ParamName" ParamVal )
        ocnxlRunCalibration()
        SimResult = <Evaluate Expression>
```


© 2023 Cadence Design Systems, Inc.

```
        if(  SimResult < target
            then
            BitWord = BitWord - 2**i
        )
    )
    CalResult = BitWord * 8


; The following command adds the calibrated value as an output named
; Calibrated_ParamName so that it can be viewed in the ADE Assembler Outputs Setup
tab for
; each point.
    ocnxlAddOrUpdateOutput("Calibrated_ParamName" CalResult)


; The following command updates the value of global variable ParamName with
; the calibrated result.
    ocnxlUpdatePointVariable("ParamName" sprintf( nil "%L" CalResult))
```

After assigning the calibrated values to the output of a test, `Test1` in the above example, you can also pass the calibrated value to a global variable in another test, for example, to variable `N2` in `Test2`. In this case, you can define variable `N2` for `Test2` using the `calcVal` function, as shown below:

```
calcVal("Calibrated_ParamName" "Test1")
```

Important

It is not possible to distribute a simulation running from a pre-run script. If you specify a distribution setup in the script or in `.cdsenv`, it is ignored. For LSCS, the simulations in pre-run scripts are distributed with the netlisting jobs as per the netlisting job policy set in ADE Assembler. For ICRP, the simulations in pre-run scripts are distributed as per the simulation job policy set in ADE Assembler.

Modifying a Pre-Run Script

To modify the pre-run script for a test, do the following:

1. In the Data View pane, right-click the test and choose *Pre-Run Script*.

   The Pre-Run Script form appears.

2. Verify that the *File* field shows the path to the required script file name.
3. Click (*Edit pre-run script*).

© 2023 Cadence Design Systems, Inc.

The script file is opened in the default text editor where you can edt the script and save the changes.

© 2023 Cadence Design Systems, Inc.

Using the Run Preview Table

Video

You can view a video demonstration for this feature at Run Point Selection. Also read a blog related to this at Virtuoso Video Diary: Run Point Selection -WYSIWYG.

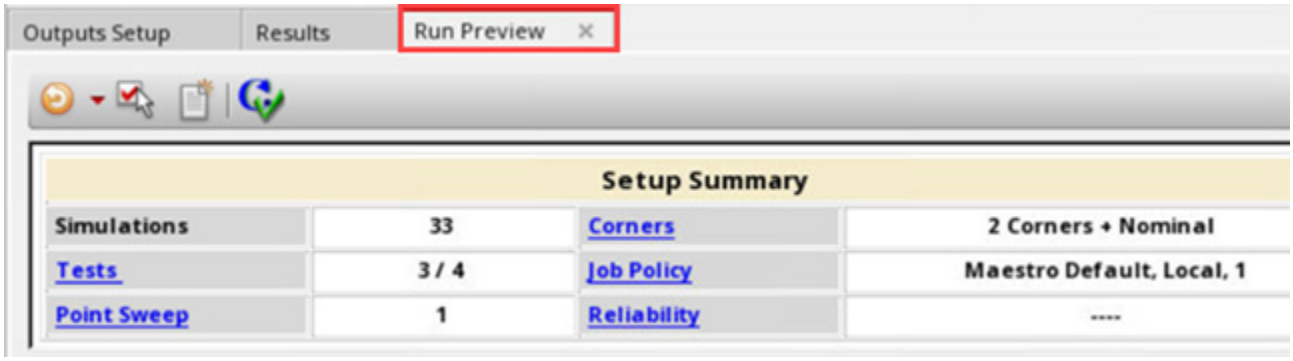
The *Run Preview* tab provides a summary of all the settings applied in the current maestro cellview. You can view various details, such as the total number of sweep points and corners that are run, the name of the test, or the job policy being used. It also displays the combinations of sweeps and corners that will be run. You can view a particular data point in the debug environment and run a simulation for that point. In addition, you can create netlist for a particular point. On this tab, you can also select specific points for which you want to run simulations.

Note: For Monte Carlo Sampling run mode, the *Run Preview* tab is visible only when you choose the simulation for a fixed number of points. This tab is not visible when you run a simulation to verify yield or to create statistical corners.

By default, the *Run Preview* tab is not displayed. To open this tab,

- Right-click anywhere on the tab bar of the Outputs Setup pane and choose *Run Preview*.
- Set the `outputTabsShowDefault` environment variable.

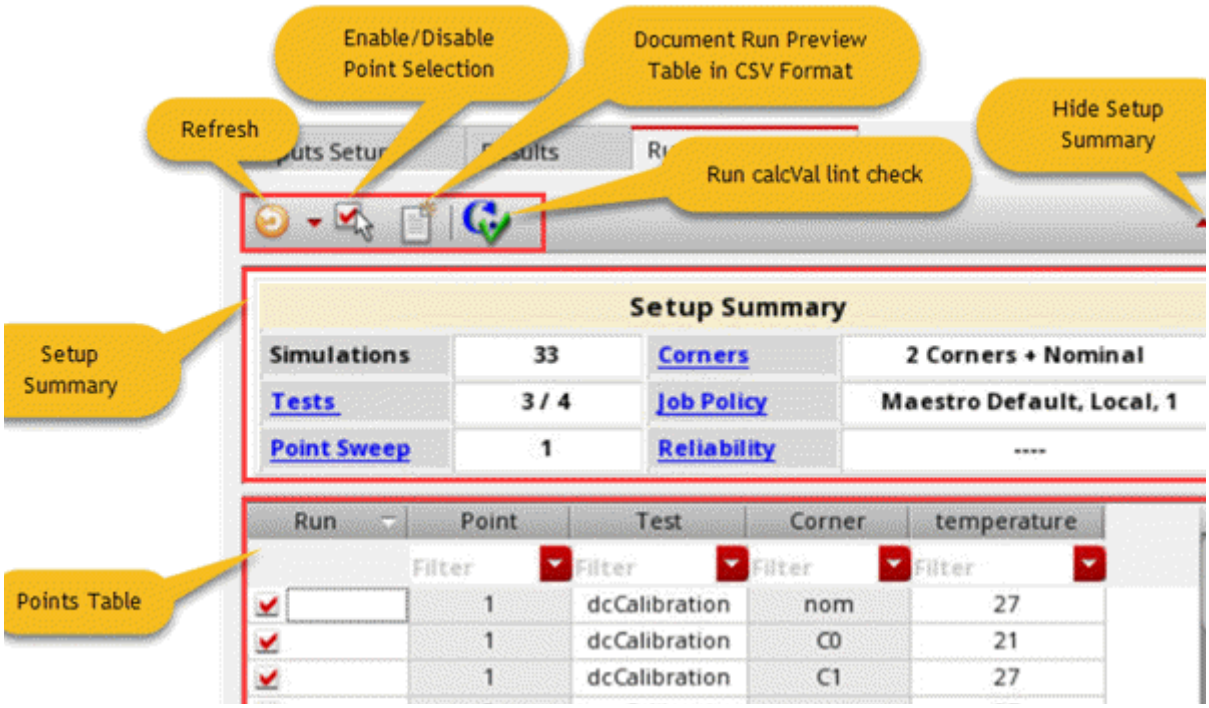
Run Preview is displayed to the right of *Results* in the Outputs Setup tab.



The *Run Preview* tab has two sections—*Setup Summary* and points table, as shown in the figure below. The design point and corner combinations, and the values of variables are

© 2023 Cadence Design Systems, Inc.

displayed in the points table. You can also enable or disable run points in the points table. For more information, see Enabling or Disabling Run Points in Points Table.



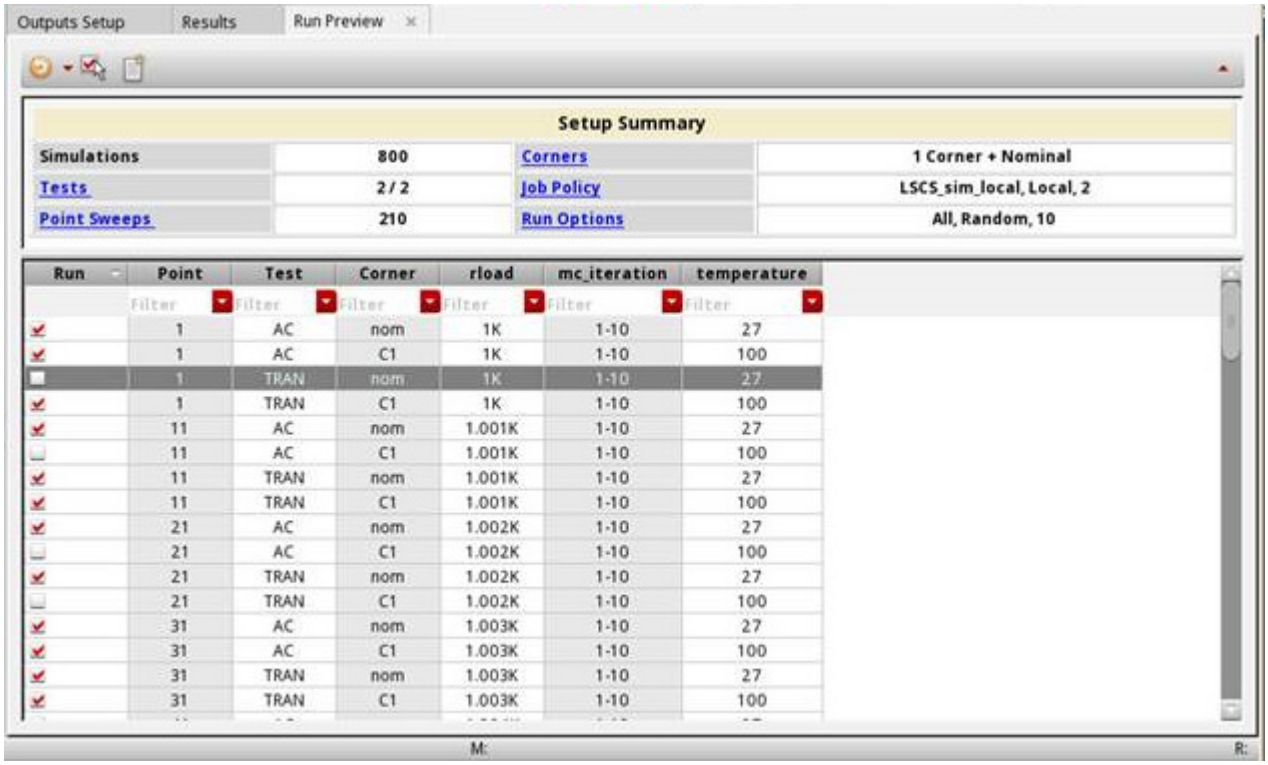
You can click the arrow button on the upper-right corner to show or hide the *Setup Summary* section. To find a specific point or set of points in the points table, you can sort the data displayed in each column by clicking the arrow button available for each column header.

You can also click the *calcVal lint check* button to run pre-simulation checks for all enabled `calcVal` expressions. You can view the resulting messages of the lint checks in the CIW. These checks are run only for the enabled output that contains `calcVal` expressions.

The *Run Preview* tab can be displayed only for the single run, sweeps and corners simulation or the Monte Carlo simulation for which the option to run a fixed number of points is selected. All Monte Carlo iterations for a particular point, test, corner, and variable

© 2023 Cadence Design Systems, Inc.

combination are combined in a single row where the *mc_iteration* column shows the range of points.



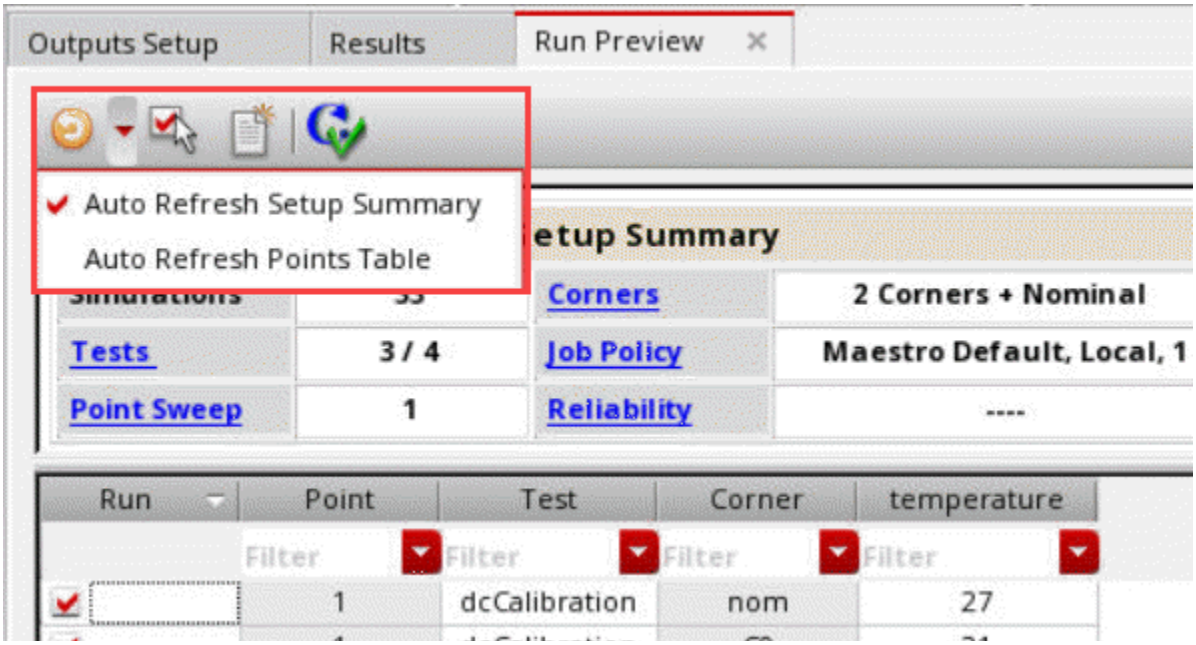
The *Run Preview* tab is not displayed when reliability analysis is enabled.

Refreshing the Setup Summary and Points Table

As you modify the setup in the current view, the setup summary is refreshed automatically and the count of corners, points, and total simulations to be run keeps varying. By default, only the summary data is displayed and updated. This is because the *Auto Refresh Setup*

© 2023 Cadence Design Systems, Inc.

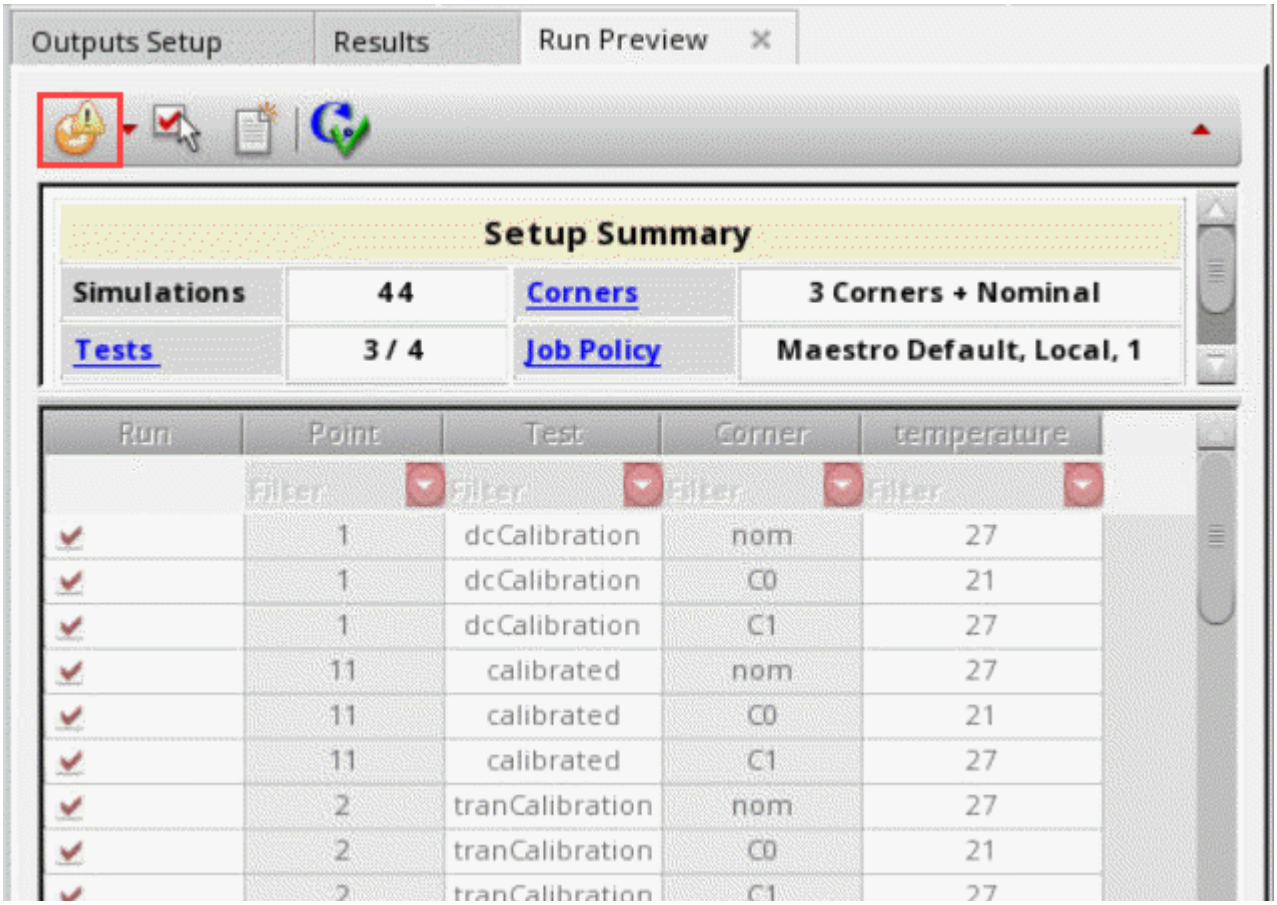
Summary check box in the drop-down menu is selected by default. You can click the *Refresh* button to display the details of the run points in points table.



When you open a new setup or move from ADE Explorer to ADE Assembler and vice-versa, the points table is not displayed by default. You need to click *Refresh* to view the points table. In addition, the points table is not automatically refreshed and becomes unavailable when the

© 2023 Cadence Design Systems, Inc.

setup changes. In this case, an alert icon is displayed on *Refresh* to indicate that the table data is not current. You can click *Refresh* to update the table contents.



To refresh the points table automatically after every change in the sweep values, corner, or any other setup changes:

- Set the `enableAutoRefreshPointsTable` environment variable to `t`.
- Click the *Refresh* drop-down list and select the *Auto Refresh Points Table* option.

If the number of the simulation points is large, it may take several minutes to display the details.

Important

If the *Auto Refresh Points Table* check box is selected, the simulation run triggers refreshing the points table. If the number of the simulation points is large, it may take more time for the simulation run to complete. Therefore, it is recommended to clear the *Auto Refresh Points Table* check box before running simulations.

© 2023 Cadence Design Systems, Inc.

To refresh only the points table and not the Setup Summary, you can clear the *Auto Refresh Setup Summary* check box and select the *Auto Refresh Points Table* check box.

Related Environment Variables

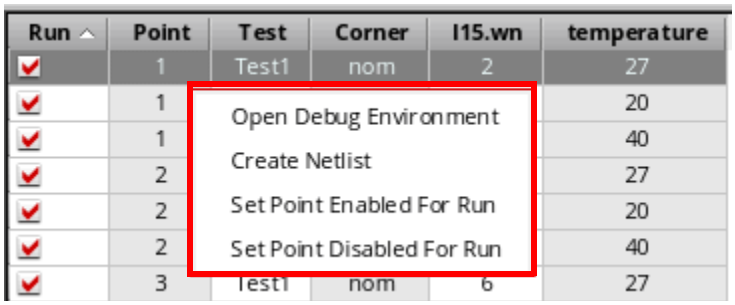
- `enableAutoRefreshSetupSummary`
- `enableAutoRefreshPointsTable`

Enabling or Disabling Run Points in Points Table

The points table shows a *Run* column that includes a check box against each row. By default, the check box is selected for all the points. The *Run* column is used to select or deselect run points for the simulation run.

If you do not want to run a specific point, clear the check box in the *Run* column.

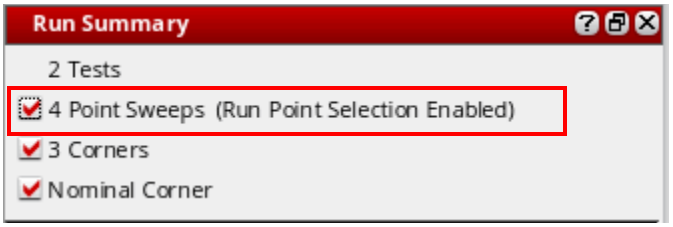
To enable or disable a set of points, hold down the `Ctrl` key and select the points, right-click and choose *Set Point Enable For Run* or *Set Point Disabled For Run*, respectively.



© 2023 Cadence Design Systems, Inc.

To enable or disable all the points, right-click the column title on the Run Preview table choose *Select All Run Points* or *Deselect All Run Points*, respectively.

Note: When you enable or disable a point, the *Simulations* field in *Setup Summary* is updated to indicate the number of simulations that are run. However, the Run Summary assistant still displays the previous data. When you change the run point selection in points table, the Run Summary assistant in ADE Assembler displays the text, `Run Point Selection Enabled`, to indicate that the given count is not current.



Enabling or Disabling Run Column

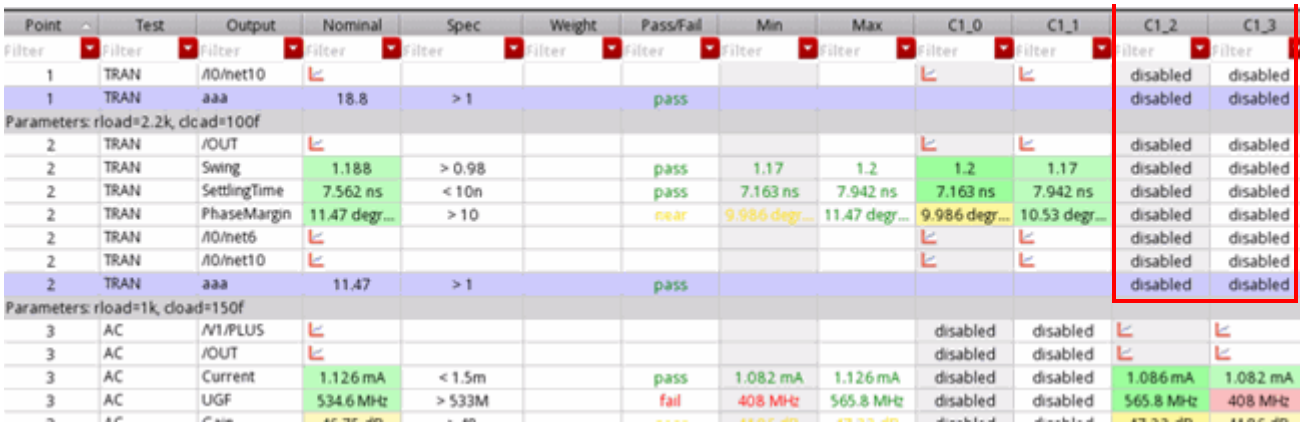
To completely disable the *Run* column and run point selection,

➡ Click .

The *Run* column becomes unavailable. In this case, all the points will be run irrespective of the selection you made previously. To enable the *Run* column, click this button again.

© 2023 Cadence Design Systems, Inc.

Note: When you disable corners or sweeps, the results in `Detail` and `Detail-Transpose` views display `disabled` in the disabled sweep-corner cell, as shown in the figure below.



Point	Test	Output	Nominal	Spec	Weight	Pass/Fail	Min	Max	C1_0	C1_1	C1_2	C1_3
1	TRAN	/I0/net10									disabled	disabled
1	TRAN	aaa	18.8	> 1		pass					disabled	disabled
Parameters: rload=2.2k, cload=100f												
2	TRAN	/OUT									disabled	disabled
2	TRAN	Swing	1.188	> 0.98		pass	1.17	1.2	1.2	1.17	disabled	disabled
2	TRAN	SettlingTime	7.562 ns	< 10n		pass	7.163 ns	7.942 ns	7.163 ns	7.942 ns	disabled	disabled
2	TRAN	PhaseMargin	11.47 degr...	> 10		near	9.986 degr...	11.47 degr...	9.986 degr...	10.53 degr...	disabled	disabled
2	TRAN	/I0/net6									disabled	disabled
2	TRAN	/I0/net10									disabled	disabled
2	TRAN	aaa	11.47	> 1		pass					disabled	disabled
Parameters: rload=1k, cload=150f												
3	AC	/V1/PLUS							disabled	disabled		
3	AC	/OUT							disabled	disabled		
3	AC	Current	1.126 mA	< 1.5m		pass	1.082 mA	1.126 mA	disabled	disabled	1.086 mA	1.082 mA
3	AC	UGF	534.6 MHz	> 533M		fail	408 MHz	565.8 MHz	disabled	disabled	565.8 MHz	408 MHz

The completely disabled rows and columns are not displayed in the results.

Some Important Points to Note

- Measurements across corners, sweeps, or all ignore the disabled run points.
- The run point selection is reset and the points table is disabled when you transition between ADE Explorer to ADE Assembler and vice-versa.
- The run point selection is not considered in Saved Ocean XL scripts.
- Any setup changes in the variables, parameters or corners will reset the check boxes in Run column and select all of them.
- The run point selection setup is persistent and is saved and loaded along with view.
- By default, the columns for fixed parameters are not displayed in the points table on *Run Preview* tab. However, if the fixed parameters are enabled in the *Results* tab for the Detail - Transpose view, they will be displayed in the points table as well. To display columns for fixed parameters on *Results* tab, you can choose the *Fixed Parameters* option in the *Configure what is shown in the table* drop-down list available on the toolbar.

Using the Information Displayed on the Run Preview Tab

You can use the information displayed on the *Run Preview* tab as explained below.

© 2023 Cadence Design Systems, Inc.

Viewing the Corner and Sweep Combinations

If you have selected the check boxes that enable automatic refresh of information, the details on the *Run Preview* tab show the updated corner and point details while you modify the ADE Assembler setup.

Modifying the Setup

You can click the hyperlinks provided in the *Setup Summary* section to open the relevant section in the ADE Assembler setup and modify the settings. For example, when you click the *Corner* hyperlink, the Corners Setup form is displayed where you can modify the corners for the current setup. After you close the Corners Setup form, the number of corners is updated according to the changes done.

Similarly, you can click the *Job Policy* hyperlink to open the Job Policy Setup form to view or modify the job policy. If any test has an overridden job policy, the Job Policy information in the Setup Summary indicates that by using the keyword `(various)`, as shown in the figure above.

Creating Netlist

You can create netlist for a specific design point and corner combination and verify details. For this, right-click anywhere in the row corresponding to that combination and choose *Create Netlist*. ADE Assembler creates and displays a netlist for that point in a separate window.

Running Simulation for a Specific Point

Before running simulation for all the points, you might be interested in viewing the results of a specific design point and corner combination. In that case, you can run a simulation for that combination only and verify results. If required, make further modifications in the setup. To run simulation for a specific design point and corner combination, right-click anywhere in the row corresponding to that combination and choose *Manage Point Results – Open Debug Environment*. ADE Assembler opens that simulation point in the ADE Assembler Debug Environment where you can run a simulation and view results.

Note: The *Create Netlist* and *Open Debug Environment* commands are not enabled for the OCEAN-based tests.

© 2023 Cadence Design Systems, Inc.

Documenting Point Details

Click *Document Run Preview Table in CSV Format* on the toolbar of this tab to document the details of points in a `.csv` file for future reference.

Modifying the Format of the Run Preview Tab

The format of information displayed on the *Run Preview* tab is defined by the `previewSetup.xsl` stylesheet saved at the following default location:

`<Virtuoso-installation-directory>/share/cdssetup/adexl/previewSetup.xsl`

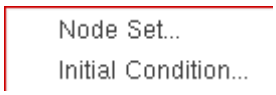
To modify this format or to define the content to be displayed on this tab, you can provide the path to a customized `.xsl` stylesheet by setting the `RUNPREVIEW` shell environment variable before running Virtuoso, as shown below.

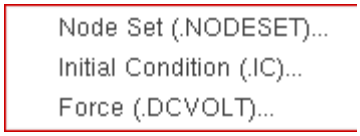
```
setenv RUNPREVIEW_XSL <path-to-cutomized-stylesheet-file.xsl>
```

ADE Assembler uses the stylesheet specified by this variable to display the information on the *Run Preview* tab.

© 2023 Cadence Design Systems, Inc.

Using Convergence Aids





To specify node set information, do the following:

1. On the Data View assistant pane, right-click the test or analysis name and choose *Convergence Aids – Node Set*.

   The appropriate form for your target simulator appears.

2. When you are done specifying node set information, click *OK*.

To specify initial conditions, do the following:

1. On the Data View assistant pane, right-click the test or analysis name and choose *Convergence Aids – Initial Condition*.

   The appropriate form for your target simulator appears.

2. When you are done specifying initial conditions, click *OK*.

(hspiceD only) To force a node to a particular voltage for the entire simulation, do the following:

To specify node set information, do the following:

1. On the Data View assistant pane, right-click the test or analysis name and choose *Convergence Aids – Force*.

   The Select Force Node Set form appears.

2. When you are done specifying force nodes, click *OK*.

© 2023 Cadence Design Systems, Inc.

Disabling and Enabling All Point Sweeps

To disable all point sweeps, do the following:

➤ On the Run Summary pane, deselect the # *Point Sweeps* check box (where # indicates the number of point sweeps).

The simulator does not run any point sweeps.

To restore all point sweeps, do the following:

➤ On the Run Summary pane, select the # *Point Sweeps* check box (where # indicates the number of point sweeps).

The simulator runs the specified number of point sweeps.

Disabling and Enabling All Corners

To disable all corners, do the following:

➤ On the Run Summary pane, deselect the *Corners* check box.

Corners are not included in simulations.

To enable all corners, do the following:

➤ On the Run Summary pane, select the *Corners* check box.

You can enable or disable simulations for specific test and corner combinations using the Corners Setup form. For details, see Chapter 6, “Working with Corners”.

© 2023 Cadence Design Systems, Inc.

Starting a Simulation

To start a simulation, do the following:

➤ On the Run toolbar, click the *Run Simulation* button.

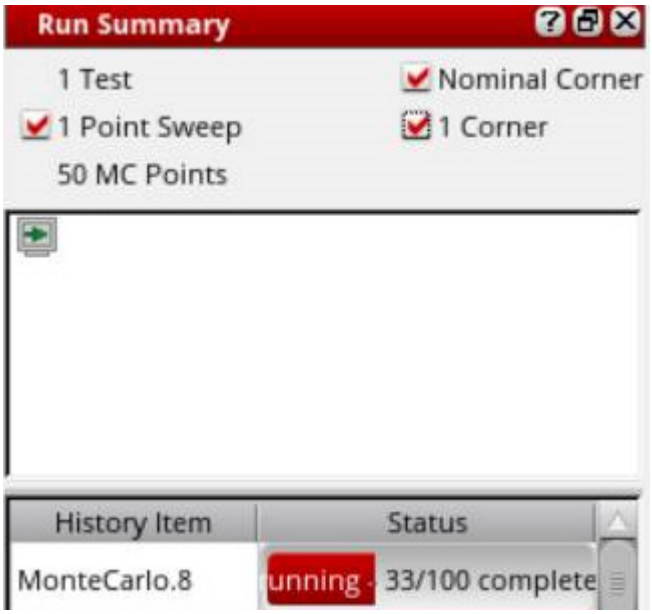
ADE Assembler does the following:

- If the adexl.icrpStartup environment variable is set to `t`, displays the Specify History Name form to prompt for a history name to be used for the current run. For more details, refer to Specifying a Name for Simulation History.
- Checks for new design variables by performing a copy-from-cellview operation.

  The simulator requires a value for every design variable or a global variable that is enabled and has a valid value or set of values.
- Checks if the references in the Global Variables and Corners section are the same as those in the Setup Library Assistant. Use the syncWithSLA environment variable to specify different behaviors.
- Generates a new netlist that reflects any design changes (any changes you made to the cellview since the last extraction).
- Updates the simulation information numbers that appear on the Run Summary assistant pane to indicate the number of tests running, the number of values varying (*Point Sweeps*), the number of corners, and the nominal corner.
- Using the simulator settings and job policy settings, submits the points to the available jobs for simulation.

© 2023 Cadence Design Systems, Inc.

- Displays the progress of the simulation run in the Run Summary assistant pane.

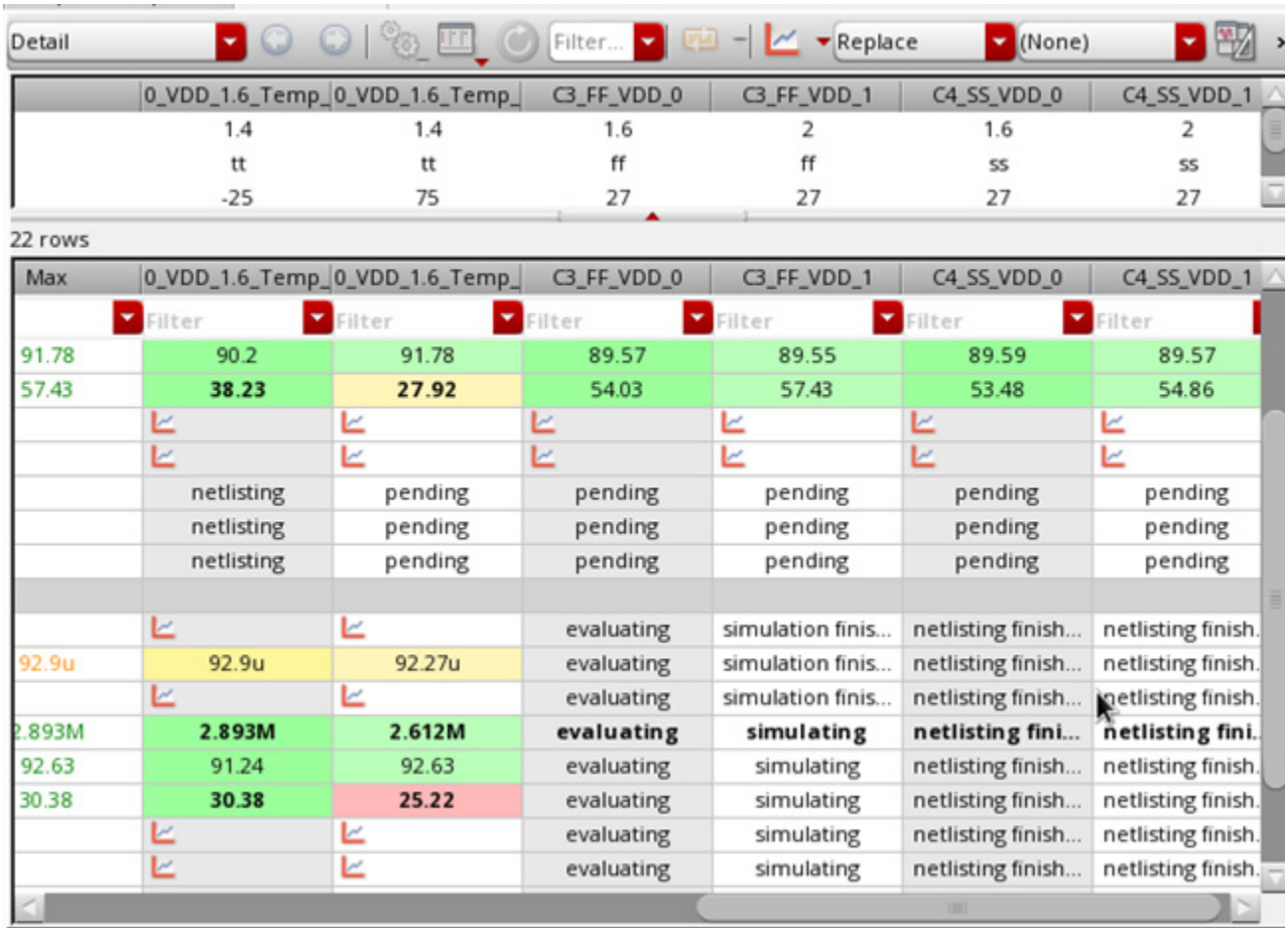


  For more details of the progress bar, refer to Checking Run Status on the Progress Bar.

- Simulates your testbench.
- Saves the results in the results database.
- Displays the progress and results of the simulation run on a new tab in the Results tab of the Outputs pane. The name of the tab matches the name of the checkpoint for the simulation run. For more information, see Working with Tabs for Simulation Checkpoints.
- If simulations are not run successfully, displays appropriate errors. For more information on the types of errors, refer to Simulation Errors on page 1009.

© 2023 Cadence Design Systems, Inc.

When the simulation is in progress, the data points on the *Results* tab show the current stage of simulation.



In some cases, the results status is shown as `Canceled`. This status indicates any one of the following scenarios:

- You have canceled the simulation run by clicking the *Stop Simulation* button.
- The ADE Assembler GUI is closed abruptly.
- There is no sufficient disk space to complete the simulation run.
- If you are running a Monte Carlo simulation, a particular simulation is canceled by ADE Assembler because the target yield level is achieved when the `Yield Verification - Basic AutoStop` method is used.

For details on how to view results on the Results tab and how to customize the view, refer to Viewing, Printing, and Plotting Results.

© 2023 Cadence Design Systems, Inc.

AMS Simulations in ADE Assembler

ADE Explorer and ADE Assembler provide seamless integration of the Spectre AMS Designer and Xcelium Mixed-Signal simulator. You can use the Virtuoso Hierarchy Editor to specify the cellviews that you want to use in your design.

A configuration (config) is a set of binding rules that defines which cellviews are part of the design for a given purpose (such as netlisting and simulation). Using the Virtuoso Hierarchy Editor, you can view the hierarchy of these cellviews and apply view switching to mix and match abstraction levels depending upon which phase of the design cycle you are performing.

To simulate your design with Spectre AMS Designer, you must specify a top-level conifg cellview for your design hierarchy. The design can contain other config cellviews at lower levels in the hierarchy. For more information on how to create a config view, see Creating a Configuration Cellview.

Related Topics

Working with AMS Simulator

Simulation Snapshots in ADE Assembler

ADE Assembler and ADE Explorer let you save snapshots of a simulation process state using the Process-Based Save and Restart (PBSR) flow. This snapshot can then be used to restart the simulation from the last-saved state. The PBSR flow leads to a comprehensive restart of the process where the simulation must be rerun on the exact same hardware.

Note the following:

- This feature is available only when the selected simulator is AMS and the AMS flex flow is being used.
- ADE Assembler supports only the PBSR flow. The classic save and restart flow is not supported.

Saving Simulation Snapshots

To save simulation snapshots:

1. Select *Enable process based save restart* on the *Assembler Save and Restart Options* group box of the Netlist and Run Options form.

© 2023 Cadence Design Systems, Inc.

2. Select *Enable save*.

   The *Save Options* group box is enabled.

   This check box is grayed out when *Enable restart* check box is selected.
3. Specify a value for the fields on the *Save Options* group box.
4. Click *OK*.
5. Run a simulation.

Note: Snapshots can be saved only for single-point runs.

For more information, see Process-Based Save and Restart (PBSR) flow.

Restarting a Simulation from a Saved Snapshot

To restart a simulation from a saved snapshot:

1. Select *Enable process based save restart* on the *Assembler Save and Restart Options* group box of the Netlist and Run Options form.
2. Select *Enable restart*.

   The *Restart Options* group box is enabled.
3. From the *Snapshot name* drop-down list, select a snapshot.

   Note: Before selecting a snapshot from this drop-down list, ensure that you select *Use saved history for restart* and add a history in the *Selected Histories* section of the Reference Histories form in ADE Assembler.
4. Click *Add*.

   An entry gets added to the *Snapshot sweep information* table for the selected snapshot. You can then select or deselect the check box in the *Enable* column of the table to enable or disable an entry.

Related Topics

Netlist and Run Options Form

Setting Up the AMS Netlister Options

© 2023 Cadence Design Systems, Inc.

Simulation Runtime Improvements

The following enhancements show significant improvements in the simulation runtime.

Performance Enhancements Using Spectre Plugin

To reduce the performance overhead of creating the complete netlist for every simulation point and the associated Spectre startup time, a Spectre plugin is used. ADE creates an initial netlist for the first design point and sends it to Spectre plugin. Spectre reuses this netlist for all the design points and makes only variable changes for which the plugin sends the details to Spectre.

If netlist recreation and Spectre startup time is comparable to the simulation time, then a significant performance improvement can be noticed when the new Spectre plugin is used. However, the performance improvement may diminish when the simulation run time is high.

Note: The Spectre plugin is not supported in LSCS job control mode.

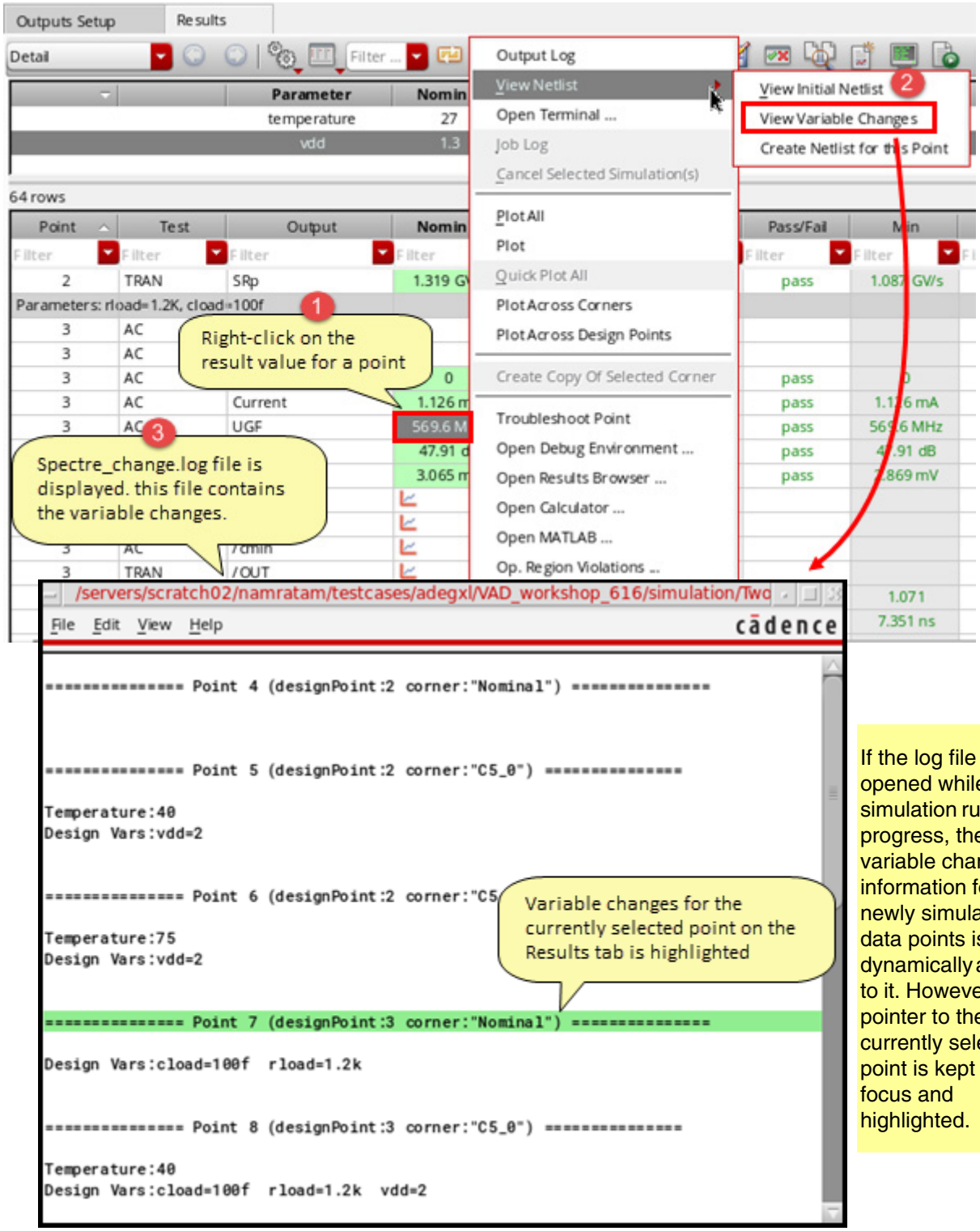
Viewing Netlist

To view or create the netlist for a design point during the simulation run, right-click in a cell in the results table and select one of the options in the *View Netlist* menu:

- *View Initial Netlist*: Opens the initial netlist generated when the first design point is simulated.
- View Variable Changes: Opens the `spectre_change.log` file that displays the variables changed for the current point with respect to the first point. This file also shows a log of variables used at each point.

© 2023 Cadence Design Systems, Inc.

- *Create Netlist for this Point*: Creates netlist for the selected point.

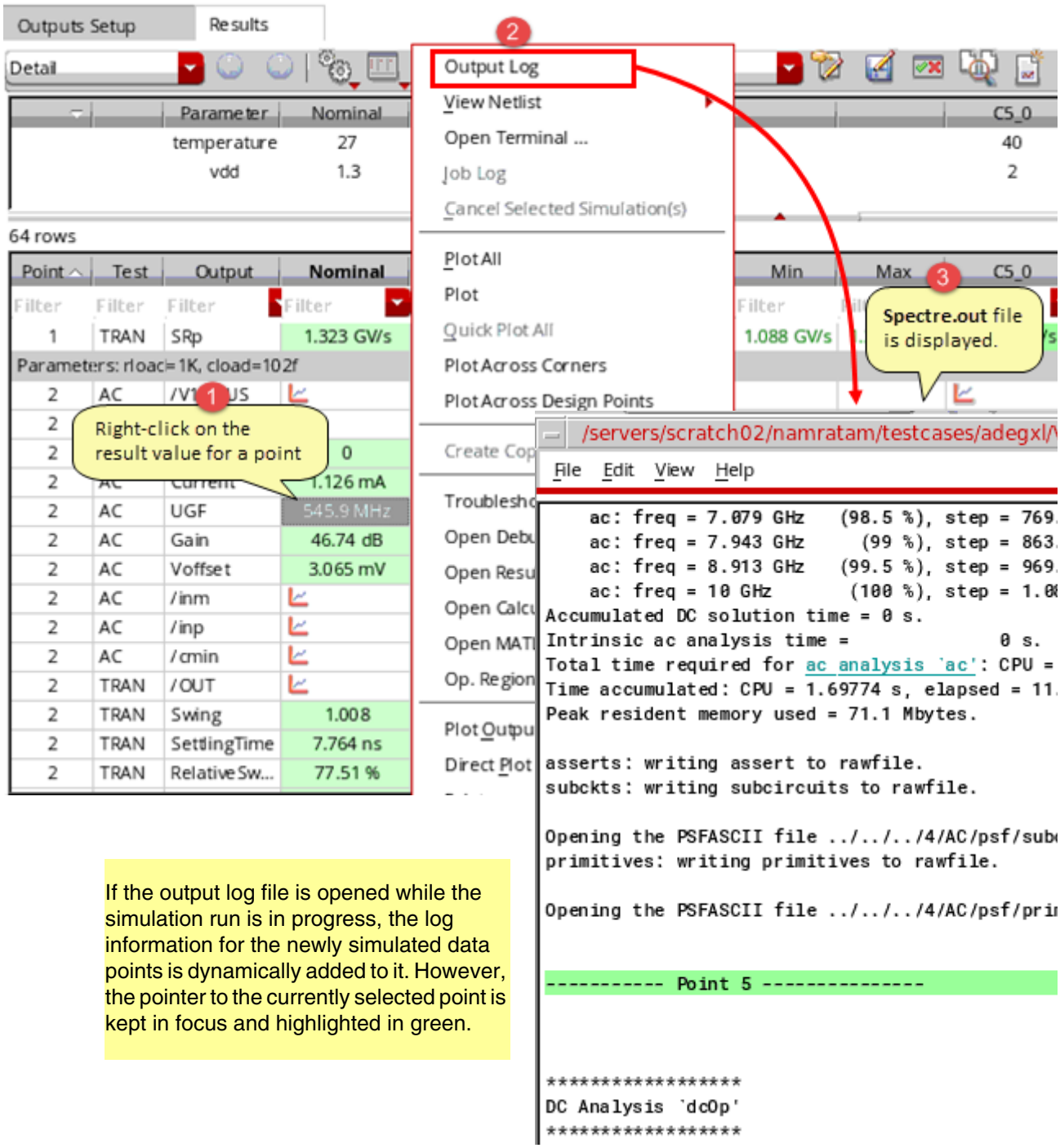


If the log file is opened while the simulation run is in progress, the variable change information for the newly simulated data points is dynamically added to it. However, the pointer to the currently selected point is kept in focus and highlighted.

© 2023 Cadence Design Systems, Inc.

Viewing Output Log

Spectre plugin maintains a single Spectre `Spectre.out` log for all the points. To view the output log for a data point, right-click in a cell on the *Results* tab and choose *Output Log*.



If the output log file is opened while the simulation run is in progress, the log information for the newly simulated data points is dynamically added to it. However, the pointer to the currently selected point is kept in focus and highlighted in green.

© 2023 Cadence Design Systems, Inc.

Limitations

- The Spectre plugin is not supported in the scenarios listed below. ADE Assembler automatically disables Spectre plugin, continues to run simulation and displays an information message in the job log:
  - The enabled corners use model sections, but a default model file is not defined and the Nominal corner is disabled
  - The enabled corners use swept model sections
  - Model file sections use VAR()
  - Multi-technology simulations
  - Checks and Asserts
  - Pcell parameters are used in the design
  - Simulators other than Spectre
  - When the *Optimize Single Point Run* check box on the Job Policy form is selected and there is a single point run
  - Monte Carlo
  - Statistical corners
  - Troubleshoot run
  - Config variables
  - String variables
  - Include analysis
  - Pre-run scripts
  - *XPS MS* performance mode is enabled on the High-Performance Simulation Options form
  - `+xps` command line option is set
  - Spice outputs
  - Electrically Aware Design flow is enabled
  - EM/IR flow is enabled
  - DSPF files used in the parasitic aware design flow
  - DSPF cellviews are used in tests

© 2023 Cadence Design Systems, Inc.

Performance Enhancements by Submitting Points in a Group

ADE Assembler used to send one point at a time to ICRPs. To improve the performance of simulation throughput and the responsiveness of the user interface, ADE Assembler now sends a group of points to each ICRP.

Running a group of points together reduces the overhead of multiple interactions of the user interface with the ICRPs. In addition, when combined with the redistribution feature that automatically reallocates simulations waiting on slow resources to other resources, it helps improve the overall runtime. To enable redistribution, set the minGroupSizeSplitAcrossIdleJobs .cdsenv variable to the count after which redistribution should take place. By default, this variable is set to `0`.

For example, when you set this variable to 20, if an ICRP has 20 or more simulations, they are redistributed to other available ICRPs.

Consider another example where all ICRPs have completed their simulations, but 5 simulations are still pending on one ICRP because it is on a slow resource. In this case:

- Redistribution helps if ADE Assembler takes away those 5 simulations and assigns them to 5 faster computers.
- Redistribution does not help if one or more of the computers receiving these new simulations are equally slow.
- Redistribution helps to a great extent if each simulation takes a few hours, and the slow resource with 5 points has become the bottleneck for the entire run to complete, and, there are other resources that are available.

When the points are redistributed to other available jobs, appropriate information messages are added to the job log files. You can review those messages for debugging purpose.

Note: Though the points are submitted to the ICRP in a group, the status on the *Results* tab shows each point running separately.

Overall improvement in runtime depends on the following factors:

- Number of points being run
- Long simulations (if simulation runtime is large, then other improvements may appear non-significant)
- Overall quality of the farm
- Amount of load on farm computers
- Number of max jobs

© 2023 Cadence Design Systems, Inc.

Related environment variable: `groupRunA`

Also see:

- Specifying a Name for Simulation History
- Checking Run Status on the Progress Bar
- Suspending and Resuming Netlisting or Simulation Runs
- Stopping a Simulation
- Stopping Jobs and Resubmitting Simulations
- Ignoring Design Changes During Run
- Continuing the In-Process Simulations After ADE Assembler GUI Exits
- Disabling and Enabling All Point Sweeps
- Simulation Errors
- Creating and Running an OCEAN Script

© 2023 Cadence Design Systems, Inc.

Specifying a Name for Simulation History

When you run a simulation, depending on the run mode, ADE Assembler assigns a default name to the new history. For example, for the `Single Run, Sweeps, and Corners` run mode, the default history name is `Interactive`, and for the `Monte Carlo Sampling` run mode, the default history name is `MonteCarlo`.

If required, you can specify a custom name to be used for a history. You can do this in either of the following two ways:

- By setting an appropriate environment variable to set the default history name. This method is useful when you want to override the default history name for a specific run mode. For example, for the `Single Run, Sweeps, and Corners` run mode, you want to use `SingleRun` instead of `Interactive` as the history name.

  To change the default history name for a run mode, you can change the value of the environment variable corresponding to the run mode. ADE Assembler provides the following environment variables:

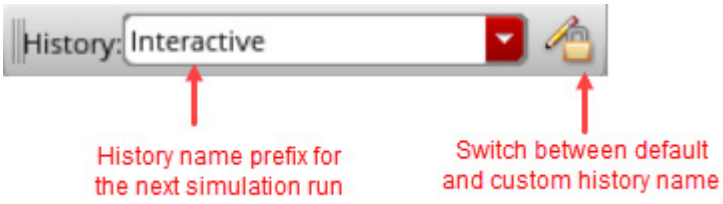
  - singleRunSweepsAndCorners
  - monteCarloSampling
  - WorstCaseCorners
  - globalOptimization
  - localOptimization
  - improveYield
  - highYieldEstimation
  - sensitivityAnalysis
  - manualTuning
  - sizeOverCorners

- By specifying a history name in the *History Name* toolbar. This method is useful when you want to use a history name different than the default or the name specified by the environment variable corresponding to the current run mode.

  By default, the default history name is used. To use a different name, click *Switch between default and custom history name* to toggle the switch. It also enables the

© 2023 Cadence Design Systems, Inc.

History name prefix for the next simulation run field on the toolbar. Specify a custom history name in this field.



The next simulation run uses the custom name specified on this toolbar.

If you do not want to customize history names, you can hide the *History Name* toolbar by setting the showHistoryPrefixToolbar environment variable to `nil`.

Related Topics

axlSetHistoryPrefixInPreRunTrigger

Important Points to Note

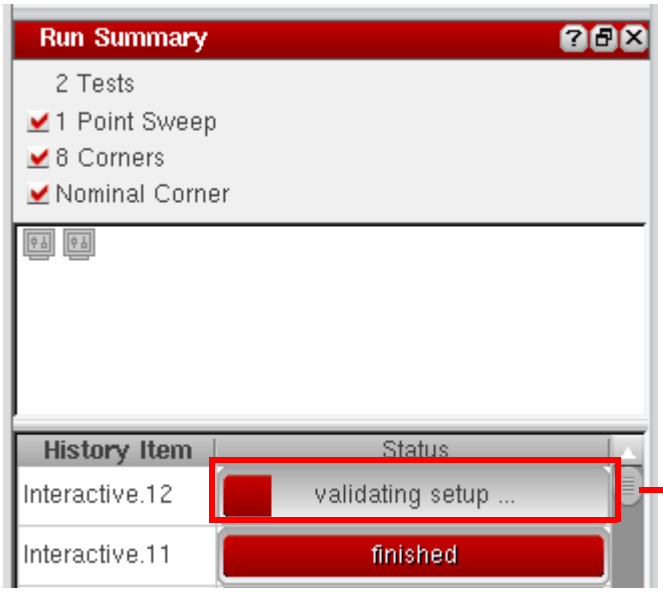
- If the *Overwrite History* option on the Save Options form is selected, the history name for the next run will be the same as that specified in the *Overwrite History* drop-down list. The setting for the adexl.icrpStartup environment variable is ignored.
- By default, all the history names are suffixed with an incremental number to keep them unique. For example, If you have specified the history name as `TestRun`, the histories will be named as `TestRun.0`, `TestRun.1`, `TestRun.2`, and so on. To exclude the unique identifier, you can set the initiallyAddHistoryNameUniquifier environment variable to `nil`.

  Note: The initiallyAddHistoryNameUniquifier environment variable is applied only when the history name being used is unique. If the name has already been used for an existing history, the value of this variable is ignored and a unique number is added as a suffix. This helps in keeping the names of both the histories unique.

© 2023 Cadence Design Systems, Inc.

Checking Run Status on the Progress Bar

While ADE Assembler is preparing to run a simulation or during a run, you can check its status on the progress bar displayed on the Run Summary assistant pane. The progress bar is displayed to the right of the history item name, as shown in the following figure.



It shows one of the following status values to correctly indicate the step being performed:

Table 17-1 Status Values Displayed on the Run Progress bar

Status Value	Indicates that...
validating setup	Verification of the test setup, variables, parameters, and corners specified in the Data View pane is in progress. ADE Assembler is also verifying the run options specified for the different types of runs in the corresponding run options form. If there are any errors or warnings, appropriate messages are displayed in the CIW or error message dialogs.
saving history	Creation of the test setup data is in progress. That is, ADE Assembler is creating the directories in which the test setup data for the simulation run will be saved.
preparing result directories	Creation of the results database (.rdb file) and the results directory is in progress.

© 2023 Cadence Design Systems, Inc.

Table 17-1 Status Values Displayed on the Run Progress bar

Status Value	Indicates that...
loading tests x/n	ADE Assembler is starting a session for each active test in the setup. Here `n` specifies the total number of active tests in the setup, not the total number of tests.
evaluating design points x/n	Creation and evaluation of the xth number of design point out of n design points is in progress. Here, a total of n design points is calculated based on the given variables, parameters, and run options. **Note:** These are not data points, but the design points that are created by swept values.
populating results UI	Preparation of the user interface to display results is in progress. After this step is complete, the *Results* tab for the current run is displayed on the ADE Assembler window. At this time, the run status for all the simulations is shown as *pending*.
running – x/n complete	ADE Assembler is running the `x`th run out of a total of `n` runs. The progress bar is reset in this stage.
finished or *finished with errors*	Successful or unsuccessful completion of all the simulation runs.

© 2023 Cadence Design Systems, Inc.

Suspending and Resuming Netlisting or Simulation Runs

A simulation run might fail when sufficient disk space is not available. In such cases, you can suspend one or all of the running simulations, clean up the disk, and resume the simulations for completion. You do not need to cancel the run.

You can suspend a simulation in the following ways:

- Suspending Simulations Automatically When the Disk Space is Low
- Suspending Simulations Manually
- Suspending Netlist Service Jobs Manually (LSCS Only)

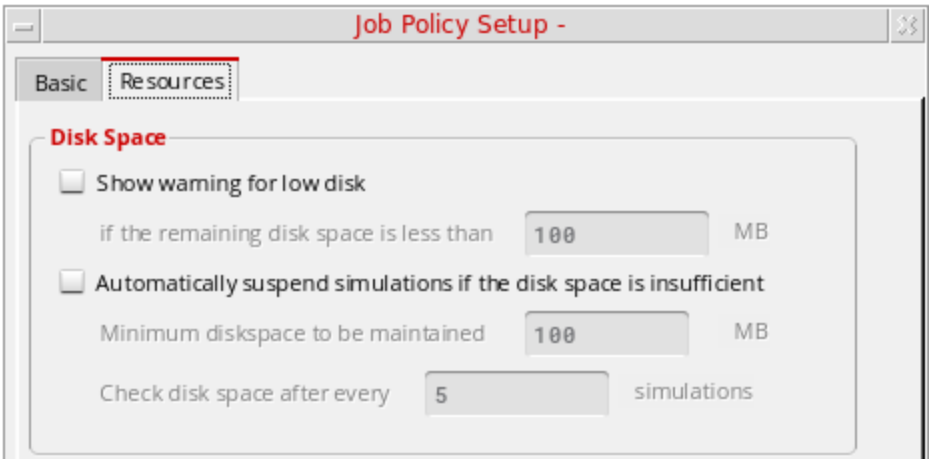
Suspending Simulations Automatically When the Disk Space is Low

You can configure the commands to automatically suspend simulations when the disk space is low, and flag the appropriate information or warning messages.

Note: If the continueICRPRunOnAbruptGUIExit environment variable is set to `t`, this feature is not supported for the ICRP job control mode. However, the feature is supported for LSCS job control mode.

To enable checking of disk space and automatic suspension of simulations depending on the configured settings:

1. Choose *Options – Job Setup* to open the Job Policy Setup form.
2. Open the Resources tab, shown below.



3. Specify the following options in the *Disk Space* group:

© 2023 Cadence Design Systems, Inc.

- *Show warning for low disk*: Select this check box to enable checking of the available disk space when a simulation run is started. ADE Assembler shows a warning in these cases:
  - If the available disk space before running a simulation for the first point is less than the threshold value specified in the *If the remaining disk space is less than _ MB* field, a warning message is flagged to report about the unavailability of the required space and to confirm whether the run is to be continued.
  - For the Single Run, Sweeps, or Corners run mode, after the first design point is simulated, the disk space consumed for that point is used to estimate the total space required to run simulations for all the points. If the **remaining (available – estimated)** disk space falls below the threshold, ADE Assembler shows the following types of messages:

    • If the remaining (available – estimated) disk space is more than the threshold specified in the *If the remaining disk space is less than* field, an information message is flagged to report the same.

    • If the remaining (available – estimated) disk space is less than or equal to the threshold specified in the *If the remaining disk space is less than* field, a warning message is flagged to report the same. It is recommended to manually suspend the simulation run for the current history, free up the disk space, and resume the run.

    Consider an example of a simulation run for three sweep points.
    - The simulation of the first design point used: 15 MB
    - The estimated disk space for the remaining two points: 30 MB
    - The available disk space after the first design point is simulated: 115 MB
    - The threshold value for disk space: 100 MB
    - The estimated remaining disk space = 115 – 30 = 85 MB, which is less than the threshold value

    In this scenario, after the simulation for the first design point is complete, a warning message is displayed to clear disk space.

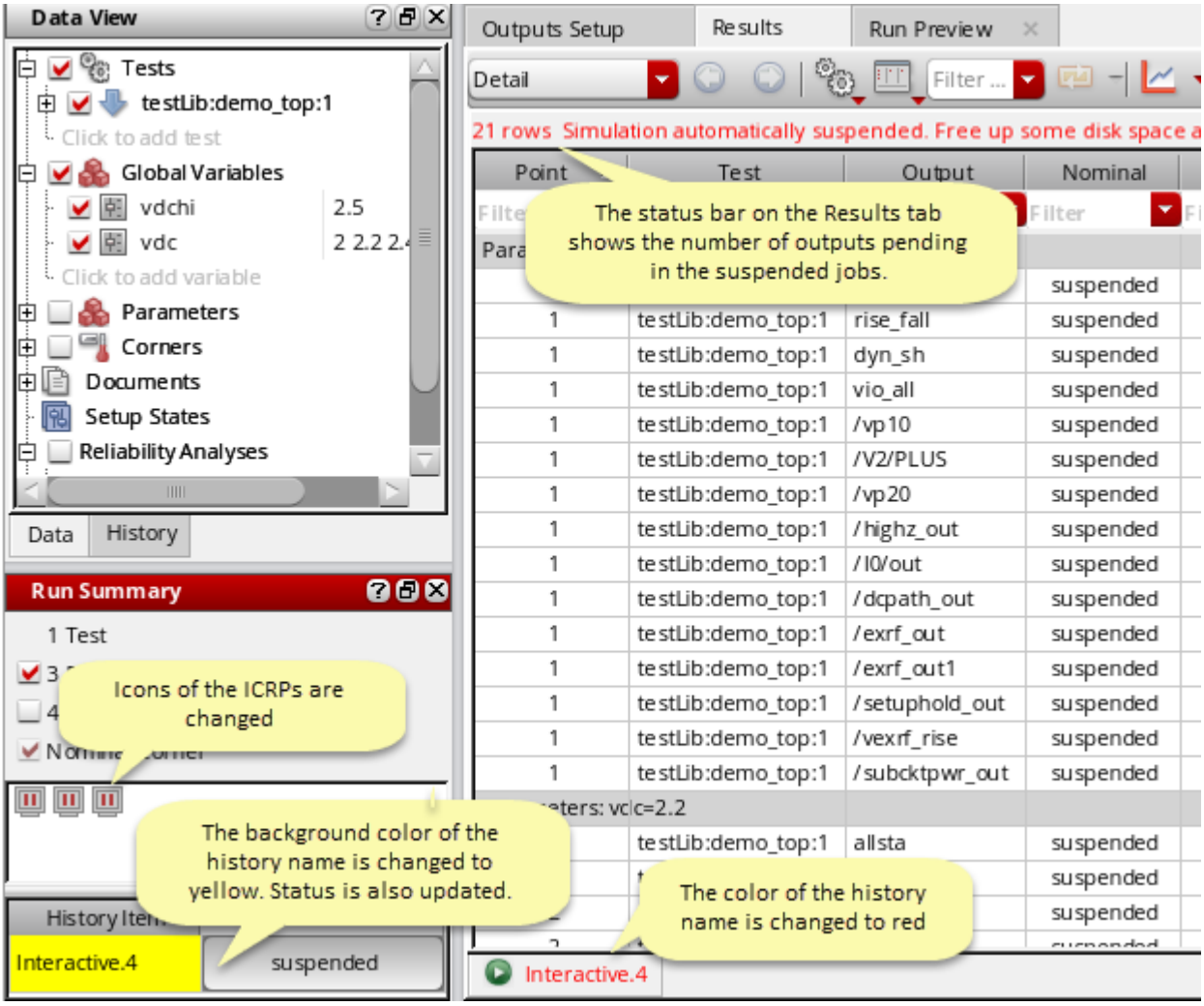
- *Automatically suspend simulations if the disk space is insufficient*: Select this check box to enable automatic suspension of all the simulations in progress whenever the available disk space falls below the minimum threshold specified in the *Minimum disk space to be maintained (in MB)* field.

  When automatic suspension is enabled, by default, ADE Assembler checks the disk space after every five simulations. You can change the frequency in the *Check disk space after every _ simulations* field.

4. Click OK to close the form.

© 2023 Cadence Design Systems, Inc.

ADE Assembler uses the settings specified on this form to constantly monitor the available and estimated disk space and displays appropriate messages. If a simulation is automatically suspended, the status on the Run Summary tab and the Results tab is updated, as shown below.



The icons of the jobs allocated for the suspended simulation run appear as paused ( ). Their tooltips also show the time since when they are in the suspended state.

Suspending Simulations Manually

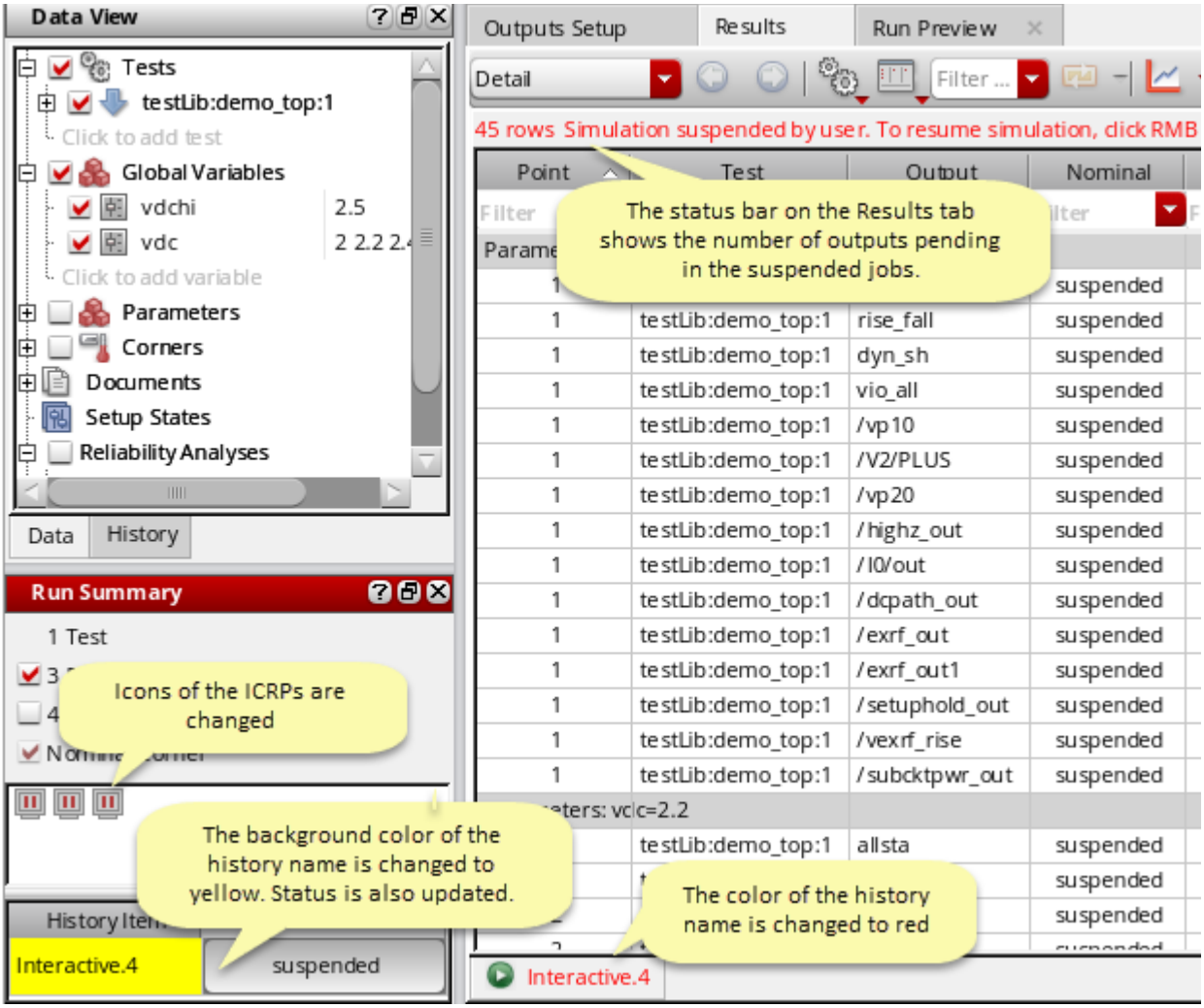
To suspend a simulation manually:

➡ Right-click the progress bar of the simulation run in the Run Summary pane and click *Suspend*.

© 2023 Cadence Design Systems, Inc.

The simulation is suspended and the following changes in the user interface reflect the status.

When you suspend a simulation run, its status is changed to `Suspended`, as shown below.



When you need to resume the simulation, right-click its progress bar in the Run Summary pane and click *Resume*.

To suspend and resume all the simulations, right-click any of the runs and use the *Suspend All* and *Resume All* commands.

© 2023 Cadence Design Systems, Inc.

Important Points to Note

- The settings for automatic suspension of jobs can be set only in the global Job Policy Setup form. These settings are not available in the job policy at the test level.
- You cannot suspend a simulation when the continueICRPRunOnAbruptGUIExit environment variable is set to `t`.
- When you suspend a simulation, the status of its points on the *Results* tab appears as `suspended`.
- The job policy settings to automatically suspend simulations are ignored while running simulations using OCEAN or SKILL scripts. These settings are also ignored for the reliability analysis.
- If you change the status of automatic suspension after a simulation has been suspended, and then resume the simulation, the new settings for automatic suspension take effect for the rest of the run.
- Starting from MMSIM 15.1 ISR15, Spectre supports the `+disk_check=<`*`threshold value`*`>` command-line option to automatically suspend simulations if the disk space is below the given threshold limit. To use this feature of Spectre, specify this option in the *User Command-line Options* field Environment Options form for each test in ADE Assembler. When this option is given and Spectre suspends a simulation, the corresponding ICRP is suspended. If the resources running other the jobs submitted by other ICRPs have enough resources, they continue to run.

  When the Spectre sessions for all the ICRP jobs in ADE Assembler are suspended, the status of the history tab is changed to suspended and its background color is changed to yellow.

Suspending or Resuming Simulations From a DRMS

When the simulations are running in a DRMS, if you need to preempt any simulation to manage the resources, send the SIGTSTP signal to the ICRP process. The signal is then sent to the Spectre process spawned by that ICRP to suspend the simulation and release the license. To retrieve the license and resume simulation, send the SIGCONT signal to that ICRP process.

Note: When suspended, Spectre will release the license only if it is called with the `+lsuspend` option. If `+lsuspend` is not used, Spectre will suspend the simulation, but the license will not be released.

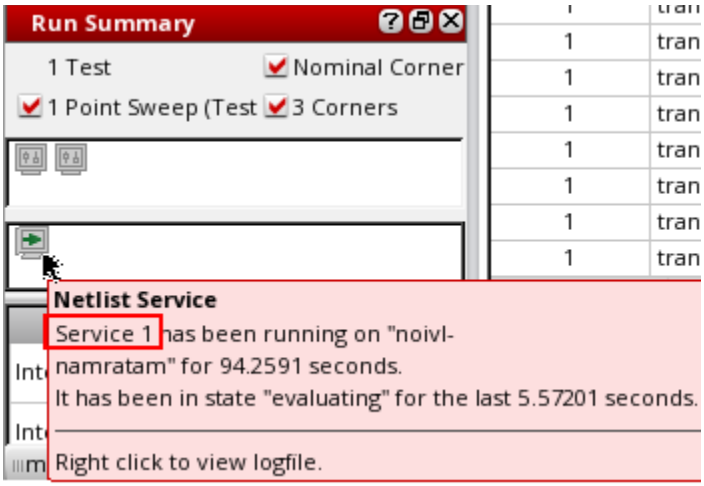
© 2023 Cadence Design Systems, Inc.

Suspending Netlist Service Jobs Manually (LSCS Only)

In the LSCS job control mode, dedicated jobs are assigned to create netlists for data points. These netlist service jobs can run on a local computer or on other resources on the network. At times, you might need to suspend a netlisting job to facilitate an efficient management of resources.

To suspend a netlist service job that is running on the local computer:

1. Move the mouse pointer over a netlist service job icon in the Run Summary assistant and note the service number displayed in the tooltip.



2. Open the netlist service log that is located at the following location:

   current-working-dir`/logs_`*user-name*`/logs0/Service`*Service-number*`.log`

3. Search for `Process Id` in the service log file and note the ID. For example,

   ```
   \# Process Id:          427101
   ```

4. Run the following command at the UNIX command prompt:

   ```
   kill -s TSTP process-ID
   ```

Similarly, to resume a netlist service job, run the following command:

```
kill -s CONT process-ID
```

To suspend a netlist service job that you started by using the LSF distribution method:

1. Use the following command at the UNIX command prompt:

   ```
   bkill -s SIGTSTP LSF-job-ID
   ```

Similarly, to resume a netlist service job, run the following command:

```
bkill -s SIGCONT LSF-job-ID
```


© 2023 Cadence Design Systems, Inc.

When you suspend or resume a netlist service job, the signal handler for the LSCS job control mode logs signals received from LSF or the local resource. These messages are printed in the netlisting log.

```
\o compose simulator input file...
\o       ...successful.
\o [2021-06-11 15:53:03.780 +0530] [process] - Current memory used: 334.8281 MBytes, peak memory used: 493.7227 MBytes [At
\o [2021-06-11 15:53:03.784 +0530] [info] - Finished netlisting [Attributes] {"point": 1, "testname": "tran"}
\o [2021-06-11 15:53:03.797 +0530] [process] - Received suspend signal, suspend netlister service. [Attributes] {"hostname"
\o [2021-06-11 15:53:23.890 +0530] [process] - Received resuming signal, resume netlister service. [Attributes] {"hostname"
\o [2021-06-11 15:53:23.894 +0530] [event] - NewPoint [Attributes] {"point": 2, "testname": "tran", "event": "NewPoint", "
\o [2021-06-11 15:53:23.894 +0530] [info] - Started netlisting [Attributes] {"point": 2, "testname": "tran"}
\o Resetting statistical vars
```

You can also view these messages in the Log Viewer window for the history.

History: Interactive.10

	Time	Level	Tool	Test	Point	Message(double click item to open
6	06-11 15:52:59.029	process	Netlist Service	tran		Received resuming signal, resume netlister servic
47	06-11 15:53:23.890	process	Netlist Service			Received resuming signal, resume netlister servic
153	06-11 15:55:21.810	info	ADE GUI			GUI Resumes Netlister Services '8 '
154	06-11 15:55:28.916	info	ADE GUI			GUI Resumes Netlister Services '8 '
163	06-11 15:59:00.158	process	Netlist Service			Received resuming signal, resume netlister servic

The signal handler also sends a message to the ADE Explorer or ADE Assembler environment that updates the icons corresponding to the netlist service to indicate the correct status.

© 2023 Cadence Design Systems, Inc.

Stopping a Simulation

To stop a simulation run, do one of the following:

- On the Run toolbar, click *Stop Simulation* .
- On the Results tab of the Outputs pane, right-click the tab name for the checkpoint of the simulation that is running and choose *Stop Run*.
- On the Run Summary pane, right-click a history item and choose *Stop All* to stop all the jobs you started during the current session regardless of their current state (started, getting configured, running).
- Switch to the Status view on the Results tab and click the *Stop* button.

ADE Assembler stops all the simulations that are running. Results for the points that are already successfully complete are saved in the results database for the history checkpoint.

Important Points to Note

- Instead of stopping the entire simulation run, you can stop the simulation for only selected tests or points. For more details, refer to Canceling Simulations for Selected Tests or Corners.
- If required, you can rerun the simulation for incomplete points later. For more details, refer to Simulating Only Unfinished or Erroneous Points.

Important

If you are using the Spectre circuit simulator, you can release your license by clicking the button.

© 2023 Cadence Design Systems, Inc.

Canceling Simulations for Selected Tests or Corners

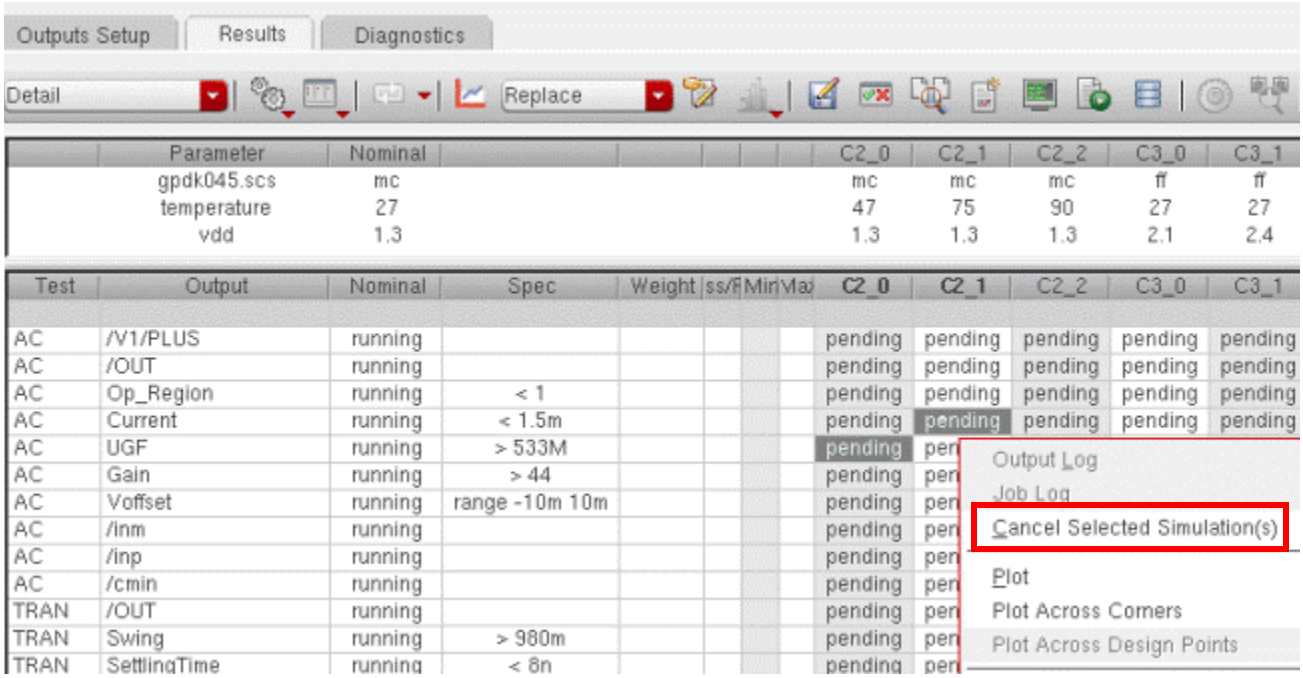
In certain cases, you might need to cancel the simulation for a specific test or corner and complete the simulation for the remaining. This is helpful in the following scenarios:

- When the results of a corner indicate that there is scope of improvement in the design and you do not want to run similar corners.
- When the results of a test indicate that the setup needs to be changed and you want to cancel all of the pending runs for that test, but complete the simulations for the other tests.
- When you are running resource-intensive simulations and you want to allow only the jobs that are already running to continue, and to cancel all the pending jobs so that the limited MMSIM licenses can be released for other high-priority simulations.

Canceling Simulations for Selected Corners

To cancel a simulation for one or more corners, do the following:

1. In the *Detail* view of the *Results* tab, select one or more result points in the corner columns for which you do not want to run simulation.

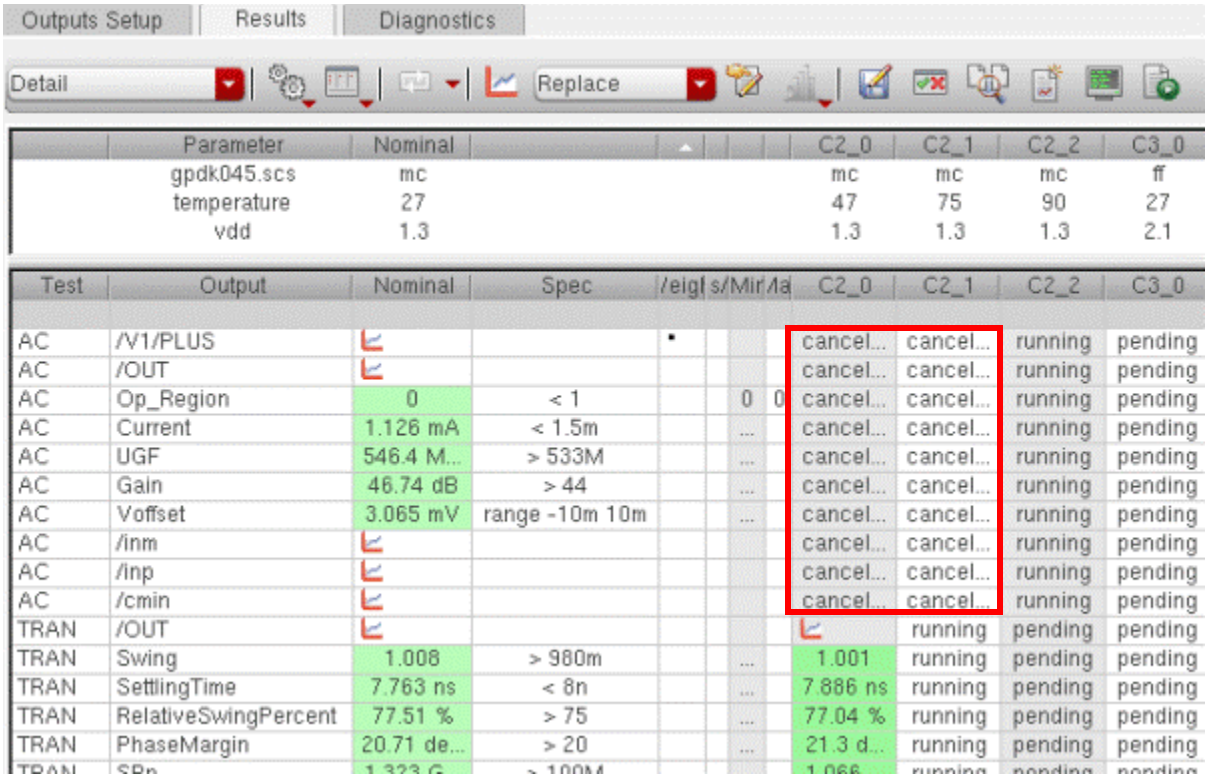


Note: You can cancel the simulation in `pending` as well as `running` state.

© 2023 Cadence Design Systems, Inc.

2. Right-click one of the selected points and choose *Cancel Selected Simulations*.

   ADE Assembler cancels the simulation for the points corresponding to the selected cells and completes the remaining simulations. The status of the simulations that you stopped appears as `canceled`, as shown below.



	Parameter	Nominal				C2_0	C2_1	C2_2	C3_0
	gpdk045.scs	mc				mc	mc	mc	ff
	temperature	27				47	75	90	27
	vdd	1.3				1.3	1.3	1.3	2.1

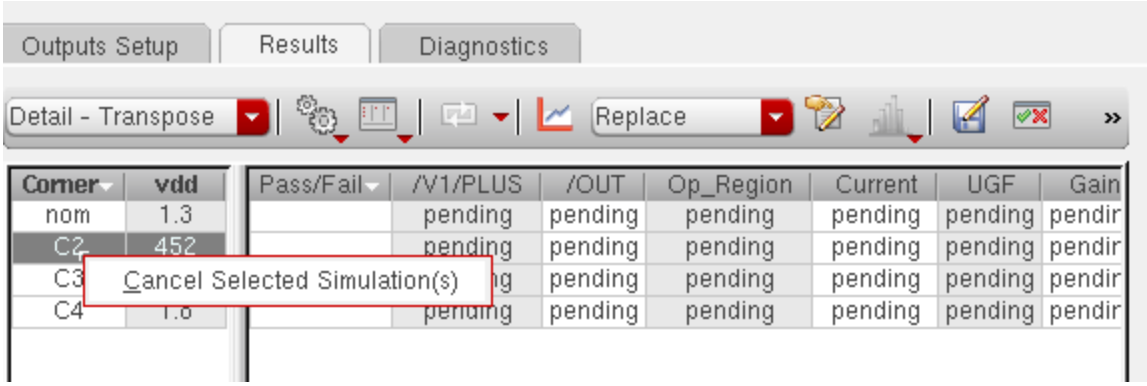
Test	Output	Nominal	Spec	/eigl	s/Mir	Ma	C2_0	C2_1	C2_2	C3_0
AC	/V1/PLUS			•			cancel...	cancel...	running	pending
AC	/OUT						cancel...	cancel...	running	pending
AC	Op_Region	0	< 1		0	0	cancel...	cancel...	running	pending
AC	Current	1.126 mA	< 1.5m		...		cancel...	cancel...	running	pending
AC	UGF	546.4 M...	> 533M		...		cancel...	cancel...	running	pending
AC	Gain	46.74 dB	> 44		...		cancel...	cancel...	running	pending
AC	Voffset	3.065 mV	range -10m 10m		...		cancel...	cancel...	running	pending
AC	/inm						cancel...	cancel...	running	pending
AC	/inp						cancel...	cancel...	running	pending
AC	/cmin						cancel...	cancel...	running	pending
TRAN	/OUT							running	pending	pending
TRAN	Swing	1.008	> 980m		...		1.001	running	pending	pending
TRAN	SettlingTime	7.763 ns	< 8n		...		7.886 ns	running	pending	pending
TRAN	RelativeSwingPercent	77.51 %	> 75		...		77.04 %	running	pending	pending
TRAN	PhaseMargin	20.71 de...	> 20		...		21.3 d...	running	pending	pending

Important Points to Note

- When you cancel the simulations for a corner, ADE Assembler cancels the simulations only for the tests corresponding to the selected points. For other tests, simulations for those corner and point combinations are completed.
- To cancel simulations for all the corners in a test, hold down the `Shift` key and select all the corners in a row for that test, right-click and choose *Cancel Selected Simulations.* If there are multiple design points, all the corner simulations for only that design point and test combination will be canceled.
- To cancel simulations for a particular corner in all the tests, hold down the `Shift` key and select all the rows in the corner column, right-click and choose *Cancel Selected Simulations*.

© 2023 Cadence Design Systems, Inc.

- Canceled simulations are considered to be specification failures, so the *Pass/Fail* column shows the `Fail` status for any output specification that contains canceled points.
- When you plot an output across corners, the canceled corners for which results were not generated are not shown in the plot.
- You can cancel simulations for the selected corners in the Detail - Transpose view as well. For this, do the following:
  - To cancel simulations for one or more corners across all the tests, in the left pane of the view, click to select the rows for the corners, right-click and choose *Cancel Selected Simulations*, as shown in the figure below.



Simulations for the corresponding design point and corner combinations are canceled across all the tests.

  - To cancel simulations for specific design point, corner, and test combinations, select the corresponding cells in the right pane of this view, right-click and choose *Cancel Selected Simulations*.

© 2023 Cadence Design Systems, Inc.

Canceling Simulations for Selected Tests

To cancel simulation for one or more tests, do one of the following:

1. In the *Detail* view of the *Results* tab, select one or more tests in the *Test* column.
2. Right-click one of the selected test names and choose *Cancel Selected Tests*.

   Note: You can also select a combination of test and corners to cancel their simulations. In this case, when you right-click any one of the selected tests or corners, the name of the command changes to *Cancel Selected Tests and Simulations*.

ADE Assembler cancels all the pending or running simulations for the selected tests and completes the remaining tests.

You can cancel simulations for the selected tests in the *Summary* view as well. For this, select one or more rows for tests, right-click and choose *Cancel Selected Tests*.

In LSCS job control mode, netlisting is done in the beginning at the test level. If netlist creation at the test level fails, the status for all points is shown as *netl err* irrespective of their submission status. Therefore, it is possible that if you cancel simulation for any test or corner, its status is shown as *netl err*.

Known Limitations

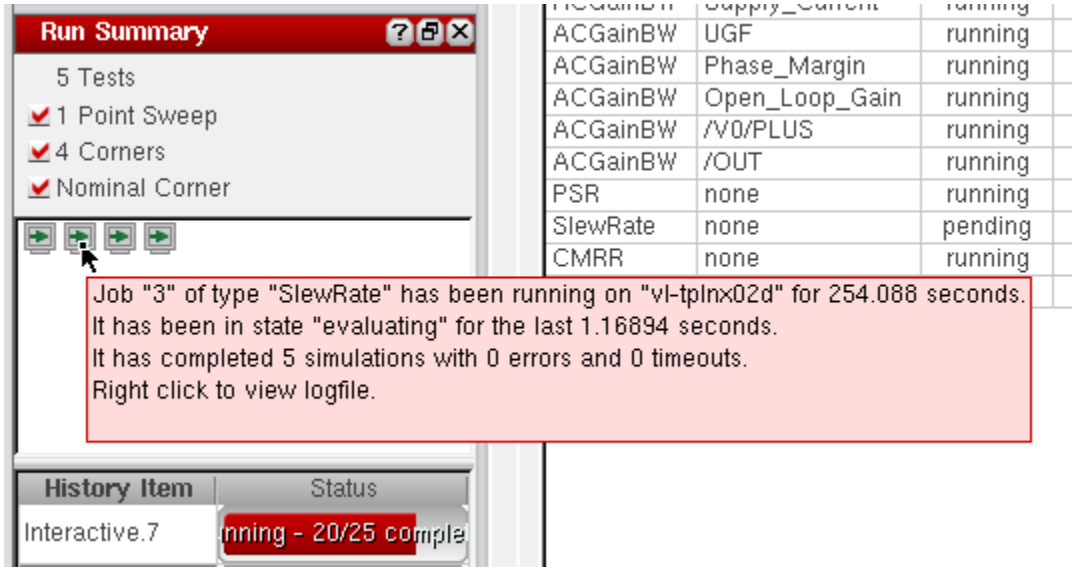
You cannot cancel selected simulations while running Monte Carlo Sampling simulations with Yield Verification - Basic AutoStop enabled.

Stopping Jobs and Resubmitting Simulations

During a simulation, the *Run Summary* assistant shows the status of the ICRP jobs that are running simulations. You can check the status of each individual ICRP job by placing the pointer over it. A tooltip is displayed giving the details of the current simulation in progress and

© 2023 Cadence Design Systems, Inc.

the total number of simulations completed by the job. An example is shown in the following figure.



If you observe that a particular job is taking an unusually long time to complete, you can stop that job and resubmit the simulation.

➥ To stop a job and resubmit the simulation, right-click the job and choose *Stop and Resubmit*.

When you use this command, ADE Assembler does the following:

- The icon for the stopped job changes to and then disappears. A new job is created to replace the stopped job.
- The simulation that was running on the stopped job is placed in the beginning of the queue of pending simulations. The simulation for this point is then submitted to the next available job.
- If no more simulations are pending, ADE Assembler might not create a new job.

Stopping a job is helpful in the following scenarios:

- When a job is submitted to a computer that is known to be slow or with limited memory: If the tooltip of a job shows the name of the slow computer and you observe that the currently running simulation is taking an unusually long time to complete, you can stop the job and restart it. The queuing software might distribute the resubmitted job to another computer that has better resources.

© 2023 Cadence Design Systems, Inc.

- When a particular job has simulated a significantly smaller than the average number of simulations: If the tooltip of a job shows that the number of points simulated by it are less than the points simulated by other jobs, you can stop the slow job and resubmit it.
- When nearing the end of a run, only one or two jobs remain pending and are taking more than the expected time to complete the simulation for the current design points: In this case, you can try to stop and resubmit the job. Resubmission of job can allocate it to another better computer.

Note: If no simulation is running on a job, the context-sensitive menu shows the *Stop* command in place of the *Stop and Resubmit* command.

© 2023 Cadence Design Systems, Inc.

Troubleshooting Jobs that Stay Pending until Timeout

Some jobs may stay pending until timeout because a Message Passing System (MPS) communication cannot be established between the ICRP and GUI processes.

Any one of the following reasons can cause this gap in communication:

- There may be a firewall blocking the TCP/IP communication between computers.
- ADE may have started a large number of jobs and does not have the sufficient time frame to respond back to all jobs. This causes the communication timeout to fail.

  Note: Communication timeout is a subset of the start timeout, and refers to the time taken to establish a TCP/IP communication.

  In this case, you can modify the communication timeout value in the *Communication Timeout* field on the Job Policy Setup form or in the communicationTimeout property of a job policy set using SKILL.
- The cdsNameServer versions may be incompatible. If the computer initially ran a 5141 (or a version prior to 6.1.3) cdsNameServer then any attempt to establish connection using a cdsNameServer version later than 6.1.3 would fail.

  Note the following:

  - The cdsNameServer program is used to establish communications between Cadence programs. It relies upon the TCP port number, 7325. In addition, whenever needed, the cdsNameServer program starts automatically and exits itself after being idle. To know more, see *Cadence Application Infrastructure User Guide*.
  - `axlBeanstalkd` is a daemon process that provides the messaging service required by different processes to communicate with each other when Virtuoso ADE is running in LSCS job distribution mode. It relies upon the TCP port number 11300.

Important points to note about MPS:

- When virtuoso is running on the GUI_Host, a cdsNameServer process should be running on the GUI_Host listening to port 7325. This can be ensured by:
- GUI_Host% mpsinfo -c
- This listener port must also be accessible from the Remote_host. This can be confirmed by:
- Remote_Host% mpsinfo GUI_Host -c

© 2023 Cadence Design Systems, Inc.

- Access to port 7325 from remote hosts is necessary, but insufficient for the ADE flow. There are other ports which are allocated from the ephemeral range on the GUI_Host that must be accessible from remote hosts.
- Ephemeral ports are generally determined by the OS. For example, ephemeral ports have the range from 32768 to 61000 for Linux.

These ports (7325 and ephemeral ports) need to be opened up for both the GUI host and the remote host, in case these are blocked by a firewall.

© 2023 Cadence Design Systems, Inc.

Ignoring Design Changes During Run

Important

This feature is not supported when the ADE Assembler setup includes device parameterization.

If you modify the schematic while a simulation run is in progress, the design changes may get considered for the netlist creation and simulation of the pending design points in the current run. This is because a new netlist is created before simulating every design point and any design change, such as, a change in pin name, instance name, instance parameter property, or config binding or any addition or deletion of a design component, may get reflected in the netlist.

However, if you want to ignore any design changes in the simulation run that is already running and to consider them only for the subsequent runs, set the `ignoreDesignChangesDuringRun` flag to `t` as shown below:

```
envSetVal("adexl.simulation" "ignoreDesignChangesDuringRun" 'boolean
t)
```

When this flag is set and you run simulation, a netlist is created upfront for each test in ADE Assembler and saved as a master netlist in the top psf directory for the respective test in the simulation directory. A progress bar is also displayed for each test in the ADE Assembler window to indicate that the netlist is being generated. The saved master netlist for each test is then reused for all the point simulations for the test.

In addition, the user setting for the singleNetlistForAllPoints and includeStatementForNetlistInSimInputFile environment variables are ignored when the ignoreDesignChangesDuringRun variable is set. That is, the netlist file in the netlist directory for each point is symbolically linked to the master netlist residing in the top level PSF netlist directory that is saved under the top level psf directory for the test. The following figure shows the netlist directory for point `1` of the `Interactive.2` simulation run.

© 2023 Cadence Design Systems, Inc.

The netlist for point `1` is a symbolic link to the netlist in the `netlist` directory under the psf directory for the test.

The `input.scs` file for Spectre and `input.ckt` file for Ultrasim simulator contain the `include "netlist"` statement, which reflects that the netlist being used is the same netlist for all the points. The following figure shows the `input.scs` file for point `1`.

This helps in minimizing the size of the simulation directory.

© 2023 Cadence Design Systems, Inc.

Important

The `ignoreDesignChangesDuringRun` environment variable is always set to `t` when the LSCS job control mode is used. To create a separate netlist for each point in this mode, set the maestro.simulation singleNetlistForAllPoints environment variable to `nil`.

Limitation

When the ADE Assembler setup includes device parameterization, an incremental netlist needs to be generated for each simulation point during the simulation run. This is required to execute any CDF callbacks during the run. Therefore, any changes that are made to the schematic design during this simulation run cannot be ignored.

In this case, the tool displays a warning message (ASSEMBLER - 1749) requiring confirmation that you want to continue. Now, if you continue, any design changes that happen while the simulation is running are considered in the current run.

To suppress this warning message, you can add the following code to the `.cdsinit` file or run it directly in CIW:

```
axlSuppressPersistedQuestionDialog(1749)
; to suppress the message when the ICRP mode is set
;
axlSuppressPersistedQuestionDialog(8022)
; to suppress the message when the LSCS mode is set
```

Continuing the In-Process Simulations After ADE Assembler GUI Exits

By default, in case of an abrupt exit of ADE Assembler or Virtuoso GUI, any in-process ADE Assembler simulation stops immediately. This results in loss of results of the incomplete simulations and there is no way to recover them.

However, you can choose to keep the in-process simulations running in the background even if the ADE Assembler or Virtuoso GUI is closed. After the simulations are complete, their results are saved in the results database. Next time, you open the ADE Assembler GUI and if the in-process simulations are complete, you can load the history to view results. In this case, you do not need to rerun the simulations that were running when the ADE Assembler GUI was closed.

See the following topics for more information:

- Enabling Continuation of the In-Process Simulations on page 988

© 2023 Cadence Design Systems, Inc.

- Supported Scenarios on page 988
- Limitations on page 994

Enabling Continuation of the In-Process Simulations

To continue and complete the in-process simulations even after the ADE Assembler GUI is closed, set the continueICRPRunOnAbruptGUIExit environment variable to t. When this variable is set to `t`, the IC Remote Processes (ICRPs) that were running the simulations are kept active to complete the simulations. By default, the ICRPs are closed immediately after the ADE Assembler GUI is exited.

Important

After the ADE Assembler GUI is closed abruptly, the existing ICRPs will use the ADE Assembler license to complete the in-process simulations. If more than one ICRP was running, all the processes will share a single license. A new ADE Assembler session will require another license token.

Supported Scenarios

This feature is supported for the following scenarios:

- When the ICRPs are Running and the ADE Assembler GUI Exits Abruptly
- When the ICRPs are Running and Virtuoso Exits Abruptly
- When the Simulations are Running and Virtuoso is Closed

The following sections describe how the running simulations are completed in the scenarios given above.

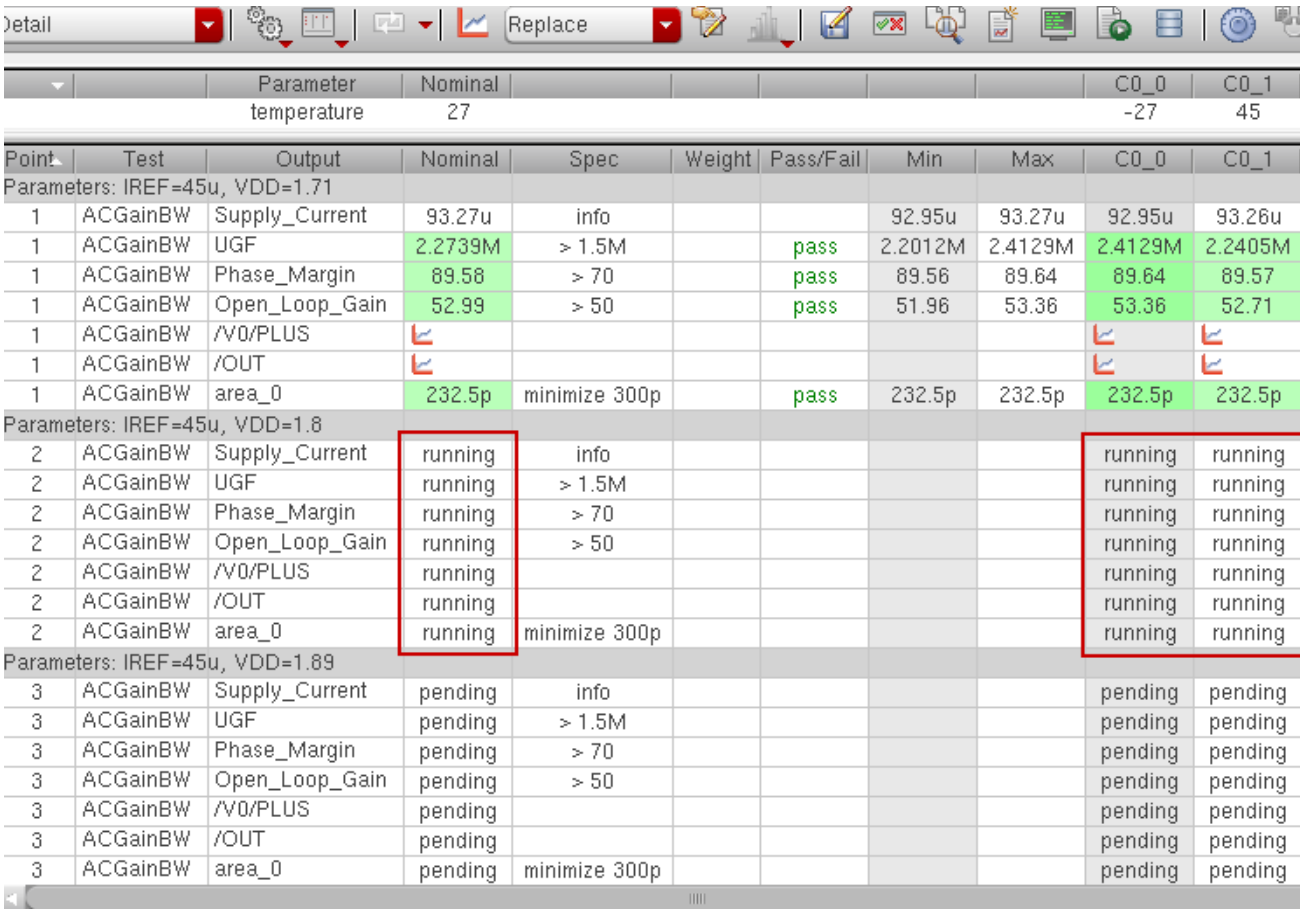
Note: When the `continueICRPRunOnAbruptGUIExit` environment variable is set to t, ADE Assembler creates the runICRPWrapper# file in the current working directory. This file is used internally to manage the child processes when the parent ADE Assembler process is killed.

When ICRPs are Running and Virtuoso or ADE Assembler Environment Exits Abruptly

In the first two scenarios, when the Virtuoso or ADE Assembler GUI exits abruptly, the ICRPs complete the simulations that are currently running, evaluate the output expressions, save results in the results database and then exit.

© 2023 Cadence Design Systems, Inc.

Consider the following scenario in which the ADE Assembler GUI is closed abruptly when some of the simulations are in the `running` state, as shown below.



		Parameter	Nominal						C0_0	C0_1
		temperature	27						-27	45
Point	Test	Output	Nominal	Spec	Weight	Pass/Fail	Min	Max	C0_0	C0_1
Parameters: IREF=45u, VDD=1.71										
1	ACGainBW	Supply_Current	93.27u	info			92.95u	93.27u	92.95u	93.26u
1	ACGainBW	UGF	2.2739M	> 1.5M		pass	2.2012M	2.4129M	2.4129M	2.2405M
1	ACGainBW	Phase_Margin	89.58	> 70		pass	89.56	89.64	89.64	89.57
1	ACGainBW	Open_Loop_Gain	52.99	> 50		pass	51.96	53.36	53.36	52.71
1	ACGainBW	/V0/PLUS								
1	ACGainBW	/OUT								
1	ACGainBW	area_0	232.5p	minimize 300p		pass	232.5p	232.5p	232.5p	232.5p
Parameters: IREF=45u, VDD=1.8										
2	ACGainBW	Supply_Current	running	info					running	running
2	ACGainBW	UGF	running	> 1.5M					running	running
2	ACGainBW	Phase_Margin	running	> 70					running	running
2	ACGainBW	Open_Loop_Gain	running	> 50					running	running
2	ACGainBW	/V0/PLUS	running						running	running
2	ACGainBW	/OUT	running						running	running
2	ACGainBW	area_0	running	minimize 300p					running	running
Parameters: IREF=45u, VDD=1.89										
3	ACGainBW	Supply_Current	pending	info					pending	pending
3	ACGainBW	UGF	pending	> 1.5M					pending	pending
3	ACGainBW	Phase_Margin	pending	> 70					pending	pending
3	ACGainBW	Open_Loop_Gain	pending	> 50					pending	pending
3	ACGainBW	/V0/PLUS	pending						pending	pending
3	ACGainBW	/OUT	pending						pending	pending
3	ACGainBW	area_0	pending	minimize 300p					pending	pending

If the `continueICRPRunOnAbruptGUIExit` variable is set to `t`, all ICRPs that are currently running these simulations will continue and complete the simulations. Next time when you open the history of this run and view results, the results will appear as shown below.

© 2023 Cadence Design Systems, Inc.

	Parameter	Nominal						C0_0	C0_1	C0_2
	temperature	27						-27	45	70

Test	Output	Nominal	Spec	Weight	Pass/Fail	Min	Max	C0_0	C0_1	C0_2
eters: IREF=45u, VDD=1.71										
ACGainBW	Supply_Current	93.27u	info			92.95u	93.27u	92.95u	93.26u	93.18u
ACGainBW	UGF	2.2739M	> 1.5M		pass	2.2012M	2.4129M	2.4129M	2.2405M	2.2012M
ACGainBW	Phase_Margin	89.58	> 70		pass	89.56	89.64	89.64	89.57	89.56
ACGainBW	Open_Loop_Gain	52.99	> 50		pass	51.96	53.36	53.36	52.71	51.96
ACGainBW	/V0/PLUS									
ACGainBW	/OUT									
ACGainBW	area_0	232.5p	minimize 300p		pass	232.5p	232.5p	232.5p	232.5p	232.5p
eters: IREF=45u, VDD=1.8										
ACGainBW	Supply_Current	94.78u	info			94.38u	94.86u	94.38u	94.84u	94.86u
ACGainBW	UGF	2.2822M	> 1.5M		pass	2.2096M	2.4165M	2.4165M	2.2499M	2.2096M
ACGainBW	Phase_Margin	89.57	> 70		pass	89.56	89.62	89.62	89.56	89.59
ACGainBW	Open_Loop_Gain	53.68	> 50		pass	50.56	54.16	54.16	52.86	50.56
ACGainBW	/V0/PLUS									
ACGainBW	/OUT									
ACGainBW	area_0	232.5p	minimize 300p		pass	232.5p	232.5p	232.5p	232.5p	232.5p
eters: IREF=45u, VDD=1.89										
ACGainBW	Supply_Current	pending	info					pending	pending	pending
ACGainBW	UGF	pending	> 1.5M					pending	pending	pending
ACGainBW	Phase_Margin	pending	> 70					pending	pending	pending
ACGainBW	Open_Loop_Gain	pending	> 50					pending	pending	pending
ACGainBW	/V0/PLUS	pending						pending	pending	pending
ACGainBW	/OUT	pending						pending	pending	pending
ACGainBW	area_0	pending	minimize 300p					pending	pending	pending

The results of the simulations were completed and saved in the results database. You need to run only the pending simulations, whose status appears as `pending`. For this, on the *History* tab of the *Data View* assistant pane, right-click the history item that was running when the GUI exited and choose *Rerun Unfinished/Error Points*.

If the `continueICRPRunOnAbruptGUIExit` variable is set to `nil`, all the ICRPs are immediately exited without completing the simulations that were already running. In such a case, if you open the history results, the status of simulations is shown as `running`, but in fact, they are not running.

© 2023 Cadence Design Systems, Inc.

	Parameter	Nominal						C0_0	C0_1	C0_2
	temperature	27						-27	45	70

Test	Output	Nominal	Spec	Weight	Pass/Fail	Min	Max	C0_0	C0_1	C0_2
IREF=45u, VDD=1.71										
GainBW	Supply_Current	93.27u	info			92.95u	93.27u	92.95u	93.26u	93.18u
GainBW	UGF	2.2739M	> 1.5M		pass	2.2012M	2.4129M	2.4129M	2.2405M	2.2012M
GainBW	Phase_Margin	89.58	> 70		pass	89.56	89.64	89.64	89.57	89.56
GainBW	Open_Loop_Gain	52.99	> 50		pass	51.96	53.36	53.36	52.71	51.96
GainBW	/V0/PLUS									
GainBW	/OUT									
GainBW	area_0	232.5p	minimize 300p		pass	232.5p	232.5p	232.5p	232.5p	232.5p
IREF=45u, VDD=1.8										
GainBW	Supply_Current	94.78u	info			94.38u	94.78u	94.38u	running	running
GainBW	UGF	2.2822M	> 1.5M			2.2822M	2.4165M	2.4165M	running	running
GainBW	Phase_Margin	89.57	> 70			89.57	89.62	89.62	running	running
GainBW	Open_Loop_Gain	53.68	> 50			53.68	54.16	54.16	running	running
GainBW	/V0/PLUS								running	running
GainBW	/OUT								running	running
GainBW	area_0	232.5p	minimize 300p			232.5p	232.5p	232.5p	running	running
IREF=45u, VDD=1.89										
GainBW	Supply_Current	running	info					running	pending	pending
GainBW	UGF	running	> 1.5M					running	pending	pending
GainBW	Phase_Margin	running	> 70					running	pending	pending
GainBW	Open_Loop_Gain	running	> 50					running	pending	pending
GainBW	/V0/PLUS	running						running	pending	pending
GainBW	/OUT	running						running	pending	pending
GainBW	area_0	running	minimize 300p					running	pending	pending

If you rerun the unfinished points, all the simulations that were in the `running` or `pending` state are run again.

Important Points to Note

- If you reopen Virtuoso and load the history results while the simulations are still running, you might not see results of the in-process simulations that have the status set as `running`. In such cases, it is recommended to view the results again after an estimated time of simulation run.

  Note: The ADE Assembler GUI of the reopened Virtuoso session does not provide any status update about the ICRP jobs started by the previous Virtuoso session, but you can use the job monitoring commands provided by your DRMS to monitor the status of the jobs. For example, you can use the `bjobs` command to monitor the LSF jobs. After verifying that all the ICRP jobs started by the previous Virtuoso session have completed, you can expect to view the results by reloading the history, as shown below.

© 2023 Cadence Design Systems, Inc.

a. On the *Results* tab, make sure to close any opened instance of the history that was running when the GUI exited.

b. On the *History* tab of the Data View assistant pane, right-click the history item that was running when the GUI exited and choose *Load Setup to Active* or *View Results*.

- If you reopen Virtuoso and ADE Assembler immediately after the GUI was closed abruptly and the in-process simulations are still running, you might not see the updated results after reloading the results. The results are available only after the simulations are complete and saved to the results database.

  To view the results:

  a. On the *Results* tab, make sure to close any opened instance of the history that was running when the GUI exited.

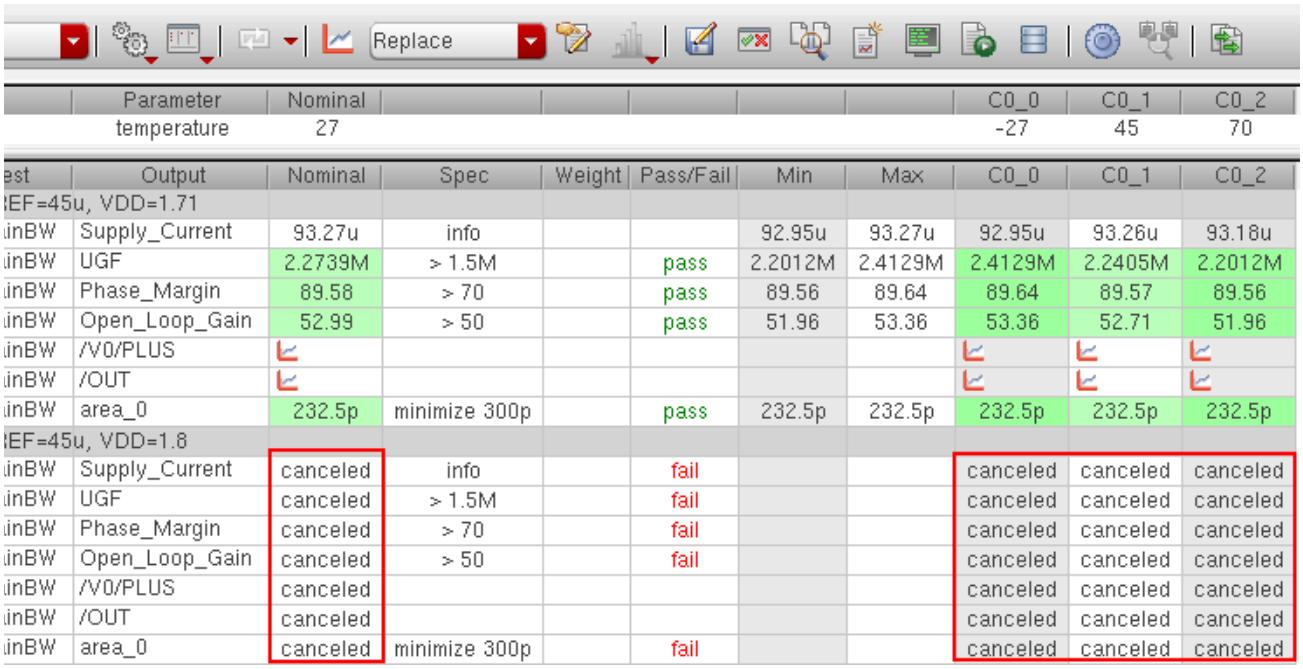
  b. On the *History* tab of the *Data View* assistant pane, right-click the history item that was running when the GUI exited and choose *Load Setup to Active* or *View Results*.

When the Simulations are Running and Virtuoso is Closed

By default, if the `continueICRPRunOnAbruptGUIExit` variable is set to `nil` and you close Virtuoso while simulations are running, Virtuoso displays a message to alert that some simulations are running and to confirm if you want to exit Virtuoso without completing those simulations.

If you click *Yes*, all running simulations are stopped and Virtuoso is closed. If you click *Yes*, simulations that are currently running are stopped and ADE Assembler is closed. Next time, when you view the history results, the status of the pending simulations is shown as `Canceled`, as shown below.

© 2023 Cadence Design Systems, Inc.



	Parameter	Nominal						C0_0	C0_1	C0_2
	temperature	27						-27	45	70

est	Output	Nominal	Spec	Weight	Pass/Fail	Min	Max	C0_0	C0_1	C0_2
EF=45u, VDD=1.71										
inBW	Supply_Current	93.27u	info			92.95u	93.27u	92.95u	93.26u	93.18u
inBW	UGF	2.2739M	> 1.5M		pass	2.2012M	2.4129M	2.4129M	2.2405M	2.2012M
inBW	Phase_Margin	89.58	> 70		pass	89.56	89.64	89.64	89.57	89.56
inBW	Open_Loop_Gain	52.99	> 50		pass	51.96	53.36	53.36	52.71	51.96
inBW	/V0/PLUS									
inBW	/OUT									
inBW	area_0	232.5p	minimize 300p		pass	232.5p	232.5p	232.5p	232.5p	232.5p
EF=45u, VDD=1.8										
inBW	Supply_Current	canceled	info		fail			canceled	canceled	canceled
inBW	UGF	canceled	> 1.5M		fail			canceled	canceled	canceled
inBW	Phase_Margin	canceled	> 70		fail			canceled	canceled	canceled
inBW	Open_Loop_Gain	canceled	> 50		fail			canceled	canceled	canceled
inBW	/V0/PLUS	canceled						canceled	canceled	canceled
inBW	/OUT	canceled						canceled	canceled	canceled
inBW	area_0	canceled	minimize 300p		fail			canceled	canceled	canceled

If you want that even if you close Virtuoso, the simulations that are already running should continue to run, set the `continueICRPRunOnAbruptGUIExit` variable is set to `t`. If this variable is set, when you close Virtuoso, it checks the state of simulation runs. There can be the following two cases:

- Simulation run has started, but no job is running when you close Virtuoso

  A message is displayed to alert that some simulation runs are pending to run and to prompt whether you want to exit without completing simulation of those points. To exit Virtuoso and cancel all the pending simulations, click *Yes*. Click *No* to stop from exiting Virtuoso.

- Simulation run has started and one or more jobs are running

  In this case, a message is displayed to confirm if you want to continue the running simulations or stop them after closing Virtuoso. To exit Virtuoso without completing the running simulation jobs, click *Exit and stop simulations*.

  To exit Virtuoso, but to continue running simulations, click *Exit but continue with running jobs*. Next time, when you run Virtuoso, you can load the history results to view the updated simulation results.

© 2023 Cadence Design Systems, Inc.

Limitations

The known limitations of this feature are as follows:

- While the ICRPs of a history are running to complete the in-process simulations, if you restart the ADE Assembler GUI, the new session is not connected to the already running ICRPs.
- A new ADE Assembler session will not share a license with the ICRPs that are already running to complete simulations for a history run. The new ADE Assembler session will require another license.
- If you open a new ADE Assembler session while the in-process simulations of a history are still running and load the results of that history, the status of the in-process runs appears as `running`. After the simulations are complete, the status is not refreshed automatically. To view the updated status, you need to reload the history by using the *Load Setup to Active* command in the context menu of the history on the History tab of the Data View pane.

Viewing Job Status

The Run Summary assistant displays the job status of simulation runs using one of the following views:

- Job List
- Job Summary

When you open the Run Summary assistant, by default, it displays the Job List.

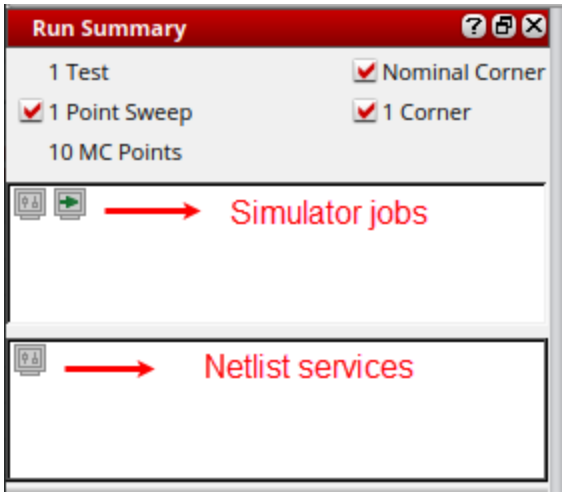
Important

The Job List and Job Summary views are available for all LSCS simulations, but the job status of ICRP simulations is only available in the Job List view.

© 2023 Cadence Design Systems, Inc.

Job List

For each job the program submits, a computer terminal icon appears in the Run Summary assistant pane.



This is the default Job List view where each icon indicates the status of its respective job as as follows:

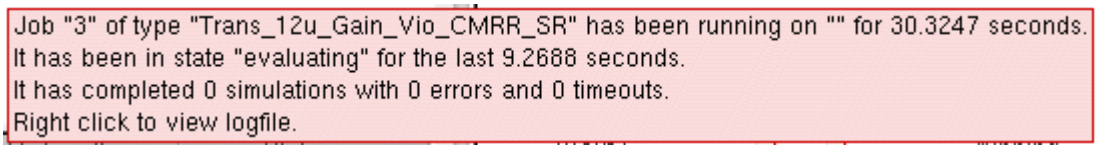
Icon	Job Status
	Pending
	Starting
	Configuring or Idle
	Running
	Suspended
	Cleaning up

Additionally, you can monitor the job status by doing the following:

➤ Hover the mouse cursor over the job status icon.

© 2023 Cadence Design Systems, Inc.

A job status pop-up appears.



The job status pop-up contains information about how long the job has been running, what state it is in, how many simulations have completed with how many errors and how many timeouts.

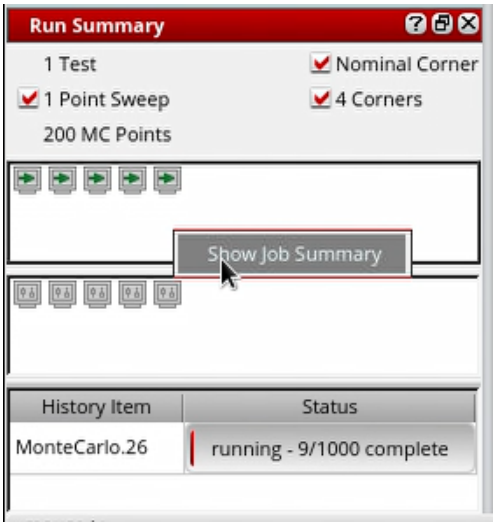
To view the log file while the program is writing it, do the following:

➤ Right-click the job status icon and choose *Job log*.

The log file appears in a separate window. Simulator messages appear in the window as the program writes them. As the window fills up with messages, the program scrolls down so that the most recent messages are always visible.

To switch to the Job Summary view:

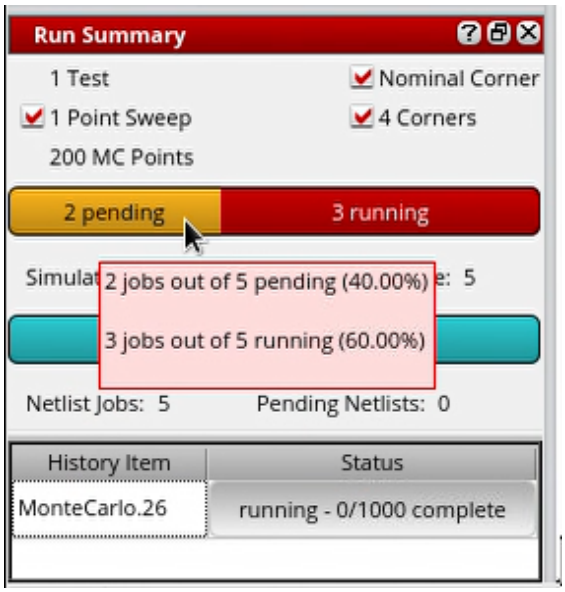
➤ Right-click anywhere in the Run Summary assistant and select *Show Job Summary*.



© 2023 Cadence Design Systems, Inc.

Job Summary

The Run Summary assistant provides a graphical view of all simulation and netlisting jobs in the Job Summary view. For all netlisting and simulation jobs that the tool submits, two progress bars are displayed in the Run Summary assistant pane.



The progress bars display the number of jobs in the `pending`, `starting`, `configuring` or `idle`, `running`, `suspended`, or `cleaning up` states, similar to the job icons available in the Job List view. The following details are displayed below the progress bars:

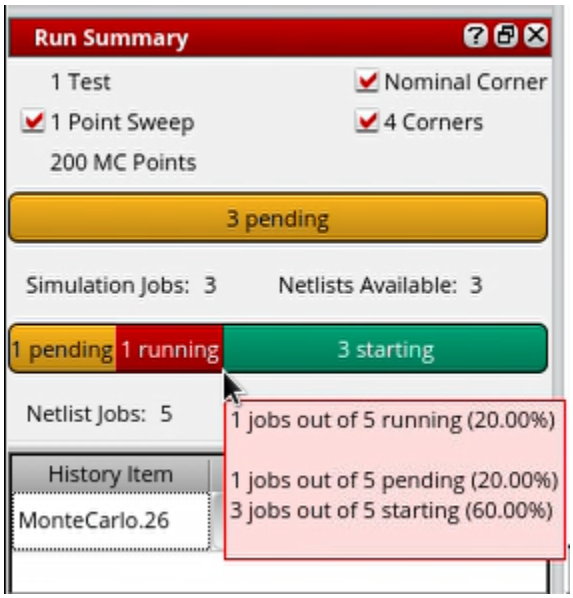
Label	Description
Simulation Jobs	The current number of active simulation jobs.
Netlist Jobs	The current number of active netlisting jobs.
Netlists Available	The number of netlists that have been processed but not simulated.
Pending Netlists	The number of netlists that still need to be processed.

Additionally, you can monitor the job status by doing the following:

➤ Hover the mouse cursor over the job status progress bar.

© 2023 Cadence Design Systems, Inc.

A job status tooltip appears.



The job status tooltip contains information about the number of pending, running, idle, and total jobs. The tooltip also displays the percentage of jobs that are in each simulation state. When you move the mouse pointer to any of the job state categories on the progress bar, the tooltip is updated dynamically to display the currently selected information on top.

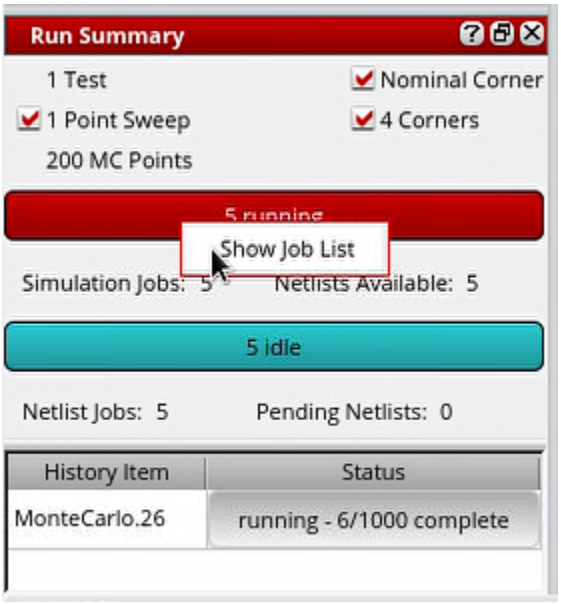
To set the default Run Summary view to the Job Summary, set the `displayJobSummaryBar` environment variable to `t`.

To switch to the Job List view from the Job Summary view:

➤ Right-click anywhere in the Run Summary assistant and select *Show Job List*.

Contrary to the Job List view, the *Job log* is not available in the Job Summary view. To view the job log files, switch to the Job List view.

© 2023 Cadence Design Systems, Inc.



Viewing the Netlist

To view the netlist file, do the following:

➤ On the Results tab of the Outputs pane:

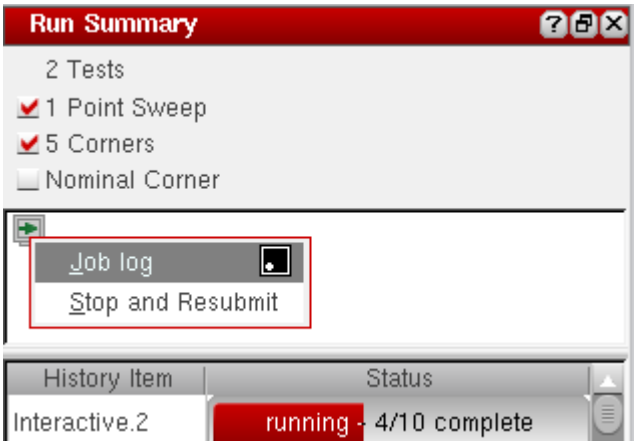
- ❑ Right-click a test or the result for the nominal corner and choose *View Netlist* to view the netlist for the nominal corner.
- ❑ Right-click the result for a corner and choose *View Netlist* to view the netlist for the corner.

The netlist file appears in a text window.

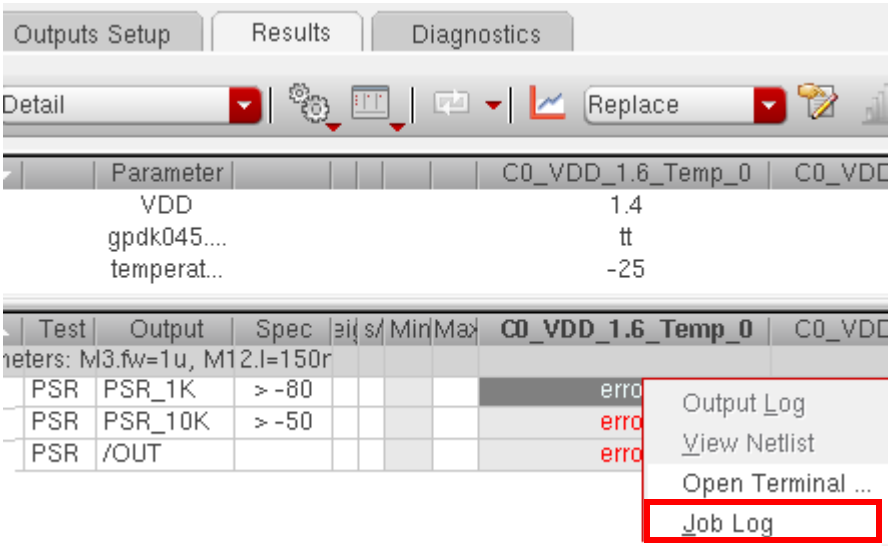
© 2023 Cadence Design Systems, Inc.

Viewing the ADE Assembler Logs After Running Simulations

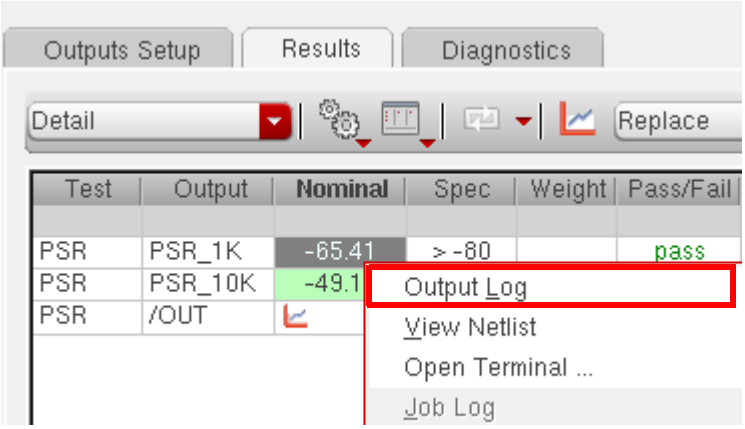
After running simulations, ADE Assembler saves the information of ICRP jobs, simulation runs, and outputs in different log files. This section describes the various types of log files ADE Assembler maintains and the details provided in those files.

Log File	Contains...
Job log (for an ICRP job)	Information about an ICRP job and all the simulations that were run on that job. It includes various details, such as the Cadence software version number, operating system version, working directory, the testbench details, variables and parameters for every run, simulator details, results location, data directory, netlist directory, and simulation errors (if any). Each ICRP has a separate log file. To view the job log for an ICRP, right-click an ICRP icon in the *Run Summary* assistant and choose *Job log*.  Also see: Viewing the Job Log

© 2023 Cadence Design Systems, Inc.

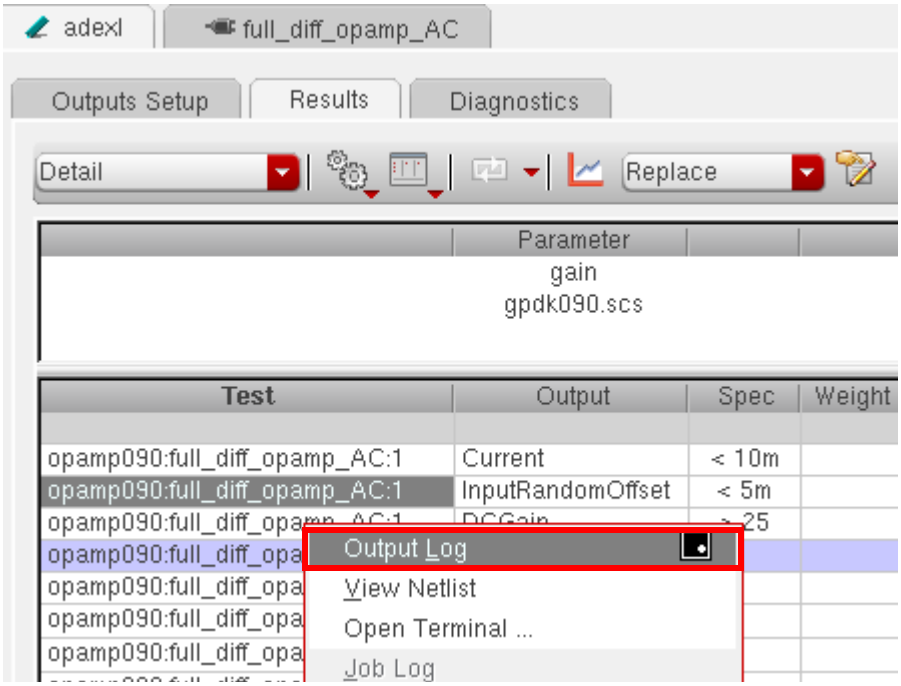
Log File	Contains...
Job log (for a data point)	Information related to the job for a selected data point. This log contains the details for a data point such as the variable and parameter values, the path to the netlist directory and the data directory, and the output or error details.To view the job log for a point, right-click any result value for that point on the *Results* tab and choose *Job Log*. The simulator log for that point is displayed in a new window.  Environment variable to control the creation of the job log for a data point: generateJobFileOnlyOnError At the data point level, ADE Assembler saves job logs only for the simulations that result into an error. To save the job logs for all the jobs irrespective of their result status, set the `generateJobFileOnlyOnError` environment variable to `nil`. However, this would increase the disk space requirement.

© 2023 Cadence Design Systems, Inc.

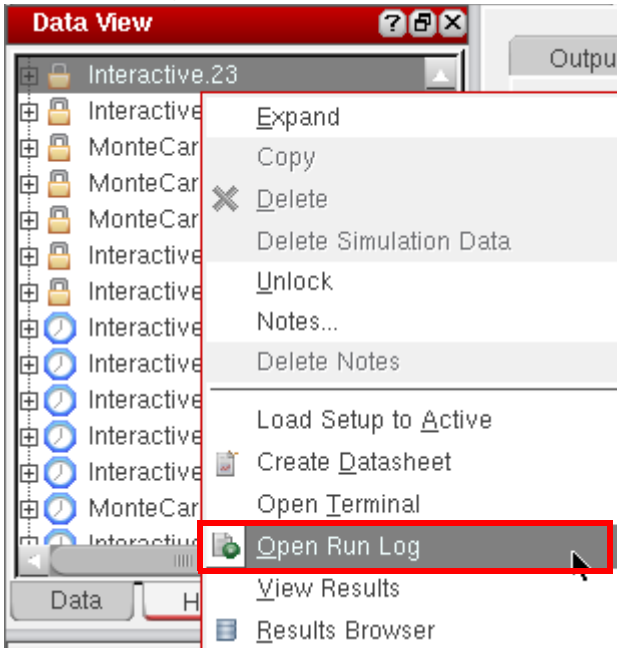
Log File	Contains...
Output log	The simulator output log for a particular data point. To view the output log for a point, right-click any result value for that point on the *Results* tab and choose *Output Log*.  Also see: Viewing the Simulation Output Log File showSimLogForOnePointSim

© 2023 Cadence Design Systems, Inc.

Log File	Contains...

Composite output log	The composite output logs for all the simulations run for all the points and corners. In this composite log, log for each simulation is separated by the following two statements:
	`********* LOG ENDS **************`
	`**  **`
	`**  **`
	`********* LOG STARTS ************`
	To view the composite output log, on the *Results* tab, right-click in the *Test* or *Output* column and choose *Output Log*.
	 
	
	Environment variable to control the creation of the composite output log depending on the number of simulations: createCompositeSimLogFileWhenSimCountFewerThan

© 2023 Cadence Design Systems, Inc.

Log File	Contains...
Run log	The summary of a simulation run in ADE Assembler. To view the run log, on the *History* tab of the *Data View* assistant, right-click a history item and choose *Open Run Log*.  This log contains the start time and end time for the simulation run, the number of points completed and the number of simulation errors. In addition to these, by default, ADE Assembler also saves the details of the best design point when the number of points is less than the count specified by createRunLogWhenSimsFewerThan or when the Single Run, Sweeps and Corners run mode is run as part of the Manual Tuning run mode. You can choose to include this information in other scenarios as well by using the following two environment variables: Environment variables to control the creation of the run job log: ■ createRunLogForSweepsCorners ■ createRunLogWhenSimsFewerThan

© 2023 Cadence Design Systems, Inc.

Viewing the Job Log

A job log can be viewed for a particular ICRP on which simulation is running or a particular data point in the Results tab.

Viewing Job Log for an ICRP

To view the log file for a particular ICRP job, do the following:

➡ In the Run Summary assistant pane, right-click the job status icon and click *Job Log*.

The Job.log file appears in a text window. This file contains information about the ICRP job that is running simulations. It includes various details, such as the Cadence software version number, operating system version, working directory, design path, simulator, results location, data directory, netlist directory, and simulation errors, if any.

How Virtuoso Saves the Job Logs?

By default, each Virtuoso process creates a unique log subdirectory such as `logs<num>` under `logs_<username>`, where all the ICRP processes started by that Virtuoso process write their job log files. For example, if you start a Virtuoso process from a directory and that process further starts three ICRP jobs, Virtuoso creates a directory `logs_<username>/logs0` and the three ICRPs create their job log files, `Job0.log, Job1.log` and `Job2.log` in `logs_<username>/logs0 directory`. While this Virtuoso process is running, if the user starts another Virtuoso process from the same directory, which further starts two ICRP jobs, the second Virtuoso process will create a directory `logs_<username>/logs1` and two ICRPs started by it write their job log files `Job0.log` and `Job1.log` in the `logs_<username>/logs1` directory.

Viewing Job Log for a Result

Note: By default, ADE Assembler saves job logs only for the simulations that result into an error.

To view the job log file for a particular data point in the Results tab, do the following:

➡ On the Results tab of the Outputs pane, right-click a result cell and choose *Job Log*.

© 2023 Cadence Design Systems, Inc.

The job log file for the selected result appears in a text window. This file contains all the information related to the job, such as the parameter values, the path to the netlist directory and the data directory, and the output or error details.

To view the job logs for multiple points in the same log window, do the following:

➡ On the Results tab of the Outputs pane, hold down the *Ctrl* key and click multiple cells, right-click and choose *Job Log*.

© 2023 Cadence Design Systems, Inc.

Important Points to Note

- ADE Assembler saves the job log files in the psf directory and can be viewed when you load the results for a history.
- By default, to enable debugging, ADE Assembler saves the job log files only for those jobs that failed due to some reason and did not give any result. The *Job Log* command is not enabled for the points that successfully showed results. To save the job logs for all the jobs, set the generateJobFileOnlyOnError environment variable to `nil`. However, this would increase the disk space requirement.

© 2023 Cadence Design Systems, Inc.

Viewing the Simulation Output Log File

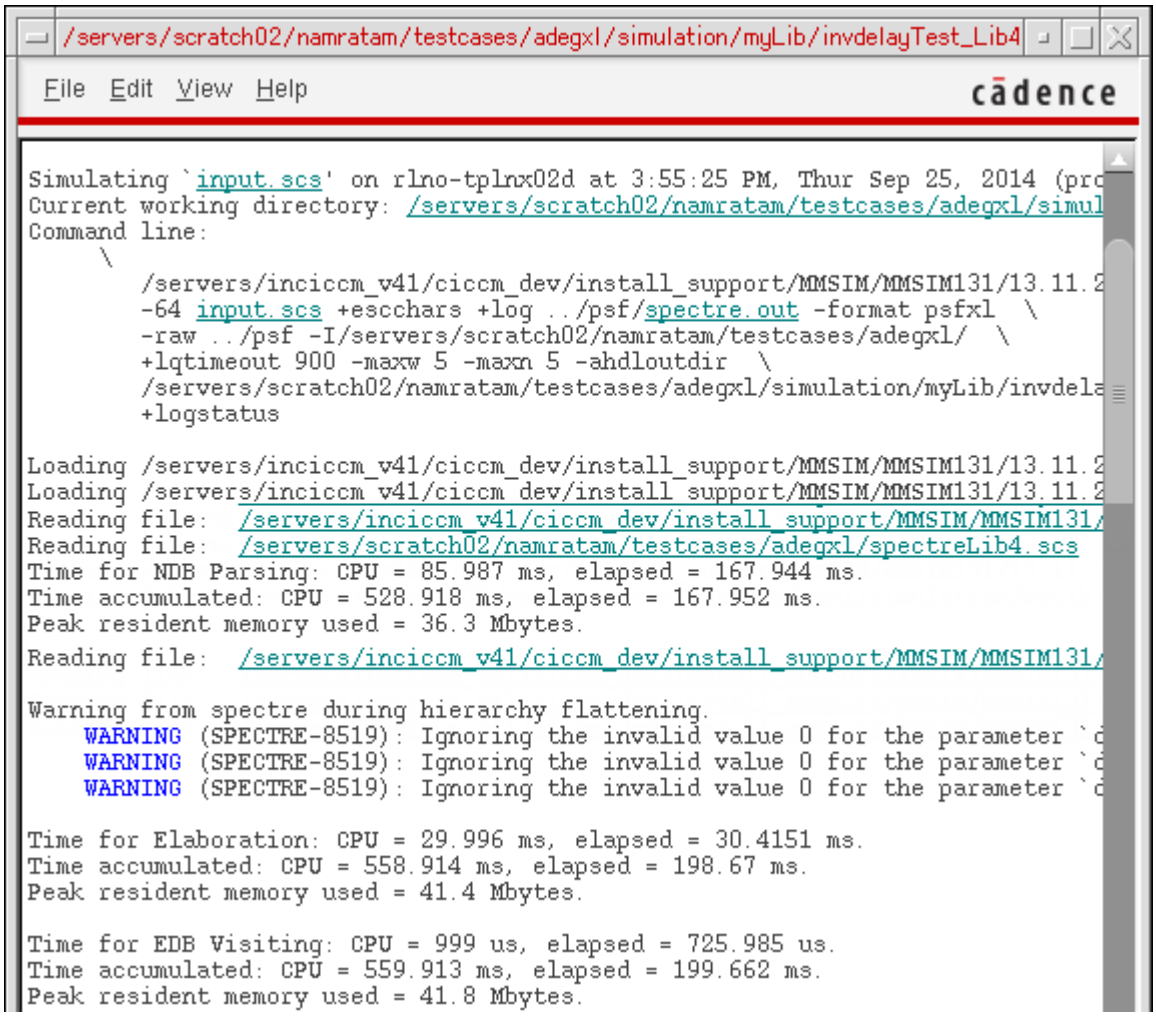
To view the simulation log file, do the following:

➤ On the Results tab of the Outputs pane:

- Right-click a test or the result for the nominal corner and choose *Output Log* to view the simulation log file for the nominal corner.
- Right-click the result for a corner and choose *Output Log* to view the simulation log file for the corner.

The simulation output log file appears in a text window. This file contains information about the simulation environment, the command line sent to the simulator, the details of analyses, the simulation summary, and any simulation error or warning messages.

An example of the output log is shown below.



© 2023 Cadence Design Systems, Inc.

The log contains hyperlinks that provide direct access to the netlist file, input files, instances in the schematic, the setup forms for analyses, and error or warning information. These links are helpful in debugging. For more details on the hyperlinked logs, refer to Viewing the Simulation Results in the *Virtuoso ADE Explorer User Guide*.

Also see:

- Simulation Errors
- showSimLogForOnePointSim

Simulation Errors

If the simulation of an analysis or evaluation of an expression fails, ADE Assembler reports appropriate errors in the Results tab or in the output log. The errors can be categorized as described in the following table.

Type of error	Reported when...
netlist error *(error)*	■ Generation of netlist fails because of some reason, such as a missing device master.
simulation error *(sim err)*	■ A simulation was killed. This can be due to various reasons such as process killed in the background or insufficient disk space. ■ The analysis on which an expression is dependent failed due to errors. In this case, all the expressions that depend on the failed analysis show *sim err* in the results table. Other outputs are not affected due to this failure.
evaluation error *(eval err)*	■ An expression was wrongly constructed, and therefore, could not be properly evaluated ■ No data was generated after simulation due to some reason ■ The analysis on which evaluation of an expression is dependent was not selected for simulation **Note:** In case of evaluation errors, the min and max values on the Results tab and in Spec Summary are calculated by using the numeric values of the successful results. The results with errors are ignored. The tooltip of the result value also indicates this.

© 2023 Cadence Design Systems, Inc.

Important Points to Note

- You can rerun the design points that could not run because of simulation errors. However, you cannot rerun the design points that failed with eval errors because they will result in the same error again.
- If the eval error occurs because of an incorrectly built expression, you can update the expression on the *Output Setup* tab and click *Re-evaluate* on the *Results* tab to rerun the expression evaluation.
- If the following ASSEMBLER-2200 error message is reported:

```
ERROR (ASSEMBLER-2200): Encountered the following problem with the results
database engine:
SQL statement: SELECT count(name) FROM sqlite_master WHERE type='table' and
name='psfTable';
Error code: 6
Description: database schema is locked: main.
The results database uses SQL statements internally.
Contact Cadence Customer Support to report the problem.
```

  You can try to resolve the issue as described below.

  Check the value of the RDBCreatePolicy variable to identify the type of database being used for the simulation run. If RDBCreatePolicy is set to `"TmpDir"`, ensure that the temporary location has sufficient space to save result database and simulation waveforms. Alternatively, you can reset the RDBCreatePolicy variable to `"NFSDisk"` to write the results on an NFS disk, and then rerun simulation. If the problem persists, contact Cadence Customer Support.

See also:

- Viewing the Simulation Output Log File
- Viewing the Job Log
- Simulating Only Unfinished or Erroneous Points
- showOutputLogOnError
- useLocalPsfDir

© 2023 Cadence Design Systems, Inc.

Running an Incremental Simulation

The results for simulation runs in ADE Assembler are stored in history items. An incremental simulation allows you to use one or more histories as reference for a new simulation run for the following purposes:

- Running an Incremental Simulation to Reuse Results
- Running an Incremental Monte Carlo Simulation
- Reusing a Netlist from Another History
- Rerunning Simulation after Modifying the Netlist

Important

You can run an incremental simulation for the Single Run, Sweeps and Corners, and Monte Carlo Sampling run modes only. However, for Monte Carlo Sampling, you can use only one reference history.

Also see: Viewing Differences Between the Active Setup and the Reference History Setup

Running an Incremental Simulation to Reuse Results

After running a simulation, you can make certain changes in the setup. For example, you can add more corners or sweep points, or modify the values of the existing points. After making these changes, you may want to simulate only the changed corners or sweep points and copy the results for the remaining points from the previous history.

For example, you change the value of a variable `vdd` from `1,2,3` to `1,2,5`. In such a case, you can use the previous history as reference for the next simulation run. For the common or matching points (when `vdd=1` or `vdd=2`) results are reused from the reference history and a simulation is run only for the new value (`vdd=5`).

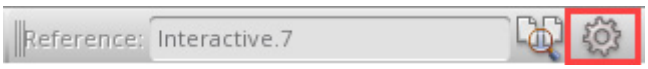
This is also helpful when you have to correct any value of a corner or sweep point.

To run an incremental run by reusing the results, do the following:

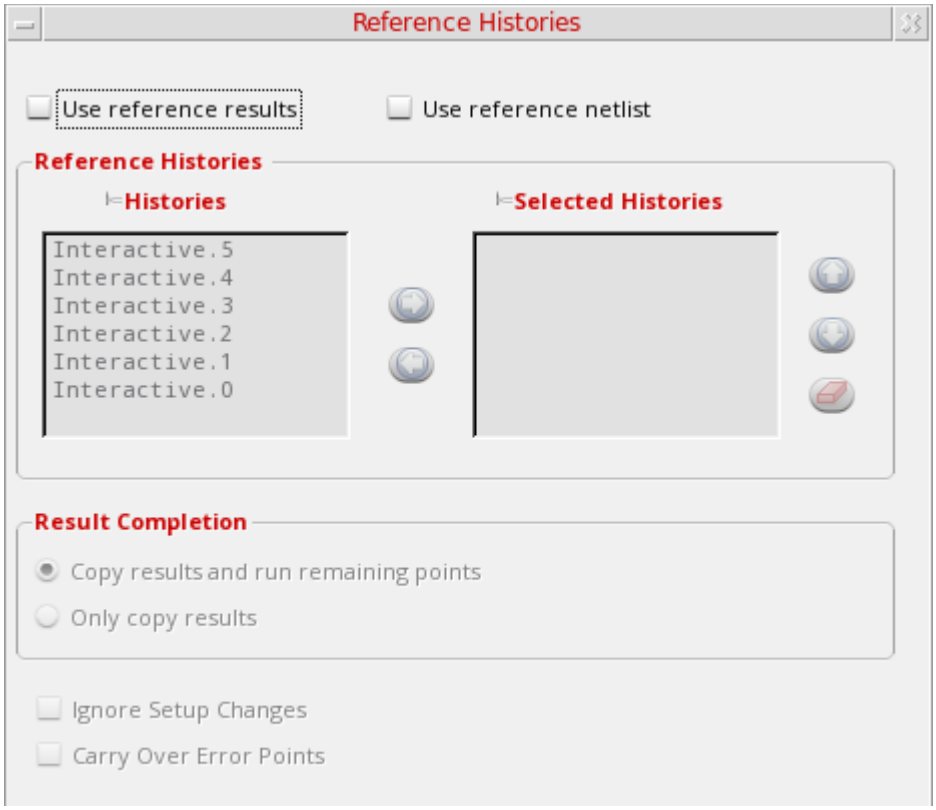
1. Identify the reference histories in the *History* tab of the Data View assistant.
2. Right-click one of the identified histories and click *Load Setup to Active*.
3. Select a run mode from the *Select a Run Mode* list on the Run toolbar.

© 2023 Cadence Design Systems, Inc.

4. Click *Reference History Options* on the Reference History toolbar.



The Reference Histories form is displayed.



5. Select the *Use reference results* check box.

   All other fields on this form are enabled.

6. From the *Histories* list box, select the histories you want to use as the reference histories and click to add them to the *Selected Histories* list box.

   You can use the following buttons to manage the selected reference histories:

   - : To remove the selected history name from the *Selected Histories* list.
   - and : To change the order of histories in the *Selected Histories* list.
   - : To clear the *Selected Histories* list.

   Note: The *Histories* list does not show the names of children histories saved for grouped runs. For example, Sensitivity Analysis.

© 2023 Cadence Design Systems, Inc.

7. (Optional) If you want to use a reference netlist, select the *Use reference netlist* check box.
8. In the *Result Completion* group box, specify how you want to populate the results for the incremental simulation run.
   - *Copy results and run remaining points*: By default, this option is selected. It copies the results for common points from the reference history and runs simulation for the remaining points. For the points that resulted into errors in the reference history, the error status is retained. However, if the *Carry Over Error Points* check box is deselected, error points are simulated again.
   - *Only copy results*: This option copies only the results for common points from the reference history and does not run simulation for the remaining points. The status of any canceled or error points in the reference history is retained. In this case, all new points are shown as canceled.
9. (Optional) Select the *Ignore Setup Changes* check box to run an incremental simulation even if there are any test-level changes in the active setup. For example, analysis options have changed.

   By default, ADE Assembler does not run an incremental simulation by using a reference history if the test setup, variables, or output setup are different in the reference history and the active setup. The tool displays detailed warning messages in the CIW listing the differences in the setup.

   Select this option if you want to ignore the changes and run an incremental simulation. However, it is recommended to use this check box only when you are making minor modifications to the test-level setup because the results of an incremental simulation that is run by ignoring any setup changes might differ from the results of a non-incremental simulation because the latter considers the setup changes.

   Consider an example of a test AC where an initial simulation is run with the value of variable `vdd` set to `1, 2, 3`. Before running an incremental simulation, you make the following two changes:

   - Change the value of `vdd` to `1, 2, 5`
   - Change the Accuracy Defaults option for the tran analysis from *moderate* to *liberal*.

   Next, select the *Ignore Setup Changes* check box in the *Reference Histories* form and run an incremental simulation by using the previous history as reference. For the common data points (where `vdd = 1` or `2`), the new history ignores the change in the transient options and copies the results from the reference history. For `vdd=5`, no results are found in the reference history, therefore, ADE Assembler runs a new simulation by using the updated transient option, which is Accuracy Defaults = *liberal*.

© 2023 Cadence Design Systems, Inc.

In comparison to this, if you run a non-incremental simulation after the two changes mentioned above, for all values of `vdd`, ADE Assembler uses the Accuracy Defaults transient option set to *liberal*. This might cause a difference in the results of the incremental and non-incremental simulation.

10. (Optional) Select *Carry Over Error Points* if you want to retain the status of error points from the reference history. When this check box is deselected, ADE Assembler also simulates the points that resulted in errors in the previous run.

Note: ADE Assembler considers this check box only when the *Copy results and run remaining points* option is selected in the *Result Completion* group box.

11. Click *OK* to save the changes and close the Reference Histories form.

Important

The color of the *Run Simulation* command on the Run toolbar changes to blue. This gives a visual indication that the setup for the next simulation uses reference histories.

12. You can view the selected reference history items in the Reference History toolbar.

13. (Optional) Click *Show Differences* on the Reference History toolbar to view the differences between the active setup and the setup in the reference history. For more information, see Viewing Differences Between the Active Setup and the Reference History Setup.

Note: The *Show Differences* command is available only if a single reference history is selected.

14. On the Run toolbar, click *Run Simulation* to start the incremental simulation run.

If the *Use Reference Netlist* option is selected, a message box appears to seek your confirmation that you want to use the netlist of the reference history for the incremental run. Click *Yes* to continue.

Observe that a new history for the incremental run is added to the Run Summary assistant.

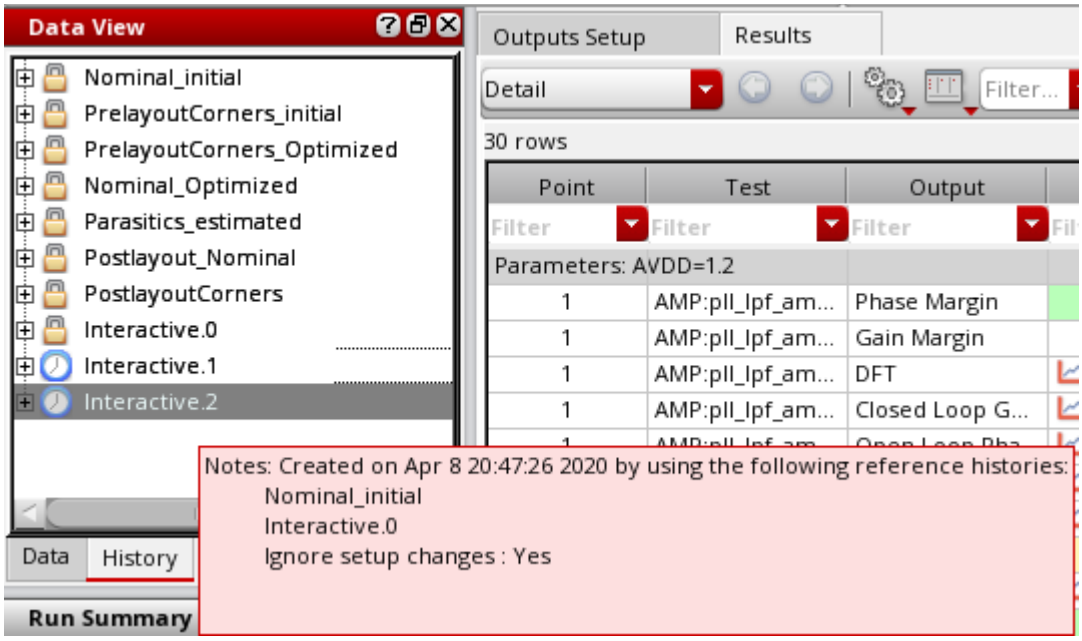
© 2023 Cadence Design Systems, Inc.

Important Points to Note

- The incremental simulation run uses the active setup.
- The run mode of the current run can be different from the run modes of the selected reference histories.
- The number of completed points displayed on the run status reflects the number of points for which the results are already available in the referenced history. Result data for those points is reused from the reference history. Simulation is then continued for the missing and failed points in the reference histories.
- If the same sweep or corner is used in more than one reference history, the results from the first reference history in the list box are considered.
- If a test contains variables defined using `calcVal` or `VAR`, results are not reused from a reference history. ADE Assembler reruns simulations for those tests.
- If you modify the outputs or their spec values in the active setup as compared to the reference history, a warning message is displayed and the Incremental simulation is not performed because the setup is incompatible with the setup in the reference history.
- The tooltip of a history saved for a simulation run using reference histories shows the following details:
  - The date and time of the history
  - The names of the reference histories used for the run

© 2023 Cadence Design Systems, Inc.

- The status of the *Ignore Setup Changes* check box on the Reference Histories form.



- ADE Assembler locks all writable histories used as reference histories. It displays an error if you use the *Unlock* command to unlock a reference history. You cannot remove the lock from a history until you remove it from the *Selected Histories* list on the Reference Histories form.

© 2023 Cadence Design Systems, Inc.

Running an Incremental Monte Carlo Simulation

Similar to Single Run, Sweeps, and Corners run mode, you can use one or more reference histories for Monte Carlo Sampling run mode.

To use a reference history for a Monte Carlo simulation:

1. Select *Monte Carlo Sampling* run mode on the *Run* toolbar.
2. Configure the reference history settings as described in the Running an Incremental Simulation to Reuse Results section.

   If the multiple reference histories contain similar points, the results are used from the first found occurrence. Any duplicate point is ignored.

   Statistical parameter data is not copied from the reference history to the new history.

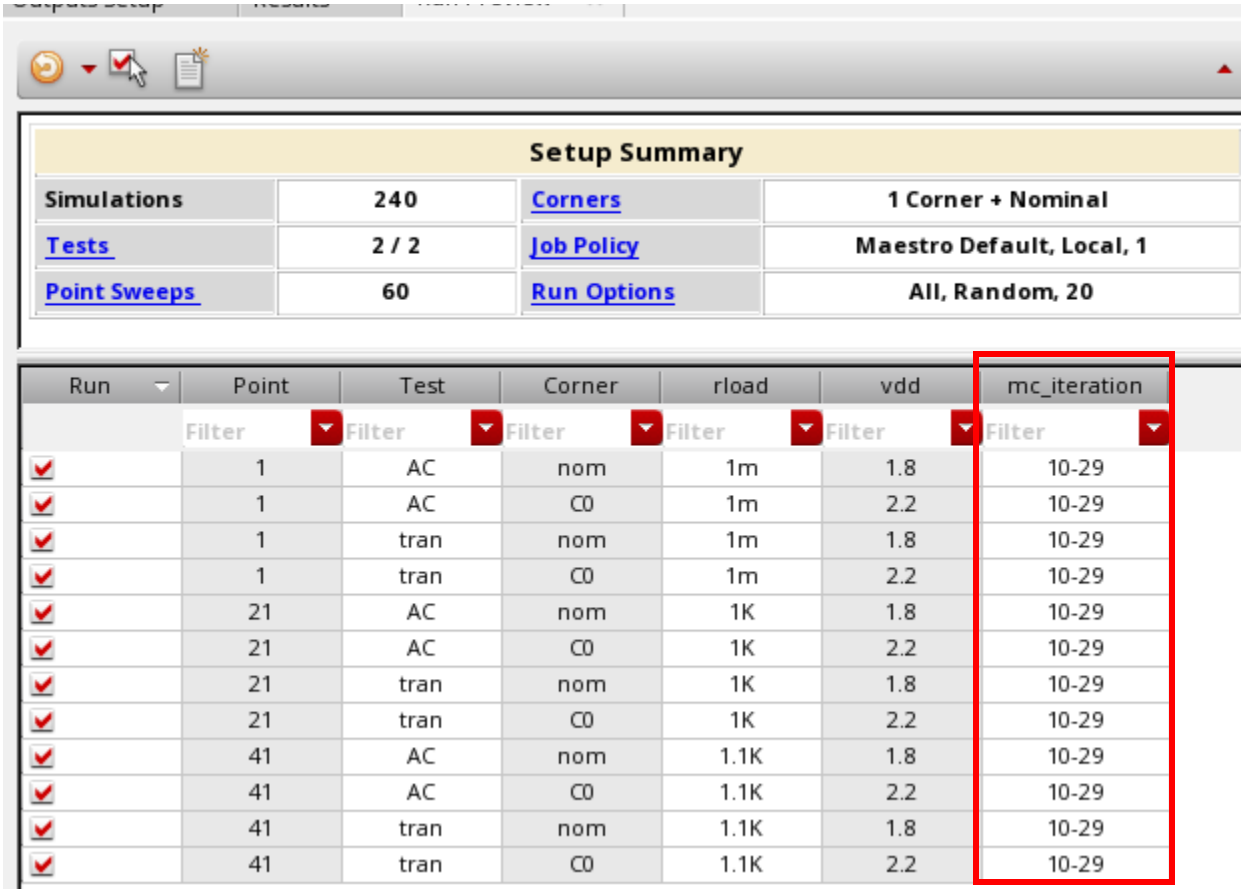
This feature is supported only when in the Monte Carlo options form:

- The *Method* field is set to `Standard Monte Carlo`
- The *Sampling Method* field is set to `Random` or `Low-Discrepancy Sequence`
- The *Save Statistical Parameter Data* check box is not selected

If you are running a Monte Carlo simulation using a fixed number of points and not to verify yield or to create statistical corners, you can review the setup and run points in the *Run Preview* table before running a simulation. All Monte Carlo iterations for a particular point,

© 2023 Cadence Design Systems, Inc.

test, corner, and variable combination are combined in a single row where the *mc_iteration* column shows the range of points. You can enable or disable points to control the run.



Setup Summary

Simulations	240	Corners	1 Corner + Nominal
Tests	2 / 2	Job Policy	Maestro Default, Local, 1
Point Sweeps	60	Run Options	All, Random, 20

Run	Point	Test	Corner	rload	vdd	mc_iteration
	Filter	Filter	Filter	Filter	Filter	Filter
☑	1	AC	nom	1m	1.8	10-29
☑	1	AC	C0	1m	2.2	10-29
☑	1	tran	nom	1m	1.8	10-29
☑	1	tran	C0	1m	2.2	10-29
☑	21	AC	nom	1K	1.8	10-29
☑	21	AC	C0	1K	2.2	10-29
☑	21	tran	nom	1K	1.8	10-29
☑	21	tran	C0	1K	2.2	10-29
☑	41	AC	nom	1.1K	1.8	10-29
☑	41	AC	C0	1.1K	2.2	10-29
☑	41	tran	nom	1.1K	1.8	10-29
☑	41	tran	C0	1.1K	2.2	10-29

While configuring an incremental simulation, you can select the *Ignore Setup Changes* check box on the Reference Histories form to ignore the following types of setup changes:

- A change in the maximum number of points
- A change in the first point
- Changes in simulation settings to improve convergence

Multiple references are useful in the following scenarios of a Monte Carlo simulation:

- **Running an incremental Monte Carlo simulation for different tests:** You have already run a Monte Carlo simulation for a test, say `test1`, and saved the history as `Monte1`. You want to run a simulation for another test, `test2` , with the same number of points and the same variable and corner settings. By using `Monte1` as a reference history, you can merge the results from `Monte1` in the history of the second run `Monte2` for a complete result analysis without rerunning the simulation of `test1`.

© 2023 Cadence Design Systems, Inc.

- **Running an incremental Monte Carlo simulation for the same tests but different points:** You have already run two Monte Carlo simulations. `Monte1` contains the results for the first 100 points and `Monte2` contains the results for points 101 to 200. Now, you want to run a simulation for a few more points and also include the points already simulated. Instead of rerunning the already run points, you can use `Monte1` and `Monte2` as reference histories. The results from all the histories are available in the third run `Monte3`.
- **Running an incremental Monte Carlo simulation with different settings for an improved convergence:** You have run a Monte Carlo simulation for 100 points out of which 8 failed. In the second run, you change the simulation settings and run the same points out of which only 3 failed. Now, you again want to change the settings to improve the results. In this case, you can use both of the previous histories as reference histories and use the results for points that did not return errors. Run an incremental run for only those points that had errors and reuse the results for the remaining points.

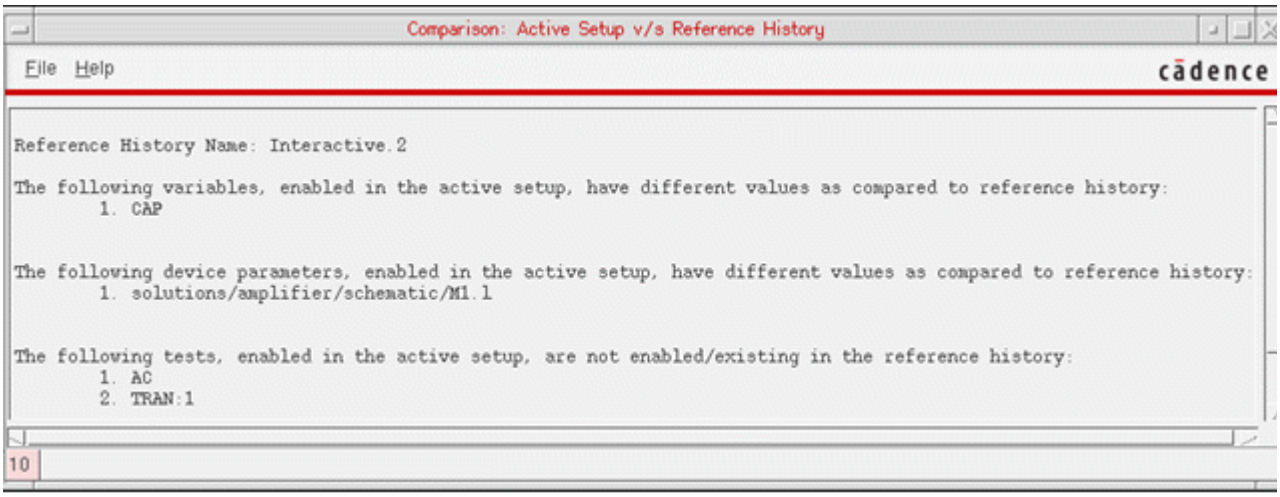
Related Topics: Using the Run Preview Table

Viewing Differences Between the Active Setup and the Reference History Setup

To view the differences between the active setup and the setup in the reference history, do the following:

➡ Click the *Show Differences* button on the Reference History toolbar.

The differences between the active setup and the setup in the reference history are displayed in the Comparison: Active Setup v/s Reference History form.



© 2023 Cadence Design Systems, Inc.

Reusing a Netlist from Another History

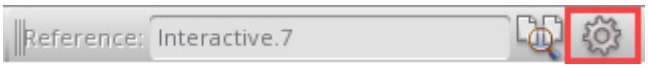
Reusing a netlist saved by a previous run is helpful in the following scenarios:

- When you have changed the design, but you do not want to consider the changes in a simulation run.
- When the netlist creation takes a huge time and you are sure that reusing a netlist will save time.

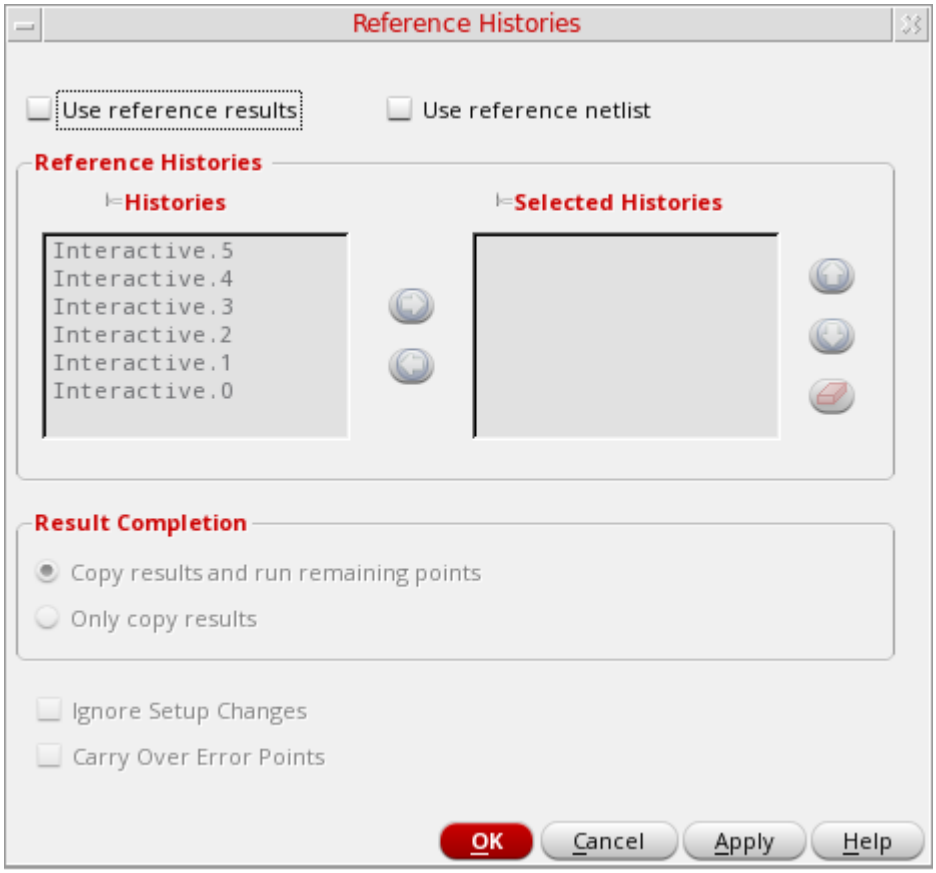
Note: You can reuse a netlist only when you are using a single reference history.

To reuse a netlist from a simulation run in a subsequent simulation, do the following:

1. In the setup for a new simulation, click *Reference History Options* on the Reference History toolbar.



The Reference Histories form is displayed.



© 2023 Cadence Design Systems, Inc.

2. Select the *Use reference netlist* check box.

   The *Reference Histories* section is enabled.

3. In the *Histories* list, select a history whose netlist you need to use in the next run.

   Note: The tool allows you to select histories for which netlist is successfully created and the simulation has finished or simulation is not complete yet.

4. Click the right arrow icon to move the selected history to the *Selected History* list.

   Note: Select only one reference history. Netlists cannot be reused from more than one history.

5. Click *OK* to close the form.

The color of the *Run Simulation* command on the Run toolbar changes to blue. This gives a visual indication that the setup for the next simulation uses a reference netlist.

When you run the next simulation, ADE Assembler compares the setup of the reference history with the active setup and reuses the netlist as described below.

- If the tests are same in both, it copies the netlist from the reference history to the netlist directory for each test in the current history.
- (LSCS job control mode only) If the setup contains config sweeps in global variables or corners, for each swept cellview, the tool uses the reference netlist from the corresponding point in the reference history.

  For example, if the setup contains corners C1, C2, C3, where each corner uses a different config cellview, ADE Assembler uses the reference netlist from the corresponding corner in the specified reference history. This feature is not available in the ICRP job control mode, which uses the netlist of the nominal corner for each swept cellview.
- If there are any differences in the setup and the netlist cannot be reused, ADE Assembler displays an appropriate message and stops the simulation run.

Related Topic

Creating Config Sweep Variables

Creating Corners

© 2023 Cadence Design Systems, Inc.

Rerunning Simulation after Modifying the Netlist

After running a simulation, you might need to make some manual modifications in the netlist and rerun a simulation without regenerating a new netlist using ADE Assembler. In some cases, you might need to completely replace the netlist file with another one and run a simulation with the existing ADE Assembler setup.

In the scenarios described above, you can perform the following steps to modify or replace a netlist file used in the previous run and use it for the next simulation:

1. After the reference simulation run is complete, note the name of a history saved for it. For example, `Interactive.3`.
2. At the command prompt, traverse to the directory where the netlist was saved for that history.

   Note: By default, the netlist file is saved in the simulation results directory specified by the saveResDir or saveDir environment variables in the given order of preference.

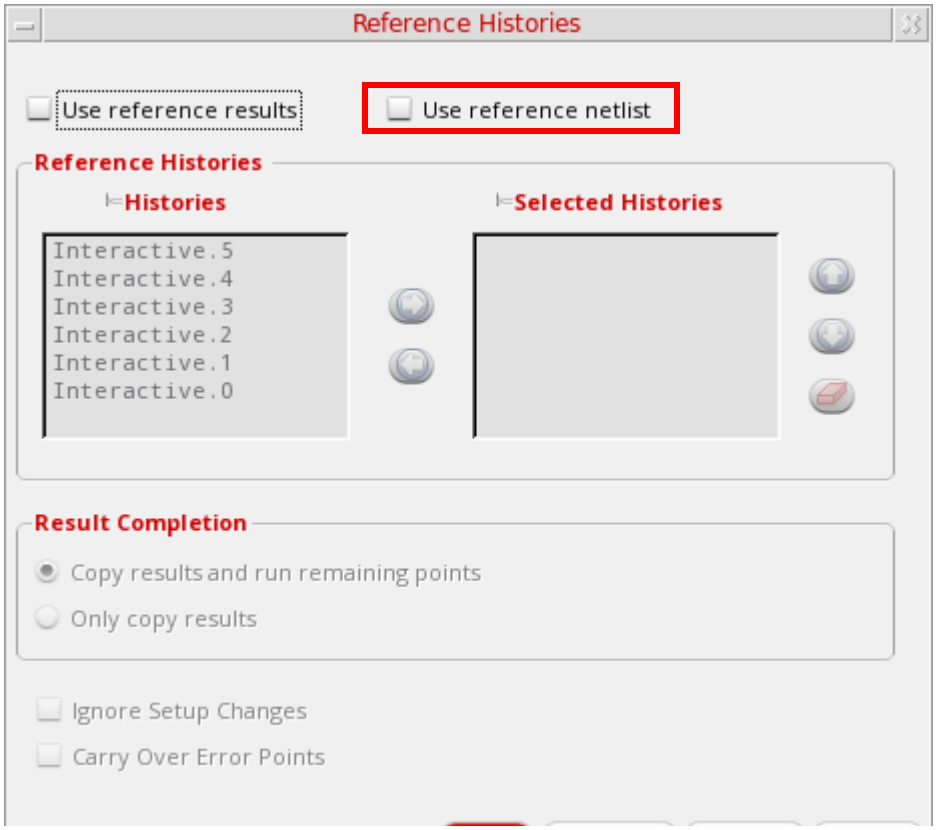
   For example, if the name of the history item for the test `ACGainBW` is `Interactive.14`, the netlist file for this history is saved in the *<sim-dir-path>*/simulation/*<libName>*/*<cellName>*/maestro/results/data/Interactive.14/psf/ACGainBW directory.
3. Open the netlist file in a text editor and make changes as required.
4. Save and close the netlist file.

   Note: You can also replace the existing netlist file with another netlist for the same design.
5. In the ADE Assembler environment, click *Reference History Options* on the Reference History toolbar.

   The Reference History form is displayed.

© 2023 Cadence Design Systems, Inc.

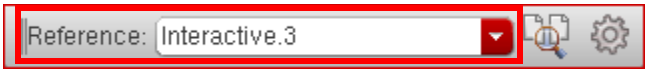
6. In the Reference History form, select the *Use reference netlist* check box, as shown below.



When the *Use reference netlist* mode is set, ADE Assembler does not generate netlist during the next simulation run. Instead, it uses the netlist file available in the results directory of the referenced history name. In this case, this is the netlist file that was edited in **step 3**.

7. Click *OK* to close the form.

8. From the *Reference* drop-down list on the Reference History toolbar, select the name of a reference history corresponding to the edited netlist file.



9. Run simulation.

   ADE Assembler copies the edited netlist file from the results directory for `Interactive.3` to the results directory for the new simulation run.

© 2023 Cadence Design Systems, Inc.

Simulating Only Unfinished or Erroneous Points

If a history item includes points that are incomplete or that show errors, it could be because the simulation run was stopped before completion, or some simulations failed due to issues like license check failures, an unavailable model file, system issues on a remote host used for distributed simulation etc. For such history items, you can run incremental simulations to simulate only the incomplete or error points in the history item. This helps in saving time because the results for the completed points will be reused from the previous history item and the simulations will not run again for those points.

ADE Assembler provides three ways in which you can run only those points that had errors or that were incomplete in the previous run:

- By using the Rerun Unfinished/Erroneous Points command
- By using the Rerun Point command to rerun specific points
- By setting reference to the simulation results of a previous run

Rerunning All Unfinished or Erroneous Points

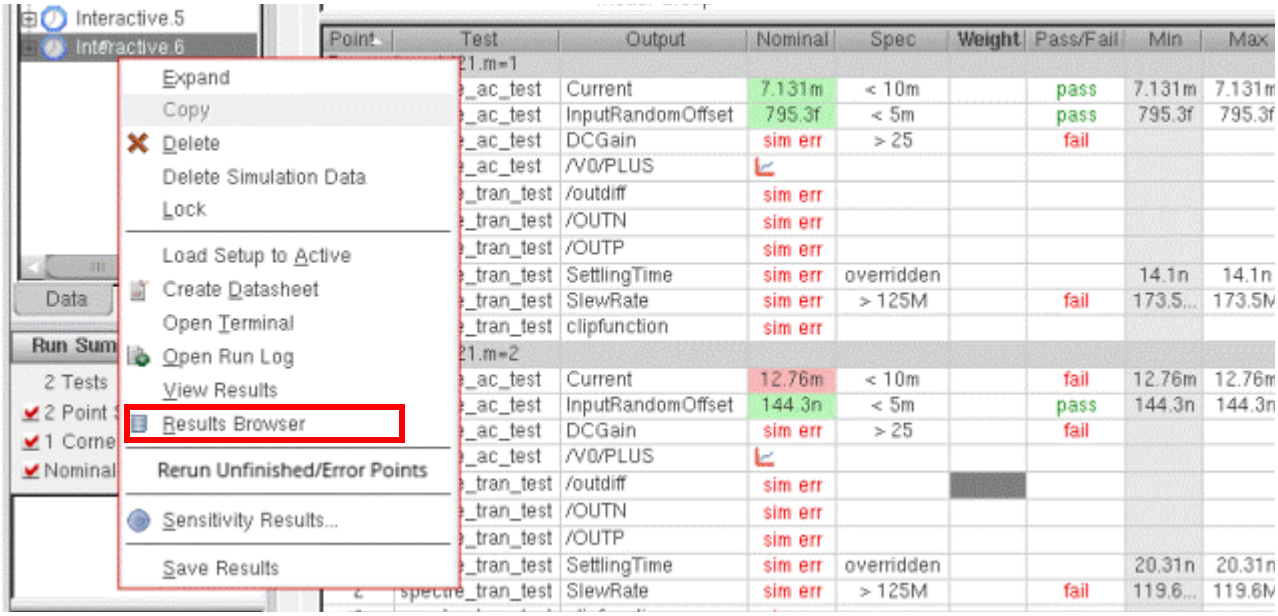
This method is useful in scenarios when the design has not changed and you want to use the previous simulation setup while completing the unfinished points. If the design has changed, a new netlist will be generated, which may be different from the one used in the original run with unfinished points.

To simulate all the unfinished or erroneous points in a history item, do the following:

1. Ensure that the History tab of the Data View assistant pane is open.

© 2023 Cadence Design Systems, Inc.

2. Right-click the desired history item that contains erroneous points and choose *Rerun Unfinished/Error Points*.



When you use this command, ADE Assembler ignores any changes in the active setup after the last run and re-runs simulations for all the erroneous or unfinished points in the selected history run. All the consolidated results are saved in a new history named *`history_name`*`.rerun.`*`seqNum`*, where *`seqNum`* starts from `0` and increases incrementally in the subsequent simulation runs.

Note the following:

- The simulation run uses the setup of the history item on which you ran *Rerun Unfinished/Error Points.* So any changes you make in the active setup will be ignored during the run.
- When you run *Rerun Unfinished/Error Points* on a history item, the simulation results of that history item are moved to the new history item. As a result, you will not be able to perform postprocessing operations (like plotting, printing, annotation, re-evaluation, and so on) on the history item on which you ran *Rerun Unfinished/Error Points*. You can use the `retainReferenceSimResults` environment variable to retain the simulation results of the history item on which you ran *Rerun Unfinished/Error Points*. For more information about this environment variable, see retainReferenceSimResults.

Related Topics

Rerun Simulations for Erroneous or Unfinished Points in a Run Plan

© 2023 Cadence Design Systems, Inc.

retainNetlistLinkForRerun

Rerunning Specific Unfinished or Erroneous Points

To rerun simulation for specific unfinished or erroneous points:

1. In the *Results* tab, press the `ctrl` key and select the points to be simulated again.
2. Right-click any of the selected points and choose *Manage Point Results – Rerun Point* to rerun the simulation.

   The suffix `rerun` is added to the name of the related history.

Note: This feature is available only for *Detail* and *Detail - Transpose* result views.

Related Topics

Rerun Simulations for Erroneous or Unfinished Points in a Run Plan

© 2023 Cadence Design Systems, Inc.

Running Erroneous or Incomplete Points By Referencing the Simulation Results of a Previous Run

Note: This method is useful when the design has changed, but you want to reuse the previous netlist and simulation setup while completing the unfinished points.

In this method, you can reference the netlist and the existing simulation results of a previous run. For details on the pre-requisites of this method and steps to run an incremental simulation, refer to Running an Incremental Simulation to Reuse Results.

© 2023 Cadence Design Systems, Inc.

Setting Up Debug Options

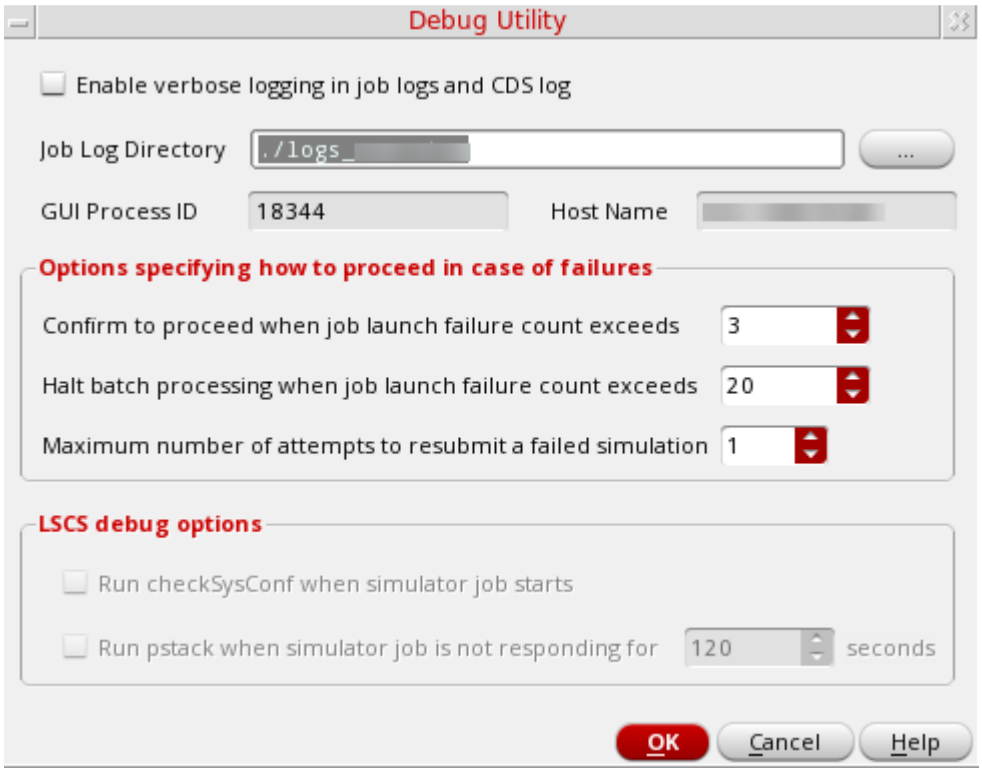
Note: Read a blog related to this at Virtuosity: Let's Have Fun with ADE Debugging – Part 1.

You can set debugging options that help you troubleshoot several problems, such as non-responsive user-interface, simulation not starting, simulations showing error or warning messages, and so on. When set, these debug options are instantly applied on all the active simulation jobs, however, the current simulation run remains unaffected.

Perform the following steps to set the debugging options:

1. Choose *Tools – Debug*.

   The Debug Utility form appears.



2. Specify the following fields to set the debugging options:

   a. *Enable verbose logging in job logs and CDS log*—Select this check box to enable the debugging options listed below:

   - Write the debug information and prepend a time-stamp in each debug message
   - Write the callback evaluation information

© 2023 Cadence Design Systems, Inc.

- Capture the full stack trace if an error is thrown
- Write the Monte Carlo debug information
- Write the calcVal debug information
- Write the debug information for wave spec evaluation or re-evaluation
- Print the DE integration debug information and log all SKILL_CALLS issued by the UI
- Report information during runObjFile and interactive plotting operations
- Write the OCEAN debug information

When you click this check box, the following environment variables are enabled to write the std out and std error messages to the CDS.log file:

```
"adexl.icrpStartup" "showJobStdout" 'boolean t
"adexl.icrpStartup" "showJobStderr" 'boolean t
```

b. *Job Log Directory*—Specify a location where you want to redirect the job logs. This field is useful when you select the *Enable verbose logging in job logs and CDS log* field and want to write the job log files to a different location so that they can be shared for further debugging.

Alternatively, you can set the location using the following environment variable:

```
"adexl.distribute" "jobFileDir" 'string "<myDir>"\
```

c. *GUI Process ID*—Show the GUI process ID.

d. *Host Name*—Show the host name of the computer on which you are running Virtuoso.

e. *Options specifying how to proceed in case of failures*—Set the following fields to specify how you want to proceed when failures are found:

- *Confirm to proceed when job launch failure counts exceeds*—Specify a count after which if a job launch is failed, an error message is displayed.

  For example, if you specify this field value as `3`, then if the job fails three times, no error message is displayed. However, if the job continues to fail for the fourth time, an error message is displayed in CIW. This indicates that the number of attempts permitted to retry a job launch in the given case is three.

  Alternatively, you can set the following environment variable to specify how many number of times to retry a job launch before reporting the error:

© 2023 Cadence Design Systems, Inc.

```
"adexl.distribute" "maxJobFailPerPolicy" 'int 1000
```

- *Halt batch processing when job log failure count exceeds*—Specify a count after which if a job has failed to launch, the batch processing is stopped. The value you specify is applied to all the scripts generated from the current session.

  Alternatively, you can set the following environment variable to specify how many number of times a job launch can be retried before stopping the batch processing of jobs:

  ```
  "adexl.distribute" "maxJobFailPerPolicyInBatch" 'int 1000
  ```

- *Maximum number of attempts to resubmit a failed simulation*—Specify how many times ADE Explorer can try resubmitting a failed simulation. By default, ADE Explorer resubmits a failed simulation only once, and if the simulation fails again, an error message is displayed.

  Alternatively, you can set the following environment variable to specify the maximum number of attempts to resubmit a failed simulation:

  ```
  envSetVal("adexl.distribute" "numRetriesOnError" 'int 2)
  ```

f. Click *OK* to apply the specified settings.

The options in the *LSCS debug options* group box are enabled only when *LSCS* job control mode is enabled. For more details, see Using Job Logs for Debugging.

Important Points to Note

- The debug options settings specified in this form will be applied on all the jobs (ICRP sessions) that are new or waiting to run a simulation. To apply these changes on the currently running jobs as well, restart the jobs by right-clicking the ICRP icon in the Run Summary assistant and choosing *Stop and Resubmit*.
- The default values displayed for the debug options in this form are the values specified for them in the related environment variables.
- The debug options settings specified in this form temporarily override the specified `.cdsenv` or `.cdsinit` settings.
- The form settings are applicable only to the current Virtuoso and ICRP sessions, and form fields will show the default values in the next sessions.

© 2023 Cadence Design Systems, Inc.

Debugging Points

After you run simulations in ADE Assembler, if any particular data point fails or shows undesired results, you can selectively change the setup of that data point and debug it. For this, you can use the ADE Assembler Debug Environment.

Note: You cannot use the debug environment when a simulation is run using Spectre X simulation performance mode.

The ADE Assembler Debug Environment is a debug environment where you can load the setup of a data point you want to debug. The entire setup for the data point selected in ADE Assembler including the test, variables, and analyses details, corners, outputs, simulation settings, and job policy gets loaded. You can modify the settings for debugging purpose, run simulations, and plotting results. You cannot perform the following tasks in the debug environment:

- Saving and loading states
- Changing the simulator or simulation run mode
- Changing the switch view list
- Running parametric analysis

Note: Simulations that run from the debug environment do not use ICRP. They are same as ADE L simulations.

After debugging the data point, you can bring back the corrected simulation setup to the maestro view. You can then run simulation by using the updated setup in the maestro view.

To debug a data point by using the debug environment, do the following:

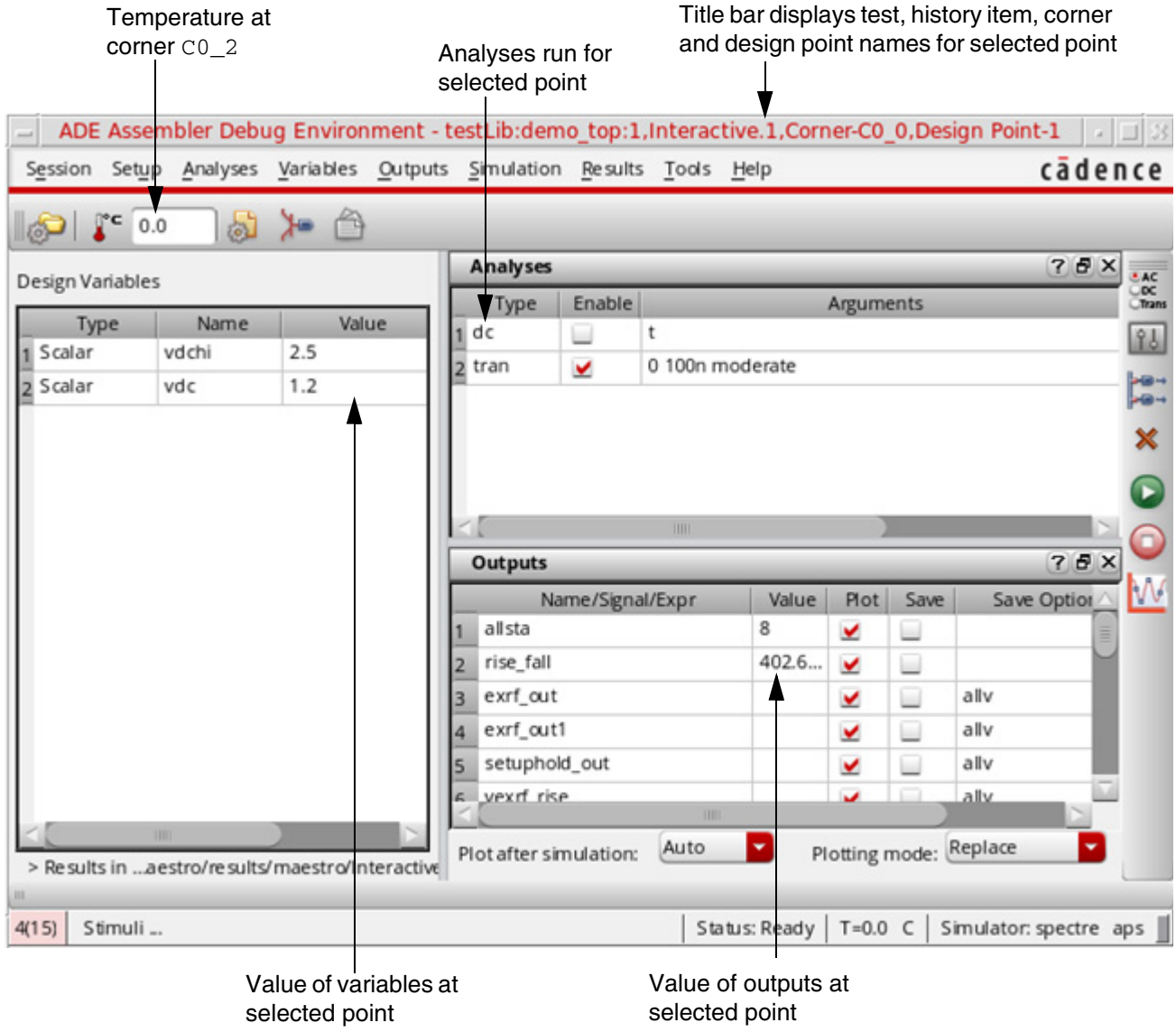
1. On the Results tab, right-click the data point and choose *Manage Point Results – Open Debug Environment*.

   Note: This command is disabled for the results of a simulation that was run using the Spectre X simulation performance mode.

   The ADE Assembler Debug Environment window appears displaying the setup for the data point. For example, the following figure displays the setup for corner `C0_2` of test `myTest` at data point `3` in history item `Interactive.1`.

© 2023 Cadence Design Systems, Inc.

Figure 17-1 ADE Assembler Debug Environment

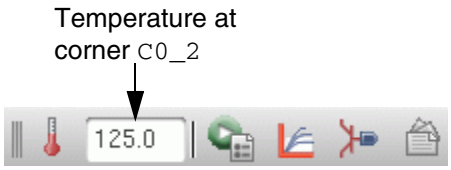


Note the following:

- As shown in the figure given above, the ADE Assembler Debug Environment window title bar displays the test name, history item name, corner name and design point ID for the selected point.

© 2023 Cadence Design Systems, Inc.

- The *Design Variables* pane displays the value of the design variables at the selected data point. The *Type* column indicates the type of the variables, as described below:

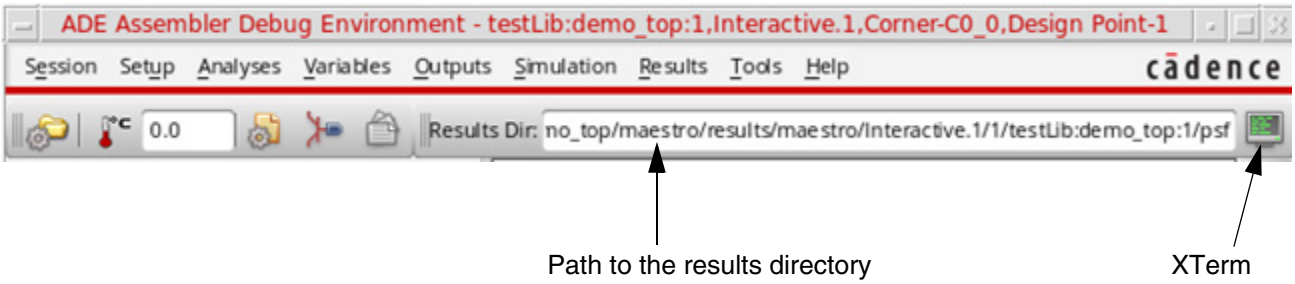
Type	Description
Test	Design variable from test
Sweep	Global variable or device parameter with sweep (multiple) values
Scalar	Global variable or device parameter with single value
Constraint	Matched or ratioed device parameter **Note:** You can modify only the value of the master parameter. Values of the matched and ratioed device parameters will change accordingly.
Corner	Design variables and parameters specified for the corner **Note:** The value of the *Temperature* variable specified for the corner is displayed on the toolbar, as shown below: 

- The default project directory for the data point is the `/results/data/`*`historyName`*`/`*`pointID`*`/`*`testName`*`/debug` directory of the maestro view.

  Note: You can choose *Setup – Simulator/Directory/Host* to modify the project directory path.

© 2023 Cadence Design Systems, Inc.

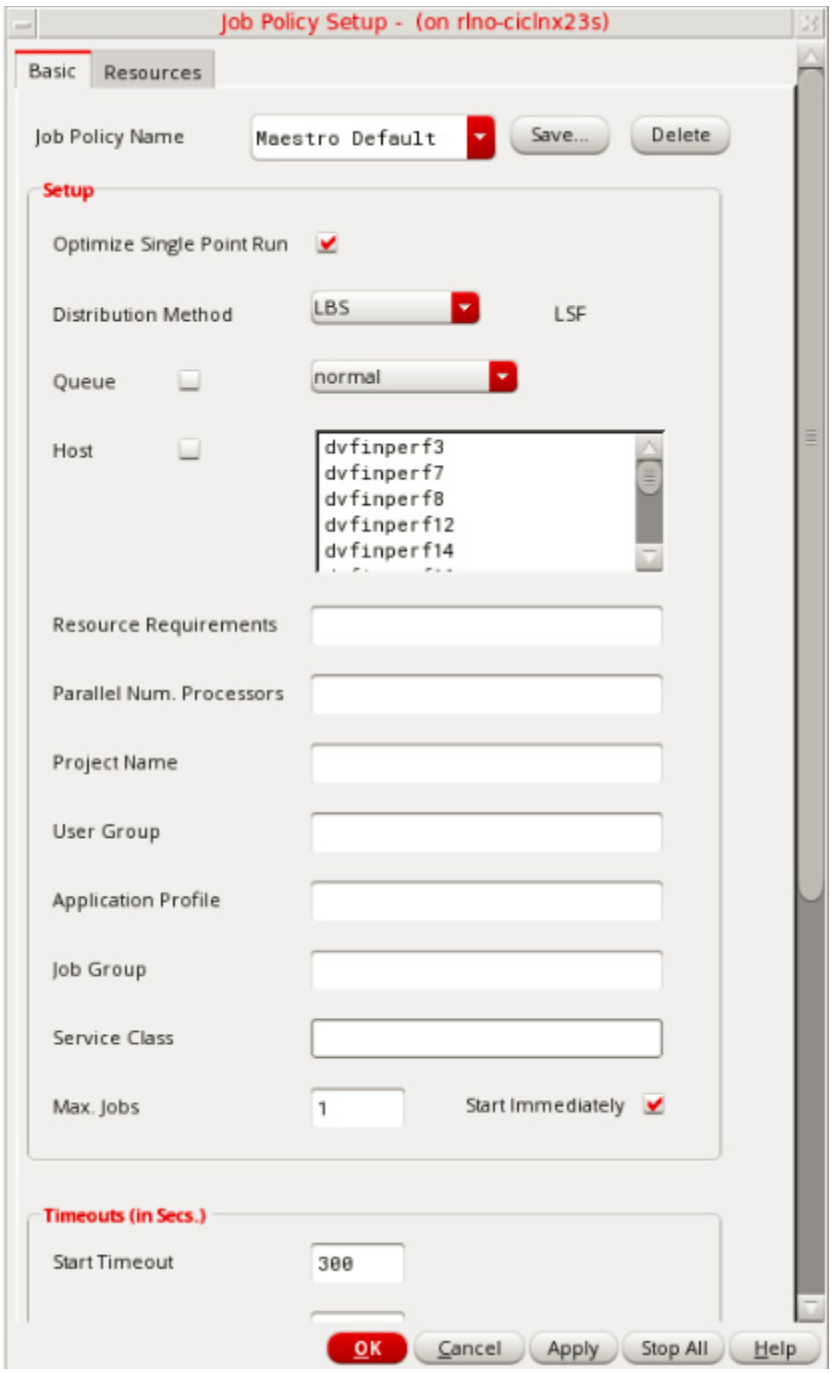
- You can see the path of the results directory for the data point in the *ADE Result Directory* toolbar. By default, this toolbar is hidden. You can right-click any toolbar and choose *ADE Result Directory* to make it visible.



  Click *XTerm* to open the terminal window from where you can access the `netlist` or `psf` directories for the data point.

- The relevant job policy settings set up in the ADE Assembler environment are copied to the debug environment. For example, if job policy is set in the ADE Assembler environment as shown below:

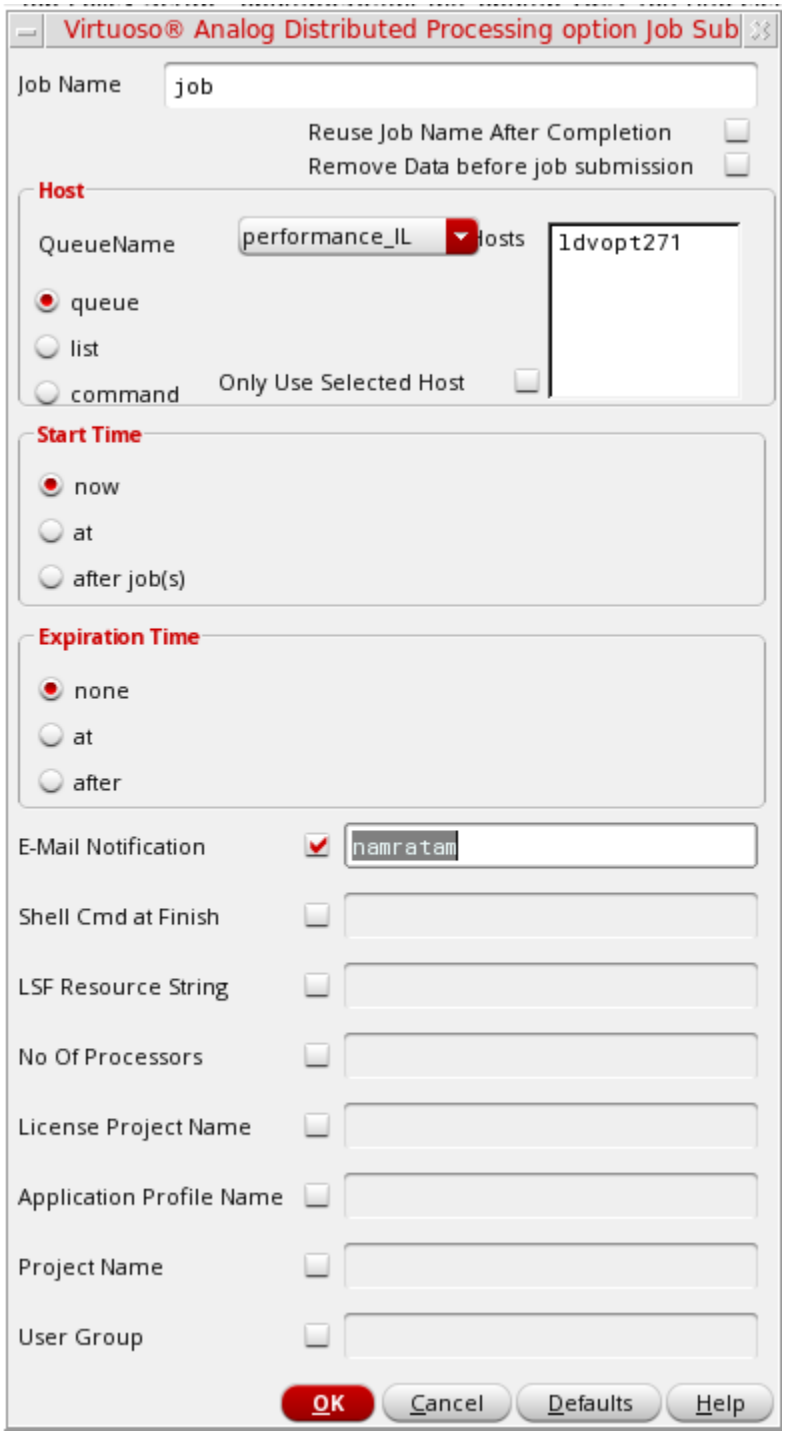
© 2023 Cadence Design Systems, Inc.



In the form shown above, settings are done for distributed processing by using the LBS method. Relevant settings for distributed processing are copied from this form

© 2023 Cadence Design Systems, Inc.

to the Analog Distributed Processing option Job Submit form in the debug environment, as shown below:



© 2023 Cadence Design Systems, Inc.

- You cannot use the debug environment to debug results from OCEAN-based tests. For more information about OCEAN-based tests, see Working with OCEAN-Based Tests on page 154.
- Any existing `.cdsinit` variables set by the user (for example, the variables to set the simulation directory or the host run mode) are applicable to the debug environment.

2. Modify the setup for the data point as required and run simulations to debug the data point.
3. (Optional) Choose *Session – Setup Back To Assembler* and click *Yes* in the message box that appears to bring the corrected setup back to the ADE Assembler view.

If you click *Yes*, ADE Assembler overwrites the active setup with the setup information in the history item for which you debugged the data point. For example, if you debug a data point in a history item named `Interactive.2`, ADE Assembler overwrites the active setup with the setup information in `Interactive.2`. Then, the changes in the debug environment setup are merged with the active setup as described below:

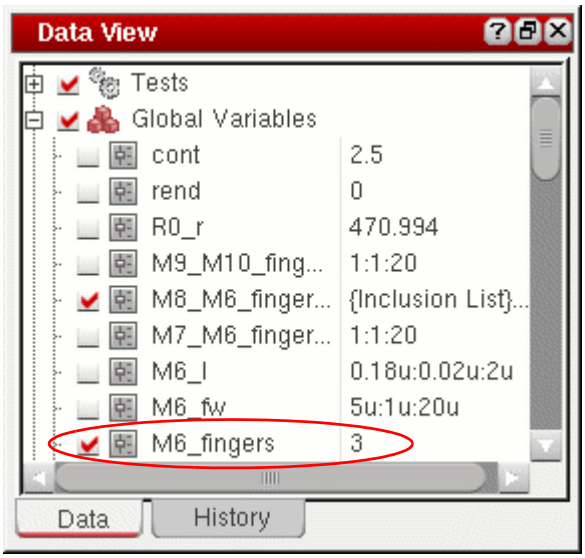
- Changes in design variables, analyses and outputs for a test in the debug environment are merged with the setup for that test in the maestro view.
- Changes in scalar variables (global variables or device parameters with single value) in the debug environment overwrite the value of the corresponding global variables and device parameters in the maestro view.
- Changes in sweep variables (global variables or device parameters with sweep values) in the debug environment are appended to the sweep range for the corresponding global variables and design parameters in the maestro view.
- Changes in corner values in the debug environment result in a new corner being created in the maestro view. The new corner is enabled only for the test for which you debugged the data point.

For example, if you choose *Session – Setup Back To Assembler* after making the following changes in the debug environment, as shown in the figure:

- Modified the value of the scalar variable `M6_fingers` from `1` to `3`

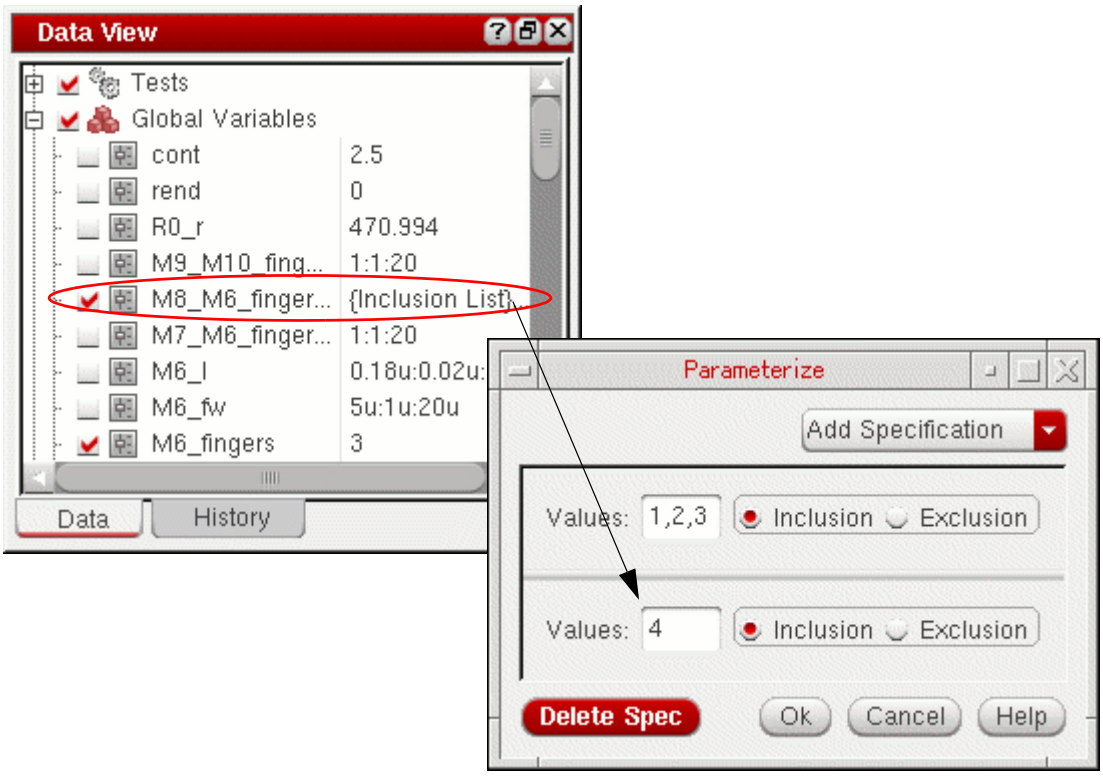
© 2023 Cadence Design Systems, Inc.

- Modified the value of the sweep variable `M8_M6_fingers_ratio` from `3` to `4`
- Modified the value of the *Temperature* variable for the corner to `120`The changes are merged into the active setup as described below:
- The value of the scalar global variable `M6_fingers` changes to `3`.



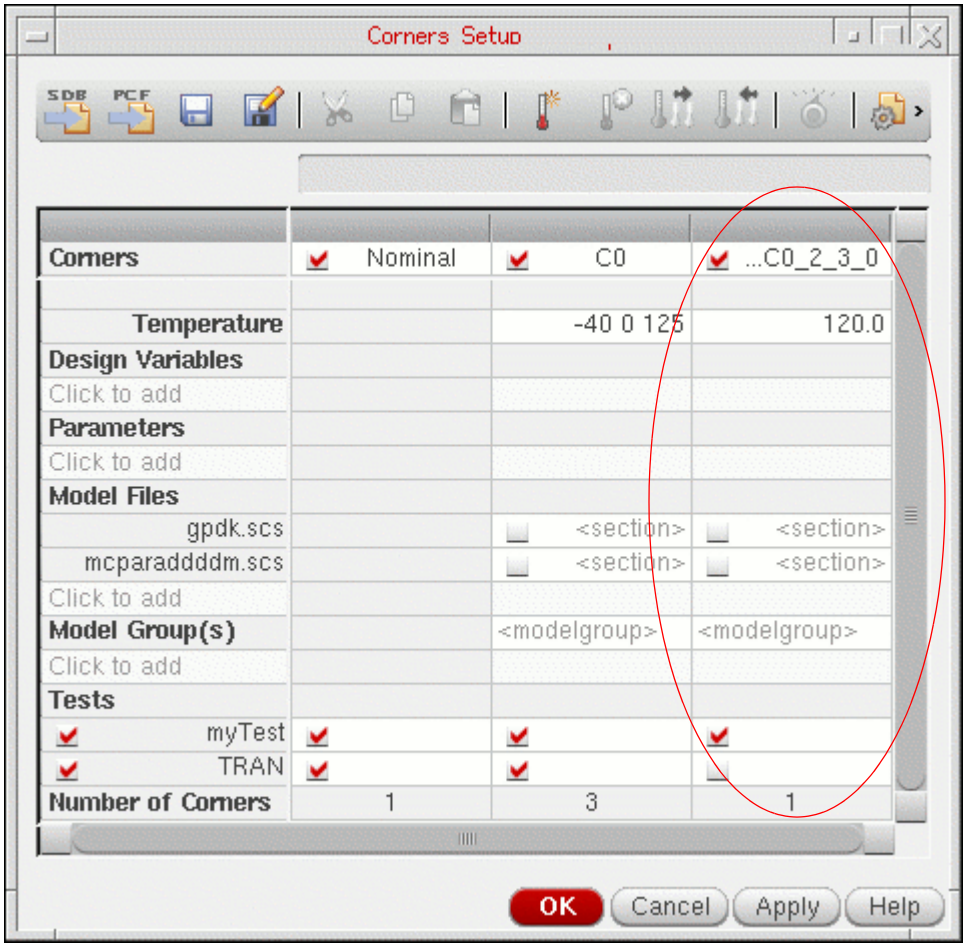
© 2023 Cadence Design Systems, Inc.

- The value `4` specified for the sweep global variable `M8_M6_fingers_ratio` is included in the sweep range for the variable.



© 2023 Cadence Design Systems, Inc.

- A new corner named `C0_2_3_0` which is enabled for test `myTest` for which you debugged the data point is created because of the change in the value of the *Temperature* variable for corner `C0_2` from `125` to `120` in the debug environment.



4. (Optional) Run simulation with the modified setup.

Important Point to Note

It is important to note the following point while working in the debug environment:

- If you try to open a debug environment for a point while this environment is already open for another point, the set up of the previous point is replaced with the new point. However, if a simulation is already running for the previous point, the following confirmation message is displayed.

© 2023 Cadence Design Systems, Inc.

Click *No* to continue with the simulation already running.

Click *Yes* to stop the simulation that is already running. The simulation running from the debug environment is stopped and the partial simulation results are plotted. In addition, the setup details of the new debug point overwrite the setup of the previous data point.

© 2023 Cadence Design Systems, Inc.

Creating and Running an OCEAN Script

You can create and run an OCEAN script to run one or more tests over zero or more corner conditions. When you save an OCEAN script, the program saves setup information for all your tests, sweeps, corner conditions and Monte Carlo analyses.

For more details, refer to the following sections:

- Creating an OCEAN Script
- Modifying an OCEAN Script in ADE Assembler
- Running an OCEAN Script
- Running Parallel OCEAN XL Simulation Runs for a maestro View
- Viewing Results of Simulations Run using OCEAN Scripts

Creating an OCEAN Script

You can create an OCEAN script in any of the following two ways:

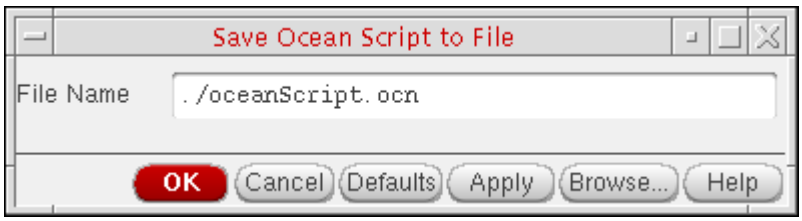
- Saving the ADE Assembler Setup to an OCEAN Script
- Scripting an OCEAN Script
- Creating OCEAN Scripts with Netlist Specified as a Design

Saving the ADE Assembler Setup to an OCEAN Script

You can create a simulation setup in ADE Assembler user interface and save it as an OCEAN script. For this, after the simulation setup is complete, do the following:

1. In the main session window, choose *File – Save Script*.

   The Save OCEAN Script form appears.



2. In the *File Name* field, type a name and location for your OCEAN script file.

© 2023 Cadence Design Systems, Inc.

Note: The default name and location is `./oceanScript.ocn`.

3. Click *OK*.

   The program saves simulation setup to the specified OCEAN script file and shows the file in a text editor.

   By default, ADE Assembler opens the text editor set by using the `$EDITOR` shell variable in UNIX. You can specify a preferred text editor by using the `editor` SKILL variable.

Note: The setup information for non-ADE Assembler tests will not be saved in the OCEAN script.

Scripting an OCEAN Script

You can use a text editor to create an OCEAN script by using OCEAN commands and save it in a `.ocn` file.

Creating OCEAN Scripts with Netlist Specified as a Design

If you have an already generated netlist of a design, you can provide the path to that netlist file to the `design` OCEAN command. By doing this, you can run a simulation without loading/requiring the complete design hierarchy. Later, you can view the simulation results in ADE Assembler user interface. For more details, refer to Viewing Results of Simulations Run using OCEAN Scripts.

Modifying an OCEAN Script in ADE Assembler

To modify an OCEAN script, open it in a text editor, make the required modifications, and rerun simulations.

You can modify the setup using the ADE Assembler user interface (such as adding or modifying global variables or corners using the Data View assistant pane), or use the OCEAN Test Editor form to modify the setup.

To use the OCEAN Test Editor form, do the following:

© 2023 Cadence Design Systems, Inc.

a. Right-click a test name on the *Tests* tree in the Data tab of Data View assistant pane and choose *Open Test Editor*.

The OCEAN Test Editor form appears.

The OCEAN Test Editor form is a text editor that displays the OCEAN commands saved in a script.

b. Change the simulation setup by modifying the OCEAN commands in the OCEAN Test Editor form.

c. Click *OK* to save the changes in the setup.

© 2023 Cadence Design Systems, Inc.

Important

When you modify the setup in the history item for the OCEAN run in ADE Assembler, the original OCEAN script that you used to run simulation from a UNIX shell will not be modified. Choose *File – Save Script* if you want to save the changes in the setup to a new OCEAN script file.

Running an OCEAN Script

An OCEAN XL script can be created or saved for the ADE Assembler setup as described in the previous section. You can run the script from the command line and process the results as needed.

Note: An OCEAN script runs in the background and often on a different machine. Therefore, it is recommended not to use interactive commands, such as `plot`.

To run simulations using OCEAN scripts, perform any one of the following steps:

- In CIW, run the following command:

  ```
  load("<name-of-OCEAN-script-file>")
  ```

- In ADE Assembler user interface, load the set up from history of a previous OCEAN run and click *Run Simulation* on the Run toolbar to start the simulation.

- In a UNIX shell, type the following commands:

  ```
  ocean
  load("<name-of-OCEAN-script-file>")
  OR
  ocean -replay <name-of-OCEAN-script-file>
  ```

  For information about running an OCEAN script from a UNIX shell, see the *OCEAN Reference*.

When you run a simulation using an OCEAN script, a history item named `Ocean.`*`n`* is created for the run. When finished, the results can be viewed in the ADE Assembler user interface.

Important

While a cellview is opened in ADE Assembler, you cannot load and run an OCEAN script for the same cellview.

When you run the script, the program reports the following information:

- Sweep parameters and their values

© 2023 Cadence Design Systems, Inc.

- Number of tests, sweep points, and corners
- Points completed and job status information
- Results location to the output area of the CIW

  For example:

```
1/1 completed.
*Info*   The result of this OCEAN XL run are saved in "Interactive.3" in library
"rfExamples", cell "ne600", view "adexl".
```

  The results location corresponds to the lib/cell/view specified in the ocnxlTargetCellView call, such as

```
ocnxlTargetCellView( "rfExamples" "ne600" "adexl")
```

See *OCEAN XL Commands Reference* for information about OCEAN script commands for ADE Assembler.

Tip

You can run an OCEAN script using LSF, for example, by submitting the job remotely using the `bsub` command (for the LSF example) as follows:

```
bsub ocean -nograph OCEANscriptFileName
```

Also see: includeSimLogInJobLog, an environment that can be used to control the inclusion of the simulator output log in the OCEAN job log.

Running Parallel OCEAN XL Simulation Runs for a maestro View

In OCEAN XL, you can simultaneously run multiple simulation runs for an maestro view. You can do this by executing multiple ocnxlRun functions in parallel from the same OCEAN XL script. The results of all the runs are saved in the results database of that maestro view.

This section describes how to prepare setup and run parallel simulation runs in OCEAN XL.

Preparing Setup for Parallel Simulation Runs

You can enable the parallel run option for your OCEAN XL scripts in any of the following ways:

- To enable the parallel run, before saving an OCEAN XL script from an maestro view, select the *Parallel* option in the **Run Options** form.

  See also:

  - Setting Up Run Options

© 2023 Cadence Design Systems, Inc.

- Use the ocnxlSetRunDistributeOptions command in the OCEAN XL script to set the parallel run option.

Running Parallel Simulation Runs

In your OCEAN script, ensure that the *waitUntilDone* argument of the ocnxlRun function is set to `nil`.

For example, if you have saved an ADE Assembler state, `Ac_State1`, and want to run a simulation for it parallel to other runs in your OCEAN XL script, use the following commands:

```
ocnxlLoadSetupState( "AC_State1" 'retain ?tests t ?vars t ?parameters t
?currentMode t ?runOptions t ?specs t ?corners t ?extensions t
    ?modelGroups nil ?relxanalysis nil )

runid1 = ocnxlRun(?waitUntilDone nil)

ocnxlLoadSetupState( "Tran_State2" 'retain ?tests t ?vars t ?parameters t
?currentMode t
    ?runOptions t ?specs t ?corners t ?extensions t
    ?modelGroups nil ?relxanalysis nil )

runid2 = ocnxlRun(?waitUntilDone nil)
```

When the *waitUntilDone* argument is set to `nil`, OCEAN XL does not wait for the run to complete. Instead, while the run is in progress, it executes the next line in the script. In the example given above, OCEAN XL loads another state, `Tran_State2`, and starts a simulation for that without waiting for the first simulation run to finish.

Note: You need to set the *waitUntilDone* argument for each OCEAN XL run that you want to run in parallel. By default, this argument is set to `t`.

When you set the *waitUntilDone* argument to `nil`, the ocnxlRun function also returns a run ID for each run. You can use this run ID later to either wait for that run to complete at a later stage or to access the history results of that run.

For example, before printing results, if you want to wait for a particular OCEAN XL run to complete simulations, you can use the following statement in your script:

```
(ocnxlWaitUntilDone rinid2)

(ocnxlOutputSummary ?forRun runid1 ?detailed nil)

; The previous command displays run summary for the simulation run with run ID as
runid1
```

After a run is complete, you can get access to its results by using its run ID, as shown in the following statement:

```
h1 = ocnxlGetHistory(runid1)
```

You can use this handle to get results for the given OCEAN XL run.

© 2023 Cadence Design Systems, Inc.

For details on the related OCEAN XL functions and their examples, refer to the following sections in the OCEAN Reference Guide:

- ocnxlRun
- ocnxlOutputSummary
- ocnxlWaitUntilDone
- ocnxlGetHistory
- ocnxlSetRunDistributeOptions
- ocnxlGetRunDistributeOptions

Viewing Results of Simulations Run using OCEAN Scripts

The results of simulations run from OCEAN scripts are saved in the results database of the maestro view. While the OCEAN XL simulations are in progress or after they are complete, you can view these results in the ADE Assembler user interface.

To view the results for an OCEAN simulation run in the ADE Assembler user interface, do the following:

➥ In the History tab of the Data View assistant pane, right-click the history item for the OCEAN run and choose *View Results*.

Note: While the simulations are in progress, you need to close and reopen the results to view the updated data.

When you view the results of simulations performed by specifying netlist as design, you cannot use the schematic-based post processing options. For example, you cannot use the `VT` calculator function to select a net on schematic or you cannot use the direct plotting feature. Therefore, when you right-click the data in the *Results* tab, some of the commands that require use of the schematic view are not enabled.

© 2023 Cadence Design Systems, Inc.

18

Debugging with ADE Diagnostics

While debugging simulations, you need information about the points with erroneous results or the entire history saved for the run. In LSCS job control mode, the tool saves multiple log files that are saved by different components of the LSCS architecture. ADE Assembler runs a list of checks on these log files and generates the ADE Diagnostics report that helps you troubleshoot simulation errors.

The ADE Diagnostics report displays accurate results only when:

- the specified job control mode is LSCS
- the specified simulator is Spectre
- the run mode is set to Single Run, Sweeps and Corners or Monte Carlo Sampling

This report can be generated in other cases as well but the results might not be reliable.

Note: The ADE diagnostics report is meant for advanced users and CAD engineers. This report provides a root-cause analysis of the issues reported during a simulation. You can use this information to identify solutions while troubleshooting simulations.

In certain scenarios, you may review the resources being used and take some predefined steps, if necessary. The Debug Utility form provides some additional checks and debugging options that you can use to save useful diagnostic information.

Related Topics

Using the Debug Utility Form

ADE Diagnostics Report

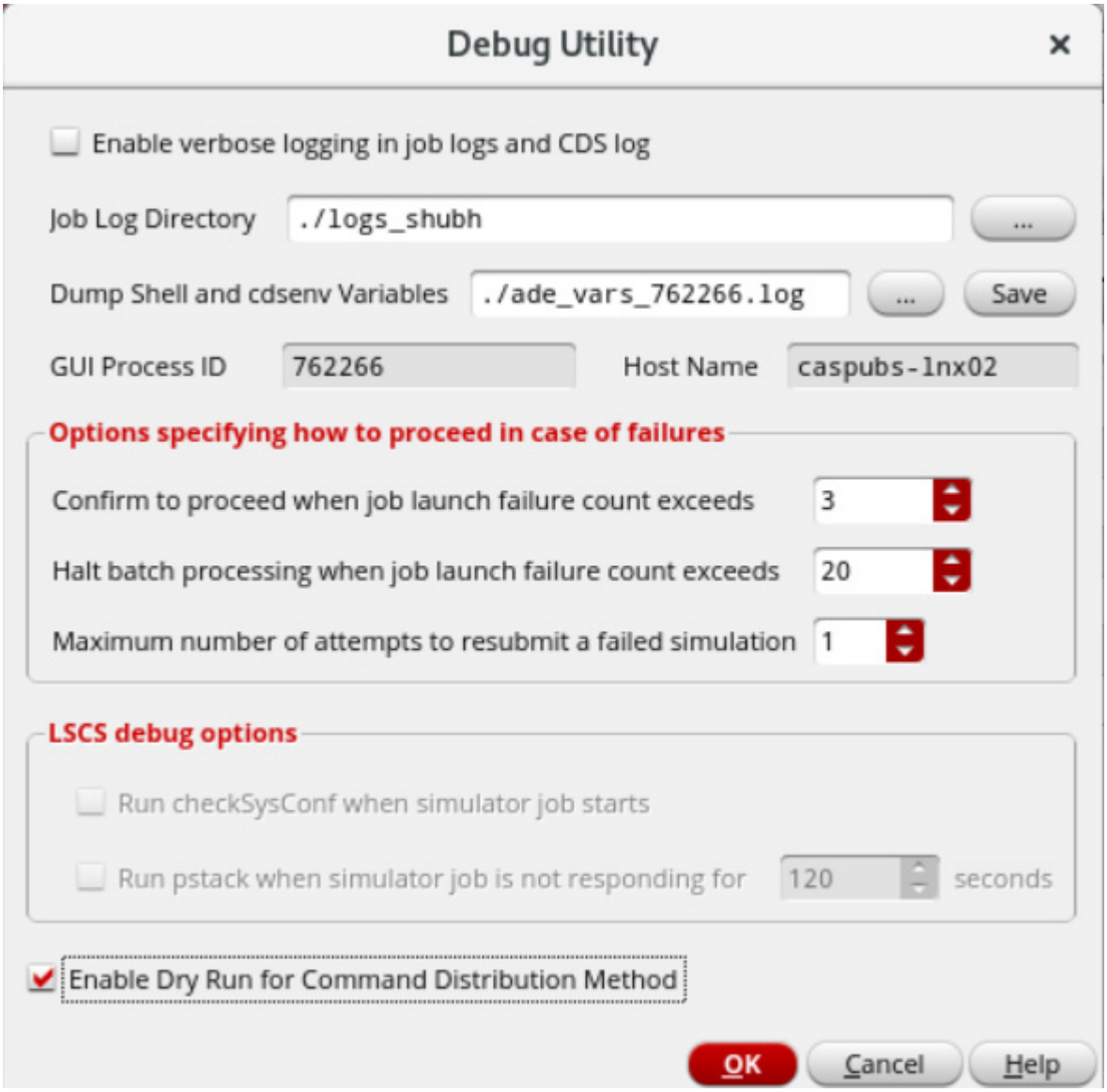
Using the Debug Utility Form

The Debug Utility form provides debugging options that help you gather information to o analyze when troubleshooting simulation runs.

© 2023 Cadence Design Systems, Inc.

To use the Debug Utility form:

1. Choose *Tools – Debug* in ADE Assembler.



   The Debug Utility form is displayed.

2. Review and modify the options given in this form.

   By default, the log files show basic error, warning, and information messages. If the information available in the log files is not sufficient and you need more details, select the *Enable verbose logging in job logs and CDS log* check box. This saves additional details and displays more detailed messages. However, it impacts the performance of the tool and increases the size of the log files significantly.

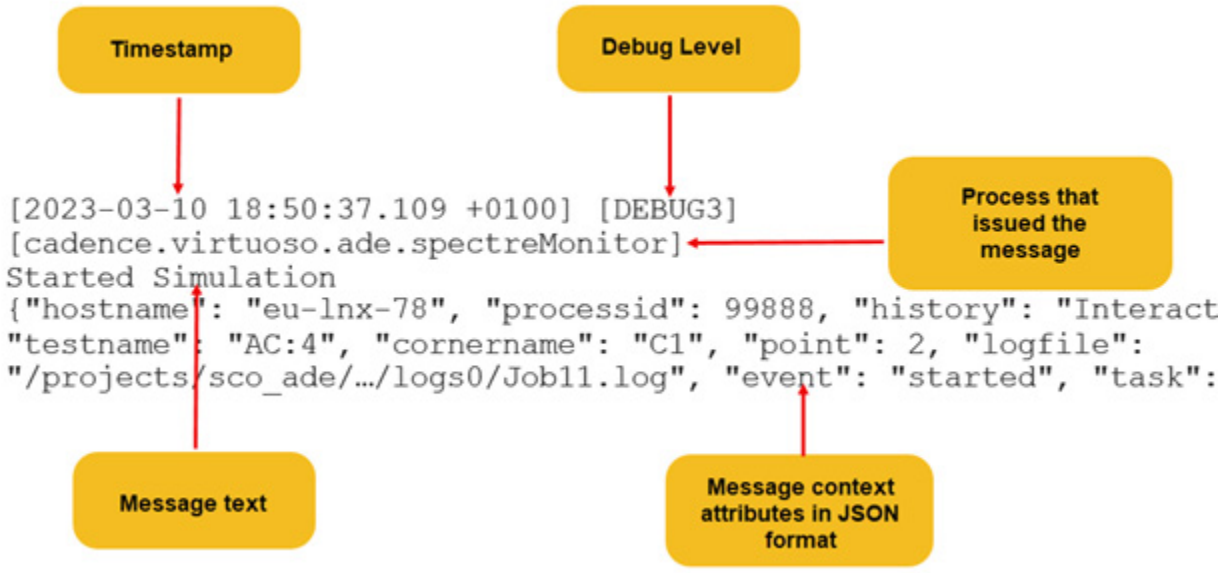
© 2023 Cadence Design Systems, Inc.

It is recommended to selectively enable the logs using the following environment variables:

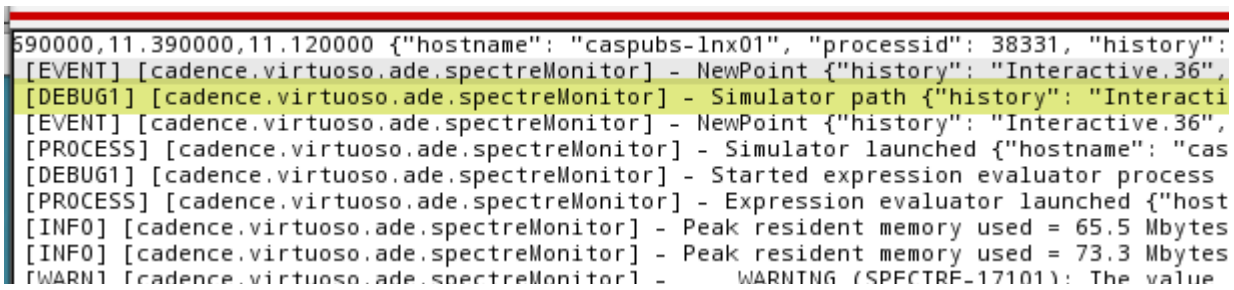
- `maestro.debug netlisting`
- `maestro.debug simAndEval`
- `maestro.debug beanstalk`
- `maestro.debug mainVirtuoso`

Through these variables, you can specify a higher debug level in which the tool saves additional details and displays more detailed messages. These messages are added to the log files in a machine-readable format which makes it easy to filter and parse the information.

The following figure shows the structure of these messages.

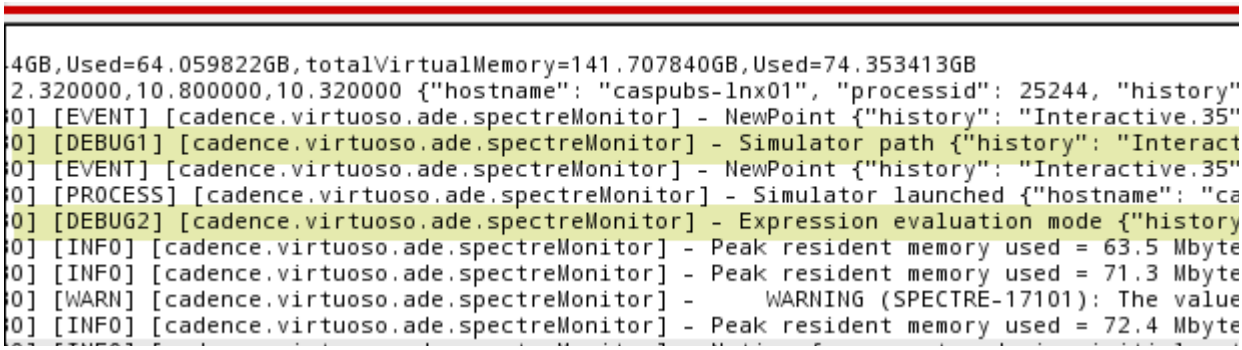


For example, if you set `maestro.debug simAndEval` to `1`, you can see extra lines marked with [DEBUG1] saved in the log, as shown below.



© 2023 Cadence Design Systems, Inc.

Similarly, at level `2`, the tool lets you print user-level debugging information for an error. This also includes status change messages for processes, as shown below.



At level `3`, the beanstalk monitor is enabled and the tool prints detailed debugging messages suitable for advanced users.

In level `2` and `3`, there might be some impact on the performance of the tool. Therefore, it is recommended to reset these variables to `0` after you have finished debugging simulations.

3. Click *OK*.

Related Topics

Debug Utility Form

maestro.debug netlisting

maestro.debug simAndEval

maestro.debug beanstalk

maestro.debug mainVirtuoso

© 2023 Cadence Design Systems, Inc.

ADE Diagnostics Report

Troubleshooting simulation errors using log files can be a difficult task because ADE Assembler saves multiple log files that are generated by different components of the LSCS architecture.

For detailed analysis of such errors and debugging, ADE Assembler runs a series of checks on these log files and displays their results in an HTML report.

This report has three panes:

- Check Results
- Details
- Point Mapping

Check Results

Th *Check Results* pane displays the status of the checks and the total number of simulation-related errors in tabular format. This table has four columns.

Column Name	Description
id	Displays the check ID. ■ *R*: Run Information ■ *T*: Timing Information ■ *D*: Design Nelisting ■ *P*: Point Netlisting ■ *S*: Simulation ■ *E*: Evaluation ■ *O*: Overlap ■ *B*: Beanstalk
Description	Displays the name and details of the checks being run.
Passed	Reports the status of the performed checks.
Problems	Reports the number of issues found while performing each check.

© 2023 Cadence Design Systems, Inc.

Details

The *Details* pane provides more details about the checks. To get more information for a specific check, select the related row in the *Check Results* table and view the *Details* pane. This information is displayed in a tabular format and may include references to other checks that might have caused the given check to fail. This table has two columns.

Column Name	Description
Field	Specifies the type of information being displayed for the failed check. For example, `test` and `point`.
Value	Displays a short description of the failed check, followed by the related information required for debugging. This includes the name of the associated test, the point where an issue was found, and any related messages. It may also include references to other checks.

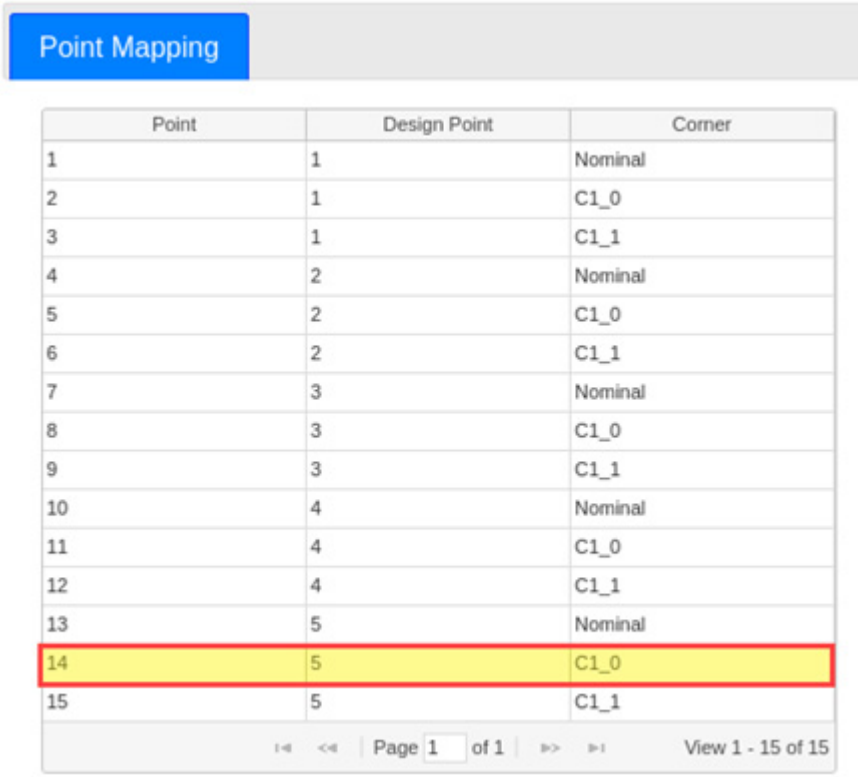
Point Mapping

The *Point Mapping* pane lets you map the points reported in the *Details* pane to the points in the *Results* tab in ADE Assembler. This information is displayed in tabular format and lets you easily identify the source of the issue. This table has three columns.

Column Name	Description
Point	Displays the serial number.
Design point	Specifies the design point in the *Results* tab where the issue was found.
Corner	Specifies the corner in the *Results* tab where the issue was found.

© 2023 Cadence Design Systems, Inc.

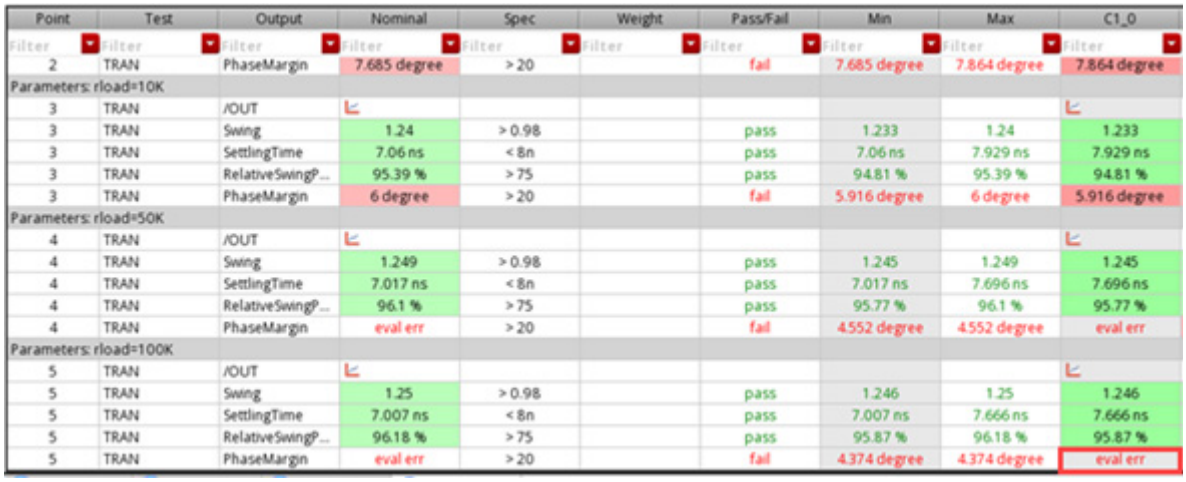
For example, if an evaluation error was found in point `14`, the following information is displayed in the *Point Mapping* pane.



Point	Design Point	Corner
1	1	Nominal
2	1	C1_0
3	1	C1_1
4	2	Nominal
5	2	C1_0
6	2	C1_1
7	3	Nominal
8	3	C1_0
9	3	C1_1
10	4	Nominal
11	4	C1_0
12	4	C1_1
13	5	Nominal
14	5	C1_0
15	5	C1_1

Page 1 of 1 View 1 - 15 of 15

This means that the issue was found at design point `5`, for corner `C1_0` in the *Results* tab.



Point	Test	Output	Nominal	Spec	Weight	Pass/Fail	Min	Max	C1_0
2	TRAN	PhaseMargin	7.685 degree	> 20		fail	7.685 degree	7.864 degree	7.864 degree
Parameters: rload=10K									
3	TRAN	/OUT							
3	TRAN	Swing	1.24	> 0.98		pass	1.233	1.24	1.233
3	TRAN	SettlingTime	7.06 ns	< 8n		pass	7.06 ns	7.929 ns	7.929 ns
3	TRAN	RelativeSwingP...	95.39 %	> 75		pass	94.81 %	95.39 %	94.81 %
3	TRAN	PhaseMargin	6 degree	> 20		fail	5.916 degree	6 degree	5.916 degree
Parameters: rload=50K									
4	TRAN	/OUT							
4	TRAN	Swing	1.249	> 0.98		pass	1.245	1.249	1.245
4	TRAN	SettlingTime	7.017 ns	< 8n		pass	7.017 ns	7.696 ns	7.696 ns
4	TRAN	RelativeSwingP...	96.1 %	> 75		pass	95.77 %	96.1 %	95.77 %
4	TRAN	PhaseMargin	eval err	> 20		fail	4.552 degree	4.552 degree	eval err
Parameters: rload=100K									
5	TRAN	/OUT							
5	TRAN	Swing	1.25	> 0.98		pass	1.246	1.25	1.246
5	TRAN	SettlingTime	7.007 ns	< 8n		pass	7.007 ns	7.666 ns	7.666 ns
5	TRAN	RelativeSwingP...	96.18 %	> 75		pass	95.87 %	96.18 %	95.87 %
5	TRAN	PhaseMargin	eval err	> 20		fail	4.374 degree	4.374 degree	eval err

Related Topics

Using the Debug Utility Form

© 2023 Cadence Design Systems, Inc.

Checks Performed by ADE Diagnostics

Generating the ADE Diagnostics Report

Checks Performed by ADE Diagnostics

The following list gives an overview of the checks performed by ADE Diagnostics. The diagnostics report provides a comprehensive description for these checks.

- `Run Information`: Reports the expected number of simulations to be run for a test. Also, lists all the expected config sweep combinations and checks whether the netlist complies with them.
- `Timing Information`: Reports the maximum, minimum, and average run time for design netlisting, point netlisting, evaluation, and simulation.
- `Design Netlisting`: Checks whether the design netlisting was successful and reports any errors that occurred during netlisting.
- `Point Netlisting`: Checks whether the point netlisting was successful and reports any errors that occurred during netlisting. Also, checks whether the SNAP netlist was created before the point netlist and if the links in the netlist point to valid SNAP files.
- `Simulation`: Checks whether the simulation was successful and if it was run only once. Also, reports any simulation errors that were encountered, along with any unusual, empty, or missing files that were found in the PSF directory.
- `Evaluation`: Checks whether the evaluation was successful and if it was run only once. Also reports any evaluation errors that were encountered.
- `Overlap`: Checks whether simulation had started before the netlisting was finished.

Related Topics

Using the Debug Utility Form

ADE Diagnostics Report

Generating the ADE Diagnostics Report

Generating the ADE Diagnostics Report

To view the diagnostics report:

© 2023 Cadence Design Systems, Inc.

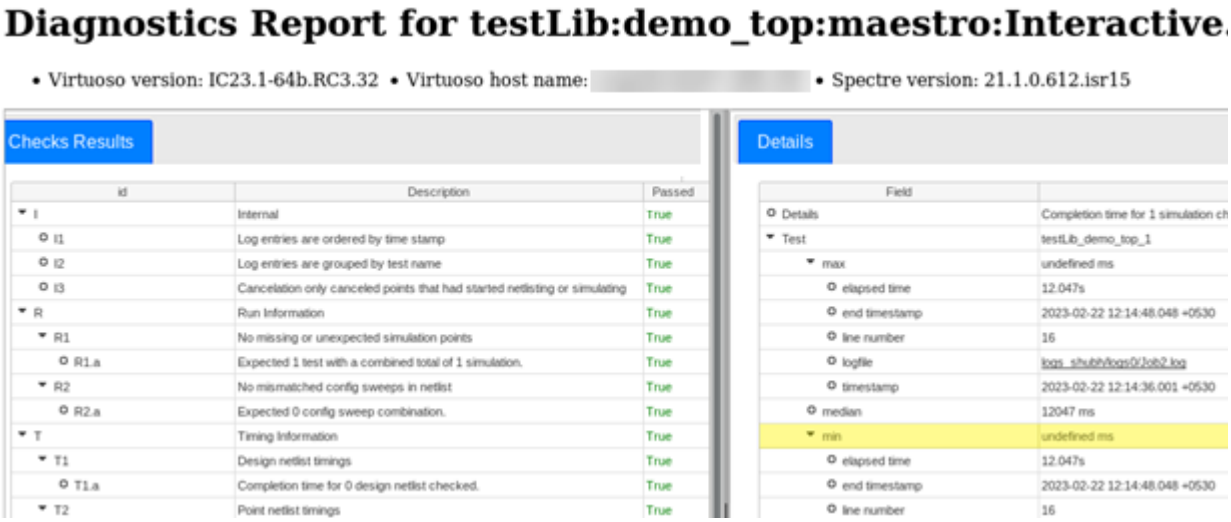
1. Before running a simulation, set the following environment variables in the CIW.

```
envSetVal("maestro.debug" "netlisting" 'int 3)
envSetVal("maestro.debug" "simAndEval" 'int 3)
envSetVal("maestro.debug" "beanstalk" 'int 3)
envSetVal("maestro.debug" "mainVirtuoso" 'int 3)
```

   Alternatively, select the *Enable verbose logging in job logs and CDS log* check box in the Debug Utility form.

2. In the *History* tab of the Data View assistant, right-click the history for which you want to generate the diagnostics report, and choose *Diagnostics – Analyze Logs*.

   An HTML report with the title `Log Diagnostics Report` is generated. The subtitle of this report is in the format *`LibName:CellName:ViewName (historyName)`*. Here, *`historyName`* represents the history being used. *`LibName`*, *`CellName`*, and *`ViewName`* represent the names of the library, cell, and view for the test associated to that history. The report also lists the Virtuoso and Spectre versions being used along with the Virtuoso host name.



Related Topics

ADE Diagnostics Report

Checks Performed by ADE Diagnostics

maestro.debug netlisting

maestro.debug simAndEval

maestro.debug beanstalk

© 2023 Cadence Design Systems, Inc.

maestro.debug mainVirtuoso

© 2023 Cadence Design Systems, Inc.

19

Viewing, Printing, and Plotting Results

ADE Assembler displays simulations results in a tabular format on the Results tab of the Outputs pane. In the results table, output values for tests and corners are displayed in rows and corners, respectively. A standard set of columns display the maximum and minimum value for each test and output combination, their specifications, and their pass or fail status. If, due to some reason, any output value is not calculated, an appropriate indication of an error is displayed. The Results tab also contains a toolbar that provides various commands to customize the view of results.

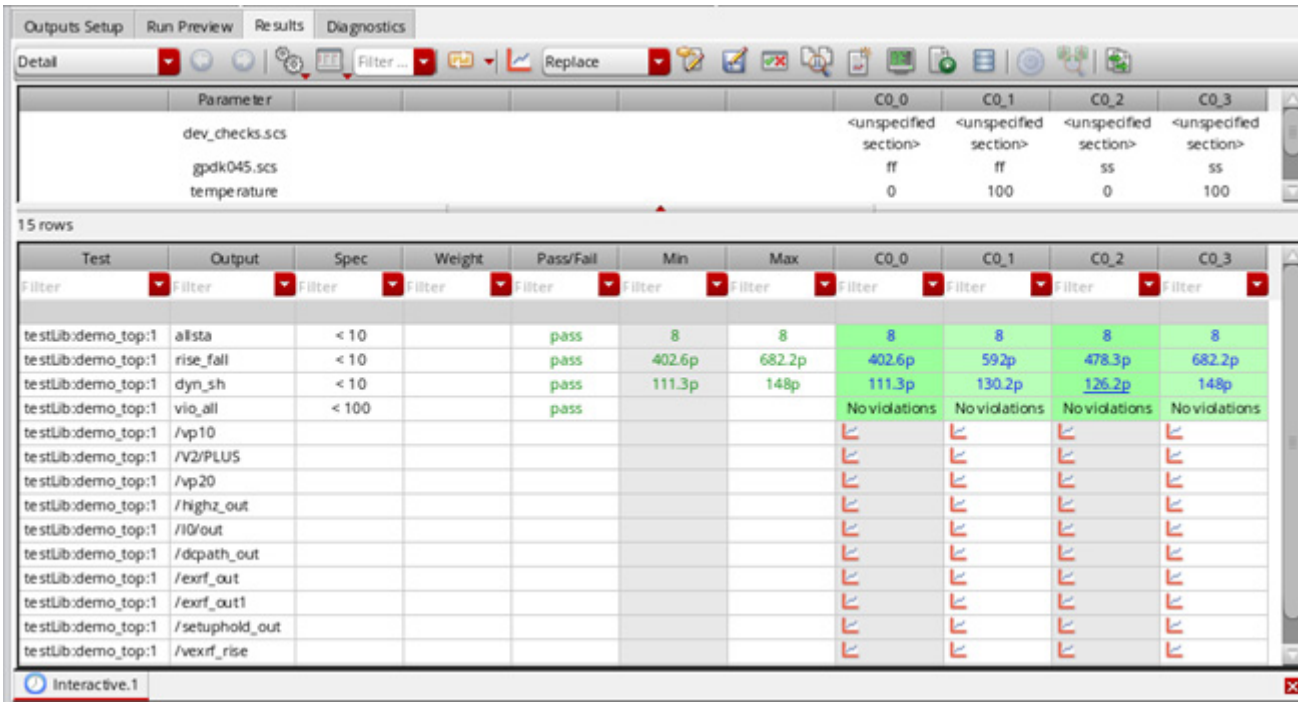
The following sections of this chapter provide more details about how to view, print, or plot the results displayed on the Results table:

- Viewing Data on the Results Tab
- Filtering Results on the Results Tab
- Plotting Results
- Printing Results
- Backannotating Data from Results to Schematic
- Viewing Waveform Info-Balloons on Schematic
- Re-Evaluation of Expressions and Specifications
- Viewing the Spec Summary
- Comparing Results Using the Spec Comparison Form
- Managing Results
- Using SKILL to Display Tabular Data
- Showing Variable and Parameter Values for Design Points on the Results Tab
- Searching for Conditional Results

© 2023 Cadence Design Systems, Inc.

Viewing Data on the Results Tab

You can view simulation results on the Results tab of the Outputs pane in the ADE Assembler environment.



Results appear on a new tab in the *Results* tab of the Outputs pane after you run a successful simulation, or view or restore results from a previous checkpoint on the History tab of the Data View assistant pane. For more information about working with tabs, see Working with Tabs for Simulation Checkpoints. For more information about how results are displayed in the Results tab, see Viewing Specification Results in the Results Tab.

By default, all columns and types of information appear on the Results tab. You can show and hide various columns and types of information using the button. Corners conditions appear in a table along the top portion of the tab. You can switch the view using the *Select the results view* drop-down list. You can export results to HTML or a comma-separated values file.

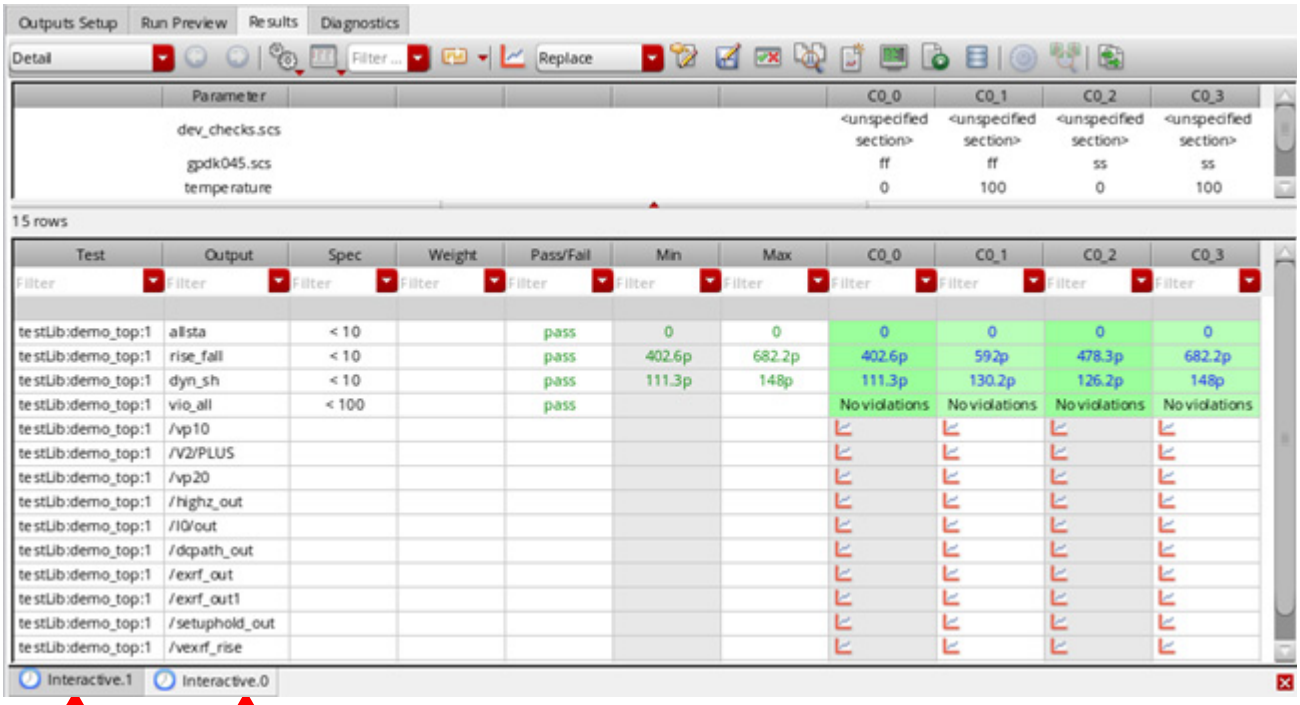
Working with Tabs for Simulation Checkpoints

When you run a simulation, or view or restore results from a previous checkpoint on the History tab of the Data View assistant pane, the simulation results appear on a new tab in the Results tab of the Outputs pane. The name of the tab matches the name of the checkpoint

© 2023 Cadence Design Systems, Inc.

for the simulation run. If you view or restore results from a previous checkpoint, the name of the tab matches the name of that checkpoint.

For example, in the following figure, the *Interactive.0* tab displays the results for the `Interactive.0` checkpoint. To view the results for the `Interactive.1` checkpoint, click the *Interactive.1* tab.



Results tab for Interactive.1 checkpoint

Results tab for Interactive.0 checkpoint

The icons that are displayed in the tab name when you run a simulation are described below:

Icon	Description
	Simulation run for `Interactive.16` checkpoint is started.
	Simulation run for `Interactive.16` checkpoint is in progress. The progress of the simulation run is displayed in the form of a pie chart.

© 2023 Cadence Design Systems, Inc.

Icon	Description
	Simulation run for `Interactive.16` checkpoint is complete.

To close the currently active tab, do the following:

➡ Click the icon on the bottom right of the Results tab.

To close a specific tab, do the following:

➤ Right-click the tab name and choose *Close Tab*.

To close all tabs other than the current one, do the following:

➤ Right-click the tab name and choose *Close Other Tabs*.

To close all tabs, do the following:

➤ Right-click the tab name and choose *Close All Tabs*.

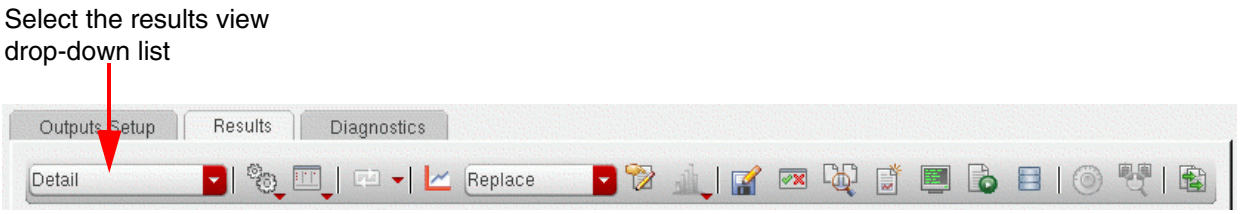
Results Views

By default, the columns listed in Table 19-1 on page 1088 are displayed in the Results tab of the Outputs pane. This view is called the *Detail* view. You can use the *Select the results view* drop-down list to switch between the following views:

- Detail
- Detail - Transpose
- Detail - Transpose with Test
- Optimization
- Status
- Summary
- Yield
- Checks/Asserts
- Fault

© 2023 Cadence Design Systems, Inc.

- Capacitance



Every run mode has a default result view mapped to it. For example, the default result view for the Monte Carlo Sampling run mode is the Yield view.

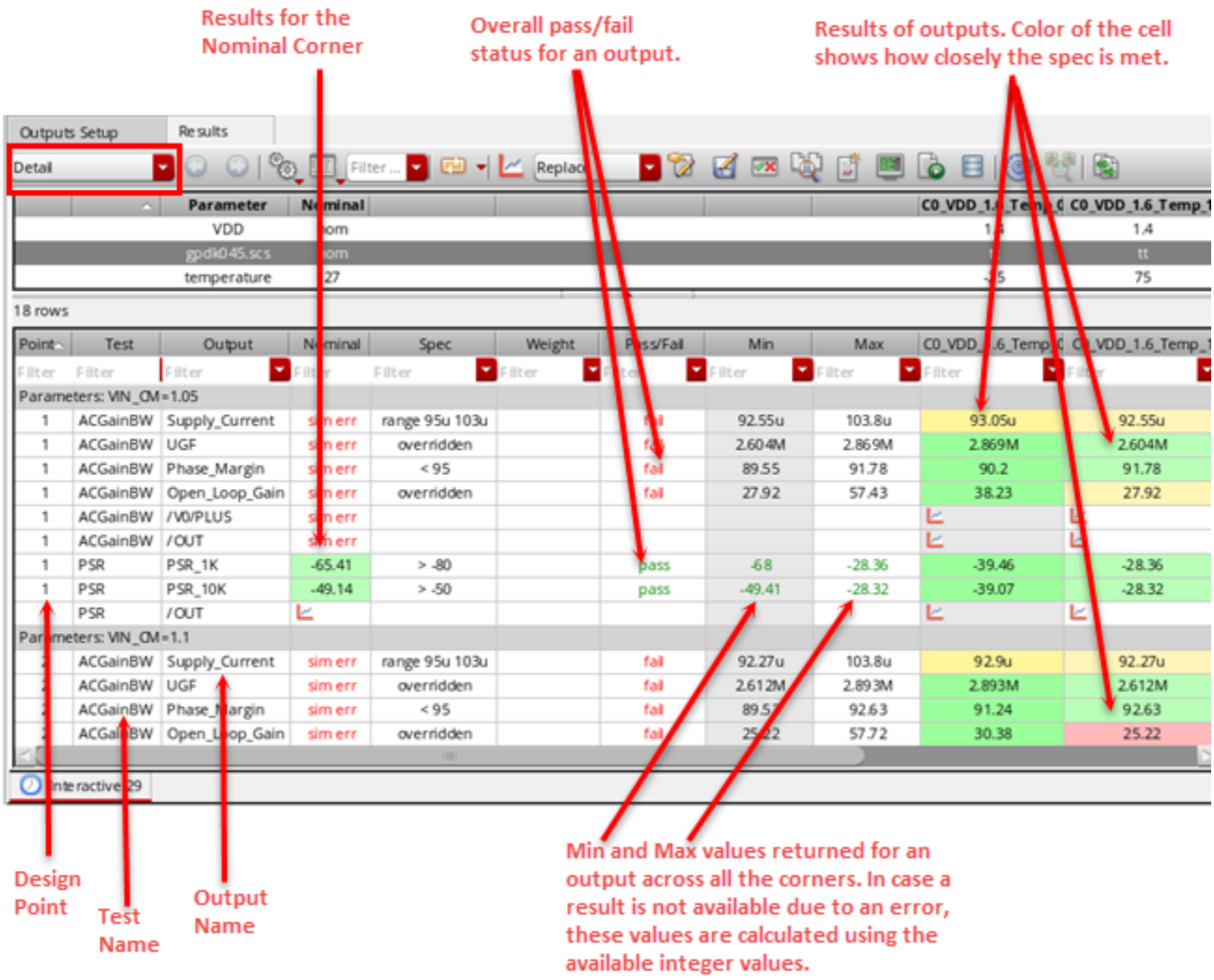
Note: You can change the default results view for the Single Run, Sweeps and Corners run mode by using the defaultResultsViewForSweepsCorners environment variable. This feature is currently not available for other run modes.

Detail

Displays the detailed results of all the design points and corners.

© 2023 Cadence Design Systems, Inc.

Figure 19-1 Detail View on the Results Tab

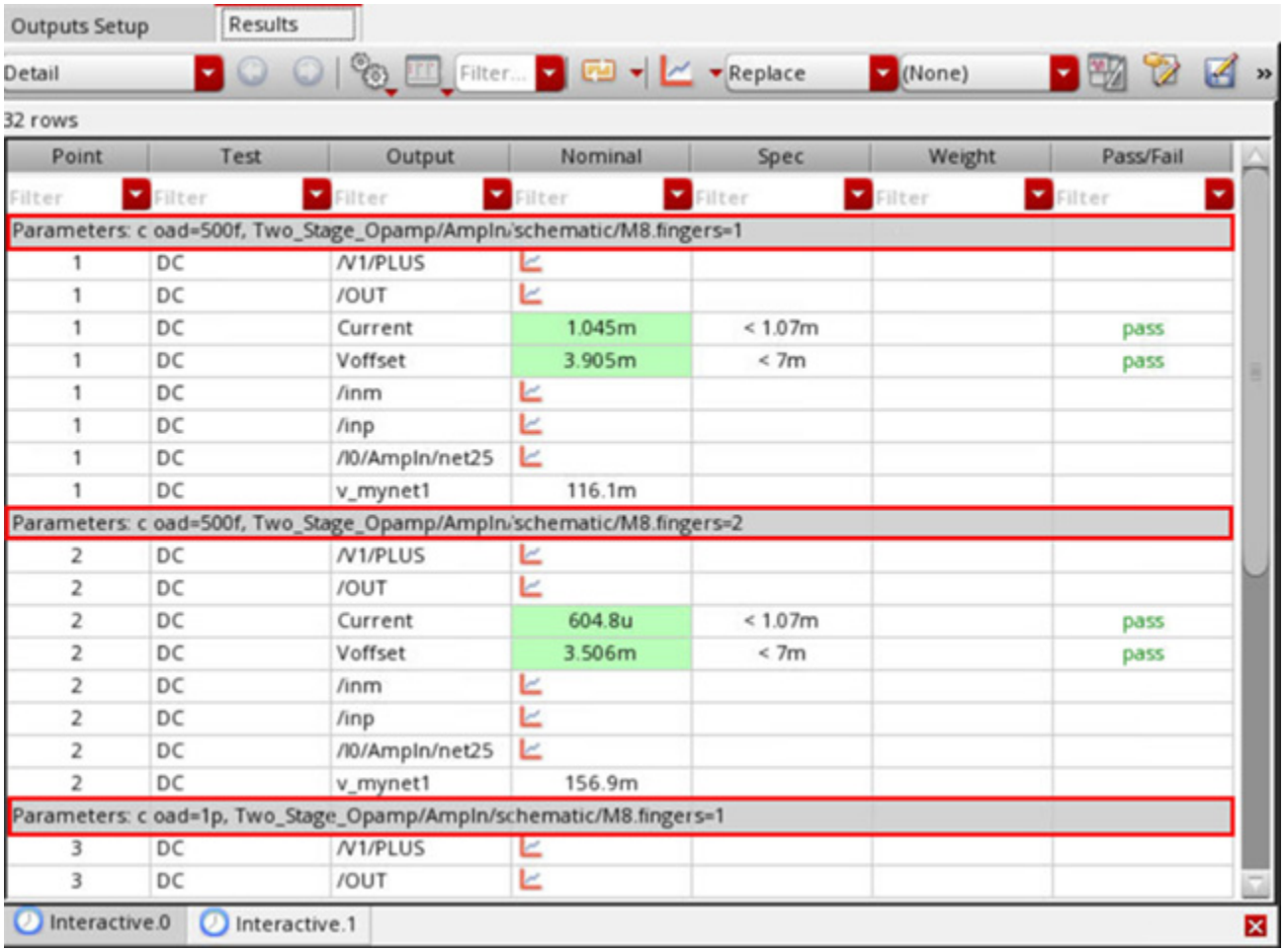


For more information about the columns in the *Detail* view, see Table 19-1 on page 1088.

When multiple instance parameters with the same instance name occur at different levels of hierarchy, the instance parameter path is displayed in a gray header row for each result point.

© 2023 Cadence Design Systems, Inc.

This row contains a prefix, Parameters. This helps in uniquely identifying the parameters when multiple instances have parameters with the same names.



Related Topics

hideStringResultsFailure

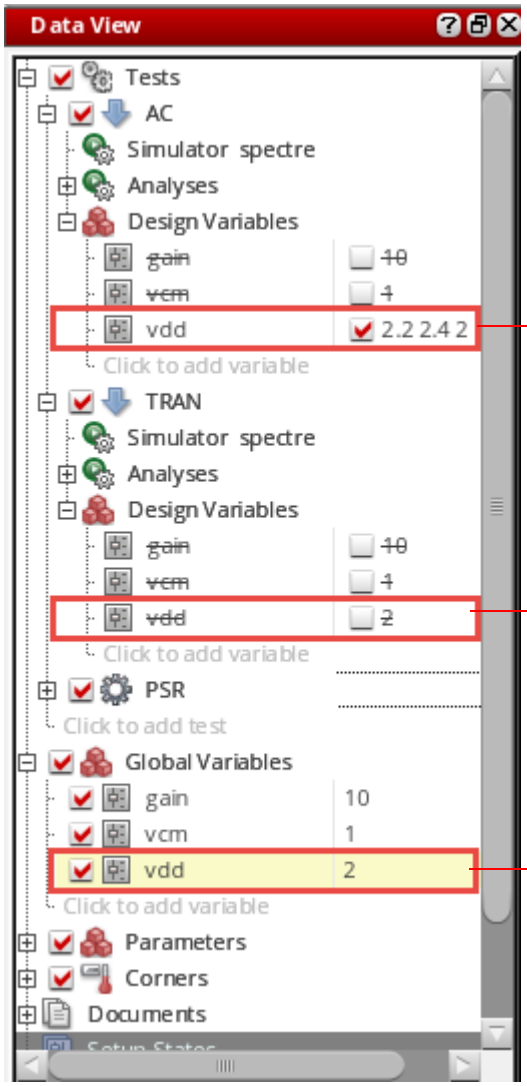
Viewing Simulation Results When Design Variables Use Swept Values

When a global variable is disabled for a test that uses sweep values, the value in the corresponding global variable is ignored. ADE Assembler creates separate sweep points for that test on the *Results* tab.

Consider the following example where the ADE Assembler setup contains three tests: `AC`, `TRAN` and `PSR`. The `vdd` global variable has its value set to `2V`, and it is disabled for the test

© 2023 Cadence Design Systems, Inc.

AC, which sweeps the value of vdd to 2.0V, 2.2V and 2.4V. The *Data View* assistant given below shows how the variables are set.



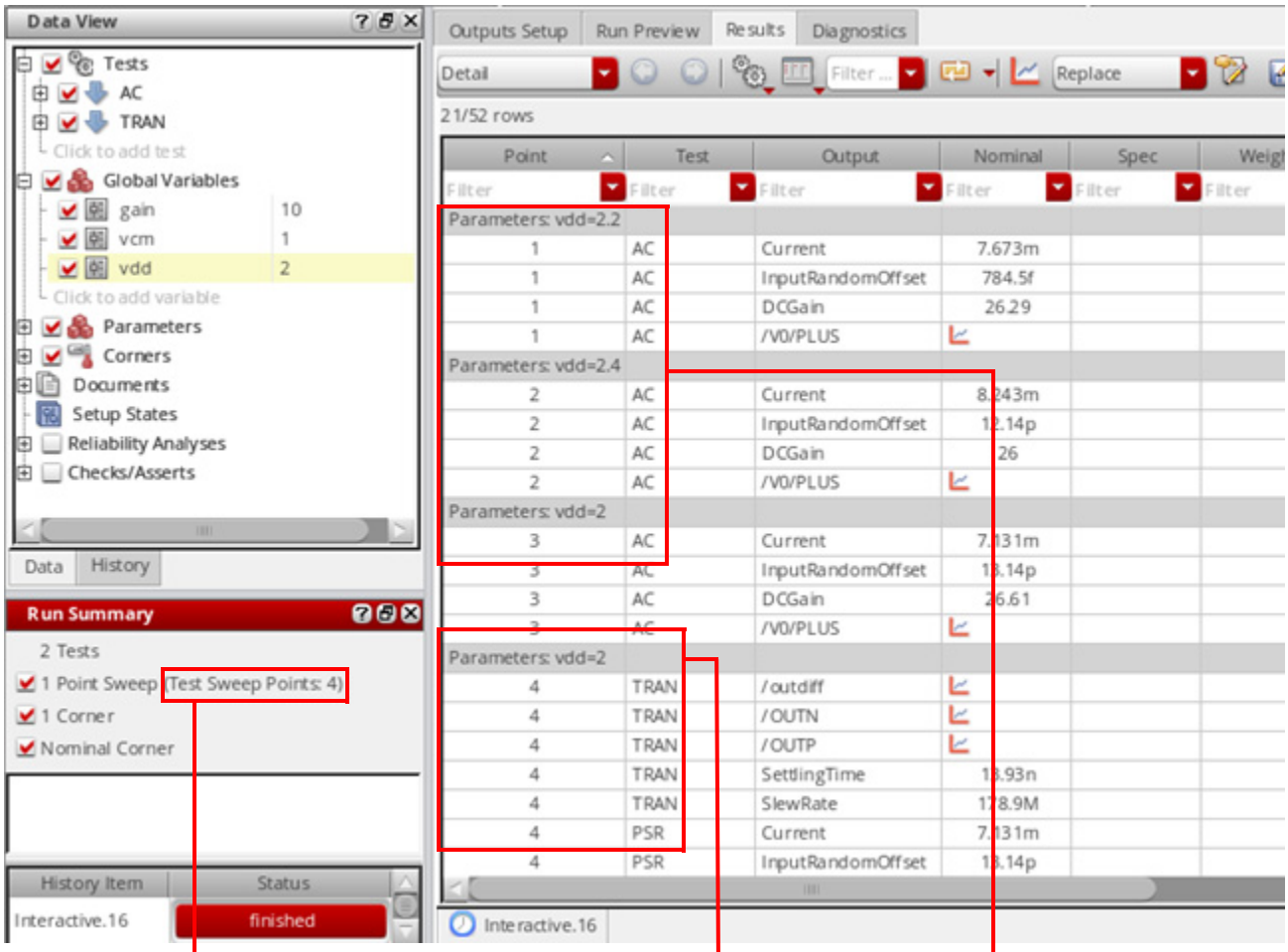
Independent **local sweep values** for test AC

Test TRAN is **using the same values** as set for the vdd **global variable**

Value set for the vdd global variable

© 2023 Cadence Design Systems, Inc.

In this case, the results are displayed as shown in the figure below.



Total number of sweep points from all the tests in the setup.

Common sweep point for tests that use the value from global variable. Here, `TRAN` and `PSR`.

Independent sweep points for the test that uses local sweep values.

Note the following points for the above example:

- The *Test Sweep Points* count on the Run Summary pane shows the number of sweep points specified for the design variables for tests in the setup. Test `AC` uses three sweep points for `vdd`. Tests `TRAN` and `PSR` use the single value from the global variable `vdd`. Therefore, the total test sweep count is `4`. Each sweep point has a separate point ID on the Results tab.
- All the results are displayed together on the Results tab.

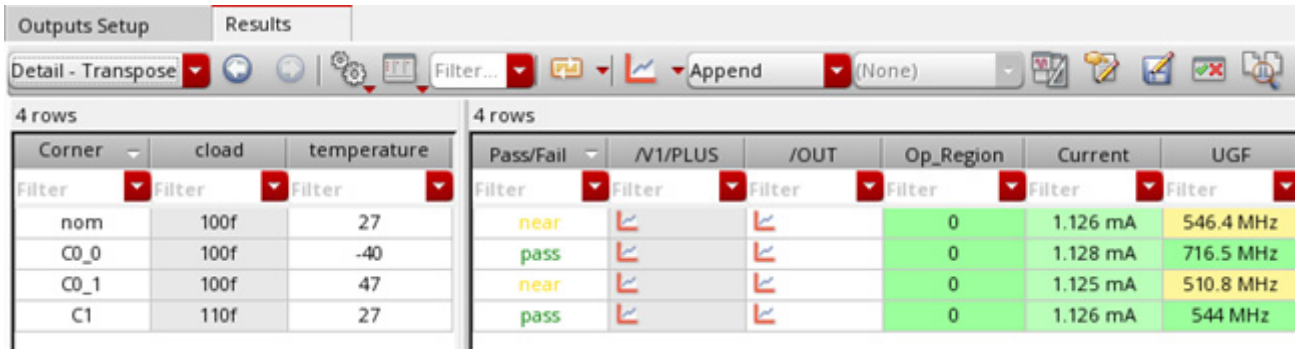
© 2023 Cadence Design Systems, Inc.

Detail - Transpose

The *Detail - Transpose* view allows visualization of results across corners. The table on the left side displays corner and sweep conditions for each simulation point, and the table on the right side displays the results and *pass*, *near* or *fail* status for output expressions at each simulation point.

Note: The *pass*, *near* or *fail* status displayed for output expressions in the *Pass/Fail* column is based on the status of all specifications across all tests. For more information about the *pass*, *near* or *fail* status for output expressions, see Viewing Specification Results in the Results Tab.

Figure 19-2 Detail - Transpose View in Results Tab



For more information about the *Detail - Transpose* view, see Customizing the Detail - Transpose View.

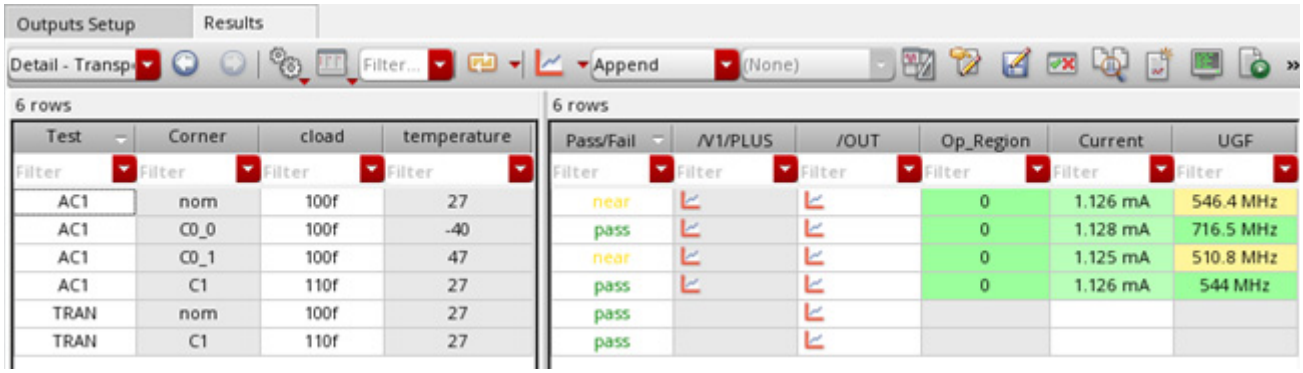
Detail - Transpose with Test

Note: This view is visible only when the detailTransposeViewShowTest environment variable is set to `t`.

The *Detail - Transpose with Test* view is similar to the *Detail - Transpose* view. The table on the left side displays test name, corners, parameters, and sweep conditions for each simulation point. The table on the right side displays the results and *pass*, *near* or *fail* status for output expressions at each simulation point.

© 2023 Cadence Design Systems, Inc.

Figure 19-3 Detail - Transpose with Test View in Results Tab



If the results contain details of instance parameters, the *Detail - Transpose* or *Detail - Transpose with Test* result views show the instance path in the tooltips of parameter names. This helps in uniquely identifying the parameters when multiple instances have parameters with same names.

This view is particular useful when you need to export the results to Excel and create pivot reports based on test names. For more details, refer to Exporting Results.

Optimization

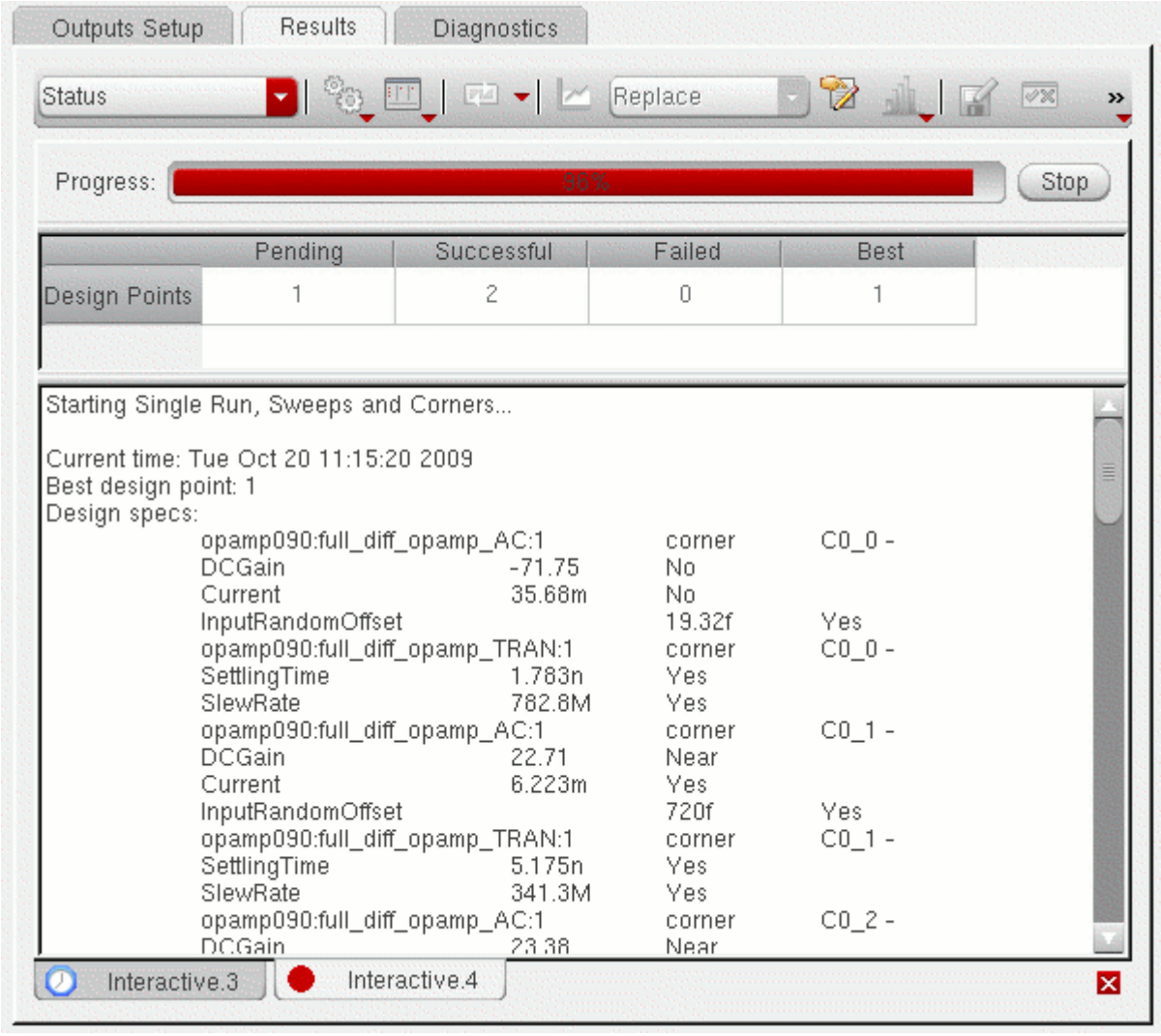
The Optimization view is displayed when you run Worst Case Corners, Global Optimization, or Local Optimization run modes. It displays the *Test*, *Output*, *Value*, *Spec*, *Weight*, *Min*, *Max* and corners-related columns.

Status

The Status view displays the progress and status of the simulation run, and the contents of the run log file. For more information about the Status view, see Using the Status View.

© 2023 Cadence Design Systems, Inc.

Figure 19-4 Status View in Results Tab

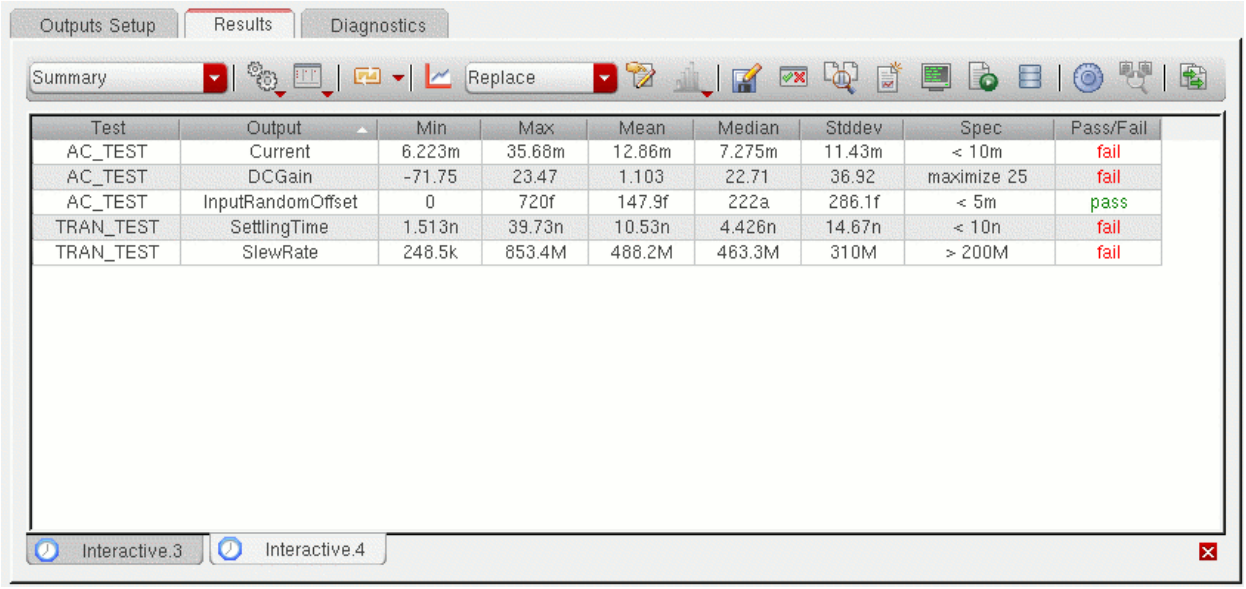


Summary

The Summary view displays the summary of the results for output expressions across all corners and sweeps.

© 2023 Cadence Design Systems, Inc.

Figure 19-5 Summary View in Results Tab



The columns in the Summary view are described below:

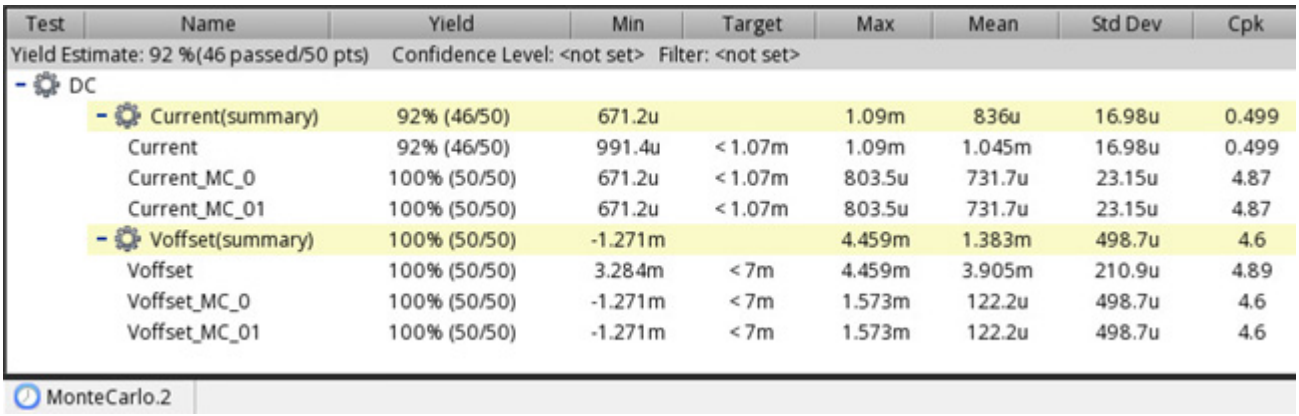
Column	Data/Content
Test	Name of the test for which the output is defined.
Output	Name of the output expression.
Min	Minimum value of the output.
Max	Maximum value of the output.
Mean	Mean value of the output.
Median	Median value of the output.
Std Dev	Standard deviation of the values for the specification. **Note:** For Single Run, Sweeps and Corners, this is the population standard deviation. For other run modes, this is the sample standard deviation.
Spec	Specification defined for the output expression.
Pass/Fail	The pass/fail status of the specification across all sweeps and corners.

© 2023 Cadence Design Systems, Inc.

Yield

The Monte Carlo Yield view displays for all specifications the overall yield estimate based on pass or fail status. The yield estimate for each specification and statistics, such as mean and standard deviation, are also reported. A sample Yield view is displayed in the following figure.

Figure 19-6 Yield View on Results Tab



The columns displayed in the *Yield* view are described in the table given below:

Column	Data/Content
Test	Test name, top of expandable tree of outputs
Name	Expression/measurement or corner name
Yield	Resulting yield for each specification. In addition, it shows number of points passed / number of points completed for each corner.
Min	Minimum value resulted for an output
Target	Target specification
Max	Maximum value resulted for an output
Mean -K Sigma	Mean -K Sigma value resulted for an output. The default value of K is `3`, but you can reset it. For this, right-click the column header and choose any one of the given sigma values. To use any other sigma value, choose *Set K-igma* to open the Set K-Sigma form and specify a value for K between `0` and `100000`.
Mean	Mean value resulted for an output

© 2023 Cadence Design Systems, Inc.

Column	Data/Content
Mean +K Sigma	Mean +K Sigma value resulted for an output. The default value of K is `3`, but you can reset it in the same way as described for the *Mean -K Sigma* column above.
Std Dev	Worst case value of the standard deviation as computed per corner
CV(%)	Coefficient of variation or relative standard deviation (in percentage)
Sigma To Target	Distance from the standard deviation value to the target value, defined as follows: `mean value to target value / standard deviation` **Note:** This column is hidden by default.
Cpk	Cpk is the process capability index, a statistical measure of the process capability. It is the capability of a process to produce an output within the given specification limits. If USL and LSL are the upper and lower specification limits, μ is the estimated mean of the process and σ is the estimated variability of the process, Cpk is calculated by using the following equation. $$Cpk = \min\left[\frac{USL-\mu}{3\sigma}, \frac{\mu-LSL}{3\sigma}\right]$$ Cpk is a good estimation of yield if the output distribution is Gaussian. `Cpk > 1` indicates greater than 3-sigma yield and `Cpk < 0` indicates that the process mean falls outside the USL or LSL limits.
Errors	Number of points that resulted in evaluation errors for the output

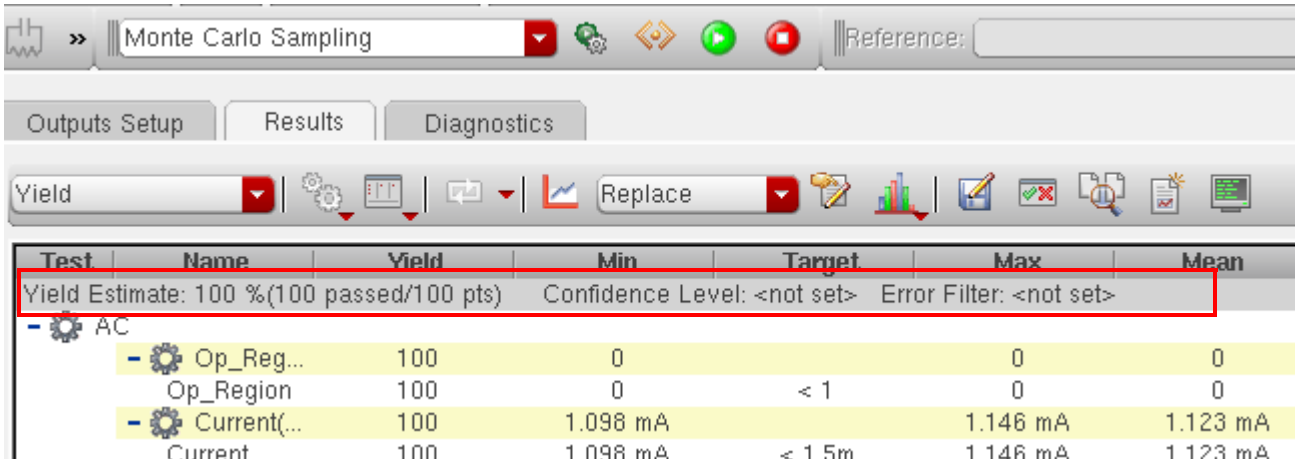
Note: By default, the *Mean -K Sigma* and *Mean +K Sigma* columns are not visible. To view these columns, click *Configure what is shown in the table* and select the names of these columns. Alternatively, you can add the names of these columns to the yieldViewShowDefault environment variable.

© 2023 Cadence Design Systems, Inc.

The summary row for each specification displays the analysis for that specification across all corners. Summary values are derived as follows:

Column	Summary value
Test	Test name
Yield	Minimum yield value across all corners
Min	Minimum value across all corners
Max	Maximum value across all corners
Mean -K Sigma	Mean -K Sigma value across all corners
Mean	Mean value across all corners
Mean +K Sigma	Mean +K Sigma value across all corners
Std Dev	Sample standard deviation across all corners
Spec	Specification defined for the output expression
Pass/Fail	The pass/fail status of the specification across all sweeps and corners
Cpk	Minimum Cpk value across all corners
Errors	Number of points that resulted in simulation or evaluation errors for the output

The gray-colored row at the top displays the *Yield Estimate* – an estimate of the circuit yield taking into account all of the specifications.



© 2023 Cadence Design Systems, Inc.

You can specify a confidence level to view the yield estimate confidence intervals. For more details, refer to Viewing Data for a Specific Confidence Interval Viewing Data for a Specific Confidence Interval.

You can also set the error filter to filter out the results with simulation error or evaluation errors. For more details, refer to Filtering Out Data from the Yield View.

Tip

To specify the names of columns you want to view in the Yield view, on the Results toolbar, click *Configure what is shown in the table* and select the names of columns to show. Alternatively, you can specify the names of columns to show by setting the yieldViewShowDefault environment variable.

© 2023 Cadence Design Systems, Inc.

Capacitance

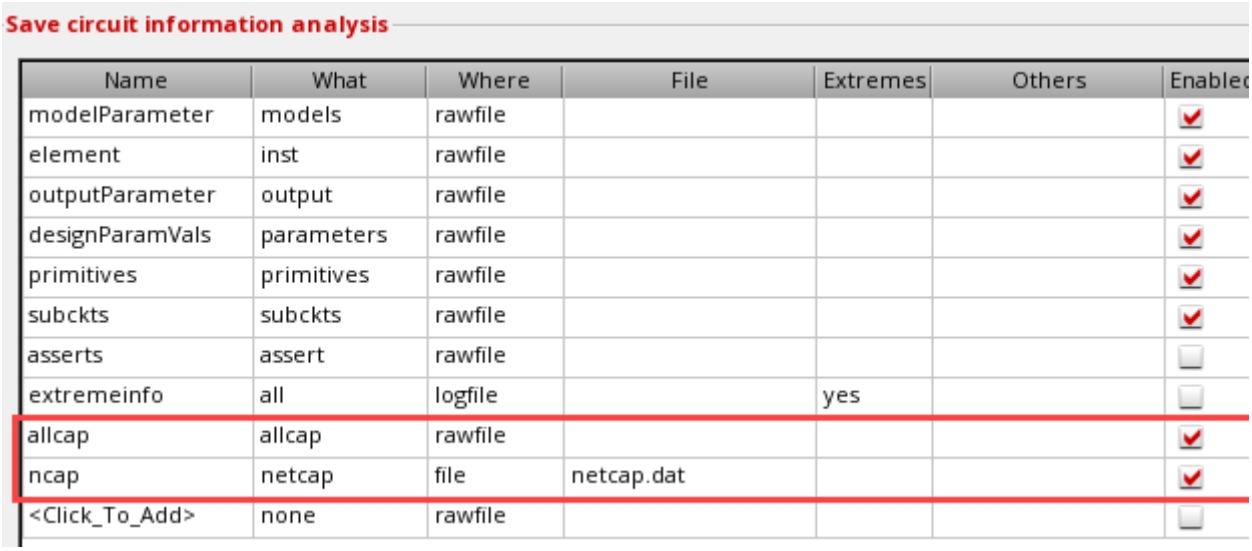
Note: Read a blog related to this at Virtuosity: Examining Post-Layout Capacitance Using Virtuoso ADE Assembler and ADE Explorer.

The *Capacitance* results view displays the capacitance values saved by Spectre by running the netcap analysis.

When is the *Capacitance* results view available in ADE Assembler?

The *Capacitance* results view is displayed when:

- You enable the *allcap* , *netcap*, or *moscap* parameters in the *Save circuit information analysis* table of the Save Options form for Spectre.



  The *allcap* parameter saves the parasitics for both network and devices. *netcap* saves the parasitics for only network. You can change the names of these parameters in the *Name* column of the *Save circuit information analysis* section.

  By default, both parameters are disabled. You can change their default status to enabled by modifying the value of the `spectre.outputs infoOptions` environment variable as shown below.

```
allcap;allcap;file;;;;true netcap;netcap;rawfile;;;;true
```

  When these parameters are enabled, Spectre runs the netcap analysis as part of its simulation and prints the capacitance contribution, design capacitance, and MOSFET-terminal capacitances.

- When you are running simulations with the Single Run, Sweeps, and Corners run mode.

© 2023 Cadence Design Systems, Inc.

- When you are running post-layout simulations with a DSPF file or a Cadence Quantus® Smart View™.

  Note: The netcap analysis is not supported for extracted views, DSPF views, or SPICE files included using Virtuoso Hierarchy Editor.

Tip

If you are not saving the capacitance values, set the showCapacitanceView environment variable to `nil` to hide *Capacitance* from the drop-down list.

Difference Between moscap and oppoint parameters used for info analysis

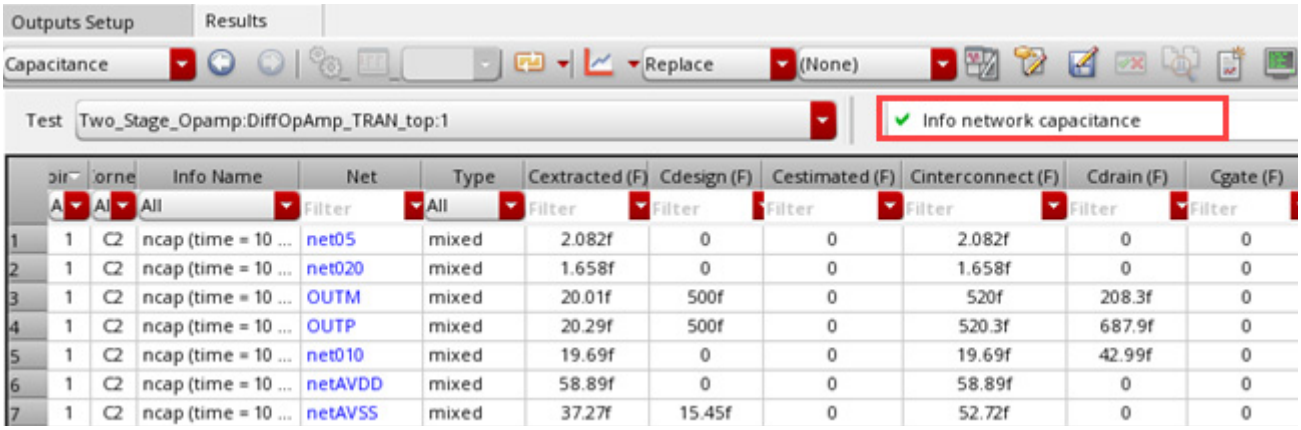
The difference between `moscap` and `oppoint` setting is in the junction capacitances for drain-bulk and source-bulk. `moscap` is used to calculate the capacitance of a single mosfet. A diagonal capacitor equals to the sum of all `moscap` capacitances connected to the terminal, so it includes junction capacitances. The following table shows how `moscap` is calculated in terms of `oppoint`:

moscap	Expression by oppoint
`cdd`	`cdd+cjd`
`css`	`css+cjs`
`cbb`	`cbb+cjd+cjs`
`cdb`	`cdb-cjd`
`cbd`	`cbd-cjd`
`csb`	`csb-cjs`
`cbs`	`cbs-cjs`

© 2023 Cadence Design Systems, Inc.

Details of the Capacitance View

The following figure shows an example of the *Capacitance* results view showing network capacitance data.



	pir	orne	Info Name	Net	Type	Cextracted (F)	Cdesign (F)	Cestimated (F)	Cinterconnect (F)	Cdrain (F)	Cgate (F)
	All	All	All	Filter	All	Filter	Filter	Filter	Filter	Filter	Filter
1	1	C2	ncap (time = 10 ...	net05	mixed	2.082f	0	0	2.082f	0	0
2	1	C2	ncap (time = 10 ...	net020	mixed	1.658f	0	0	1.658f	0	0
3	1	C2	ncap (time = 10 ...	OUTM	mixed	20.01f	500f	0	520f	208.3f	0
4	1	C2	ncap (time = 10 ...	OUTP	mixed	20.29f	500f	0	520.3f	687.9f	0
5	1	C2	ncap (time = 10 ...	net010	mixed	19.69f	0	0	19.69f	42.99f	0
6	1	C2	ncap (time = 10 ...	netAVDD	mixed	58.89f	0	0	58.89f	0	0
7	1	C2	ncap (time = 10 ...	netAVSS	mixed	37.27f	15.45f	0	52.72f	0	0

Note: You can only view or plot data from this results view. You cannot use the capacitance data to create measurements, expressions, or specifications on the *Outputs Setup* tab.

By default, the results table in the *Capacitance* view shows the net capacitance data for the first test in the setup. You can choose another test from the *Test* drop-down list on top.

The table shows two types of data that you can choose from the drop-down list to the right of the *Test* drop-down list:

- Info Network Capacitance
- Info Device Capacitance

Info Network Capacitance

The table for network capacitance contains the following columns:

Column	Description
Point	Point number
Corner	Name of the corner
Info Name	Name of the information as specified in the *Save circuit information analysis* section of the Save Options form.
Net	Name of the net for which capacitance values are displayed. A net name is also a link to the net in the schematic view.

© 2023 Cadence Design Systems, Inc.

Type	Type of net. It contains one of the following values: ■ `schematic`, which means that the net exists only in the pre-layout design ■ `polo`, which means that the net exists inside the post-layout view or file and there is a corresponding net in the schematic too. ■ `mismatch`, which means that the net exists only inside the post-layout view or file ■ `mixed`, which means that the net exists in both pre-layout and post-layout **Note:** Hyperlinks do not work for the nets of type mismatch because the nets do not exist in the schematic view.
Cextracted	Capacitance contribution from the capacitors in an extracted view (DSPF, SPEF).
Cdesign	Capacitance contribution from the capacitors in the schematic view.
Cestimated	Capacitance contribution from capacitors in files with special formats. For example, an SPEF file.
Cinterconnect	The sum of values in columns *Cextracted*, *Cdesign*, and *Cestimated*.
Cdrain	Capacitance contribution from the MOSFET drain.
Cgate	Capacitance contribution from the MOSFET gate.
Csource	Capacitance contribution from the MOSFET source.
Cbulk	Capacitance contribution from the MOSFET bulk.
Ctotal	The sum of values in columns *Cinterconnect*, *Cdrain*, *Cgate*, *Csource*, and *Cbulk*.

Note: Devices with fingers or m-factor parameters are shown in separate rows.

You can use the filters in each column header to view data for a particular corner, instance, location, or capacitance value.

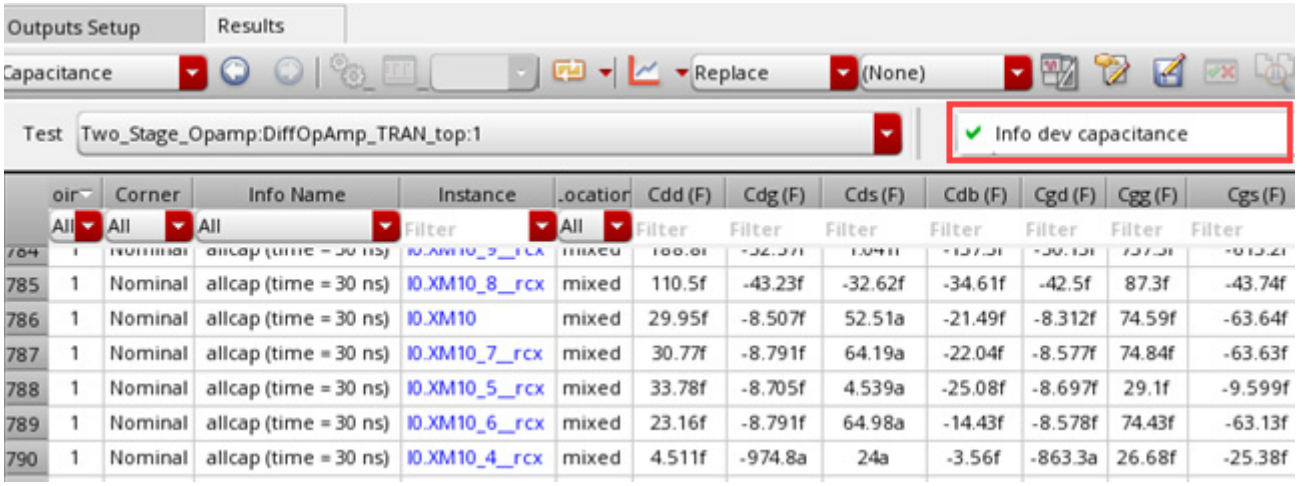
The net capacitance data is saved in the `netlist.info.netcap` file in ASCII format.

© 2023 Cadence Design Systems, Inc.

Info Device Capacitance

You can view the capacitance values for devices or MOSFETs by choosing *Info device capacitance* from the drop-down list to the right of *Test*.

The following figure shows an example of the *Capacitance* results view showing device capacitance data.



	oir	Corner	Info Name	Instance	.ocatior	Cdd (F)	Cdg (F)	Cds (F)	Cdb (F)	Cgd (F)	Cgg (F)	Cgs (F)
	All	All	All	Filter	All	Filter	Filter	Filter	Filter	Filter	Filter	Filter
785	1	Nominal	allcap (time = 30 ns)	I0.XM10_8__rcx	mixed	110.5f	-43.23f	-32.62f	-34.61f	-42.5f	87.3f	-43.74f
786	1	Nominal	allcap (time = 30 ns)	I0.XM10	mixed	29.95f	-8.507f	52.51a	-21.49f	-8.312f	74.59f	-63.64f
787	1	Nominal	allcap (time = 30 ns)	I0.XM10_7__rcx	mixed	30.77f	-8.791f	64.19a	-22.04f	-8.577f	74.84f	-63.63f
788	1	Nominal	allcap (time = 30 ns)	I0.XM10_5__rcx	mixed	33.78f	-8.705f	4.539a	-25.08f	-8.697f	29.1f	-9.599f
789	1	Nominal	allcap (time = 30 ns)	I0.XM10_6__rcx	mixed	23.16f	-8.791f	64.98a	-14.43f	-8.578f	74.43f	-63.13f
790	1	Nominal	allcap (time = 30 ns)	I0.XM10_4__rcx	mixed	4.511f	-974.8a	24a	-3.56f	-863.3a	26.68f	-25.38f

The table for MOSFET or device capacitance shows the point, corner, info name, name of device instance, device location, and the capacitance between the different terminals of MOSFETs. For example, the capacitance between two drains (Cdd), a drain and a gate (Cdg), or a drain and a source (Cds). The device name is also a link to the device in the schematic view.

Note: Devices with fingers or m-factor parameters are shown in separate rows.

You can use the filters in each column header to view data for a particular corner, instance, location, or capacitance value.

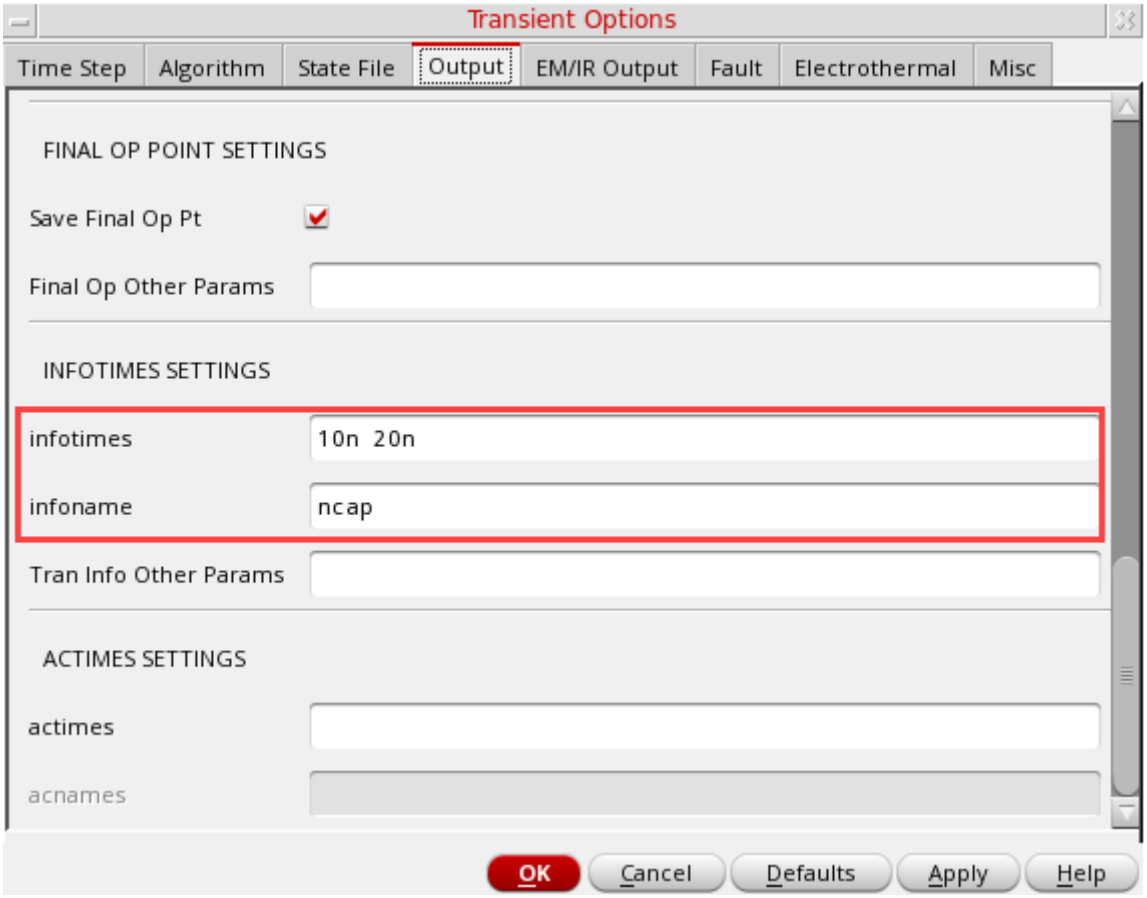
Saving Capacitance Data at Multiple Time Points

You can save the capacitance information at different time points by modifying the settings in the *INFOTIMES SETTINGS* on the *Output* tab of the Transient Options form:

- The *infotimes* field specifies the time points.
- The *infoname* field specifies the name of the information type for which data is to be saved at the given time points. This value in this field must match the value in the *Name* column of the *Save circuit information analysis* section of the Save Options form.

© 2023 Cadence Design Systems, Inc.

The following example shows the settings to compute net capacitance information at two time points 10 nanoseconds and 20 nanoseconds.



When you save information at multiple time points, the information name in the *Info Name* column of the Info Network Capacitance table includes the time point as well.

© 2023 Cadence Design Systems, Inc.

Customizing the Detail - Transpose View

Most of the customization that you can perform in the *Detail - Transpose* view are the same as in other results views. The following topics describe the tasks that are specific to the *Detail - Transpose* view, or different from those in other results views:

- Hiding and Showing Columns in the Detail - Transpose View
- Changing the Order of Columns in the Detail - Transpose View
- Viewing the Test Name for Outputs in the Detail - Transpose View

Hiding and Showing Columns in the Detail - Transpose View

To hide a column, do the following:

➤ Right-click the name of the column and choose *Hide Column*.

To hide multiple columns in the Detail-Transpose view:

1. Select one or more columns.
2. Right-click any one of the selected column.
3. Choose *Hide Selected Columns*.

To display all the columns, do the following:

➤ Right-click the name of any column and choose *Show All Column*s.

Changing the Order of Columns in the Detail - Transpose View

To change the order of columns, do the following:

1. Click a column name.
2. Drag and drop the column at the desired location.

Viewing the Test Name for Outputs in the Detail - Transpose View

To view the name of the test for which an output is defined, do the following:

➡ Place the mouse pointer on the name of the column for the output on the right hand side of the *Detail - Transpose* view.

The name of the test for which the output is defined is displayed in a pop-up.

© 2023 Cadence Design Systems, Inc.

Important

In the *Detail - Transpose* view, you can also view the statistical parameter values used on each point and create a statistical corner that can be used to rerun a particular point in the distribution. These commands are available in the context-sensitive menu commands.

Using the Status View

See the following topics for more information about using the *Status* view on the Results tab of the Outputs pane:

- Stopping the Simulation Run from the Status View
- Hiding and Showing Information in the Status View

Stopping the Simulation Run from the Status View

To stop the simulation run, do the following:

➡ Click the *Stop* button next to the *Progress* bar.

Hiding and Showing Information in the Status View

Click the button on the Results tab. A check mark next to an item indicates that information for that item is being displayed.

To hide information, do the following:

➤ Click the button on the Results tab and choose the item.

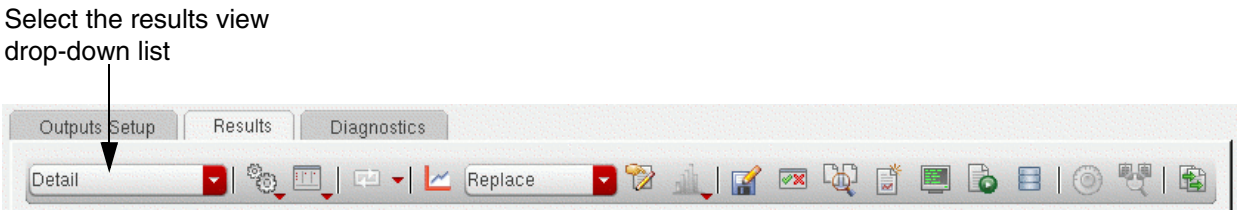
To unhide the information click the button once again and choose the item.

Switching Between Results Views

To switch between views, do the following:

© 2023 Cadence Design Systems, Inc.

➤ From the *Select the results view* drop-down list, select the view you want to display.

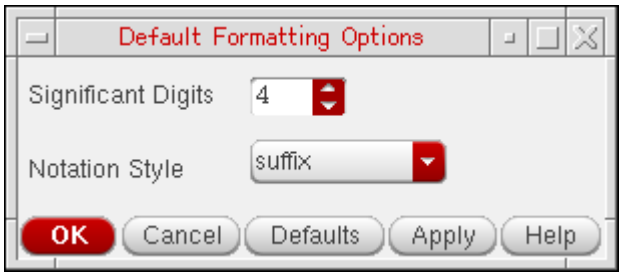


Specifying the Default Formatting Options

To specify the default number of significant digits and notation style for the measured results displayed in the Results tab of the Outputs pane, do the following:

1. Choose *Options – Outputs Formatting*.

   The Default Formatting Options form appears.



2. Specify the number of significant digits in the *Significant Digits* field.

   The default value is `4`. Valid values are `2` to `15`.

3. Select the notation style in the *Notation Style* drop-down list.

Select	To
suffix	Display the results in the suffix notation. For example, `5.802m` is displayed as `5.802m` when number of significant digits is `4`. This is the default value.

© 2023 Cadence Design Systems, Inc.

Select	To
engineering	Display the results in the engineering notation. For example, `5.802m` is displayed as `5.802e-3` when number of significant digits is `4`.
scientific	Display the results in the scientific notation. For example, `5.802m` is displayed as `5.802e-03` when number of significant digits is `4`.

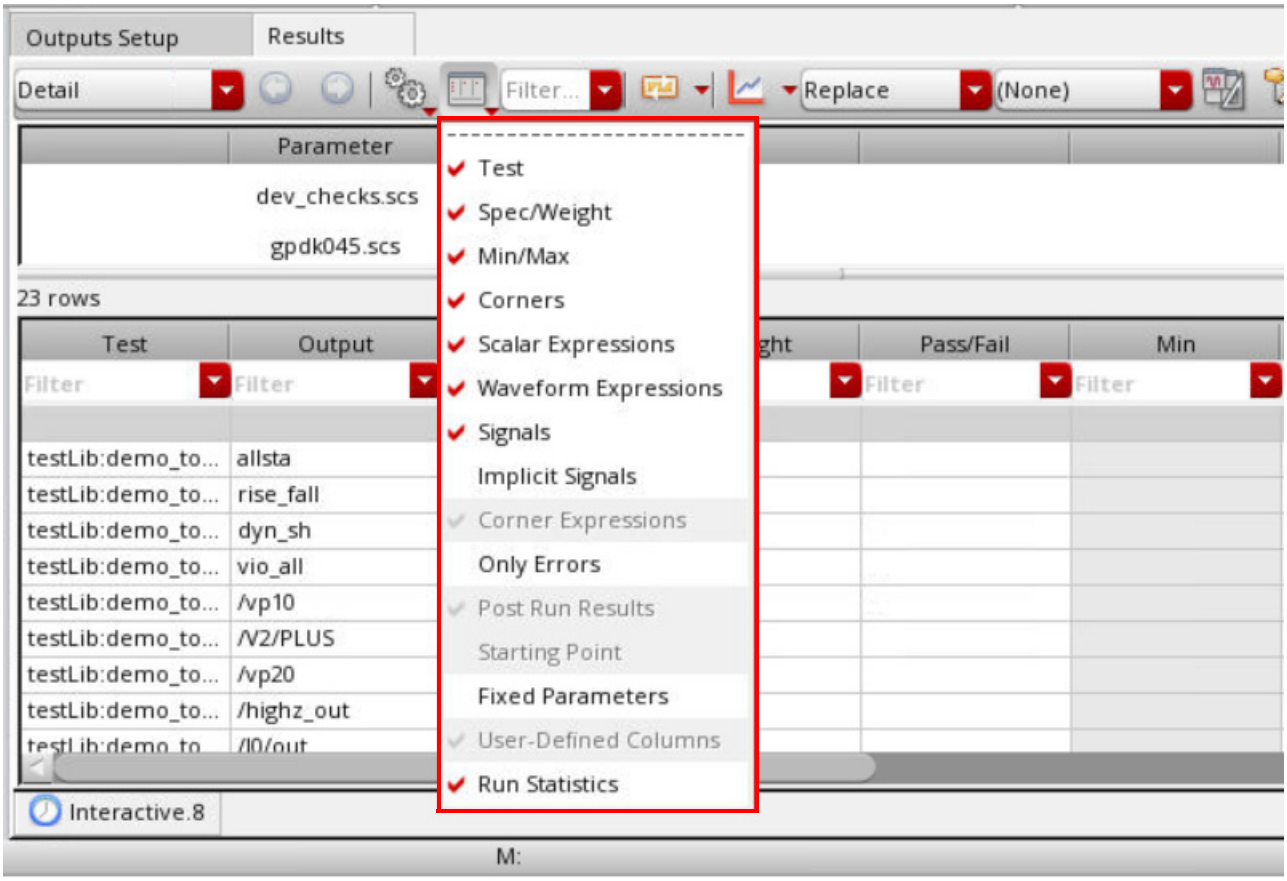
Tip

You can use the *Units*, *Digits*, *Notation* and *Suffix* columns in the Outputs Setup tab of the Outputs pane to override the default number of significant digits and notation style for displaying the results for individual output expressions. For more information, see Adding or Modifying an Output Expression on page 367.

© 2023 Cadence Design Systems, Inc.

Configuring the Results Tab

You can view or hide columns and rows displayed on the *Results* tab by using the C*onfigure what is shown in the table* list on the toolbar. When you click this button, ADE Assembler shows the following menu.



Note: The *Run Statistics* command is available only if the simulation is run using the LSCS job control mode and the environment variable enableRunStatisticInfoDisplay is set to `t`. The specified simulator must be Spectre or AMS.

For information about these options, see the following topics:

- Hiding Test Details
- Hiding Specification Details
- Hiding Corner Results
- Hiding Signals
- Hiding Measured Result Values

© 2023 Cadence Design Systems, Inc.

- Hiding Minimum and Maximum Values
- Viewing Only Errors
- Viewing the Starting Point
- Customizing the Display of Variables and Parameters for Design Points
- Viewing the Run Statistics
- Focusing on the Results Values that Fail to Meet Specifications (Detail-Transpose view only)
- Focusing on the Results with Large Variations from Specifications (Detail-Transpose view only)

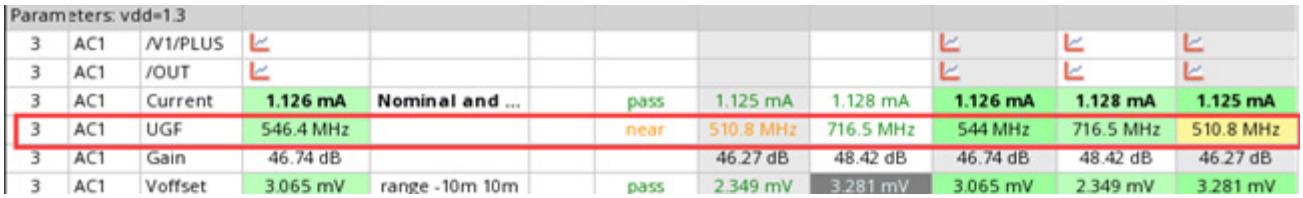
Configuring Text Color for Pass/Fail, Min, and Max Columns

The default text color in the *Min* and *Max* columns is black, but you can specify different colors to be used to indicate the pass, fail, or near status of values in these columns. The same set of colors are applied to the *Pass/Fail* column too. To change the text colors, you can use the following environment variables in CIW or .cdsenv file:

- `textColorForSpecFailInResults`
- `textColorForSpecPassInResults`
- `textColorForSpecNearInResults`

Note: If you set these variables in CIW, the revised colors are applied only for the results displayed after setting these variables, or when you run re-evaluation for the results already displayed in the *Outputs* tab.

In the example given below, the text color for result values nearing spec has been changed to orange.



Parameters: vdd=1.3											
3	AC1	/V1/PLUS									
3	AC1	/OUT									
3	AC1	Current	**1.126 mA**	**Nominal and ...**		pass	1.125 mA	1.128 mA	**1.126 mA**	**1.128 mA**	**1.125 mA**
3	AC1	UGF	546.4 MHz			near	510.8 MHz	716.5 MHz	544 MHz	716.5 MHz	510.8 MHz
3	AC1	Gain	46.74 dB				46.27 dB	48.42 dB	46.74 dB	48.42 dB	46.27 dB
3	AC1	Voffset	3.065 mV	range -10m 10m		pass	2.349 mV	3.281 mV	3.065 mV	2.349 mV	3.281 mV

© 2023 Cadence Design Systems, Inc.

Hiding and Showing Data on the Results Tab

The following data appears on the Results tab of the Outputs pane for the Detail view. See Searching for Conditional Results for more information about other views in the Results tab.

Table 19-1 Columns in the Detail View

Column	Data/Content	Hide/Unhide
Point	Number of the design point for which results are displayed.	
Test	Test name, top of expandable tree of outputs	See Hiding Test Details
Output	Signal or expression/measurement name	
Nominal	Plot icon for signals and waveform expressions; status indicator box and measured value for expressions at the nominal corner: green indicates a value within the specifications; yellow indicates a value that is no more than 10% outside the target value of specifications; red indicates a value that is more than 10% outside the target value	See Hiding Signals See Hiding Measured Result Values
Spec	Specification information	See Hiding Specification Details
Weight	Weighting factor given to a specification	
Pass/Fail	*pass* when values are within specifications; *near* when one or more measured values are no more than 10% outside the target value of the specification; *fail* when one or more measured values are more than 10% outside the target value of the specification. For more information about the *pass*, *near* or *fail* status, see Viewing Specification Results in the Results Tab on page 1403.	
Min	Minimum value for the specification	Hiding Minimum and Maximum Values
Max	Maximum value for the specification	
`CornerName`	Measured value or waveform at each corner	See Hiding Corner Results

© 2023 Cadence Design Systems, Inc.

Hiding Test Details

Test details appear in the *Test* column on the Results tab of the Outputs pane. To hide test details, do the following:

➤ Click the (*Configure what is shown in the table*) button on the Results tab and select *Test*.

The *Test* column disappears from the Results tab.

To unhide this column, click the button once again and select *Test*.

Hiding Specification Details

Specification details appear in the *Spec* and *Weight* columns on the Results tab of the Outputs pane. To hide specification details, do the following:

➤ Click the button on the Results tab and select *Spec/Weight*.

The *Spec* and *Weight* columns disappear from the Results tab.

To unhide these columns, click the button once again and select *Spec/Weight*.

Hiding Corner Results

Corner results appear in columns on the Results tab of the Outputs pane, each result in its own column. To hide corner results columns, do the following:

➤ Click the button on the Results tab and select *Corners*.

The corner results columns disappear from the Results tab.

To unhide these columns, click the button once again and select *Corners*.

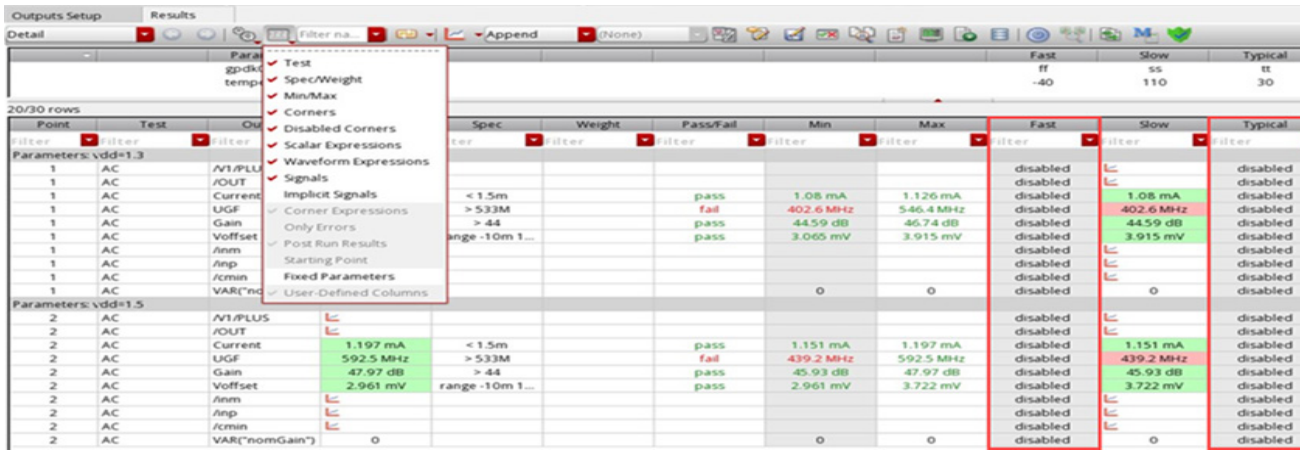
Hiding Disabled Corners

To hide the disabled corners in the *Results* tab, deselect the *Disabled Corners* command from the *Configure what is shown in the table* drop-down list on the toolbar of the *Results* tab.

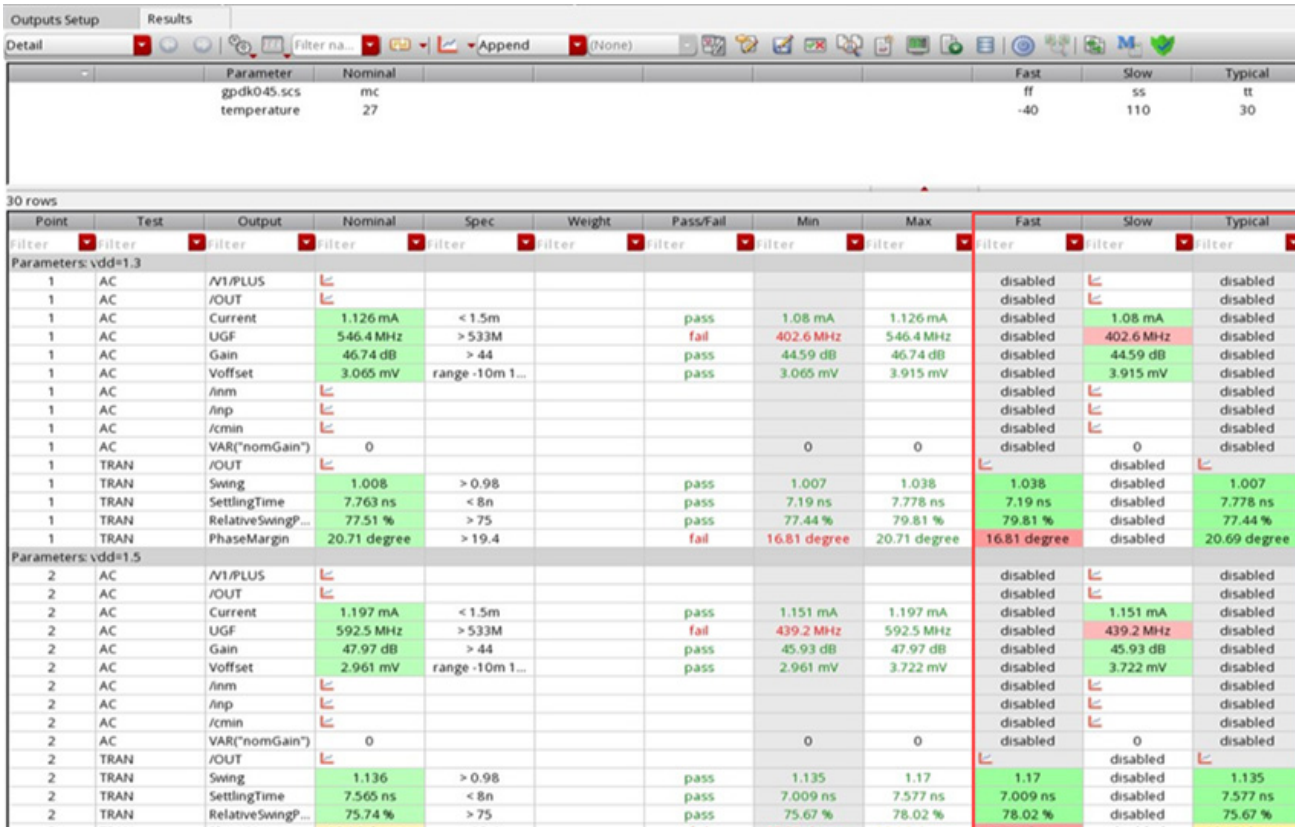
While working with this command, the following cases are possible.

© 2023 Cadence Design Systems, Inc.

- On selecting the *Disabled Corners* command, the disabled corners are indicated in the *Results* tab with the text `disabled` written in the corner column.

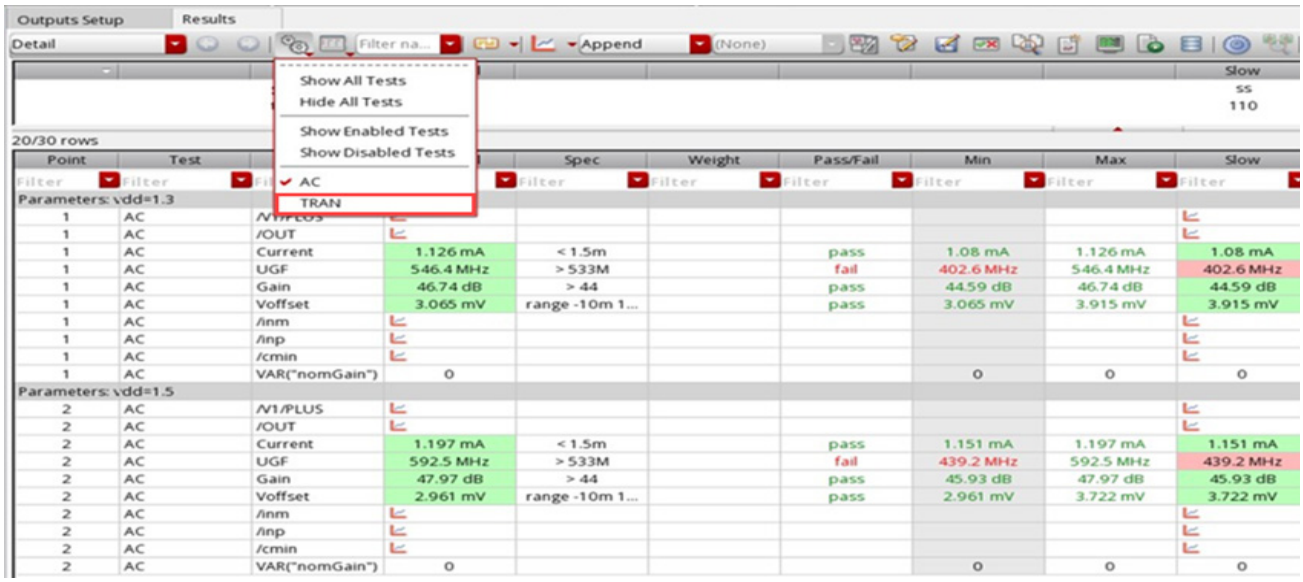


- On disabling a few corners for each test, corner columns with partially filled values are displayed in the *Results* tab.



- On hiding the test containing the disabled corners, the related corner columns are removed from the *Results* tab.

© 2023 Cadence Design Systems, Inc.



Hiding Signals

Signals appear as rows with the icon in the *Nominal* column and the columns for corners on the Results tab of the Outputs pane.

To hide signals, do the following:

➤ Click the button on the Results tab and select *Signals*.

The signal rows disappear from the Results tab. If you have any measured values for expressions, those are the only values that appear.

To unhide these rows, click the button once again and select *Signals*.

Note: This command is available only in the *Detail*, *Detail - Transpose*, and *Optimization* result views.

Hiding Measured Result Values

Measured result values appear in the *Nominal* column and the columns for corners on the Results tab of the Outputs pane.

To show or hide these measured result values, you can use the following commands from the (*Configure what is shown in the table*) drop-down list on the toolbar of the Results tab:

© 2023 Cadence Design Systems, Inc.

- *Scalar Expressions*: Shows or hides results of the expressions that evaluate to scalar results.
- *Waveform Expressions*: Shows or hides results of the expressions that evaluate to waveform results.

Note: These commands are available only in the *Detail*, *Detail - Transpose*, and *Optimization* result views.

Hiding Minimum and Maximum Values

The minimum and maximum values appear in the *Min* and *Max* columns on the Results tab of the Outputs pane. To hide minimum and maximum values, do the following:

➤ Click on the Results tab and select *Min/Max*.

The *Min* and *Max* columns disappear from the Results tab.

To unhide these columns, click the button once again and select *Min/Max*.

Viewing Only Errors

To show errors only, do the following:

➤ Click the button on the Results tab and select *Only Errors*.

To hide these rows, click the button once again and select *Only Errors*.

Viewing the Starting Point

The starting point specified for Monte Carlo Sampling, Global Optimization, Local Optimization, Improve, Sensitivity Analysis, and Size Over Corners runs is displayed as the starting point on the Results tab of the Outputs pane. To show the starting point on the top of the Results tab, do the following:

➤ Click on the Results tab and select *Starting Point*.

The rows displaying the starting point appears on the top of the Results tab.

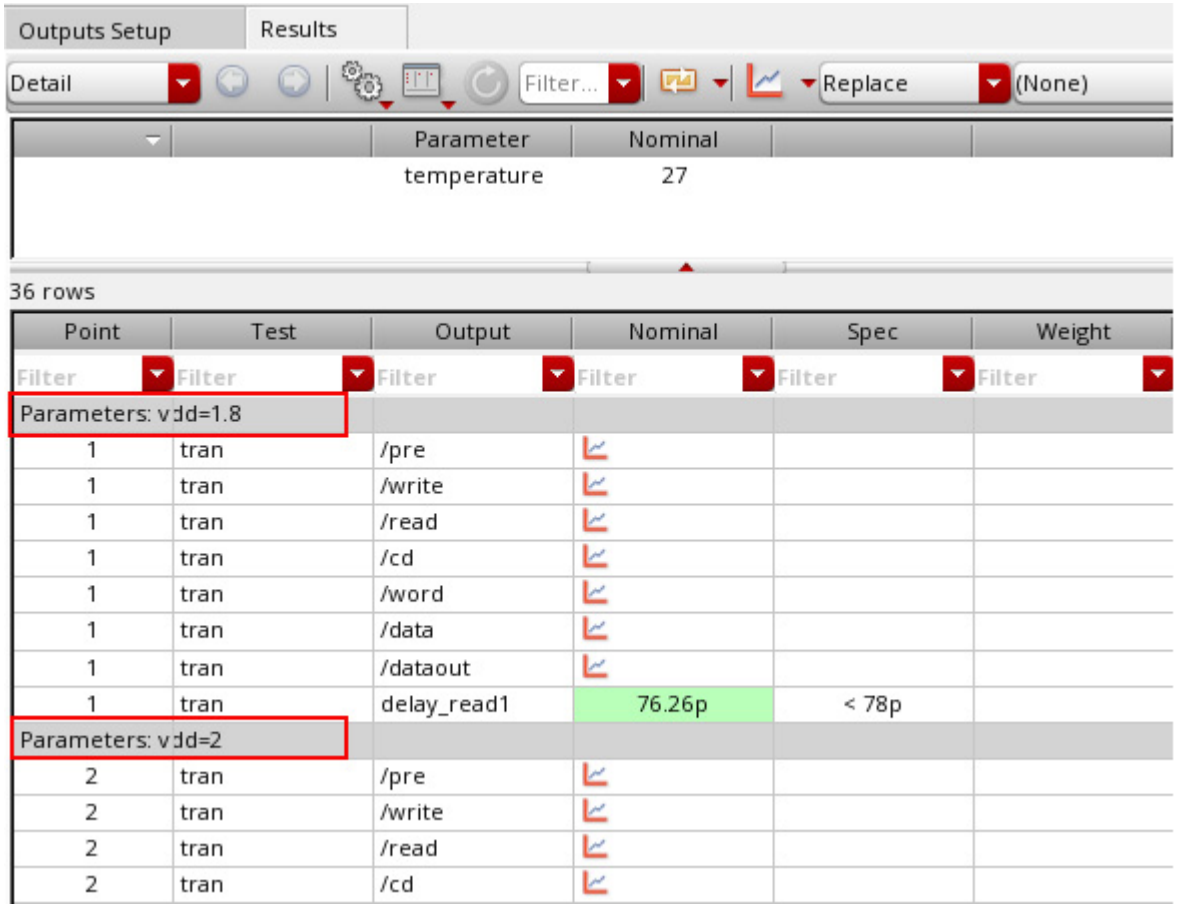
To hide these rows, click once again and select *Starting Point*.

© 2023 Cadence Design Systems, Inc.

Customizing the Display of Variables and Parameters for Design Points

ADE Assembler shows the design variables and device parameters on the gray header of each design point displayed in the *Results* tab. When a lot of variables or parameters are used, it may not be possible to accommodate all of those on the gray header or the header may look overfilled. Therefore, by default, ADE Assembler shows only those variables and parameters that are swept or varied across design points.

For example, in the following figure, for each design point, the gray bars show the values of the swept parameter, `vdd`. Other variables or parameters that had fixed values across all design points are hidden.



Depending on your requirement to view the values of fixed variables, you can customize the view to show or hide selected design variables or device parameters on the gray headers in the following two ways:

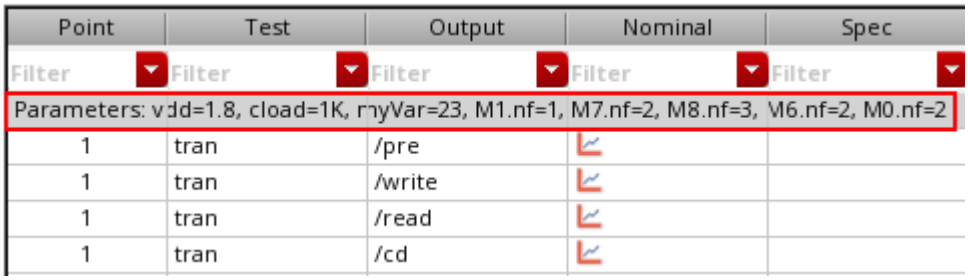
- Display All Fixed and Swept Variables and Parameters
- Show or Hide Design Variables Fixed for Some Tests

© 2023 Cadence Design Systems, Inc.

Display All Fixed and Swept Variables and Parameters

To view the values of all design variables and device parameters, fixed or swept, used for a design point, click *Configure what is shown in the table* on the toolbar of the *Results* tab and select *Fixed Parameters/Variables*.

All design variables and parameters, swept or fixed, for a design point are displayed on the header.



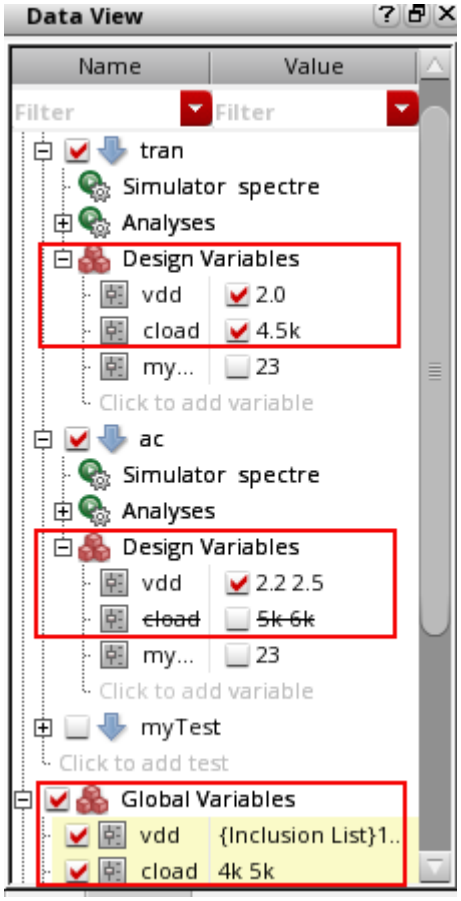
Note: The *Fixed Parameters/Variables* command is not selected by default.

Show or Hide Design Variables Fixed for Some Tests

By default, any design variable that is swept in the setup is shown in the gray headers for all design points even is it is fixed for a particular design point.

© 2023 Cadence Design Systems, Inc.

Consider an example with the following setup:



Test tran uses an overridden static value for vdd and cload. Values are not swept.

Test ac uses swept overridden values for vdd.
It also uses swept values for cload from the global variables.

By default, all design points show the values of `vdd` and `cload` in the gray header.

Point	Test	Output
Filter	Filter	Filter
Parameters: vdd=2, cload=4.5K		
1	tran	/pre
1	tran	/word
1	tran	/write
Parameters: vdd=2.2, cload=4K		
2	ac	/pre
2	ac	/write

However, if you want to identify only those design points that are using swept values for `vdd` and `cload`, you can hide the design variables that are fixed for the design point that belong to test `tran` in this example.

© 2023 Cadence Design Systems, Inc.

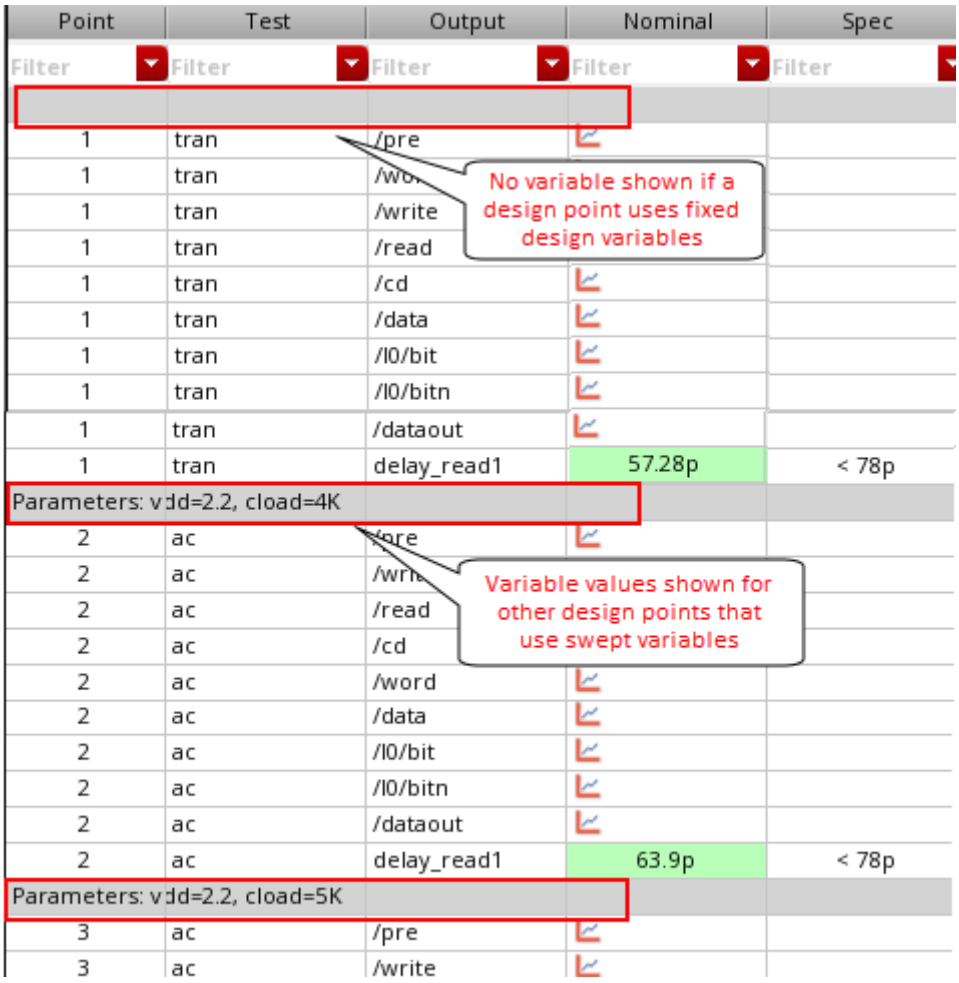
To hide the design points with fixed values for variables:

1. Set the `displayFixedTestVariables` environment variable to `nil`.

   This variable is set to `t` by default.

2. Deselect the *Fixed Variables/Parameters* command in the *Configure what is shown in the table* drop-down list on the toolbar.

   Next time, when the results table is refreshed, the gray header hides the variables that not swept for a design point, as shown below.



Point	Test	Output	Nominal	Spec
Filter	Filter	Filter	Filter	Filter
1	tran	/pre		
1	tran	/wo		
1	tran	/write		
1	tran	/read		
1	tran	/cd		
1	tran	/data		
1	tran	/I0/bit		
1	tran	/I0/bitn		
1	tran	/dataout		
1	tran	delay_read1	57.28p	< 78p
Parameters: vdd=2.2, cload=4K				
2	ac	/pre		
2	ac	/wri		
2	ac	/read		
2	ac	/cd		
2	ac	/word		
2	ac	/data		
2	ac	/I0/bit		
2	ac	/I0/bitn		
2	ac	/dataout		
2	ac	delay_read1	63.9p	< 78p
Parameters: vdd=2.2, cload=5K				
3	ac	/pre		
3	ac	/write		

Related Topics

displayFixedTestVariables

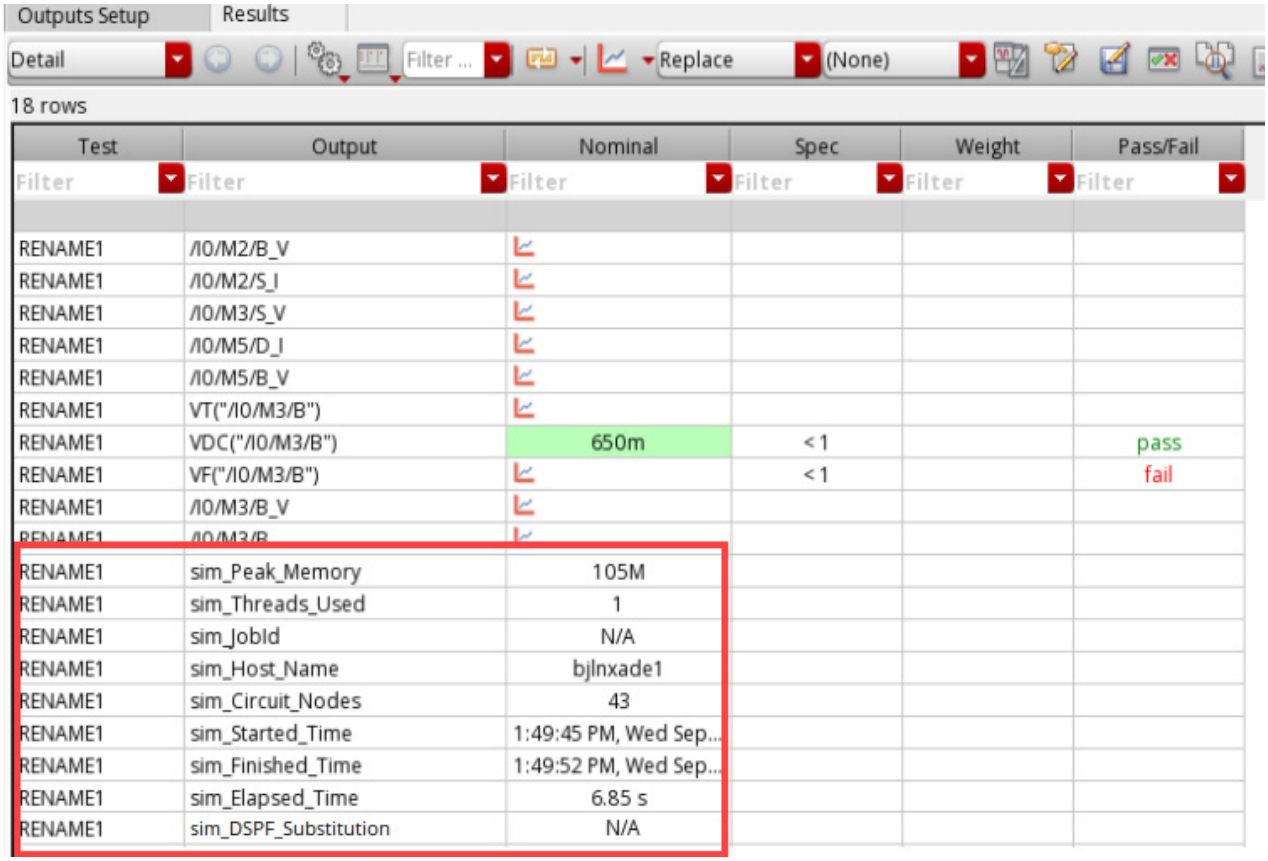
Showing Variable Parameters for Design Points in the Variable Display Assistant

© 2023 Cadence Design Systems, Inc.

Viewing the Run Statistics

Note: To view the run statistics information in the *Results* tab, select the *Run Statistics* command from the *Configure what is shown in the table* list on the toolbar of the *Results* tab. This command is available only if the simulation is run using the LSCS job control mode and the environment variable enableRunStatisticInfoDisplay is set to `t`.

ADE Assembler adds outputs displaying run statistics information in the *Results* tab.



Test	Output	Nominal	Spec	Weight	Pass/Fail
Filter	Filter	Filter	Filter	Filter	Filter
RENAME1	/I0/M2/B_V				
RENAME1	/I0/M2/S_I				
RENAME1	/I0/M3/S_V				
RENAME1	/I0/M5/D_I				
RENAME1	/I0/M5/B_V				
RENAME1	VT("/I0/M3/B")				
RENAME1	VDC("/I0/M3/B")	650m	< 1		pass
RENAME1	VF("/I0/M3/B")		< 1		fail
RENAME1	/I0/M3/B_V				
RENAME1	/I0/M3/B				
RENAME1	sim_Peak_Memory	105M			
RENAME1	sim_Threads_Used	1			
RENAME1	sim_JobId	N/A			
RENAME1	sim_Host_Name	bjlnxade1			
RENAME1	sim_Circuit_Nodes	43			
RENAME1	sim_Started_Time	1:49:45 PM, Wed Sep...			
RENAME1	sim_Finished_Time	1:49:52 PM, Wed Sep...			
RENAME1	sim_Elapsed_Time	6.85 s			
RENAME1	sim_DSPF_Substitution	N/A			

These outputs show the following information:

- `sim_Peak_Memory`: Peak memory consumption for Spectre
- `sim_Threads_Used`: Number of threads used by Spectre
- `sim_Threads_Recommended`: Number of threads recommended Spectre X or APS for a simulation
- `sim_JobId`: LSF job identifier, shown only when you are using the LBS distribution method
- `sim_Host_Name`: Name of the host on which simulation was run

© 2023 Cadence Design Systems, Inc.

- `sim_Circuit_Nodes`: The node number in the Spectre inventory
- `sim_Started_Time`: Start time of the simulation run
- `sim_Finished_Time`: End time of the simulation run
- `sim_Elapsed_Time`: Elapsed time of the Spectre simulation run
- `sim_DSPF_Substitution`: The subckt information related to DSPF substitution that is extracted from the Spectre output file.
- `sim_Intrinsic_Tran_Analysis_Time`: Transient intrinsic elapsed time for the Spectre simulation run
- `sim_CPU_Time`: CPU time reported by Spectre
- `sim_CPU_Utilization`: CPU utilization reported by Spectre

More values can be displayed for debugging. You can customize the list of outputs by using the environment variable defaultRunStatisticItems.

Focusing on the Results Values that Fail to Meet Specifications (Detail-Transpose view only)

In the Detail-Transpose view, by default, the result columns are arranged in the same order in which they are defined on the *Output Setup* tab. If the number of results is high, you need to scroll to the right to view all the columns. You can either hide columns by using the *Hide Column* command in the context-sensitive menu of the columns.

To focus on the result values that failed to meet the specification, you can choose the *Failure Count* command from the (*Configure what is shown in the table*) drop-down list on the toolbar of the *Results* tab.

The *Failure Count* command rearranges the output columns in the Detail-Transpose view to display the columns with failing results in the left.

Note: You can choose the *Default* command from the drop-down list to view the default arrangement of the columns.

Focusing on the Results with Large Variations from Specifications (Detail-Transpose view only)

In the Detail-Transpose view, by default, the result columns are arranged in the same order in which they are defined on the *Output Setup* tab. If the number of results is high, you need to scroll to the right to view all the columns.

© 2023 Cadence Design Systems, Inc.

To focus on the result values with a large variation from the specification, you can choose the *Range Spread* command from the (*Configure what is shown in the table*) drop-down list on the toolbar of the *Results* tab.

The *Range Spread* command rearranges the output columns in the Detail-Transpose view to display the columns in the decreasing order of variation from the specifications. The columns with results that have the largest variation from the given specification are displayed on the left.

Note: You can choose the *Default* command from the drop-down list to view the default arrangement of the columns.

Hiding and Showing Results for Tests

By default, the *Results* tab displays the results for all the tests. However, if you need to focus on the results of a particular test, you can choose to hide the rows related to that test.

To view only the results for a specific test, do the following:

1. Click the button on the Results tab and select *Hide All Tests*.
2. Click the button on the Results tab and select a specific test.

To view the results for all the tests, do the following:

➤ Click the button on the Results tab and select *Show All Tests*.

To hide the results for all tests, do the following:

➤ Click the button on the Results tab and select *Hide All Tests*.

The tool filters out the results for the selected tests. These filters are applied to both `Detail` and *Detail - Transpose* result views.

In case the tests use overridden values for variables, the parameter header rows are updated to show the variable values related to only those tests that are visible. For more details, refer to Updating Global Variable Values with Design Variable Values.

Similarly, if a corner is disabled for a test, only the corner columns that are enabled for the visible tests are shown. Other corners are hidden.

© 2023 Cadence Design Systems, Inc.

Freezing Columns in the Detail and Optimization View

You can view two areas of the Results tab and lock the columns in one area by freezing columns. When you freeze columns, you select specific columns that remain visible when scrolling in the Results tab.

Note: You can freeze columns only in the Detail, Yield, and Optimization views in the Results tab.

To freeze columns, do the following:

- Right-click the column to the right of where you want the split to appear and choose *Freeze columns*.

For example, to freeze the *Output* column and all the columns to the left of the *Output* column, right-click the heading for the *Output* column and choose *Freeze columns*. This allows you to view the contents of the *Output* column and all the columns to the left of the *Output* column when you use the horizontal scroll bar in the Results tab scroll through the results for corners.

© 2023 Cadence Design Systems, Inc.

Sorting Results in the Detail and Detail-Transpose Views

You can sort data in the *Detail* or *Detail-Transpose* result views to view the related data rows together in a the ascending or descending order. For example, you can sort rows by the output name, test name, or in the order of result values in any column.

To sort results based on a specific column in the *Detail* or *Detail-Transpose* result views:

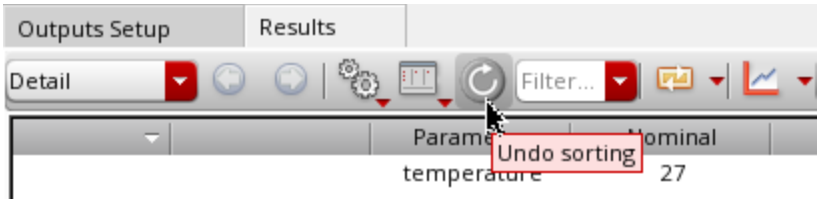
➡ Click the header of the column that you want to use as base for sorting.

All the rows in the results table are sorted according the values in that column. The up or down arrow on the top-right area of the column header shows if the rows are being arranged in the ascending or the descending order.

To undo sorting of results:

➡ Click *Undo sorting* on the toolbar of the *Results* tab.

The results are rearranged in the default order as they were visible before sorting.



© 2023 Cadence Design Systems, Inc.

Filtering Results on the Results Tab

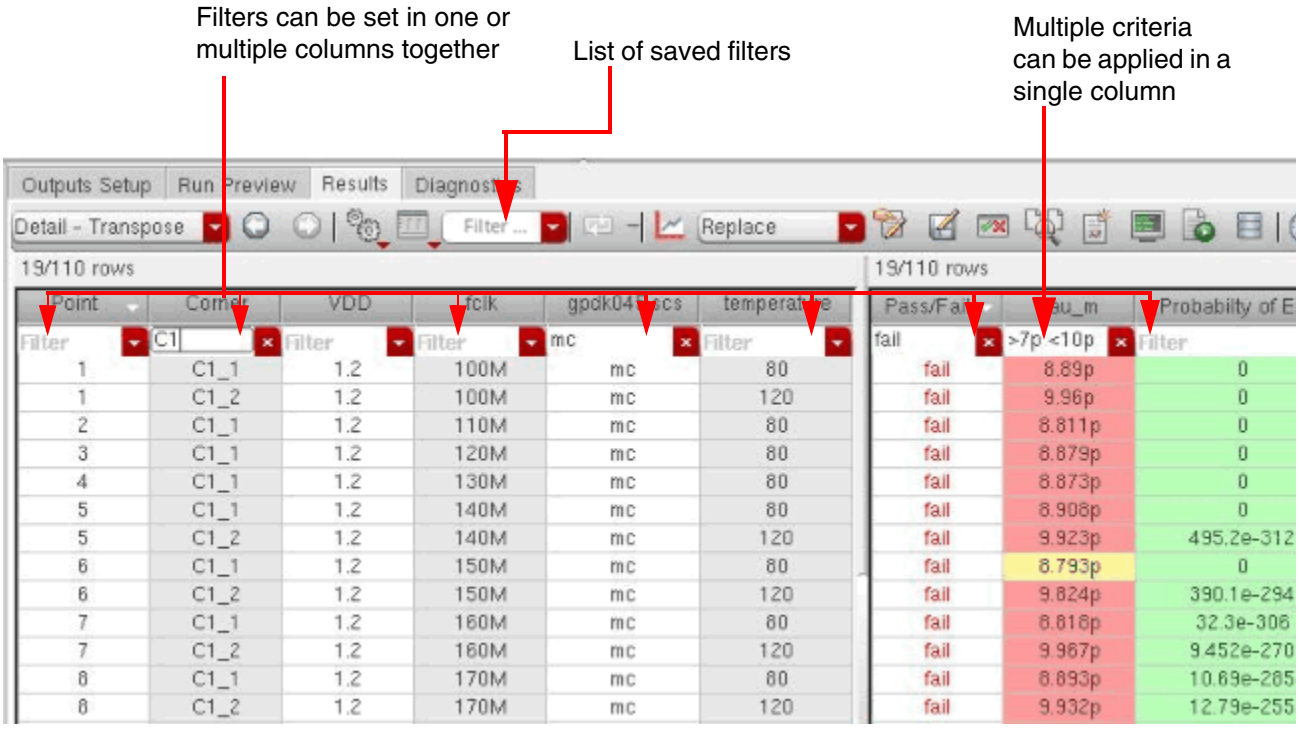
Video

You can view a video demonstration for this feature at Filtering Data on the Outputs Setup and Results Panes.

The *Results* tab of ADE Assembler displays the results of measurements for all the tests and their outputs enabled in the active setup. When the number of outputs are large, you can apply filters to selectively display the results you are interested in viewing.

You can apply filters to view only those rows in the Results table that contain values specified by a filter criteria. You can either enter the filter criteria in the *Filter* text field on top of a column, or select values form the drop-down list to the right of that field. For example, in the Detail results view, you want to view all the rows that show the results for output `Gain`. For this, select `Gain` from the drop-down list. All the rows that show the results for other outputs are hidden.

The following figure shows a filter applied in the *Detail - Transpose* results view.



© 2023 Cadence Design Systems, Inc.

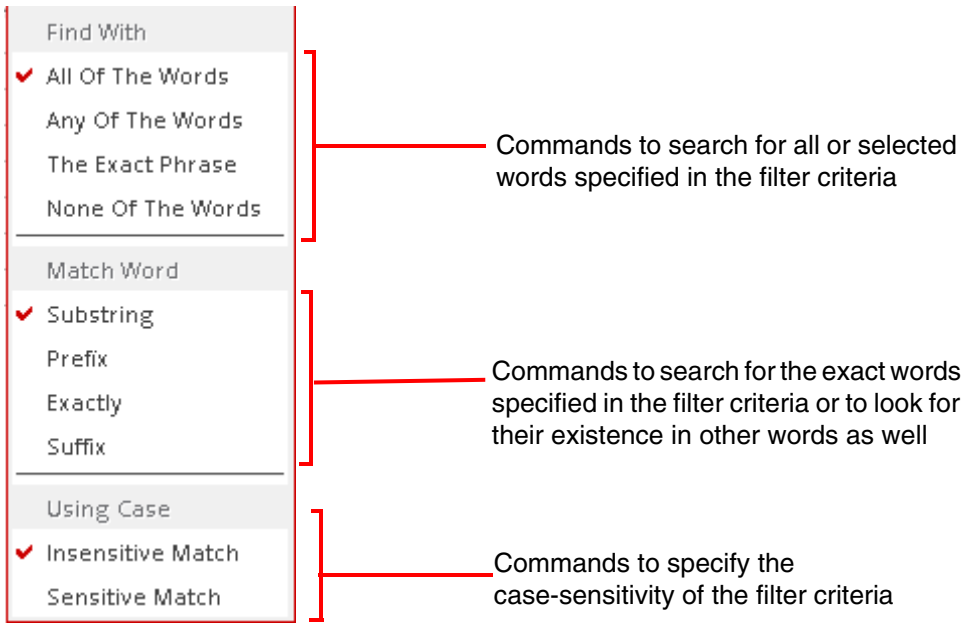
Important Points to Note

- Filters are supported only in the *Detail*, *Detail–Transpose*, *Checks/Asserts*, *Yield*, and *Optimization* results views.
- Filters set on multiple columns apply the `AND` operation.

  For example, to view all the results for which the value of current is greater than `5m`, you can select `Current` in the *Output* column, and specify `>5m` in the *Nominal* column on the *Results* tab.
- You can apply a compound criteria in a filter by using the operators given in the table below.

Operator	Use	Example
`> <`	Defines an exclusive range	`>x <y`
`=> <=`	Defines an inclusive range	`=> x <=y`
`^`	Defines an exclusive range	`^ x y`

- If you are not using comparison operators, the numerical values are compared like strings.
- You can make a filter more precise by using the commands in the right-click menu of a filter text field.



© 2023 Cadence Design Systems, Inc.

For example, to view the results for two outputs, `Gain` and `BW`, you can specify `Gain BW` in the filter field for the Output column. In addition, select *Any of The Words* from the right-click menu of the filter text field. Results for only these two outputs will be displayed.

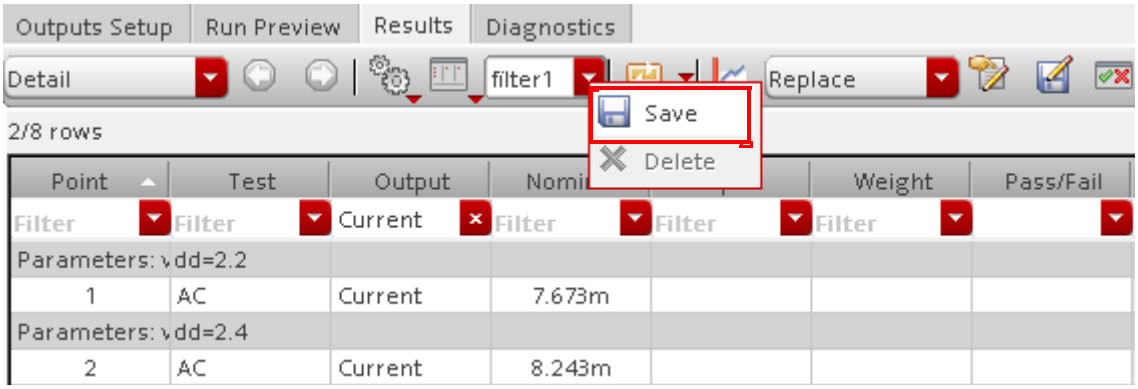
Saving Commonly Used Filters

ADE Assembler allows you to save commonly-used filters by name so that they can be again applied to the same or other similar results.

Another benefit is that, if you have saved the filters applied to the results columns, the filter settings are retained when you switch the result views.

To save a filter on the *Results* tab, do this:

1. Apply the required filter settings. You can apply settings in multiple columns.
2. Specify a name in the *Filter name* drop-down list on the Results toolbar.
3. Right-click the *Filter name* drop-down list and choose *Save*.



The filter criteria is saved with the given name in the drop-down list and can be used to filter the results using the same criteria.

Note: Filters are saved in the maestro view. Therefore, a filter saved in one view cannot be used in other views.

If a saved filter is not required, you can delete it. For this, select the name of the filter to be deleted from the *Filter name* drop-down list. Right-click the drop-down list and choose *Delete*.

© 2023 Cadence Design Systems, Inc.

Important Points to Note

- Filters applied on a column are saved in the filter history for that column. You can clear this history by selecting the *Clear Search History* command in the right-click menu of the *Filter* drop-down list for that column.
- You can hide the filters by using the *Hide Filters* command in the right-click menu of a column header on the *Results* tab. The hidden filters can be viewed again by selecting the *Show Filters* command in the right-click menu of a column header.
- Filters are not available in the Spec Summary of results.
- Filters are useful only for viewing selected results on the *Results* tab. They are not used while plotting results on graphs. The *Plot All* command plots graphs for all the results.
- By default, the filters are always shown on the *Results* tab. When the number of rows in the results is small, you can choose to hide the filters by setting the resultsFilteringColumnsList environment variable.

© 2023 Cadence Design Systems, Inc.

Plotting Results

You can plot the results that appear on the Results tab of the Outputs pane. The plots are displayed in a waveform window.

Important

As a result, You may observe the following in the structure of the psf directory where simulation results are saved for single point runs:

i) Similar to multiple point runs, the simulation results for a single point run are now also saved under the top level psf directory. That is, the results are saved in:
- `Interactive.<n>/psf/<test_name>/psf`, which is the top level psf directory
- `Interactive.<n>/1/<test_name>/psf`, which is the psf directory for the single point

ii) When you set context from Result Browser to the top level psf directory and evaluate a scalar expression in Calculator, the result is returned as a waveform object. However, when you set context to the results directory of the single point run, the result of a scalar expression is returned as a scalar value.

iii) When you plot results of a single point run, if no global variables or parameters were used, the trace legend of the graph shows the Design Point column. If global variables or parameters were used to run the simulation for that point, name of last variable or parameter in the global list is shown in the trace legend.

For more information, see the following topics:

- Plotting Across All Corners and Sweeps
- Plotting Across Corners
- Plotting Across Design Points
- Plotting Outputs for Selected Points
- Plotting Outputs from the Specified Plot Set
- Other Plotting Methods
- Plotting Templates
- Plotting Associated Signals

© 2023 Cadence Design Systems, Inc.

You need not wait for the simulation run to complete before plotting results. You can plot the results for completed data points and corners even when the simulation run is in progress.

Plotting Across All Points

To plot across all points, do the following:

1. Select the plot mode. For more information, see Selecting the Plot Mode on page 1125.
2. On the Results tab of the Outputs pane, click the button.

   The waveforms appear in a waveform window.

   Note: If you click the button after modifying measurement expressions or specifications in the Outputs Setup tab, the tool shows a message box and prompts you to confirm that you want to re-evaluate the results. Click *Yes* to update the results displayed in the Results tab based on the new or modified expressions and specifications and then plot the results. Click *No* to ignore the changes in expressions and specifications and plot the results.

© 2023 Cadence Design Systems, Inc.

Plotting Across All Corners and Sweeps

To plot waveforms across all corners and sweeps for a signal or waveform expression, do the following:

1. Select the plot mode. For more information, see "Selecting the Plot Mode" on page 1125.
2. Do one of the following:
   - Double-click a test name in the *Test* column or a signal or waveform expression name in the *Output* column of the Results tab.
3. Right-click any of the following on the Results tab and choose *Plot All*:
   - A test name in the *Test* column
   - A signal or waveform expression name in the *Output* column
   - In the *Nominal* column or in the column for a corner

   The waveforms for all the corners or sweep points are displayed in a waveform window.

Important

To consider any changes in outputs or specs after simulation, *Plot All* re-evaluates the outputs before plotting them. This may take some time depending on the number of outputs. If you want to avoid re-evaluation, set the `reevaluateBeforePlotAll` environment variable to `nil` before running *Plot All*.

Plotting Across Corners

To plot waveforms across all corners for a signal or waveform expression, do the following:

1. Select the plot mode. For more information, see "Selecting the Plot Mode" on page 1125.
2. Right-click any of the following on the Results tab and choose *Plot Across Corners*:
   - A test name in the *Test* column
   - A signal or waveform expression name in the *Output* column
   - In the *Nominal* column or in the column for a corner

   The waveforms are displayed in a waveform window.

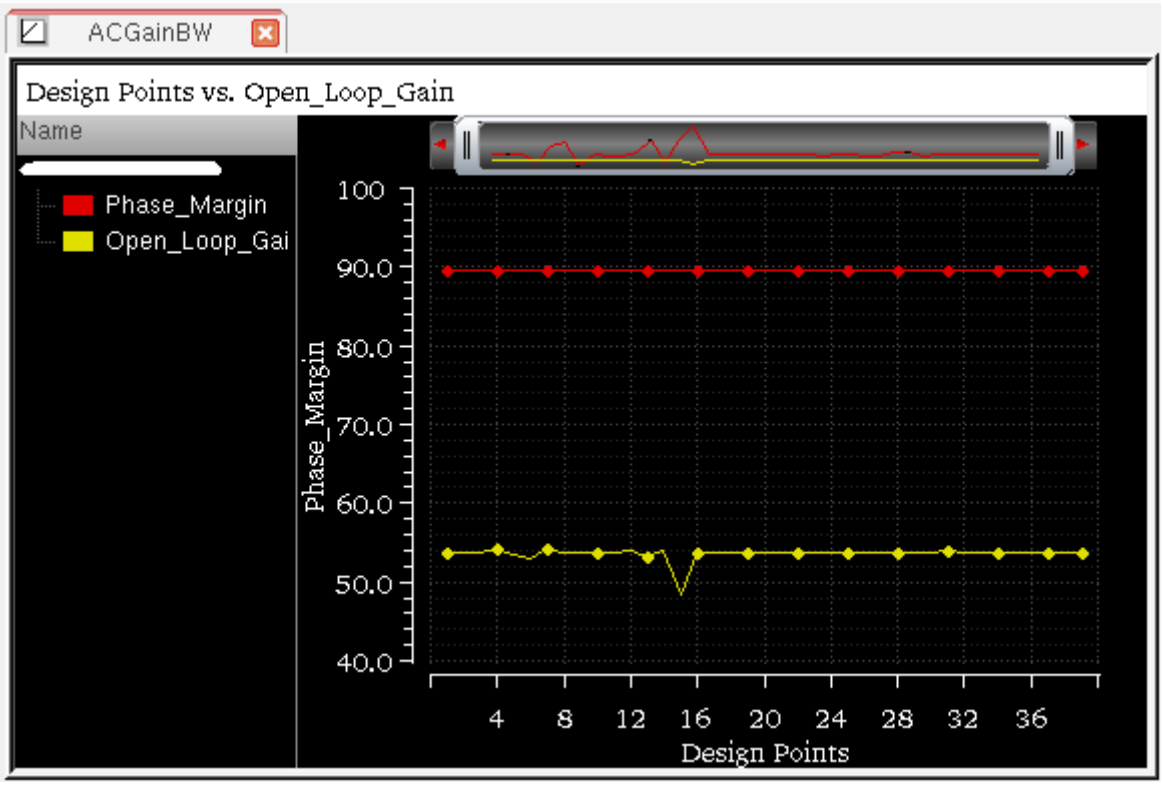
© 2023 Cadence Design Systems, Inc.

Plotting Across Design Points

To plot waveforms across all design points for a signal or waveform expression, do the following:

1. Select the plot mode. For more information, see "Selecting the Plot Mode" on page 1125.
2. On the Results tab, right-click the *Nominal* column or in the column for a corner and choose *Plot Across Design Points*.

   The waveforms displayed in a waveform window show results plotted across all design points, as shown below.



   In the above figure, graphs have been plotted after setting the `Append` plotting mode.

Plotting Outputs for Selected Points

To plot waveforms selectively, select the plot mode (see Selecting the Plot Mode) and then do one of the following:

- Right-click a signal (indicated by the icon) or a waveform expression in the *Nominal* column or in the column for a corner and choose *Plot*.

© 2023 Cadence Design Systems, Inc.

Note: You can use this command to plot signals for only ac, dc, and tran analyses. For other analyses, you need to create expressions to plot signals. For example, `vh('hb "/net1").`

To plot multiple signals or waveform expressions, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection) and click the next signal or waveform expression to add it to the selection set. Right-click and choose *Plot*.

Note: You cannot plot multiple signals or waveform expressions for Monte Carlo simulation results.

- Double-click a signal or waveform expression in the *Nominal* column or in the column for a corner.

  The waveforms for the selected signals and waveform expressions appear in a waveform window.

Plotting Outputs from the Specified Plot Set

To plot outputs from the specified plot set, do the following:

1. Select the plot mode. For more information, see Selecting the Plot Mode on page 1125.
2. Do one of the following on the Results tab of the Outputs pane:
   - To plot outputs across all corners and sweeps for a signal or waveform expression, right-click a test name in the *Test* column or a signal or waveform expression name in the *Output* column and choose *Plot Outputs*.
   - To plot outputs for a specific point, right-click a signal or waveform expression in in the *Nominal* column or in the column for a corner and choose *Plot Outputs*.
3. Select one of the following from the submenu that appears:

Transient	Plots the transient response for each node
AC	Plots the AC response for each node
Noise	Plots the squared noise voltage for each node
DC	Plots the DC sweep response for each node
Expressions	Plots the waveforms for expressions you define on the Outputs Setup tab of the Outputs pane.

© 2023 Cadence Design Systems, Inc.

The outputs that you specified for plotting appears in a plot window according to the plotting mode you selected.

Other Plotting Methods

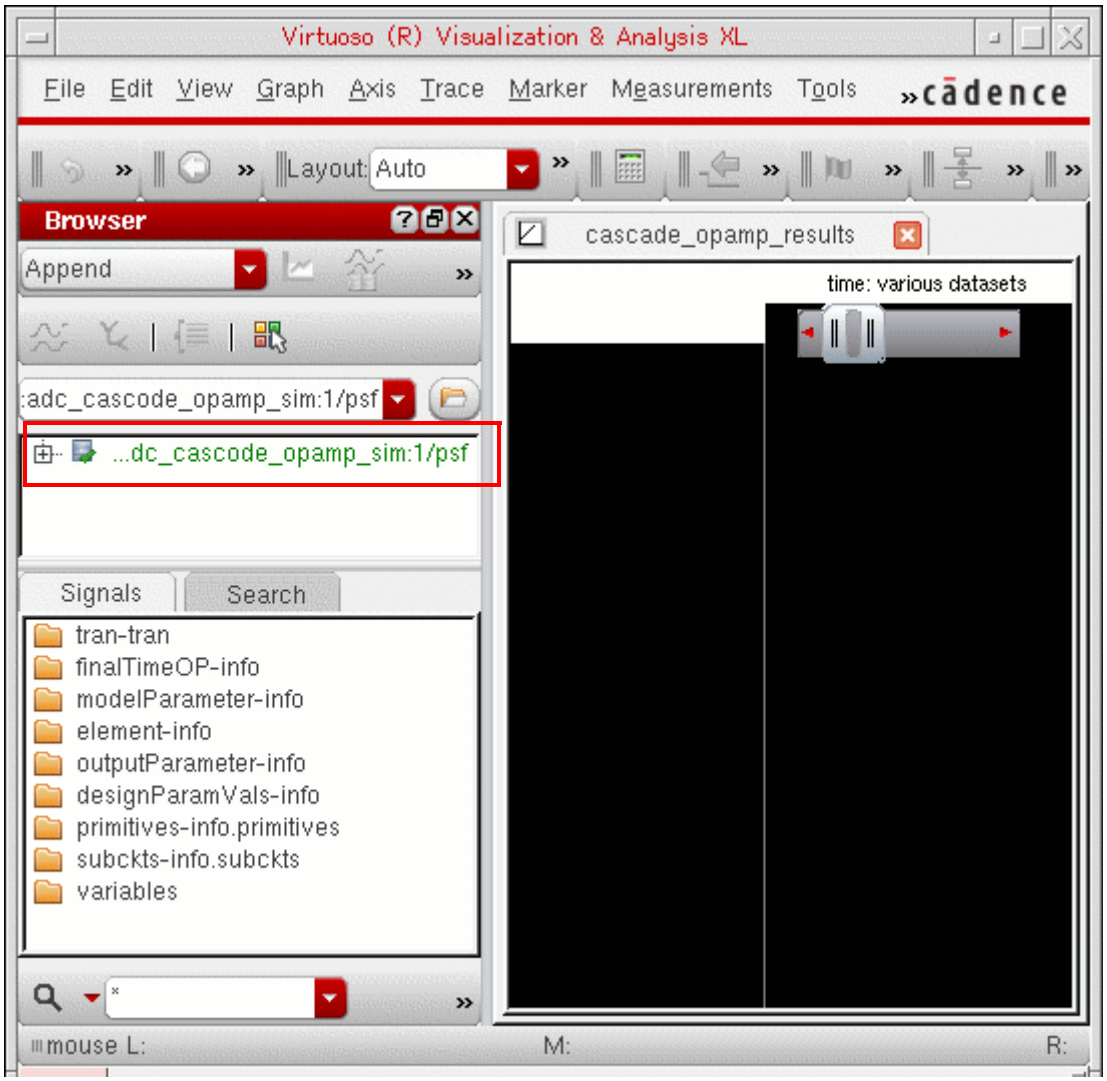
You can also use any of the following methods to plot selected simulation results in a waveform window:

- On the Results tab of the Outputs pane, right-click any of the following and choose *Open Results Browser* to load the selected results in the Results Browser:
  - Right-click a test name in the *Test* column or a signal or waveform expression name in the *Output* column to plot across all the corners for the signal or waveform expression.
  - Right-one or more signal or waveform expressions in the *Nominal* column to plot across the nominal corner.
  - Right-click one or more signal or waveform expressions in a corner column to plot across the corner.

  When you load a single result in the Results Browser, the Virtuoso Visualization and Analysis XL window is opened. The selected results are opened in the Results Browser

© 2023 Cadence Design Systems, Inc.

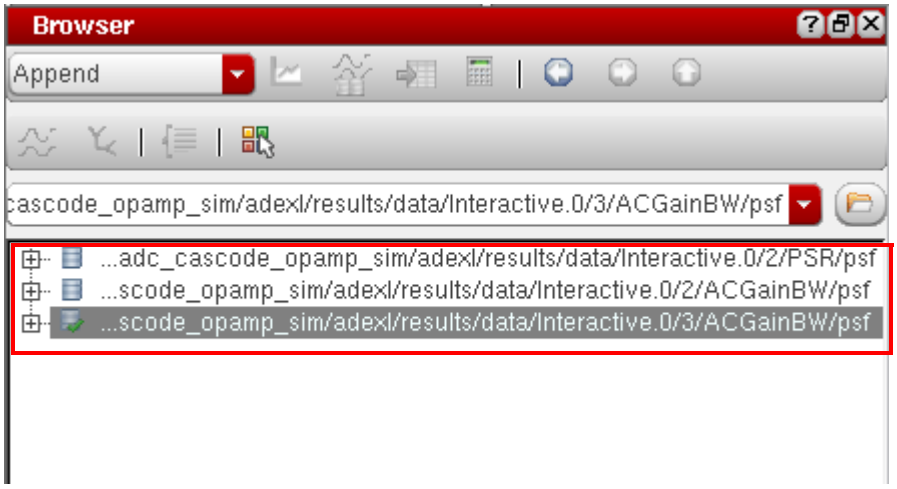
and context is set to the psf directory of these results. This is indicated by a green check mark on the name of results, as shown in the following figure.



Navigate to a node that contains waveform data and right-click to select a plot mode (such as *Append*, *Replace*, *New SubWin*, *New Win*). For more details, refer to *Virtuoso Visualization and Analysis User Guide*.

© 2023 Cadence Design Systems, Inc.

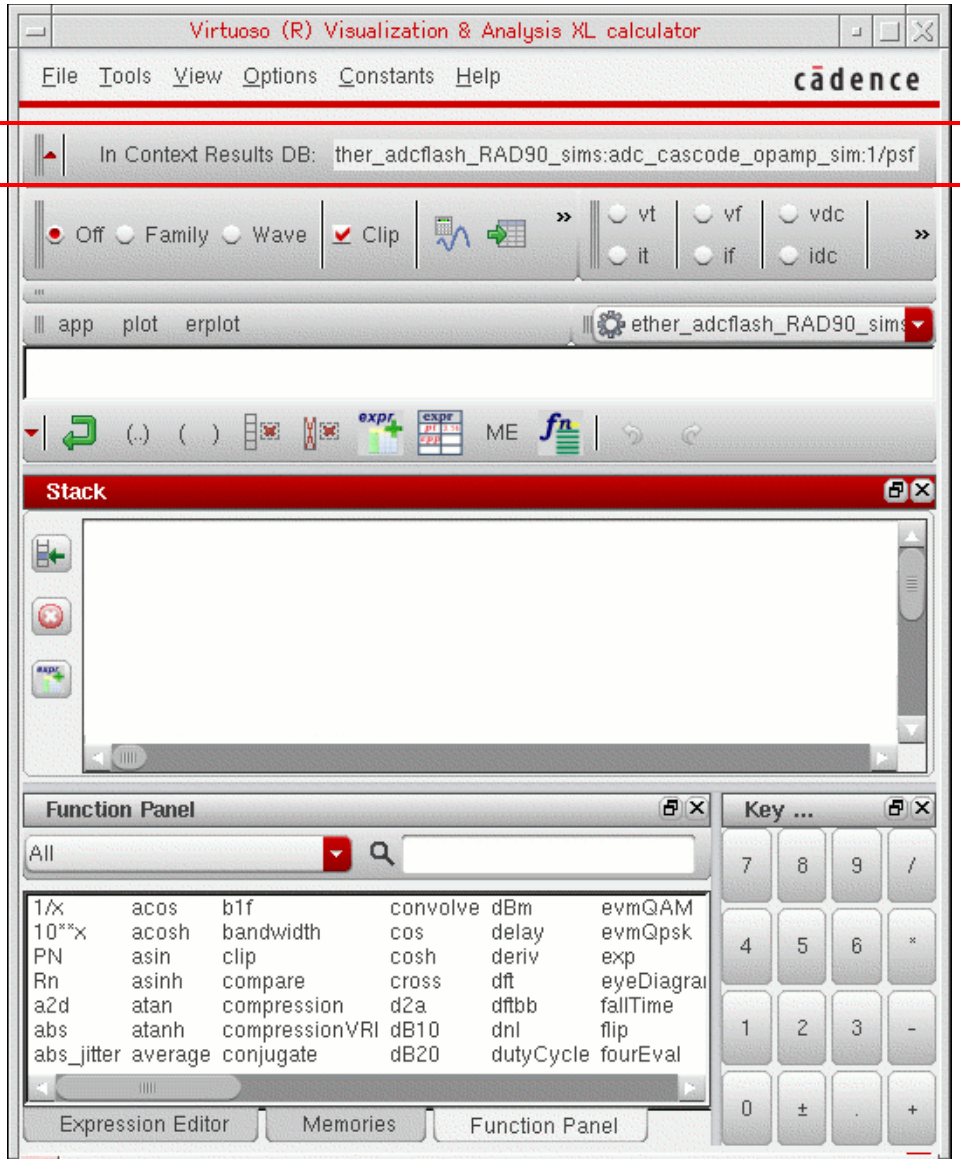
When you load multiple results in the Results Browser, the psf directories of all the points are opened in the sequence of selection. The context is set to the psf that was selected last, as shown in the following figure.



- On the Results tab of the Outputs pane, right-click any of the following and choose *Direct Plot* for a submenu of functions for plotting selected nets and terminals from your schematic.
  - Right-click a test name in the *Test* column or a signal or waveform expression name in the *Output* column to direct plot across all the corners for a signal or waveform expression.
  - Right-click a signal or waveform expression in the *Nominal* column to direct plot across the nominal corner.
  - Right-click a signal or waveform expression in a corner column to direct plot across the corner.
- On the Results tab of the Outputs pane, right-click a test name in the *Test* column or a signal or waveform expression name in the *Output* column or a single data point and choose *Open Calculator*. The Virtuoso Visualization and Analysis XL calculator window

© 2023 Cadence Design Systems, Inc.

is opened. The current context in this window is set to the psf directory of selected results data.



Now, you can create expressions using the data point and plot outputs from calculator.

See also

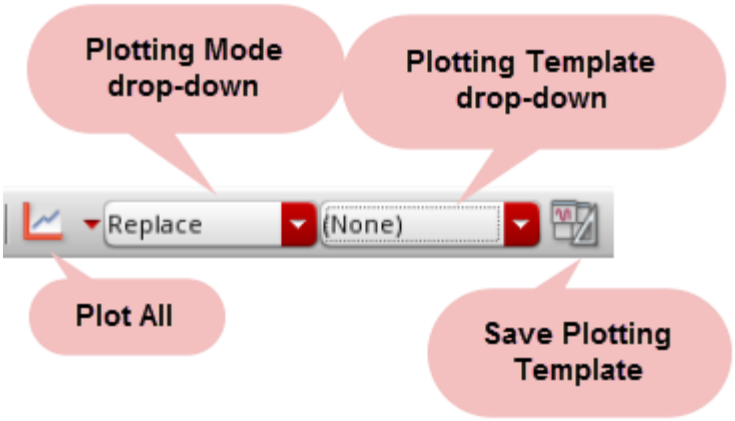
- Chapter 10, “Selecting Data to Save and Plot”
- Selecting the Plot Mode

© 2023 Cadence Design Systems, Inc.

Plotting Templates

You can create a plotting template in ADE Explorer and ADE Assembler to plot resultant waveforms, signals, or scalar values in a specified format in the Virtuoso Visualization and Analysis XL window.

The plotting template drop-down list is available on the toolbar in the *Results* tab, as shown in the figure below:



Note: The plotting template drop-down list is enabled only when the plotting mode is `Replace` or `New Win`. If the plotting mode is set as `Append` or `New SubWin`, this drop-down is disabled.

The plotting template drop-down list is displayed by default. However, you can set the following environment variable in CIW to disable this feature.

```
envSetVal("maestro.gui" "usePlottingTemplate" 'boolean nil)
```

To view the list of available plotting templates, open the drop-down list. You can hover the pointer on any template listed in this list to view the complete name and description of that template.

For more details, refer to Working with Plotting Templates in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

Distributed Plot Commands

The following table describes the distributed plot commands that are displayed when you click the arrow next to the icon in the toolbar of the *Results* tab of ADE Assembler.

Command	Description
Open Graph	Opens the Distributed Plot graph window.
Job Distribution	Opens the Job Policy Selection for Distributed Plot form. This form lets you select the job policy that you have set up earlier in the Job Policy form.
Show Process List	Displays information about the currently running processes in an xterm window.
Show Assistant	Opens the Distributed Plot assistant. The Distributed Plot assistant lets you view and manage various plotting jobs
Exit	Exits the distributed command menu.

These commands become unavailable if you disable distributed plot. The icon indicates that distributed plot is disabled.

Related Topics

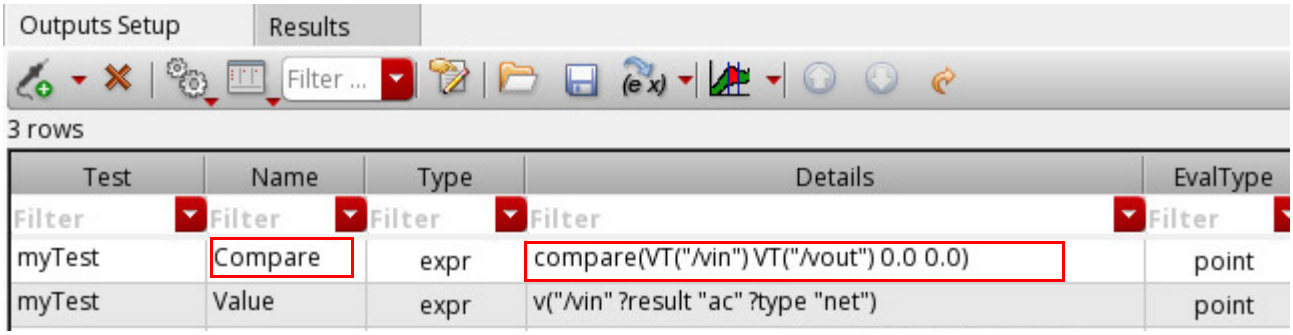
Distributed Plot

© 2023 Cadence Design Systems, Inc.

Plotting Associated Signals

You can plot all the signals used in an expression directly from the *Results* tab.

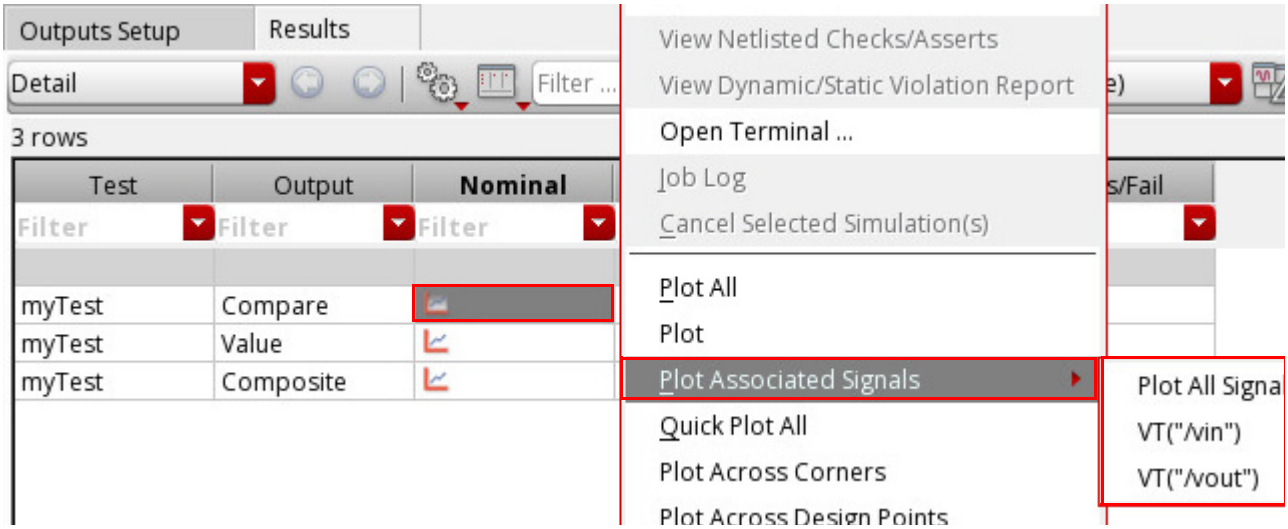
On the *Outputs Setup* tab, consider the `Compare` expression, which compares two waveform signals: `VT("/vin")` and `VT("/vout")`.



To plot the signals used in the expression, do the following:

1. Run the simulation to evaluate the expression.

   The results are displayed in the *Results* tab.



2. On the *Results* tab, right-click the *Nominal* column, and then choose one of the following commands from the context menu:

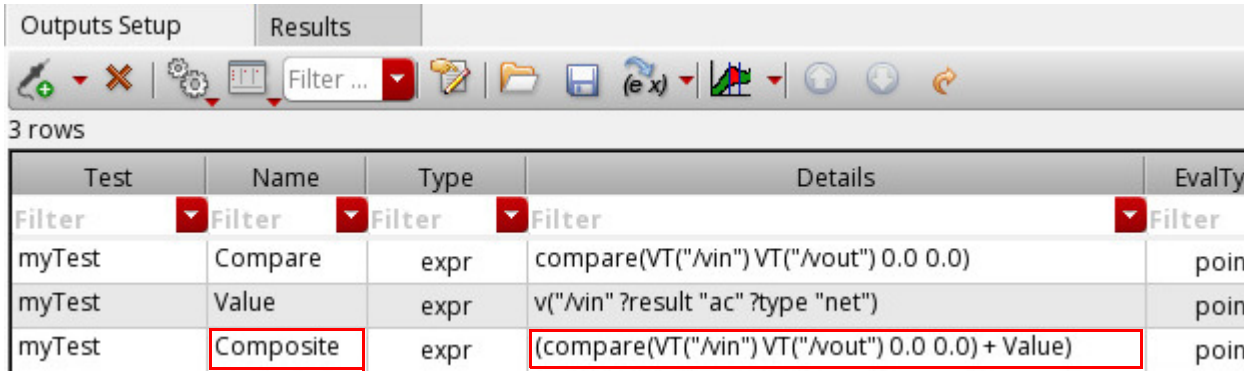
   - *Plot Associated Signals — Plot All Signals*: Plots all the associated signals used in the expression.
   - *Plot Associated Signals — <Signal 1>*: Plots the first signal used in the expression. For example, `VT("/vin")`.

© 2023 Cadence Design Systems, Inc.

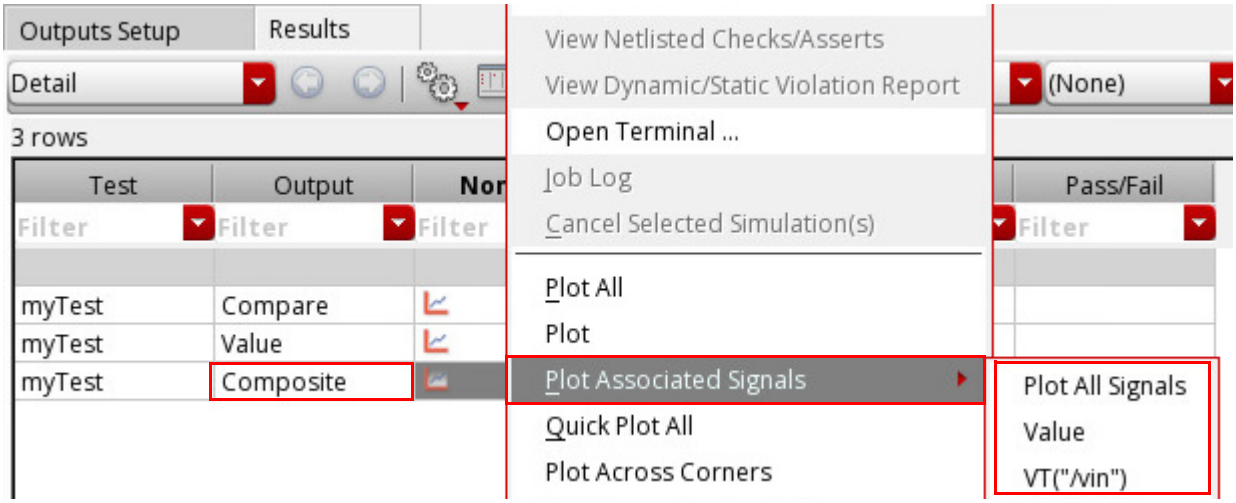
- *Plot Associated Signals—<Signal 2>*: To plot the second signal used in the expression. For example, `VT("/vout")`.

In addition to the associated signals, you can also plot the first-level dependent used in the expression.

Consider the `Composite` expression, as shown in the following figure:



If you plot the associated signals of the expression, the first-level dependent, `Value`, can also be plotted directly from the Results tab, as shown in the following figure.



Using Direct Plot

You can use the direct plotting feature to plot a function from a set of one or more nets or terminals that you select on your schematic.

© 2023 Cadence Design Systems, Inc.

Tip

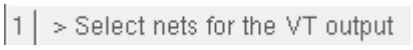
You need not wait for the simulation run to complete before using direct plotting. You can use direct plot for completed data points and corners even when the simulation run is in progress.

To use direct plot, do the following:

1. Select a plot mode. For more information, see Selecting the Plot Mode.
2. On the Results tab of the Outputs pane, right-click any of the following and choose *Direct Plot*.
   - Right-click a test name in the *Test* column or a signal or waveform expression name in the *Output* column to direct plot across all the corners for a signal or waveform expression.
   - Right-click a signal or waveform expression in the *Nominal* column to direct plot across the nominal corner.
   - Right-click a signal or waveform expression in a corner column to direct plot across that corner.

   A submenu of functions appears. Only those functions that apply to the current results are available for selection.
3. Select a function.

   If you have the prompt line turned on, you can read the prompt text for a hint about what you need to select.

4. In the schematic window, select one or more nets or terminals to plot using the function you selected.
5. Press *Esc* when you are done selecting.

   The selected nets or terminals are plotted in a waveform window according to the function you selected. If a waveform window was not already open, then one appears. If a waveform window was already open, then it appears in the foreground.

   Note: If the *Add To Outputs* check box is selected in the Direct Plot Form, expressions are added for the plotted results in the Outputs Setup tab of the Outputs pane. Re-evaluate the expressions to display the updated results in the Results tab of the Outputs pane. For more information about re-evaluating expressions, see Re-Evaluation of Expressions and Specifications.

© 2023 Cadence Design Systems, Inc.

Here is a table of direct plot functions.

Direct Plot Functions

Function	Description
Main Form	Opens the Direct Plot Form for specifying plotting mode, analysis, function, and modifier (see "Using the Direct Plot Main Form" in the *Virtuoso ADE Explorer User Guide*).
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; you are prompted for the signals
Transient Difference	Two signals subtracted (sig1- sig2) and plotted; you are prompted for two signals
AC Magnitude	AC voltage or current gain waveform
AC dB10	The magnitude on a decibel scale 10log(V1)
AC dB20	The magnitude of selected signals on a decibel scale 20log(V1)
AC Phase	AC voltage or current phase waveform
AC Magnitude & Phase	The db20 gain and phase of selected signals simultaneously
AC Gain & Phase	The differences between two magnitudes and two phases; you are prompted for two signals 20log(V2)-20log(V1) which is equivalent to 20log(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
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

© 2023 Cadence Design Systems, Inc.

Direct Plot Functions, *continued*

Function	Description
Noise Figure	Noise figure of selected signals according to the input, output, and source resistance
DC	DC sweep voltage or current waveform

Specifying the Default Direct Plot Mode

To specify the default direct plot mode for all tests, do the following:

➡ In the *Direct Plot* group box, select the plot mode from the *Plotting Mode* drop-down list.

For more information about plot modes, see Table 19-2 on page 1126.

Return to main procedure.

© 2023 Cadence Design Systems, Inc.

Performing Quick Plotting

Tip

Read a blog related to this feature at Virtuosity: Can I Speed up My Plots?

When the ADE Assembler setup includes multiple sweep or corner points, or the transient simulations are time consuming, plotting output waveforms through the *Plot All* option could consume significant time and memory. In this case, you can use the *Quick Plot* option to plot outputs faster and with much lesser memory usage (data size less than 1 percent of PSF XL format). The quick plot is faster because it uses a reduced set of data points, whereas the Plot All option generates a full view that covers all data points. The quick plot compresses data, and therefore, saves memory.

However, quick plots have lower resolution as compared to the full view that is generated using the Plot All option. When you send the quick plot waveforms to Calculator, the full plot data is loaded so that the calculations are not affected. In addition, if you perform any operations on quick plot signals using Results Browser, such as `Diff`, the output waveform is also of low resolution.

Note: The quick plot option is available only when the simulation run is complete and the output results are generated. Currently, this option works only for the transient signals and waveform expressions and is available only for the `Detail` and `Detail-Transpose` results views. In addition, quick plot can be generated for all signals and expressions except for histograms.

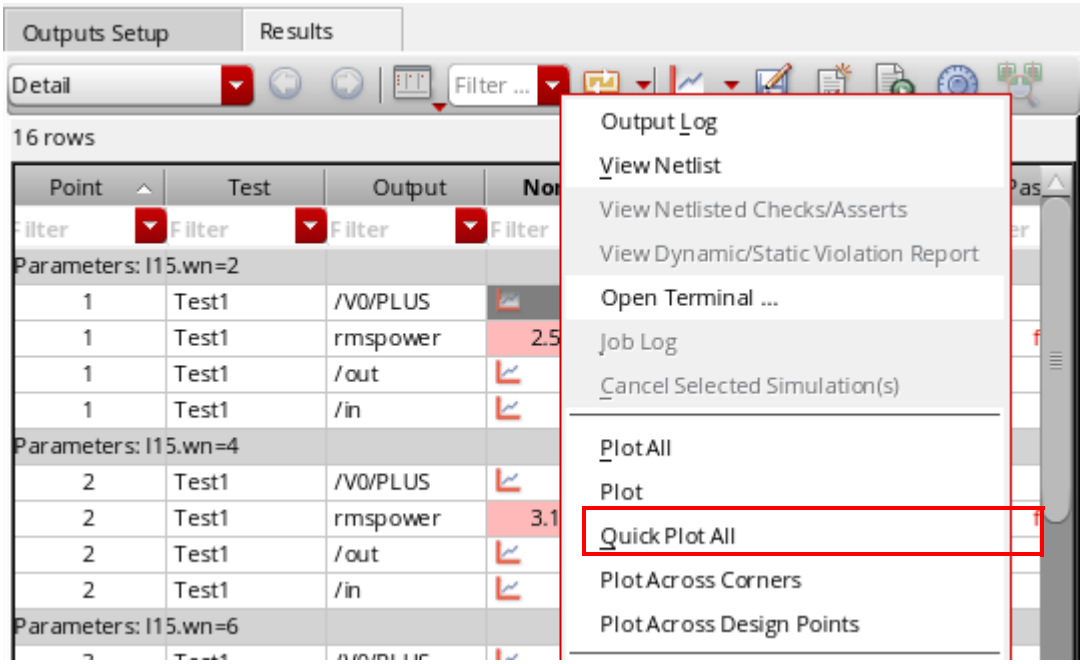
Note: If you perform the quick plotting of scalar signals, the full view plot is generated.

To generate the quick plot for a selected signal or expression:

- Right-click the waveform icon for a signal or waveform expression output and choose *Quick Plot All*.

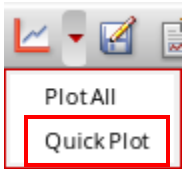
© 2023 Cadence Design Systems, Inc.

A quick plot is generated in the Virtuoso Visualization and Analysis XL window for the selected output signal.



To generate the quick plot for all the signals and expressions:

➡ In the *Plot All* drop-down list on the *Results* tab, click *Quick Plot*.



The quick plot waveforms for all the signal outputs are generated in the ViVA graph window.

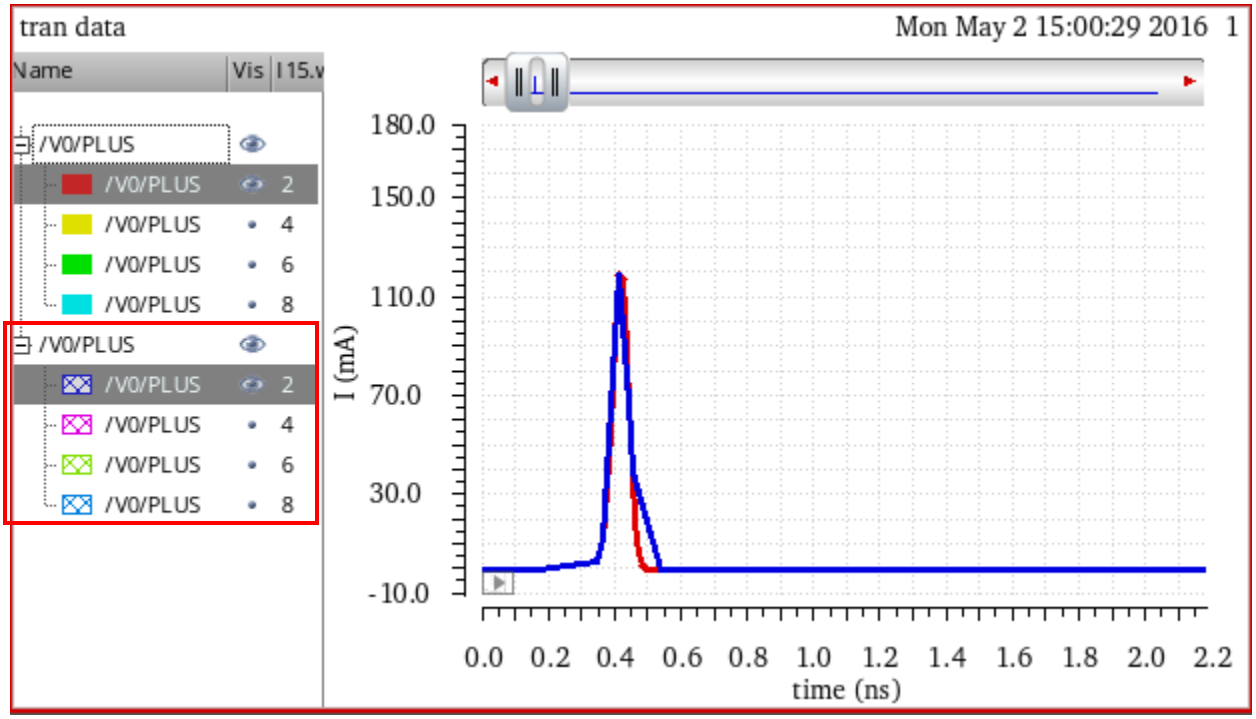
Important

The quick plots are generated only when the *Save Quick Plot Data* option is selected on the ADE Assembler Printing/Plotting Options form, which appears when you choose *Results – Printing/Plotting Options*. If this option is disabled, an error message is displayed in CIW. For more information, see Specifying Whether to Save Quick Plot Data.

The figure below shows the waveforms from normal plot (full view) displayed in red and quick plot displayed in blue for a current signal, `/V0/PLUS`. The quick plot waveform includes fewer

© 2023 Cadence Design Systems, Inc.

points as compared to the full view. When the full view and quick plot are displayed in the same window in append mode, you can zoom into the plots to compare them and see the differences between them. In addition, notice that the legend of the quick plot is different from the full view, which helps differentiating between the two plots.



The equivalent full data is loaded when you send the quick plot waveform data to Calculator. The loaded waveform is a leaf, for example:

```
leafValue( IT("/V0/PLUS") "I15.wn" 2 )
```

In the Results Browser, the signals are displayed in the order they are computed and may not match the order in which they are viewed in the full view data.

The saved quick plot data can be loaded into ViVA graph and is located at the location below:

```
<Interactive>/wavedb/<test>.vqv
```

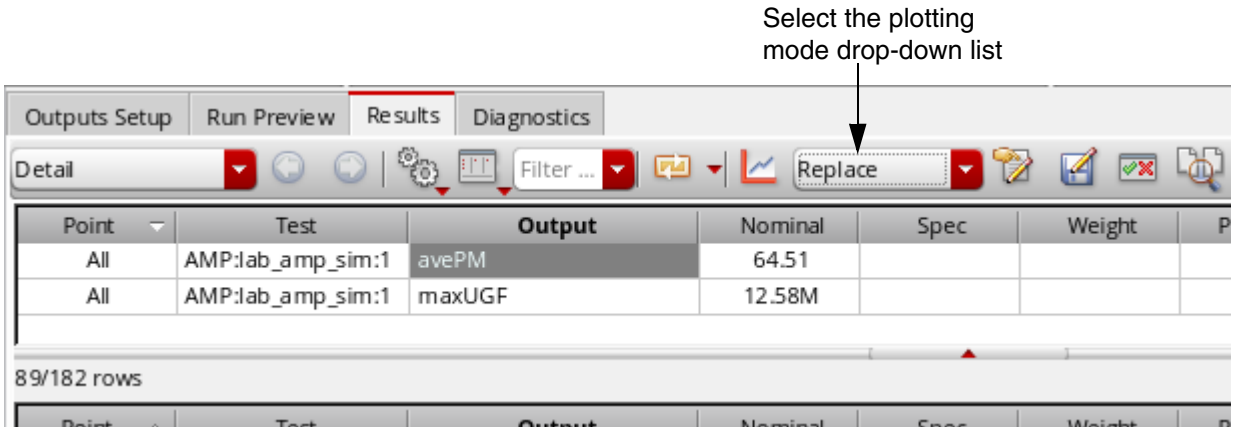

© 2023 Cadence Design Systems, Inc.

You can also generate a full view from the quick plot data. To do this, right-click the quick plot waveform and choose *Quick Plot Expand to – New Window/New Subwindow/Current Window*.

Selecting the Plot Mode

On the Results tab of the Outputs pane, do the following:

➤ From the *Select the plotting mode* drop-down list on the Results tab, select one of the following plot modes:



© 2023 Cadence Design Systems, Inc.

Table 19-2 Plot Modes

Plot Mode	Description
Append	A new trace is appended to the active waveform window if the test name and X-axis match. ***Important Points To Note*** ■ Traces for different tests are always plotted in different waveform windows tabs in the Virtuoso Visualization and Analysis XL window. ■ If the test name for the active waveform window tab is different from the test name of the new trace to be plotted, ADE Assembler looks for an existing tab (from left) for the same test. If found, the new trace is appended to that window tab. Otherwise, a new window tab is created. ■ The axes units of the new trace must match with the axes of the existing traces. If the units of X-axis differ, the new trace is plotted in a new subwindow in the existing window tab. If the units of Y-axis differ, a new Y-axis is created in the existing subwindow.
Replace	New traces replace the existing traces in the active waveform window. If the active waveform window has multiple subwindows, all the subwindows are deleted and new traces are plotted.
New SubWin	New traces are plotted in a new subwindow in the active waveform window
New Win	New traces are plotted in a new waveform window

Note: These plot modes can also be set from the Direct Plot, Calculator, and Results Browser windows.

Setting Default Plotting Options for All Tests

You can use the ADE Assembler Plotting/Printing Options form to set the default printing and plotting options for all tests. These options will be used:

© 2023 Cadence Design Systems, Inc.

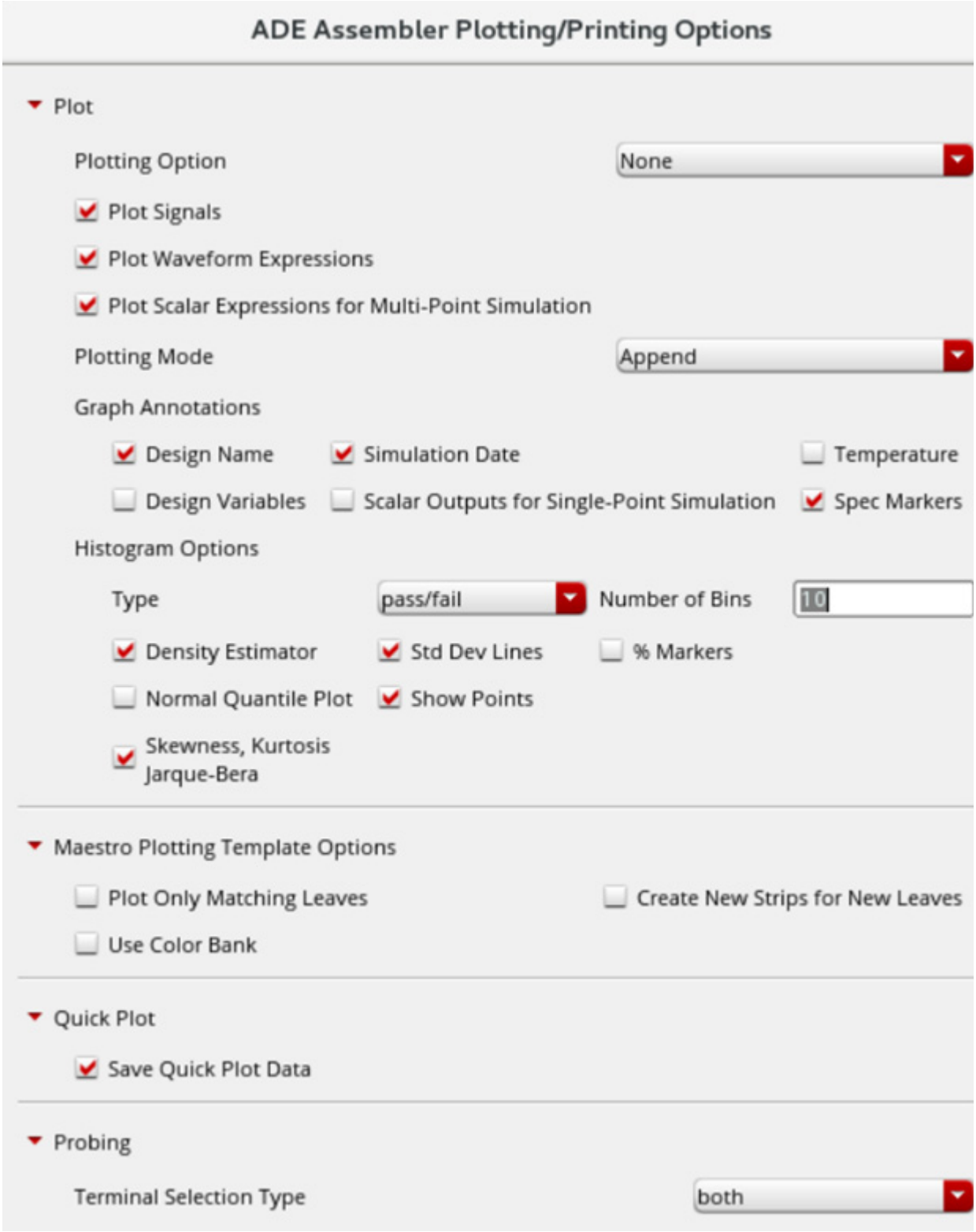
- For automatic plotting of results after a simulation run is complete
- When you click the *Plot all waveforms* button on the Results tab

Note: You can use the Setting Plotting Options form to override the default printing and plotting options for individual tests. For more information, see Searching for Conditional Results.

1. To open the ADE Assembler Plotting/Printing Options form, do one of the following:
   - Choose *Options – Plotting/Printing*.
   - Click the button on the *Outputs Setup* tab or the *Results* tab of the Outputs pane.

© 2023 Cadence Design Systems, Inc.

The ADE Assembler Plotting/Printing Options form appears.



© 2023 Cadence Design Systems, Inc.

2. Specify plotting options.

   See the following topics for assistance:

   - Specifying the Default Plotting Option for All Tests
   - Specifying the Outputs that Need to be Plotted
   - Specifying the Default Plotting Mode for All Tests
   - Specifying Annotations for the Graph Window
   - Specifying When to Plot Direct Plot Results
   - Specifying When to Print Results

3. Click *OK*.

Specifying the Default Plotting Option for All Tests

You can specify the default plotting option for all tests. These options will be used for the following:

- Automatic plotting of results after a simulation run is complete
- When you click the *Plot all waveforms* button on the Results tab

To specify the default plotting option for all tests, do the following in the ADE Assembler Printing/Plotting Options form:

➡ In the *Plot* group box, select the plotting option from the *Plotting Option* drop-down list.

Select	To
None	Disable automatic plotting of results after the simulation run. This is the default option.
Auto	Automatically plot outputs after the simulation run. For every subsequent simulation run, a new graph replaces the existing graph. You can choose to append the new graph to the existing graph of the previous simulation run or plot it in a new window using the *Plotting Mode* drop-down list. When this option is selected, any customization done in the Virtuoso Visualization and Analysis graph windows that are currently open, such as setting up traces, colors, or zoom levels, are not reused. Every time a graph is plotted, the default settings are used.

© 2023 Cadence Design Systems, Inc.

Select	To
Refresh	Plot the results by updating the existing graphs in the Virtuoso Visualization and Analysis XL graph window that is already open. Use this option when you want to save and reapply graph and trace settings on the plots that you want to review across different simulation runs. **Note:** Currently, this feature is supported only in the `Single Run, Sweeps, and Corners` run mode. For more details on how the graphs are refreshed, refer to Refreshing Graphs. This mode ignores the following options in the ADE Assembler Printing/Plotting Options form: ■ Plotting mode selected in the *Plotting Mode* drop-down list ■ *Plot Signals*, *Plot Waveform Expressions* and *Plot Scalar Expressions for Multi-Point Simulation* options ■ *Graph annotation* options

You can configure the default plotting option by using the `plotType` environment variable.

Return to main procedure.

Specifying the Outputs that Need to be Plotted

To specify the outputs that need to be plotted automatically after the simulation finishes, do the following in the *Plot* group box:

Select	To
Plot Signals	To plot signals.
Plot Waveform Expressions	To plot expressions that evaluate to waveforms.
Plot Scalar Expressions for Multi-Point Simulation	To plot scalar expressions for a multi-point simulation.

Return to main procedure.

© 2023 Cadence Design Systems, Inc.

Specifying the Default Plotting Mode for All Tests

To specify the default plotting mode for all tests, do the following:

➡ In the *Plot* group box, select the plot mode from the *Plotting Mode* drop-down list.

For more information about plot modes, see Table 19-2 on page 1126.

Return to main procedure.

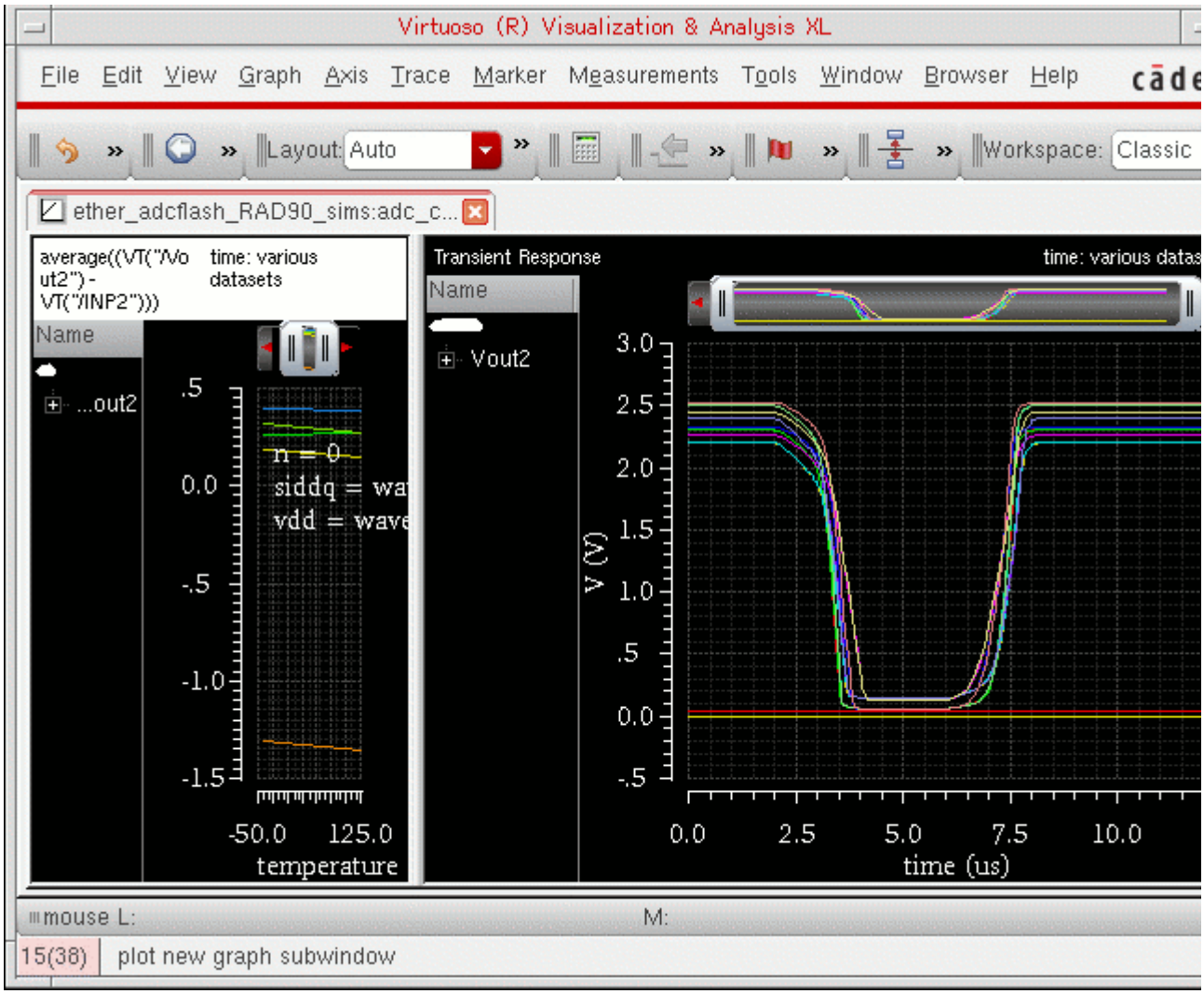
Specifying Annotations for the Graph Window

To specify annotations for the graph window, do the following:

1. Select one or more of the following *Annotations* check boxes:
   - *Design Name* — Displays design name in the title banner of the waveform window.
   - *Simulation Date* — Displays date and time of simulation in the title banner of the waveform window.
   - *Temperature* — Displays simulation temperature in the plot area of the waveform window.
   - *Design Variables* — Displays design variables and their values in the plot area of the waveform window.
   - *Scalar Outputs for Single-Point Simulation* — Annotates scalar outputs for a single-point simulation.
   - Spec Markers — Displays spec markers in the plot area of the waveform window. For more details about the spec markers, refer to Displaying Spec Markers on Graphs on page 1151.
2. Click *Apply*.

© 2023 Cadence Design Systems, Inc.

Here is a waveform window with all annotations applied:



Return to main procedure.

Specifying When to Plot Direct Plot Results

To specify when to plot results you select from your schematic when you use the *Results – Direct Plot* submenu, do the following:

1. Select one of the following *Direct Plots Done After* options:
   - *Each Selection* – Plot results after each item you select on the schematic.
   - *All Selections Are Made* – Plot results after you select all items and press *Esc* to end selection mode.

© 2023 Cadence Design Systems, Inc.

2. Click *Apply*.

Return to main procedure.

Specifying When to Print Results

To specify when to print results to the Results Display Window, do the following:

1. Select one of the following *Print After* options:
   - *Each Selection* – Print results after each item you select on the schematic.
   - *All Selections Are Made* – Print results after you select all items and press *Esc* to end selection mode.
2. Click *Apply*.

Return to main procedure.

Enabling the Distributed Plot Service

To enable the distributed plot service:

1. Select the *Enable Distributed Plot* check box.
2. (Optional) In the *Linger Time (seconds)* field, specify the time in seconds for which the distributed plot process waits before exiting when there is no active plotting window is visible. The default value is `60` seconds. It is recommended not to set the linger time to less than `30` seconds.
3. Click *Apply*.

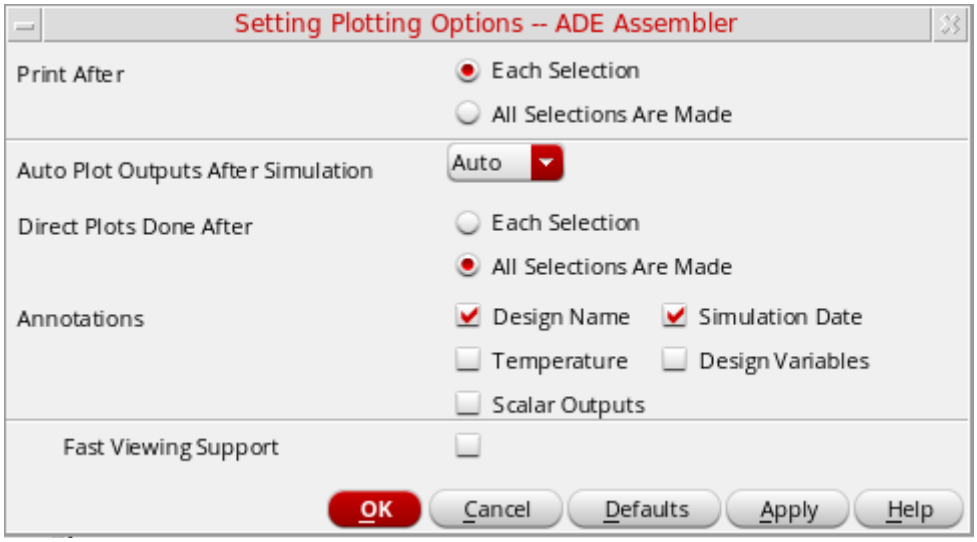
Setting Plotting Options for Specific Tests

You can use the Setting Plotting Options form to set various options related to printing and plotting for specific tests. The options you specify in this form override the options specified in the Setting Plotting Options - ADE Assembler form. For more information about the Setting Plotting Options - ADE Assembler form, see Setting Default Plotting Options for All Tests.

1. On the Outputs Setup tab of the Outputs pane, right-click a test name and choose *Printing/Plotting Options*.

© 2023 Cadence Design Systems, Inc.

The Setting Plotting Options form appears.



2. Specify plotting options.

   See the following topics for assistance:

   - Specifying When to Print Results
   - Specifying Whether to Plot Automatically After Simulation
   - Specifying When to Plot Direct Plot Results
   - Specifying Annotations for the Graph Window
   - Enabling Fast Waveform Viewing Format for PSF Output

3. When you are finished specifying options, click *OK*.

Specifying When to Print Results

To specify when to print results to the Results Display Window, do the following:

1. Select one of the following *Print After* options:

   - *Each Selection* – Print results after each item you select on the schematic.
   - *All Selections Are Made* – Print results after you select all items and press *Esc* to end selection mode.

2. Click *Apply*.

Return to main procedure.

© 2023 Cadence Design Systems, Inc.

Specifying Whether to Plot Automatically After Simulation

To specify that you want to plot outputs automatically after the simulation finishes, do the following:

1. Select the *Auto Plot Outputs After Simulation* check box.
2. Click *Apply*.

To specify that you do not want to plot outputs automatically after the simulation finishes, do the following:

1. Deselect the *Auto Plot Outputs After Simulation* check box.
2. Click *Apply*.

Return to main procedure.

Specifying When to Plot Direct Plot Results

To specify when to plot results you select from your schematic when you use the *Results – Direct Plot* submenu, do the following:

1. Select one of the following *Direct Plots Done After* options:
   - *Each Selection* – Plot results after each item you select on the schematic.
   - *All Selections Are Made* – Plot results after you select all items and press *Esc* to end selection mode.
2. Click *Apply*.

Return to main procedure.

Enabling Fast Waveform Viewing Format for PSF Output

Using the PSF output in the fast waveform viewing format, Virtuoso Visualization and Analysis can render extremely large datasets (where signals have a large number of data points, for example 10 million) within seconds.

To enable the fast waveform viewing format for PSF output, do the following:

- Select the *Fast Viewing Support* check box.

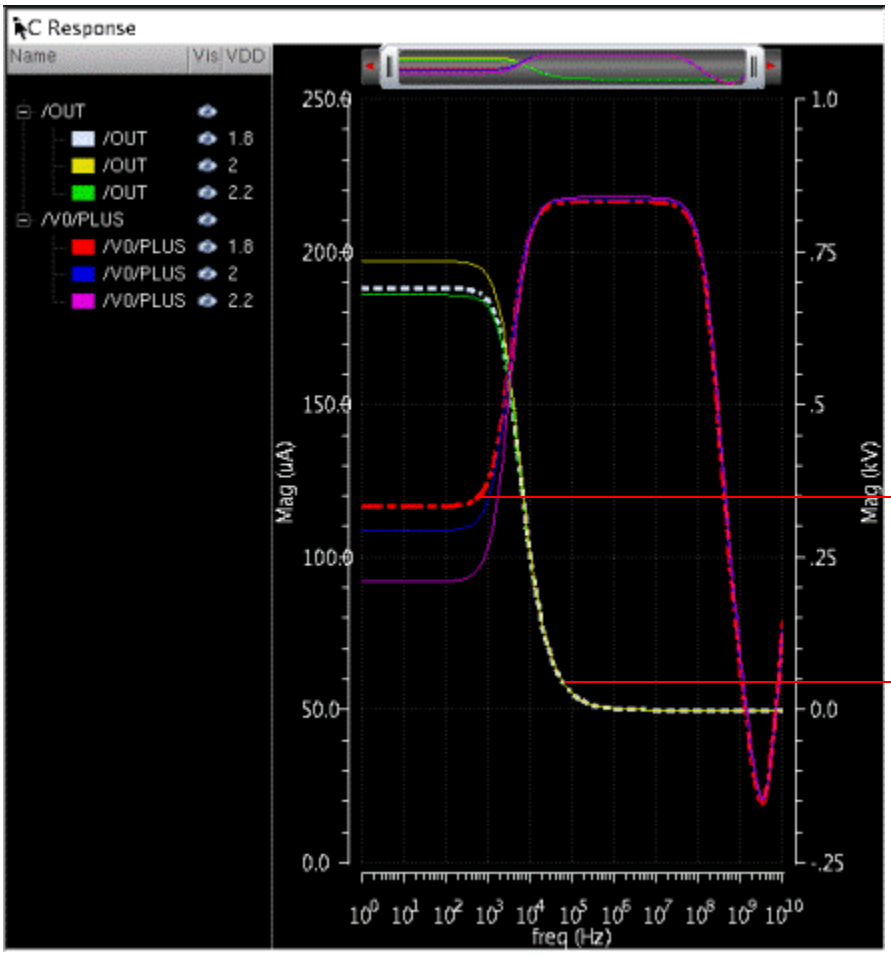
Return to main procedure.

© 2023 Cadence Design Systems, Inc.

Refreshing Graphs

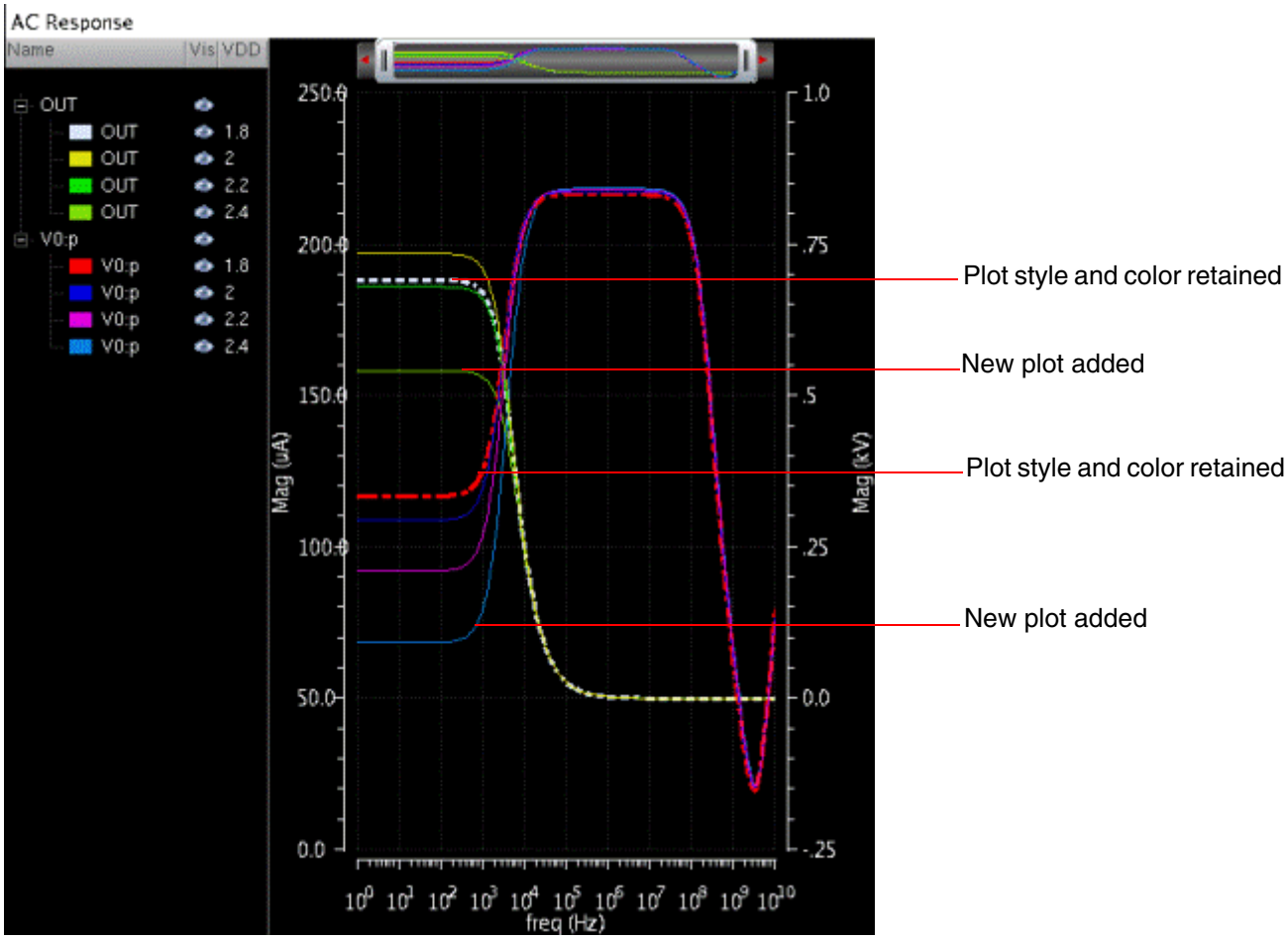
Graphs can be refreshed by using the `Refresh` plotting option in the ADE Assembler Printing/Plotting Options form. This option updates already open graphs with the new simulation data and retains graph and trace settings.

You can use the `Refresh` plotting option to review graphs across different simulation runs. For example, for a test, `ACGainBW`, you want to run multiple simulations with varying values of the global variable *`VDD`*. In the first run, you sweep *`VDD`* for three values 1.8, 2.0, and 2.2. After the simulation results are plotted in the graph, you customize two plots, as shown in the figure below.



© 2023 Cadence Design Systems, Inc.

Now, do not close this graph and set *Refresh* in the *Plotting Option* field. In the next run, sweep `VDD` for four values: `1.8`, `2.0`, `2.2`, and `2.4`. After the simulation, the graph appears as shown in the following figure.



Important Points to Note

- Traces for `VDD` = `1.8`, `2.0`, and `2.2` are updated in the same graph and trace settings are retained.
- New trace for `VDD` = `2.4` is added to the same graph.
- You can use *File – Reload* to reload the plotted graphs based on the results from a new in-context results directory selected in the Results Browser. If you rerun the simulation even without changing the corner settings, Reload plots traces from all the corners in the graph. To know more, see Reloading Graph When Opened From Within ADE Assembler in the Virtuoso Visualization and Analysis XL User Guide.

© 2023 Cadence Design Systems, Inc.

- When you set the context to a new results directory after running a simulation and plotting results, using Reload will plot traces from all the corners of the new context results directory in the graph window. To know more, see Changing In-Context Results Directory in the Virtuoso Visualization and Analysis XL User Guide.

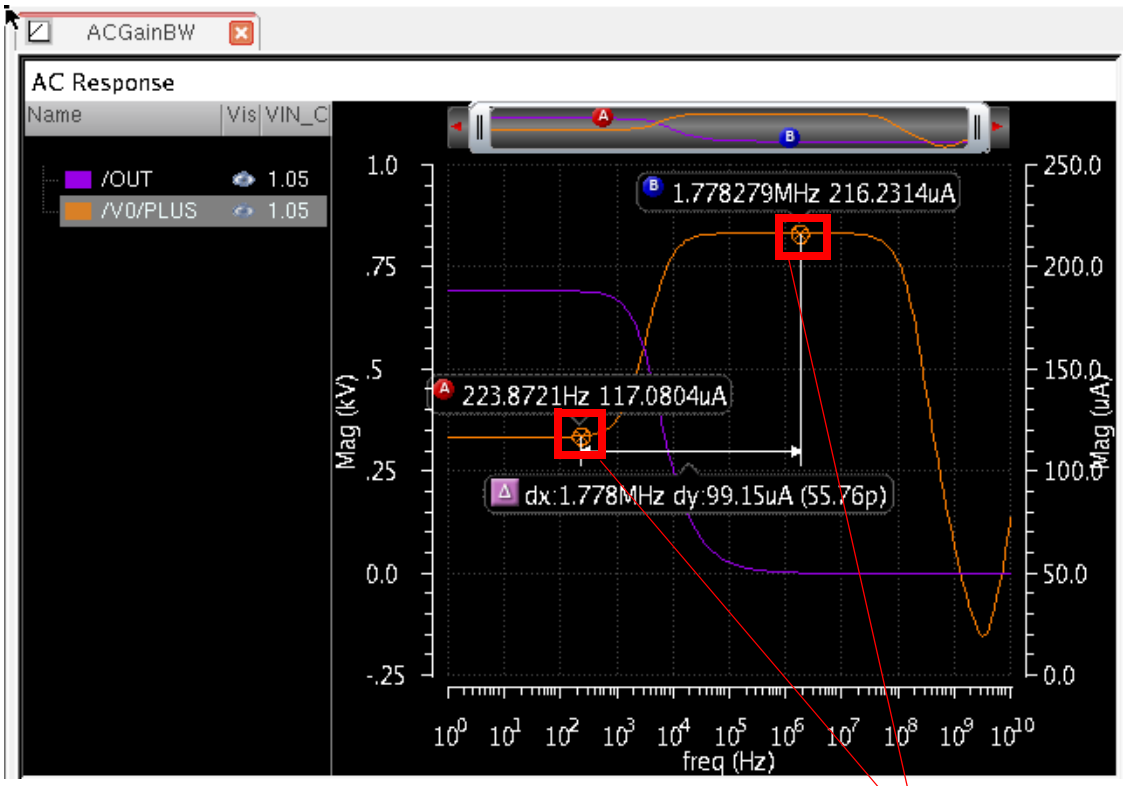
Graph Settings Retained During Refresh Plotting

When the *Plotting Option* field is set to `Refresh`, the graphs retain the following settings across simulation runs:

- Trace color, type, style, width, or symbols
- Visibility status of graphs
- Axes settings
- Pan and zoom settings
- Graph layout
- Strip layout
- Markers and marker locations

© 2023 Cadence Design Systems, Inc.

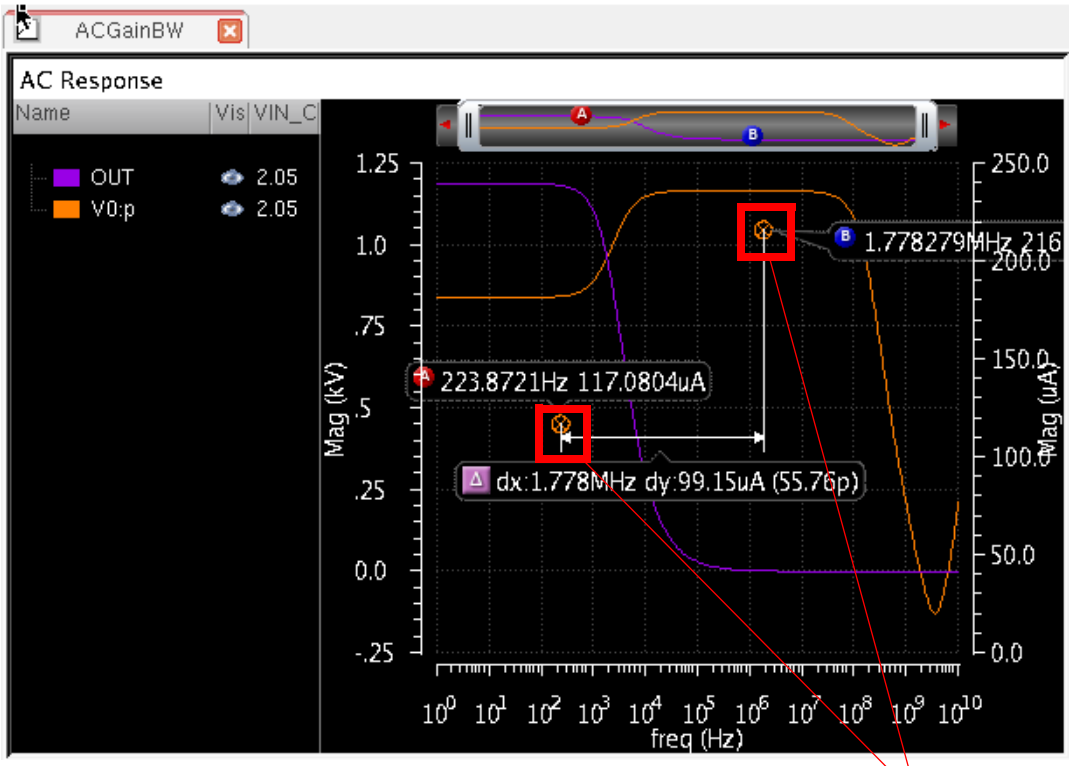
The following example shows how markers are retained with refresh plotting. Add an AB delta marker on the `V0/PLUS` trace in your graph, as shown in the following figure.



Markers applied on the plots

Now, if you change the values of variables and run the simulation again, the refreshed graph is plotted for the new simulation results, as shown in the figure below.

© 2023 Cadence Design Systems, Inc.



The A and B markers remain displayed at their original positions and are not connected with the trace in the refreshed plot.

Important

If you save graphs as part of an ADE Assembler state, all graph settings specified above are retained and when you reload the state, the trace settings are retrained. However, if you save the graph (in a .grf file) from standalone graph window, these settings are not saved.

Graph Settings Not Retained During Refresh Plotting

The following graph settings are not retained:

- Swapping of sweep variable on the X-axis
- Addition of a new graph window

© 2023 Cadence Design Systems, Inc.

- Any plot added to the graph in any of the following ways:
  - By using the Direct Plot main form
  - By using the Plot or Plot Across Corners commands on the Results tab
  - By plotting directly from the Results Browser
  - By plotting from the Calculator. For example, by plotting analog to digital, digital to analog, eye diagram, or spectrum measurements.
  - By adding outputs from a different history

  These graphs are not refreshed because any plot that is not on the Outputs tab of ADE Assembler is deleted after the graph is refreshed. To retain these additional plots and their settings, you can send the plot from the graph to the ADE Assembler *Outputs* tab by using the *Send to ADE* command on the shortcut menu of the graph. For more details, refer to Sending Traces to ADE in *Virtuoso Visualization and Analysis XL User Guide*.

The following sections describe how the graphs are updated with refresh plotting in different scenarios:

- Refresh Plotting with Varying Tests
- Refresh Plotting with Varying Analyses
- Refresh Plotting with Varying Output Setup
- Refresh Plotting with Varying Sweep Variables
- Refresh Plotting with Varying Corners

Refresh Plotting with Varying Tests

The following table summarizes how graphs are refreshed when tests are varied across different simulation runs:

© 2023 Cadence Design Systems, Inc.

Table 19-3 Effect of Test Variations on Graph Settings

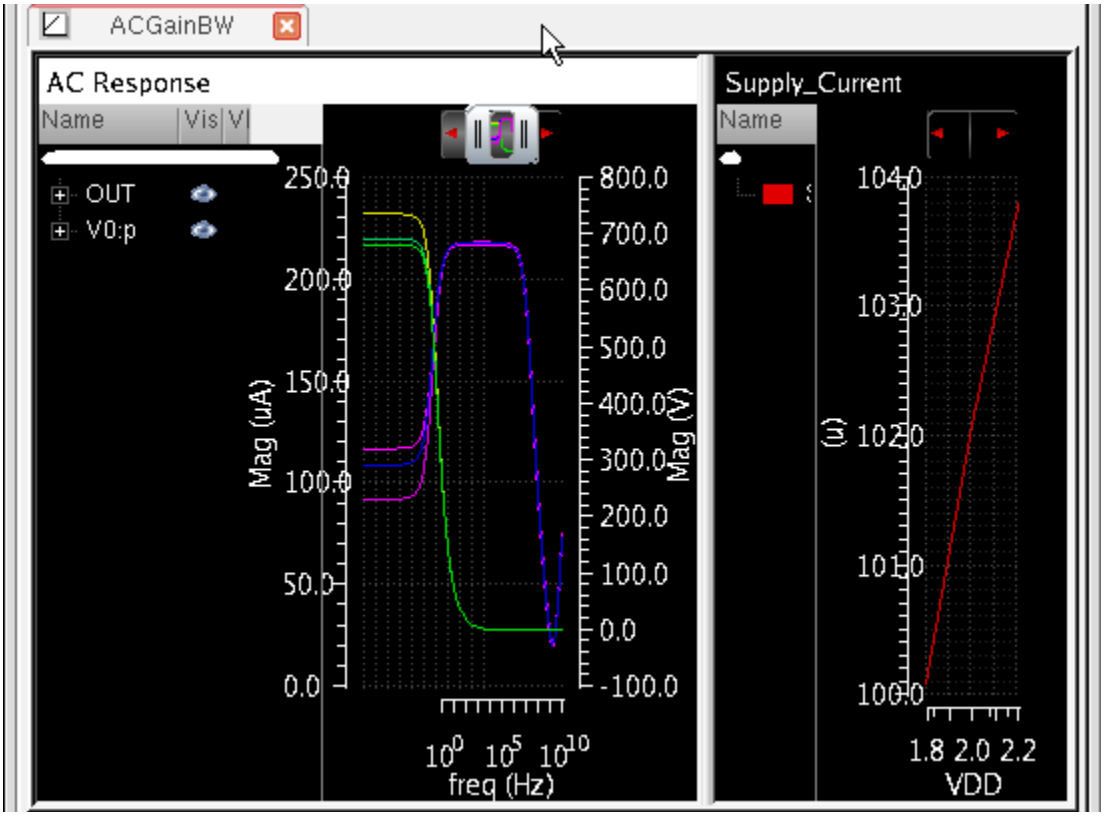
Test Variation	Effect on Graph Settings
Add or enable a test	■ A new graph window is added to display the plots of the newly added/enabled test. ■ Graph windows for the already existing tests are retained. Their graphs are updated with the new simulation data and trace settings are retained.
Change the test for which simulation is run	■ A new graph window is added to display the plots of the new/changed test. ■ Graph windows displayed in the previous run are retained.
Delete or disable a test	■ Graph window for the test that is deleted or disabled is retained, but their graphs are not updated.

The following example shows how graphs are updated when a new test is added:

When you run simulation with a single test, the graph appears as shown in the following figures.

© 2023 Cadence Design Systems, Inc.

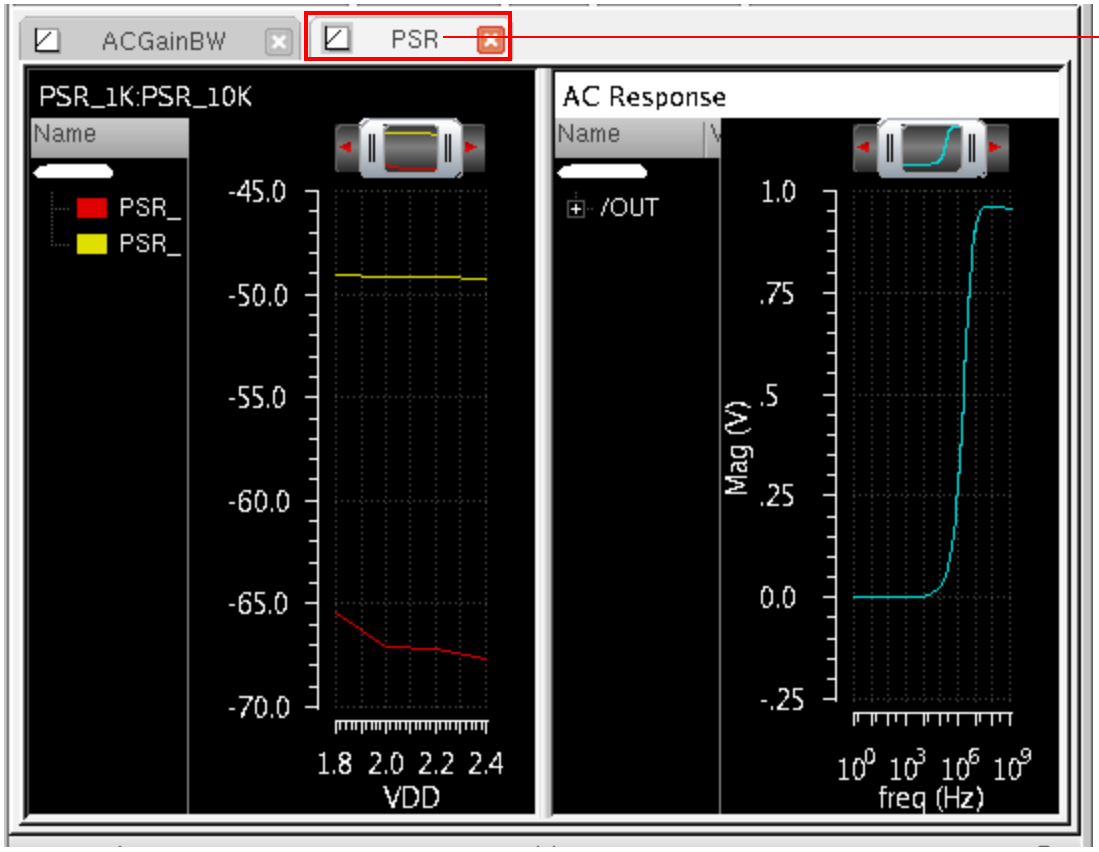
Figure 19-7 Graphs Plotted for a Simulation Run for a Single Test



Now, if you add one more test, `PSR`, and run simulation, a new graph window is added, as shown in the following figure.

© 2023 Cadence Design Systems, Inc.

Figure 19-8 Graphs Plotted for Simulation Run for an Additional Test



Refresh Plotting with Varying Analyses

The following table summarizes how graphs are refreshed when analyses are varied across different simulation runs (assuming that the test remains same):

Table 19-4 Effect of Analyses Variations on Graph Settings

Analyses Variation	Effect on Graph Settings
Add or enable an analysis	■ A new subwindow is added to the graph to display the plots of the newly added or enabled analysis. ■ Graphs for the analysis that are common across the two runs are updated with the new simulation data and any trace settings are retained.
Delete or disable an analysis	■ Subwindows related to the analysis that has been deleted or disabled after the previous simulation run are removed from the graph.

© 2023 Cadence Design Systems, Inc.

Refresh Plotting with Varying Output Setup

The following table summarizes how graphs are refreshed when outputs are varied across different simulation runs:

Table 19-5 Effect of Changes in Output Variations on Graph Settings

Output Variation	Effect on Graph Settings
Add or enable an output	■ Newly added or enabled output is added in a new subwindow. ■ Existing graphs are updated with new simulation data and their graph and trace settings are retained.
Delete or disable an output	■ The plot for the deleted or disabled output is removed. ■ Other graphs are updated with new simulation data and their graph and trace settings are retained.

Note: The Refresh plotting option does not update the graphs plotted for MATLAB measurements.

Refresh Plotting with Varying Sweep Variables

If you vary sweeps across simulation runs, the traces that use common sweep values are updated with new simulation data. For new sweep values, new traces are added to the same graph. Traces for unmatched sweep values are removed.

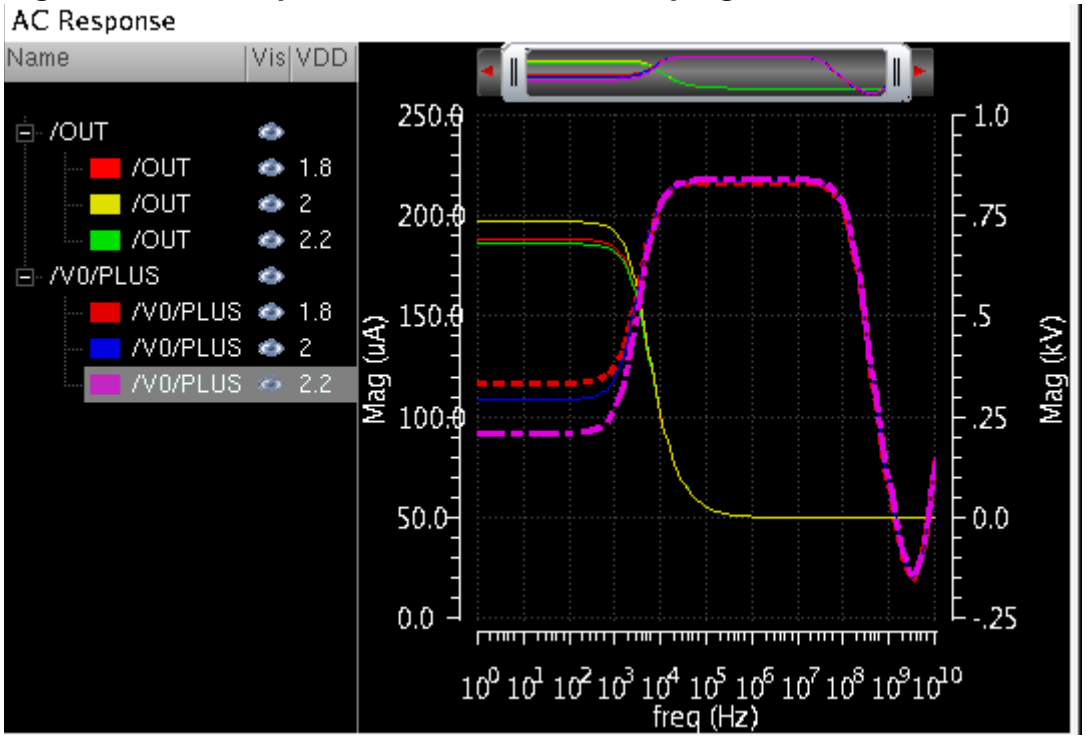
For example, set the variables as shown below.

Figure 19-9 Variables Used for Simulation

© 2023 Cadence Design Systems, Inc.

The graph plotted in this case includes plots for all three sweep values for *VDD* as shown below.

Figure 19-10 Graphs Generated after Sweeping Variable VDD



For the next simulation run, change the sweep values for *VDD* to `2.0:0.2:2.6`. Note the change in values of *VDD*, as shown below (mismatch values are underlined).

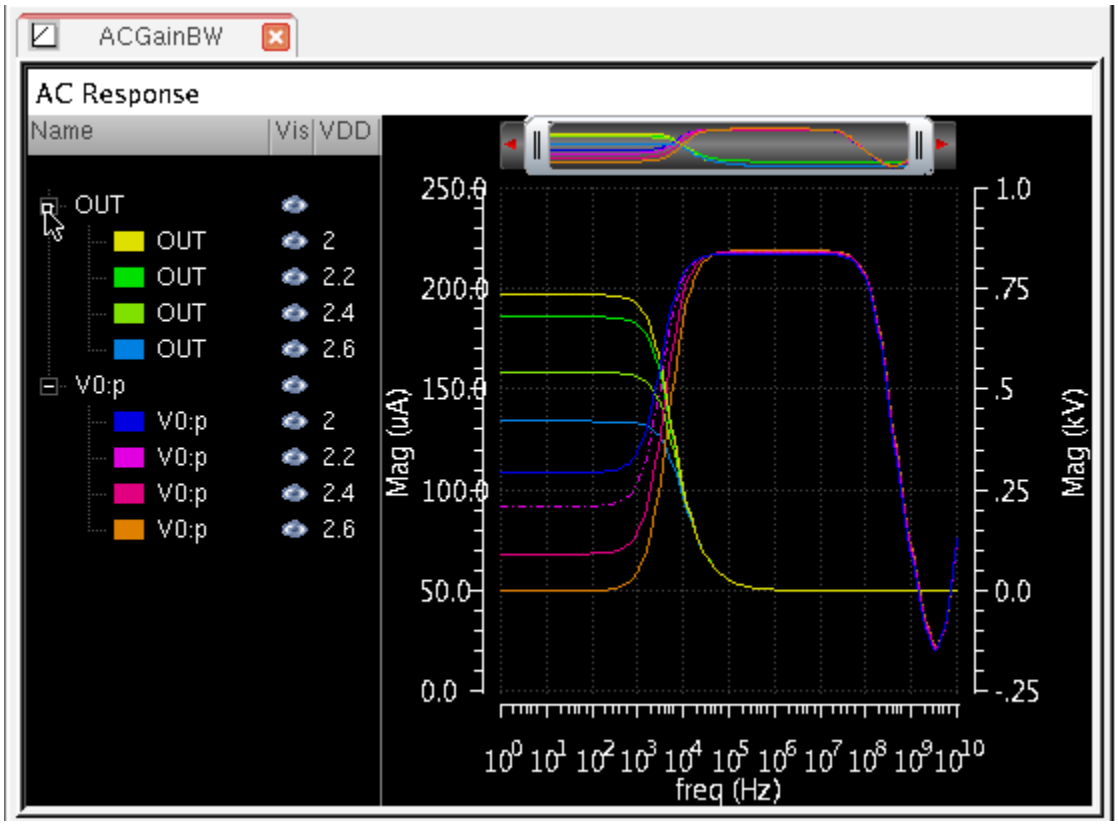
Values of *VDD* in the first run: `1.8`, `2.0`, and `2.2`

Values of *VDD* in the second run: `2.0`, `2.2`, `2.4`, and `2.6`

With the Refresh plotting option, the traces generated for the matching values of *VDD*, that is, when *VDD* is `2.0` and `2.2`, are updated with new simulation data. The trace that was earlier plotted for *VDD*=1.8 is deleted. New traces are added for *VDD*= `2.4` and `2.6`.

© 2023 Cadence Design Systems, Inc.

Figure 19-11 Graph settings Saved for the Plots with Common Sweep Values



However, there are some exceptions to this. In some cases, the traces plotted for common sweep values of the swept variables are not refreshed. These cases are listed below:

- Change from a single run to the sweep run and vice-versa.
- Change in the number of swept variables. For example, in the first simulation run, you sweep variable x, and in the subsequent runs, you sweep variables x and y. In this case, traces plotted for x are not refreshed.

Refresh Plotting with Varying Corners

Note: The following scenarios are based on the assumption that the test name remains same because only the traces for same test are plotted in the same waveform window. Traces from different tests are plotted in different waveform windows.

If you vary corners across simulation runs, traces generated for the common set of corners are updated with the new simulation data. For other corners, new traces are added in the same graph. Traces for unmatched corners are deleted.

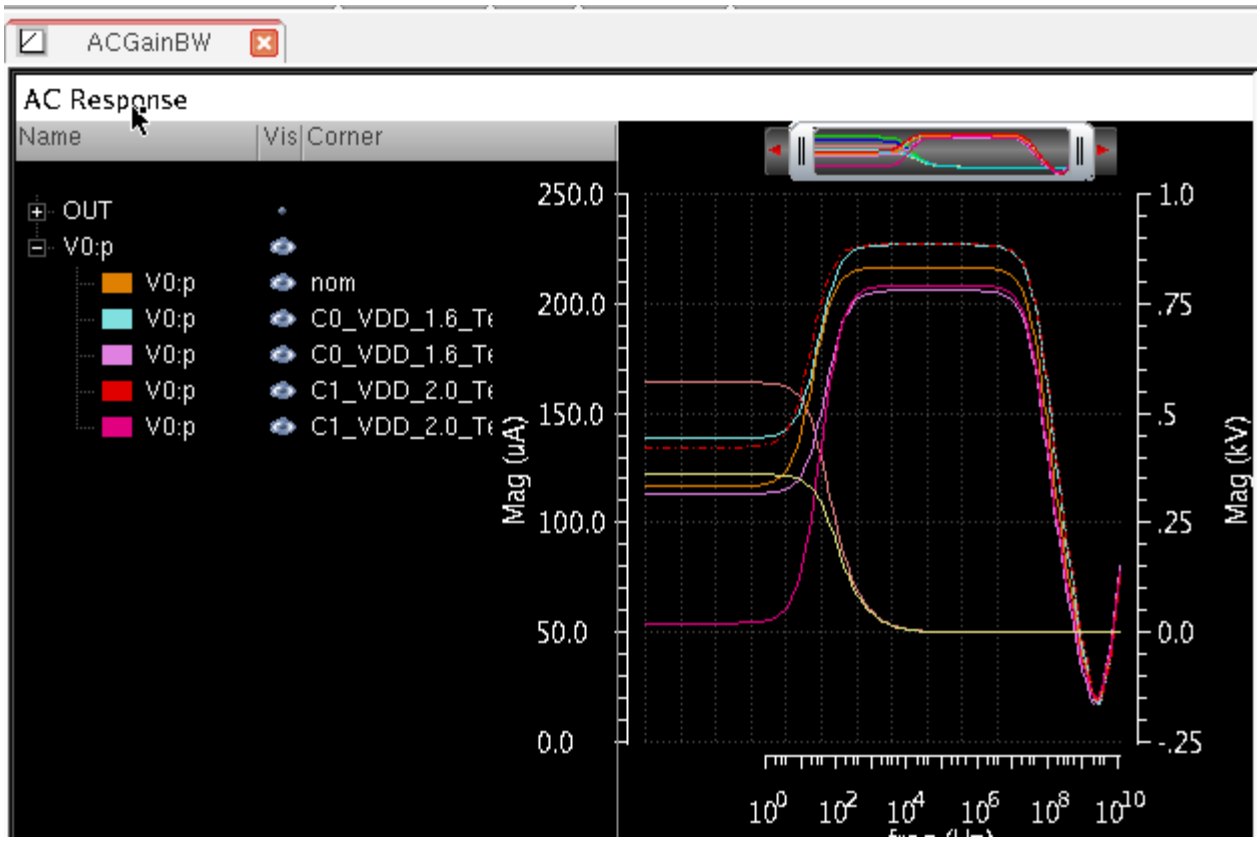
© 2023 Cadence Design Systems, Inc.

The following example shows are graphs are updated with different corners across three different runs:

Run 1:

Run a simulation for three corners: nominal, `C0`, and `C1`. Both `C0` and `C1` sweep temperature for two values, `-25` and `75`, and use a common section from the model file, but vary the value of *`VDD`*. The graph is plotted as shown below.

Figure 19-12 Graphs Plotted for Corners: nominal, C0, and C1

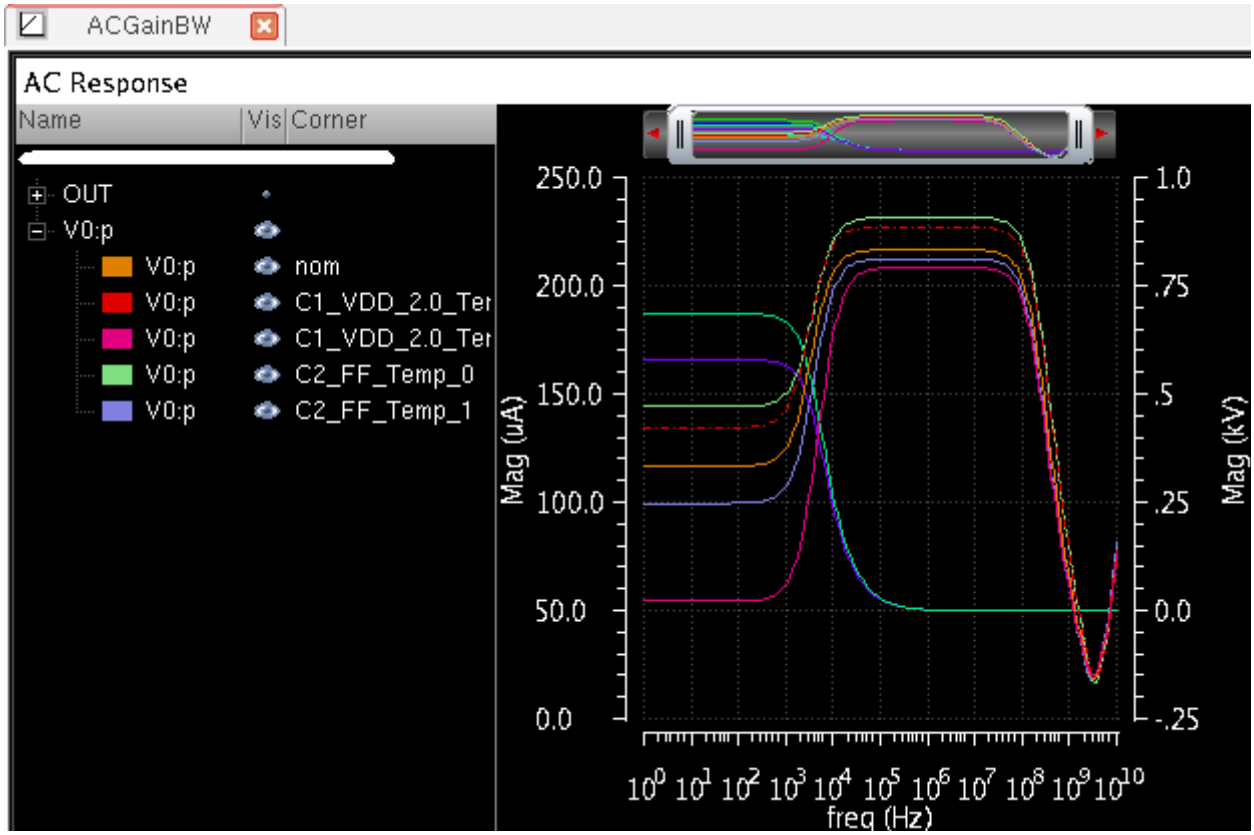


Run 2:

For the next simulation run, use the `nominal` and `C1` corners. In place of corner `C0`, use corner `C2`, which uses the same value of *`VDD`* as is being used in `C1`, but a different section from the model file. After simulation, the graph is updated as shown below.

© 2023 Cadence Design Systems, Inc.

Figure 19-13 Graphs Plotted for Corners: nominal, C1, and C2



Note the following changes:

- Traces plotted for the corners, `nominal` and `C1` have been retained.
- New traces have been plotted for corner `C2`.
- Traces for corner `C0` have been removed from the graph.

Run 3:

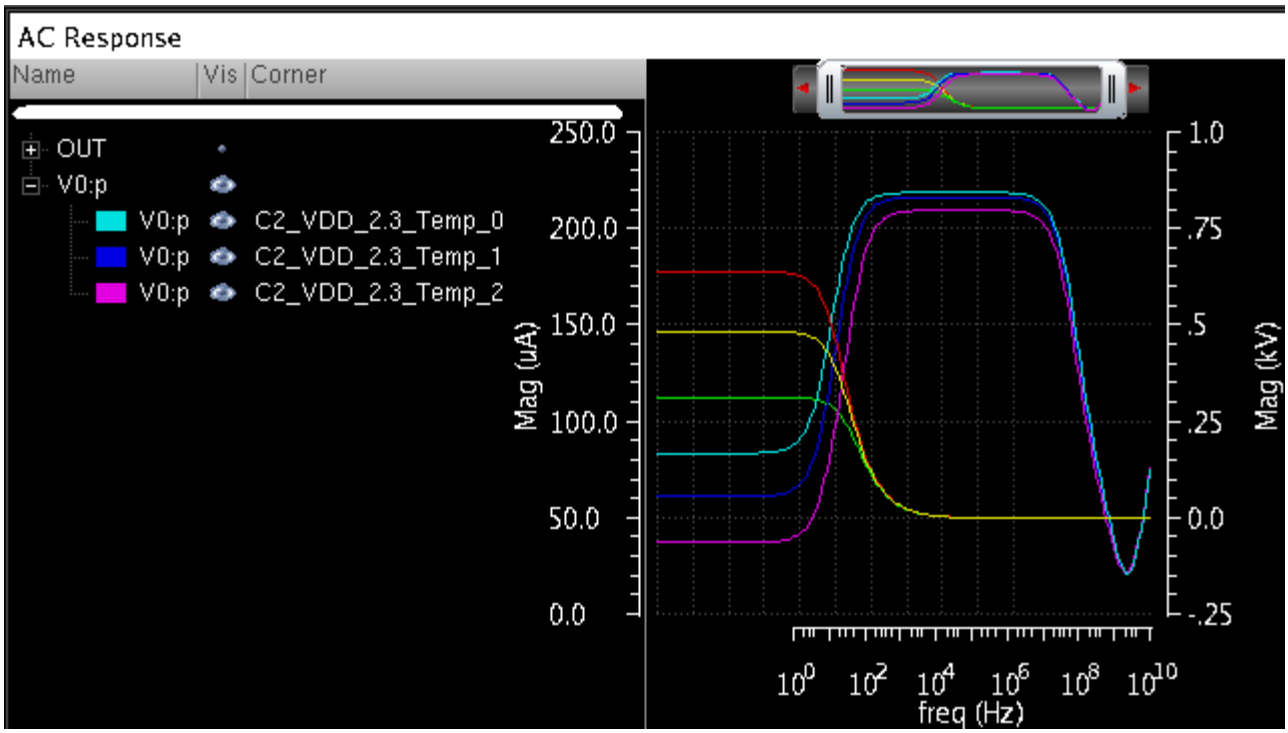
For the third run, use change the values of corner variables for corner `C2` and disable all other corners. Use the following variables for corner `C2`:

VDD = `2.3`
Temp = `25, 40, 75`

In this case, corner conditions of the new simulation run do not match with that of the previous run. Therefore, no trace is updated and only new traces are plotted, as shown below.

© 2023 Cadence Design Systems, Inc.

Figure 19-14 Graphs Plotted For Corner C2



The following table summarizes how graphs are updated when corners and sweep variables are varied across different simulation runs:

Table 19-6 Effect of Corners and Sweep Variations on Graph Settings

Corner or Sweep Variations Across Two Simulation Runs	Effect on Graph Settings
In the first run, a single run is used with the nominal corner. In the next run, the global variable is swept and the same nominal corner is used	■ Traces for the sweep values that are same across both the runs are refreshed and their trace settings are retained. For example, in the first run, *VDD*=1.8, and in the second run, *VDD*=1.8, 2.0, and 2.2. In this case, trace plotted for *VDD*=1.8 is refreshed. ■ If there is no common value for the swept variable across different run, new traces are plotted and settings are not retained. For example, in the first run, *VDD*=1.8, and in the second run, *VDD*=2.0, 2.2, and 2.4. In this case, trace plotted for *VDD*=1.8 is removed.

© 2023 Cadence Design Systems, Inc.

Table 19-6 Effect of Corners and Sweep Variations on Graph Settings

Corner or Sweep Variations Across Two Simulation Runs	Effect on Graph Settings
Both simulation runs sweep a common global variable. Nominal corner is same in both the runs.	■ Traces for the sweep values that are same across both the runs are refreshed and their trace settings are retained. ■ If none of the sweep values match across the runs, new traces are plotted and no settings are not retained.
In the first run, a corner is used. In the next run, a new corner is added to the existing setup.	■ Traces plotted for the existing corner are updated and their settings are retained. ■ For the new corner, new traces are plotted in the same graph.
Corners used in the first run are disabled and a new corner is added.	■ No traces are updated with new simulation data. ■ New traces are plotted in the same graph.
In both simulation runs, only nominal corner is used, but the values of variables in the nominal corner are different.	■ No traces are updated with new simulation data. ■ New traces are plotted in the same graph.
Both simulation runs use the same set of sweep values for global variables and nominal corner. In addition, the second run uses an additional corner.	■ Traces plotted for the nominal corner are updated with new simulation data and their settings are retained. ■ For the new corner, new traces are plotted in the same graph.

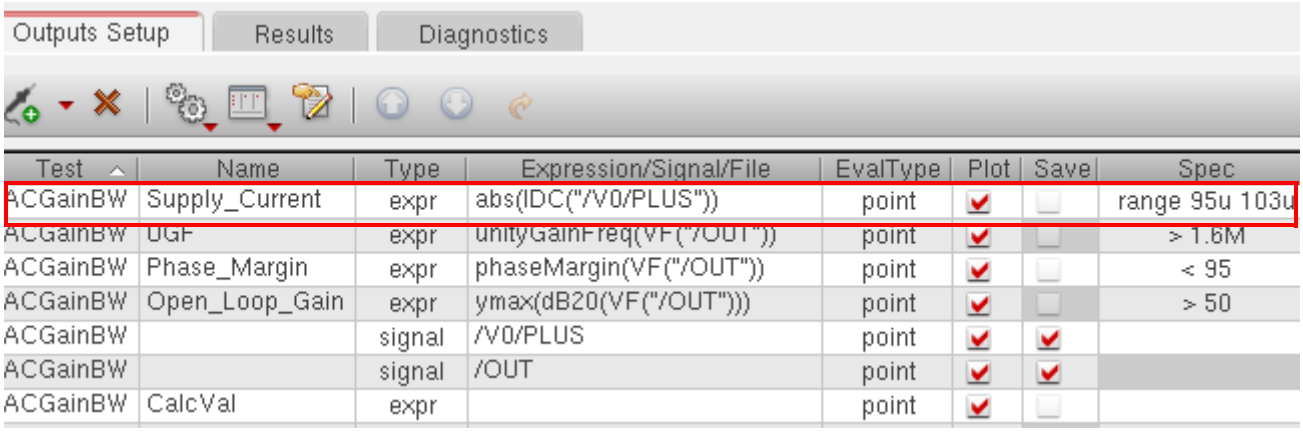
Displaying Spec Markers on Graphs

Spec markers are color markers that are displayed on a graph to demarcate the region under a trace as pass or fail region. This demarcation is done by shading the pass and fail regions with green and red line patterns, respectively. The demarcation depends on the status of simulation results with respect to the specifications given on the Outputs Setup tab.

For example, in the Outputs Setup tab, a specification is set for `Supply_Current` in the *Spec* column, as shown in the figure below.

© 2023 Cadence Design Systems, Inc.

Figure 19-15 Specifications in the Output Setup Tab

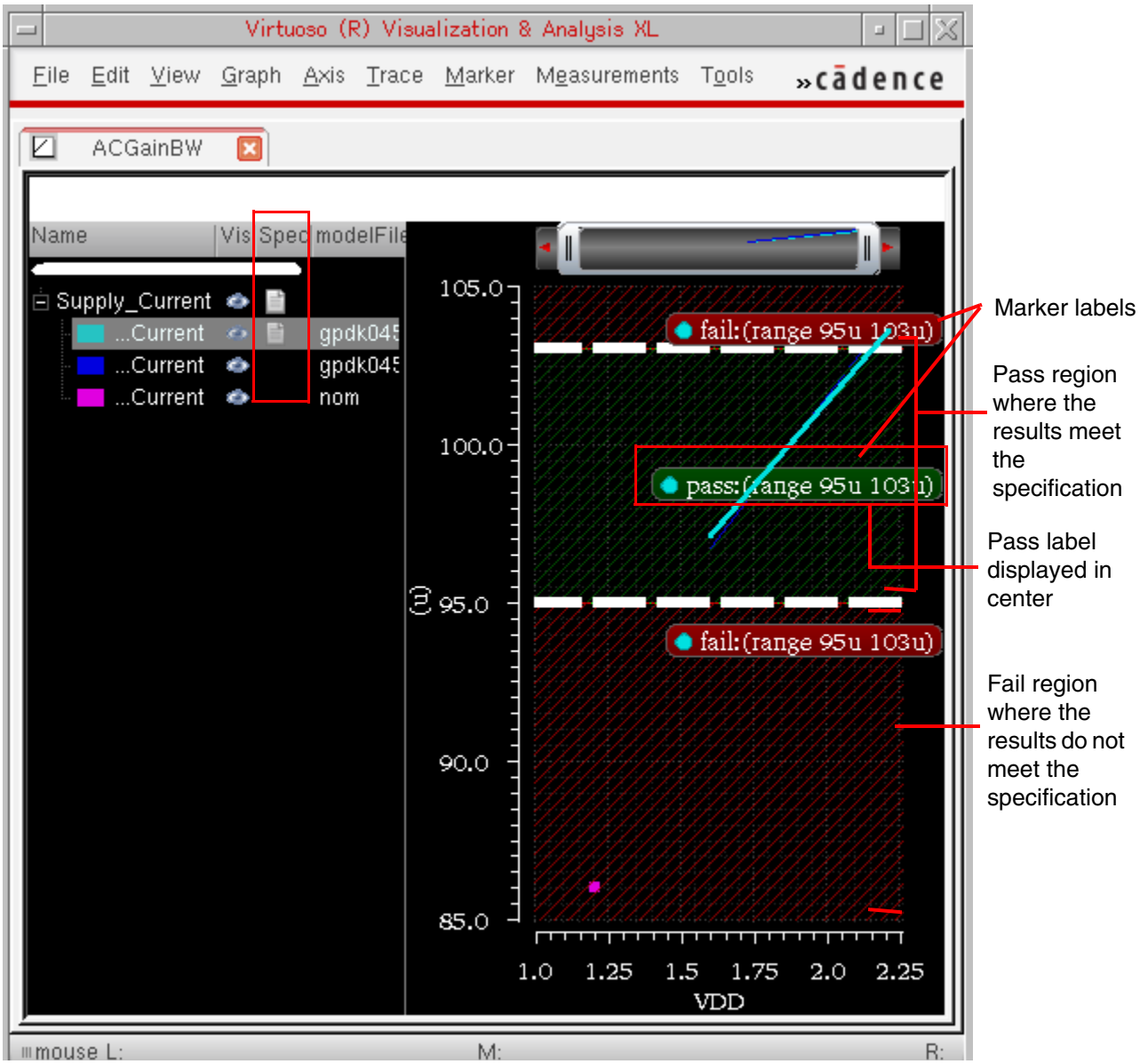


Test	Name	Type	Expression/Signal/File	EvalType	Plot	Save	Spec
ACGainBW	Supply_Current	expr	abs(IDC("/V0/PLUS"))	point	☑	☐	range 95u 103u
ACGainBW	UGF	expr	unityGainFreq(VF("/OUT"))	point	☑	☐	> 1.6M
ACGainBW	Phase_Margin	expr	phaseMargin(VF("/OUT"))	point	☑	☐	< 95
ACGainBW	Open_Loop_Gain	expr	ymax(dB20(VF("/OUT")))	point	☑	☐	> 50
ACGainBW		signal	/V0/PLUS	point	☑	☑	
ACGainBW		signal	/OUT	point	☑	☑	
ACGainBW	CalcVal	expr		point	☑	☐	

When you plot the results in Virtuoso Visualization and Analysis XL window, spec markers are displayed for `Supply_Current`, as shown in the figure below.

© 2023 Cadence Design Systems, Inc.

Figure 19-16 Spec Markers on a Graph



In the above figure, depending on the spec range, pass and fail regions are shaded with green and red patterns, respectively. The two white dotted bold lines that indicate the threshold value separate these regions. When you select a trace in the graph or in the trace legend, the thickness of the threshold line increases and the spec marker label background changes to its respective pass or fail color. Marker label displayed in each region shows the pass or fail status and the specification type and values. For example, the `pass:(range 95u 103u)` marker label displayed in the above graph shows that the specification type is `range` and the target range values are `95u` and `103u`. The results that are plotted in the region marked by this label meet the specification.

© 2023 Cadence Design Systems, Inc.

By default, the pass and fail labels appear near the threshold line. However, if the specification type is `range`, the pass label is displayed in the center as shown in Figure 19-16 on page 1153.

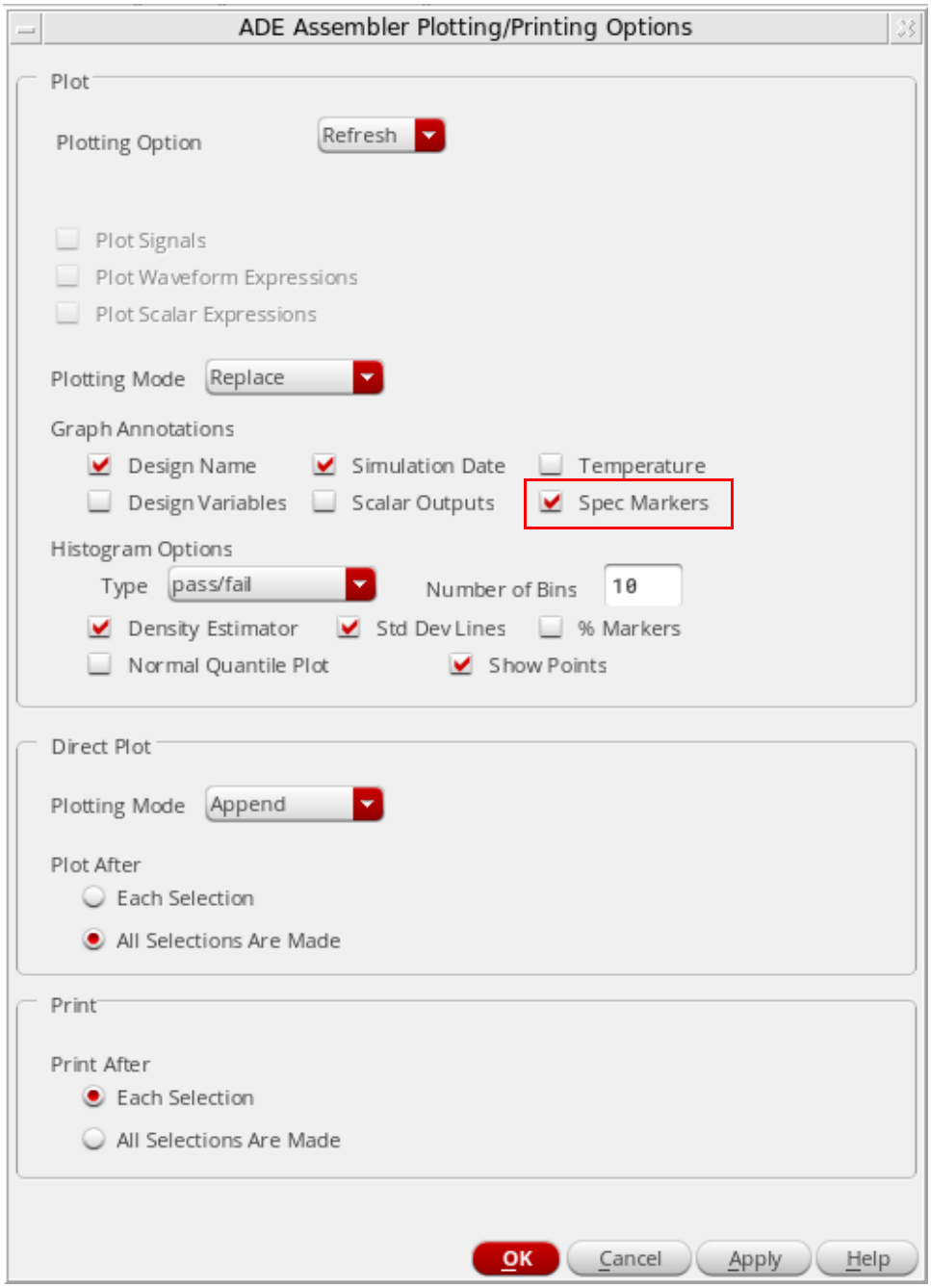
To display the *Spec* column in trace legend, right-click the trace legend header and choose *Spec*.

Enabling Display of Spec Markers

To display spec markers, in the ADE Assembler Plotting/Printing Options form, select the *Spec Markers* graph annotation and click *Apply*.

© 2023 Cadence Design Systems, Inc.

Figure 19-17 ADE Assembler Plotting/Printing Options Form



Note: By default, the *Spec Markers* option is deselected.

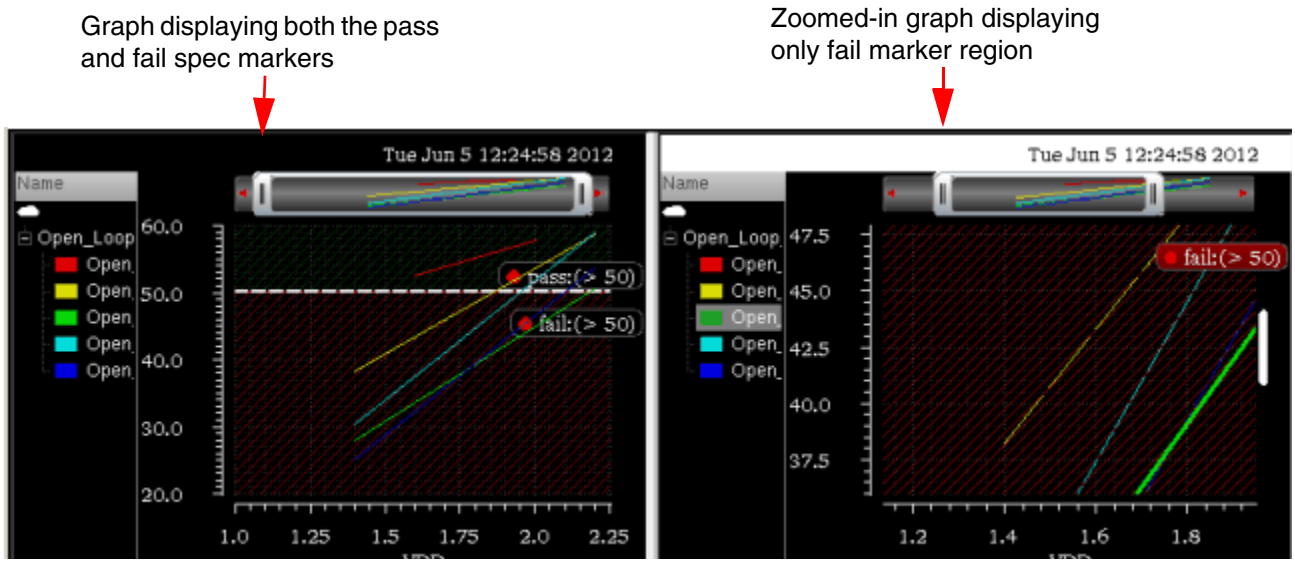
If you select the *Spec Markers* option and plot graphs, ADE Assembler checks for the presence of specifications for the result to be plotted. For the measurements to be plotted, if you have specified spec values in the ADE Assembler Outputs section, the tool shades the pass and fail regions in the graph.

© 2023 Cadence Design Systems, Inc.

Displaying Spec Markers on Zoomed-in Graph

Spec markers are displayed on the graph only if the associated trace is visible. When you zoom in a graph, only the spec markers that fall in the zoomed-in graph portion are visible. This helps improve visibility when multiple spec markers are applied on multiple traces.

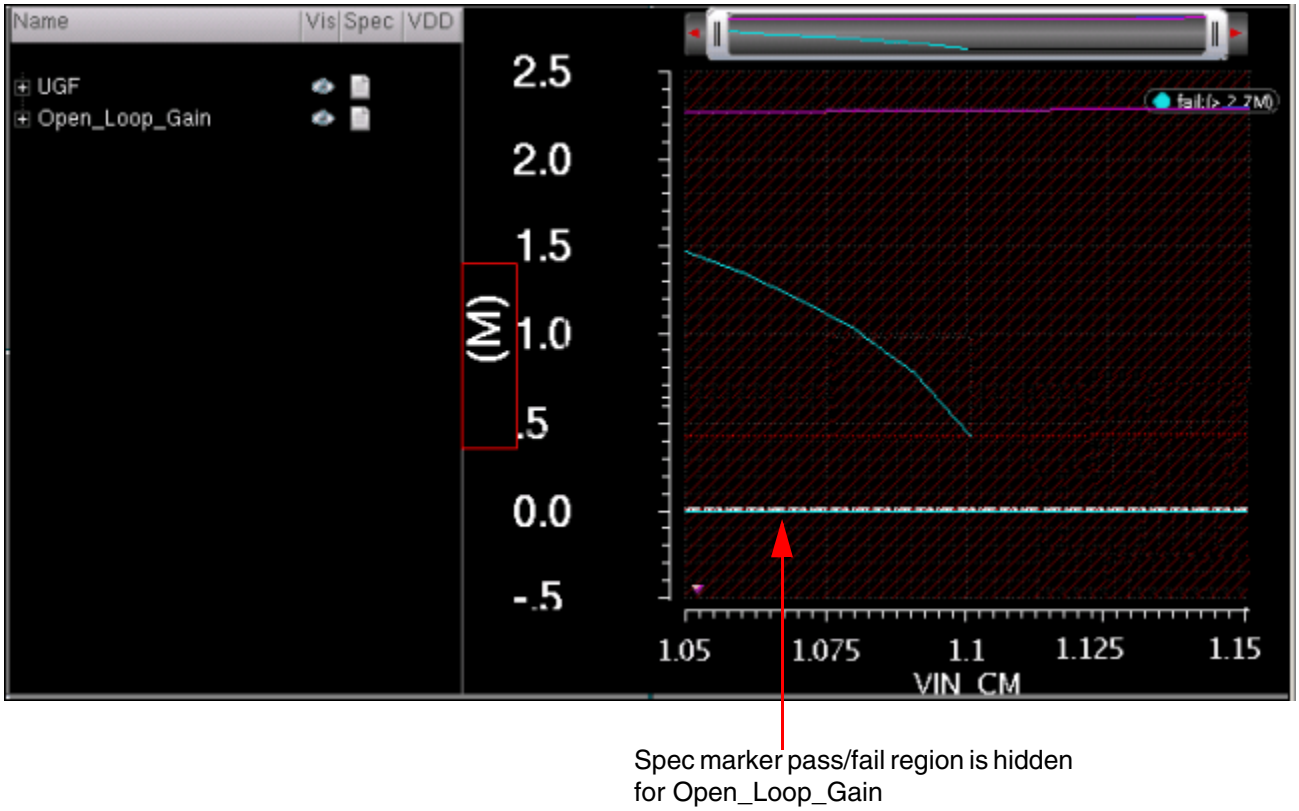
Figure 19-18 Spec Markers on Zoomed-in Graph



The figures below demonstrate an example when two traces are plotted in the same graph and share the same Y-axis. In this example, the trace `Open_Loop_Gain` has a huge Y-axis scale and the other overlaid trace, `UGF`, has a comparatively small Y-axis scale. Therefore, the spec markers for `Open_Loop_Gain` are not visible in the graph (as shown in Figure 19-18 on page 1156). To display the hidden spec markers, you need to zoom in the graph until you get the small Y-axis scale.

© 2023 Cadence Design Systems, Inc.

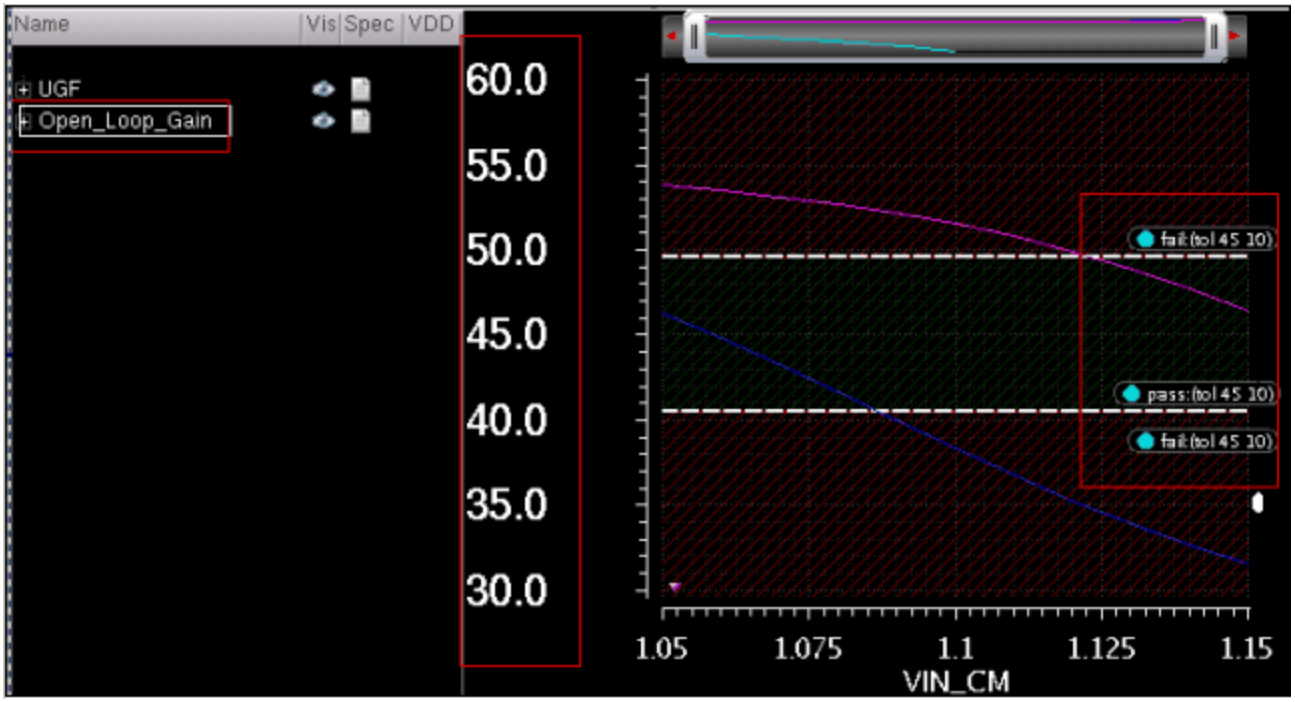
Figure 19-19 Example: Spec Markers on a Zoomed-in Graph



When you zoom in the above displayed graph to `ymax=60` and `ymin=30`, the hidden spec markers become visible as shown in Figure 19-20 on page 1158.

© 2023 Cadence Design Systems, Inc.

Figure 19-20 Example: Spec Markers on a Zoomed-in Graph

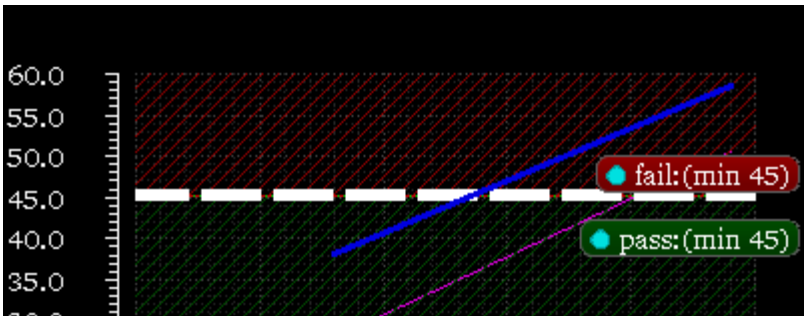
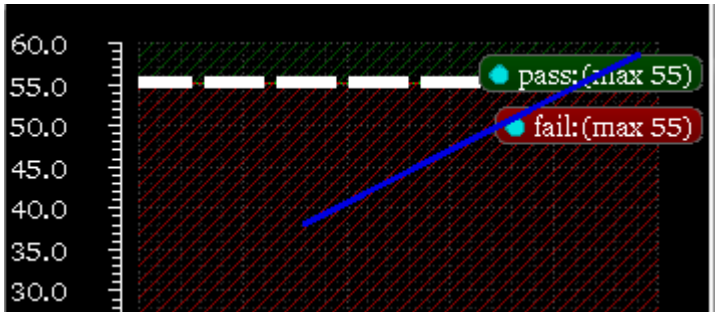


Spec Markers for Different Specification Types

Depending on the type of specification, ADE Assembler displays spec markers in different ways, as described in the following table.

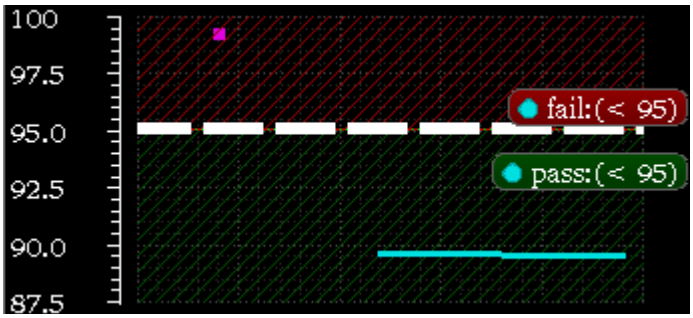
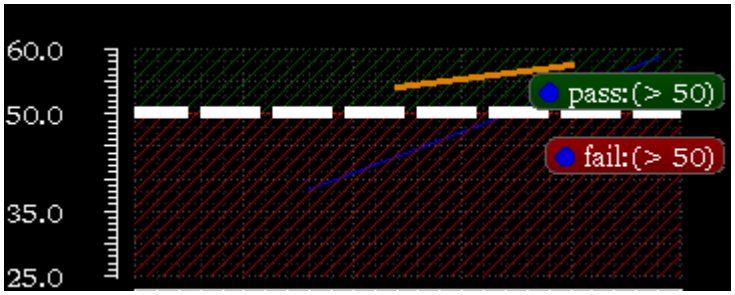
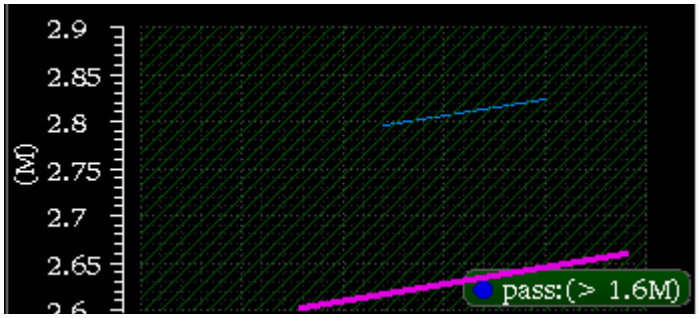
© 2023 Cadence Design Systems, Inc.

Table 19-7 Spec Markers for Different Specification Types

Specification Type	Spec Marker Display
minimum	Displays one horizontal marker line at the target minimum value. The region in which the results are below the minimum target is shaded with green and the region above it is shaded with red. 
maximum	Displays one horizontal marker line at the target maximum value. The region in which the results are greater than the maximum target is shaded with green and the region below it is shaded with red. 

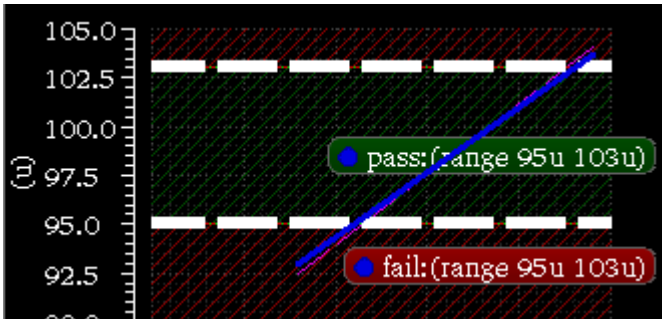
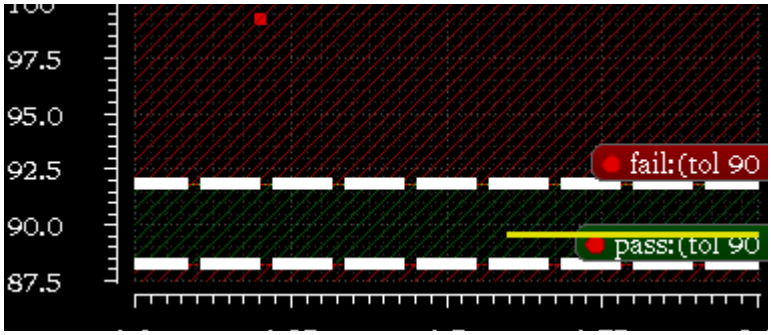
© 2023 Cadence Design Systems, Inc.

Table 19-7 Spec Markers for Different Specification Types

Specification Type	Spec Marker Display
<	Displays one horizontal marker line at the spec value. This specification requires that the results should be less than the spec value. Therefore, the region below the marker is shaded with green and the region above it is shaded with red. 
>	Displays one horizontal marker line at the spec value. This specification requires that the results should be greater than the spec value. Therefore, the region below the marker is shaded with red and the region above it is shaded with green.  If all the results appear in one region, the other region is not displayed, as shown in the figure below: 

© 2023 Cadence Design Systems, Inc.

Table 19-7 Spec Markers for Different Specification Types

Specification Type	Spec Marker Display
range	Displays two horizontal marker lines—one at the lower bound value and the other at the upper bound value. This specification type requires that results should be within the given range. Therefore, the region between the two marker lines is shaded with green and the region outside these lines is shaded with red. 
tol	Displays two horizontal marker lines—one at the lower end of the tolerance range and the other at the higher end. This specification type requires that results should be within the given tolerance range. Therefore, the region between the two marker lines is shaded with green and the region outside these lines is shaded with red. 
info	No marker is displayed for this specification type.

Important

While displaying spec markers, ADE Assembler does not consider overridden specifications for corners. If any overridden specification exists, spec markers are displayed by considering only the global specifications.

© 2023 Cadence Design Systems, Inc.

Changing Properties of Spec Markers

You can change the display of spec markers by setting their properties in the Virtuoso Visualization and Analysis XL graph. For example, you can change the shading color or you can choose to show or hide the markers in the pass or fail region.

For more details, refer to the Changing Spec Marker Properties section in the *Virtuoso Visualization and Analysis XL User Guide*.

© 2023 Cadence Design Systems, Inc.

Printing Results

To print text results and reports to the Results Display Window, do the following:

1. On the Results tab of the Outputs pane, right-click a test name and choose *Print*.

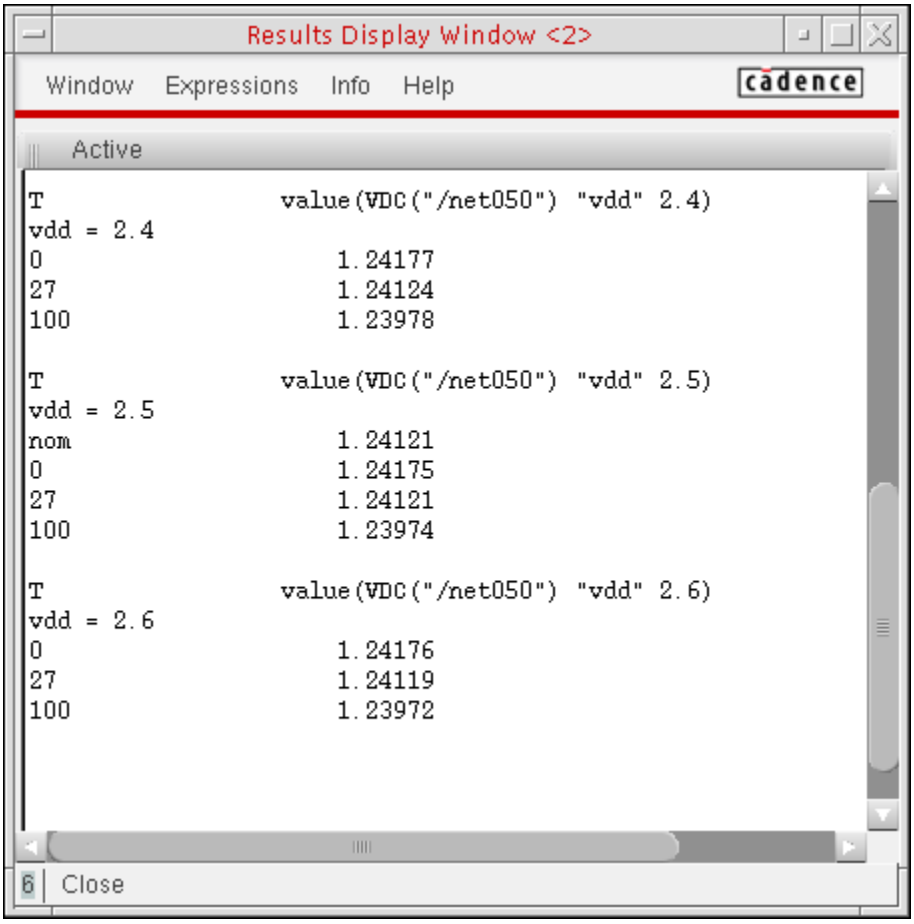
   A submenu of functions appears.
   Only those functions that apply to the current results are available for selection.

2. Select a function.

   If you have the prompt line turned on, you can read the prompt text for a hint about what you need to select.

© 2023 Cadence Design Systems, Inc.

The Results Display Window appears.



3. In the schematic window, select one or more nets or terminals to plot using the function you selected.

   The values for the selected nets or terminals appear in the Results Display Window according to the function you selected. If a Results Display Window was not already open, then one appears. If a Results Display Window was already open, then it appears in the foreground.

   Note: See also Specifying When to Print Results.

Here is a table of print functions.

© 2023 Cadence Design Systems, Inc.

Print Functions

Function	Description
DC Node Voltages **Note:** This function is available following a DC analysis.	Print DC node voltages of selected nodes **1.** In the schematic window, select one or more nodes. **2.** Press *Esc* when you are finished selecting.
DC Operating Points **Note:** This function is available following a DC analysis.	Print DC operating points of selected instances **1.** In the schematic window, select one or more instances. If you select an instance that is a subcircuit definition, the program prints operating points for all devices in the subcircuit. **2.** Press *Esc* when you are finished selecting. **Note:** It may take some time to search for all instances in a subcircuit definition. To disable this feature, you can set the following environment variable in your `.cdsenv` file: `asimenv.printing printInlines boolean nil`
Model Parameters	Print model parameters of selected instances **1.** In the schematic window, select one or more instances. If you select an instance that is a subcircuit definition, the program prints model parameters for all devices in the subcircuit. **2.** Press *Esc* when you are finished selecting. **Note:** It may take some time to search for all instances in a subcircuit definition. To disable this feature, you can set the following environment variable in your `.cdsenv` file: `asimenv.printing printInlines boolean nil`

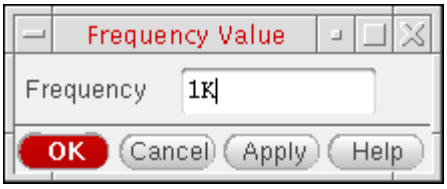
© 2023 Cadence Design Systems, Inc.

Print Functions, *continued*

Function	Description
Transient Node Voltages **Note:** This function is available following a transient analysis.	Print transient node voltages of selected nodes The Select Time Value form appears when you select this function. 1. In the *Time* field, type the time value for which you want to print transient node voltages. 2. Click *OK*. 3. In the schematic window, select one or more nodes. 4. Press *Esc* when you are finished selecting.
Transient Operating Points **Note:** This function is available following a transient analysis.	Print final transient operating points of selected items 1. In the schematic window, select one or more instances or nodes. If you select an instance that is a subcircuit definition, operating point values for all devices in the subcircuit appear in the Results Display Window. 2. Press *Esc* when you are finished selecting. **Note:** It may take some time to search for all instances in a subcircuit definition. To disable this feature, you can set the following environment variable in your `.cdsenv` file: `asimenv.printing printInlines boolean nil`
S-Parameter	Print S-parameter data

© 2023 Cadence Design Systems, Inc.

Print Functions, *continued*

Function	Description
Noise Parameters **Note:** This function is available following a noise analysis.	Print noise parameters of selected instances The Select Frequency Value form appears when you select this function. The default frequency is 1K.  1. In the *Frequency* field, type the frequency value for which you want to print noise parameters. 2. Click *OK*. 3. In the schematic window, select one or more instances. 4. Press *Esc* when you are finished selecting.
Noise Summary **Note:** This function is available following a noise analysis.	Opens the Noise Summary form so you can print noise contributions of selected instances See "Printing Noise Summary Information" on page 1168 for more information.
DC Mismatch Summary **Note:** This function is available following a `dcmatch` analysis.	Opens the Dcmatch Summary form so you can print DC mismatch information for your design See "Printing DC Mismatch Summary" on page 1177 for more information.
Stability Summary **Note:** This function is available following a stability analysis.	Print phase margin and gain margin results for every combination of sweep variable values
Pole Zero Summary **Note:** This function is available following a pole-zero analysis.	Opens the Pole-Zero Summary form so you can print pole-zero information for your design See "Printing Pole Zero Summary" on page 1179 for more information.

© 2023 Cadence Design Systems, Inc.

Print Functions, *continued*

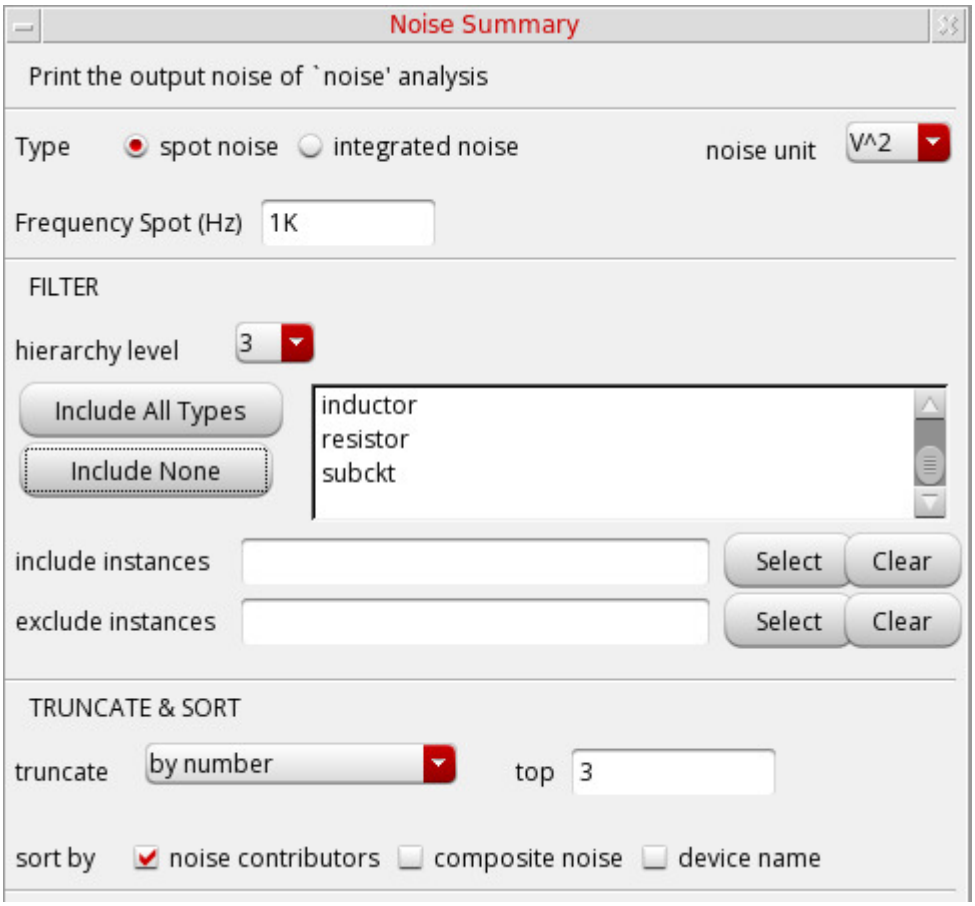
Function	Description
Sensitivities **Note:** This function is available following a sensitivity analysis.	Print sensitivities 1. In the schematic window, select one or more nets or ports. 2. Press *Esc* when you are finished selecting.

Printing Noise Summary Information

To print noise contributions of selected instances, do the following:

1. On the *Results* tab of the *Outputs* pane, right-click a test name and choose *Print – Noise Summary*.

   The *Noise Summary* form appears.



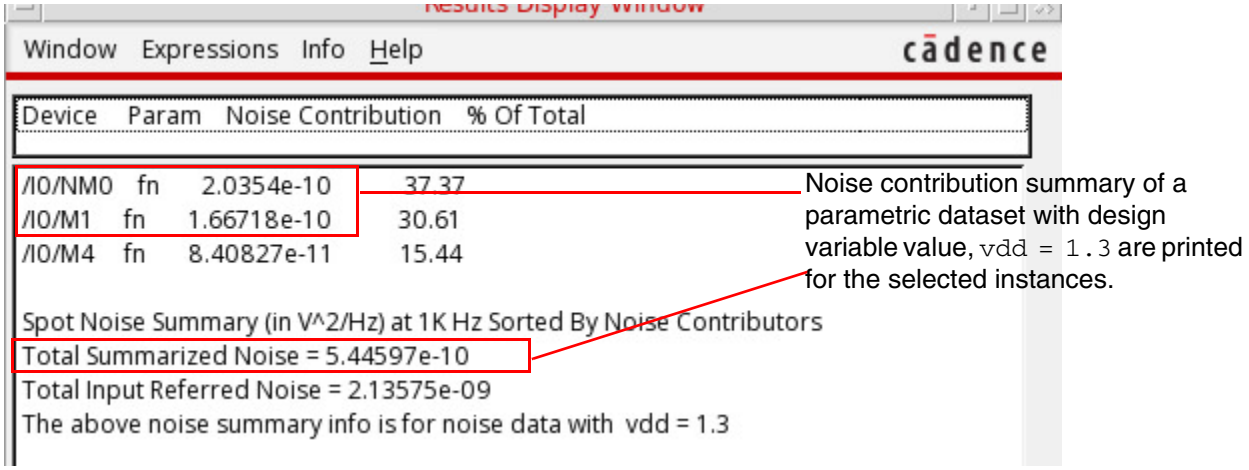
© 2023 Cadence Design Systems, Inc.

Note: To print the noise summary of a leaf dataset, right-click a result in the <*corner_name*> column.

2. Choose one of the following noise summary types:
   - *spot noise* – Type a frequency in the *Frequency Spot (Hz)* field.
   - *integrated noise* – Specify a range and weighting option.
3. In the *FILTER* group box, do the following:
   a. From the *hierarchy* level drop-down list, specify the hierarchy levels up to which you want to view the noise contribution results in the *Results Display Window*.
   b. Specify the options to include or exclude device types or instances. Refer to the Specifying Device Types and Instances to Include or Exclude section for more information.
4. (Optional) In the *TRUNCATE & SORT* group box, specify options to truncate and sort your noise summary data.
5. From the PARAMETRIC VARIABLES list, select a value of the design variable.

   Note: This list is displayed only for parametric dataset.
6. Click *OK*.

   The noise summary information you specified appears in the *Results Display Window*.



See also "Controlling Precision of Printed Noise Data" on page 1175.

© 2023 Cadence Design Systems, Inc.

Specifying Options for Integrated Noise

When you select *integrated noise*, the *From (Hz)* and *To (Hz)* fields become active and you can specify a *weighting* option.

To specify options after selecting *integrated noise*, do the following:

1. In the *From (Hz)* field, type a starting value for the frequency range for the integration.
2. In the *To (Hz)* field, type an ending value for the frequency range for the integration.
3. Select one of the following *weighting* options:
   - *flat* – The program integrates over the original unweighted waveform.
   - *from weight file* – You can specify a file containing weight factors to apply to the noise contributions of particular frequencies prior to integrating. The file must contain one of the following entries as the first line: `db`, `mag`, `dbl`, `DB`, `MAG`, `DBL`. Each additional line must contain a pair of X and Y values. All the pairs together must define a function. For example:

     ```
     mag
     1          .001641
     60         .001641
     100        .007499
     200        .05559
     ```
4. Return to "Printing Noise Summary Information" on page 1168.

Specifying Device Types and Instances to Include or Exclude

Device types in your design appear in the list box in the *FILTER* group box.

To specify which device types to include and which to exclude from the noise summary, do the following:

1. To include all device types in the summary, click *All Types*.
2. (Optional) To exclude individual device types, hold down the *Ctrl* key, and then select the device types you want to exclude.

Alternatively, you can do the following:

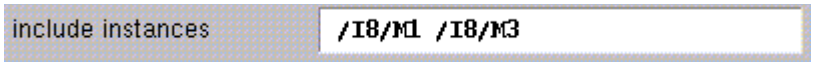
1. To exclude all device types from the summary, click *None*.
2. Select each device type you want to include in the summary:
   - You can hold down the *Shift* key to select more than one contiguous device type.
   - You can hold down the *Ctrl* key to select more than one noncontiguous device type.

© 2023 Cadence Design Systems, Inc.

To specify instances to include in the noise summary, do the following:

1. To the right of the *include instances* field, click *Select*.
2. In the schematic window, select one or more instances.

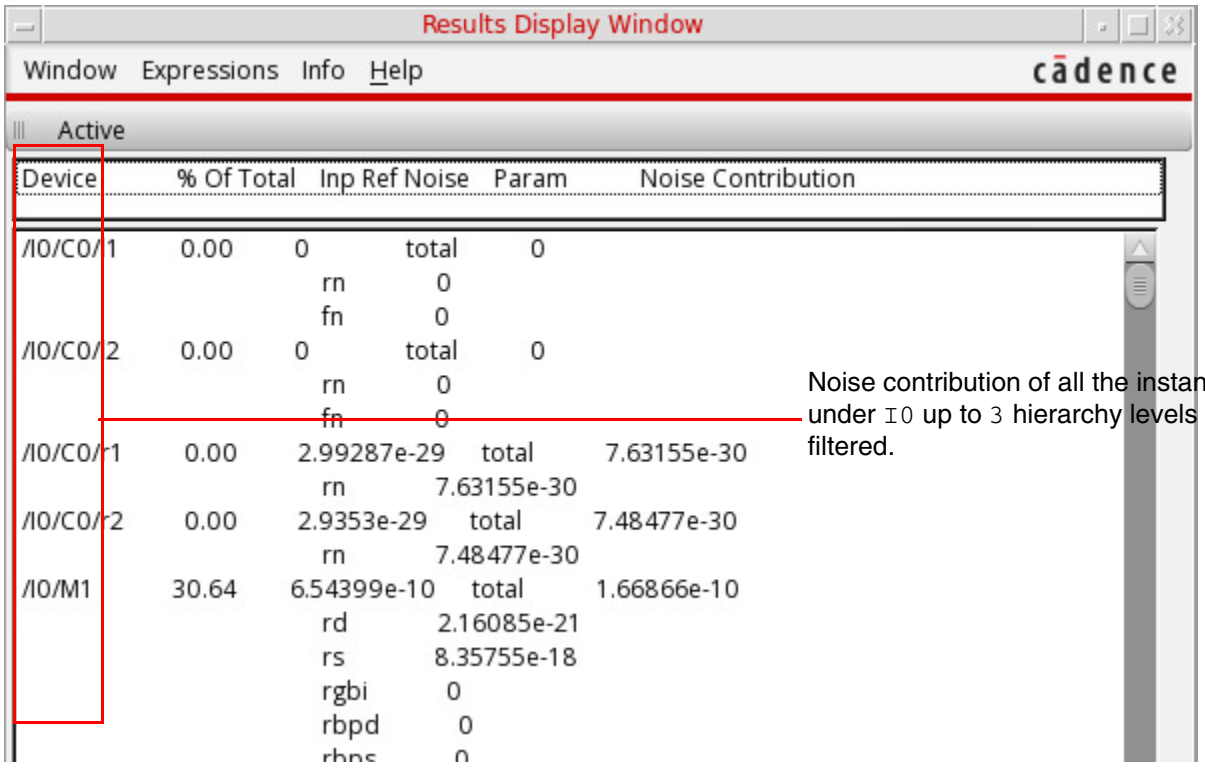
   Each instance path appears in the *include instances* field.



3. Press *Esc* when you are finished selecting.

Alternatively, in the include instances field, you can type the instance names for which you want to filter the noise contribution results. For example, if you want to view the noise contribution only for the `/I0/C0` instance, type `/I0/C0`.

Note: You can also use the wildcard character `*` to filter the results. For example, if you want to view the noise contribution for all the instances under `I0`, type `/I0/*` in the *include instances* field. The *Results Display Window* shows the results for all the instances under `I0` up to the specified hierarchy levels, as shown in the following figure:



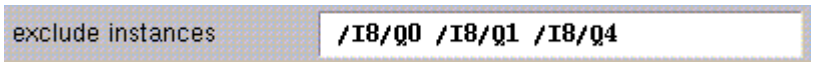
To specify instances to exclude from the noise summary, do the following:

1. To the right of the *exclude instances* field, click *Select*.

© 2023 Cadence Design Systems, Inc.

2. In the schematic window, select one or more instances.

   Each instance path appears in the *exclude instances* field.



3. Press *Esc* when you are finished selecting.

Truncating Noise Summary Data

From the *TRUNCATE & SORT* group box on the Noise Summary form, you can select a truncation option to limit the number of noise contributors that appear in the Results Display Window when you click *Apply* or *OK*.

To specify no truncation of noise data, select *none*. All noise contributors appear in the noise summary report.

To limit the number of noise contributors that appear in the summary, do the following:

1. Select *by number*.
2. In the *top* field, type the number of highest noise contributors you want to see.

   The program reports only that number of noise contributors.

To limit the report to only those devices that contribute a certain percentage of the total noise, do the following:

1. Select *by rel. threshold*.
2. In the *noise %* field, type the minimum percentage noise contribution threshold.

   The program reports only those devices that contribute at least the minimum percentage of the total noise.

To limit the report to only those devices that contribute a minimum level of noise, do the following:

1. Select *by abs. threshold*.
2. In the *noise value* field, type the minimum noise contribution threshold.

   The program reports only those devices that contribute at least the minimum noise value.

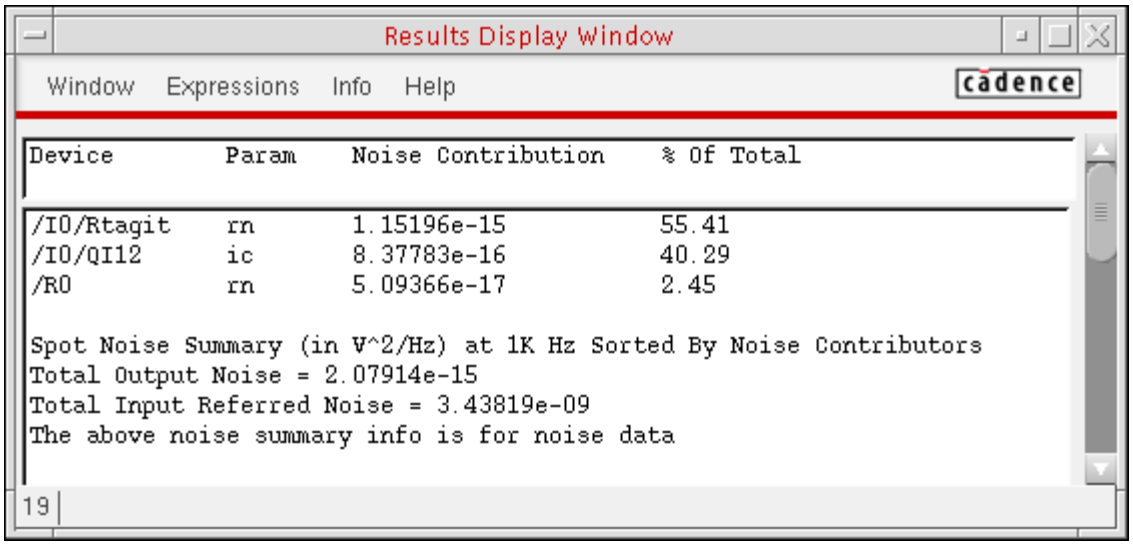
© 2023 Cadence Design Systems, Inc.

Sorting Noise Summary Data

From the *TRUNCATE & SORT* group box on the Noise Summary form, you can sort the list of devices that the program reports by noise contributors (highest to lowest), composite noise (highest to lowest), or device name (alphabetical from A to Z). The report appears in the Results Display Window when you click *Apply* or *OK*.

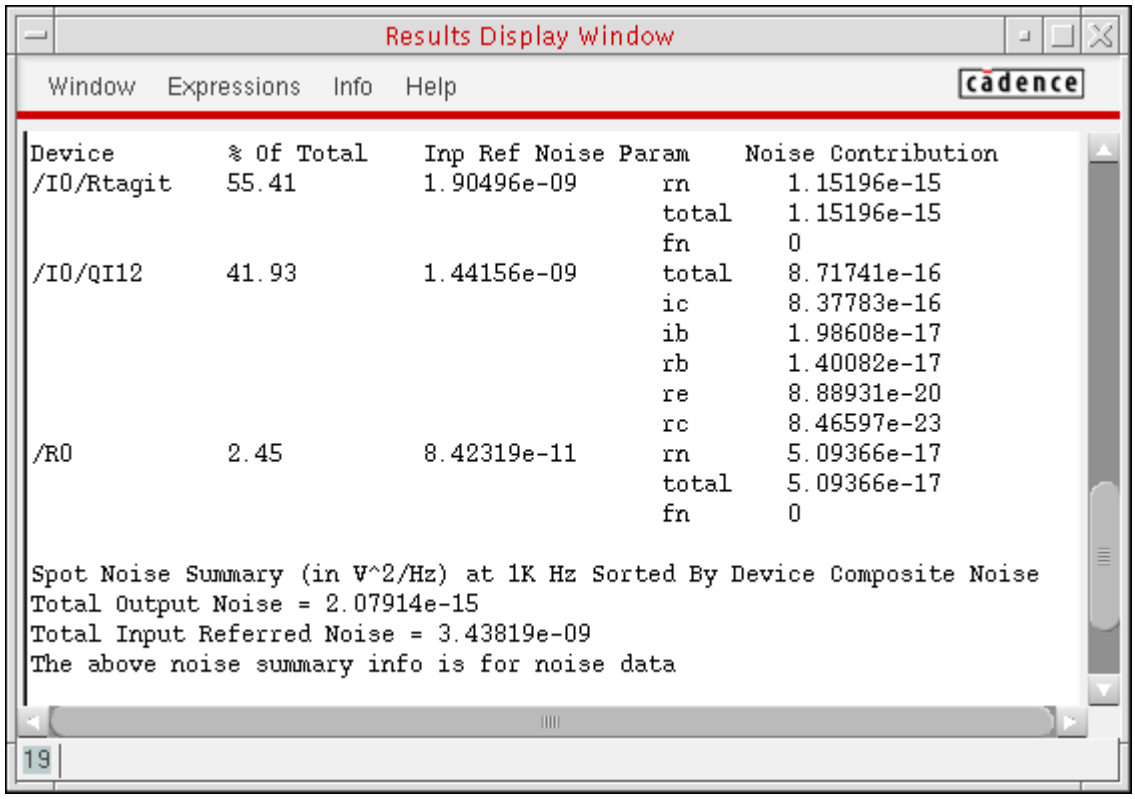
To request one or more sorted lists of noise contributors in your noise summary report, do the following:

- Select one or more of the following *sort by* options:
  - *noise contributors* – Sorts devices by noise contribution, highest to lowest.



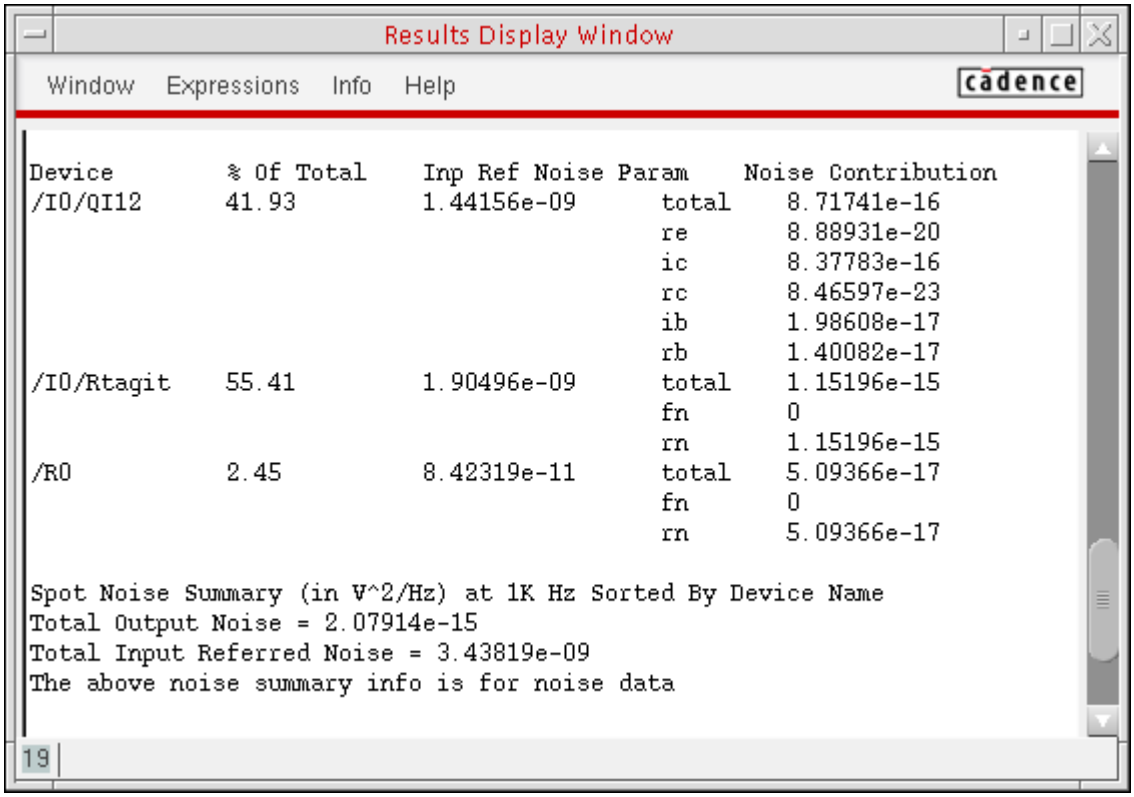
© 2023 Cadence Design Systems, Inc.

- *composite noise* – Sorts devices by composite noise contribution, highest to lowest.



© 2023 Cadence Design Systems, Inc.

- *device name* – Sorts devices by device name in alphabetical order from A to Z.



You can select one, two, or three check boxes. The number of boxes you select determines the number of lists that appear in your Results Display Window when you click *Apply* or *OK*.

Controlling Precision of Printed Noise Data

You can control the precision of them noise data that you print by setting the following variable in your `.cdsenv` file:

```
asimenv.noiseSummary digits int numberOfDigits
```

where *`numberOfDigits`* is the number of digits to print.

For example, to specify ten digits of precision, use the following setting:

```
asimenv.noiseSummary digits int 10
```

The default value for this variable is `6`.

Alternatively, you can set this value for the current session using the following command in your CIW:

© 2023 Cadence Design Systems, Inc.

```
envSetVal("asimenv.noiseSummary" "digits" 'int 10)
```

You can specify the number of digits to use when printing relative noise contributions by setting the following variable in your `.cdsenv` file:

```
asimenv.noiseSummary percentDecimals int numberOfDigits
```

where *numberOfDigits* is the number of digits to print.

For example, to specify four digits for relative contributions, use the following setting:

```
asimenv.noiseSummary percentDecimals int 4
```

The default value for this variable is `2`.

Alternatively, you can set this value for the current session using the following command in your CIW:

```
envSetVal("asimenv.noiseSummary" "percentDecimals" 'int 4)
```

You can control the precision of your printed results using the `aelPushSignifDigits` SKILL function as follows:

```
aelPushSignifDigits(numDigits)
```

where *numDigits* is the number of digits of precision you want.

Example

```
aelPushSignifDigits(4)
rn              37.9322e-18    fn              0               total
37.9322e-18

aelPushSignifDigits(8)
rn              37.932238e-18  fn              0               total
37.932238e-18
```


© 2023 Cadence Design Systems, Inc.

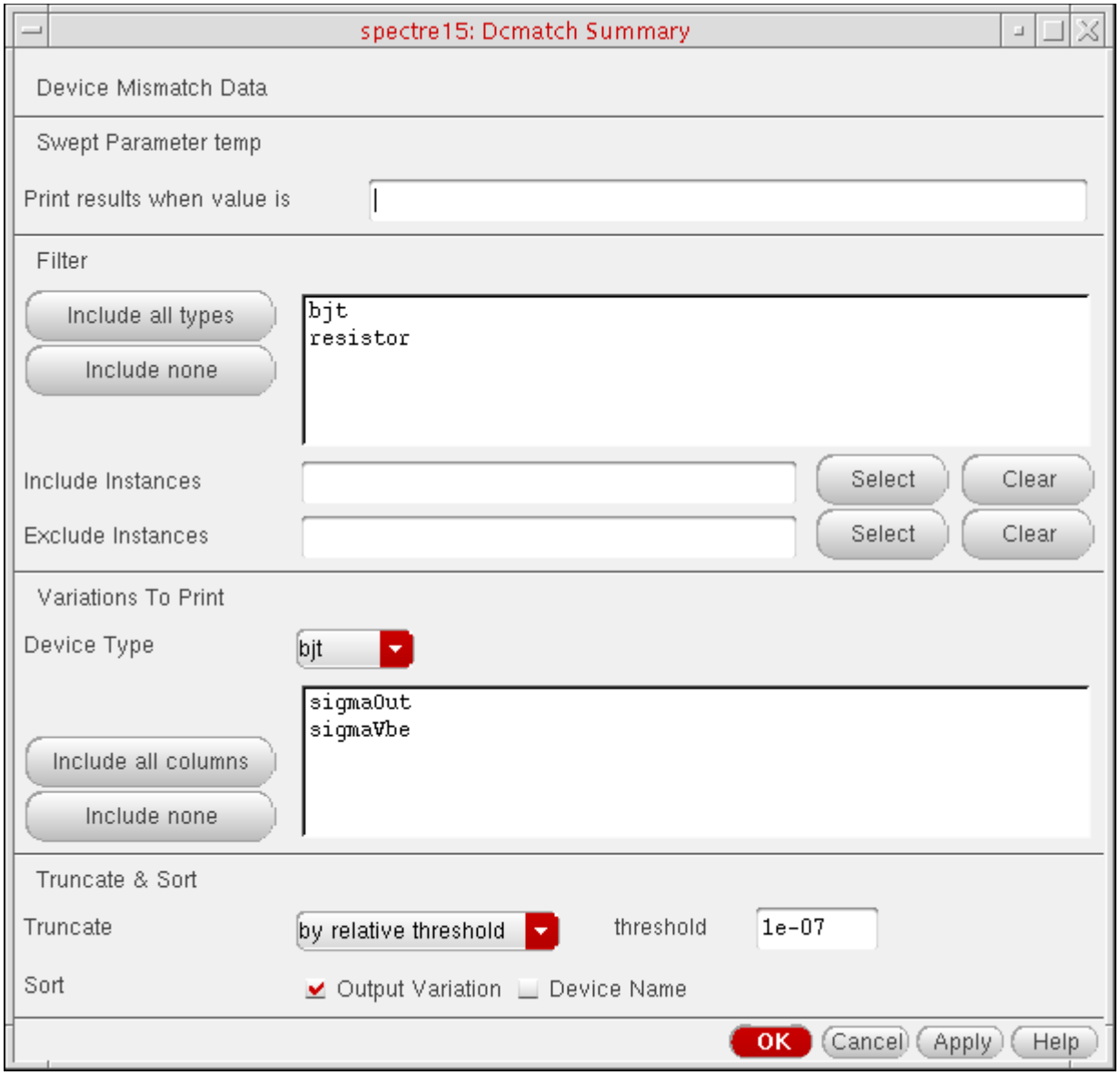
Printing DC Mismatch Summary

To print the DC mismatch summary for your circuit, do the following:

1. On the Results tab of the Outputs pane, right-click a test name and choose *Print – DC Mismatch Summary*.

   Note: This menu option is available when you run a `dcmatch` analysis or when you select a test that has `dcmatch` analysis results.

   The Dcmatch Summary form appears.



© 2023 Cadence Design Systems, Inc.

2. In the *Print results when value is* field, type a value.

   Note: The *Swept Parameter* group box appears on the form only when you sweep a parameter through numeric values.

3. In the Filter group box, specify the device types you want to include in the summary.
   - Click *Include all types* to include all device types.
   - Click *Include none* to exclude all device types.
   - Use the *Include Instances* and *Exclude Instances* fields to include and exclude specific instances.
     - Type instance names in these fields.
     - Click *Select* to select instances from the schematic.
     - Click *Clear* to clear the fields.

4. In the *Variations to Print* group box, do the following:
   a. In the *Device Type* drop-down list, select a device type.
   b. In the list area below, select one or more variations for the selected device type.
      - Click *Include all columns* to select all variations in the list area.
      - Click *Include none* to clear your selection.

5. In the *Truncate & Sort* group box, select one of the following truncation methods:
   - *none* – No truncation.

   - *by number* – In the *top* field, type the number of top contributors you want to see.

   - *by relative threshold* – In the *threshold* field, type a relative threshold value.

   - *by absolute threshold* – In the *mismatch* field, type an absolute threshold value.

© 2023 Cadence Design Systems, Inc.

6. For *Sort*, select one or both of the following sorting methods:
   - *Output Variation*
   - *Device Name*

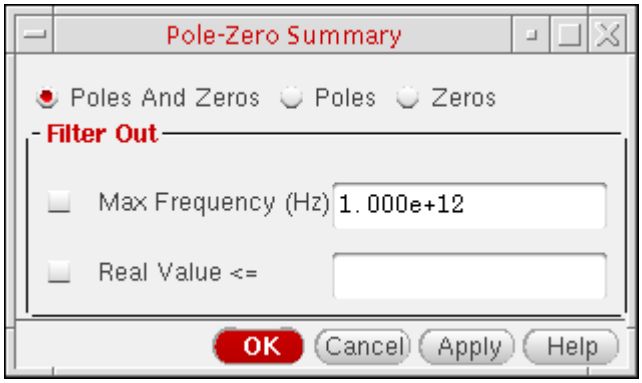
Printing Pole Zero Summary

To print the pole-zero summary for your circuit, do the following:

1. On the Results tab of the Outputs pane, right-click a test name and choose *Print – Pole-Zero Summary*.

   Note: This menu option is available when you run a pole-zero analysis or when you select a test that has pole-zero analysis results.

   The Pole-Zero Summary form appears.



2. Select one of the following options:
   - *Poles and Zeros* if you want to plot both poles and zeros.
   - *Poles* if you want to plot only poles.
   - *Zeros* if you want to plot only zeros.
3. In the *Filter Out* group box, select zero or more of the following filtering mechanisms:
   - *Max Frequency*

     This option enables you to filter out poles and zeros that are outside the frequency band of interest (FBOI) and that do not influence the transfer function in the FBOI. The default value is whatever appears in the *fmax* field on the Pole-Zero Options form. For the Direct Plot form, *fmax* is read from the header of the `psf` data. The program filters out any poles and zeros whose magnitudes exceed the frequency value you type in this field.

© 2023 Cadence Design Systems, Inc.

- *Real Value*

  This option enables you to specify the real part of the frequency. The program filters out any poles and zeros whose real values are less than or equal to the real value you type in this field.

4. Click *OK*.

   Pole-zero data appears in the Results Display Window according to the criteria you specified.

© 2023 Cadence Design Systems, Inc.

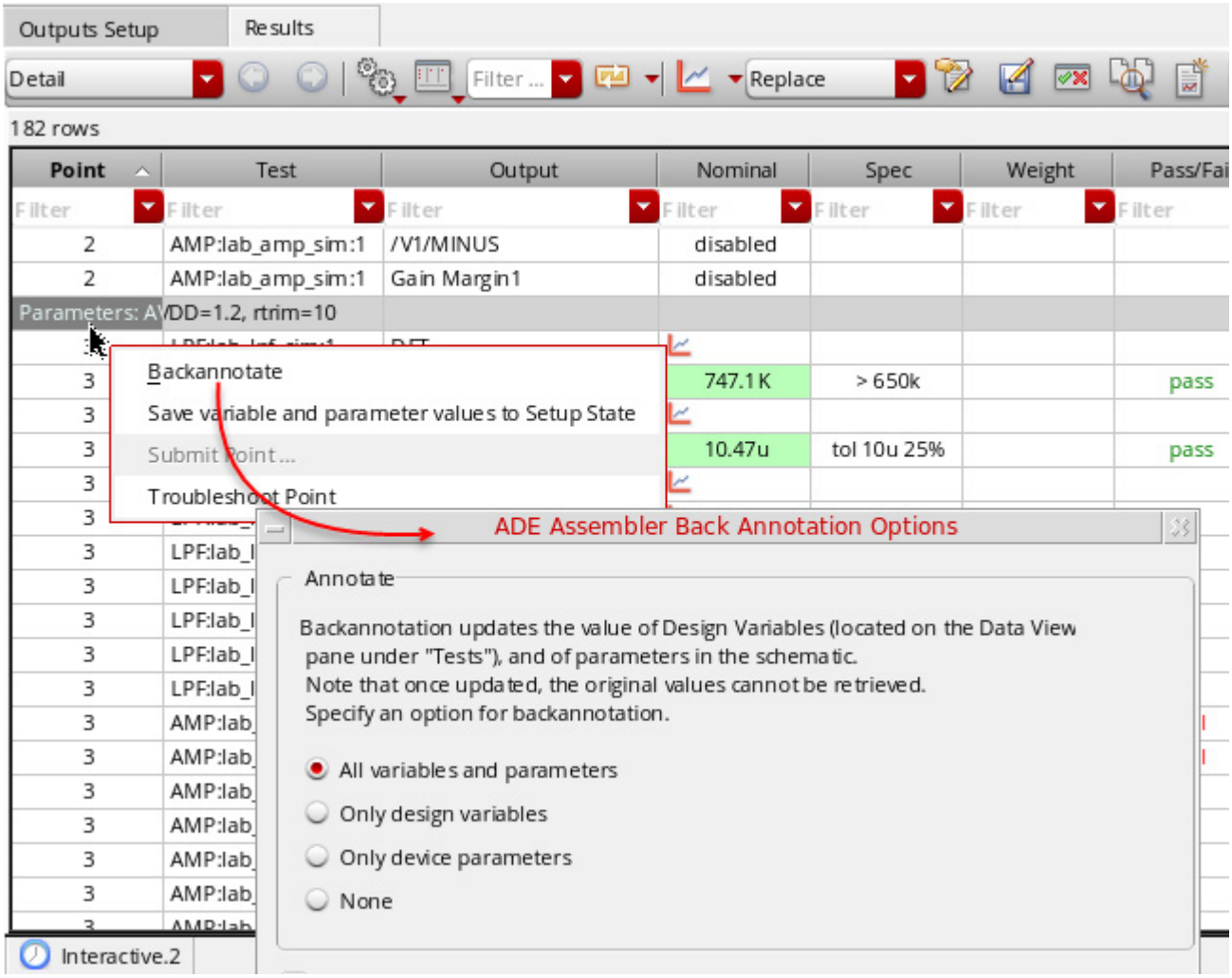
Backannotating Data from Results to Schematic

After running an ADE Assembler simulation, you can backannotate the results from the best or the desired design point to make appropriate changes in the design and simulation setup. When you backannotate the results from a design point, the values of global variables used for that point are copied to the *Design Variables* list in the Data View pane of the ADE Assembler setup and the values of the device parameters are copied to the schematic view of the design.

To backannotate the ADE Assembler results, do the following:

1. Ensure that the `Details` view is open on the *Results* tab.
2. Right-click the gray bar on top of the selected design point and select *Backannotate*.

   The ADE Assembler Back Annotation Options form is displayed, as shown below.



© 2023 Cadence Design Systems, Inc.

3. Select an appropriate option on this form to specify the types of values that you want to backannotate from the results:

   - *All variables and parameters*: This option backannotates the values of all the global variables and device parameters. This option is selected by default.

     Note: You can change the default value by using the defaultBackAnnotationOption environment variable.

   - *Only design variables*: This option backannotates only the global variables.
   - *Only device parameters*: This option backannotates only the device parameters.
   - *None*: This option does not backannotate any value.

4. Select the *Do not show this dialog again* check box if you do not want to view this form every time you backannotate values.

   The check box will be set for the session only. Once you close ADE Assembler and open the next session, you will have to set the check box again.

5. Click *OK* to apply the changes and close the form.

   You cannot restore the original values after the results are backannotated to the schematic or the ADE Assembler setup.

Viewing Waveform Info-Balloons on Schematic

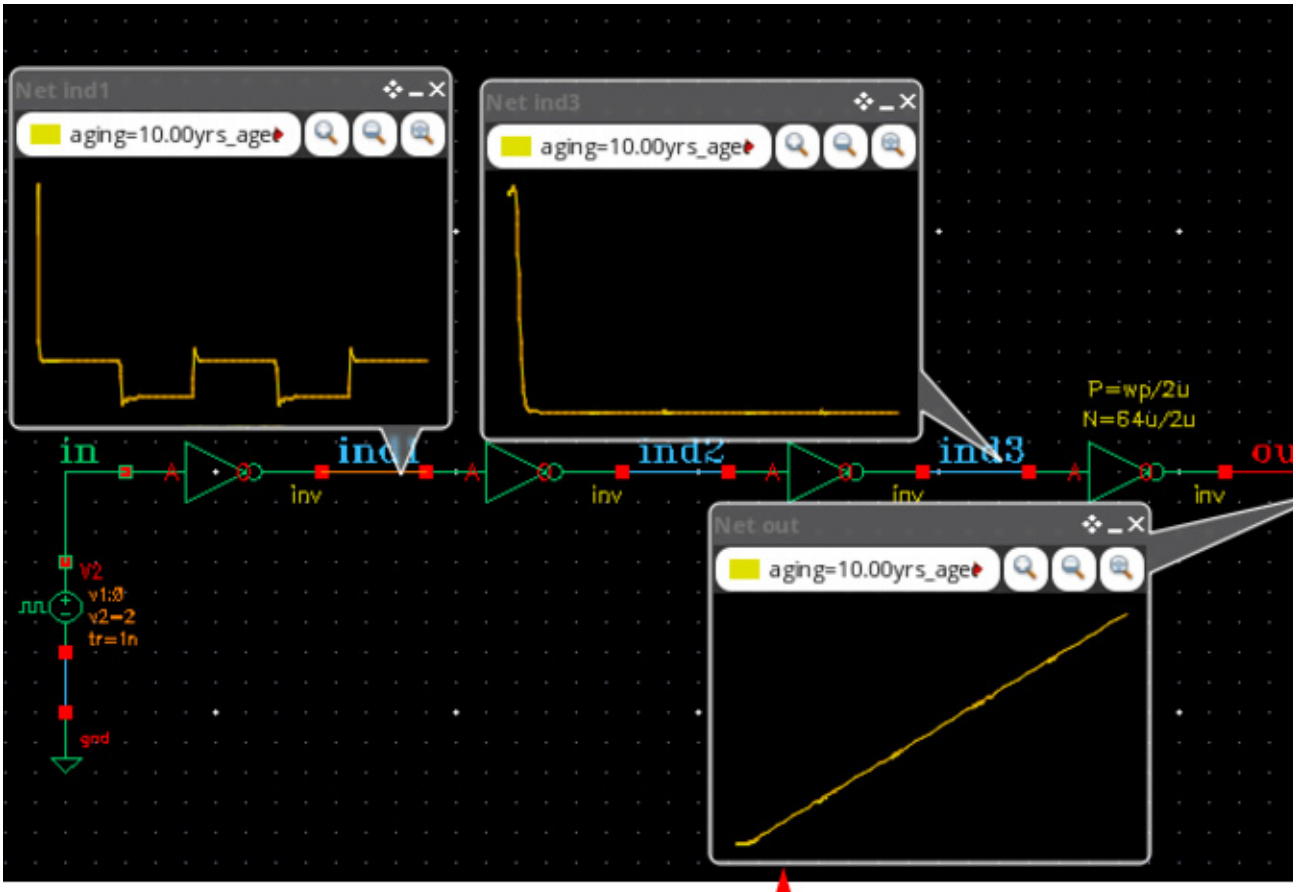
After a simulation run is complete, the output expressions are evaluated and resultant waveforms are plotted in the Virtuoso Visualization and Analysis XL graph. You can also view these waveforms in the schematic design in the form of waveform info-balloons or thumbnails.

To enable the display of waveform info-balloons in schematic, do one of the following:

- Choose View - Info Balloons.
- Press the bindkey Ctrl+B.
- In the schematic window, click the Show/Hide Balloons drop-down ( ) and select Waveforms.

© 2023 Cadence Design Systems, Inc.

After the simulation run is complete, the info-balloons appear when you place the pointer over a terminal or an instance associated with the waveform, as shown in the figure below.



Waveforms for **ind1**, **ind3**, and **out** signals displayed in the form of info-balloons. Notice that the legend is displayed in the balloons by default and includes a drop-down list.

For more details on how to customize waveform info-balloons, refer to Viewing Waveform Info-Balloons on Schematic in *Virtuoso ADE Explorer User Guide*.

© 2023 Cadence Design Systems, Inc.

Re-Evaluation of Expressions and Specifications

After running a simulation, if you add, modify, or delete output expressions or specifications on the *Outputs Setup* tab, you can re-evaluate them without having to rerun the simulation. ADE Assembler provides two modes for re-evaluation, Foreground re-evaluation (FRE) and Distributed re-evaluation (DRE). Once the re-evaluation is complete, the updated results are displayed in the *Results* tab and the results database is automatically updated.

Note the following:

- When you re-evaluate expressions and specifications, the updated results are not plotted. For information about how to plot the results, see Refreshing Graphs.
- If you re-evaluate expressions and specifications for an optimization run, costs are re-calculated for all the design points and a new best point is identified. If the *Save all design points* option was not selected in the Save Options form before running optimization, waveform data may not exist for the new best point.
- You can re-evaluate expressions for scalar outputs even without saving the simulation data because re-evaluation is performed using data saved in the results database. However, re-evaluation does not work for the expressions that require waveform data input.
- Some expressions do require simulation data and might need to be re-simulated.

Related Topics

Re-evaluating Results Using FRE Mode

Distributed Re-Evaluation (DRE) Mode

Re-Evaluating Results for an Older History

Re-evaluating Results Using FRE Mode

Foreground re-evaluation (FRE) mode lets you re-evaluate results by occupying memory on the main ADE process. The maestro session freezes until the re-evaluation process is complete.

To re-evaluate results using FRE mode:

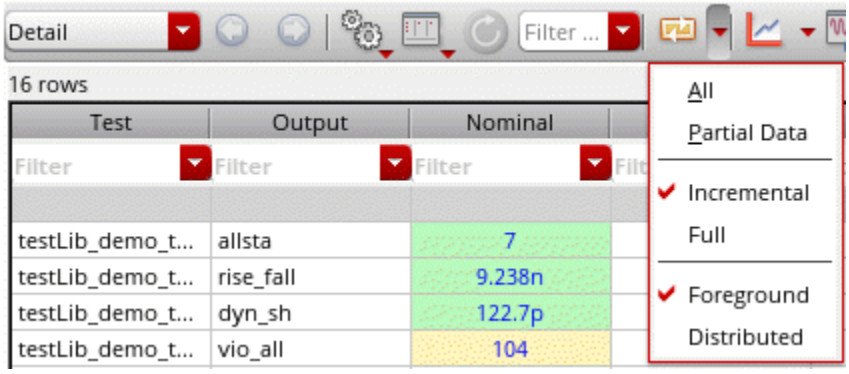
➡ Click on the *Results* tab.

FRE mode is selected by default.

© 2023 Cadence Design Systems, Inc.

Alternatively, follow these steps:

1. Click the drop-down button next to on the *Results* tab to display additional options.



2. Select *Foreground* to enable FRE mode.
3. Select one of the following options:
   - *Incremental*: Re-evaluates the results for revised outputs and the outputs for points that were canceled in a previous re-evaluation run.

     This is the default selection.
   - *Full*: Re-evaluates all results.

   You can set a default value for this setting by using the reEvaluationMode environment variable.
4. Select one of the following options:
   - *All:* Re-evaluates the results for all entries in the Outputs Setup table.

     This is the default selection.
   - *Partial Data*: Re-evaluates the results using the partially or fully complete simulation data. You can use this option to evaluate partial results if a simulation is not complete but there are substantial results that can be used to analyze the results. For any output value that is evaluated on the partial data, the tooltip shows the text *[Evaluated on partial data]*.

Related Topics

Adding or Modifying an Output Expression

Re-Evaluation of Expressions and Specifications

© 2023 Cadence Design Systems, Inc.

Distributed Re-Evaluation (DRE) Mode

Distributed Re-Evaluation (DRE) Mode

Distributed Re-Evaluation (DRE) Mode

Distribution re-evaluation (DRE) mode lets you re-evaluate results in a background process without freezing the maestro session. The re-evaluation is done through netlisting and simulation jobs. The netlisting jobs work on pre-processing of expressions, including dependency calculations for calcVal expressions. The simulation jobs use the pre-processed expressions and perform the re-evaluation.

Note: This feature can be used only when the specified job control mode is LSCS.

DRE mode has the following advantages:

- Enhances performance when using large testcases.
- Lets you distribute jobs to remote machines.
- Lets you re-evaluate the results for multiple histories simultaneously.
- Eets you suspend or resume re-evaluations.

R**e-evaluating Results Using DRE Mode**

To re-evaluate results using DRE mode:

1. Click the drop-down button next to on the *Results* tab to view additional options.
2. Select the *Distributed* option.

   The re-evaluation icon changes to , which reflects that DRE mode is enabled.
3. Click to start re-evaluation.

   When re-evaluation is in progress, the icon changes to . After successful completion, the icon is reverted to .

   However, if you want to interrupt re-evaluation without completion, click icon to stop the re-evaluation run.

   Alternatively, to suspend the re-evaluation run, right-click the related history in the *Run Summary* assistant and select *Suspend*. The history is then highlighted in yellow. Right-click this history and select *Resume* to resume the suspended run.

© 2023 Cadence Design Systems, Inc.

Related Topics

Adding or Modifying an Output Expression

Re-Evaluation of Expressions and Specifications

Re-evaluating Results Using FRE Mode

Distributed Re-Evaluation (DRE) Mode

© 2023 Cadence Design Systems, Inc.

Enabling Re-Evaluation for Unsuccessful Simulations

If a simulation is not completed successfully due to a severe problem, such as a hardware issue in a server on which simulation is running or an abrupt closure of Virtuoso, the status of all or a few results remains stuck at a value that prevents re-evaluation. However, you think that good simulation results are available in the database, but they were not used for expression evaluation.

Here are a few examples of such scenarios:

- Virtuoso stops abruptly when a simulation is nearing completion. When you rerun Virtuoso, the status of outputs in the last history is shown as *simulating* instead of showing the result values.
- Due to a server issue, the simulation for a history was not completed. ADE Assembler receives a failed status for the overall run and shows *running* or *netlisting finished* for some points.

If the simulations can be rerun quickly, it is recommended to run simulation and create a new history with good results. However, if you have large designs and a new simulation takes a significant time to complete, you can first try to forcefully change the status of results and enable re-evaluation of output expressions by filling up the missing values in the results database.

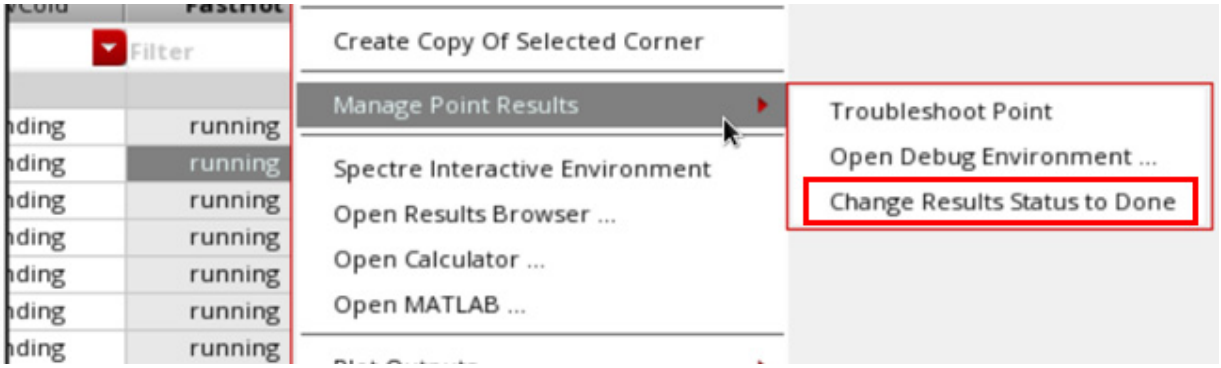
This is a relatively strong measure to enable expression evaluation by modifying the results database and should be used only when there is not other solution.

To re-evaluate the outputs for an unsuccessful simulation:

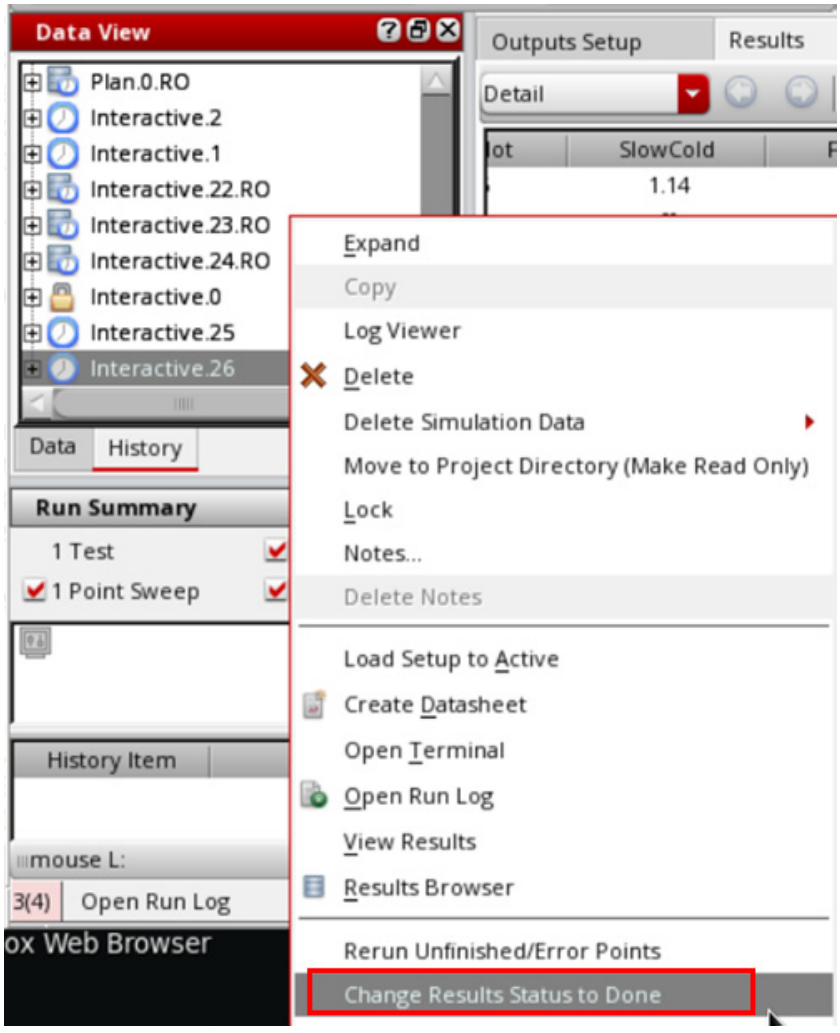
1. Set the forceRdbStatusEnable environment variable to `t`.
2. Load results from the history of the unsuccessful run.
3. To change the status of results of a specific design point in a history, in the *Detail* or *Detail-Transpose* results view, right-click a point that has a status other than *pending*

© 2023 Cadence Design Systems, Inc.

or *canceled* and choose *Change Results Status to Done* from the context-sensitive menu.



Alternatively, to update the status for all points, right-click a history and choose *Change Results Status to Done* from the context-sensitive menu.



© 2023 Cadence Design Systems, Inc.

The *Change Results Status to Done* command changes the status of incomplete points, which do not have status that is mentioned in the forceRdbStatusIgnoreList environment variable, to *done* or another status value set by the forceRdbStatusTo environment variable. Irrespective of the value of `forceRdbStatusTo`, the command is always named *Change Results Status to Done*.

After the points are internally marked as *done*, you can re-evaluate the outputs without requiring resimulation.

Note: This feature marks the points as *done* even though the simulation results are not complete, so it should be expected that some of the points will fail during re-evaluation and you might see various error messages during re-evaluation. This is the expected behavior.

Related Topics

forceRdbStatusEnable

forceRdbStatusIgnoreList

forceRdbStatusTo

maeForceRdbStatus

Re-Evaluating Results for an Older History

By default, the reEvalOnlyMostRecentHistory environment variable is set to `t` and you can re-evaluate the results only for the recent history. However, if you set this variable to `nil`, you can re-evaluate the results for an older history item by viewing its results.

To re-evaluate the results for an older history, do the following:

1. On the *History* tab of the Data View pane, right-click a history name and choose *View Results*.

   The results of the history item are displayed on the *Results* tab of the *Outputs* pane.

2. Click the drop-down button next to the command on the *Results* tab of the *Outputs* pane, and do one of the following:

   - Select *All* to re-evaluate both expressions and specifications.
   - Select *Partial Data* to re-evaluate both expressions and specifications using the partially complete simulation data.

© 2023 Cadence Design Systems, Inc.

ADE Assembler uses the settings from the active setup to re-evaluate the results for the history item and the updated results are displayed on the *Results* tab. However, the settings specified by the following environment variables are also considered to decide how to re-evaluate and save the new results:

- confirmReEvaluationWhen
- reEvaluationAgeHoursThreshold
- reEvaluationRemovingOutputsThreshold
- reEvaluationWhenActiveAndHistoryTestsDiffer

See also

- Adding or Modifying an Output Expression on page 367

© 2023 Cadence Design Systems, Inc.

Viewing the Spec Summary

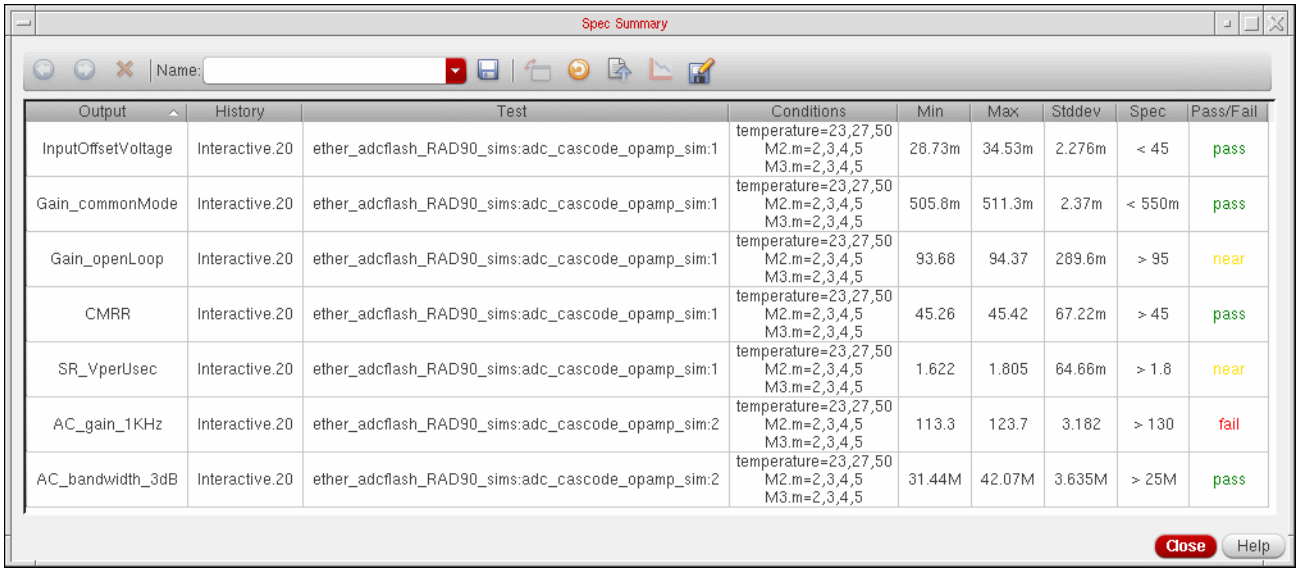
To view the specification (spec) summary, do the following on the Results tab of the Outputs pane:

Note: You can view the spec summary only for the results of a Single Run, Sweeps and Corners or Monte Carlo Sampling simulation run. You cannot view the spec summary for the results of a Global Optimization, Local Optimization, Size Over Corners or Sensitivity Analysis simulation run.

➤ Do one of the following on the currently active Results tab for a Single Run, Sweeps and Corners or Monte Carlo simulation run:

- ❑ Choose *Create – Spec Summary*.
- ❑ Click the toolbar button on the ADE Assembler window.
- ❑ Click the button on the Results tab.

The spec summary for the results in the currently active Results tab is displayed in the Spec Summary form.



Output	History	Test	Conditions	Min	Max	Stddev	Spec	Pass/Fail
InputOffsetVoltage	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:1	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	28.73m	34.53m	2.276m	< 45	pass
Gain_commonMode	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:1	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	505.8m	511.3m	2.37m	< 550m	pass
Gain_openLoop	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:1	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	93.68	94.37	289.6m	> 95	near
CMRR	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:1	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	45.26	45.42	67.22m	> 45	pass
SR_VperUsec	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:1	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	1.622	1.805	64.66m	> 1.8	near
AC_gain_1KHz	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:2	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	113.3	123.7	3.182	> 130	fail
AC_bandwidth_3dB	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:2	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	31.44M	42.07M	3.635M	> 25M	pass

The toolbar in the Spec Summary form is described in the following table:

© 2023 Cadence Design Systems, Inc.

Table 19-8 Spec Summary Toolbar

Icon	Name	Description
	Back to Summary View	Switches from the detail view to the summary view. For more information, see Viewing Specification Results in the Results Tab.
	Show Detail View	Displays the detailed results for the selected specifications in the detail view. For more information, see Viewing the Detailed Results for Specifications.
	Delete	Deletes the selected specifications. For more information, see Viewing the Detailed Results for Specifications.
	Save Spec Summary	Saves the spec summary. For more information, see Saving a Spec Summary.
	Transpose Table	Displays the results in the detail view in horizontal or vertical format. For more information, see Viewing the Detailed Results for Specifications.
	Update with Latest Results Data	Updates the spec summary with the latest results from the history item selected for each specification. For more information, see Updating the Spec Summary with the Latest Results.
	Recreate for the Current History Results	Recreates the spec summary using the results in the active Results tab. For more information, see Recreating the Spec Summary from the Results in the Active Results Tab.
	Plot Signals	Plots the results for the selected row. For more information, see Plotting the Results for Specifications.

© 2023 Cadence Design Systems, Inc.

Icon	Name	Description
	Export to CSV or HTML File	Exports a spec summary to a HTML or comma-separated values (CSV) file For more information, see Exporting a Spec Summary to a HTML or CSV File.

The columns in the Spec Summary form are described in the following table:

Table 19-9 Spec Summary Columns

Column	Description
Output	Displays the names of the outputs for which the specifications are evaluated.
History	Displays the names of the history items from which the results of outputs are displayed.
Test	Displays the names of the tests for which the results of outputs are displayed.
Conditions	Displays the values of each swept parameter as a list or range of values. For example, if a parameter `p` is swept through a list of values `x`, `y`, and `z`, then the *Conditions* column displays the values as `p=x,y,z` If a parameter `p` is swept through a range of values, the *Conditions* column displays the values as *`p=startValue:increment:stopValue`*
Min	Displays the minimum measured value for each output.
Max	Displays the maximum measured value for each output.

© 2023 Cadence Design Systems, Inc.

Column	Description
Std Dev	Displays the standard deviation of the measured values for each output. You can hover the mouse pointer over a standard deviation value to view the following information in a pop-up: ■ The number of simulations for the output expression. ■ The mean of the measured values for all simulations for the output expression. ■ The variance in the measured values. **Note:** For Single Run, Sweeps and Corners, this is the population standard deviation. For other run modes, this is the sample standard deviation.
Spec	Displays the specifications for outputs.
Pass/Fail	Displays a *pass*, *near* or *fail* status for each specification. ■ *pass* means that all the measured values are within the limits defined by the specification. ■ *near* means that one or more measured values are no more than 10% outside the target value of the specification. ■ *fail* means that one or more measured values are greater than 10% outside the target value of the specification. To investigate the measured values further, see Viewing the Detailed Results for Specifications.

© 2023 Cadence Design Systems, Inc.

Comparing Results Using the Spec Comparison Form

You can use the Spec Comparison form to compare measured values of output expressions for:

- Any two history items.
- Any two tests in the same history item or in two different history items.
- Any two design points in the same history item or in two different history items.

For more information, see the following topics:

- Opening the Spec Comparison Form
- Comparing Results for History Items
- Comparing Results for Specific Tests
- Comparing Results for Specific Design Points
- Comparing the Detailed Results for Output Expressions
- Hiding and Showing the Comparison Data for Tests
- Hiding and Showing an Output Expression in the Spec Comparison
- Updating the Spec Comparison with the Latest Results
- Sorting Data in the Spec Comparison Form
- Saving a Spec Comparison
- Opening a Spec Comparison
- Deleting a Spec Comparison
- Exporting a Spec Comparison to a HTML or CSV File

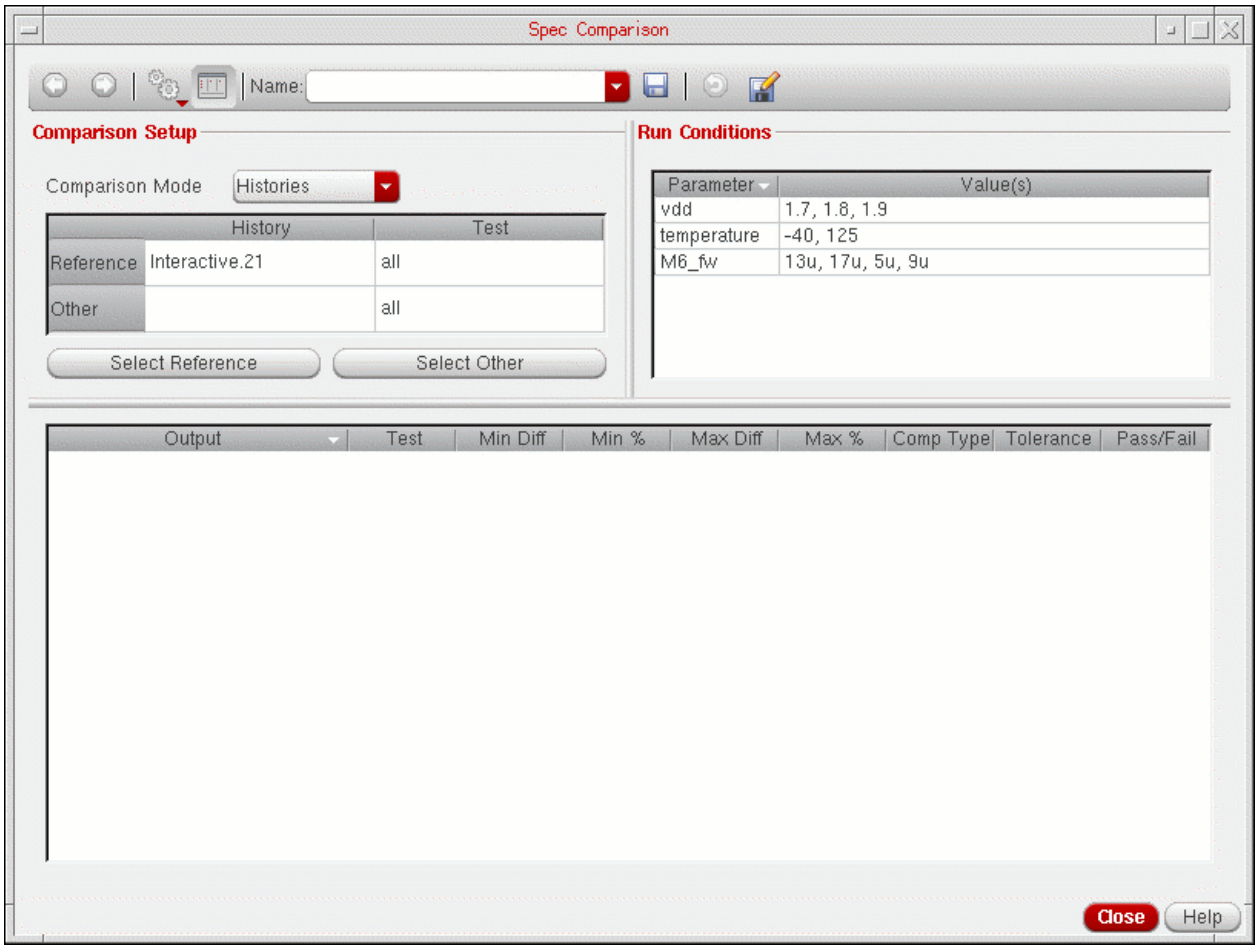
© 2023 Cadence Design Systems, Inc.

Opening the Spec Comparison Form

To open the Spec Comparison form, do one the following:

- Choose *Create – Spec Comparison*.
- Click the toolbar button on the ADE Assembler window.
- Click the button on the Results tab.

  The Spec Comparison form appears.



The toolbar in the Spec Comparison form is described in the following table:

© 2023 Cadence Design Systems, Inc.

Table 19-10 Spec Comparison Toolbar

Icon	Name	Description
	Back to Summary View	Switches from the detail view to the summary view. For more information, see Comparing the Detailed Results for Output Expressions.
	Show Detail View	Displays the comparison of detailed results for the selected output expression in the detail view. For more information, see Comparing the Detailed Results for Output Expressions.
	Select which tests are displayed in the table	Hides or shows the comparison data for specific tests. For more information, see Hiding and Showing the Comparison Data for Tests.
	Show/Hide Setup Conditions Tables	Hides or shows the *Comparison Setup* and *Run Conditions* group boxes in the Spec Comparison form.
Name	Name of Saved Spec Comparison	Lets you do the following: ■ Specify the name by which a spec comparison is to be saved. For more information, see Saving a Spec Comparison. ■ Open an existing spec comparison. For more information, see Opening a Spec Comparison
	Save Spec Comparison	Saves the spec comparison with the name specified in the *Name* field. For more information, see Saving a Spec Comparison.
	Update with Latest Results Data	Updates the spec comparison with the latest results from the history items selected for comparison. For more information, see Updating the Spec Comparison with the Latest Results.
	Export to CSV or HTML File	Exports a spec comparison to a HTML or comma-separated values (CSV) file. For more information, see Exporting a Spec Comparison to a HTML or CSV File.

© 2023 Cadence Design Systems, Inc.

The fields in the Spec Comparison form are described in the following table:

Field	Description
Comparison Setup	
Comparison Mode	Specifies the comparison mode. Do one of the following: ■ Select *Histories* to compare measured values of output expressions for: ❑ Any two history items. For more information, see Comparing Results for History Items. ❑ Any two tests in the same history item or in two different history items. For more information, see Comparing Results for Specific Tests. ■ Select *Design Points* to compare measured values of output expressions for any two design points in the same history item or in two different history items. You can compare two design points in the same test or in two different tests. For more information, see Comparing Results for Specific Design Points.
History	Displays the history items for which you are comparing the measured values of output expressions. For more information, see Comparing Results for History Items.
Test	Displays the tests in the history items for which you are comparing the results. For more information, see Comparing Results for Specific Tests.
Point	Displays the design points in the history items for which you are comparing the measured values of output expressions. This column is displayed only if the *Design Points* is selected in the Comparison Mode drop-down list. For more information, see Comparing Results for Specific Design Points.

© 2023 Cadence Design Systems, Inc.

Field	Description
Run Conditions	Displays the conditions that were used for the simulation run in the reference history item. The values of each swept parameter in the reference history item is displayed as a list or range of values. For example, if a parameter `p` is swept through a list of values `x`, `y`, and `z`, then the *Value(s)* column displays the values as `p=x,y,z` If a parameter `p` is swept through a range of values, the *Value(s)* column displays the values as *`p=startValue:increment:stopValue`*
Parameter	Displays the names of swept parameters.
Value(s)	Displays the values of swept parameters.

Comparing Results for History Items

To compare results for history items, do the following:

1. From the *Comparison Mode* drop-down list, choose *Histories*.
2. The history item for which you are viewing the results in the Results tab is displayed as the default reference history item. Do one of the following to select a different reference history item:
   - Double-click in the *History* column in the *Reference* row and select a different history item from the drop-down list.
   - Click the *Select Reference* button, click the tab for the history item in the Results tab, then click anywhere on the results displayed for the history item.

   Note: The Spec Comparison form does not allow comparison of results of history items for optimization runs. So you cannot select history items for optimization runs when the comparison mode is *Histories*.
3. Do one of the following to select the history item whose results you want to compare with the reference history item.
   - Double-click in the *History* column in the *Other* row and select the history item from the drop-down list.

© 2023 Cadence Design Systems, Inc.

- Click the *Select Other* button, click the tab for the history item in the Results tab, then click anywhere on the results displayed for the history item.

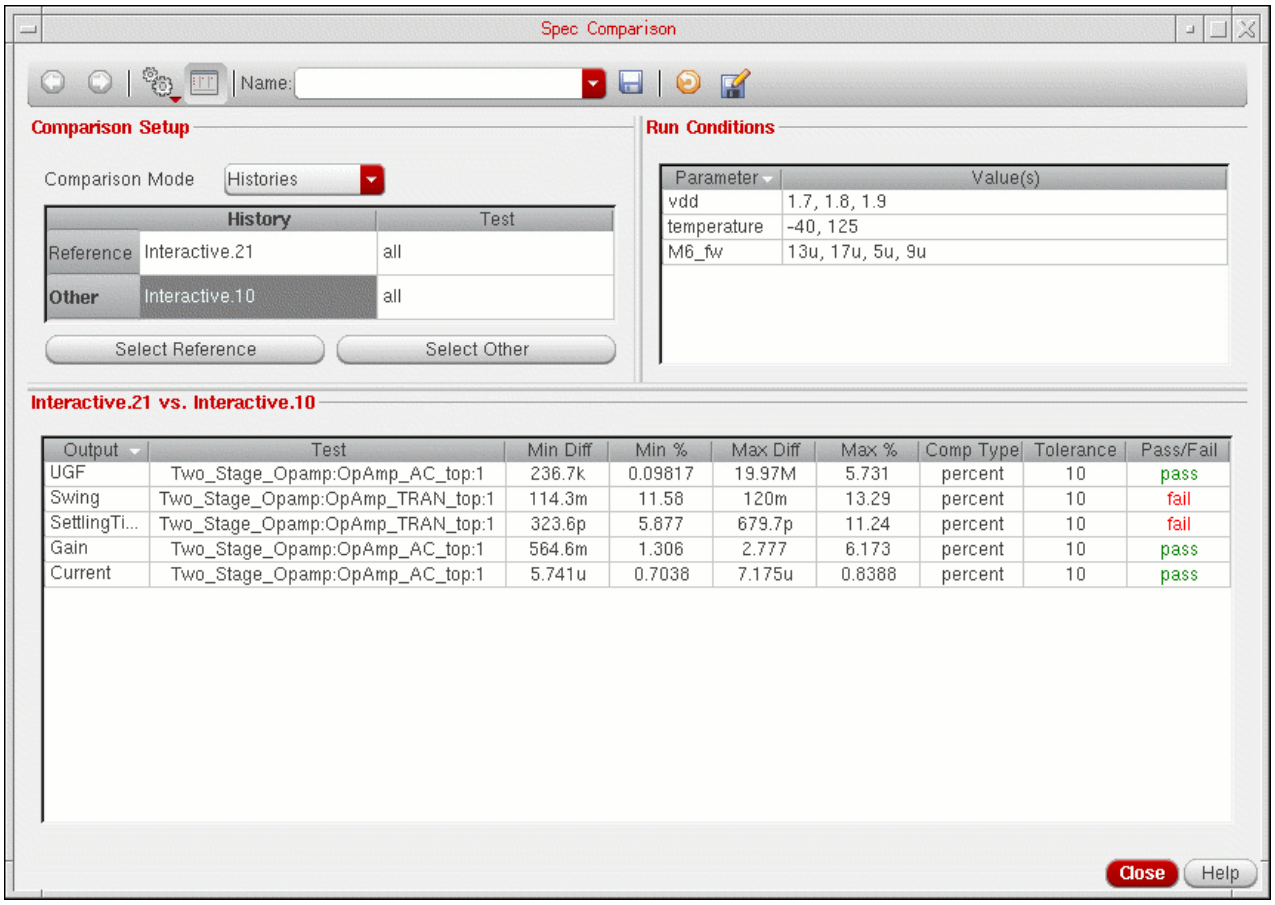
Note: Earlier, in the *Other* row, you could select only that history item that has the same set of conditions (displayed in the *Run Conditions* field) as the reference history item. Starting IC 6.1.5, you can select a history with different run conditions, for example, different sweep values for variables.

Tip

After you click *Select Other*, you can select any number of history items in the Results tab without clicking the button again. For example, you can click the *Select Other* button and select a history item in the Results tab to view the results comparison. Once you are done, you can select another history item in the Results tab without clicking *Select Other* again.

The comparison of results for the history items is displayed in the Spec Comparison form.

Figure 19-21 Spec Comparison for History Items



© 2023 Cadence Design Systems, Inc.

Note: By default, the Spec Comparison form displays the comparison of results for all the tests in the reference and other history item. The text *all* in the *Test* column indicates that the results for all the tests will be compared. For information about viewing the spec comparison for specific tests, see Comparing Results for Specific Tests.

The columns that are displayed in the Spec Comparison form when you compare the specifications for two history items are described in the following table.

Table 19-11 Spec Comparison Columns for History Items and Tests

Column	Description
Output	Displays the names of output expressions for which the results are being compared. **Note:** Output expressions that evaluate to waveforms are not displayed in the Spec Comparison form.
Test	Lists the tests used to generate the results for the output expressions.
Min Diff	Displays the minimum difference between the measured values across all the points in the two history items.
Min %	Displays the percentage error between the two measured values in the history items for which the minimum difference is reported.
Max Diff	Displays the maximum difference between the measured values across all the points in the two history items.
Max %	Displays the percentage error between the two measured values in the history items for which the maximum difference is reported.

© 2023 Cadence Design Systems, Inc.

Column	Description

Comp Type	Comparison type, selected from a drop-down list, that specifies how the results from the two history items are compared. Pass/fail criteria are as follows:
	■ *absolute*—the absolute value of the difference in measured values for the two history items must be less than the tolerance specified. If *absolute*, the output expression will have a pass status if:
	`abs(refHistoryValue - otherHistoryValue) < Tolerance`
	■ *percentage*—the absolute value of the difference in measured values for the two history items must be less than the tolerance multiplied by the absolute value of the measured value for the reference history item. If *percentage*, the output expression will have a pass status if:
	`abs(refHistoryValue - otherHistoryValue) < abs(refHistoryValue * Tolerance)`
	■ *range*—the absolute value of the difference in measured values for the two history items must be less than the tolerance multiplied by the absolute value of the difference between the maximum and minimum measured values for the reference history item. If *range*, the output expression will have a pass status if:
	`abs(refHistoryValue - otherHistoryValue) < abs(maxrefHistoryValue - minOtherHistoryValue) * Tolerance`
	■ *absolute/percentage*—the absolute value of the difference in measured values for the two history items must be less than the specified tolerance, or less than the tolerance multiplied by the absolute value of the measured value for the reference history item (see equations for *absolute* and *percentage*, above). This comparison type is useful for comparing measured values near zero.
	■ *none*—does not apply the tolerance to determine pass/fail criteria. The pass/fail status will also not be displayed in the *Pass/Fail* column.

© 2023 Cadence Design Systems, Inc.

Column	Description
Tolerance	Specifies the tolerance value that determines the pass or fail status of the output expression. **Note:** Specify a positive value if the comparison type is *absolute* or *range*, and a positive percentage if the comparison type is *percentage*.
Pass/Fail	Displays the pass or fail status of the output expression based on the specified comparison type and the tolerance.

See also:

- Comparing the Detailed Results for Output Expressions
- Hiding and Showing the Comparison Data for Tests
- Hiding and Showing an Output Expression in the Spec Comparison
- Updating the Spec Comparison with the Latest Results
- Sorting Data in the Spec Comparison Form
- Exporting a Spec Comparison to a HTML or CSV File

Comparing Results for Specific Tests

By default, the Spec Comparison form displays the comparison of measured values of output expressions for all the tests in the reference and other history item. The text *all* in the *Test* column indicates that the results for all the tests will be compared.

You can compare the results for any two tests in the same history item or in two different history items.

To compare the results for specific tests, do the following:

1. Select the reference and other history item by performing steps 1 to 3 described in Comparing Results for History Items.

   Note: To compare the results for any two tests in the same history item, select the same history item as the reference and other history item.

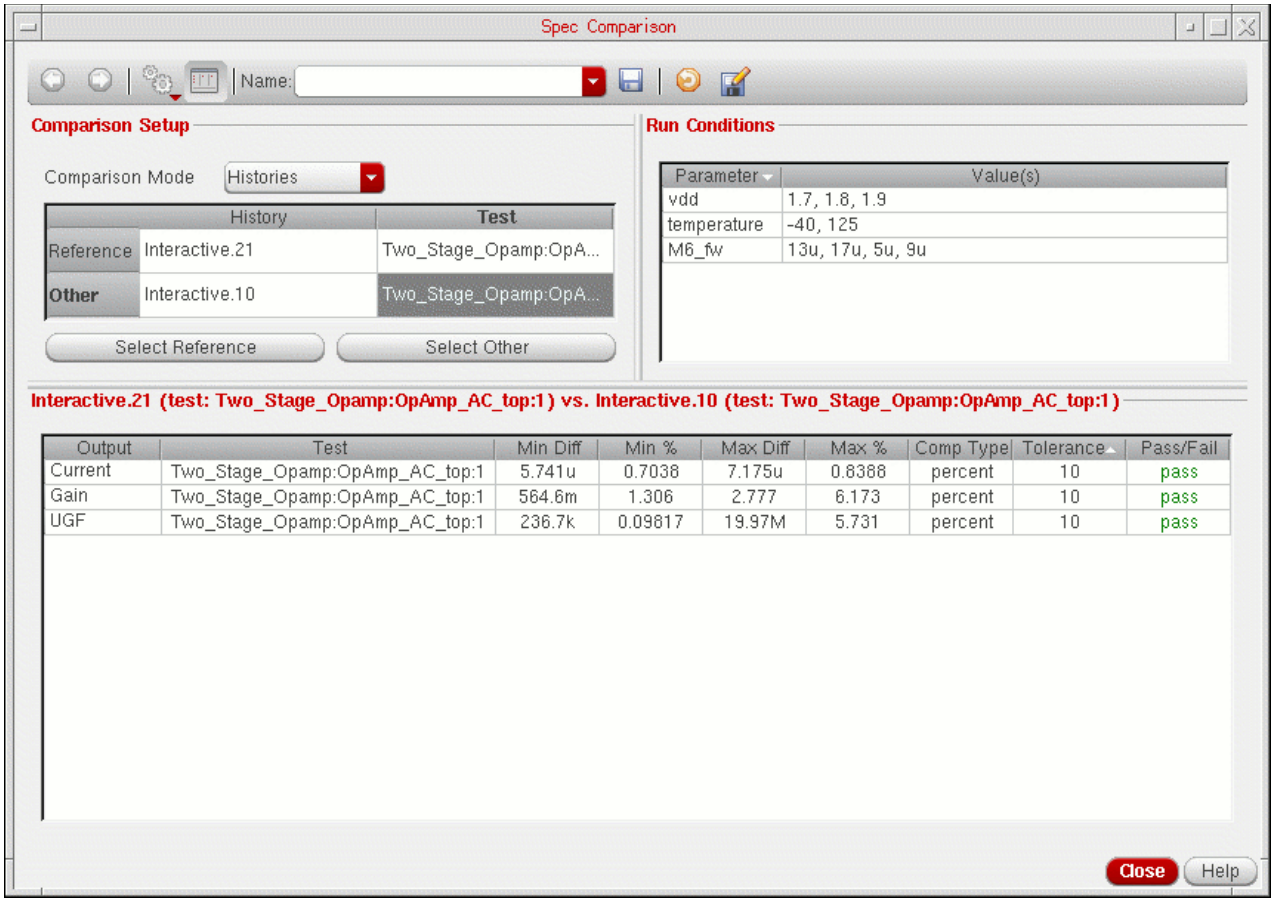
2. Double-click in the *Test* column in the *Reference* row and select a test name for the reference history item from the drop-down list.

© 2023 Cadence Design Systems, Inc.

3. Double-click in the *Test* column in the *Other* row and select a test name for the other history item from the drop-down list.

   The comparison of results for the tests is displayed in the Spec Comparison form.

Figure 19-22 Spec Comparison for Tests

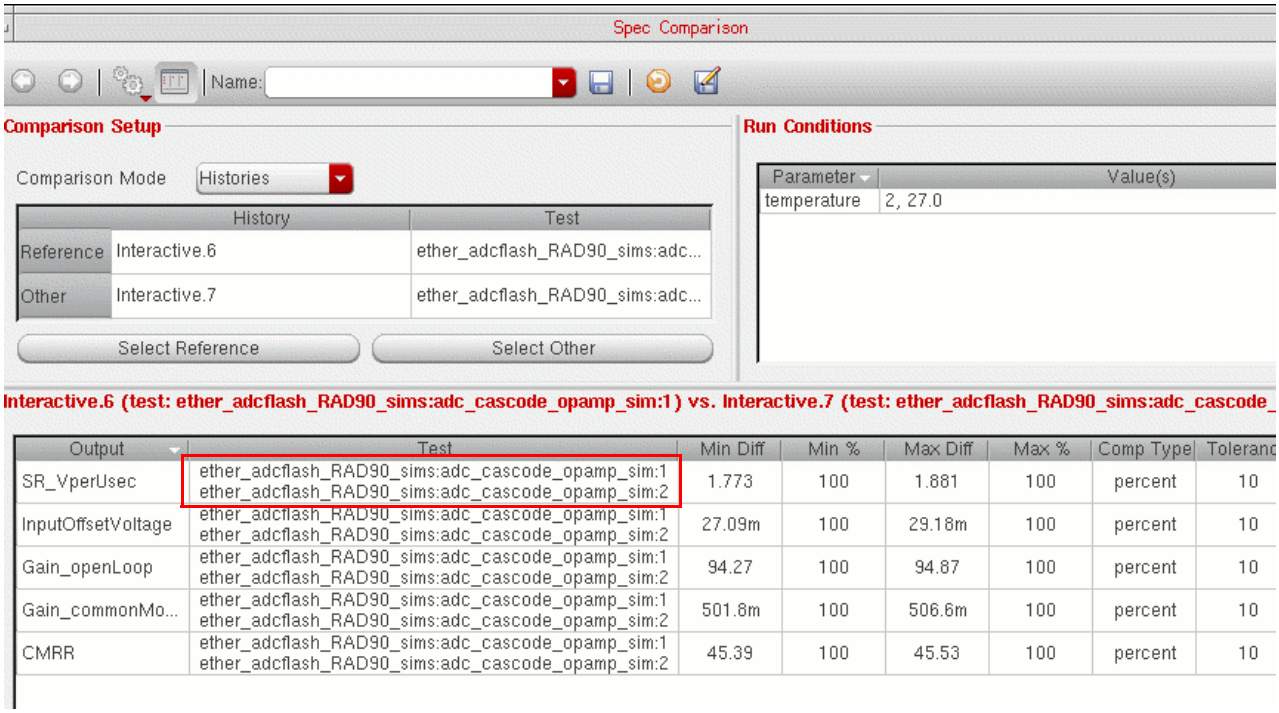


The columns that are displayed in the Spec Comparison form when you compare the specifications for two tests are described in Table 19-11 on page 1202.

When you compare results of different tests in two history items, the *Test* column shows the names of both the referenced test and the other test, as shown below.

© 2023 Cadence Design Systems, Inc.

Figure 19-23 Spec Comparison for Different Tests



See also:

- Comparing the Detailed Results for Output Expressions
- Hiding and Showing an Output Expression in the Spec Comparison
- Updating the Spec Comparison with the Latest Results
- Sorting Data in the Spec Comparison Form
- Exporting a Spec Comparison to a HTML or CSV File

Comparing Results for Specific Design Points

By default, the Spec Comparison form displays the comparison of measured values of output expressions for all the design points in the reference and other history item.

You can compare the results for any two design points in the same history item or in two different history items.

To compare the results for specific design points, do the following:

1. From the *Comparison Mode* drop-down list, choose *Design Points*.

© 2023 Cadence Design Systems, Inc.

2. Select the reference and other history item by performing steps 2 and 3 described in Comparing Results for History Items.

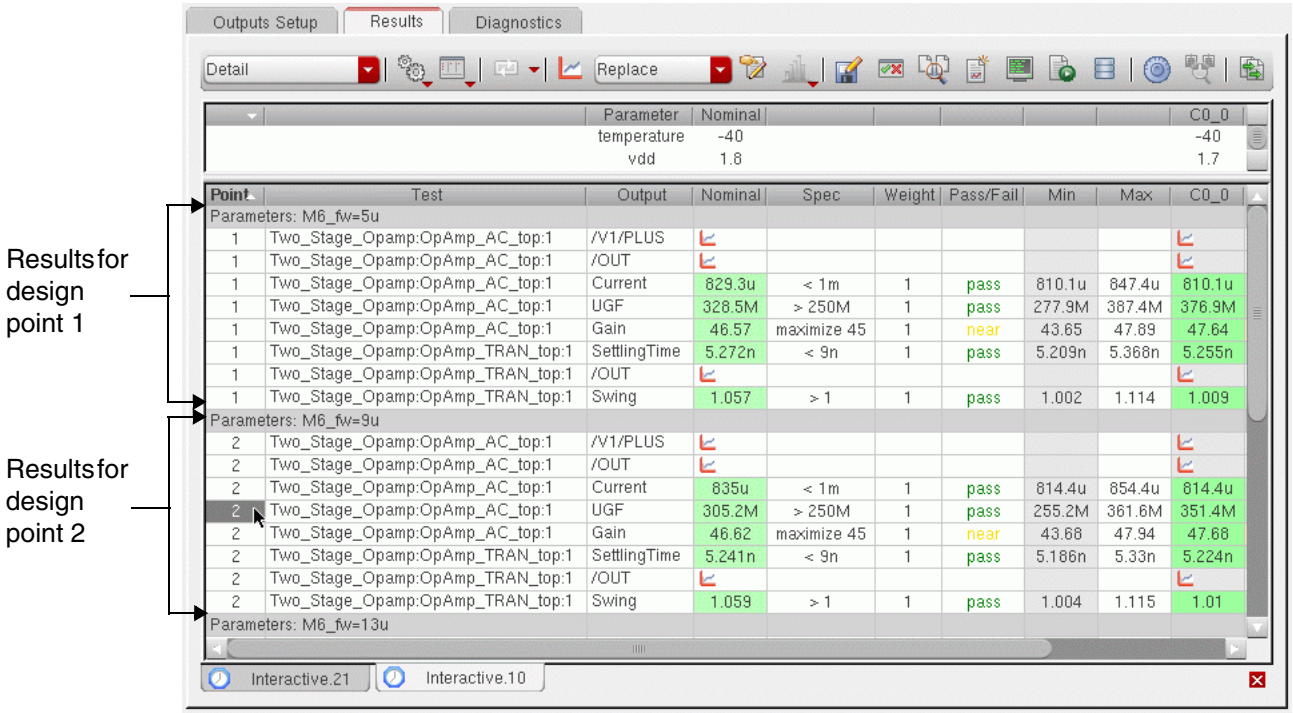
3. By default, the Spec Comparison form displays the comparison of results for design points in all the tests in the reference and other history item. To compare the results for design points in specific tests, select the tests by performing steps 2 and 3 described in Comparing Results for Specific Tests.

4. By default, the ID of the first design point is displayed as the default design point for the reference history item. If the reference history item is for an optimization run, the ID of the best design point in the reference history item is displayed as the default point for the reference history item.

   Do one of the following to select a different design point for the reference history item:

   - Double-click in the *Point* column in the *Reference* row and specify a different design point.
   - Click the *Select Reference* button and click the results for a specific design point for the history item in the Results tab.

     The following figure shows an example of selecting design point `2` for comparing results by clicking on the results for design point `2`.



5. Do one of the following to select a design point for the other history item:

© 2023 Cadence Design Systems, Inc.

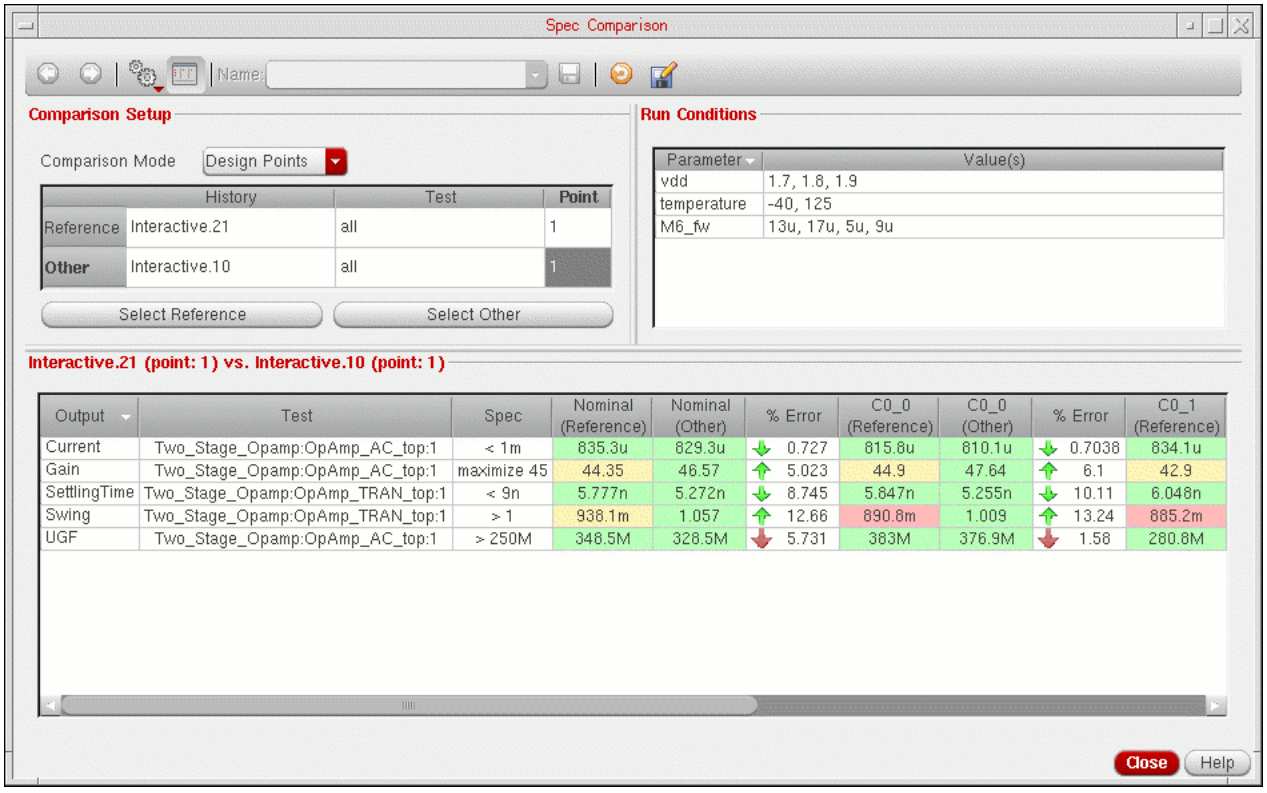
- Double-click in the *Point* column in the *Other* row and select a design point for the other history item from the drop-down list.
- Click the *Select Other* button and click the results for a specific design point for the history item in the Results tab.

Tip

After you click *Select Other*, you can select any number of design points in the Results tab without clicking the button again. For example, you can click the *Select Other* button, and select a design point in the Results tab to view the results comparison. Once you are done, you can select another design point in the Results tab without clicking *Select Other* again.

The comparison of results for the design points is displayed in the Spec Comparison form.

Figure 19-24 Spec Comparison for Design Points



The columns that are displayed in the Spec Comparison form when you compare the results for two design points are described in the following table.

© 2023 Cadence Design Systems, Inc.

Table 19-12 Spec Comparison Columns for Design Points

Column	Description
Output	Displays the names of output expressions for which the results are being compared. **Note:** Output expressions that evaluate to waveforms are not displayed in the Spec Comparison form.
Test	Lists the tests used to generate the results for output expressions.
Spec	Displays specification information.
Nominal (Reference)	Displays the status and measured value for expressions at the nominal corner in the reference history item. The color of a cell indicates the status of the result with respect to the specification: ■ Green indicates a value within the specifications. ■ Yellow indicates a value that is no more than 10% outside the target value of the specification. ■ Red indicates a value that is more than 10% outside the target value of the specification.
Nominal (Other)	Displays the status and measured value for expressions at the nominal corner in the other history item.

© 2023 Cadence Design Systems, Inc.

Column	Description
% Error	Displays the status (using arrow icons) and percentage error of the measured values at the nominal corner in the other history item relative to the values in the reference history item. The arrow color indicates how well the specification is satisfied by the value in the other history item relative to the value in the reference history item. A green arrow indicates that the value in the other history item more effectively meets the specification relative to the reference history item, whereas a red arrow indicates that the value in the other history item is less effective at meeting the specification. The arrow direction indicates the magnitude of the value in the other history item relative to the value in the reference history item. An up arrow indicates that the value in the other history item is higher relative to the reference history item, whereas a down arrow indicates that the value in the other history item is lower. For example, the icon indicates that the value in the other history item more effectively meets the specification and has a higher value relative to the value in the reference history item.
cornerName (Reference)	Displays the status and measured value for expressions at each corner in the reference history item. The color of a cell indicates the status of the result with respect to the specification: ■ Green indicates a value within the specifications. ■ Yellow indicates a value that is no more than 10% outside the target value of the specification. ■ Red indicates a value that is more than 10% outside the target value of the specification.
cornerName (Other)	Displays the status and measured value for expressions at each corner in the other history item.

© 2023 Cadence Design Systems, Inc.

Column	Description
% Error	Displays the status (using arrow icons) and percentage error of the measured values at the corner in the other history item relative to the values in the reference history item. The arrow color indicates how well the specification is satisfied by the value in the other history item relative to the value in the reference history item. A green arrow indicates that the value in the other history item more effectively meets the specification relative to the reference history item, whereas a red arrow indicates that the value in the other history item is less effective at meeting the specification. The arrow direction indicates the magnitude of the value in the other history item relative to the value in the reference history item. An up arrow indicates that the value in the other history item is higher relative to the reference history item, whereas a down arrow indicates that the value in the other history item is lower. For example, the icon indicates that the value in the other history item more effectively meets the specification and has a higher value relative to the value in the reference history item.

See also:

- Comparing the Detailed Results for Output Expressions
- Hiding and Showing an Output Expression in the Spec Comparison
- Updating the Spec Comparison with the Latest Results
- Sorting Data in the Spec Comparison Form
- Exporting a Spec Comparison to a HTML or CSV File

Comparing the Detailed Results for Output Expressions

You can compare the detailed results for output expressions when the *Histories* option is selected in the *Comparison Mode* drop-down list.

To compare the detailed results for an output expression, do one of the following:

- Select the row for the output expression and click the button.

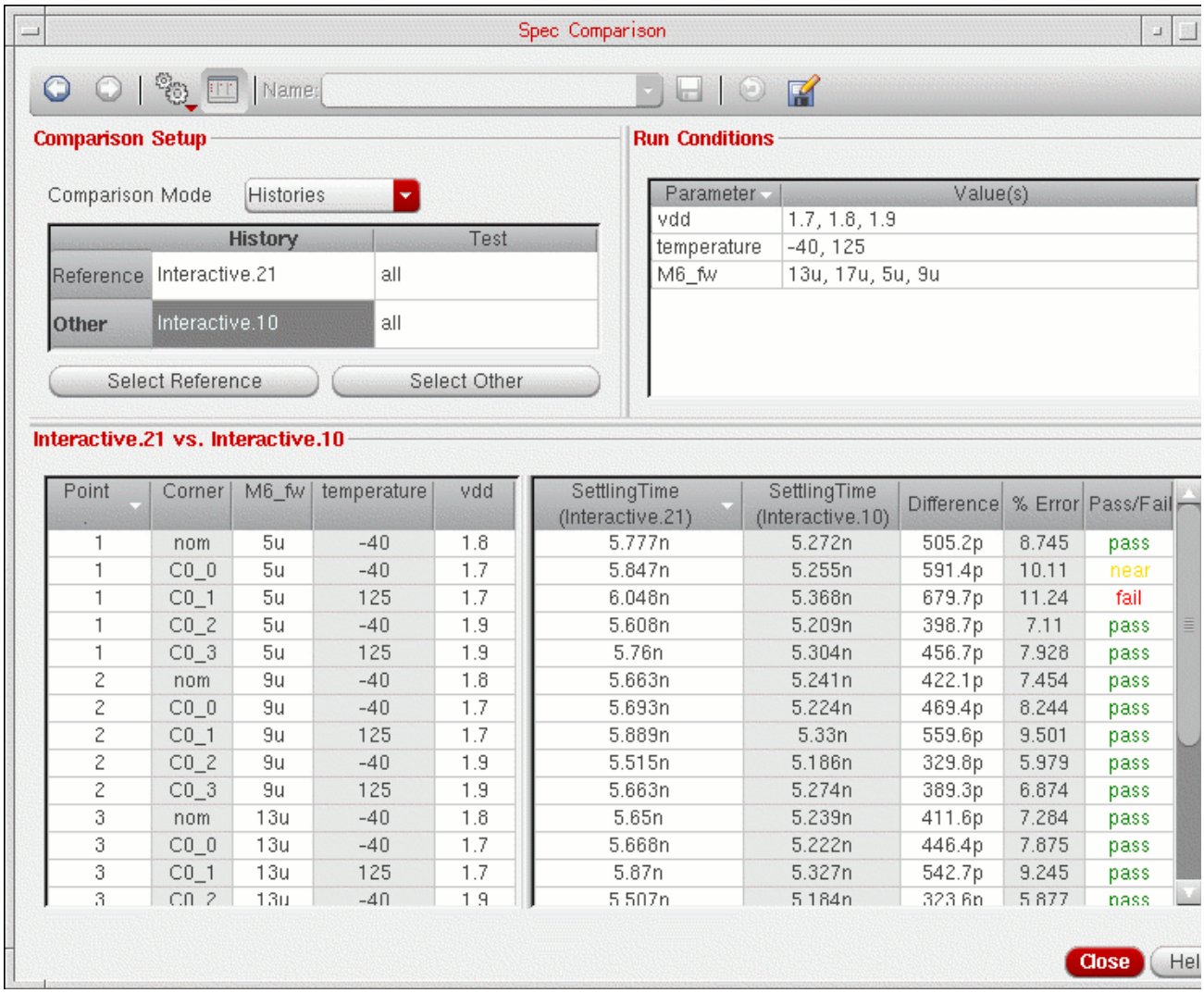
© 2023 Cadence Design Systems, Inc.

- Double-click the row for the output expression.
- Right-click the row for the output expression and choose *Show Detail View*.

The comparison of the detailed results are displayed in the detail view.

Note: You cannot compare the detailed results for an output expression if the reference or other history item does not exist.

Figure 19-25 Spec Comparison Detailed View



The columns in the detail view in the Spec Comparison form are described in the following tables.

© 2023 Cadence Design Systems, Inc.

Table 19-13 Columns in Table on Left Side of Spec Comparison Detailed View

Column	Description
Point	Displays the design point number.
Corner	Displays the name of the corner.
paramName	Displays the values of swept parameters at each design point.

Table 19-14 Columns in Table on Right Side of Spec Comparison Detailed View

Column	Description
exprName (*referenceHistoryName*)	Displays the measured value for the expression in the reference history item at each design point.
exprName (*otherHistoryName*)	Displays the measured value for the expression in the other history item at each design point.
Difference	Displays the difference between the measured values for the expression in the reference history item and the other history item.
% Error	Displays the percentage error of the measured values for the expression in the other history item relative to the values in the reference history item.
Pass/Fail	Displays a *pass*, *near* or *fail* status for the specification. ■ *pass* means that all the measured values are within the limits defined by the specification. ■ *near* means that the measured values are no more than 10% outside the target value of the specification. ■ *fail* means that the measured values are greater than 10% outside the target value of the specification.

To go back to the summary view, do one of the following in the detail view:

- Click the button.
- Right-click any row and choose *Go back*.

© 2023 Cadence Design Systems, Inc.

Hiding and Showing the Comparison Data for Tests

By default, the Spec Comparison form displays the comparison data for all the tests in the reference and other history item.

Click the button. A check mark next to a test indicates that data for that test is being displayed.

To hide the comparison data for a test, do the following:

➤ Click the button and choose the test.

To unhide the data click the button once again and choose the test.

Hiding and Showing an Output Expression in the Spec Comparison

To hide an output expression in the Spec Comparison form, do the following.

➤ Right-click the row for the output expression and choose *Hide Selected Rows*.

To hide the rows for multiple output expressions, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection), right-click and choose *Hide Selected Rows*.

To display all the output expressions in the Spec Comparison form, do the following:

➤ Right-click any row and choose *Show All Hidden Rows*.

Updating the Spec Comparison with the Latest Results

To update the spec comparison with the latest results from the history items selected for comparison, do the following:

➤ Click the button.

Note: You cannot update the spec comparison with the latest results if the reference or other history item does not exist.

Sorting Data in the Spec Comparison Form

To sort data in the Spec Comparison form, do the following:

➤ Click the name of the column based on which you want to sort the data.

© 2023 Cadence Design Systems, Inc.

Saving a Spec Comparison

To save a spec comparison, do the following:

1. In the *Name* field, type a name for the spec comparison.
2. Click the button.

The spec comparison is saved to the `documents` directory of the ADE Assembler view and displayed in the *Documents* tree on the Data View pane. For more information about working with documents, see Chapter 23, “Working with Documents and Datasheets.”

Note: The spec comparison file is saved with the `.speccomparison` extension.

Opening a Spec Comparison

To open an existing spec comparison, do one of the following:

- In the *Name* cyclic field on the Spec Comparison form, select the spec comparison.
- In the *Documents* tree on the Data View pane, double-click the spec comparison.

The spec comparison is displayed in the Spec Comparison form.

Deleting a Spec Comparison

To delete an existing spec comparison, do the following:

➡ In the *Documents* tree on the Data View pane, right-click the spec comparison and choose *Delete*.

The spec comparison will not be displayed in the *Name* drop-down list in the Spec Comparison form.

Exporting a Spec Comparison to a HTML or CSV File

You can export the spec comparison data displayed in the Spec Comparison form to a HTML or comma-separated values (CSV) file.

© 2023 Cadence Design Systems, Inc.

Tip

You can hide the spec comparison data you do not want to export to the HTML or CSV file. For more information on hiding spec comparison data, see the following topics:

- Hiding and Showing the Comparison Data for Tests
- Hiding and Showing an Output Expression in the Spec Comparison

To export a spec comparison to a HTML or CSV file, do the following:

1. Click the button.

   The Export Results form appears.

2. In the *File name* field, type a filename with the `.html`, `.htm` or `.csv` extension.

3. Click *Save*.

   The program exports the results to the file you specified. By default, the file is saved in the `documents` directory of the ADE Assembler view and displayed in the *Documents* tree on the Data View pane. For more information about working with documents, see Chapter 23, “Working with Documents and Datasheets.”

If you click the button in the detail view, the resulting HTML or CSV file contains the contents of the detail view and summary view.

© 2023 Cadence Design Systems, Inc.

Managing Results

The following topics in this section provide details on how to manage simulation results displayed in the Results tab:

- Annotating Simulation Results
- Saving Results
- Exporting Results

Annotating Simulation Results

Video

You can view a video demonstration for this feature at Results Annotation Flow in ADE Assembler.

You can annotate the schematic to show parameters, operating points, net names, and voltages of individual design components from the results displayed on the Results tab of the Outputs pane.

Results can be displayed by performing any one of the following actions:

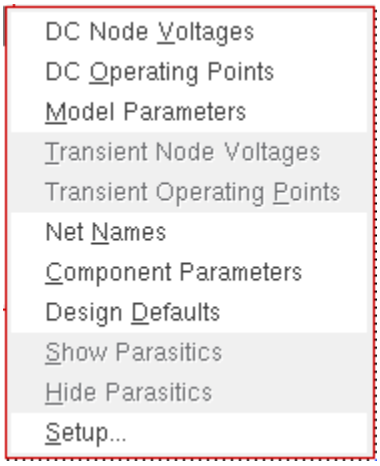
- Run a simulation.
- View results.

To annotate results to the schematic, do the following:

1. On the Results tab of the Outputs pane, right-click a data point and choose *Annotate*.

© 2023 Cadence Design Systems, Inc.

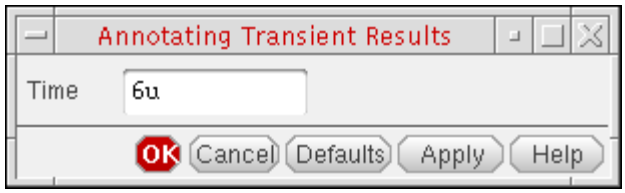
A submenu of items that you can annotate appears. Only those items for which you have annotation data are active on the submenu.



2. Select the required item from the Annotate submenu. The following table describes each submenu item:

Annotate Submenu Item	Description
DC Node Voltages	The program annotates your design schematic with DC node voltage values.
DC Operating Points	The program annotates your design schematic with DC operating point data.
Model Parameters	The program annotates your design schematic with enabled model parameter values.

© 2023 Cadence Design Systems, Inc.

Annotate Submenu Item	Description
Transient Node Voltages	The Annotating Transient Results form appears.  **1.** In the *Time* field, type the transient time point. **2.** Click *OK*. The program annotates your design schematic with transient node voltage values from the specified simulation time point.
Transient Operating Points	The program annotates your design schematic with operating point data for the given time point.
Net Names	The program annotates your design schematic with net names.
Component Parameters	The program annotates your design schematic with component parameters.
Design Defaults	The program restores the annotations on your design schematic from the CDF.
Show Parasitics *Hide Parasitics*	The program shows or hides parasitics when you descend (*Edit – Hierarchy – Descend Edit*) into the schematic view of a cell you simulated using its extracted view—`(out of context)` appears in the title bar **Note:** These submenu items are available only when you have DC operating point results. These submenu items also appear on the *Parasitics* menu on the main menu bar.
Setup	The Annotation Setup form appears. You can configure the annotation settings in the design. For more information, see Using Annotation Setup Form section in the *Virtuoso Schematic Editor L User Guide.*

© 2023 Cadence Design Systems, Inc.

To save the annotation settings, choose *File – Save* from the Annotation Setup form. For more information on saving the annotation settings, see Saving Annotation Settings section in the *Virtuoso Schematic Editor L User Guide*.

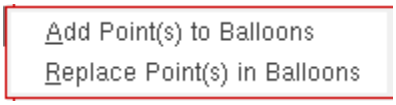
Using Annotation Balloons for Annotation

After running a simulation or viewing the results, you can annotate the simulation results from different data points using the annotation balloons on the schematic canvas. You can set multiple data points for annotations from different data points from a history and from different histories. The annotated data is displayed in the tabular form on the annotation balloons. As a result, you can compare and debug data from multiple data points.

To annotate data points on the annotation balloons, do the following:

1. On the Results tab of the Outputs pane, right-click a data point and choose *Annotation Balloons*.

   A submenu of options appears.



2. Select an available option from the *Annotation Balloons* submenu:

Annotate Submenu Item	Description
Add Point(s) to Balloons	Adds the values of the selected data point to the existing data on the annotation balloon and displays the data in a tabular form.
Replace Point(s) in Balloons	Replaces the value of the last selected data point with the existing data on the annotation balloon.

Annotation balloons are displayed when you move the cursor over any instance on the schematic. When you add multiple data points for annotation, the values of the last selected data point are annotated on the schematic as well as displayed on annotation balloons. As a result, the last selected data point result is highlighted in different color on annotation balloon to differentiate from the other data point results. In the annotation balloon, the color of the data point annotated on canvas appears yellow when the transparency is set from 21 to 65; and orange when the transparency is set from 0 to 20, or from 66 to 100. You can change the color settings of the annotation balloon using the Annotation Balloon Options.

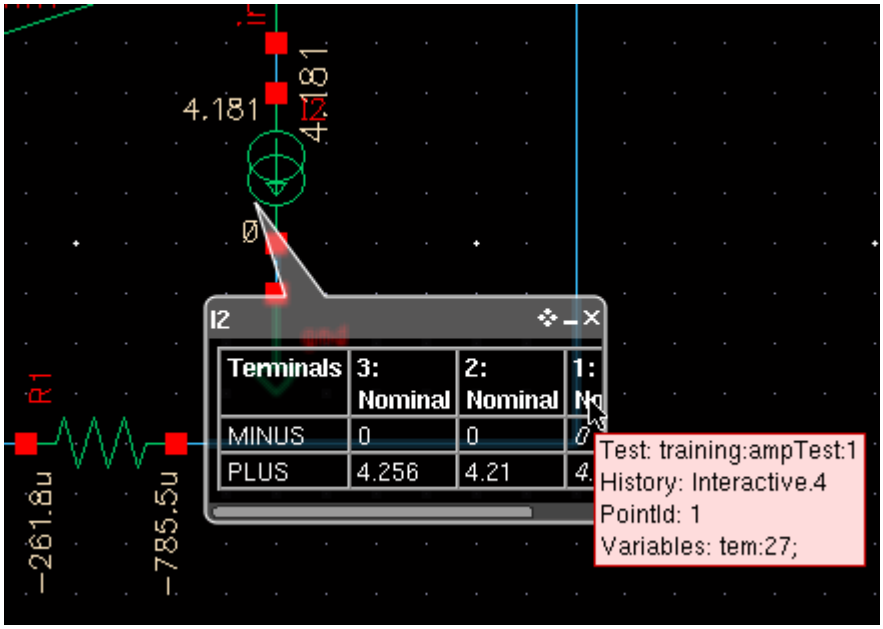
© 2023 Cadence Design Systems, Inc.

The annotation balloon shows the following information in the tooltip of the column header, when displaying information for multiple data points:

- Test name
- History name
- Data point ID
- Sweep parameters

The header of the annotation balloon shows the following information:

- Design point ID
- Corner name



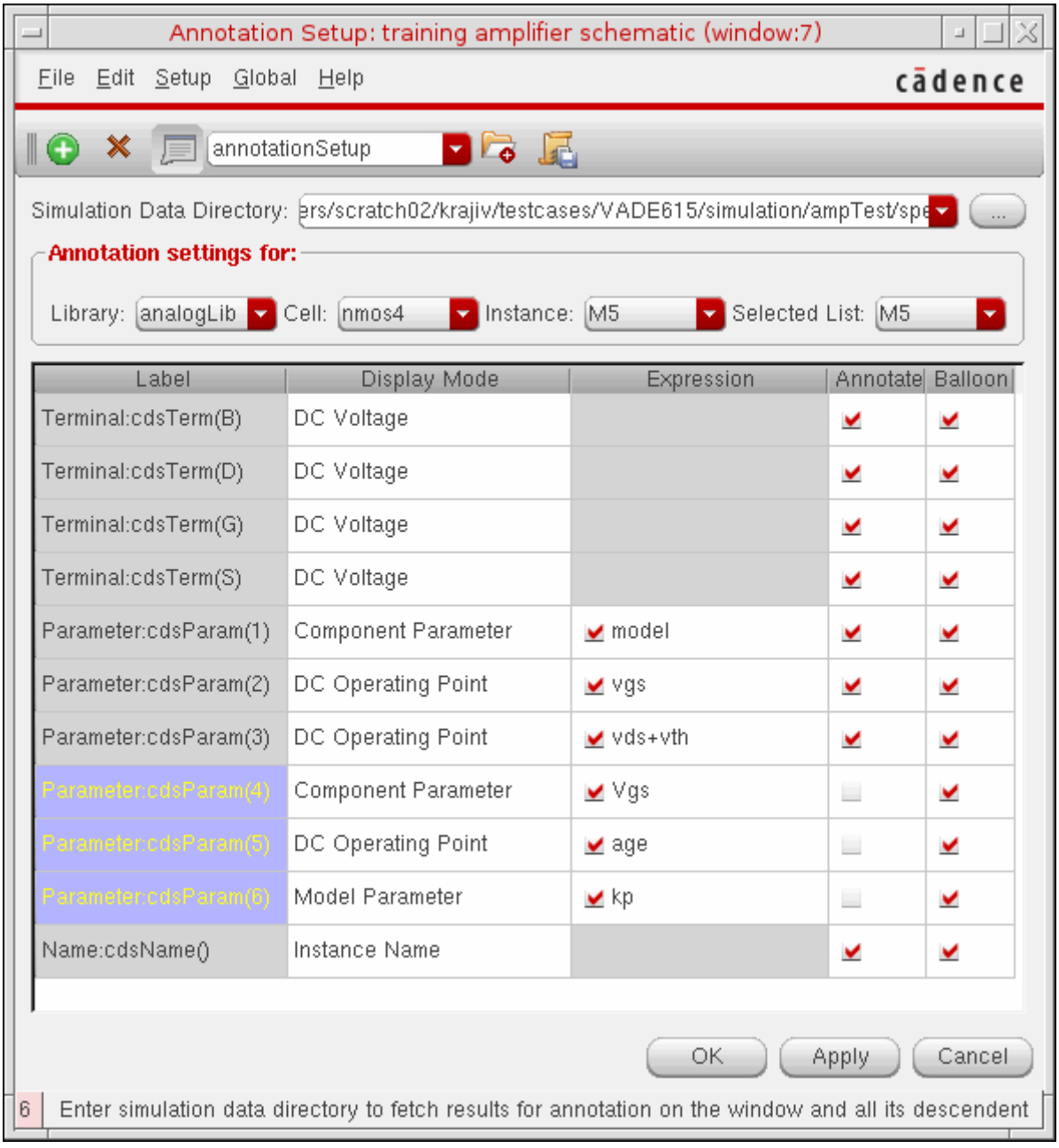
Specifying the Data Directory for Labels

To specify the simulation data directory (run directory) for labels, do the following:

1. In the schematic window, choose *View – Annotations – Setup.*

© 2023 Cadence Design Systems, Inc.

The Annotation Setup form appears.



2. In the *Simulation Data Directory* field, type the path to the simulation run directory.
3. Click *OK*.

Note: You do not need to use this form if

© 2023 Cadence Design Systems, Inc.

- The analog circuit design environment is active and you have specified the correct directory as the run directory
- You used the Results Browser to select results for the current schematic
- The most recent simulation you ran was for this schematic

Saving Results

To save results for specific tests, do the following:

1. On the Results tab of the Outputs pane, right-click a test name and choose *Save*.

   The Save Results form appears.
2. (Optional) By default, the results are saved to a directory named `schematic-save`. To use a different directory name, type the directory name in the *Save As* field.
3. (Optional) In the *Comment* field, type a comment so that you can differentiate simulation results.
4. Enter the path to the directory where you want to save the results.

   Alternatively click the browse button to specify the directory.
5. Click *OK*.

To save results for all tests, do the following:

1. Click the button on the Results tab of the Outputs pane.

   The Save Results form appears.
2. Enter the path to the directory where you want to save the results.

   Alternatively click the browse button to specify the directory.
3. (Optional) Select the *Copy PSF Results?* check box if you want to copy PSF results.
4. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Exporting Results

You can export results in CSV, HTML, or Excel formats where you can manipulate data and generate the required reports.

To export results to an external file:

1. Click *Export data in CSV, HTML, or Excel format* on the *Results* tab of the Outputs pane.

   The Export Results form is displayed.

2. Specify a file name in the *File Name* field.
3. From the *Files of type* drop-down list, choose the format in which you want to save results. The supported file formats are `html`, `.htm`, `.csv`, `.pivot.csv` , `.xls`, or `.pivot.xls`.
4. Click *Save*.

   ADE Assembler exports the results to the file you specified.

Important Points to Note

- By default, the exported results file is saved in the `documents` directory of the ADE Assembler view and displayed in the *Documents* tree in the Data View assistant. For more information about working with documents, see Working with Documents and Datasheets.
- If the results contain details of instance parameters, the *Detail - Transpose* or *Detail - Transpose with Test* result view shows the instance path in the tooltips of parameter names. The exported data also contains the complete hierarchical path of parameters. This helps in uniquely identifying the parameters when multiple instances have parameters with same names.

Video

You can view a video demonstration for this feature at Exporting Simulation Results in a Pivot Ready Format

Exporting Selected Results Data

To export a selected subset of results from the results table, use the `Ctrl+C` keyboard shortcut to copy and then the `Ctrl+V` shortcut to paste them in any other application, such as Excel.

© 2023 Cadence Design Systems, Inc.

Exporting Results in Excel Format for Pivot Tables

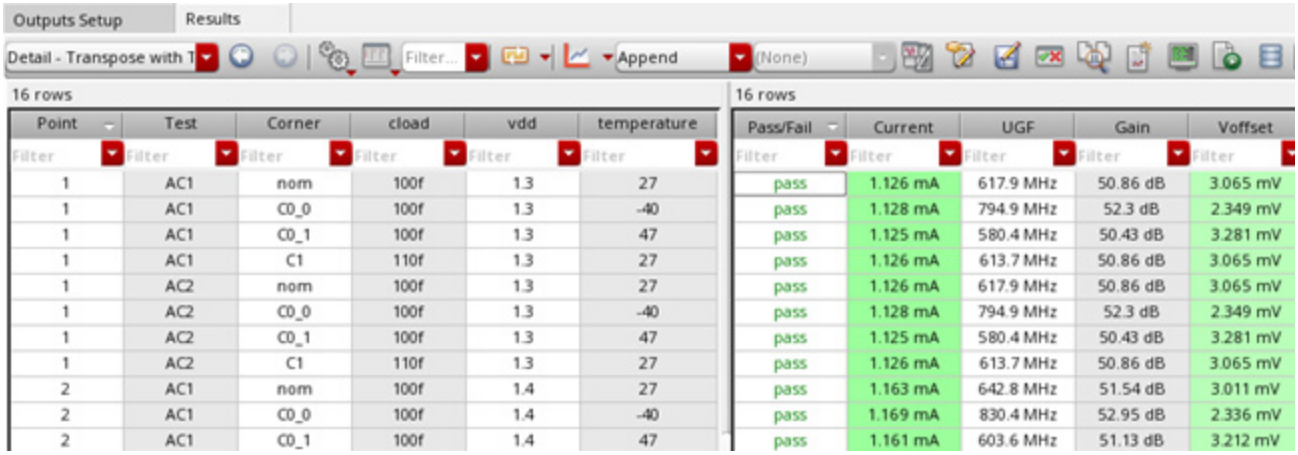
Pivot tables and graphs are very useful in reviewing and analysing multi-dimensional data, and therefore, are used for post-processing of simulation results. You can use the data exported in both `.xlsx` and `.pivot.xlsx` files to create pivot reports. However, the `Ready for Pivot Excel Files (pivot.xlsx)` file type is the best format to be used for pivot charts.

The main difference between the two formats is that when you export data in the `.xlsx` files, each expression from ADE Assembler is assigned a separate column, whereas in the `.pivot.xlsx` files, all output expressions from the results are added to a single column named `Expression`, which can be used to create a data slicer.

The following example shows the difference in the two files exported from the common results.

© 2023 Cadence Design Systems, Inc.

Results in ADE Assembler



Outputs Setup | Results

Detail - Transpose with T | Filter... | Append | (None)

16 rows

Point	Test	Corner	cload	vdd	temperature
1	AC1	nom	100f	1.3	27
1	AC1	C0_0	100f	1.3	-40
1	AC1	C0_1	100f	1.3	47
1	AC1	C1	110f	1.3	27
1	AC2	nom	100f	1.3	27
1	AC2	C0_0	100f	1.3	-40
1	AC2	C0_1	100f	1.3	47
1	AC2	C1	110f	1.3	27
2	AC1	nom	100f	1.4	27
2	AC1	C0_0	100f	1.4	-40
2	AC1	C0_1	100f	1.4	47

16 rows

Pass/Fail	Current	UGF	Gain	Voffset
pass	1.126 mA	617.9 MHz	50.86 dB	3.065 mV
pass	1.128 mA	794.9 MHz	52.3 dB	2.349 mV
pass	1.125 mA	580.4 MHz	50.43 dB	3.281 mV
pass	1.126 mA	613.7 MHz	50.86 dB	3.065 mV
pass	1.126 mA	617.9 MHz	50.86 dB	3.065 mV
pass	1.128 mA	794.9 MHz	52.3 dB	2.349 mV
pass	1.125 mA	580.4 MHz	50.43 dB	3.281 mV
pass	1.126 mA	613.7 MHz	50.86 dB	3.065 mV
pass	1.163 mA	642.8 MHz	51.54 dB	3.011 mV
pass	1.169 mA	830.4 MHz	52.95 dB	2.336 mV
pass	1.161 mA	603.6 MHz	51.13 dB	3.212 mV

Results exported in Excel Format (.xlsx)

Point	Test	Corner	cload	vdd	temperatu	Pass/Fail	Current	UGF	Gain	Voffset
1	AC1	nom	100e-15	1.3	27	pass	1.126e-3	617.9e6	50.86	3.065e-3
1	AC1	C0_0	100e-15	1.3	-40	pass	1.128e-3	794.9e6	52.3	2.349e-3
1	AC1	C0_1	100e-15	1.3	47	pass	1.125e-3	580.4e6	50.43	3.281e-3
1	AC1	C1	110e-15	1.3	27	pass	1.126e-3	613.7e6	50.86	3.065e-3
2	AC1	nom	100e-15	1.4	27	pass	1.163e-3	642.8e6	51.54	3.011e-3
2	AC1	C0_0	100e-15	1.4	-40	pass	1.169e-3	830.4e6	52.95	2.336e-3
2	AC1	C0_1	100e-15	1.4	47	pass	1.161e-3	603.6e6	51.13	3.212e-3

Each expression is assigned a separate column

Results exported in Ready for Pivot Excel Format (.pivot.xlsx)

	A	B	C	D	E	F	G	H
	Expression	Point	Test	Corner	cload	vdd	temperature	Value
	Current	2	AC2	nom	100e-15	1.4	27	1.16E-03
	Current	2	AC2	C0_0	100e-15	1.4	-40	1.17E-03
	Current	2	AC2	C0_1	100e-15	1.4	47	1.16E-03
	Current	2	AC2	C1	110e-15	1.4	27	1.16E-03
	UGF	1	AC1	nom	100e-15	1.3	27	6.18E+08
	UGF	1	AC1	C0_0	100e-15	1.3	-40	7.95E+08
	UGF	1	AC1	C0_1	100e-15	1.3	47	5.80E+08

All output expressions are exported to a single column, named **Expression**, and values are added in the **Value** column.

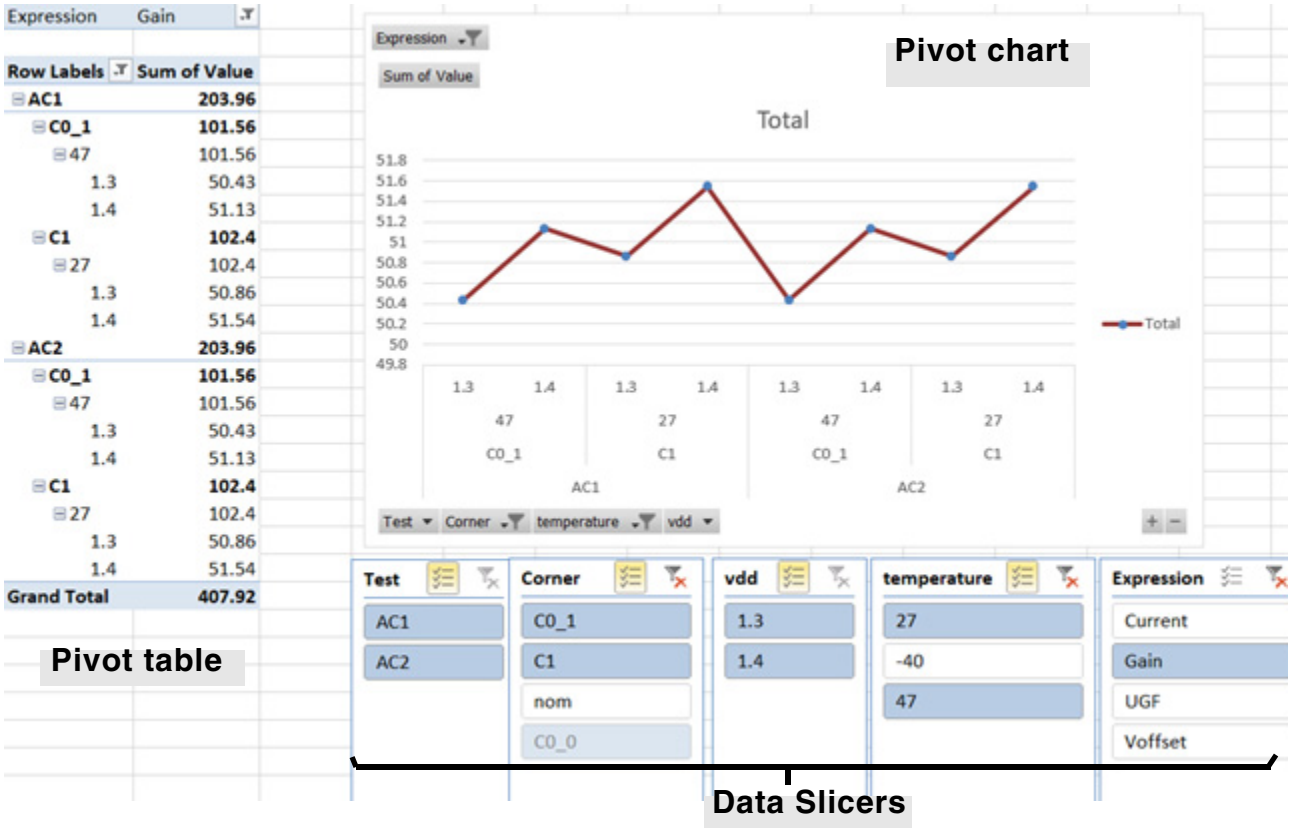
© 2023 Cadence Design Systems, Inc.

To export result data in the ready for pivot format, do the following:

1. (Optional) Set the detailTransposeViewShowTest environment variable to `t` to view the *Detail - Transpose with Test* result type on the *Results* tab.
2. Choose the *Detail - Transpose* or *Detail - Transpose with Test* result view.

   Note: This is particularly helpful for a multi-test setup because it adds a column to save the test name, which gets exported to the Excel file, and can be used to filter data in pivot reports.
3. Set the `exportFormat` environment variable to `"numbersOnly"`.
4. Click on the *Results* tab of the *Outputs* pane.
5. In the Export Results form, specify a location and a file name with the `.pivot.xlsx` extension.
6. Choose the `Ready for Pivot Excel Files` format.

The results are exported to an Excel file where you can use the data to create pivot table and chart. An example of a pivot chart created using the .pivot.xlsx file is shown below.



© 2023 Cadence Design Systems, Inc.

Exporting Selected Data

You can choose the rows and columns to be exported. For this, you can do the following:

- Hide the data for which you do not want to export results. For more information, see Hiding and Showing Data on the Results Tab.
- Hide columns in the *Detail - Transpose* view. For more information, see Hiding and Showing Columns in the Detail - Transpose View.
- Hide the results for the tests for which you do not want to export results. For more information, see Hiding and Showing Results for Tests.
- By default, the *Results* tab displays the best 10 design points from the simulation run. To display more design points in the Results tab, use the numberOfBestPointsToView environment variable. Only the data displayed in the Results tab is exported to the HTML or CSV file.
- By default, results are exported to CSV files in the scientific notation format. Set the exportPreserveScalingFactors environment variable to export results in the same format as they are displayed in the Results tab to the CSV file.

© 2023 Cadence Design Systems, Inc.

Working in the Results Display Window

The Results Display Window appears when you right-click a test name on the Results tab of the Outputs pane and choose an item from the *Print* submenu.

Using the menus in the Results Display Window, you can perform the following tasks:

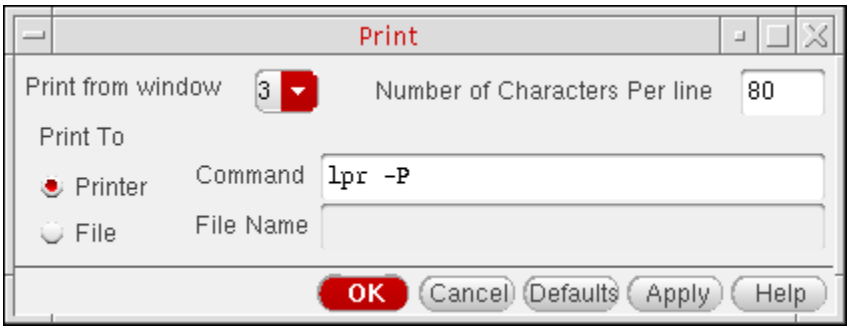
- Printing Results from the Results Display Window
- Saving Results Display Window Setup Information
- Loading Results Display Window Setup Information
- Updating Results in the Results Display Window
- Clearing the Results Display Window
- Making a Window Active
- Closing a Results Display Window
- Editing Expressions
- Setting Results Display Options
- Displaying Untruncated Output Information

Printing Results from the Results Display Window

To print results from the Results Display Window, do the following:

1. Choose *Window – Print*.

   The Print form appears.



2. In the *Print from window* drop-down list, verify that the number from the lower left corner of the Results Display Window appears.

© 2023 Cadence Design Systems, Inc.

3. In the *Number of Characters Per Line* field, type an integer number of characters you want printed on each line.
4. Select one of the following *Print To* options:
   - *Printer* – In the *Command* field, type a valid printer command.
   - *File* – In the *File Name* field, type a filename.
5. Click *OK*.

   The program prints the contents of your Results Display Window to a file or printer.

Saving Results Display Window Setup Information

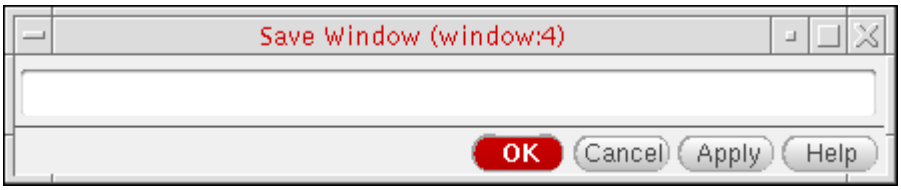
You can save your current Results Display Window setup (such as printing format, setting a printing range if the amount of data is too large, printing at a certain interval, sorting) to a file so that you can load it again after running another simulation. The saved setup information determines how the program displays your results.

Note: You can only save setup information for expressions that can evaluate to waveforms: If you print a single number, like a node voltage, the *Save State* command is not active.

To save the contents and format of a Results Display Window, do the following:

1. Choose *Window – Save State*.

   The Save Window form appears.



2. In the empty field, type a filename.
3. Click *OK*.

   The program saves the setup of your current Results Display Window.

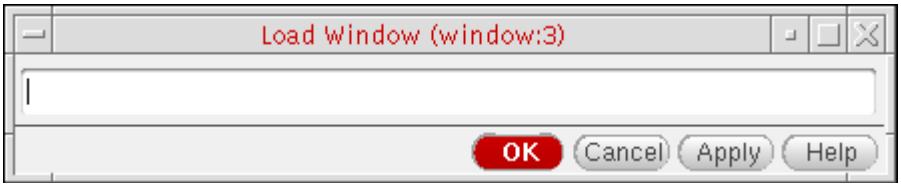
© 2023 Cadence Design Systems, Inc.

Loading Results Display Window Setup Information

To load a Results Display Window state that you previously saved, do the following:

1. Choose *Window – Load State*.

   The Load Window form appears.



2. In the empty field, type the name of the saved window state file.
3. Click *OK*.

   The program loads the specified window state file.

Updating Results in the Results Display Window

To update the Results Display Window with results from a new simulation, do the following:

➤ Choose *Window – Update Results*.

The program uses the current window setup when updating the results data. The program updates only results data that can evaluate to a waveform: This command is not active for updating single-number data (such as a node voltage).

Clearing the Results Display Window

To clear the results from the Results Display Window, do the following:

➤ Choose *Window – Clear*.

The program clears the Results Display Window.

Making a Window Active

There is no limit to the number of Results Display Windows you can have open, but only one is active at a time. The program writes printed results to the active window.

To make a window active, do the following:

© 2023 Cadence Design Systems, Inc.

➤ Choose *Window – Make Active*.

The word *Active* appears beneath the menu bar in the Results Display Window.

Closing a Results Display Window

To close a Results Display Window, do the following:

➤ Choose *Window – Close*.

The Results Display Window closes.

Editing Expressions

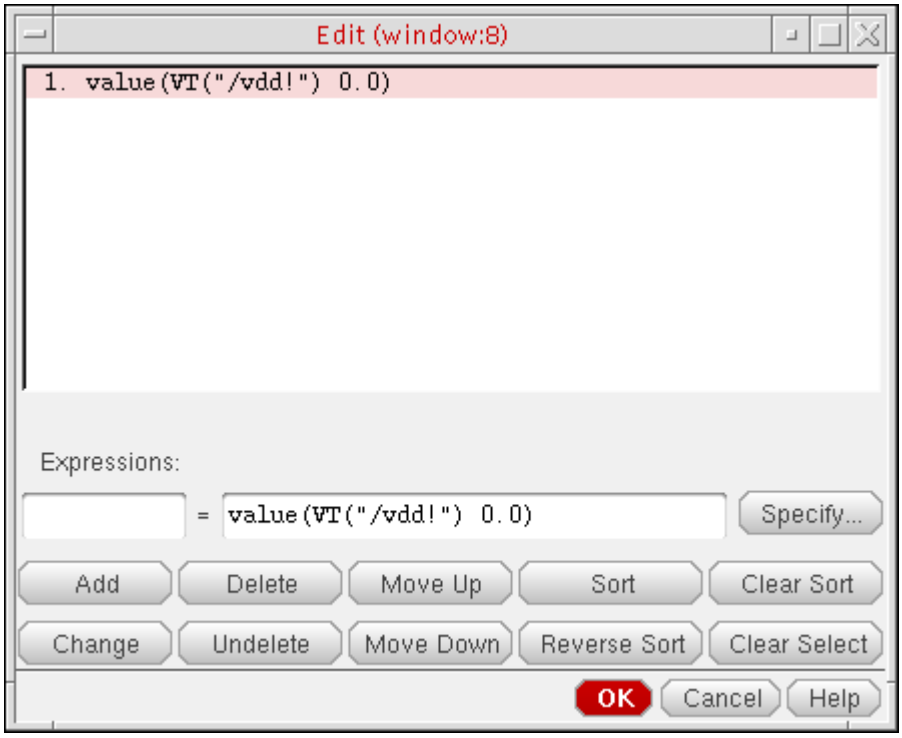
You can edit any expressions that evaluate to waveforms (such as DC operating point, transient operating point, and model parameters). If you print only one value, the *Edit* menu choices are not active. The editing commands operate only on the last table in the active Results Display Window.

To edit expressions in the currently active Results Display Window, do the following:

1. Choose *Expressions – Edit*.

© 2023 Cadence Design Systems, Inc.

The Edit form appears.



2. Using the buttons and fields on the form, edit the expressions you want to edit. When more than one expression appears in the window, you must select the expression you intend to edit.

Expressions	If you specified an optional aliased name for the expression, that name appears in the edit field to the left of the equal sign. Otherwise, that field is blank. The expression appears in the edit field to the right of the equal sign.
Specify	Retrieves the expression from the calculator buffer into the edit field. **Note:** If you click *Specify* when there is no expression in the buffer, the Calculator window appears so that you can create an expression.
Add	Adds the expression from the edit field to the list box and as a column in your Results Display Window. If you specified an optional name, this name appears in the list box and as the column heading.

© 2023 Cadence Design Systems, Inc.

Change	Replaces the selected expression in the list box with the one in the edit field and its values in the Results Display Window.
Delete	Removes the selected expression from the list box and its column from the Results Display Window.
Undelete	Undoes the last *Delete* action.
Move Up	Moves the selected expression up one position in the list box and its column over one to the left in the Results Display Window. If the selected expression is already at the top of the list, it moves (wraps around) to the bottom and its column moves (wraps around) to the rightmost side.
Move Down	Moves the selected expression down one position in the list box and its column over one to the right in the Results Display Window. If the selected expression is already at the bottom of the list, it moves (wraps around) to the top and its column moves (wraps around) to the leftmost side.
Sort	Sorts the values of the selected expression (which appear in the Results Display Window) such that they increase down the column (smallest value on top).
Reverse Sort	Sorts the values of the selected expression (which appear in the Results Display Window) such that they decrease down the column (largest value on top).
Clear Sort	Reverts to the default sort order (alphabetical by parameter name).
Clear Selection	Clears the selection in the list box. **Note:** You can also clear a selection by clicking it while holding down the *Control* key.

3. Click *OK* when you are finished editing.

Note: See also "Saving Results Display Window Setup Information" on page 1230.

© 2023 Cadence Design Systems, Inc.

Setting Results Display Options

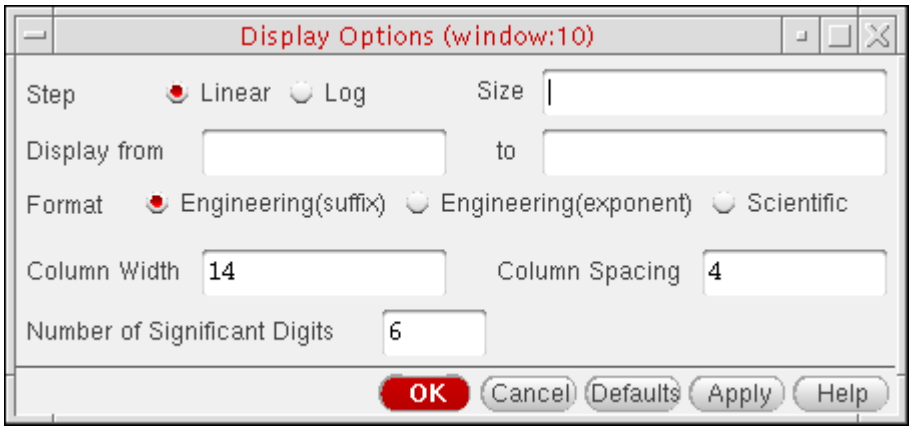
Tip

The Results Display Window may contain more than one type of results. Display options settings apply only to the last result (if the last result can evaluate to a waveform). After you edit the data, only the last results appear in the window. You can preserve the previous results by opening a new Results Display Window and printing the results you want to edit in the new window.

To change the results display options in the currently active Results Display Window, do the following:

1. Choose *Expressions – Display Options*.

   The Display Options form appears.



2. Use the fields and buttons on the form to specify one or more of the following display options:

Step	Specifies one of the following scales for the step size: ■ *Linear* ■ *Log*
Size	Specifies the interval for printing data using the *Step* scale specified above.

© 2023 Cadence Design Systems, Inc.

Display from, to	Specifies the range of data to print. If you leave the *Display from* field blank, the program uses the beginning of the data as the *Display from* value. If you leave the *to* field blank, the program uses the end of the data as the *to* value. **Note:** You can set the print range only after printing data.
Format	Specifies one of the following formats for the printed data: ■ *Engineering Suffix* (default) If you select *Engineering Suffix*, the program represents 0.0001 as 0.1m. ■ *Engineering* If you select *Engineering*, the program represents 0.0001 as 0.1e-3. ■ *Scientific* If you select *Scientific*, the program represents 0.0001 as 1e-4.
Column Width	Specifies the number of characters allowed for column width. You can specify a number from 4 to 20. The default width is 14 characters.
Column Spacing	Specifies the number of blank spaces the program uses to separate columns. You can specify a number from 1 to 10. The default spacing is 4.
Number of Significant Digits	Specifies the number of significant digits the program uses when printing results data. You can specify a number from 2 to 10. The default is 4 digits.

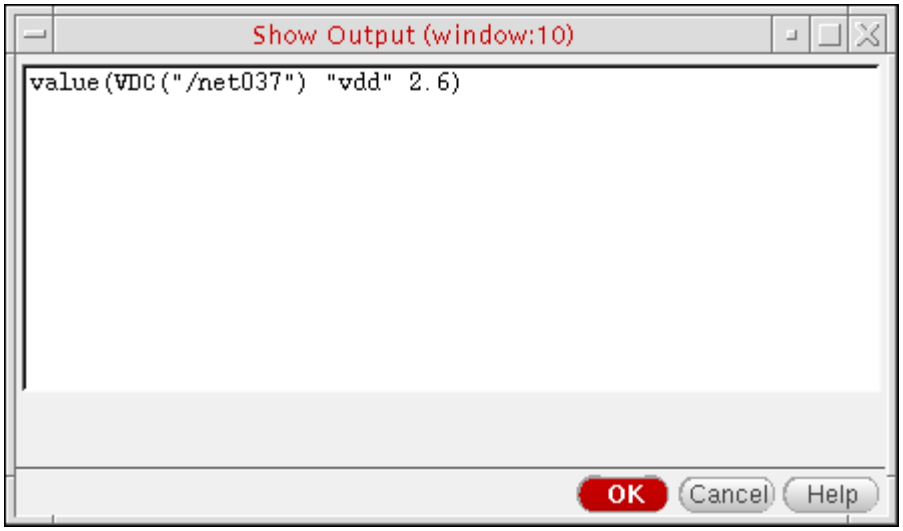
Displaying Untruncated Output Information

When output names are too long to fit into columns in the Results Display Window, the program truncates them. To see untruncated output names, do the following:

➤ In the Results Display Window, choose *Info – Show Output*.

© 2023 Cadence Design Systems, Inc.

The Show Output window appears.



If you have more than one output in your Results Display Window, only the last one appears in the Show Output window.

© 2023 Cadence Design Systems, Inc.

Using SKILL to Display Tabular Data

You can use the SKILL language for queries to request other kinds of simulation results, to build output format macros, and to automate test and result reporting sequences. The syntax for queries is shown at the beginning of the line in the Results Display window.

To display…	Type this command in the CIW
A list of operating-point parameter names and their values for `R1`	`OP("/R1","??")`
A list of just the operating-point parameter names for `R1`	`OP("/R1","?")`
A single operating-point parameter (`v` for voltage, for example) and its value for `R1`	`OP("/R1","v")`
A list of transient operating-point parameter names and their values for `C1`	`OPT("/C1","??")`
A list of just the transient operating-point parameter names for `C1`	`OPT("/C1","?")`
A single transient operating-point parameter (`i` for current, for example) and its value for `C1`	`OPT("/C1","i")`
A list of model parameter names and their values for `Q1`	`MP("/Q1","??")`
A list of just the model parameter names for `Q1`	`MP("/Q1","?")`
A single model parameter (`is` for saturation current, for example) and its value for `Q1`	`MP("/Q1","is")`
Noise parameter information for a device with only one noise parameter (a resistor `R4`, for example)	`VNP("/R4")`
A list of noise parameter names for a device with more than one noise parameter (a device `D24`, for example) and their values	`VNPP("/D24","??")`
A list of just the noise parameter names for a device with more than one noise parameter (a device `D24`, for example)	`VNPP("/D24","?")`
A single noise parameter (`rs` for saturation resistance, for example) and its value for a device with more than one noise parameter (a device `D24`, for example)	`VNPP("/D24","rs")`

© 2023 Cadence Design Systems, Inc.

Showing Variable and Parameter Values for Design Points on the Results Tab

By default, the *Results* tab displays the names and values of device parameters and swept test variables that were used for each design point on the gray row above the results for that point.

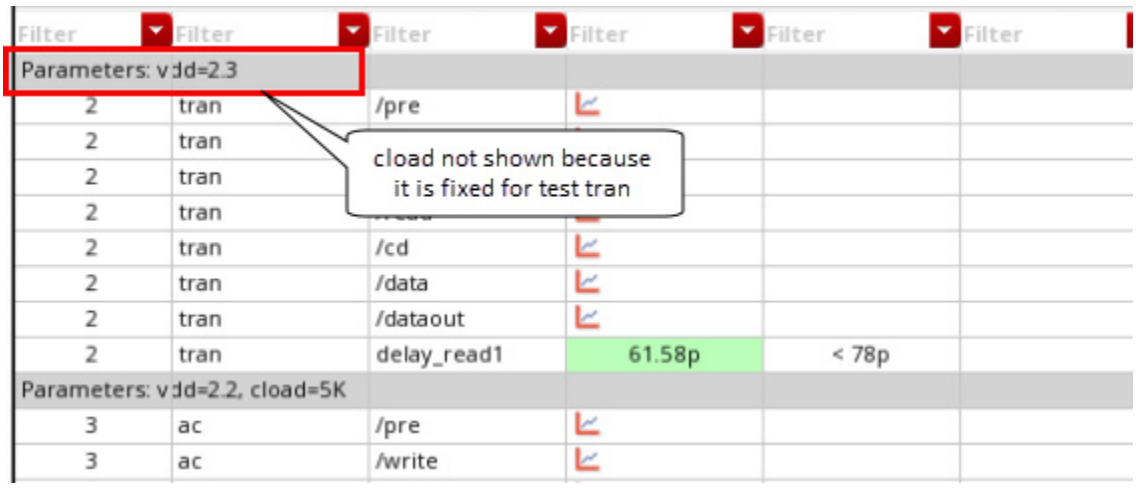
However, you can show the values of the fixed or constant device parameters and test variables too, as described in the following cases:

- Showing Test Variables that are Fixed for Selected Tests
- Showing Test Variables that are Fixed for All Tests

Showing Test Variables that are Fixed for Selected Tests

If the value of a test variable is fixed for some tests and variable for other tests, by default, the values of variables are visible in the gray rows for only those design points in which the test variable is swept. For other design points, the values of fixed test variables is not displayed.

For example, a variable `cload` is swept in test `ac` and is fixed in test `tran`. In this case, the *Results* tab appears as show shown below.



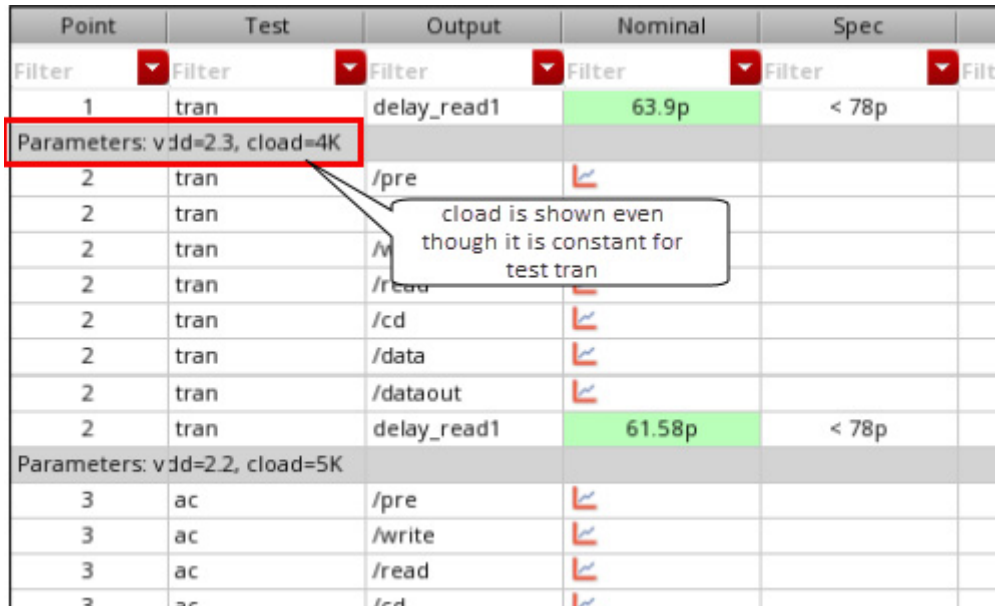
To show the value of a variable in the gray bar for all design points even if it is fixed in some design points and variable for other design points:

➡ Set the displayFixedTestVariables environment variable to `t`

After this variable is set to `t`, the fixed test variables are shown for all design points when the next time the *Results* tab is refreshed.

© 2023 Cadence Design Systems, Inc.

For the example described above, the results are displayed as shown in the example below.



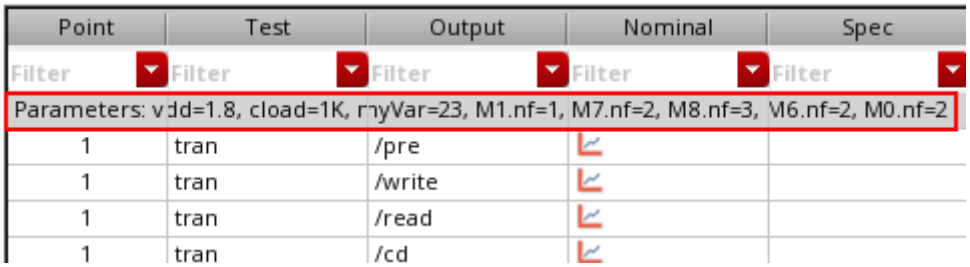
The displayFixedTestVariables environment variable is effective only when the *Fixed Variables/Parameters* command in the *Configure what is shown in the table* drop-down list on the toolbar of the *Results* tab is not selected.

Showing Test Variables that are Fixed for All Tests

To view the values of all test variables and device parameters that are fixed for all tests:

➡ Select the *Fixed Variables/Parameters* command in the *Configure what is shown in the table* drop-down list on the toolbar of the *Results* tab

All fixed test variables and parameters are displayed on the gray row above the results for each design point in the *Results* tab.



© 2023 Cadence Design Systems, Inc.

Showing Variable Parameters for Design Points in the Variable Display Assistant

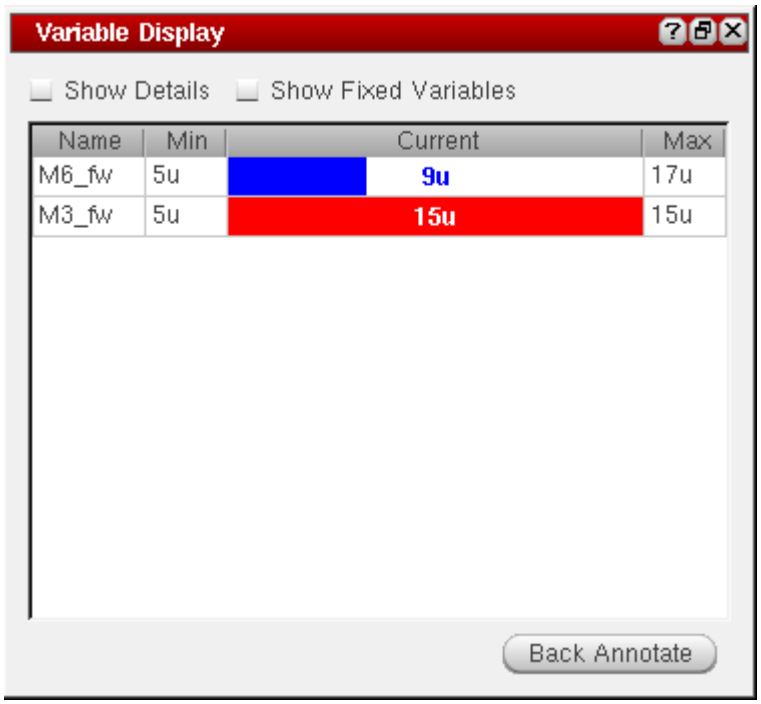
To view the list of all the variable parameters in the Variable Display assistant:

1. Choose *Window – Assistant – Variable Display* to open the Variable Display assistant.

   The assistant is displayed at the bottom of the ADE Assembler window.

2. Click a design point on the *Results* tab.

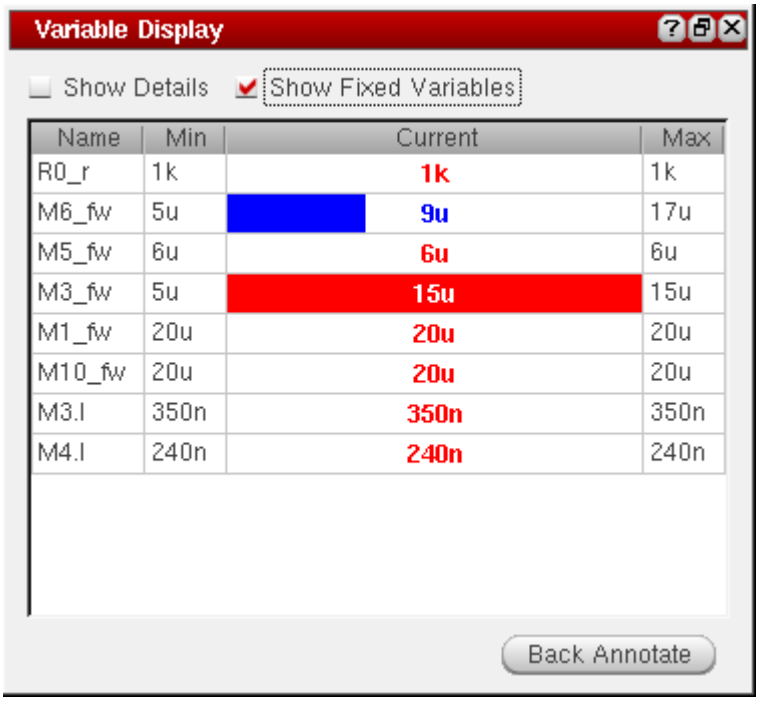
   A list of all the variables and their values used for a particular design point are displayed in the Variable Display assistant.



The Variable Display assistant shows the name of each variable with the minimum and maximum values and the current value used for the currently selected design point. The blue and red color bars indicate the sweep stage for a variable. For example, in the above figure, the blue bar for *M6_fw* shows that `9u` is one of the sweep values for this variable and there are few more sweeps. The red bar for *M3_fw* shows that `15u` is the maximum sweep value used for this variable.

By default, the fixed parameters are not shown in the Variable Display assistant. Click *Show Fixed Variables* on top to show all fixed and variable parameters, as shown below.

© 2023 Cadence Design Systems, Inc.



© 2023 Cadence Design Systems, Inc.

Searching for Conditional Results

Important

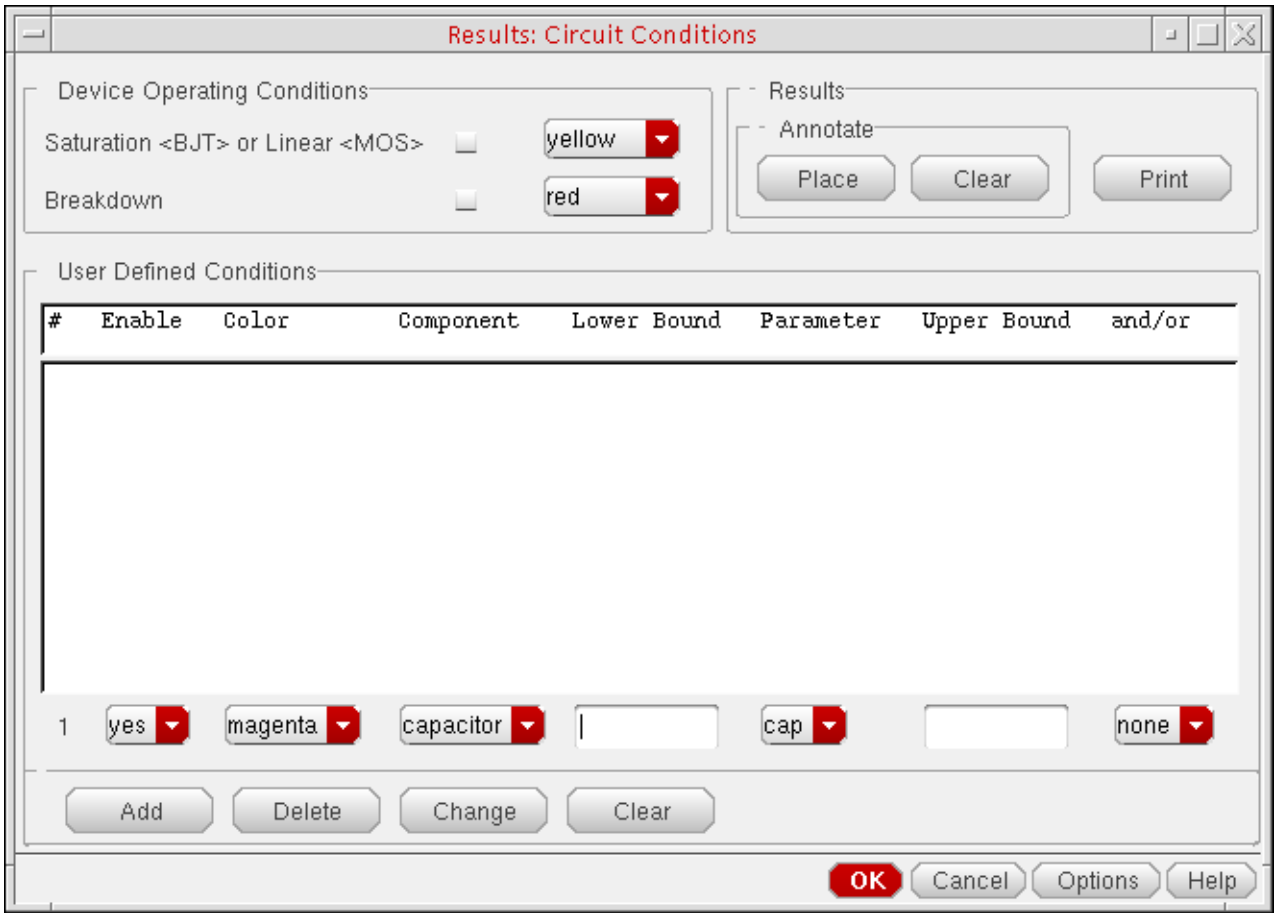
You must run a DC operating-point analysis to use the circuit conditions capability.

After running a simulation, you can search the results for components in the saturation region, breakdown region, or any user-defined region.

To do a conditional search for results, do the following:

1. On the Results tab of the Outputs pane, right-click a test name and choose *Circuit Conditions*.

   The Circuit Conditions form appears:



2. In the *Device Operating Conditions* group box, specify device operating conditions.

© 2023 Cadence Design Systems, Inc.

You can choose to view components in the saturation (for BJT devices), linear (for MOS devices), or breakdown region.

Note: The appropriate model parameters must be set for the simulator to calculate these conditions. These features might not be available for simulators other than Spectre.

3. In the *User Defined Conditions* group box, do the following:
   a. Use the cyclic and type-in fields to specify custom conditions.
   b. Click *Add*.
4. In the *Results* group box, do the following:
   - (Optional) Click *Place* to highlight the instances that meet the specified conditions on the schematic.
   - (Optional) Click *Print* to print the values of instances that meet the specified conditions in a print window.
5. Click *Add*.

See also

- Filtering Out Components by Model Name on page 1245
- Sorting Components by Parameter Value on page 1247

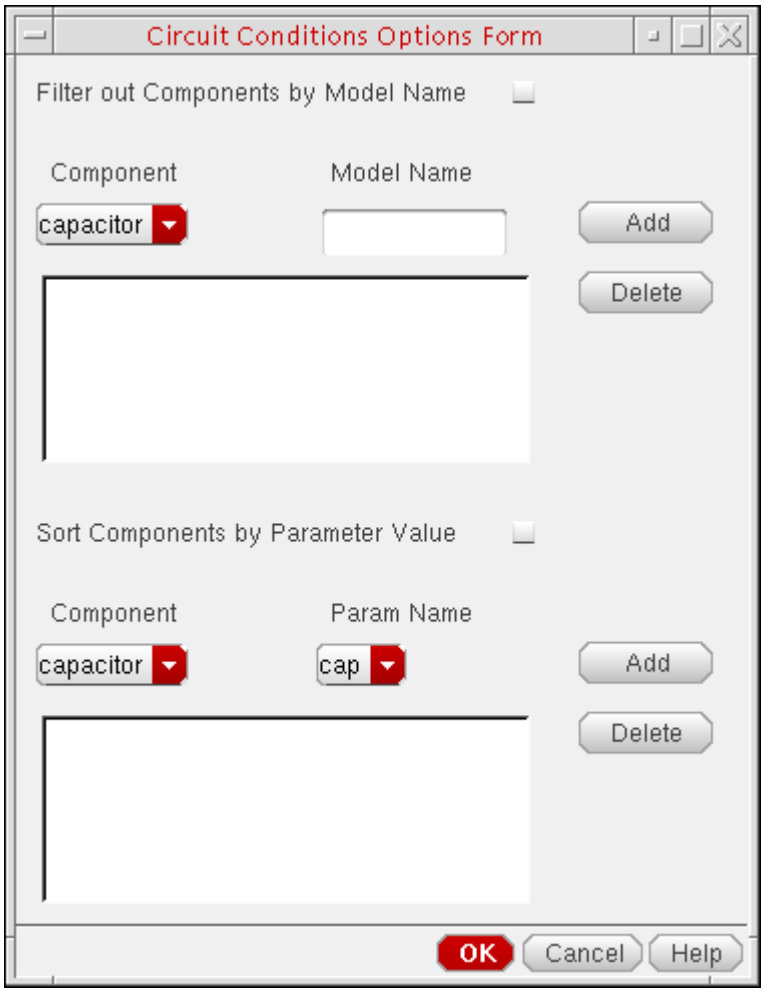
Note: Filters and sorting conditions are active only if you have selected either *and* or in the *Boolean* drop-down list at the bottom right corner of the *User Defined Conditions* group box on the Circuit Conditions form.

© 2023 Cadence Design Systems, Inc.

Filtering Out Components by Model Name

1. On the Circuit Conditions Circuit Conditions form, click *Options*.

   The Circuit Conditions Options Form appears.



2. Select the *Filter out Components by Model Name* check box.
3. In the *Component* drop-down list, select a component type.
4. In the *Model Name* field, type a model name.
5. Click *Add*.

   The specified filter appears in the list box.

Note: Filters are active only if you have selected either *and* or in the *Boolean* drop-down list at the bottom right corner of the *User Defined Conditions* group box on the Circuit

© 2023 Cadence Design Systems, Inc.

Conditions form. The program filters out any components that match the filters and these components do not appear in the output when you click *Print*.

To remove a filter, do the following:

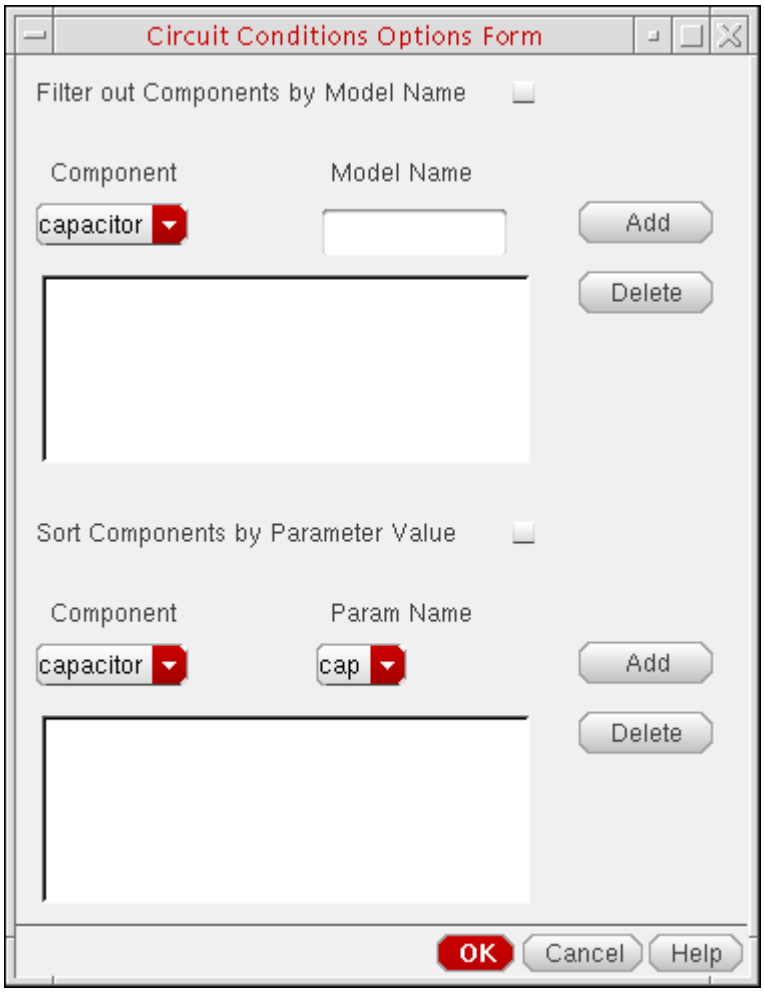
1. In the list box, select one or more filters that you want to delete.
2. Click *Delete*.

© 2023 Cadence Design Systems, Inc.

Sorting Components by Parameter Value

1. On the Circuit Conditions form, click *Options*.

   The Circuit Conditions Options Form appears.



2. Select the *Sort Components by Parameter Value* check box.
3. In the *Component* drop-down list, select a component type.
4. In the *Param Name* field, type a parameter name.
5. Click *Add*.

   The specified sort condition appears in the list box.

Note: Sort conditions are active only if you have selected either *and* or in the *Boolean* drop-down list at the bottom right corner of the *User Defined Conditions* group box on the

© 2023 Cadence Design Systems, Inc.

Circuit Conditions form. The program sorts the specified components by parameter name when you click *Print*.

© 2023 Cadence Design Systems, Inc.

20

Working with History Checkpoints

You can use history checkpoints in the ADE Assembler environment to save the active configuration of data such as corners analysis setup, global variables, sweep and test definitions. The environment creates a history checkpoint prior to running a simulation and stores checkpoints in your project directory. You can restore fragments of a checkpoint (such as just the sweep setup), edit the data, and run the simulation again. Checkpoints appear as part of history items on the History tab of the Data View assistant pane in your ADE Assembler environment.

See the following topics for more information:

- Specifying How Many Histories to Save on page 1250
- Overwriting a History Item during Subsequent Simulation Runs on page 1252
- Viewing Histories on page 1256
- Adding Notes to a History on page 1257
- Renaming Histories on page 1259
- Restoring a History Checkpoint on page 1260
- Viewing Results from a Particular History Checkpoint on page 1262
- Opening a Terminal Window in the Results Directory for a Particular History Item on page 1263
- Deleting a History Checkpoint on page 1264
- Locking and Unlocking a History Item on page 1265
- Importing Histories from Another Cellview on page 1266
- Merging Histories on page 1268

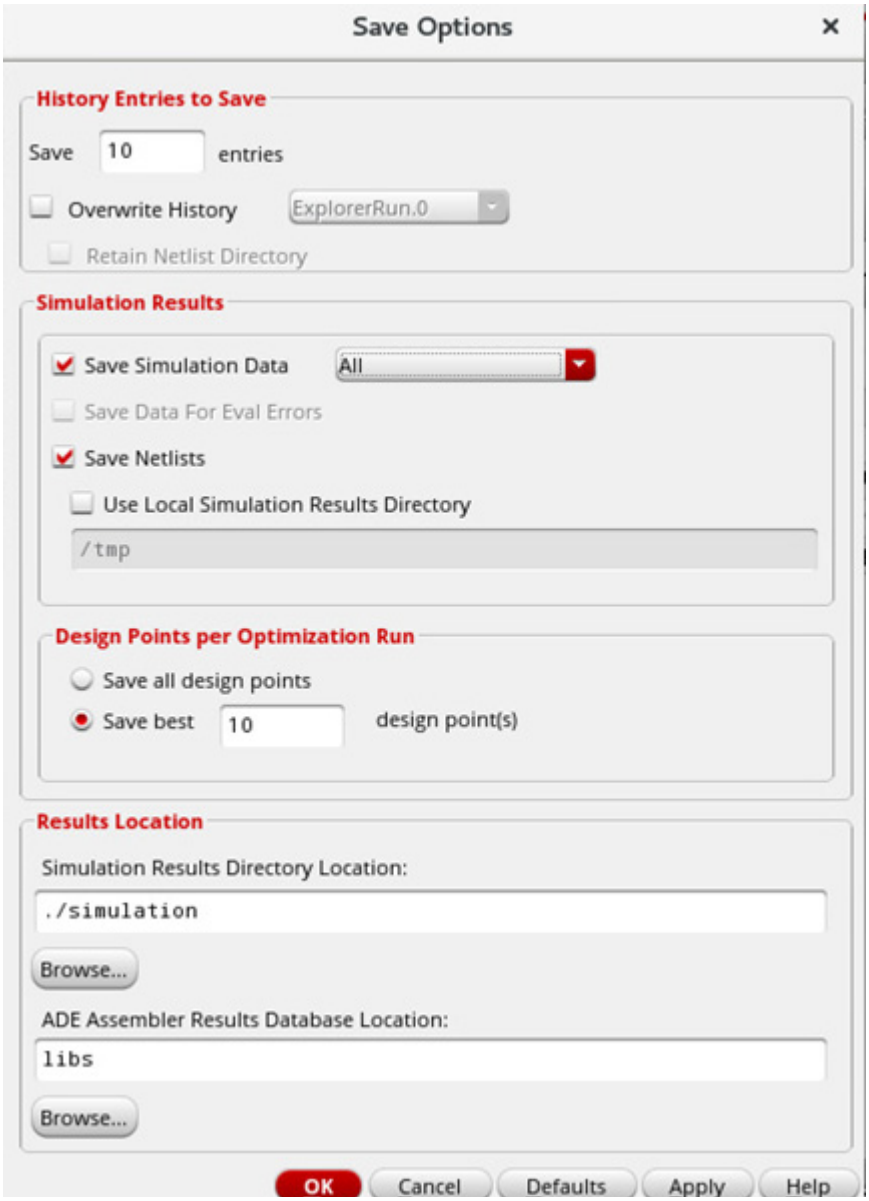
© 2023 Cadence Design Systems, Inc.

Specifying How Many Histories to Save

To specify the number of history entries for which you want to save simulation data, do the following:

1. Choose *Options – Save*.

   The Save Options form appears.



2. In the *History Entries to Save* group box, type the number of history items you want to save.

© 2023 Cadence Design Systems, Inc.

3. Click *OK*.

   The program saves only as many history items as the number you typed and automatically removes (does not retain) any unlocked history items above and beyond that number. The reference history item used for incremental simulation runs (see Running an Incremental Simulation to Reuse Results) is not automatically removed.

   For information about how history entries are saved when you open a maestro view in read only mode, see Number of History Entries Saved for Read Only maestro Views.

   See Locking and Unlocking a History Item for information about how to lock or unlock history items.

Note: For information on the *Design Points per Optimization Run* group box settings, see "Specifying How Much Optimization Data to Save".

© 2023 Cadence Design Systems, Inc.

Overwriting a History Item during Subsequent Simulation Runs

By default, a new history item is created for each simulation run. If you want to do any of the following, you can specify that the selected history item be overwritten during subsequent simulation runs:

- Save disk space by using the same history item for subsequent simulation runs.
- Reuse the netlist from a history item.
- Reuse the OCEAN and other scripts that you use with your design

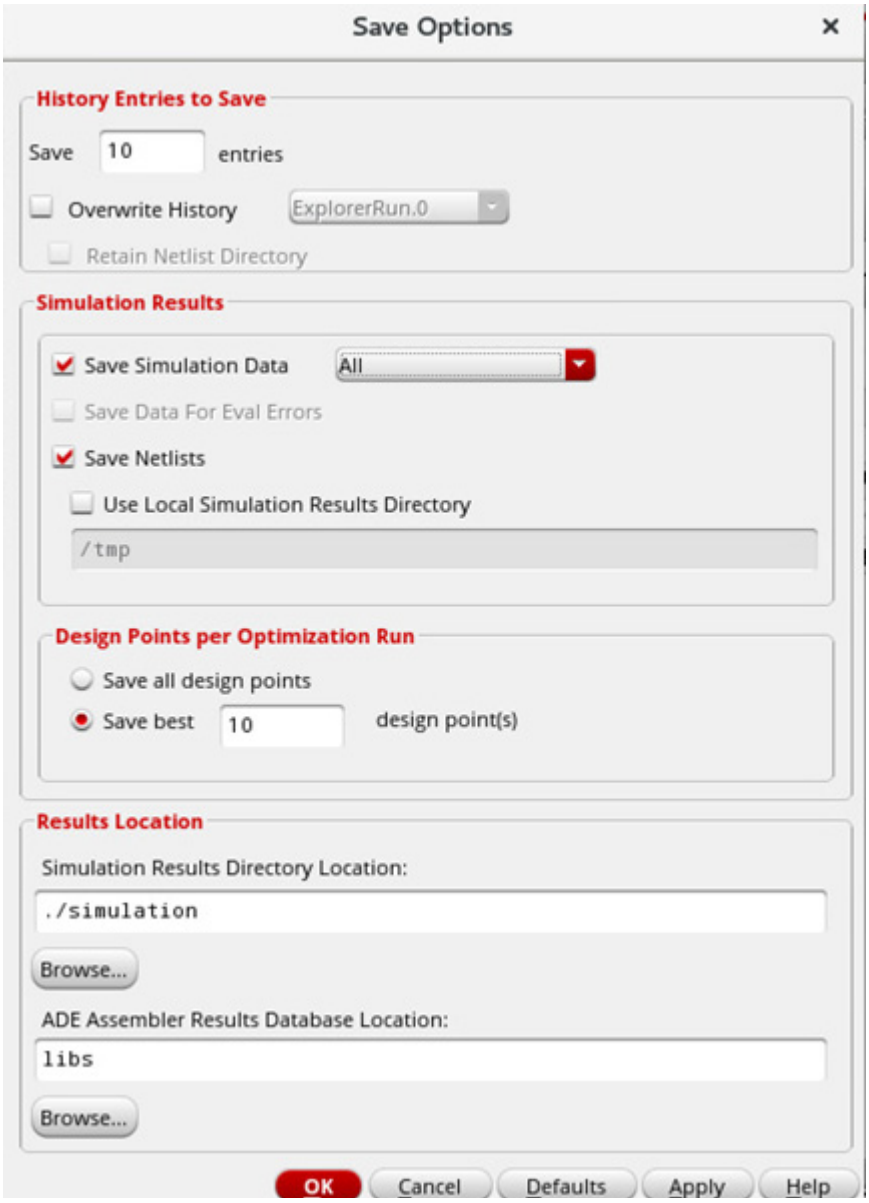
  If a new history item is created for each simulation run, the netlist and result directory paths change for each simulation run. This requires you to modify these directory paths in the OCEAN and other scripts that you use with your design for each simulation run. However, when a history item is set to be overwritten, the same netlist and result directories are used for subsequent simulation runs. This enables you to reuse the OCEAN and other scripts because you need not change the directory paths in the scripts.
- Perform any other task that requires that the netlist directory does not change for subsequent simulation runs. Such tasks include:
  - Saving simulation snapshots at specified timepoints during a transient analysis simulation that is run by using the Spectre or Virtuoso UltraSim simulator, and later restarting the simulation from a specific snapshot. For more details about how to save the state of a simulation run at a given timepoint, refer to the Virtuoso *ADE Explorer User Guide*.
  - Reusing state files created during simulation runs for subsequent simulation runs. For example, you can reuse the `spectre.ic` and `spectre.fc` state files containing DC operating point output at the first and last step of a transient analysis sweep in subsequent simulation runs.

To overwrite a history item during subsequent simulation runs, do the following:

1. Choose *Options – Save*.

© 2023 Cadence Design Systems, Inc.

The Save Options form appears.



2. Select the *Overwrite History* check box.

   Note: If the number of histories to be saved is specified as `1` in the *Save* field, the *Overwrite History* check box is automatically selected. The value displayed in the *Save* field is determined by the value of the saveLastNHistoryEntries environment variable. If required, you can edit the value in this field.

3. From the *Overwrite History* drop-down list, select one of the following:

   - Select the history item you want to be overwritten every time a simulation is run.

© 2023 Cadence Design Systems, Inc.

Note: While working with read-only maestro views, you can overwrite only the read-only histories.

- Select *Next History Run* if you want the history item created for the next simulation run to be overwritten during simulation runs.

  For example, if you run Single Run, Corners and Sweeps analysis after selecting *Next History Run*, the history item, say `Interactive.5`, that is created will be automatically selected as the overwrite history item. The `Interactive.5` history item will then be overwritten for subsequent simulation runs.

4. By default, the netlist directory of a previous run history is deleted and then created again before the next run. If you want to reuse the files related to simulation, such as the state files that are created during the previous history runs, you can select the *Retain Netlist Directory* check box. When this check box is selected, the netlist directory of the previous history is retained to reuse the simulation information in the next run. The netlist is generated incrementally in case you change any design information or some of the simulation settings.

   Note: The netlist directory of a history item is retained only if the run mode has not changed after the last simulation run.

5. Click *OK*.

Important Points to Note

- The *Overwrite History* drop-down list displays the names of only the history items that are not locked or selected as a reference history item (see Running an Incremental Simulation to Reuse Results on page 1011 for more details about reference history items).
- The *Overwrite History* option is automatically set to *Next History Run* if you lock, rename, or delete the history item that was set to be overwritten.
- The *Overwrite History* check box is automatically deselected if an operation on the history item set to be overwritten requires the history item to be referenced. History items are referenced when you do any of the following:
  - Submit a point (see Submitting a Point)
  - Simulate only error or incomplete points (see Simulating Only Unfinished or Erroneous Points)
  - Run incremental simulation (see Running an Incremental Simulation to Reuse Results)

© 2023 Cadence Design Systems, Inc.

For example, if the history item named `Interactive.5` that is set to be overwritten is also selected as a reference history item for an incremental simulation run, the *Overwrite History* check box is automatically deselected and `Interactive.5` is not overwritten during subsequent simulation runs.

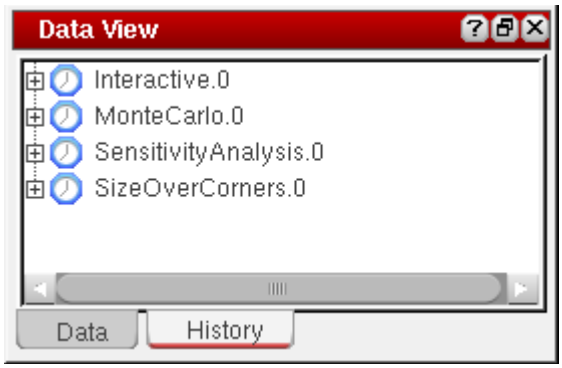
- The values for the *Overwrite History* and *Next History Run* options are saved in the setup database, and the values for the other save options are saved in the ADE Assembler project directory.
- To apply default values for the two overwrite history options that can be used by all the maestro views saved from ADE Assembler, you can use the `axlSetOverwriteHistory` and `axlSetOverwriteHistoryName` SKILL functions in the `.cdsinit` file. For more details on these functions, see *ADE SKILL Reference Guide*.

  For more details on saving simulation data by using the Save Options form, see Specifying Options for Saving Simulation Results.

© 2023 Cadence Design Systems, Inc.

Viewing Histories

To view simulation run histories, click the *History* tab of the Data View assistant pane.



All the histories saved for the cellview are listed on this tab.

The name of each history item is in the *runType*.*seqNum* format, where:

- *runType* is a unique default name corresponding to the selected run mode. For example, for the Single Run, Sweeps and Corners run mode, *runType* is `Interactive`; for the Global Optimization run mode, *runType* is `GlobalOpt`; and for simulations running a run plan, *runType* is the name of the plan.
- *seqNum* is `0` for the first history of a particular run mode in a cellview. After that, then the *seqNum* of a particular run type is incremented by 1.

When you troubleshoot a design or data point, the name of the resulting history item is in the *runType*.*seqNum*.*TS*.*seqNum* format.

To view the details for a particular history:

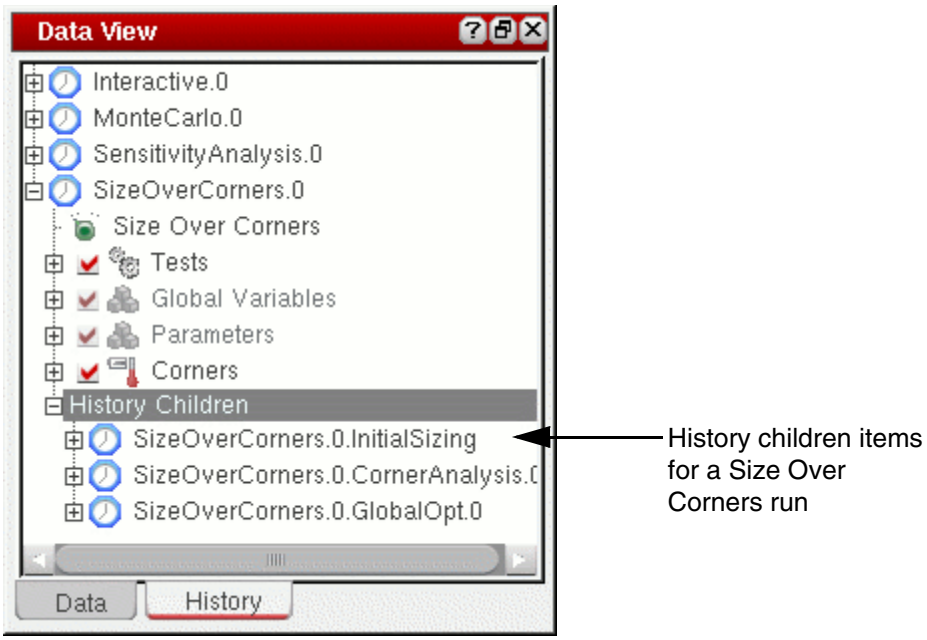
1. On the *History* tab of the Data View assistant pane, click + to the left of the history name. Alternatively, right click a history name and choose *Expand*.

   The history is expanded to display all details, including the size of the history, path to the results database, the tests that were run, the parameters that were varied, and any specifications or corners.

2. (Optional) Click + to the left of each item in the tree to expand it.

© 2023 Cadence Design Systems, Inc.

Note: The *ImproveYield*, *Manual Tuning*, and *SizeOverCorners* history items contain the *History Children* items in addition to other items saved in all histories. *History Children* contains the details of other simulations that are run during these runs.



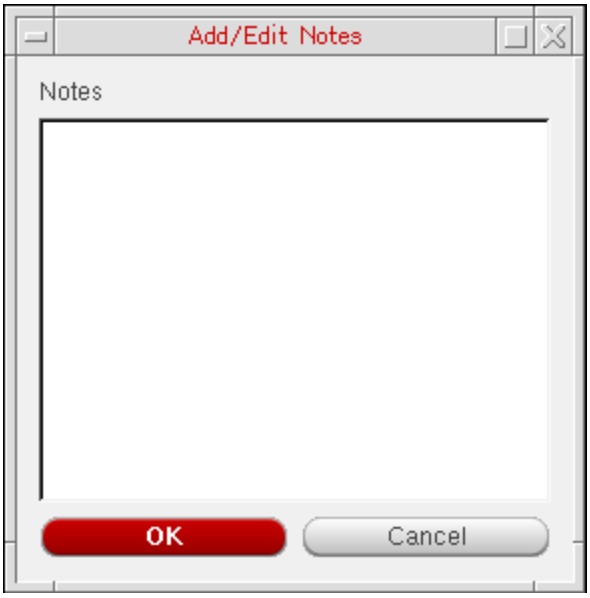
Adding Notes to a History

To add notes to a history:

1. On the *History* tab of the Data View assistant pane, right-click the history for which you want to add notes and choose *Notes*.

© 2023 Cadence Design Systems, Inc.

The Add/Edit Notes form is displayed.



2. In the *Notes* field, add notes for the history.

   Note: By default, the notes field can accept only 512 characters. To change the default maximum characters limit, you can set the maxNotesLength environment variable.

3. Click *OK*.

For related information, see Adding Notes to a Test.

Deleting Notes from a History Item

To delete notes from a history item:

➡ In the *History* tab of the *Data View* assistant, right-click the history item from which you want to delete the notes, and then choose *Delete Notes*.

Important points to note:

- The *Delete Notes* command is enabled when any of the selected history items contain notes.
- You can select multiple history items. You can then right-click the selected history items and choose *Delete Notes*. Notes are deleted from all the history items that contain notes.

© 2023 Cadence Design Systems, Inc.

Renaming Histories

When ADE Assembler creates a history, it is added to the *History* tab of the Data View assistant pane.

To rename a history, do the following:

1. Double-click the history name.

   The name becomes editable.

2. Type a new name or click-drag to highlight the portion of the name you want to change and type new text.

   The name you type must be unique and should not contain a space or any of these special characters: *, <, >, ?, |, ', \, ", ~, (,), /, and ;.

3. Press `Return` when finished.

   The new name appears in place of the old name in the History tab.

Note the following:

- If you rename *ImproveYield* and *SizeOverCorners* history items, the corresponding *historychildren* items are also automatically renamed.
- You cannot rename *historychildren* items.

© 2023 Cadence Design Systems, Inc.

Restoring a History Checkpoint

You can restore all or parts of a checkpoint history in one or more maestro cellview at the same time.

Restoring an Entire Checkpoint

To restore an entire checkpoint, do the following:

➤ In the History tab of the Data View assistant pane, right-click the checkpoint you want to restore and choose *Load Setup to Active*.

The program restores the test setup (including run mode), parameter information, and run plan to the various assistant panes. You can access results on the Results tab of the Outputs pane. The results database context in Calculator and Results Browser is also set to the results of this history. See Chapter 19, "Viewing, Printing, and Plotting Results" for more information.

Important

The *Load Setup to Active* command is not available for the `ExplorerRun.0` checkpoint. This checkpoint is saved when you run a simulation while editing a test in the ADE Explorer environment window.

Restoring Part of a Checkpoint

To restore part of a checkpoint, do the following:

1. In the History tab of the Data View assistant pane, expand the history item.
2. In the expanded history item, right-click the item or set of selected items you want to restore and choose *Load Setup to Active*.

   For example, you can

   - Right-click the *Tests*, *Global Variables, Parameters* or *Corners* branch and choose *Load Setup to Active* to restore all tests, corners, or specifications in the checkpoint.
   - Select/highlight one or more:
     - tests under the *Tests* branch, or
     - corners under the *Corners* branch.

© 2023 Cadence Design Systems, Inc.

then right-click and choose *Load Setup to Active* to restore them.

The program restores the selected item or items.

See also "Viewing Results from a Particular History Checkpoint" next.

© 2023 Cadence Design Systems, Inc.

Viewing Results from a Particular History Checkpoint

To view the results from a particular checkpoint, do the following:

➤ In the *History* tab of the Data View assistant pane, right-click the checkpoint whose results you want to view and choose *View Results*.

The results appear on a new tab in the Results tab of the Outputs pane. The name of the tab matches the name of the history item whose results you are viewing.

Note: You can also select multiple history items by holding down the `Ctrl` key. You can then right-click the selected histories and choose *View Results* to view the results from all the selected histories.

Important Points to Note

- You can open a history to view its results in multiple ADE Assembler sessions at a time.
- If a history is opened in multiple ADE Assembler sessions, and you rename or delete that history in one session, the history is closed in all the sessions except the one in which the change is initiated. Similarly, if you run a simulation to overwrite a history, ADE Assembler closes the history from other sessions, if found open.

See Chapter 19, “Viewing, Printing, and Plotting Results” for more information.

Saving Results from a Particular Checkpoint

To save the results from a particular checkpoint, do the following:

1. In the History tab of the Data View assistant pane, right-click the checkpoint whose results you want to view and choose *Save Results*.

   The Save Results form appears.

2. Enter the path to the directory where you want to save the results.

   Alternatively click the browse button to specify the directory.

3. (Optional) Select the *Copy PSF Results?* check box if you want to copy PSF results
4. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Viewing Results for a Particular Checkpoint in the Results Browser Window

To view the results for a particular checkpoint in the Results Browser window, do the following:

➤ In the History tab of the Data View assistant pane, right-click the checkpoint whose results you want to view and choose *Results Browser*.

The Results Browser window appears.

Viewing the Run Log for a Particular Checkpoint

To view the simulation run log file for a particular checkpoint, do one of the following:

➤ Click the *Open Run Log* button in the Results tab.

➤ In the History tab of the Data View pane, right-click the checkpoint whose run log file you want to view and choose *Open Run Log*.

The Run Log | Log File Viewer form appears displaying the simulation run log file.

Tip

You can also view the contents of the run log file in the Status view on the Results tab.

Opening a Terminal Window in the Results Directory for a Particular History Item

To open a terminal window in the results directory for a particular history item, do the following:

➤ In the History tab of the Data View assistant pane, right-click the history item whose results directory you want to view and choose *Open Terminal*.

A terminal window opens in the `results/data/<history_item>` directory. For example, if the history item you right-click is *Interactive.2*, the terminal window opens in the `results/data/Interactive.2` directory.

For more information about the location of the results directory, see Specifying Options for Saving Simulation Results on page 85.

© 2023 Cadence Design Systems, Inc.

Note: You can specify the command the program uses to open a terminal window using the `adexl.gui openTerminalCommand` environment variable.

ImproveYield and *SizeOverCorners* history items are slightly different from other history items in that the *Open Terminal* menu item appears on the *historychildren* items (such as *ImproveYield.0.GlobalOpt.0*) rather than on the history item itself.

To open a terminal window in the results directory for a particular *ImproveYield* history item, do the following:

➤ In the History tab of the Data View assistant pane, right-click the history children item whose results directory you want to view and choose *Open Terminal*.

A terminal window opens in the `results/data/`*`historyChild`* directory. For example, if the *historychildren* item you right-click is *ImproveYield.0.GlobalOpt.0*, the terminal window opens in the `results/data/ImproveYield.0.GlobalOpt.0` directory.

Deleting Simulation Data for a History Item

To delete simulation data for a history item, do the following:

1. In the *History* tab, right-click the history items for which you want to delete simulation data, and then choose *Delete Simulation Data*.
2. From the submenu, choose one of the following options:
   - *All*: Deletes all the simulation data and netlists
   - *All except Netlists*: Deletes simulation data, but retains netlists
   - *All except Netlists and Quick Plot Data*: Deletes simulation results, but retains netlists

Deleting a History Checkpoint

To delete a checkpoint from the *History* tab of the Data View assistant pane, do the following:

➤ In the *History* tab, right-click the history item that contains the checkpoint you want to delete and choose *Delete*.

The program removes the history item and its checkpoint from the *History* tree and the raw simulation results from the `results` directory.

© 2023 Cadence Design Systems, Inc.

Locking and Unlocking a History Item

To lock a history item so that you cannot delete it, do the following:

➤ In the History tab of the Data View assistant pane, right-click the history item you want to protect from accidental deletion and choose *Lock*.

The icon that appears to the left of the history item in the History tab becomes a lock.

You cannot delete this item while the lock icon is present.
(The *Delete* item is not available on the context-sensitive menu.)

The number of locked items does not count toward number of saved items.

Note: The *Lock* command in the right-click menu is not available for the `ExplorerRun.0` history. This ensures that it cannot be locked and if required, remains available for the next simulatoin run in ADE Explorer. ADE Explorer maintains only a single history that gets overwritten in every next run.

To unlock a history item, do the following:

➤ In the History tab of the Data View assistant pane, right-click the history item you want to unlock and choose *Unlock*.

The program replaces the lock icon with the clock icon and you can delete this history item. (The *Delete* item is now available on the context-sensitive menu.)

Note the following:

- If a history item is locked, all the *historychidren* items listed under the *History Children* tree are also locked. You cannot unlock a *historychidren* item when the history item is locked.
- If a history item is unlocked, all the *historychidren* items listed under the *History Children* tree are also unlocked.
- If you lock a *historychidren* item when the history item is unlocked, the history item and all other *historychidren* items are also locked.

© 2023 Cadence Design Systems, Inc.

Importing Histories from Another Cellview

ADE Assembler lets you to import histories from one cellview to another. By importing histories, you can bring together histories from different cellviews that have the same simulation setup and were used to run different verifications.

Note: You can import histories only for cellviews that use the separate history management feature.

To import a history from one cellview to another:

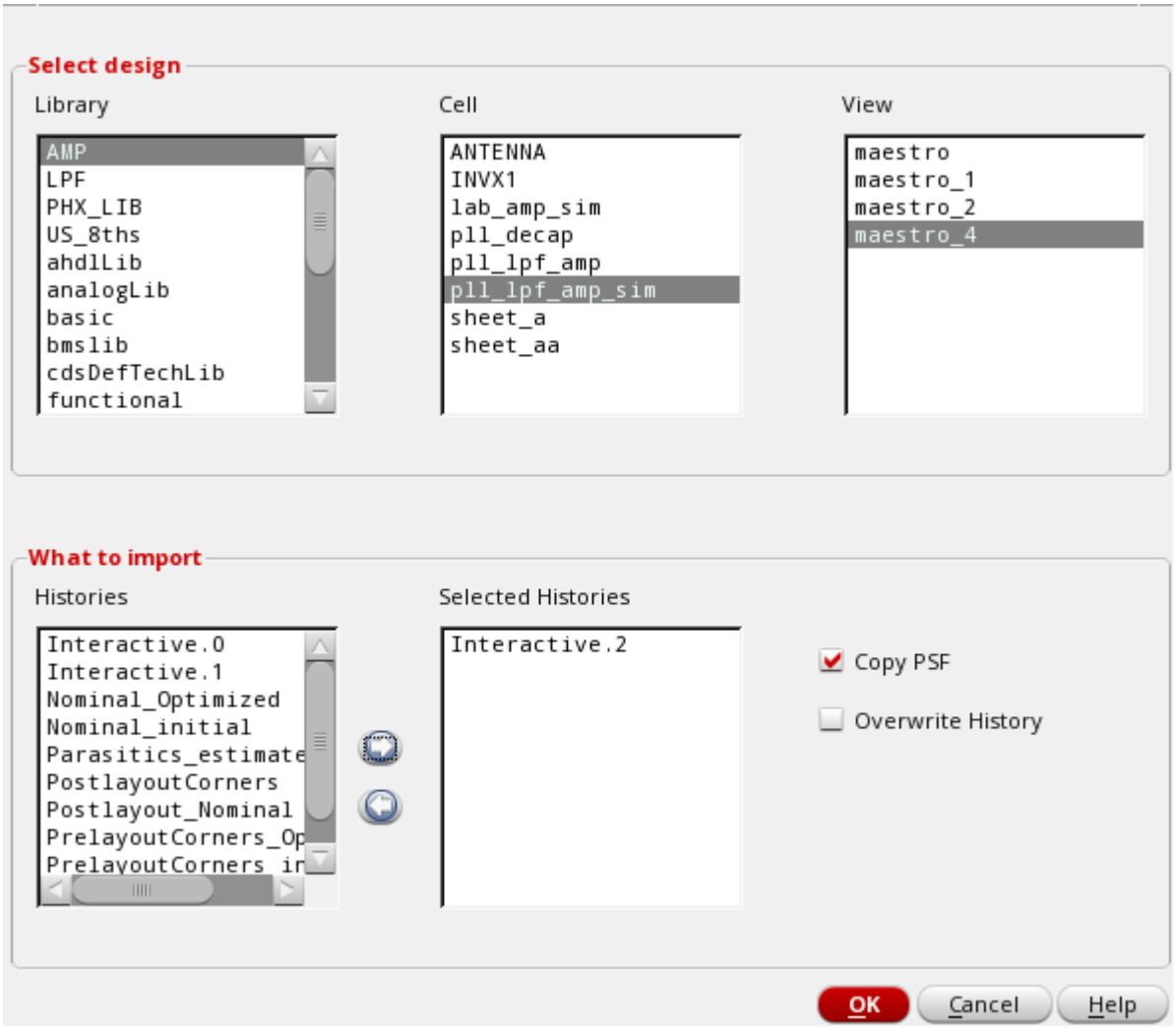
1. Open the maestro cellview to which you need to copy a history.
2. Choose *File – Import History*.

   The Import Histories form is displayed.
3. In the *Select design* section, choose the source cellview that contains the history to be imported.

   The *Histories* list in the *What to import* section displays the names of all the histories saved in the selected cellview.
4. Select one or more history names in the *Histories* list.

© 2023 Cadence Design Systems, Inc.

5. Click the right arrow to move the selected items to the *Selected Histories* list.



6. Use the *Copy PSF* check box to specify whether the `psf` directory is imported along with the history database. By default, the directory is imported.

7. Select the *Overwrite History* check box to overwrite any history of the same name already existing in the destination cellview.

   By default, this check box is not selected and the tool does not import any history names that already exists in the destination cellview.

8. Click *OK* to import the selected histories.

   The zip files of the imported histories are copied from the source cellview to the `history` directory in the destination cellview. In addition, new entries are added to the history

© 2023 Cadence Design Systems, Inc.

database saved in the `history.sdb` file at the *`libName/cellName/viewName`*`/history/` location.

For example, a history `Interactive.2` imported into the `myLib/myCell/myView` cellview is saved at the following location:

```
/myLib/myCell/myView/history/Interactive.2.zip
```

Related Topics

maeImportHistory

Merging Histories

You can merge the results from multiple histories into a single history without rerunning a simulation. Merging histories not only merges the results from different histories, but also merges setups used in those histories.

You can also choose to preview setup and modify it before merging results of the selected histories.

This section includes the following topics:

- Merging Setups and Results from Multiple Histories
- Merging Selected Run Points
- Customizing Setup before Merging Histories
- Limitations of Merging Histories

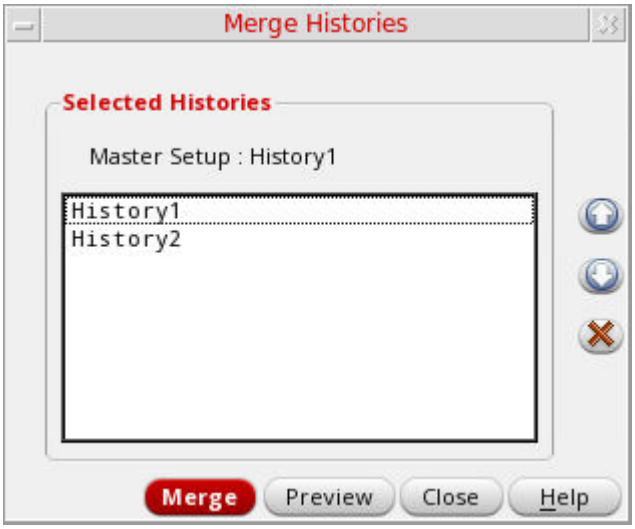
Merging Setups and Results from Multiple Histories

To merge setups and results from multiple histories:

1. In the *History* tab of the Data View assistant, select the histories that you want to merge to a new history. For example, `History1` and `History2`.
2. Right-click the selected histories, and then choose *Merge And Load to Active*.

© 2023 Cadence Design Systems, Inc.

The *Merge Histories* form appears, displaying a list of selected histories. The history displayed at the top of list is used as the master setup. You can use the arrow buttons to change the master setup.



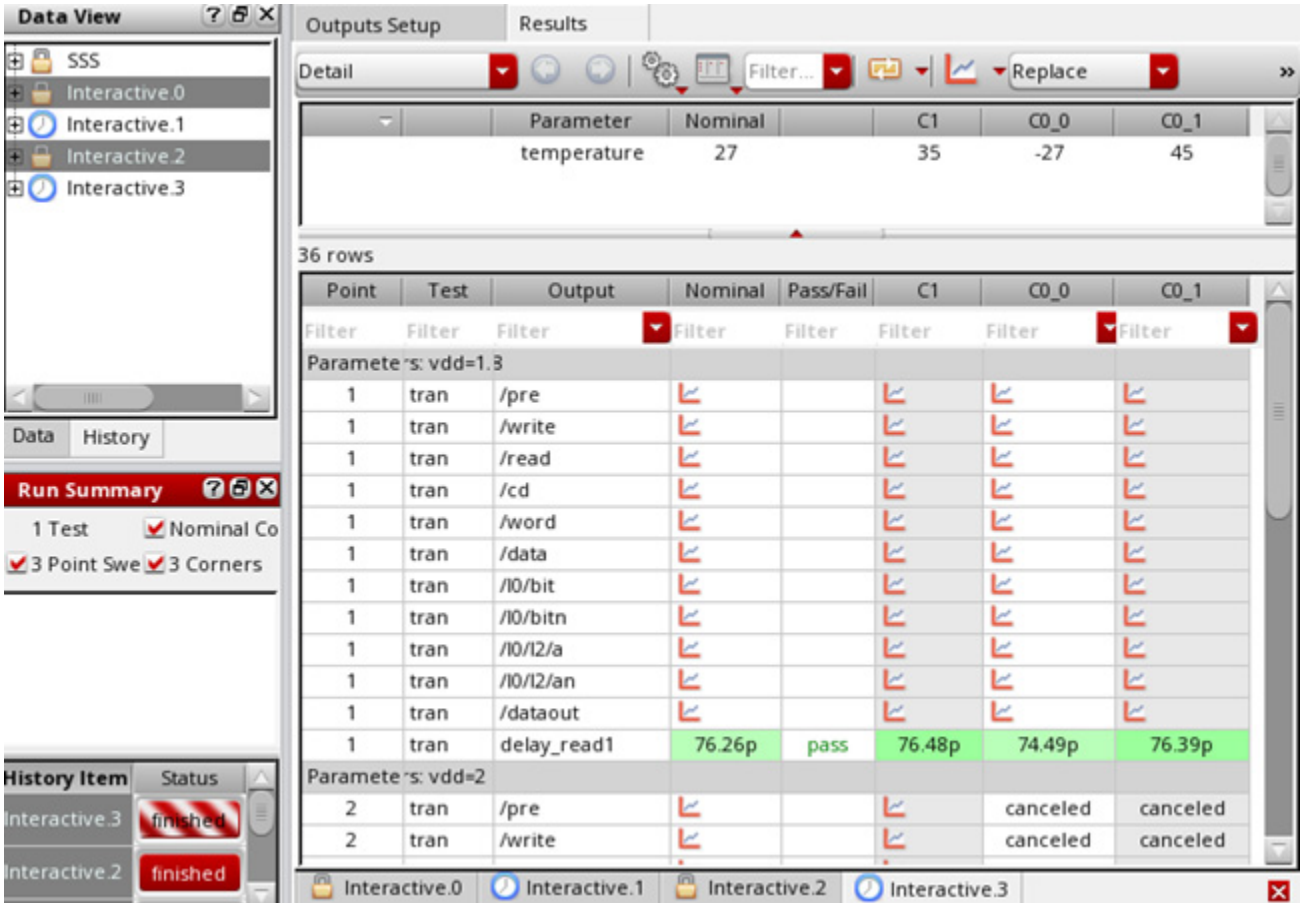
3. Click *Merge* to merge the setups and results from the selected histories.

   A message box may appear, indicating that there have been some changes to the active setup.

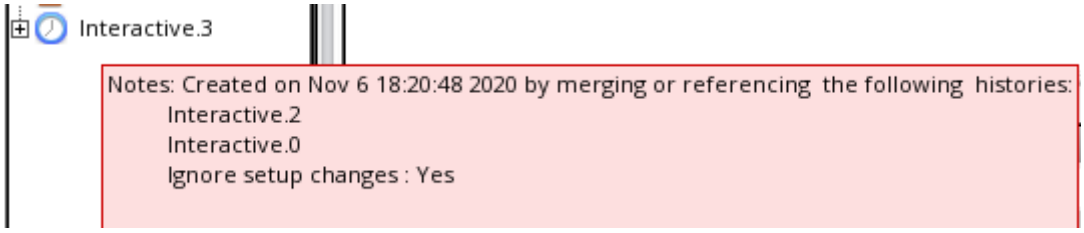
4. Click *Yes* to continue.

© 2023 Cadence Design Systems, Inc.

The setup from the master history is loaded as the active setup. Setup from other histories are merged into it. The results are also merged into a new history that is saved with a new name `Interactive.`*`incremented_number`*.



Note: You can hover over a history to check whether it is created by merging other histories. A merged history shows a tooltip with the names of histories that were merged to create it.



© 2023 Cadence Design Systems, Inc.

Important Points to Note

- It is recommended not to merge histories when the variables and output setups do not match in them.
- Settings, such as corners, tests, and variables are merged from all selected histories.

  For more information about how variables are corners are merged, see How does ADE Assembler merge variables and corners?
- The merged setup may have more simulation points than the sum of those are there in the histories being merged.

  For example:

History1	`vdd= 1 2 3`	Corners= `C1` and `C2`
History2	`vdd= 1 4`	Corners= `C1`
Merged Setup	`vdd= 1 2 3 4`	Corners= `C1` and `C2`

Merging Selected Run Points

Using *Preview* option on the Merge Histories form, you can preview the merged setup before merging results of the histories. This feature provides you the flexibility to selectively merge a subset of the points available in the merged setup.

To selectively merge a few run points:

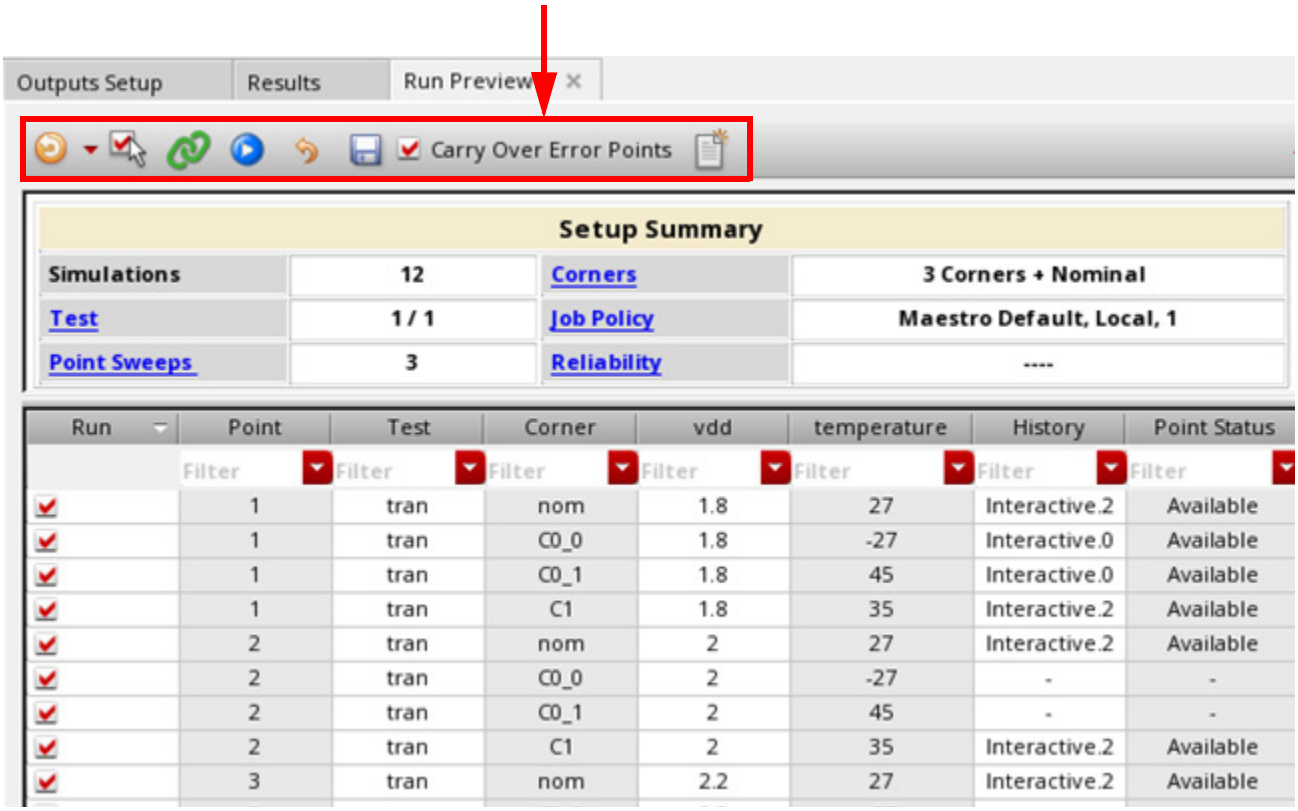
1. In the *History* tab of the Data View assistant, select the histories that you want to merge to a new history. For example, `History1` and `History2`.
2. Right-click the selected histories, and then choose *Merge And Load to Active*.

   The *Merge Histories* form opens, displaying a list of selected histories.
3. Click *Preview*.

© 2023 Cadence Design Systems, Inc.

The *Run Preview* tab opens, displaying a preview of the merged setup with its summary.

The Run Preview toolbar during merging histories



The *Run Preview* tab shows the following information:

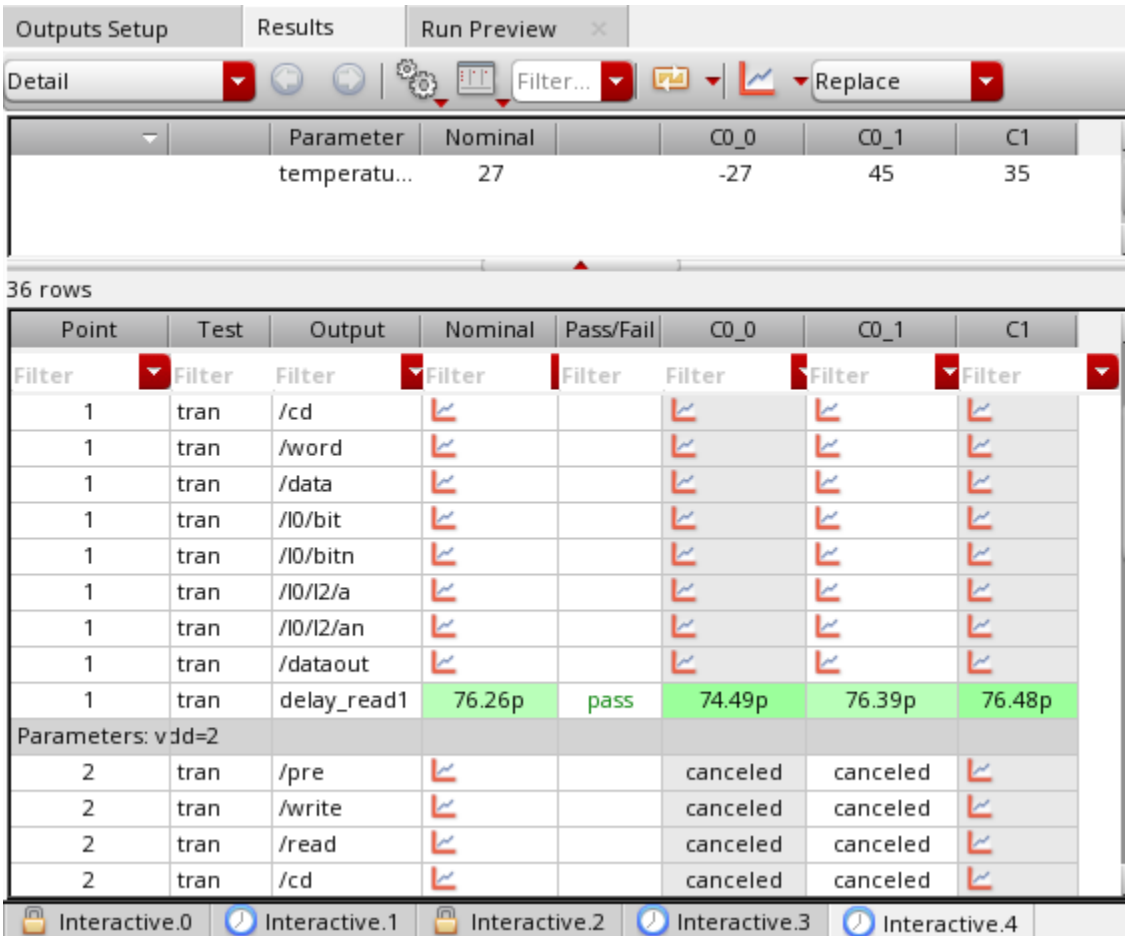
- The *Setup Summary* section displays the number of unique tests, corners, sweep points, and number of simulations.
- The table below the summary shows the details of all the points and their results. The *History* column displays the name of the histories from which the results are retrieved. These histories are considered as reference histories from which results are reused in the next run. The blank entry in the *History* column against a run point indicates that no results are available for that point.
- The *Point Status* column displays the status of a point. This column can display the following values:
  - `Available`–Indicates that the results for this point are found in the history mentioned in the *History* column
  - `Error`–Indicates that the results for this point were not generated successfully due to an error in the history mentioned in the *History* column

© 2023 Cadence Design Systems, Inc.

- (–) –Indicates that the point is not found in any history merged to show this preview

4. On the *Run Preview* tab, deselect the run points that you do not want to merge.
5. Click on the *Run Preview* toolbar to merge only the selected results.

   A new history is created in which the results for the selected points are merged. No new simulations are run. The points for which results are not available are shown as canceled in the Detail view.



Customizing Setup before Merging Histories

Before merging the results of histories, you can review their merged setup and make changes in the test, variable, or corner settings. This allows you to reuse results for existing points from the merged histories and run simulations for additional points that you may want to add.

To preview and customize the merged setup:

© 2023 Cadence Design Systems, Inc.

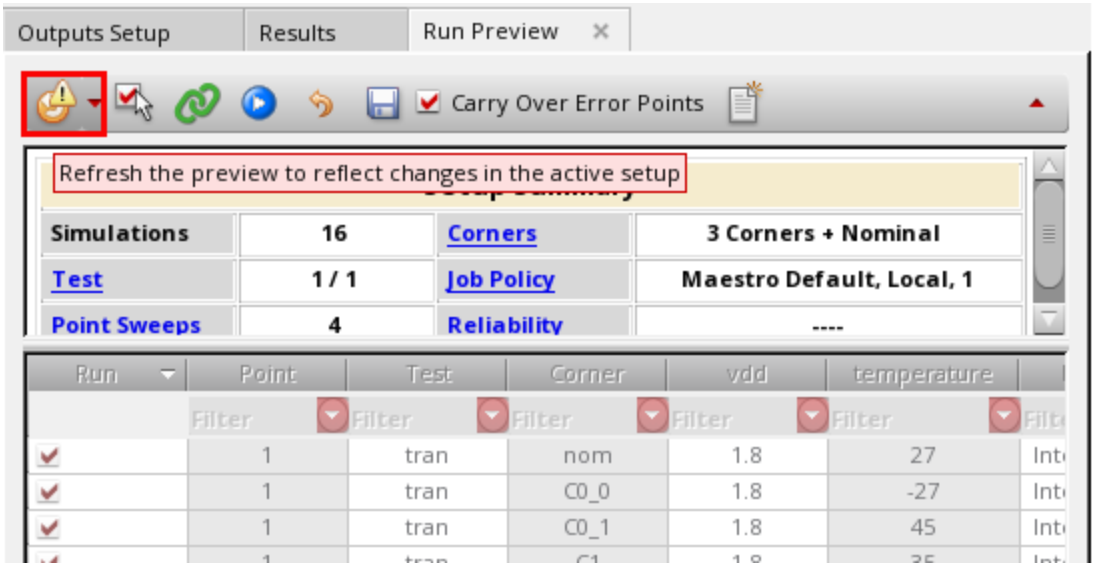
1. Click *Preview* on the Merge Histories form to preview the merged setup in the *Run Preview* tab.

   The active setup of the maestro cellview is also updated according to the merged setup.

2. Modify the active setup as required.

   For example, you can add or modify corners. When you modify the active setup, the preview is disabled and the *Refresh the preview to reflect changes in the active setup* command on the *Run Preview* toolbar shows an exclamation mark (!).

3. To reflect the changes from active setup to the preview, click the *Refresh the preview to reflect changes in the active setup* command on the *Run Preview* toolbar.

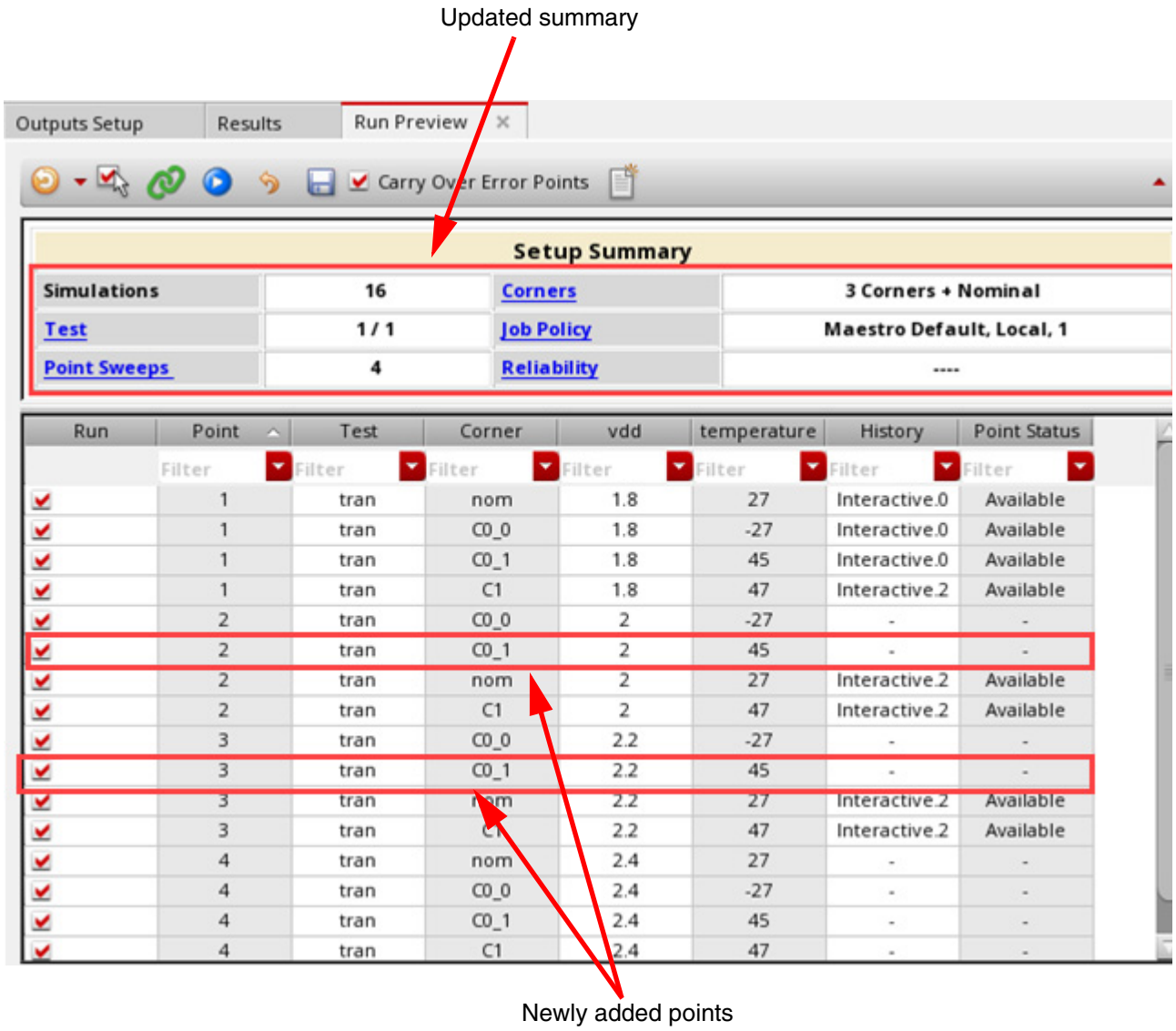


   The details are updated in the *Preview* tab, and the tab is enabled for further changes. If required, new rows are added. In addition, the summary is also updated.

   For example, adding a new corner `C2`, which is being swept adds new rows in the details. Observe the changes in the number of simulations, point sweeps and corners. Entries in

© 2023 Cadence Design Systems, Inc.

the *History* column for newly added points show `(-)` because these points were not available earlier in the histories being merged.

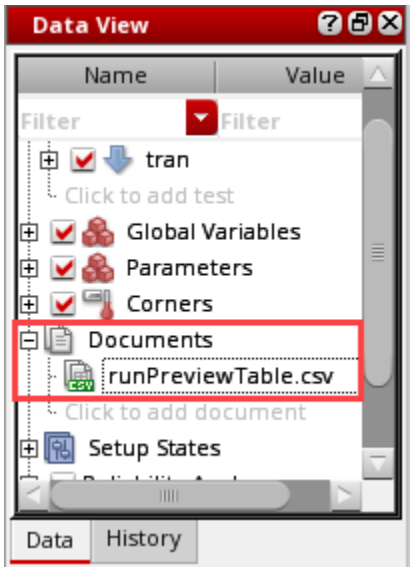


4. Select *Carry Over Error Points* if you want to retain the status of error points from the reference history. When this check box is deselected, ADE Assembler also simulates the points that resulted in errors in the previous run.
5. Click *Document Run Preview Table in CSV Format* to document the details of the points of merged histories in a `.csv` file for future reference.

   The Document Run Preview Table form is displayed. Specify a location and name for the file where you want to save the details.

© 2023 Cadence Design Systems, Inc.

The details from the Run Preview table are saved in a `.csv` file. The file is also shown in the *Documents* tree in the Data View assistant.



6. Click *Run Simulation* on the Run toolbar to run the simulation to get results for the updated points in the modified setup.

   A new history is created to which results available in the merged histories are copied. The newly-added points are simulated and their results are added to the new history.

Important Points to Note:

- The points for which results were available in the merged histories are not rerun. Results for these points are copied from their results database.
- You can check the timestamps in the *Run Log* to double-check when exactly the simulation has been performed.
- If at any point, you decide not to merge the results of histories, click to roll back the changes made to the active setup. Settings copied from the merged histories to the active setup are reverted.
- The *Reference History* toolbar is disabled in the *Preview* mode.
- If you exit the *Preview* mode by closing the *Run Preview* tab, a dialog box appears, asking you to confirm whether you want to save or discard the changes made in the active setup.
- You can save the merged setup either by clicking the *Save* button on the *Run Preview* toolbar or on the Assembler toolbar.

© 2023 Cadence Design Systems, Inc.

How does ADE Assembler merge variables and corners?

ADE Assembler merges variables and corners as follows:

- If the global and local variables for all tests in all histories being merged are same, their values are merged to create sweep points for variables.

History1	`vdd= 1 2 3`	Corners= `C1` and `C2`
History2	`vdd= 1 4`	Corners= `C1`
Merged Setup	`vdd= 1 2 3 4`	Corners= `C1` and `C2`

- A variable in the merged setup is assigned values from all histories irrespective of its global or local status in each history.
- If possible, the merged history keeps the same setup for global or local variables as in the master history. In case of conflicts in the settings across histories, ADE Assembler finds the best possible way.
- Corner definitions are used from the master setup when merging setup.
- Assume that you run a simulation and create a copy of a corner resulting from a corner group. Next, you run a simulation and then merge the results of the latest run with the previous run. In this scenario, it is possible that a corner of the latest run has the same name as that of a corner resulting after expanding a corner group in the master setup. In such cases, ADE Assembler retains the duplicate corner and its result from the reference history in the merged setup. The duplicate corner in the master setup is renamed.

  If you do not want to rename corners from the master setup, set the defaultReliabilityMode environment variable to `t` to remove the duplicate corner from the reference history and retain only the corners resulting by expanding the corner group.
- If any of the histories being merged is using a corner that is not available in the master history, the corner is added to the merged setup.
- If the same corner name is used in multiple histories, but the corner setup is different, the corner from the master setup is used. The corner with the same name from other history is ignored.
- Asserts can be defined in include files, schematic, or both. The histories being merged can have different setup for asserts. For example, `history_1` has asserts defined in include files and `history_2` has asserts defined in the schematic. While merging histories, the definition of the asserts is retained from the master setup. However, the violation results are copied from the merged histories even if their assert setup is different. New simulation runs use the assert definition from the master setup.

© 2023 Cadence Design Systems, Inc.

Limitations of Merging Histories

You cannot merge histories in the following cases:

- A history is saved for a run mode other than *Single run, sweeps, and corners*.
- A history was saved for a run plan or it was run in ADE Explorer.
- A history contains results of an incomplete simulation.
- Reliability analysis is enabled in a history setup.
- The parameters do not match in the histories being merged.

You can merge histories only when both the master and the reference histories are generated using a release version that either precedes or follows IC6.1.8 ISR14 or ICADVM20.1 base release.

© 2023 Cadence Design Systems, Inc.

21

Working with Run Plan

Run Plan in ADE Assembler provides the capability to create multiple variations of the setup within a single session, where each run has its own setup details that override the settings in the active setup. Simulations can be run for all the runs in a session together with a single click and their results are generated simultaneously. The different runs can differ in various ways. For example, they can vary in terms of the tests, analyses, or simulation options, or they can run different run modes.

ADE Assembler provides the Run Plan assistant that you can use to create and execute a run plan.

This chapter describes the following:

- The Run Plan Assistant
- Creating a New Run
- Modifying a Run in the Run Plan
- Enabling or Disabling Runs in a Run Plan
- Rules to Use Local or Global Variables for Run Plan Simulations
- Setting Up a Run Plan with the Worst Case Corners Run Mode
- Executing the Run Plan
- Viewing Results of Run Plan Simulations
- Rerun Simulations for Erroneous or Unfinished Points in a Run Plan
- Using Results of One Run in Another

Related Blogs

- Virtuosity: What's New in Run Plan - Part I
- Virtuosity: What's new in Run Plan - Part II
- Virtuosity: What's new in Run Plan - PartIII

© 2023 Cadence Design Systems, Inc.

Video

You can view a video demonstration for this feature at Creating Run Plans in the ADE Assembler.

© 2023 Cadence Design Systems, Inc.

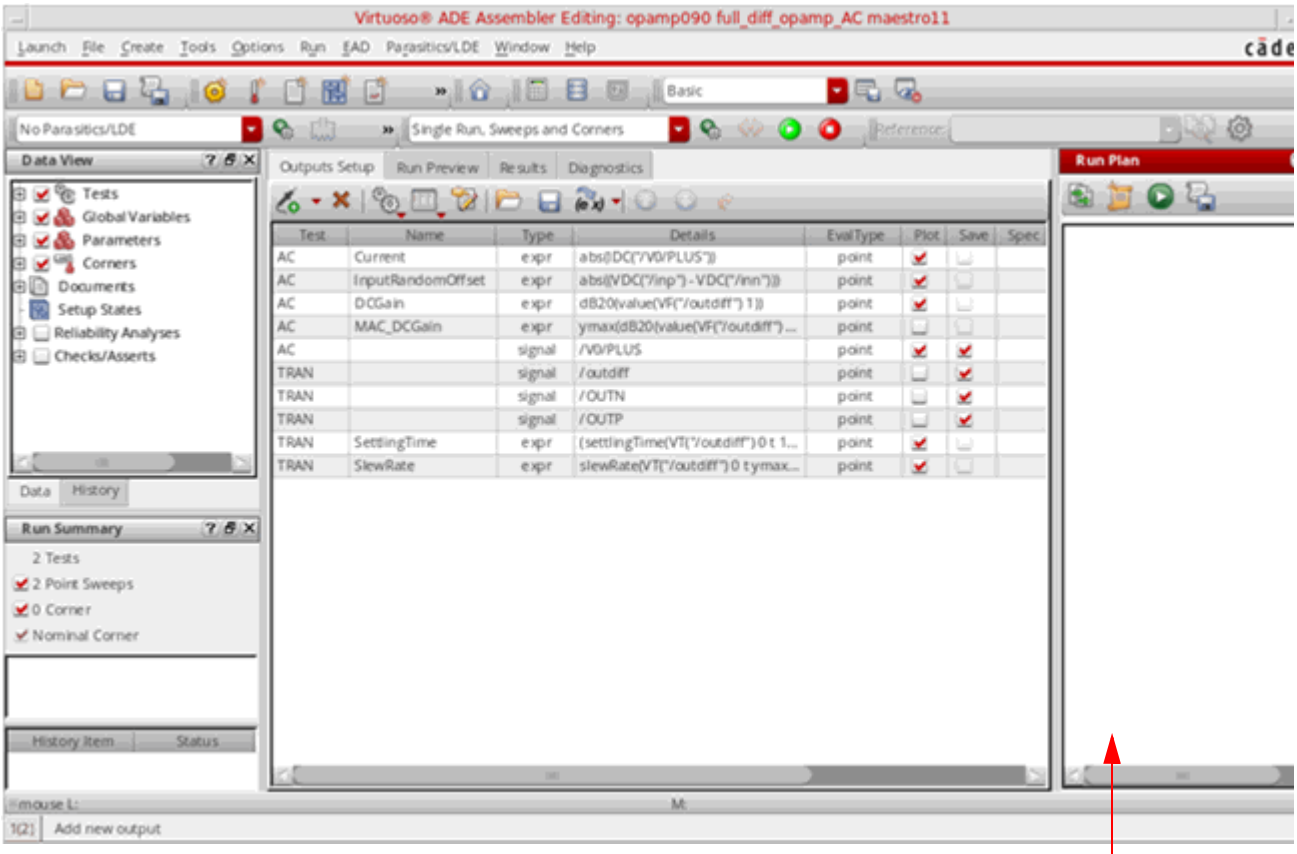
The Run Plan Assistant

The Run Plan assistant is used to create and manage run plans.

To open the Run Plan assistant, do one of the following:

- Choose *Create – Run Plan*
- Choose *Window – Assistants – Run Plan*

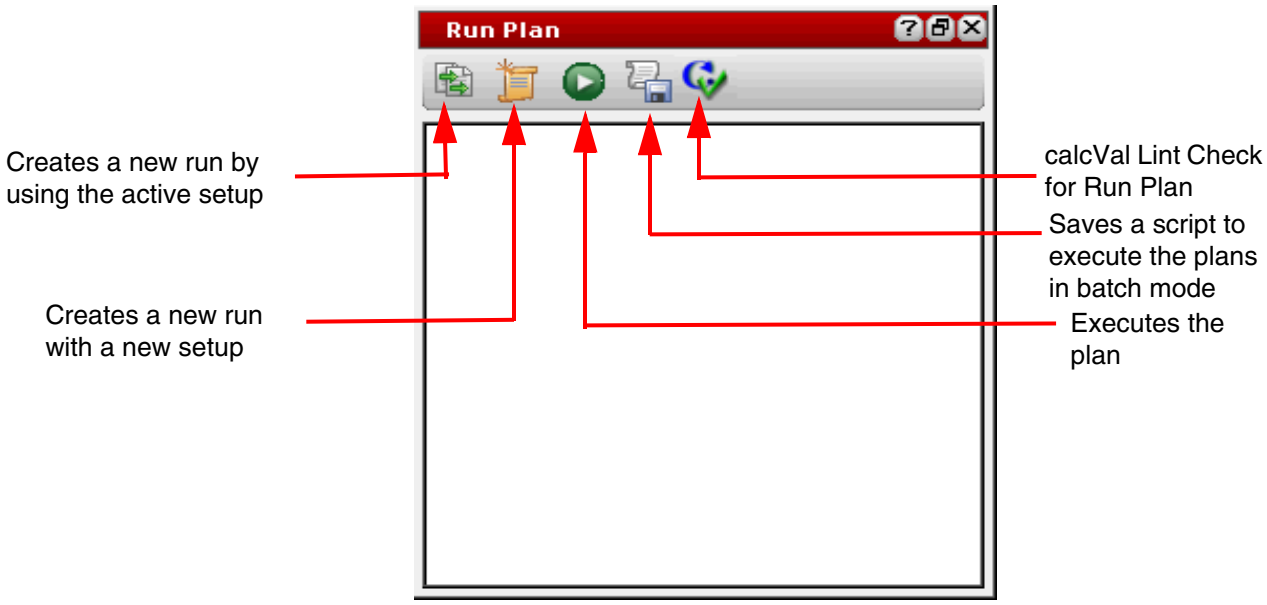
The Run Plan assistant is displayed to the right of the Outputs pane in the ADE Assembler environment.



The Run Plan assistant

© 2023 Cadence Design Systems, Inc.

The following figure describes the important commands provided in the Run Plan assistant.



Note: By default, the Run Plan assistant is blank. You can begin with creating runs. For details on how to add new runs in the Run Plan assistant, refer to Creating a New Run.

© 2023 Cadence Design Systems, Inc.

Creating a New Run

You can create a run in any of the following ways:

- By using the active setup as base and modifying it as per requirements
- By creating a new setup
- By copying another run

Important

If you want to reuse a run plan saved in a setup state of the current maestro cellview, or in another cellview, you can use the Load Setup State or Import Setup forms to update the current run plan. Depending on the retain, merge, or overwrite operation selected on that form, the current run plan is updated.

Creating a Run by Using the Active Setup

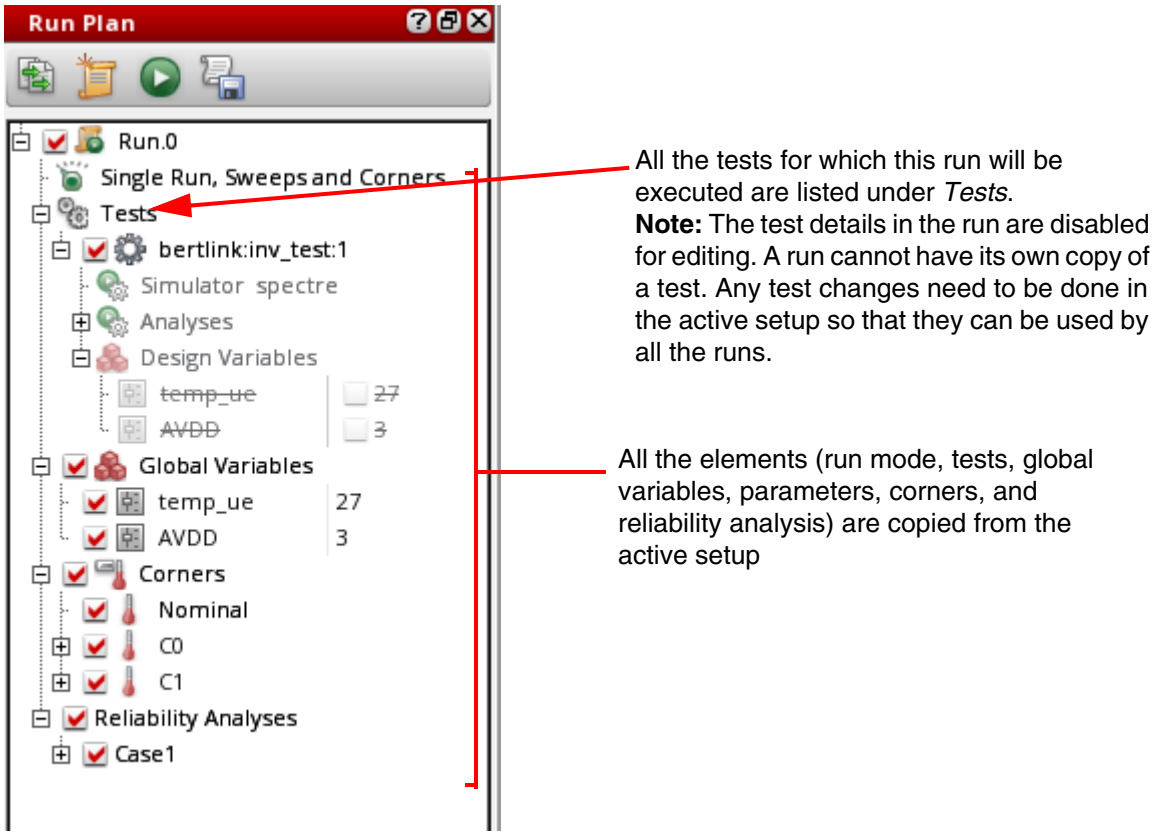
To create a run by using the active setup as base, do the following:

1. Click *Create New Run using Active Setup* ( ) on the toolbar of the Run Plan assistant.
2. A new run is added to the Run Plan assistant with the default name `Run.`*`n`*, where *`n`* is the incremental count of the number of run plans.

   You can change the name of the run, but ensure that it does not contain any special character.

© 2023 Cadence Design Systems, Inc.

3. Expand the tree for the newly added run to view its details.



Note the following points with respect to the new run:

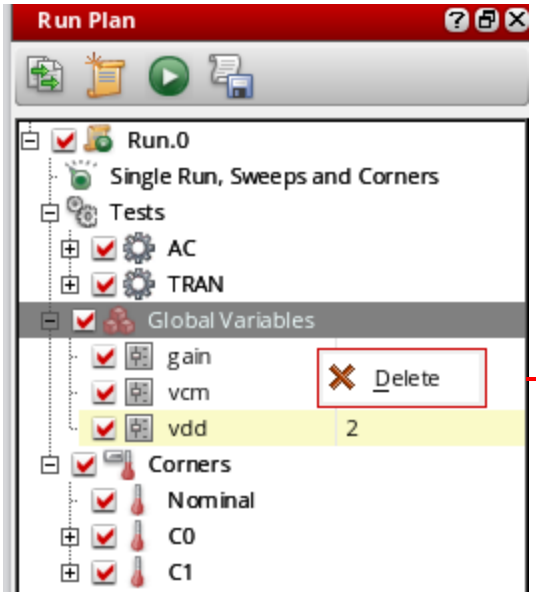
- The run mode and its simulation options are copied from the active setup to the new run. The name of the run mode is visible in the run tree.
- If the run mode in the active setup is not supported in the run plan, it is changed to the default run mode, `Single Run, Sweeps, and Corners`.
- The parasitic setup in the maestro view is ignored by the run plan.
- All the tests for which this run will be executed are listed under *Tests*.
- The details of global variables, parameters, and corners are copied from the active setup to the new run. These are kept synchronized with the corresponding elements in the active setup. For example, if any corner or variable is renamed, modified, or deleted, the change is reflected in the run plan too. However, any new variable or corner is not added automatically. You need to drag and drop that on a run.
- Parametric sets for global variables are also copied from the active setup to the new run.

© 2023 Cadence Design Systems, Inc.

- If you rename a test in the active setup, the name of the test is updated in the runs where it is referenced. If you delete a test, it is deleted from all the runs where it is referenced.
- By default, the enabled or disabled status of the check boxes next to the run plan elements is same as that of the active setup. You can change their status as per the requirements of the run plan. For example, you can disable any corner or test not required for the given run, or you can disable a global variable to use the local value of a design variable. To enable a setting, ensure that the check box next to it is selected.

  Note: To enable or disable multiple corners, select the corners, right-click and choose the *Enable* or *Disable* command from the context-sensitive menu.

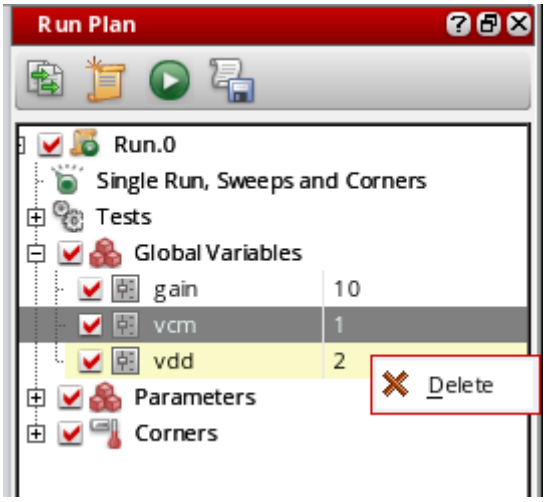
  If any test, global variable, or parameter is not applicable for the run, you can delete it. For this, right-click the element to be deleted and click *Delete*. Either you can delete selected elements from a category or delete the entire category to remove all the elements. For example, you can delete `Global Variables` to delete all the global variables in the run. In this case, the global variables defined in the active setup will be used in the simulation for this run.



Deletes all the global variables from Run.0. The global variables defined in the active setup will be used in the simulations for this run.

© 2023 Cadence Design Systems, Inc.

If you delete a selected global variable, all the remaining global variables in the run will be used in the simulation. The variables from the active Assembler setup will not be used.



Deletes only one global variable from Run.0. The remaining global variables can be used in simulations for this run.

- You cannot add a new corner to a run. However, when you are running a simulation to identify and create worst case corners, new corners are created in the setup of that run. Such worst case corners can be used in other dependent runs. For more details, refer to Using Worst Case Corners Created by Another Run.

To know how you can edit a run, refer to Modifying a Run in the Run Plan.

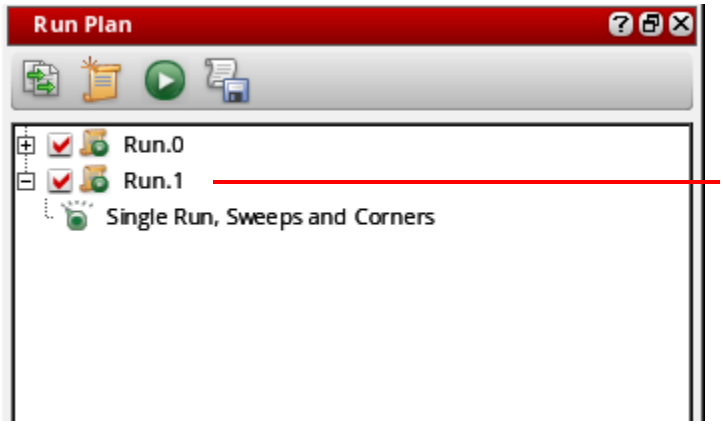
Creating a Run with a New Setup

To create a run with a new setup, do the following:

1. Click *Create New Run* ( ) on the toolbar of the Run Plan assistant.

© 2023 Cadence Design Systems, Inc.

A new run is added to the Run Plan assistant with the default name `Run.`*`n`*, where *`n`* is the incremental count of the number of run plans.



Note: The default run mode for a new run is *Single Run, Sweeps and Corners*.

2. If required, change the run mode. For this, right-click *Single Run, Sweeps and Corners* and select *Change Run Mode* – *<name of run mode>*.
3. Right-click the name of the run mode and choose *Simulation options*.

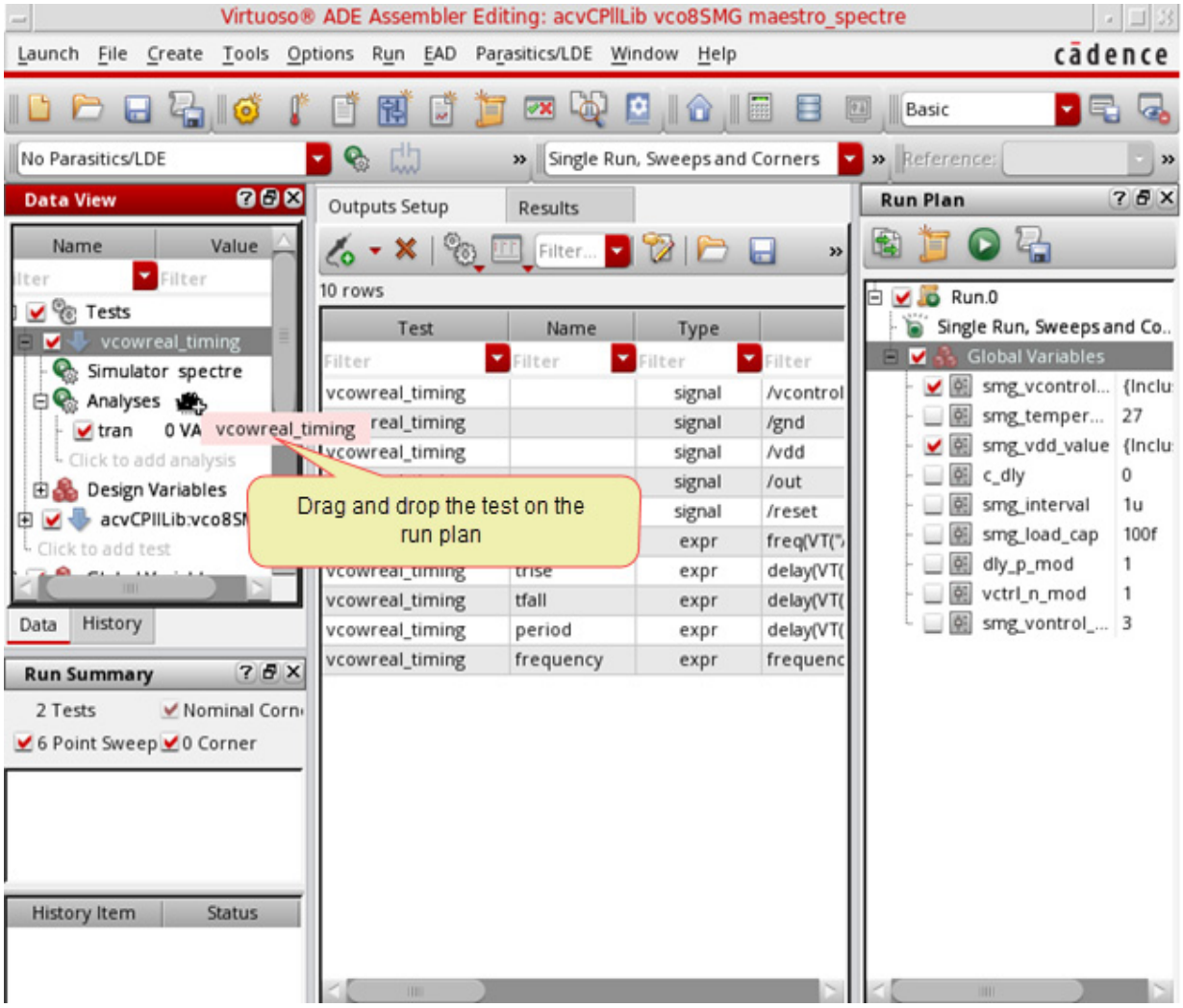
   The simulation run options form corresponding to that run mode is displayed. Make the required changes in the simulation settings.

   A new run is added to the Run Plan assistant with the default name `Run.`*`n`*, where *`n`* is the incremental count of the number of run plans.

   You can change the name of the run, but ensure that it does not contain any special character.

© 2023 Cadence Design Systems, Inc.

4. Add the elements that you need to override in the run. For example, if you want to override the tests defined in the active setup, select and drag the required tests from the Data View assistant and drop those on the name of the run, as shown below.



   The `Tests` category is added to the run setup and the tests are listed under that.

5. Similarly, you can add more elements to the run.

6. If required, edit the details of elements. For example, you can override the values of the global variables. For this, drag and drop those to the run and modify their values.

To know how you can edit a run, refer to Modifying a Run in the Run Plan.

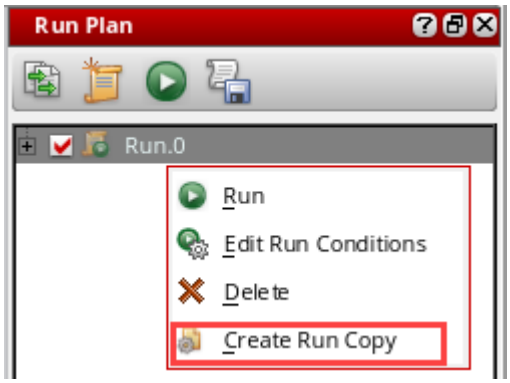
© 2023 Cadence Design Systems, Inc.

Creating a Run by Copying Another Run

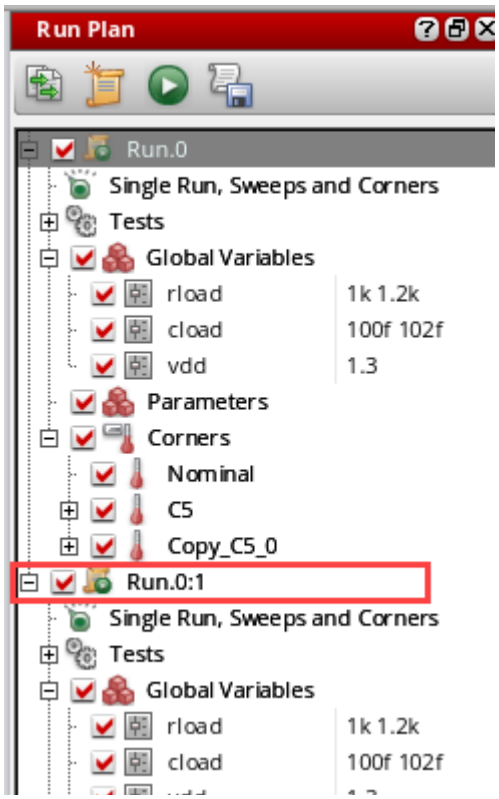
If you need to create two similar runs, you can set up one run and create its copy. After that, modify one copy to update the existing one or add more settings.

To create a run by copying another existing run, do the following:

➡ Right-click a run in the Run Plan assistant and choose *Create Run Copy*.



A copy of the selected run is created and added to the tree of runs in the Run Plan assistant.



© 2023 Cadence Design Systems, Inc.

Observe the following:

- The name of the newly created run is `<name-of-the-source-run>:<sequence-of-copy>`. You can change the name of the run, but ensure that it does not contain any special character.
- When the copy is created, all the elements, including the pre- or post-run scripts and edit conditions, of the source and the copied runs are the same. You can now modify the two runs independently.

Note: For details about how you can edit a run, refer to Creating a Run by Using the Active Setup.

Setting Up a Run Plan with the Worst Case Corners Run Mode

To set up a run with the Worst Case Corners run mode, do this in the Run Plan assistant:

1. Create a run with the run mode set as `Worst Case Corners`.

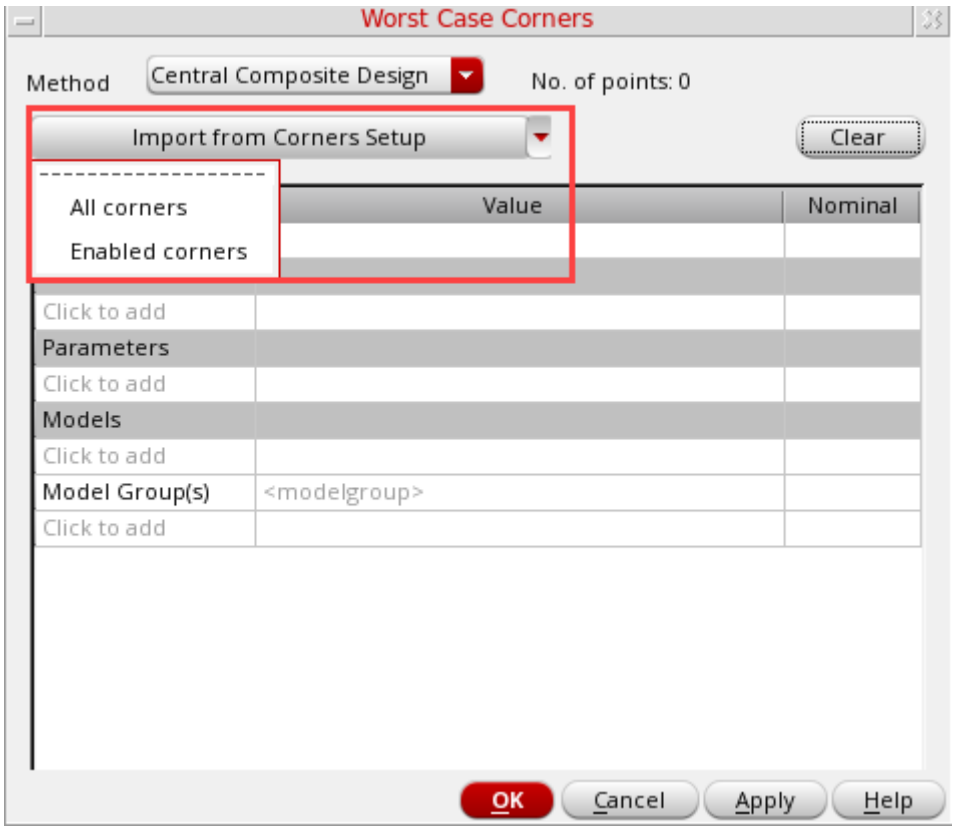
   You can create a run by copying from the active setup or by adding specific elements in a blank setup.

2. Edit the simulation options for worst case corners. For this, right-click the run name and choose *Edit Simulator Options* from the context-sensitive menu. For details, refer to Worst Case Corners.

3. If you want to run the worst case corners simulation on the existing corners in the setup, import the global variables, parameters, and model file details from the corners setup.

© 2023 Cadence Design Systems, Inc.

To do this, click *Import from Corners Setup* in the Worst Case Corners form and choose an appropriate option to import the corner details from all corners or only enabled corners.

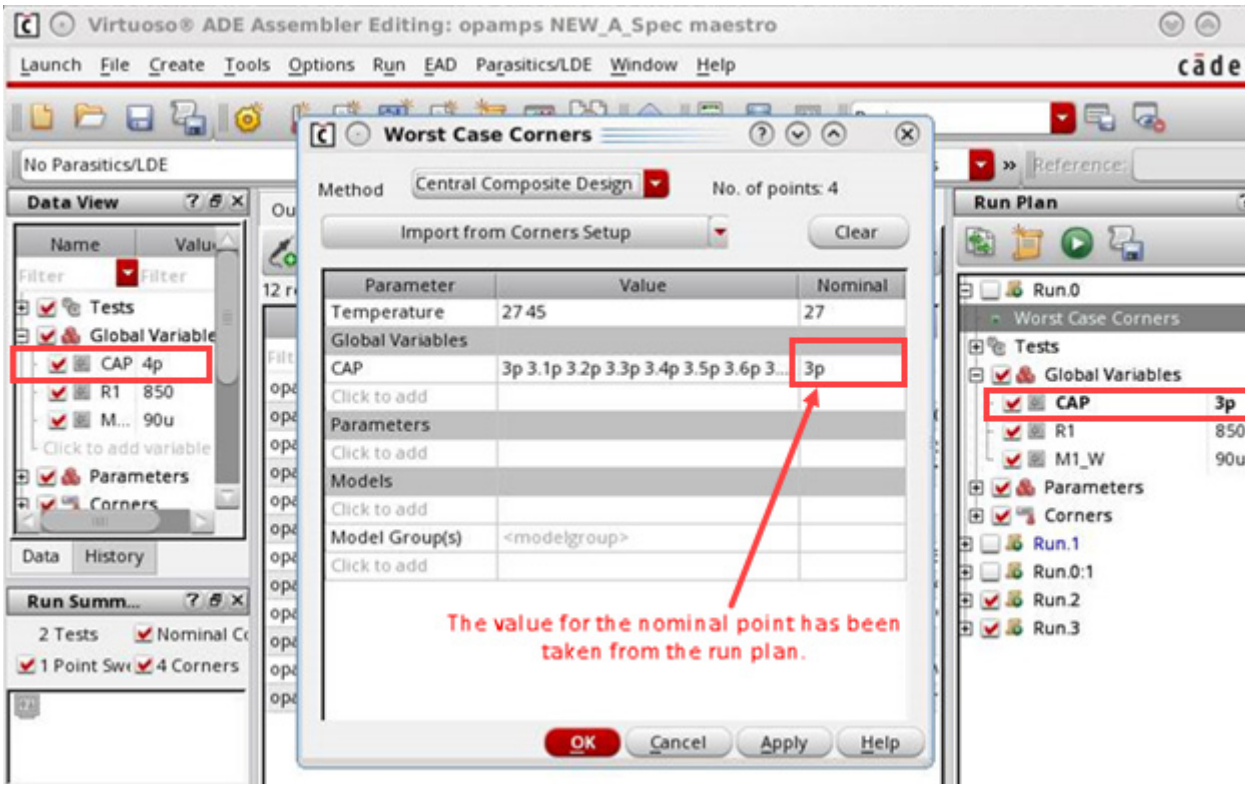


The following points describe how the corner details are imported depending on the corner setup in the run:

- If the setup of a run contains global variables, when you import corners, the values of variables are taken from the run setup. Only for the global variables that are taken from the active setup because they are not in the run setup, the values are also taken from the active setup.
- If the run does not contain corners, but the active setup contains enabled corners, simulation for the worst case corners run uses the corners from the active setup. That means, when you click *Import from Corners Setup* on the simulator options form, the corners are imported from the active setup.
- If the run setup contains corners, it uses the local run setup and ignores the corners in the active setup. In this case, when you click *Import from Corners Setup* on the

© 2023 Cadence Design Systems, Inc.

simulator options form, the corners are imported from the run setup, as shown below.



A run plan with the `Worst Case Corners` run mode adds the newly identified worst case corners to the run plan setup as well as the active setup in ADE Assembler. By default, the corners in the active setup are marked as disabled. To change the default status of these corners to `enabled`, set the enableWCCfromRunPlanInActiveSetup environment variable to `t`.

Note: You can use the worst case corners identified by one run into the another run. For more details, refer to Using Worst Case Corners Created by Another Run.

Modifying a Run in the Run Plan

After creating a new run, you can modify its elements as per your requirements. The following topics explain how to modify a run:

- Changing the Run Mode
- Modifying the Simulation Options
- Modifying the Values of Global Variables and Parameters

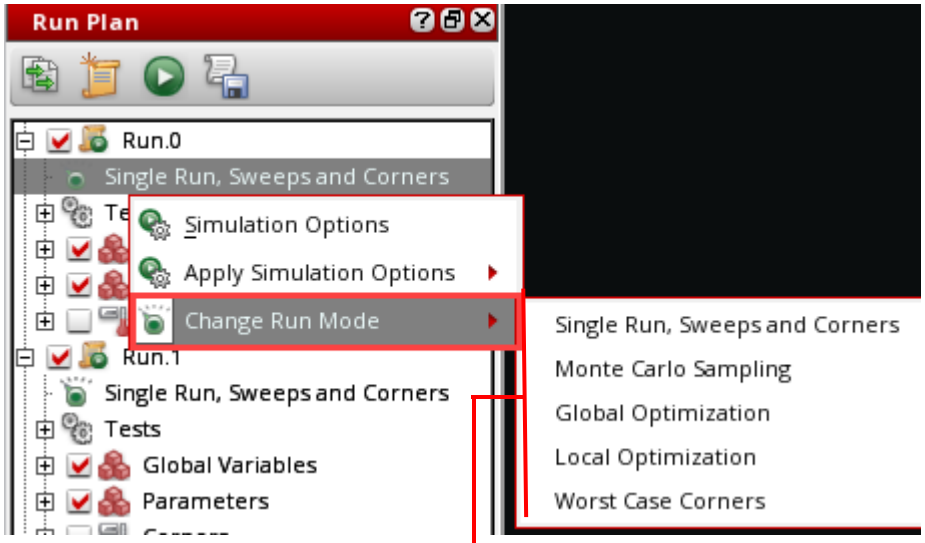
© 2023 Cadence Design Systems, Inc.

- Modifying Corners
- Modifying the Reliability Setup
- Copying Settings from One Run to Another in a Run Plan
- Changing the Display Order of Runs in a Run Plan

Changing the Run Mode

Right-click the run mode and select *Choose Run Mode – <name of another run mode>*. If required, you can select multiple runs and edit their run mode together.

Note: A run plan supports only five run modes, as shown in the figure below.



Run modes supported for Run Plans

Modifying the Simulation Options

You can modify the simulation options for a run in two ways:

- By editing the run options
- By copying the setup from another run in the run plan

© 2023 Cadence Design Systems, Inc.

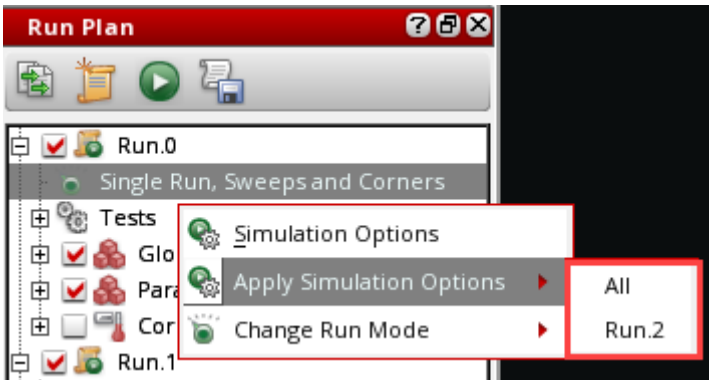
Editing the Simulation Options

To modify the simulation options for the selected run mode for a run, right-click the run mode and choose *Simulation options*.

The run options form for the selected run mode is displayed. Make the required changes and click *OK*.

Copying Simulation Options from Another Run

You can copy the simulation options from a run to other runs with the same run mode. To copy the options, right-click the run mode and choose *Apply Simulation options*, and then either choose *All* to copy options to all other runs that use the same run mode, or a specific run.



Note: Names of only those runs appear for which the run mode is same as the run mode of the selected run.

Modifying the Values of Global Variables and Parameters

The values of the global variables and parameters are copied from the active setup. After copying these elements to the run plan, you can change these values in two ways:

- **By editing the values in the run plan:** To change a value, double-click it. The value is shown in a text box where you can modify it. The updated value is saved when you click anywhere out of the text box. For visual indication, the edited values are displayed in **bold** text format.

  When setting up global variables for a run, you can create parametric sets independent of those in the active setup. To create parametric sets, select two or more global variables, right-click and choose *Group as Parametric Set*.

© 2023 Cadence Design Systems, Inc.

If parametric sets for global variables were copied from the active setup, you can modify or remove those parametric sets by using the following commands in the context-sensitive menu:

- *Add to Parametric Set*: Adds a variable to the selected parametric set.
- *Ungroup Parametric Set*: Deletes the parametric set. After this, the global variables it contained are not a part of any parametric set.
- *Remove from Parametric Set*: Removes a variable from the parametric set without deleting the set.

- **By copying edited values from the Data View assistant to the run plan:** By default, this feature is disabled. To enable copying of edited values of global variables, design variables, parameters, or corners from the Data View assistant to one or all runs in the run plan, set the `copytoRunPlanFromActiveSetup` environment variable to `t`. This command enables the *Copy to Run Plan* command in the shortcut menu of the selected element to copy the selected objects from the active setup to one or all runs in the run plan.

  Note: The variables and parameters can have different values in the active and run setups. Therefore, any change in the value of a variable or parameter in the active setup is not reflected in the runs unless copied explicitly after setting the `copytoRunPlanFromActiveSetup` environment variable.

Related Topics

Rules to Use Local or Global Variables for Run Plan Simulations

copytoRunPlanFromActiveSetup

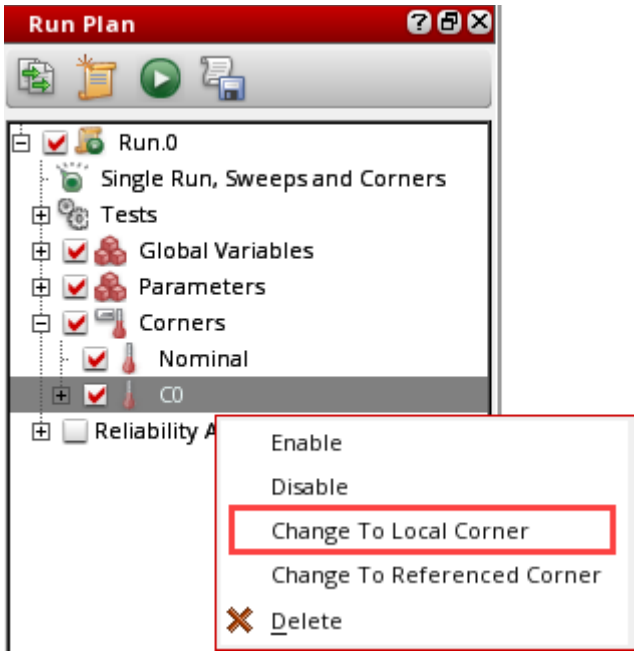
Modifying Corners

All corners in a run are referenced from the source corners in the active setup, and therefore, are synchronized to reflect any change in the active setup. By default, you cannot modify these corners. However, you can disable the synchronization for any corner so that its value is not changed to reflect any change in its reference in the active setup. This also allows you to modify its values.

To disable synchronization for a corner:

© 2023 Cadence Design Systems, Inc.

➡ Right-click the corner in the run setup and choose *Change to Local Corner*.



The corners local to a run appear in blue.

In the example shown above, corner `C0` is now a local corner for `Run.0`. If you change the value of `C0` in the Data View assistant, values of `C0` for `Run.0` will not change.

You can edit the definition of a local corner by modifying its variables, parameters, or model files.

To edit a corner definition for a run:

1. Expand the corner to view its details.

   The variables, parameters, or model file values for that corner are displayed. These values are editable for local corners.

2. Double-click the value field, and add or modify values.

   Note: For corners with model files, you cannot modify the path of a model file, but you can modify the section list.

3. Press `Enter`.

   The modified values are saved in the run plan. These changes are not reflected in the corners setup in the Data View assistant.

© 2023 Cadence Design Systems, Inc.

At any point, you can use the *Change to Referenced Corner* command in the context-sensitive menu of a corner in run plan to revert its status of local corner. When you use this command, the local values of the corner are lost and the values from the active setup are copied to it. However, a local corner is not converted to a referenced corner if the active setup does not contain any corner with that name.

Note: If a corner is desynchronized from the corner in the active setup, you can still copy updates from that corner to the one in the run plan. Set the `copytoRunPlanFromActiveSetup` environment variable to `t` to enable the *Copy to Run Plan* command in the shortcut menu of the selected corner and choose the corner to which you want to copy the corner details.

Modifying the Reliability Setup

If you create a run by copying the active setup, the reliability setup is copied to the run too. You can edit the setup and make any changes, create its copy, or delete it as required.

If your run does not have a reliability setup, you can run reliability analysis in two ways:

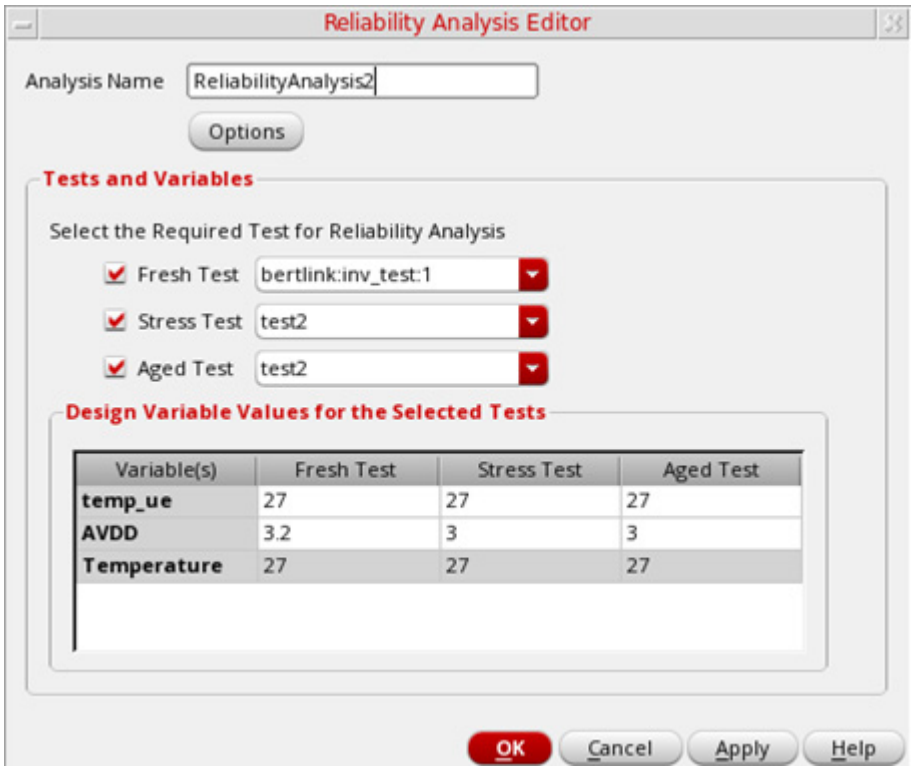
- By selectively dragging one or more reliability analyses to the run for which you want to run reliability analysis. When a reliability analysis is copied from the active setup to a run, you can use the context-sensitive menu to edit it, create its copy, or to delete it.
- By using the reliability analysis from the active setup. If the reliability analysis is enabled in the Data View assistant, but the run plan does not have reliability setup, ADE Assembler prompts you to confirm whether to include the reliability setup from the Data View assistant. By clicking *Yes* in the confirmation box, you can run reliability analysis with run plan. The only requirement is to ensure that the tests used for fresh, stress, and aged simulations are enabled for the runs in the run plan. For details about the reliability analysis, refer to Performing Reliability Analysis.

  Important Points to Note

  - You cannot rename the reliability analysis in the run plan. If you open the setup of an analysis and change its name, a copy of that analysis is created.
  - When a run does not contain any test and you copy the reliability setup from the active setup to this run, all tests available in the active setup can be used in the reliability analysis. For example, the active setup contains two tests,

© 2023 Cadence Design Systems, Inc.

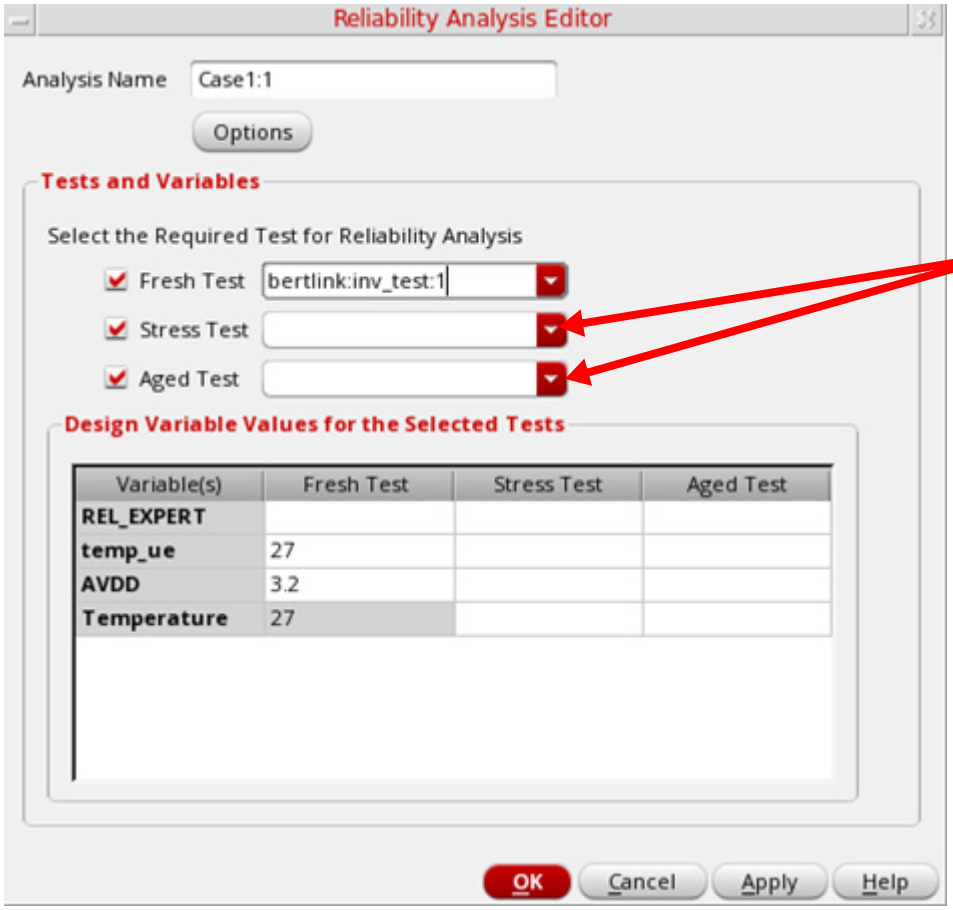
`betlink:inv_test:1` and `test2`. If your run does not contain any test, the reliability analysis can use both tests in the setup, as shown below.



- If the reliability analysis copied from the active setup to a run in the run plan refers to a test that is not available in the run, the unavailable test is removed from the

© 2023 Cadence Design Systems, Inc.

reliability analysis of the run. In that case, you need to modify the analysis to change the tests being used as fresh, stress, or aged tests.



Test names removed from the setup because they are not available in the run.

© 2023 Cadence Design Systems, Inc.

Copying Settings from One Run to Another in a Run Plan

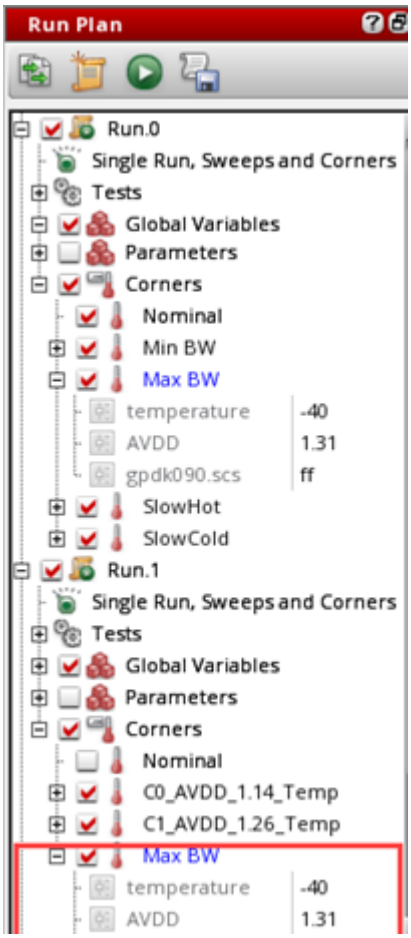
If you have modified tests, variables, parameters, or corners in a run, you can copy those to another run in the run plan.

To copy the settings from one run to another:

1. Select the settings to be copied from the source run in the run plan.
2. Drag and drop the selected settings anywhere on the other run.

   The tool adds the tests, variables, parameters, or corners in their respective placeholders in the run to which they are copied. The newly copied details are added after any existing items already defined for the run. If an item with the same name already exists, it is replaced by the copied item.

In the example shown below, the corner `Max BW` is copied from `Run.0` to `Run.1`. The blue color of this corner indicates that it is overridden as a local corner in `Run.0`. This setting is also copied in addition to its value.



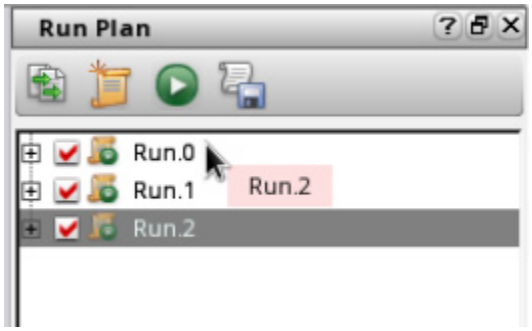
© 2023 Cadence Design Systems, Inc.

Changing the Display Order of Runs in a Run Plan

You can change the placement of runs in the run plan to keep the important runs on top or to keep related runs together. This helps in displaying the runs in an organized way in the Run Plan assistant.

To move a particular run up or down, drag and drop it on the run before which you need to place it.

In the example shown below, if you want to run simulations for `Run.2` before `Run.0` and `Run.1`, move `Run.2` to the top by dragging it on `Run.0`.



© 2023 Cadence Design Systems, Inc.

Enabling or Disabling Runs in a Run Plan

A run plan setup can contain multiple runs. To selectively use a subset of runs, you can select the check box corresponding to the run. Alternatively, press the Ctrl key and select the runs you want to include in the simulation and choose *Enable* from the short-cut menu.

Similarly, to disable multiple runs, select the runs you want to exclude from the simulation and choose *Disable* from the short-cut menu. The check box corresponding to each selected run is cleared.

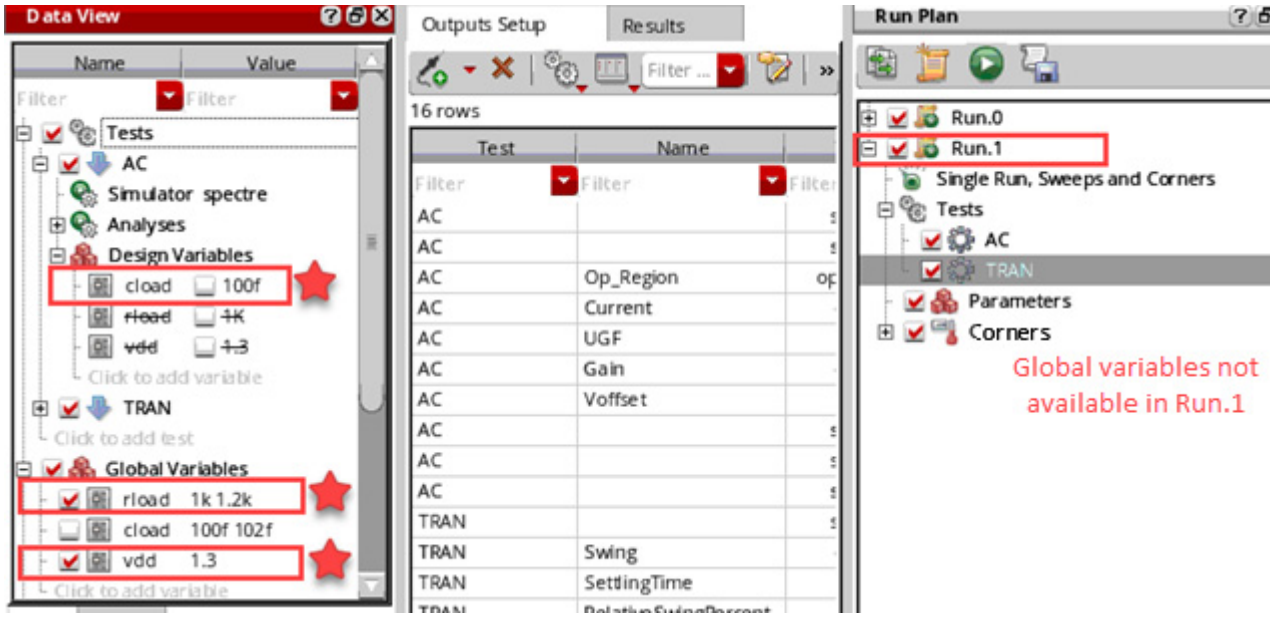
© 2023 Cadence Design Systems, Inc.

Rules to Use Local or Global Variables for Run Plan Simulations

It is important to understand how a run uses the global and local variables, depending on the enabled or disabled status in the run and the active setups. This is described below:

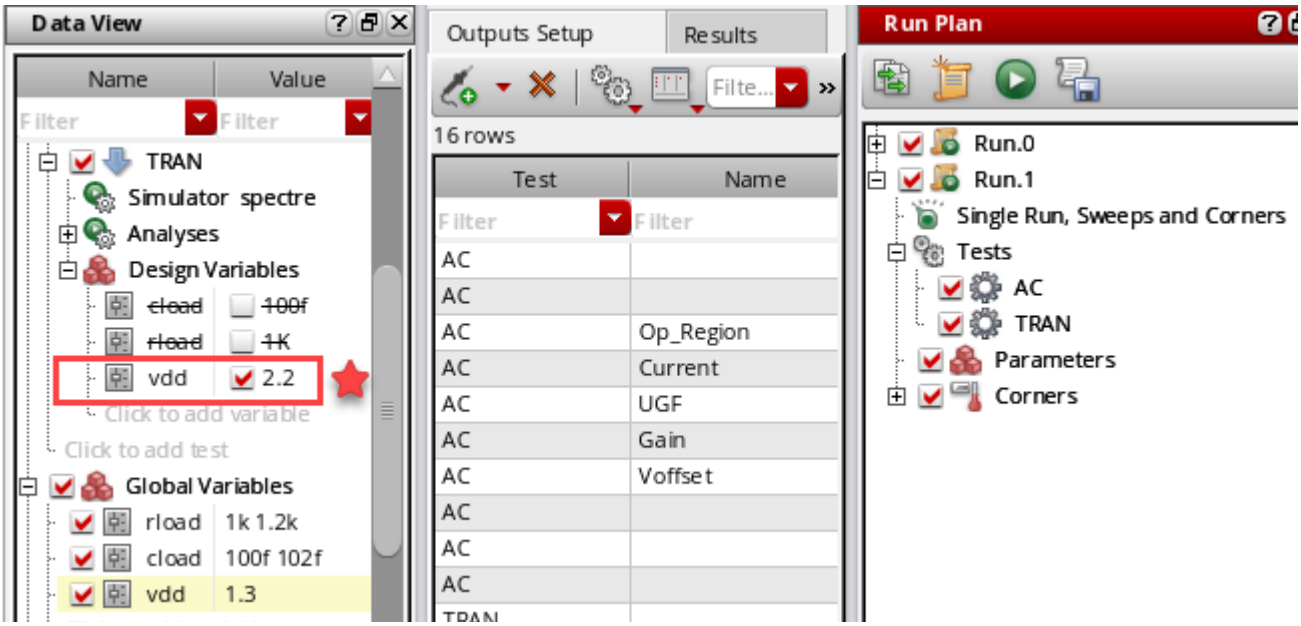
- If a global variable is not available in the run, the global variable from the active setup is used, if enabled and not overridden in the test. If the global variable is disabled in the active setup, the local test variable is used.

  In the example shown below, global variables have not been added to `Run.1`. Therefore, it will use the values of `rload` and `vdd` from the Global Variables in the active setup. For `cload`, it will use the values from test.



© 2023 Cadence Design Systems, Inc.

If the global variable in the active setup is enabled, but overridden in the test, value from the local variable is used for that test, as shown below.



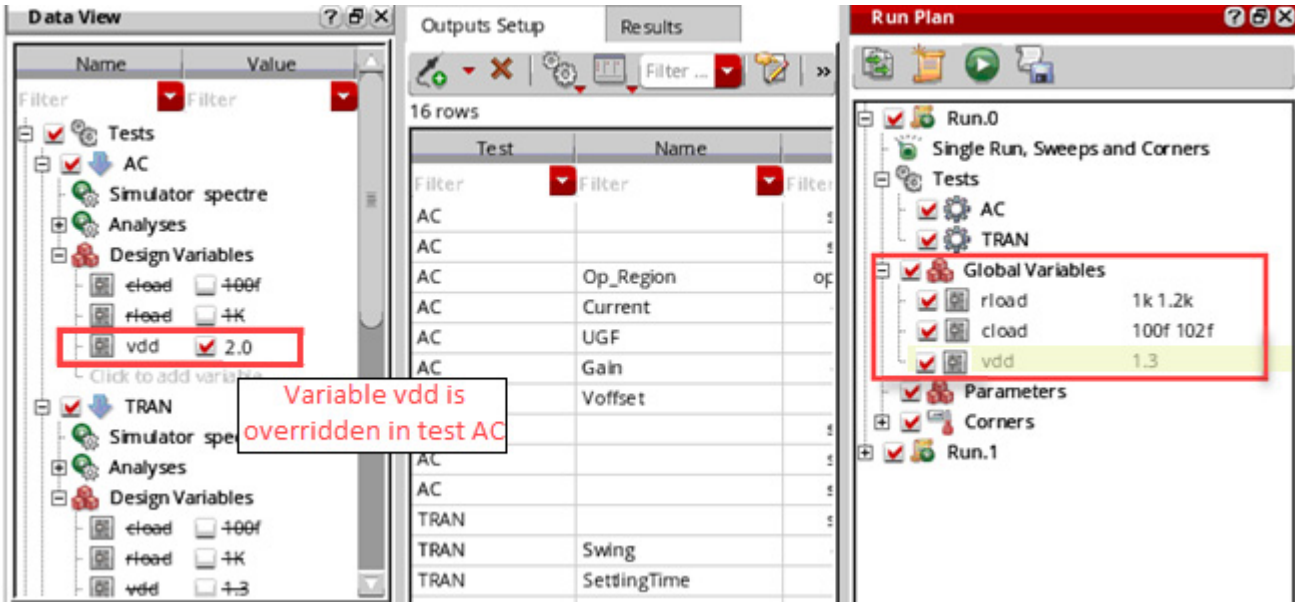
Here, `vdd` is overridden in test `TRAN`. Therefore, for `TRAN` value `2.2` is used, whereas for other tests that do not override this variable, use `vdd=1.3`.

- If global variables are available and enabled in the run:
  - Global variables from the run are used if the local test variables are not overridden.
  - Local test variables are used from the active setup if overridden in the test.

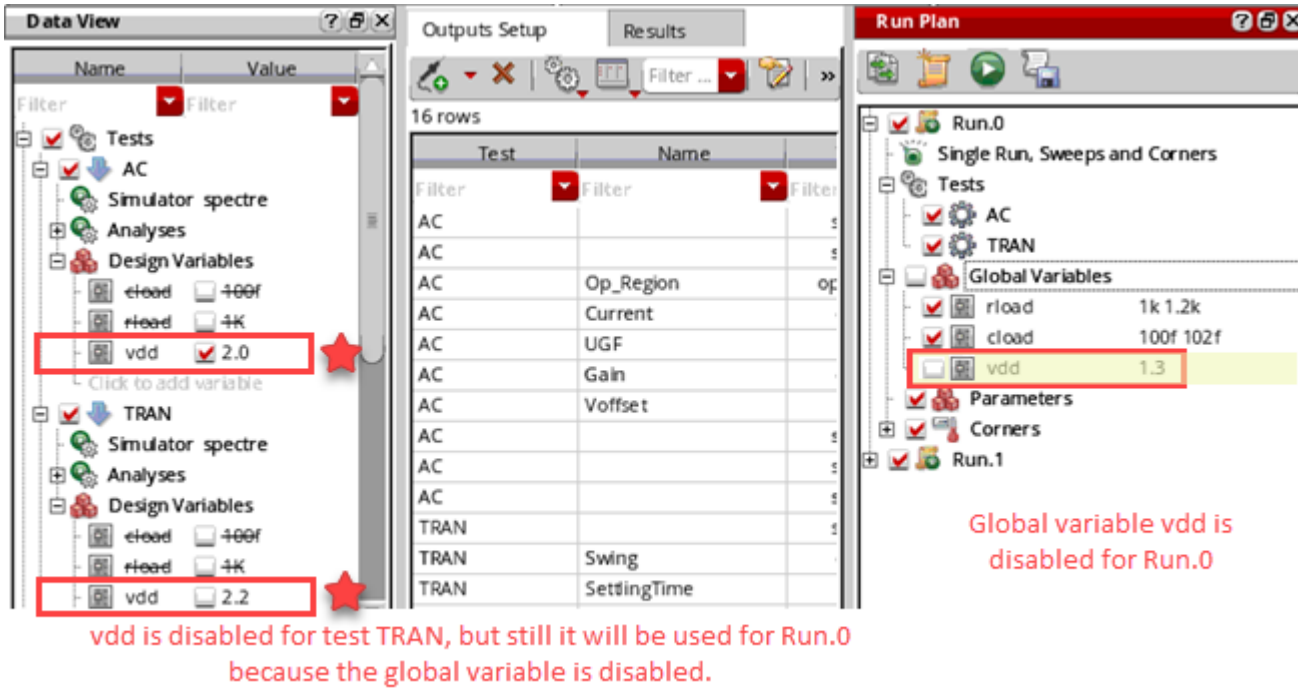
  The example given below illustrates this scenario. Variable `vdd` is overridden in test `AC`. Therefore, test `AC` will use `rload` and `cload` from Global Variables in `Run.0`. It will use the value of `vdd` as `2.0` from the local variable.

© 2023 Cadence Design Systems, Inc.

Test `TRAN` will use the values for all the variables from the Global Variables section in `Run.0`.



- If global variables are available, but disabled in the run, local test variables from the active setup are used, irrespective of their enabled status. In the example shown below, for run `Run.0`, test `AC` will use `vdd=2.0`, and test `TRAN` will use `vdd=2.2`.



© 2023 Cadence Design Systems, Inc.

Important Points to Note

- The rules described above might not work in the run plans created using IC6.1.7 ISR13 or ICADV12.3 ISR13 or their earlier versions. It is recommended to reconfigure those run plans using the later versions of ADE Assembler.
- Run Plan simulations support the config sweep values specified for the global variables in the active setup or for the global variables in a run defined in the Run Plan assistant.

© 2023 Cadence Design Systems, Inc.

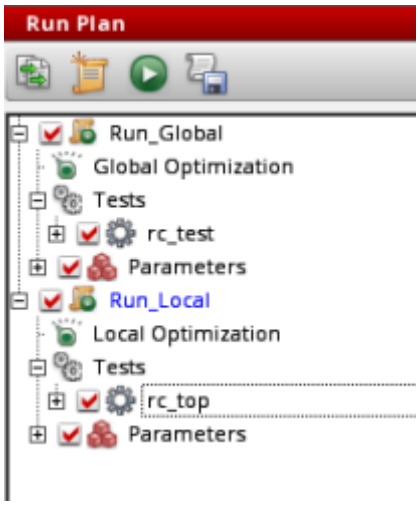
Using Pre- and Post-Run Scripts for Optimization Runs

When you have set up a local optimization run in a run plan, you need to specify a starting state that ADE Assembler uses to identify the first point for the simulation run.

If the starting state is known, you can specify that in the run options form for that run. However, if you want to dynamically use a setup state saved by another run in the run plan, you can use the following predefined SKILL functions:

- maeSaveSetupStateForOptimization
- maeSetStartingPointForOptimization

Consider an example run plan with two runs.



In this example, the run `Run_Global` is set up to run global optimization. To use the best design point returned by `Run_Global` in the next run, `Run_Local`, that runs local optimization, do this:

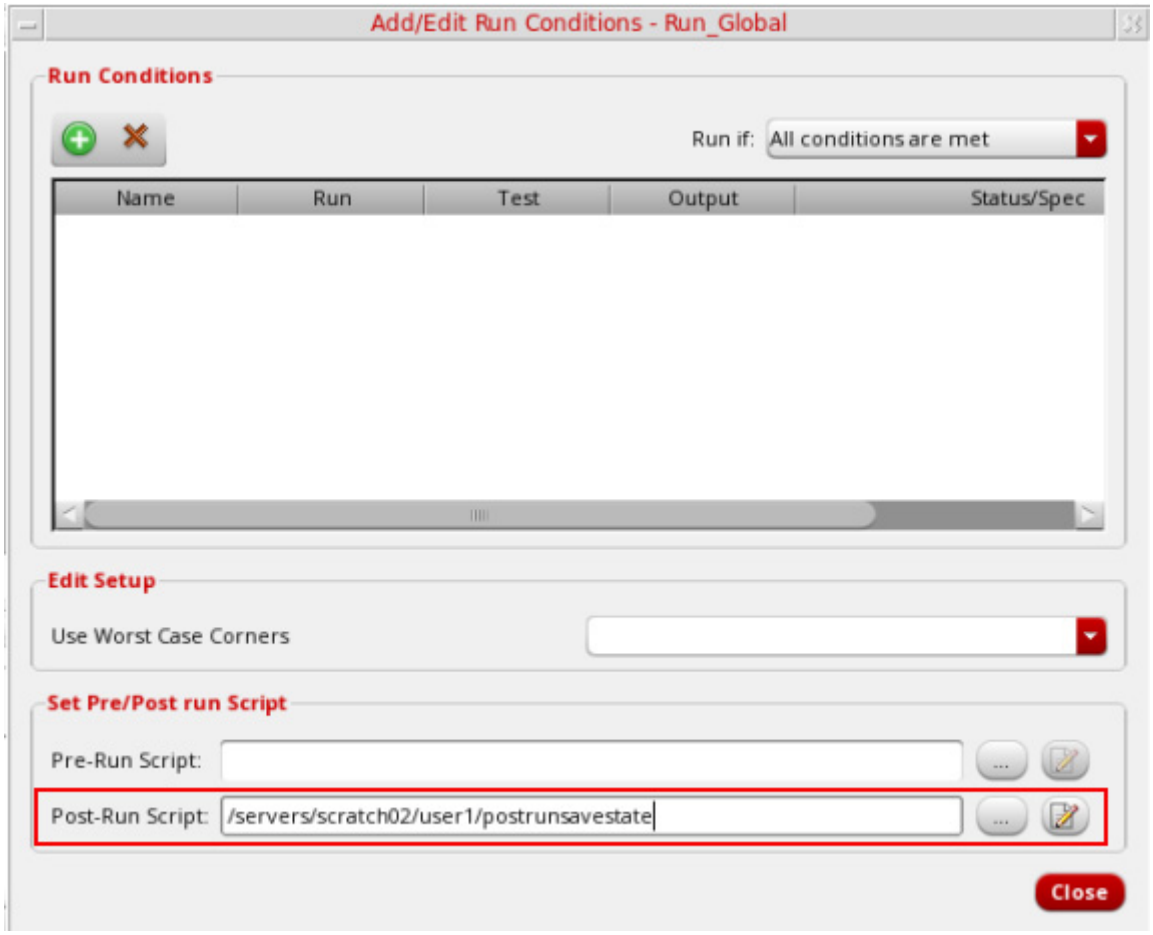
1. Create a post-run script with the `maeSaveSetupStateForOptimization` SKILL function to save a setup state for optimization from the best design point of the run `Run_Global`.

   For example:

   ```
   maeSaveSetupStateForOptimization(?run "Run_Global" ?setupState "SetupState1")
   ```


© 2023 Cadence Design Systems, Inc.

2. In the Add/Edit the Run Conditions form for `Run_Global`, specify the path to the post-run script.



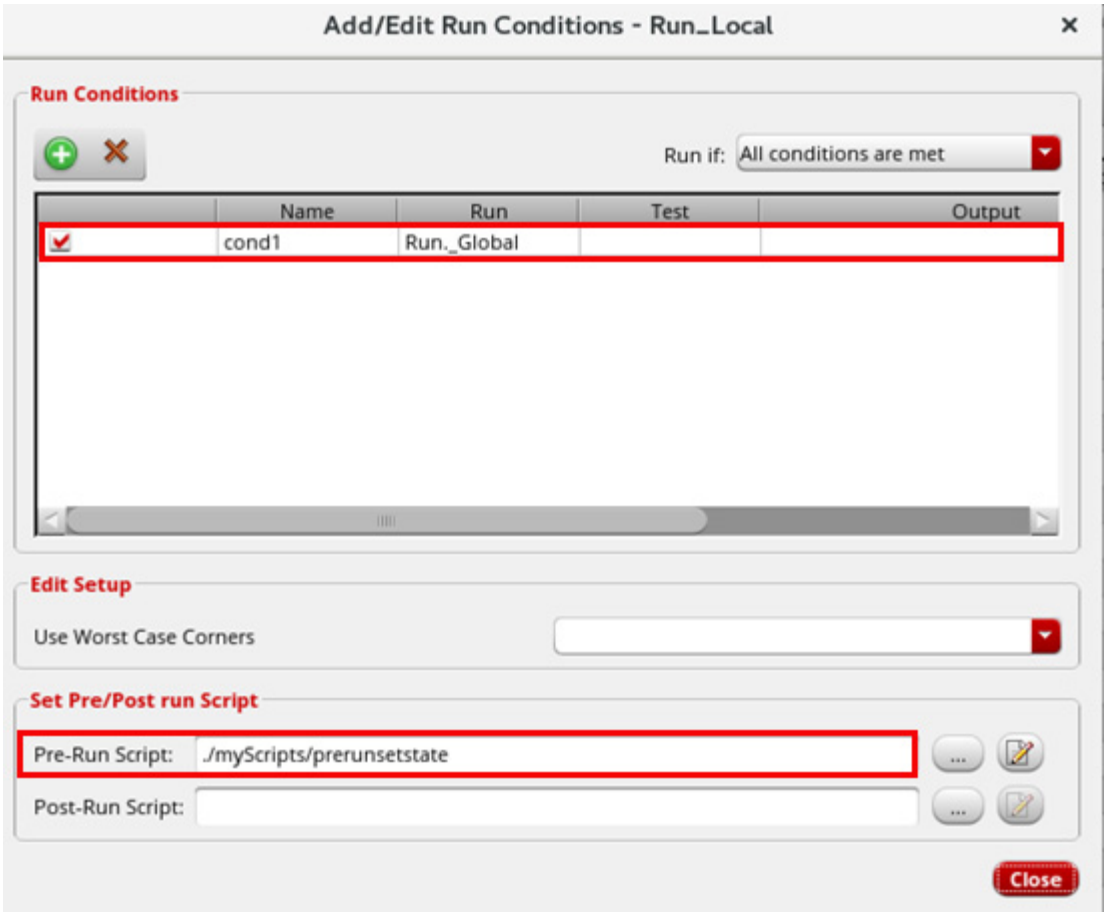
3. Create a pre-run script with the `maeSetStartingPointForOptimization` SKILL function to set the given setup state as the starting point for the run `Run_Local`.

   For example:

```
maeSetStartingPointForOptimization(?run "Run_Local" ?setupState
"SetupState1")
```


© 2023 Cadence Design Systems, Inc.

4. In the Add/Edit the Run Conditions form for `Run_Local`, specify the path to the pre-run script and add a run condition to begin this run after the successful completion of `Run_Global`.



You can create multiple conditions for a run. Depending on your run requirements, you can selectively disable conditions that you do not want to use in a particular simulation run by clicking the check box to the left of a condition name.

To enable or disable multiple conditions, select the rows of those conditions and choose the *Enable* or *Disable* command, respectively, from the context-sensitive menu.

5. Close the form and run the plan.

© 2023 Cadence Design Systems, Inc.

Simulation is started for the runs that do not have any precondition. `Run_Local` waits for the successful completion of `Run_Global`. After `Run_Global` have is run successfully, the post-run script is run to save the starting state using the best design point from its results. That state is then set by the pre-run script for `Run_Local`. The saved state is also added the *Setup States* tree in the Data View assistant.

© 2023 Cadence Design Systems, Inc.

Executing the Run Plan

A run plan can contain a single or multiple runs. Depending on the requirements, you can execute a single run individually or all the runs together.

The following topics provide more details on how you can execute the run plans:

- Executing all the Runs Together
- Executing a Selected Run
- Executing a Run Plan Based on Conditions
- Executing Runs Using a Script

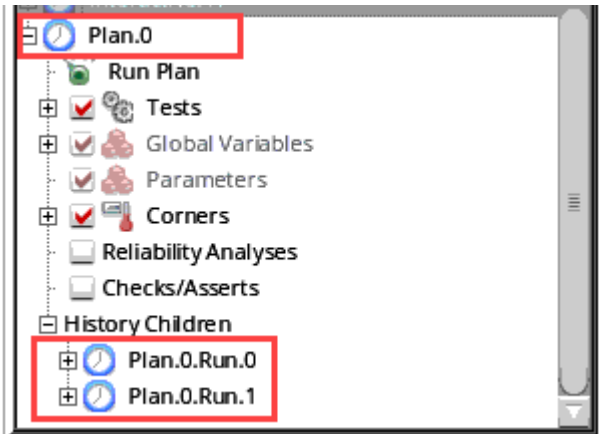
Executing all the Runs Together

To execute all the runs defined in the Run Plan assistant, do one of the following:

- Click *Execute Plan* ( ) on the toolbar of the Run Plan assistant.
- Choose `Run Plan` in the *Select a run mode* drop-down list on the Run toolbar and then click *Run Simulation*.

Simulations for all the runs defined in the Run Plan assistant will be run. A checkpoint will be saved for each run individually.

A simulation history is saved with the name *Plan.n*, where *n* is the incremental number for a history created for run plans. This history checkpoint contains the summary log and the progress report of the run plan. It also contains child histories for each run in the run plan. Each child history is named as *<run-name>.x*, where *<run-name>* is the name of the run in the run plan and *x* is the incremental number for the histories saved for that run.



© 2023 Cadence Design Systems, Inc.

If a run contains only disabled tests, simulations are not run for those tests. Their progress is also not shown on the *Results* tab. Instead, an error message is displayed.

Related Topic: Viewing Results of Run Plan Simulations

Executing a Selected Run

To execute a specific run, right-click that run in the Run Plan assistant and choose *Run*.

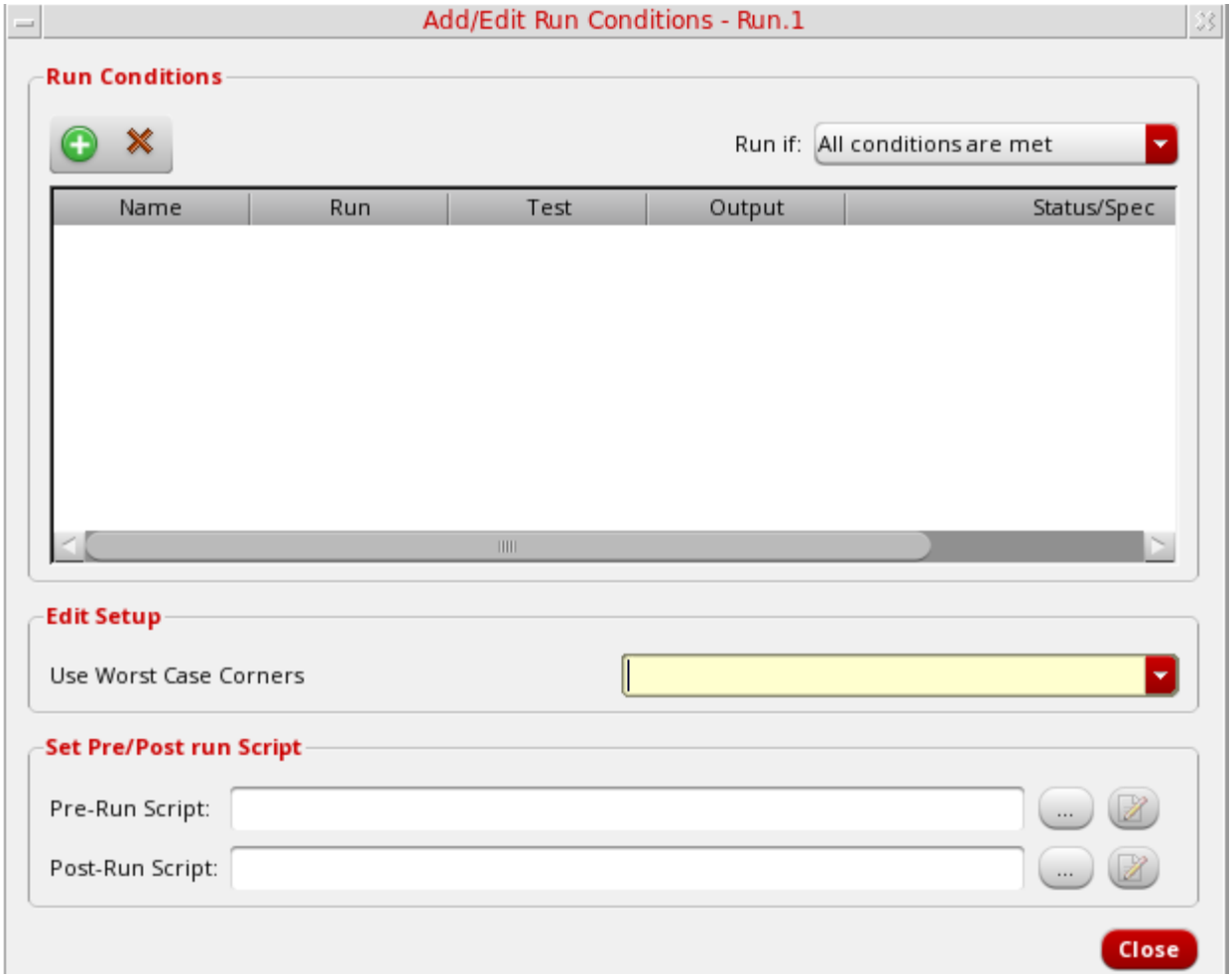
For the selected run, a history named *<run-name>.n* is created, where *n* is the incremental number used for the history of that run.

Executing a Run Plan Based on Conditions

ADE Assembler provides the flexibility to execute runs in a run plan based on certain conditions. Using this feature, you can specify one or more conditions for a run and set an execution criteria for it depending on the pass or fail status of the condition.

© 2023 Cadence Design Systems, Inc.

You can create a condition for a run by using the Add/Edit Run Conditions form for that run. Right-click a run in the Run Plan assistant and choose *Edit Run Conditions* to view this form, as shown below.

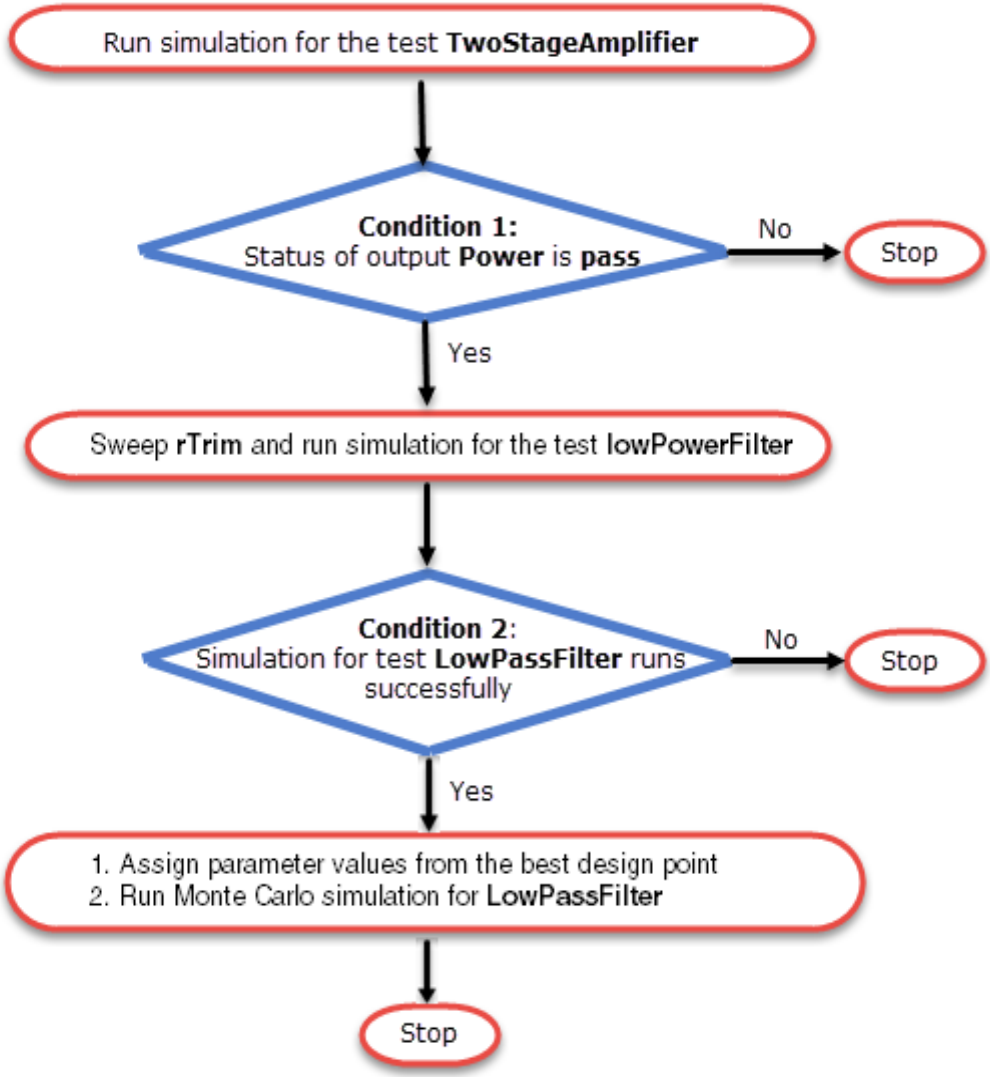


Creating a Run Condition

By default, no conditions are defined for a run. If a run plan contains multiple runs, you can create different conditions for each run.

© 2023 Cadence Design Systems, Inc.

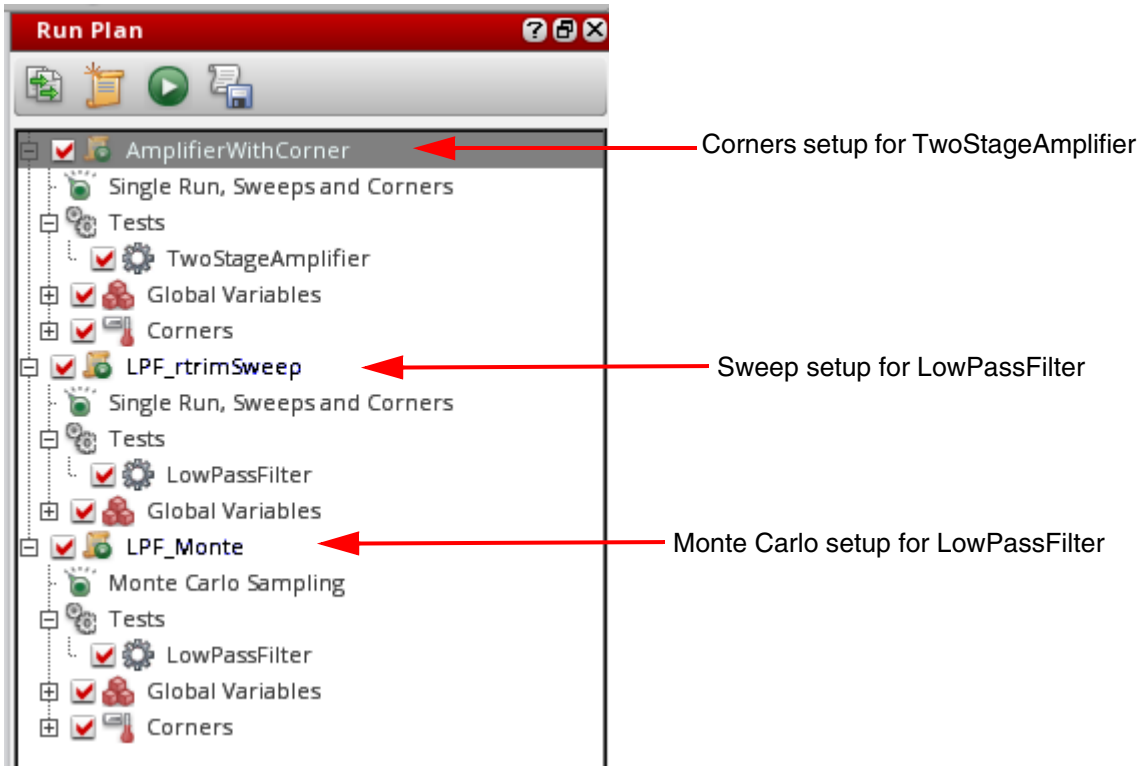
Consider an example of a testcase that contains two tests, `TwoStageAmplifier` and `LowPassFilter`. You need to create the following run plan:



To implement such a run plan, perform the following steps:

© 2023 Cadence Design Systems, Inc.

1. Create multiple runs in the run plan assistant, as shown below.



   For details on how to create a new run plan, refer to Creating a New Run.

2. To create a condition for the second run, `LPF_rtrimSweep`, right-click the run name and choose *Edit Run Condition*.

3. In the *Add/Edit Run Conditions* form, click *Create new condition run* ( ).

   A new row with the default name `cond1` is created in the conditions display area.

4. Specify a condition that can be based on the status of an output of a particular test in a previous run. An example is shown below.

© 2023 Cadence Design Systems, Inc.

The condition shown above indicates that the `LPF_rtrimSweep` run will be executed only if the status of output `Power` for test `TwoStageAmplifier` is `pass`.

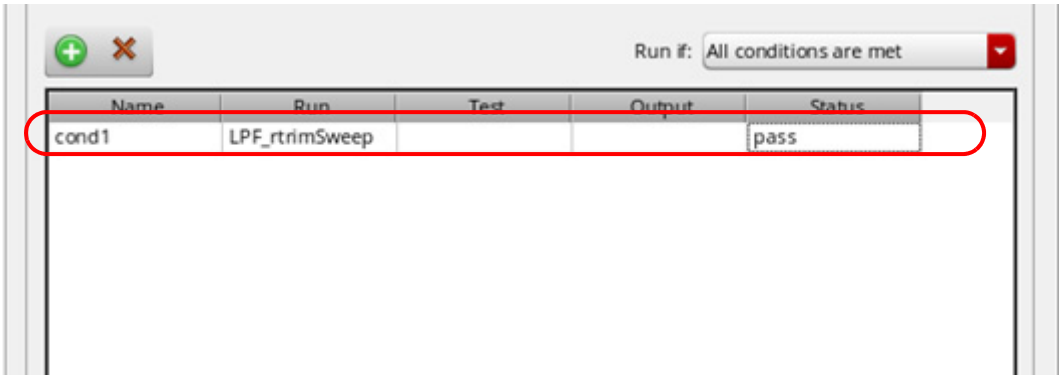
You can check for any of the following status values for an output:

- `pass`: Indicates that the output value met the spec condition
- `fail`: Indicates that the output value did not meet the spec conditional simulation error or a netlist error
- `error`: Indicates an evaluation error in the spec expression

Important Points to Note

- It is recommended not to create a run condition that uses a run with the `Worst Case Corners` run mode. This is because that run creates a grouped history and the run condition cannot check the status of its results.
- When you choose `All` in the *Test* column of a condition, an additional value, `Yield`, is also available in the *Output* drop-down list. This can be used to create a condition based on the yield of the previous run. The status for yield can be checked using the `<`, `>`, `tol`, or `range` operators. Select either of these from the drop-down list and specify a value.
- The status values are same as those appear on the Results tab and can be obtained only if an output contains a specification. Therefore, only the outputs that have specifications are listed in the *Output* drop-down list.

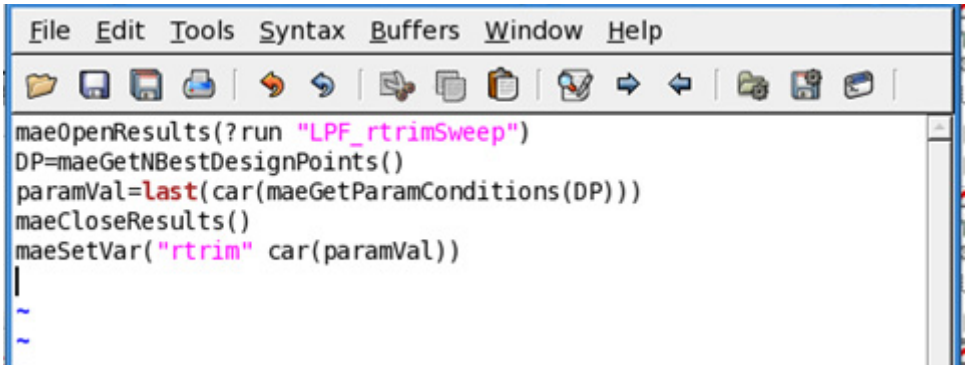
5. To create a condition for the second run, `LPF_Monte`, right-click the run name and choose *Edit Run Condition*.
6. On the *Add/Edit Run Conditions* form, click *Create new condition run* ( ).
7. Specify the run condition as shown below.



This condition indicates that the `LPF_Monte` run will be executed only if the simulation for test `LPF_rtrimSweep` is successful.

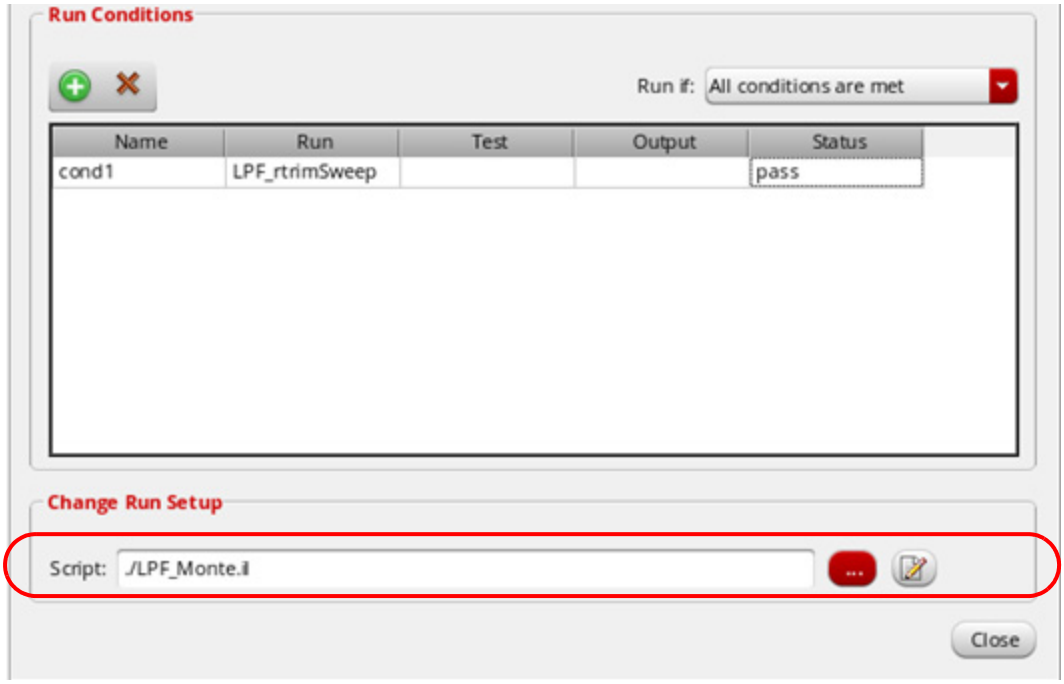
© 2023 Cadence Design Systems, Inc.

8. In a text editor, create a pre-run script for the `LPF_Monte` run to assign the value of a parameter condition from the previous run to variable `rtrim` in this run.



9. Save the script in the directory where the maestro cellview is saved.

10. On the *Add/Edit Run Conditions* form, click the browse button next to the *Script* field and select the path to the saved script.



Important

If your pre-run script is dependent on the result of another run, add a run condition to complete the reference run before running the dependent run.

11. Click *Close* to close the form.

© 2023 Cadence Design Systems, Inc.

The run conditions are now set for two runs. The color of a dependent run is changed to blue.

Next, click *Execute Plan* ( ) on the toolbar of the Run Plan assistant to execute the complete run plan. All run conditions are considered while running the plan.

The progress of a run plan is displayed on the Results tab.

Tip

Another way to create dependency and use the result value from one run in another run, you can use the `calcVal` SKILL function. For more details, see Using Results of One Run in Another.

© 2023 Cadence Design Systems, Inc.

Executing Runs Using a Script

You can save the run plans in a script and execute them from the command prompt.

Saving the Run Plan in a Script

To save the run plan in a script, perform the following steps:

1. Click *Save Script to Run Plan in Batch* ( ) on the toolbar of the Run Plan assistant.

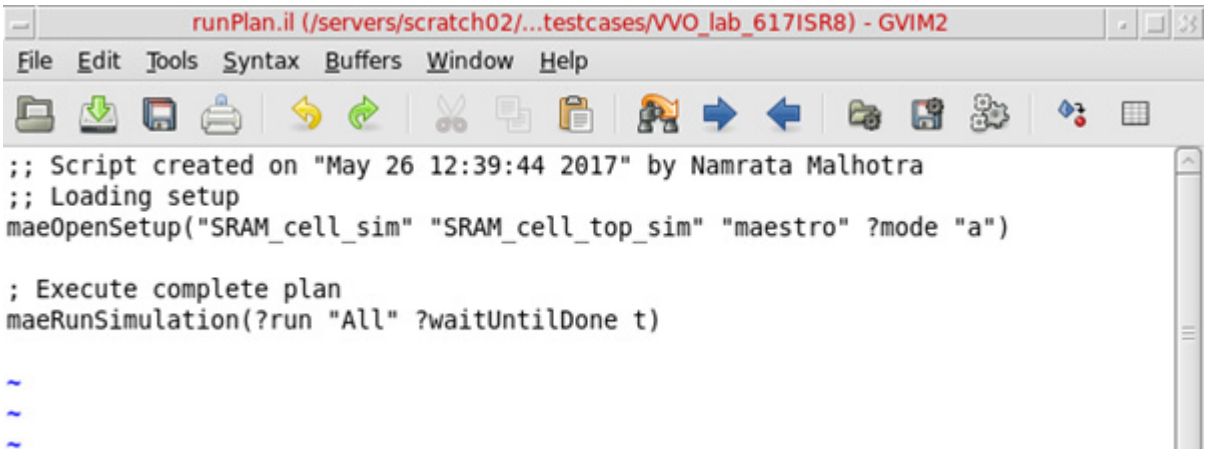
   The Save Run Plan Script form is displayed.

2. Specify a file name for the script. If required, you can also change the path where the file is saved.

   By default, the script is saved as `runPlan.il` in the current working directory.

3. Click *OK*.

   ADE Assembler creates a script that can be used to run the plan from command prompt. The script is saved with the specified file and displayed in a text editor.



Related environment variables:

- accessMode
- exitAfterFinished

© 2023 Cadence Design Systems, Inc.

- writeCallbackForSingleRun
- runPlan

Executing the Run Plan Script

To execute the script, run the following command:

```
virtuoso -replay <path-to-the-script-file> -log <path-to-the-log-file> -report
```

The runs defined in the script file are executed and a log file is created.

The history checkpoints are saved in the maestro cellview and can be viewed from the ADE Assembler environment. For more details, refer to Working with History Checkpoints.

© 2023 Cadence Design Systems, Inc.

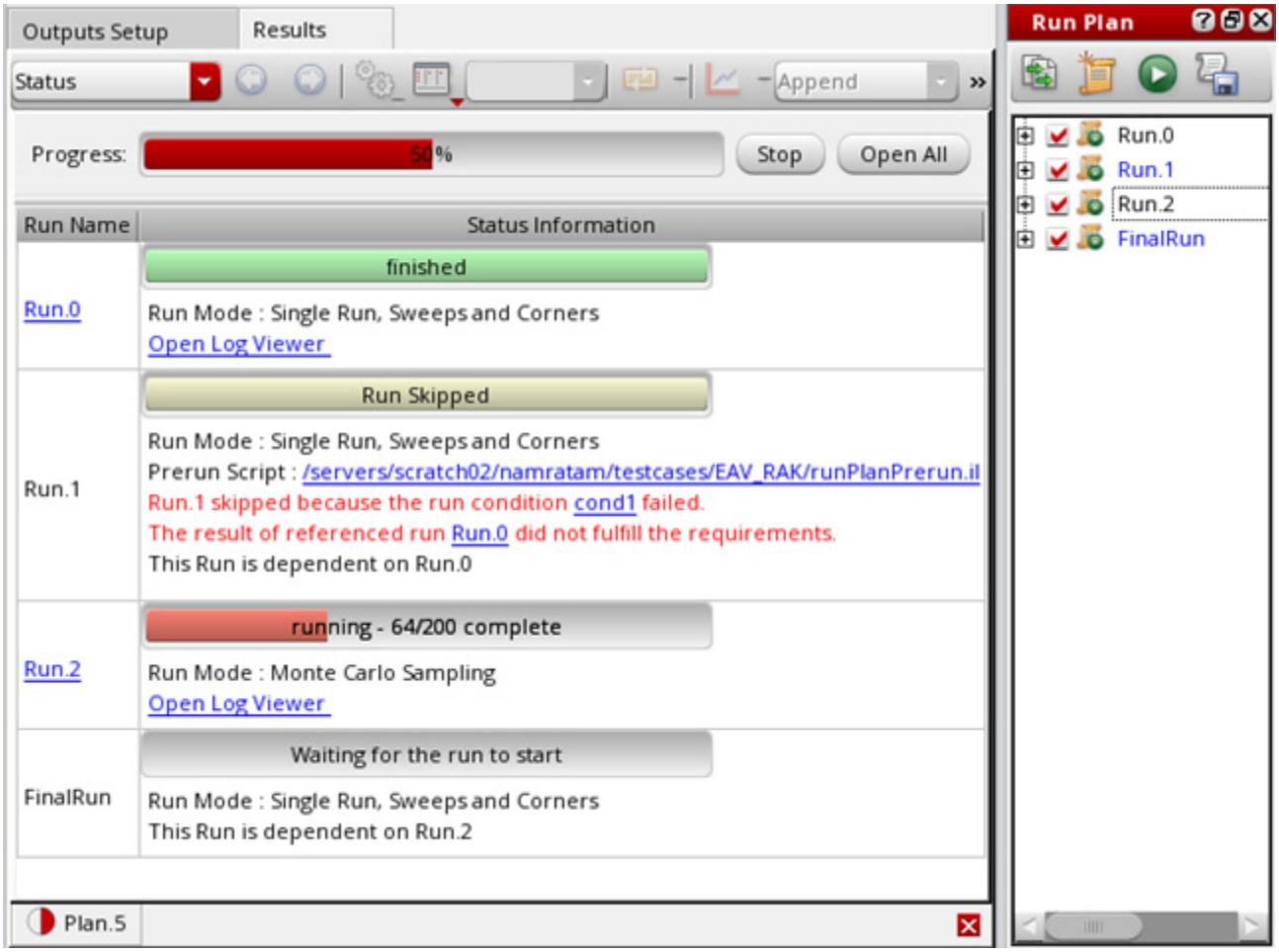
Viewing Results of Run Plan Simulations

After a run plan simulation is complete, the results are displayed in the *Status* results view.

To open the results of an existing run plan history:

➡ In the *History* tab, right-click a history name and choose *View Results*.

An example of the *Status* results view is shown below.



This view displays:

- The current state of each run in the run plan. A run can exist in one of the following states:
  - Running: When the netlist creation or simulation is in progress
  - Finished: When the simulation for the run is successfully complete

© 2023 Cadence Design Systems, Inc.

  - Run Skipped: When the simulation cannot be run because a pre-run condition is not met
  - Waiting for the run to start: When a run is waiting for another run to complete successfully before it can start
- An *Open Log Viewer* link to open the run log saved for the runs that completed successfully.
- Links to the conditions that open the Add/Edit Run Conditions form.
- Links to the pre- or post-run scripts that open the script files in the default text editor.
- Links to open the results table, for example, `Run.0`, for the runs that completed successfully.
- The *Stop* command to stop the simulation for all runs in the current run plan. This command is different from the *Stop Simulation* command on the Run toolbar that stops all simulations in the current ADE Assembler session.
- The *Open All* command to open the results for all the runs, which completed successfully in the current run plan.

Note: Run plan histories saved in the old format are displayed in the same format. They cannot be updated to the new format.

© 2023 Cadence Design Systems, Inc.

Rerun Simulations for Erroneous or Unfinished Points in a Run Plan

When the simulation for a run plan is complete, but certain points in its child histories show errors or remained unfinished either because the simulation was stopped or the resources were not sufficient, you can rerun the simulation for only the erroneous or incomplete points.

To rerun the erroneous or incomplete points in a run plan:

➡ On the *History* tab of the *Data View* assistant pane, right-click the history for the run plan and choose *Rerun Unfinished/Error Points*.

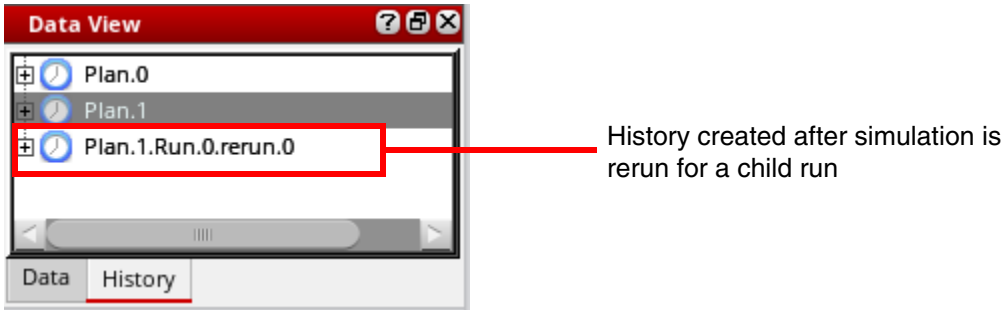
ADE Assembler scans the run plan history and resubmits simulations for child runs for which one of the following conditions is met:

- Results contain points with status as `canceled`
- Results contain points with errors

For each child run that meets these conditions, the tool creates a new history with name in the format *`runplan-history-name.child-run-name`*`.rerun.`*`incrementalnum`*.

This new history is created at the level of the parent history.

For example, the `Run.0` child history is rerun for a run plan history `Plan.1`, instead of saving the revised results in the same `Run.0` history.



You can later rename this history. The tool will rename the results database accordingly.

If a run plan history does not have any child runs with incomplete or erroneous points, the tool prints an information message in the CIW indicating the same.

Note: Child runs that were skipped in the initial run are not rerun by this command.

© 2023 Cadence Design Systems, Inc.

Using Results of One Run in Another

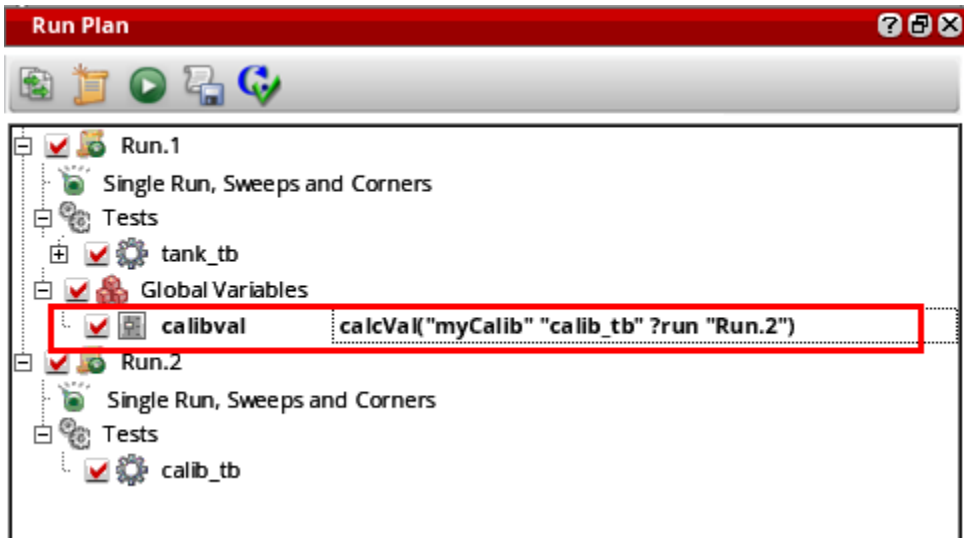
You can use the result values of one run in another of the same plan, that is, the same history in two ways:

- By creating a run condition. For more details, refer to Executing a Run Plan Based on Conditions
- By using the calcVal SKILL function. For more details, refer to Using calcVal in Run Plan.
- By using worst case corners created by one run to run simulations in another run. For more details, refer to Using Worst Case Corners Created by Another Run.

Using calcVal in Run Plan

The `calcVal` function provides the `?run` argument that can be used to refer to the values of a variable from the results of a run.

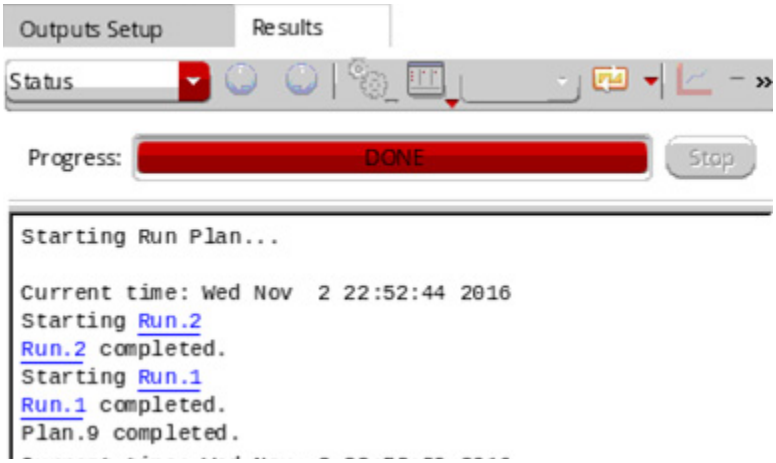
For example, in the run plan shown below, the value of the `calibVal` global variable in `Run.1` is calculated using the result value of the `myCalib` output of `calib_tb` test in `Run.2`.



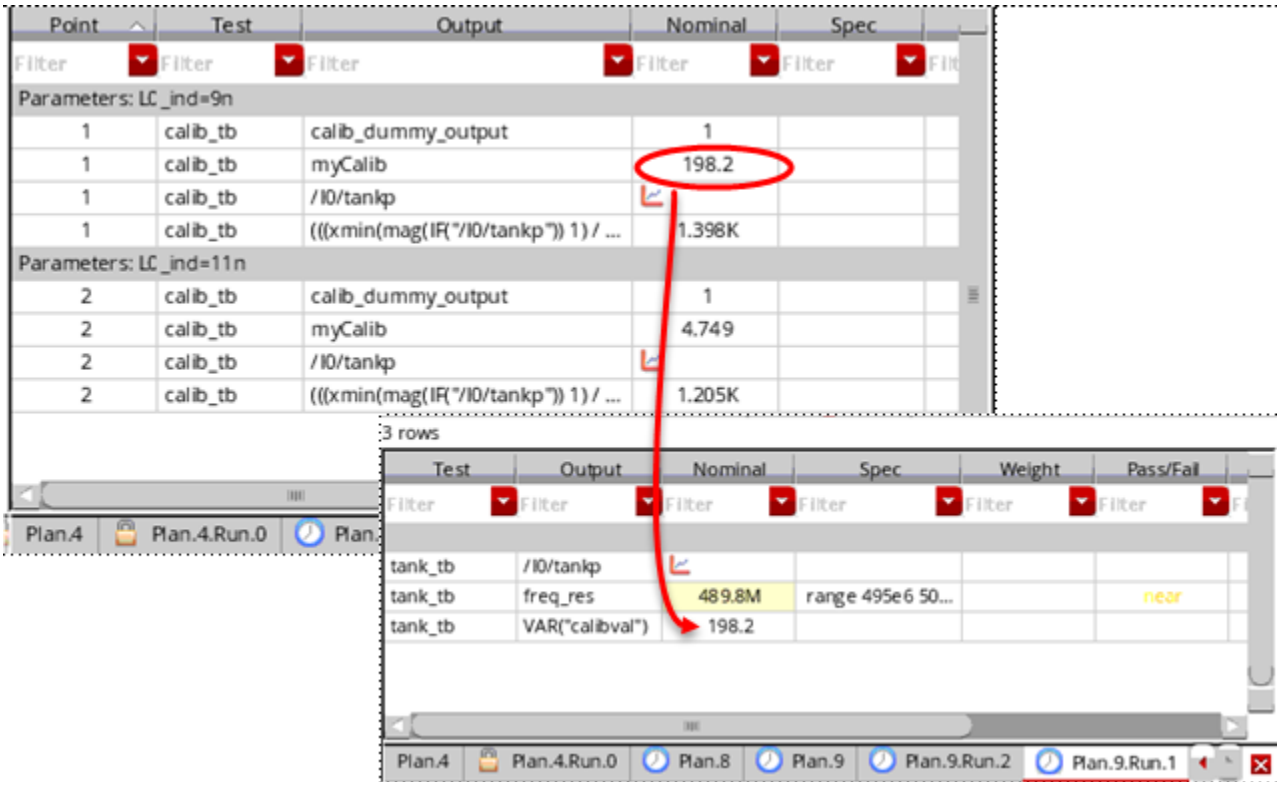
Whenever you use `calcVal` in a run to retrieve the results from another run, ADE Assembler automatically identifies the dependency and starts the simulation for the runs in a sequence—simulation for the dependent runs are started after the simulation for the referenced run is

© 2023 Cadence Design Systems, Inc.

complete. In the example shown above, ADE Assembler first runs simulation for `Run.2`, and then for `Run.1`.



If there are multiple points in the referenced run, the value of the variable from a point with matching variables and parameters is used in the dependent run, as shown in the figure below.



However, the reference result to be used depends on the value of the `?matchParams` argument in the `calcVal` expression.

© 2023 Cadence Design Systems, Inc.

You can use the *calcVal lint check* command on the toolbar to run pre-simulation checks for the `calcVal` expressions in all variables. You can view the resulting messages of the lint checks in the CIW.

The *calcVal lint check* command is supported only for Single Run, Sweeps, and Corners run mode and the `Standard` method for Monte Carlo run mode.

Important

If the dependency in the `calcVal` expression is incorrect, the run plan is not executed. For example, if the run used in the `calcVal` expression is not defined in the Run Plan assistant, or if the runs have cyclic dependency on each other, ADE Assembler flags errors and the simulation is not run.

Related Topics

calcVal

Using Worst Case Corners Created by Another Run

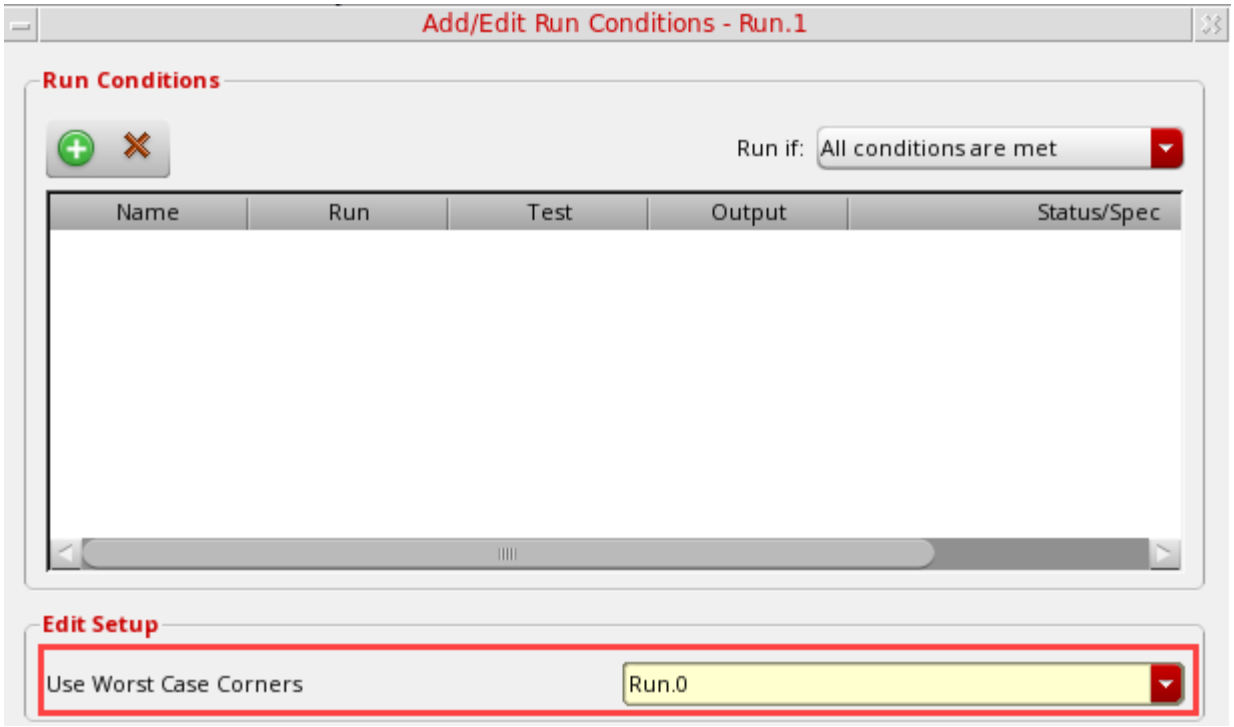
The simulations run with the `Worst Case Corners` run mode help in identifying corner conditions at which you want to measure the performance of your design. You can run a subsequent simulation with any other run mode by using the identified worst case corners.

To use the worst case corners identified by a run in another run, perform the following steps:

1. Create a run with the `Worst Case Corners` run mode. For more details on the setup, refer to Setting Up a Run Plan with the Worst Case Corners Run Mode.
2. Create another run with the desired setup.
3. Edit Setup of the second run to use the corners created by the run that ran the worst case corners simulations. For this, right-click on the second run and choose *Add/Edit Run Conditions* from the context-sensitive menu.

© 2023 Cadence Design Systems, Inc.

4. In the *Edit setup* section of the Add/Edit Run Conditions form, select the name of the run for which the worst case corners simulation is to be run.

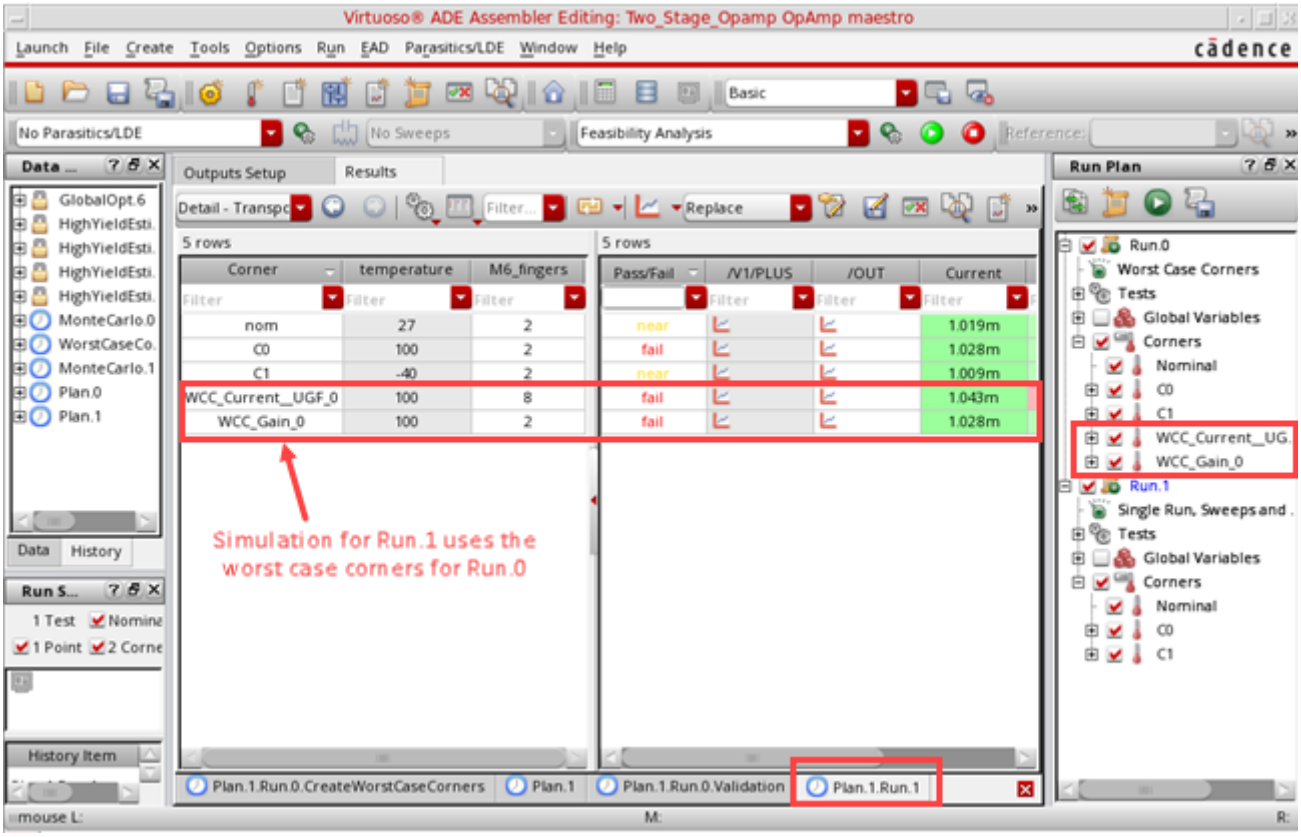


Settings in the example shown above indicate that `Run.1` uses the worst case corners created by `Run.0`. You can also add a condition to run `Run.1` only after successful completion of `Run.0`.

Note: If the dependent run, which is using the worst case corners created by another run, does not have any corners added to the run setup, only the worst case corners created by the previous run are added to it at run time. The dependent run does not use the other corners defined in active setup.

© 2023 Cadence Design Systems, Inc.

When the simulations for the run plan are complete, the results of all runs are visible in separate tabs on the Results tab in ADE Assembler, as shown in the example given below.



Important

The new worst case corners are added to the run setup as well as the active setup in ADE Assembler. However, in the active setup, the worst case corners are disabled. To change the default status of these corners in the active setup to `enabled`, set the enableWCCfromRunPlanInActiveSetup environment variable to `t`.

© 2023 Cadence Design Systems, Inc.

22

Working with Specifications

You define performance specifications on the Outputs Setup tab of the Outputs pane. After simulation, the program displays information so that you can see whether your design met, nearly met, or failed to meet your performance specifications.

You can view specification (spec) results from one or more history items (checkpoints) on the Results tab of the Outputs pane. You can select which measurement result for which corner or sweep to display. The program displays specification information such as the corner or sweep name, the analysis name, the measurement name, the conditions, the minimum or maximum values. You can view the spec for each measurement along with information about whether each spec passed or failed. You can create a summary report of pass/fail results.

You can control the format for each specification such that the program uses particular units or engineering or scientific notation when displaying each spec. You can sort by measurement name and export your spec sheet to a comma-separated values file.

See the following topics for details:

- Working with Specifications on page 1330
- Comparing Waveforms Using Wave Specifications on page 1342
- Defining Operating Region Specifications on page 1378
- Working with the Specification Summary on page 1415

© 2023 Cadence Design Systems, Inc.

Working with Specifications

The outputs of type Expression, OCEAN Script, MATLAB Script, Area Goal, or Op Region Spec that are added in the Outputs Setup tab of the Outputs pane are called measurements.

You can define a performance specification for a measurement in the *Spec* column on the Outputs Setup tab of the Outputs pane or in the Override Specifications form.

By default, the specification defined for a measurement applies to all the corners enabled for the test. You can override the measurement specification for a corner or disable the corner specification.

For more information, see the following topics:

- Defining a Specification on page 1331
- Using Variables in Specifications on page 1332
- Overriding the Measurement Specification for a Corner on page 1334
- Removing a Corner Specification Override on page 1339
- Disabling and Enabling Corner Specifications on page 1339
- Defining Operating Region Specifications on page 1378
- Viewing Specification Results in the Results Tab on page 1403
- Viewing Operating Region Violations on page 1408

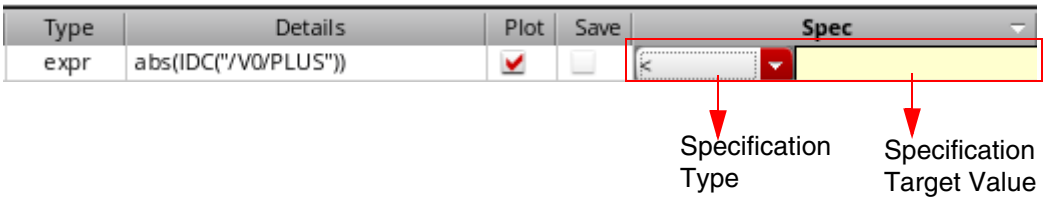
© 2023 Cadence Design Systems, Inc.

Defining a Specification

To define a specification for a measurement on the Outputs Setup tab, do the following:

1. On the Outputs Setup tab of the Outputs pane, double-click in the *Spec* column.

   The *Spec* cell has two parts: a specification type (which you select from a drop-down list) and a target value.



2. In the drop-down list that appears in the left half of the cell, select one of the following specification types and fill in the right half of the cell with an appropriate target information as indicated:

Table 22-1 Specification Types

Objective	Target	Description
minimize	`targetValue`	Minimizes the result; optional target value
maximize	`targetValue`	Maximizes the result; optional target value
<	`targetValue`	Returns the pass status when the result is less than the target value
>	`targetValue`	Returns the pass status when the result is greater than the target value
range	`lowerBound upperBound`	Returns the pass status when the result falls between the boundary values
tol	`targetValue percentageDeviation`	Returns the pass status when the result is within the specified percentage deviation of the target value
info		Returns the pass status when the result is a scalar value. It shows the fail status if the result is a waveform.

© 2023 Cadence Design Systems, Inc.

Table 22-1 Specification Types

Objective	Target	Description
wave		Compares and validates the analog and digital waveform signals for the reference and compared data sources. For details, refer to Comparing Waveforms Using Wave Specifications.

Note: A target specification value can be specified in the form of constant values or expressions using global variables or other output names. For more details, refer to Using Variables in Specifications.

3. (Optional) In the *Weight* column next to the specification, specify the weighting factor for the specification.

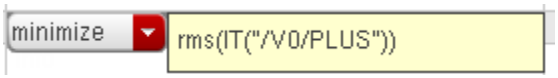
Using Variables in Specifications

The target values for specifications can be specified as a constant or an expression using variables. In case the value is specified as an expression, the evaluated value of the expression is compared with the output value to identify the *pass* or *fail* status. By using expressions in specifications, you can vary the specifications across sweep points.

Note: Currently, only the `Single Run, Sweeps, and Corners` and `Monte Carlo` run modes support expressions in specification target values.

The expressions can be created for specification target values in any of the following ways:

- **By creating calculator expressions**: An example is given below.



Note: The field for specification value is a multiline text field that is expanded automatically to accommodate the complete expression.

- **By using global variables**: You can create an expression that refers to a global variable using the `VAR` function. For example, if the ADE Assembler setup has a global variable `maxLimit`, you can specify a specification value as `VAR("maxLimit")`.
- **By using other outputs of the same test**: You can use the `calcVal` function to create an expression that refers to another output of the same test. For example, if the test `Test1` has two outputs, `expr1` and `expr2`, the specification value for output `expr2` can

© 2023 Cadence Design Systems, Inc.

use the expression `calcVal("expr1" "Test1")`, which is referring to output `expr1` of the same test.

- **By using the outputs of other tests**: For this, you can use the `calcVal` function to refer to the output of another test in the same ADE Assembler setup. For example, specification value for an output of `Test2` can use the expression `calcVal("tdcd" "Test1")`, which is referring to output `tdcd` of `Test1`.
- **By using a SKILL or OCEAN script**: For this, you can add the SKILL or OCEAN code in the *Spec* column. Alternatively, you can create a separate output with a SKILL or OCEAN script. Result of that output can be referred in the expression for a specification type.

Important Points to Note

- In case of an overridden specification, a corner can use only a constant value as an overridden target value. You cannot specify an overridden expression different from that specified for the global target value.
- For a history checkpoint, you can change a constant specification value to an expression and re-evaluate the results.

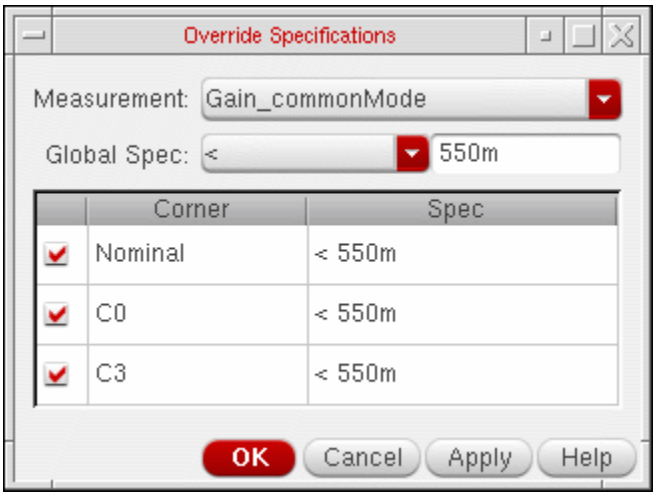
Also see: Viewing Specification Results in the Results Tab.

© 2023 Cadence Design Systems, Inc.

Overriding the Measurement Specification for a Corner

By default, the specification defined for a measurement applies to all the corners enabled for the test. For example, assume that the nominal corner and two corners named `C0` and `C3` are enabled for a test named `myTest`. If you define a measurement named `Gain_commonMode` for the test, the specification for the `Gain_commonMode` measurement applies to the nominal corner and to corners `C0` and `C3`.

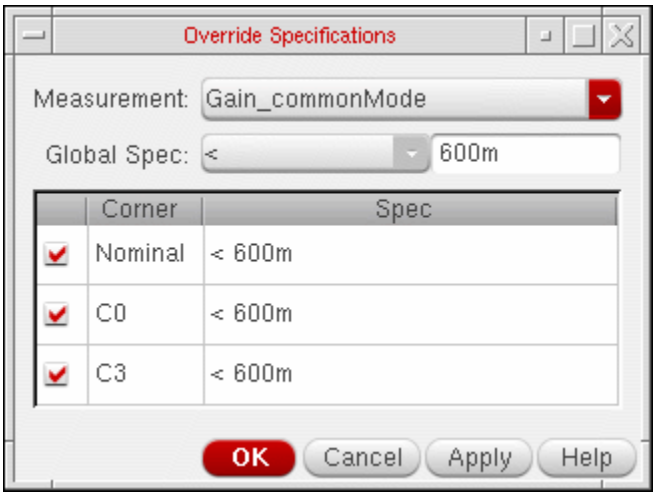
For example, in the following figure, the specification `<550m` for the `Gain_commonMode` measurement applies to all the corners displayed in the *Corner* column.



If you modify the specification for a measurement in the Outputs Setup tab of the Outputs pane or in the *Global Spec* field on the Override Specifications form, the change will apply to all the corners selected for the test. For example, the following figure indicates that when

© 2023 Cadence Design Systems, Inc.

the specification for the `Gain_commonMode` measurement is changed from `<550m` to `<600m`, the change is applied to all the corners displayed in the *Corner* column.



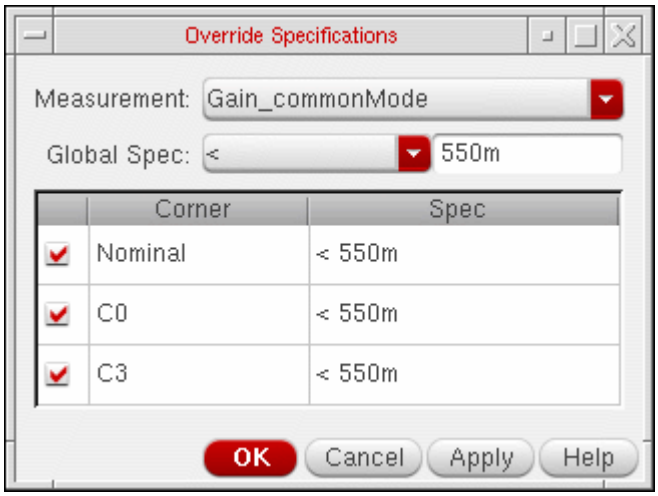
You can override the measurement specification for a corner by specifying a different specification for the corner.

To define a specification for a measurement in the Override Specifications form, do the following:

1. Right-click the row for a measurement in the Outputs Setup tab of the Outputs pane and choose *Override Specifications* from the context-sensitive menu.

© 2023 Cadence Design Systems, Inc.

The Override Specifications form appears.



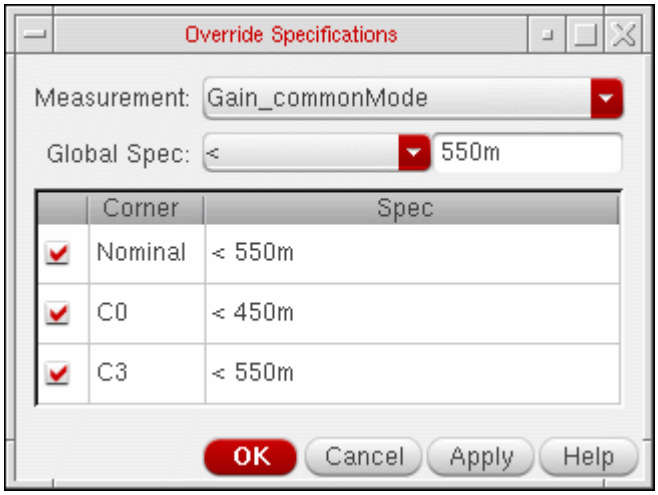
Field	Description
Measurement	Displays the measurements for which you can define specifications. Note the following: ■ By default, the name of the measurement you selected on the Outputs Setup tab of the Outputs pane is displayed. ■ This drop-down list displays only the names of the measurements for the test whose measurement you selected on the Outputs Setup tab. To define a spec for a measurement of another test, right-click a measurement for that test and choose *Override Specifications*. ■ If you have not specified the name for a measurement in the *Name* column on the Outputs Setup tab, the expression for that measurement is displayed as its name in this drop-down list.
Global Spec	Displays the specification for the measurement. To define or modify a specification, select one of the specification types from the drop-down list, and fill in the text field next to the drop-down list with appropriate target information for the measurement.

© 2023 Cadence Design Systems, Inc.

Field	Description
Corner	Displays the list of corners enabled for the test. You can disable or enable a corner specification for a specific measurement. For more information, see Disabling and Enabling Corner Specifications
Spec	Displays the specification for each corner enabled for the test. For more information about specifying a different specification for a corner, see Overriding the Measurement Specification for a Corner.

2. In the *Measurement* drop-down list, select the measurement for which you want to define the specification.
3. In the *Global Spec* drop-down list, select one of the specification types from the drop-down list, and fill in the text field next to it with appropriate target information for the measurement.
4. In the *Spec* column, specify an overridden value next to the corner for which you want to use a different specification.

   For example, in the following figure, the specification `<450m` for corner `C0` overrides the specification `<550m` for the `Gain_commonMode` measurement.

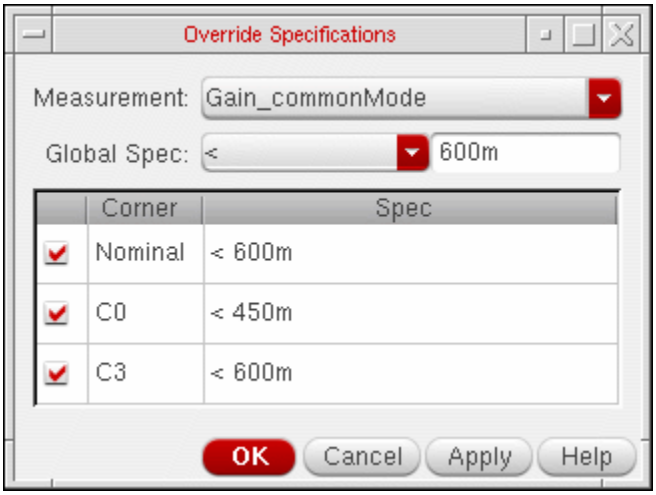


   Note: In case of an overridden specification, a corner can use only a constant value as an overridden value. You cannot specify an overridden expression different from that specified for the global target value.

5. Click *OK*.

© 2023 Cadence Design Systems, Inc.

Note: If you later modify the specification for a measurement in the Outputs Setup tab of the Outputs pane or in the *Global Spec* field on the Edit Specification form, the change will apply only to the corners for which you did not override the specification. For example, the following figure indicates that when the specification for the `Gain_commonMode` measurement is changed from `<550m` to `<600m`, the change is applied only to the `Nominal` corner and corner `C3`. The change is not applied to corner `C0` because the specification `<450m` for corner `C0` overrides the specification for the `Gain_commonMode` measurement.



By default, the specification applies to all the corners selected for the test. You can do the following:

- Override the specification for a specific corner. For more information, see Overriding the Measurement Specification for a Corner.
- Undo a corner specification override. For more information, see Removing a Corner Specification Override.
- Disable or enable the corner specification for a specific measurement. For more information, see Disabling and Enabling Corner Specifications.
- Disable or enable the corner specification across all measurements for a test. For more information, see Disabling and Enabling Corner Specifications.

The next time you run a simulation, the program measures the result against the performance specification and displays the results in the Results tab of the Outputs pane. For more information about how results are displayed in the Results tab, see Viewing Specification Results in the Results Tab.

© 2023 Cadence Design Systems, Inc.

Tip

If you add or modify measurement expressions or specifications after running a simulation, you can click the button on the Results tab to update the results displayed in the Results tab based on the new or modified expressions and specifications.

The results will also be updated based on the new or modified expressions and specifications if you click the button on the Results tab to plot across all points. For more information about using the button, see Refreshing Graphs.

Note: For information about setting up specifications for optimization, see Setting Up Specifications.

See also:

- Starting a Simulation
- Re-Evaluation of Expressions and Specifications
- Viewing Specification Results in the Results Tab

Removing a Corner Specification Override

To remove a corner specification override, do the following in the Override Specifications form:

1. In the *Measurement* drop-down list, select the measurement for which you want to undo a corner specification override.
2. In the *Spec* column, right-click the specification for the corner for which you want to undo the specification override and choose *Set to Global* from the context-sensitive menu.

   To undo the overrides for multiple corners, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection), right-click and then choose *Set to Global* from the context-sensitive menu.

   The specification for the measurement (displayed in the *Global Spec* field) is applied to the corner.

Disabling and Enabling Corner Specifications

By default, the specification defined for a measurement applies to all corners enabled for the test. You can do the following:

© 2023 Cadence Design Systems, Inc.

- Disable or enable the corner specification for a specific measurement.
- Disable or enable the corner specification across all measurements for a test.

To disable or enable a corner specification for a specific measurement, do the following:

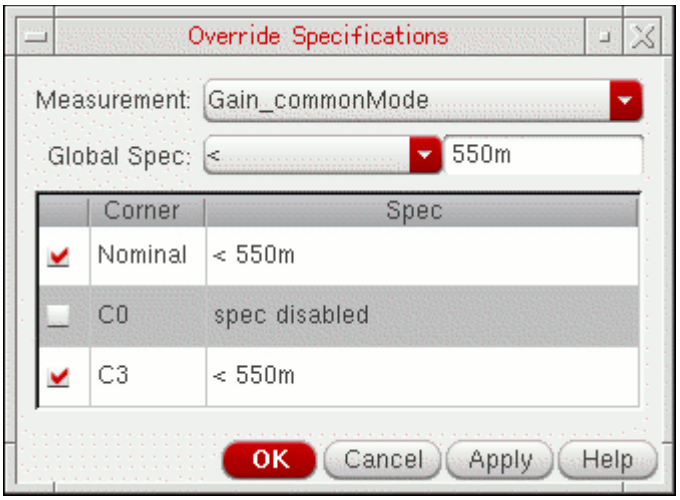
1. Right-click the row for a measurement in the Outputs Setup tab of the Outputs pane and choose *Override Specifications* from the context-sensitive menu.

   The Override Specifications form appears.

2. In the *Measurement* drop-down list, select the measurement for which you want to disable or enable the corner specification.
3. Do one of the following:

   - To disable a corner specification for the measurement, clear the check box next to the corner.

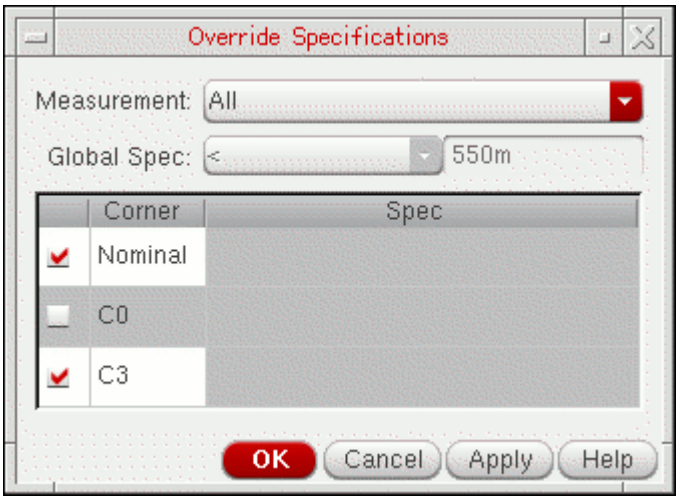
     For example, in the following figure, the specification for corner `C0` is disabled.



   - To enable a corner specification for the measurement, select the check box next to the corner.
   - To disable a corner specification for all the measurements for the test, select *All* in the *Measurement* drop-down list, then clear the check box next to the corner.

© 2023 Cadence Design Systems, Inc.

For example, in the following figure, the specification for corner `C0` is disabled for all the measurements for the test.



- To enable a corner specification for all the measurements for the test, select *All* in the *Measurement* drop-down list, then select the check box next to the corner.

Note: You can also right-click the first column in the Override Specifications form and choose *Disable All* or *Enable All* to disable or enable all the corner specifications.

4. Click *OK*.

See also:

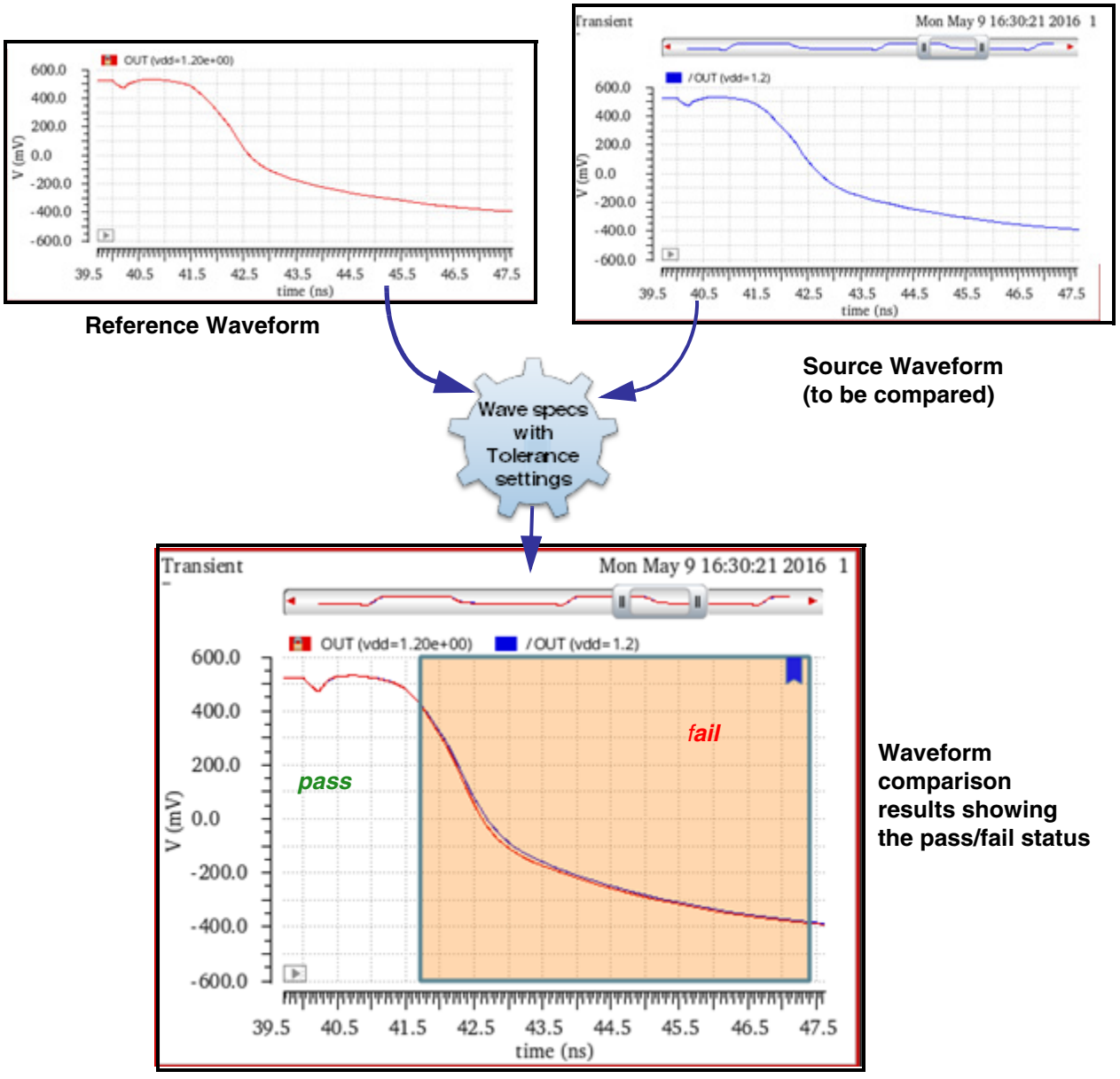
Viewing Specification Results in the Results Tab

© 2023 Cadence Design Systems, Inc.

Comparing Waveforms Using Wave Specifications

ADE Assembler provides various types of specifications, such as <, >, range, tol, etc. that you can use to compare scalar values in simulation results with the given targets. Depending on the comparison results, the status of the resulting scalar values is shown as pass/fail.

Similarly, you can also compare and validate the waveforms in simulation results with the given target waveforms using the `wave` type of specification as per the acceptable tolerance values.



© 2023 Cadence Design Systems, Inc.

ADE Assembler supports waveform comparison for the following formats:

- Cadence SignalScan Turbo 2 (SST2)
- Cadence Parameter Storage Format (PSF)
- Cadence PSF XL
- Fast Signal DataBase (FSDB)

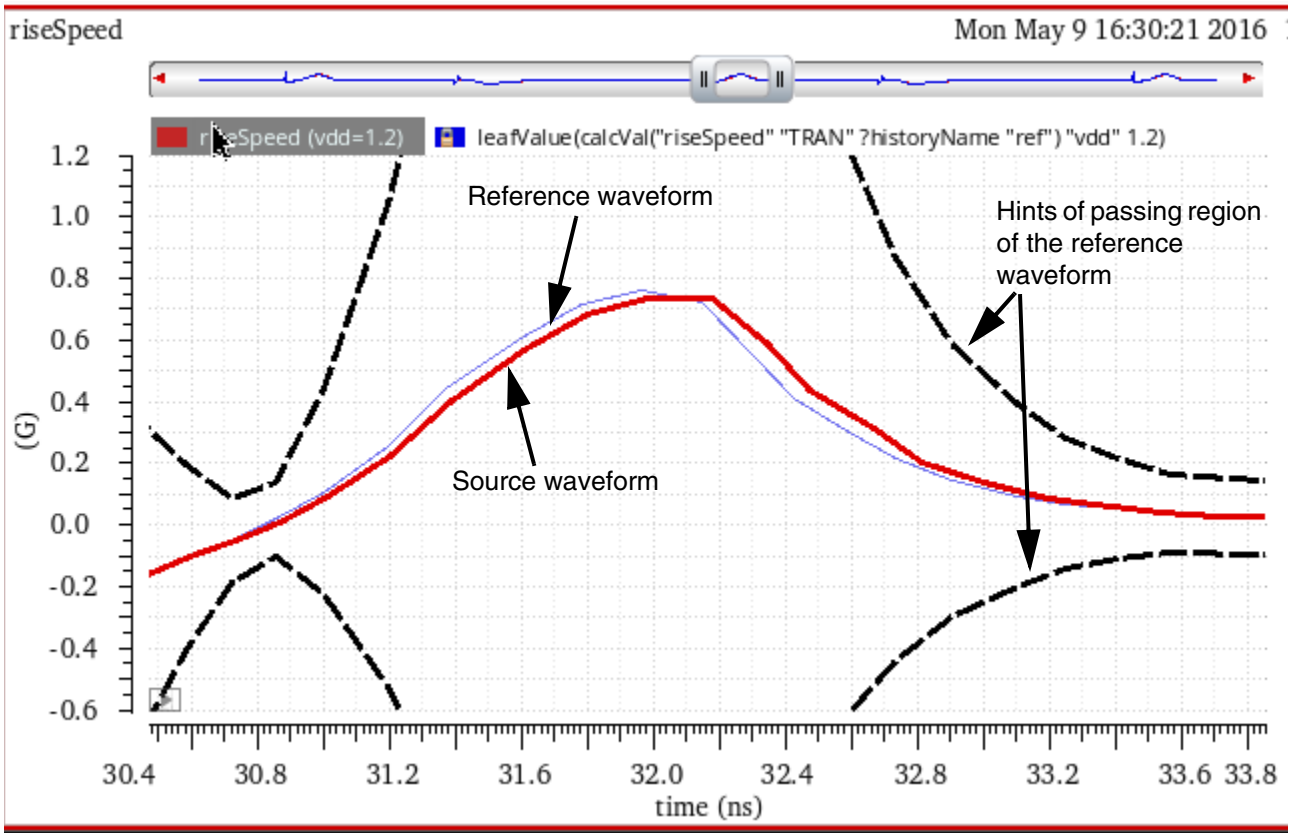
For waveform comparison, you need to specify:

- **A source (to be compared) data source:** An output or measurement in the current setup that would result in a waveform to be compared.
- **A reference data source:** An output or measurement to be taken as target or reference against which the source is to be compared.

  Note: A reference data source can be the history of the current run, or a different history saved in the same or different maestro cellview. You can also choose a reference data source from an adexl cellview.
- **A wave spec:** A specification to be set on the data source to be compared.
- **Analysis and tolerance settings:** These settings can be specified globally for all the waveforms to be compared, or individually for each waveform.

© 2023 Cadence Design Systems, Inc.

When you run waveform comparison for two signals, the waveforms or signals are plotted in Virtuoso Visualization and Analysis XL graph window.



You can identify the difference between the waves, zoom into them for a closer view, and see the passing or failing areas as per the given tolerance settings.

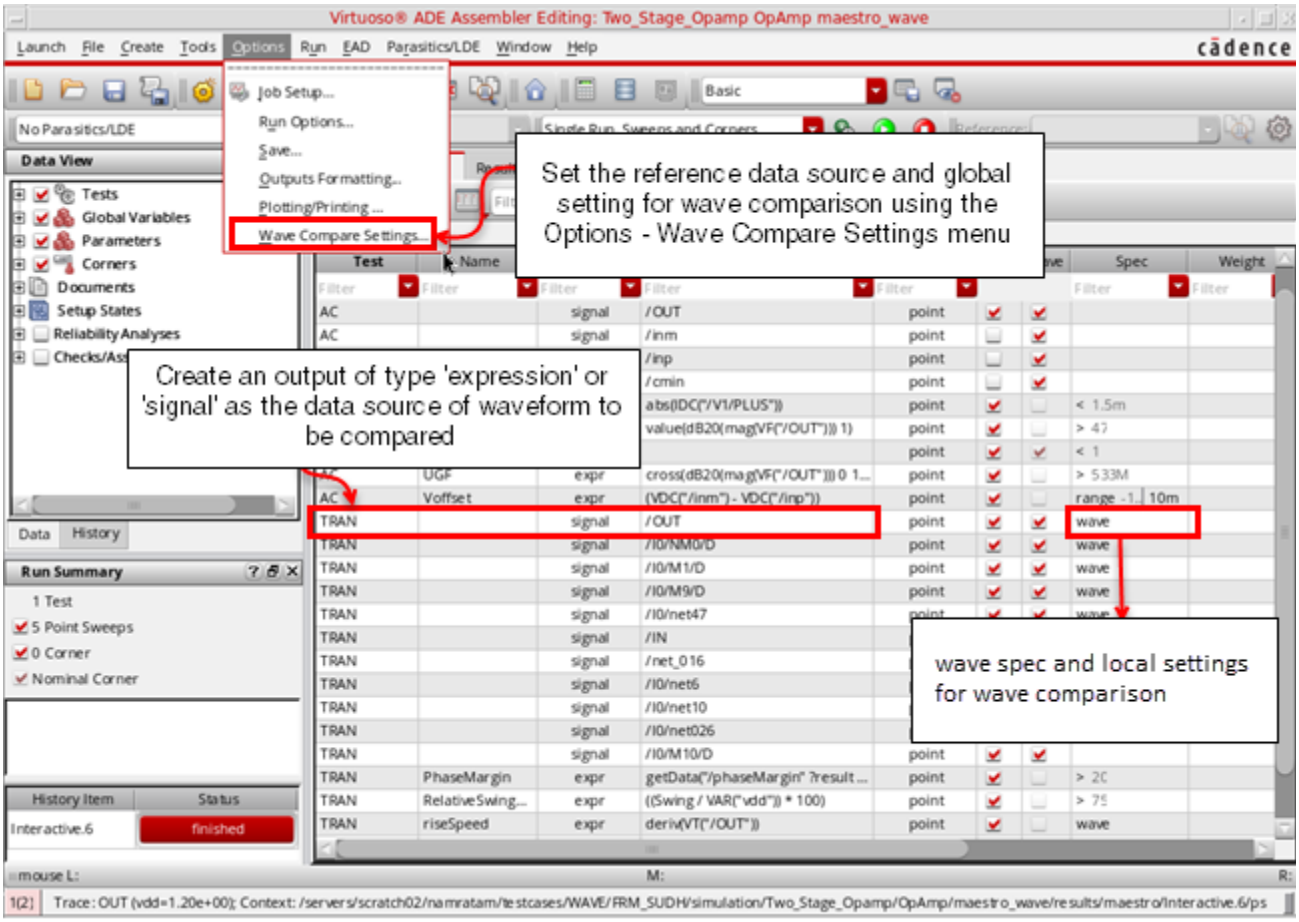
Video

You can view a video demonstration of this feature at Using Waveform Specifications in Assembler.

© 2023 Cadence Design Systems, Inc.

Performing Waveform Comparison

You can use the menus and commands in ADE Assembler to specify the data sources and tolerance settings.



To compare waveform signals, do the following:

1. Specify the global options (reference history and tolerance settings) to be used for waveform comparison. You can do this by using the Global Waveform Compare Settings form. For more details, refer to Specifying Global Options for Waveform Comparison.
2. Ensure that your test in ADE Assembler contains an analysis of type `tran`. If it does not have, create a new analysis of this type.

   Note: Currently, waveform comparison is supported only for tran analysis.
3. For tran analysis, create outputs of type `signal` or `expr` that you need to compare to a reference output from a reference data source.

   For digital outputs, you can choose bus signals. For example, `/I0/net1<0:10>`.

© 2023 Cadence Design Systems, Inc.

4. Ensure that the check box in the *Save* column for this output is selected.

   Note: For waveform comparison, it is required to save the raw waveform data.

5. Specify `wave` specification for the output to be compared. For this, click in the *Spec* column of the output and select `wave` from the drop-down list.

   By default, the global comparison options specified in **step 1** are applied to the wave spec for this output. However, you can override the global options at the output level. For this, click ( ) to the right of the drop-down list in the *Spec* column.

Tip

You can set wave specifications for multiple outputs together by using the *Create wave spec for all signals* or *Create wave spec for selected outputs* commands available in the ( ) drop-down list on the Outputs Setup pane toolbar. For more details, refer to Applying Waveform Options to Multiple Outputs Together.

6. (Optional) Verify the comparison options for the output in any of the following ways:

   - Move the mouse over the ( ) button. A tooltip is displayed to show the waveform comparison settings applied for this output.
   - Click ( ) to open the Waveform Compare Setup form in which the waveform comparison settings are displayed.

7. (Optional) If required, change the values to override the options. For more details, refer to Specifying Local Options for Waveform Comparison.

8. Run simulation to generate results that are saved in a history.

9. View the wave comparison results on the *Results* tab.

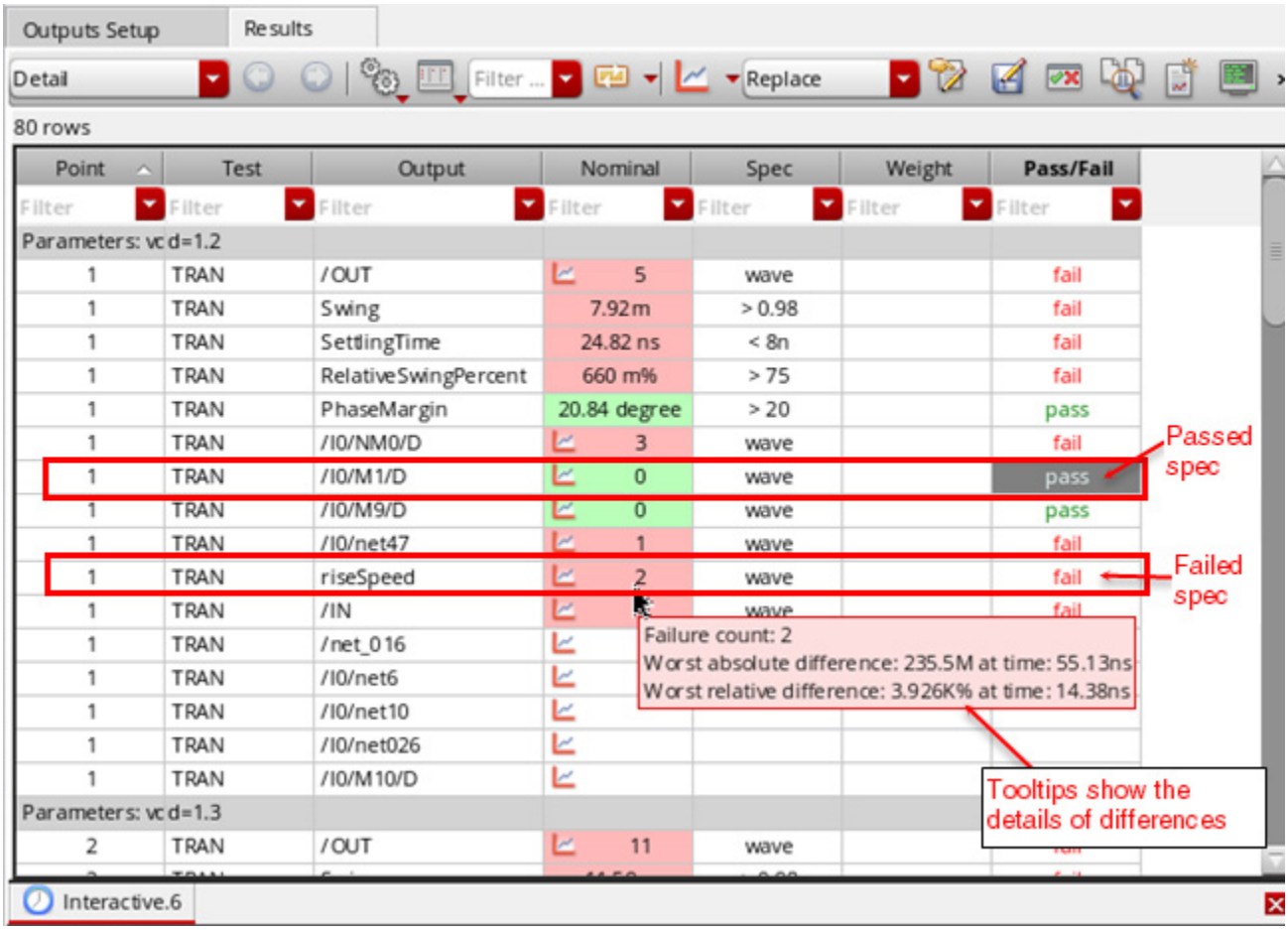
   As shown in the figure below, the status of waveform comparison is displayed in the *Pass/Fail* column.

   Observe the following:

   - If difference between the source and reference wave is within the tolerance range, the output row is colored green and the pass/fail status is displayed as `pass`. In addition, the error count next to the plot icon is set to `0`.
   - If the difference is out of the tolerance range, the pass/fail status is displayed as `fail` and the error count is greater than `0`.

© 2023 Cadence Design Systems, Inc.

Move the mouse over the error count to view the failure details in a tooltip.



10. Right-click in the result cell and choose *Plot* to view the comparison of waveforms or traces. For more details, refer to Analyzing Waveform Comparison Results.

The following topics provide more details about waveform comparison:

- Specifying Global Options for Waveform Comparison
- Specifying Local Options for Waveform Comparison
- Analyzing Waveform Comparison Results
- Waveform Compare Algorithms

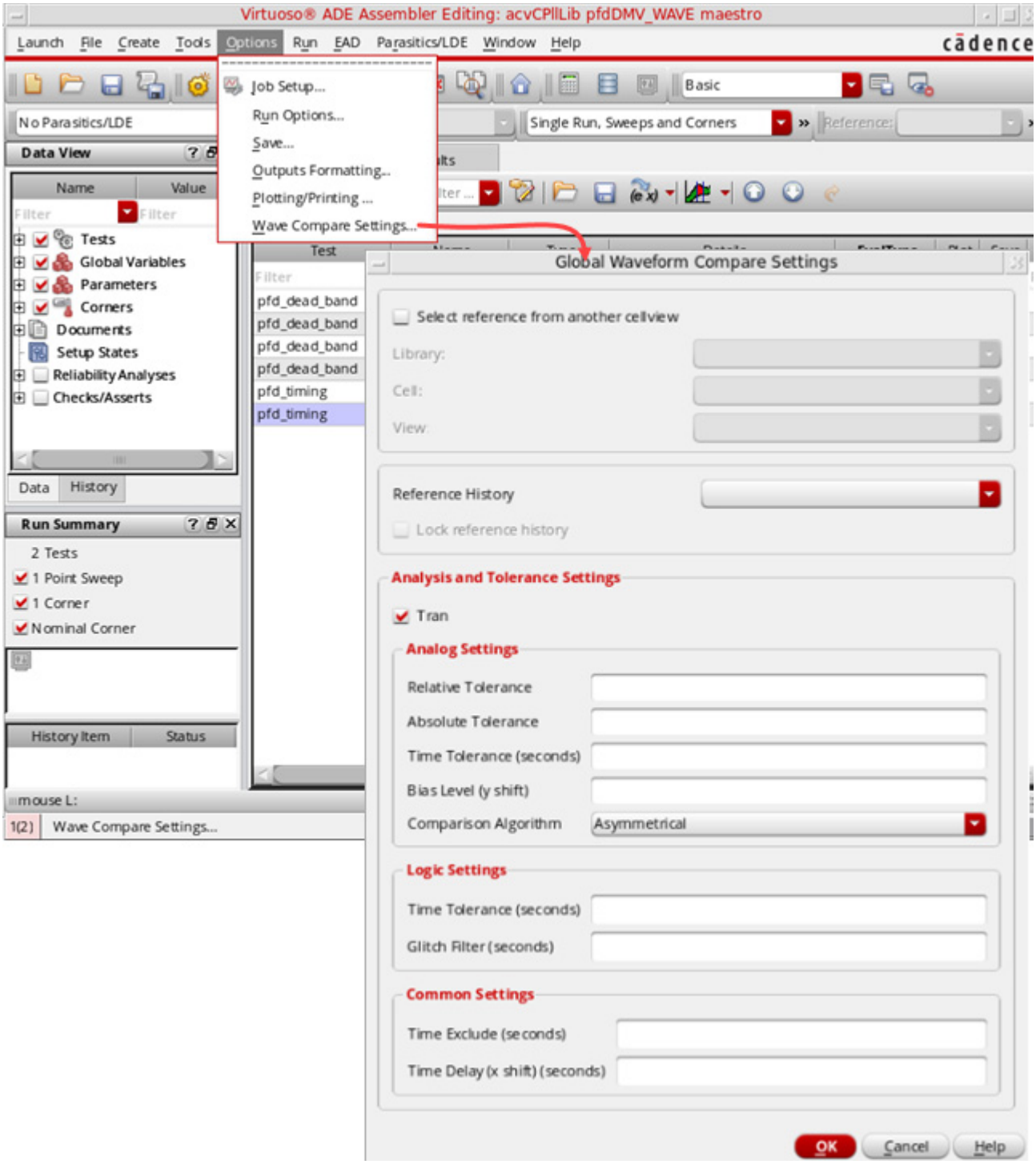
© 2023 Cadence Design Systems, Inc.

Specifying Global Options for Waveform Comparison

You can use the options in the Global Waveform Compare Settings form to control how all the waveforms are validated. These options apply to all the outputs with wave specs defined for them.

© 2023 Cadence Design Systems, Inc.

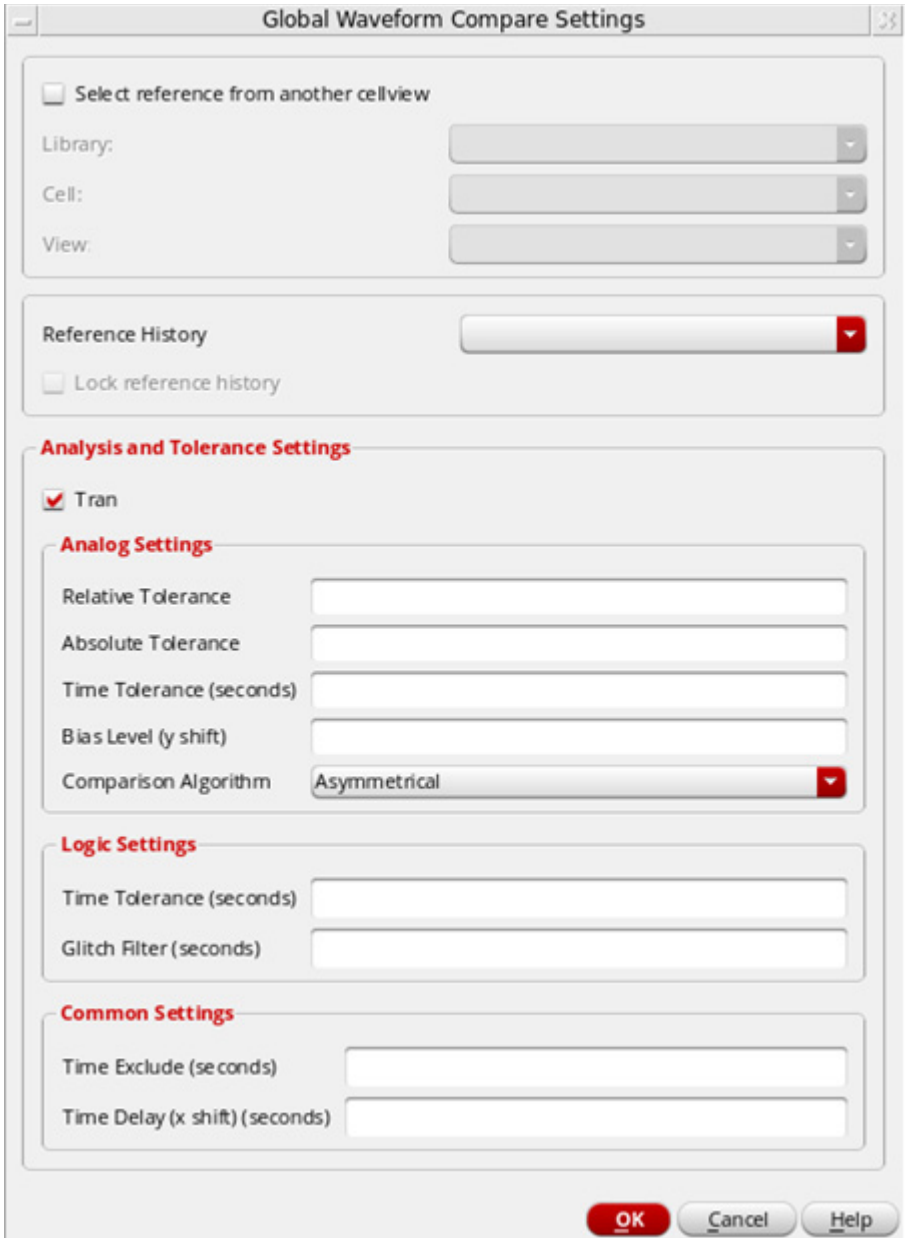
Choose *Options – Wave Compare Settings* to open the Global Waveform Compare Settings form, as shown below.



© 2023 Cadence Design Systems, Inc.

Note: If required, you can override the global options for any specific signal. For more information, see Specifying Local Options for Waveform Comparison.

By default, the fields on the Global Waveform Compare Settings form are blank. Fill in the required values.



The following sections describe the values you can set in this form:

- Reference History to be Used as Data Source

© 2023 Cadence Design Systems, Inc.

- Analysis and Tolerance Settings for Waveform Comparison
- Maximum Acceptable Tolerance Values
- Time Tolerance Values
- Time Ranges to be Excluded from Compared Waveforms
- Parameters to Shift Data of a Compared Waveform Before Comparison
- Algorithm for Comparison
- Applying Waveform Options to Multiple Outputs Together

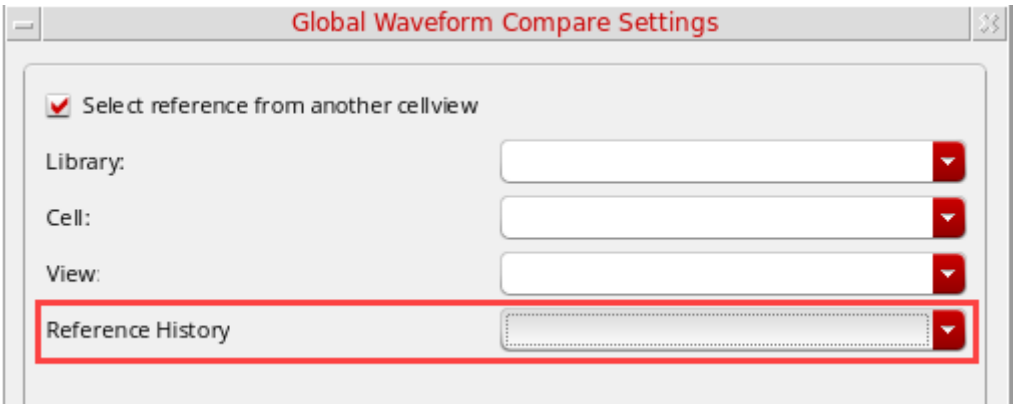
Reference History to be Used as Data Source

You can use a saved history from the current or any other cellview as a reference history for waveform comparison.

To choose a reference history from another cellview:

1. Select *Select reference from another cellview* on the Global Waveform Compare Settings form.

   The form fields to specify the library, cell, and view of another cellview are enabled. In addition, the *Reference History* field is also included in the group displayed on top of the form.



2. Use the *Library*, *Cell*, and *View* drop-down lists to specify the maestro cellview of the reference history.

   All the histories available in the specified cellview are listed in the *Reference History* drop-down list.

3. Select a reference history from the *Reference History* drop-down list.

© 2023 Cadence Design Systems, Inc.

To choose a reference history from the current cellview:

1. Select a history from the *Reference History* drop-down list, which shows the list of histories available in the current maestro cellview.
2. Select the *Lock Reference History* check box to save the reference history from being overwritten. It is recommended to lock the history if you need to use its results for many waveform comparisons. Otherwise, depending on the *History Entries to Save* settings on the Save Options form, when new simulations are run, the selected history might get overwritten.

Important Points to Note

- For the runs that create children histories, for example, Manual Tuning, only the names of children histories are available in the *Reference History* drop-down list.
- You cannot use the results saved from ADE Explorer as reference. Therefore, `Explorer.0` is not available in the *Reference History* drop-down list.
- When the setup contains multiple corners, ADE Assembler can compare waveforms only when either of the following two conditions is met:
  - Both source and referenced history contain the same number of corners (n, where n>=1). In this case, one-to-one mapping of corners is done for comparison.
  - The compared setup contains n corners and referenced history contains 1 corner. In this case, wave for each corner from the compared setup is compared with the wave for this single corner in the reference history.
  - In other cases of mismatch in the number of corners, ADE Assembler shows errors.

Analysis and Tolerance Settings for Waveform Comparison

Currently, ADE Assembler supports waveform comparison only for transient analysis. To ensure that comparison are run for the results of this analysis, select the *Tran* check box.

The tolerance settings that you can specify on this form are grouped in the following categories:

- Analog Settings
- Logic Settings
- Common Settings

© 2023 Cadence Design Systems, Inc.

Analog Settings

These settings are used when comparing waveforms for analog signals.

Field	Description
Relative Tolerance	The maximum acceptable % tolerance value for the worst-case relative deviation reported for analog signals. For more information, refer to Maximum Acceptable Tolerance Values.
Absolute Tolerance	The maximum acceptable tolerance value for the worst-case absolute deviation reported for analog signals For more information, refer to Maximum Acceptable Tolerance Values.
Time Tolerance (seconds)	The maximum acceptable tolerance value for the time difference (skew) between transition times reported for analog signals For more information, refer to Time Tolerance Values.
Bias Level (y shift)	An integer value by which the waveform needs to be shifted on y axis For more information, refer to Parameters to Shift Data of a Compared Waveform Before Comparison.
Comparison Algorithm	Algorithm to be used to compare analog waveforms For more information, refer to Algorithm for Comparison.

© 2023 Cadence Design Systems, Inc.

Logic Settings

These settings are used when comparing waveforms for digital signals.

Field	Description
Time Tolerance (seconds)	Specifies the maximum acceptable tolerance value for the time difference (skew) between transition times reported for digital signals. The default unit of time for this field is seconds. For more information, refer to Specifying the Time Tolerance Values for Digital Signals.
Glitch Filter (seconds)	Specifies the maximum glitch size (in seconds) up to which the glitches in digital signals are acceptable and ignored while comparing waveforms. For more information, refer to Filtering Glitches in Digital Signals.

Common Settings

These settings are common for both analog and digital signals.

Field	Description
Time Exclude (seconds)	The time ranges that should be excluded when validating each signal. Only the time ranges that fall outside the specified time ranges will be validated for each signal. For more information, refer to Time Ranges to be Excluded from Compared Waveforms.
Time Delay (x shift) (seconds)	An integer value by which the waveform needs to be shifted on x axis For more information, refer to Parameters to Shift Data of a Compared Waveform Before Comparison.

© 2023 Cadence Design Systems, Inc.

Maximum Acceptable Tolerance Values

You can specify the maximum acceptable tolerance values—the maximum acceptable differences between the reference and compared analog signals. The compared signal is resampled and interpolated at the reference waveform time points and the values at these time points are compared using the specified tolerances. The tolerances define a window around the reference waveform within which the compared waveform values should fall—if they do not, they fail. For more details, refer to Waveform Compare Algorithms.

To specify the maximum acceptable tolerance values, do either or both of the following:

- In the *Relative Tolerance* field, specify the % maximum acceptable tolerance value for the worst-case relative deviation reported for analog waveforms.
- In the *Absolute Tolerance* field, specify the maximum acceptable tolerance value for the worst-case absolute deviation reported for analog waveforms.

Important

It is required to specify either relative or absolute tolerance. If you do not specify any one of these, ADE Assembler will show an error.

If only one of these tolerance values is defined, Assembler uses only that and ignores the other. If both relative and absolute tolerance are defined, Assembler compares their value for each reference data point and uses the one that is larger.

Note: If a reference waveform value is zero or very close to zero, the relative tolerance evaluates to a very small value. Therefore, the relative tolerance criteria becomes difficult to achieve. In such cases, you should specify a small absolute tolerance value to ensure that the signals are compared with a reasonable tolerance value.

For example, the value of reference waveform is `13mV` at some point. The difference of reference wave and compared wave is `2mV` and the timing tolerance is not specified.

Scenario A:

Relative Tolerance is `1%` (`13mV*0.01=0.13mV`) and absolute Tolerance not specified

→ relative tolerance = fail → it is a fail overall

In this case, the relative tolerance criteria is difficult to match, given that the reference signal is close to zero.

Scenario B:

Relative Tolerance is `1%`. (`13mV*0.01=0.13mV`) and Absolute Tolerance is `10mV`

© 2023 Cadence Design Systems, Inc.

→ relative tolerance = fail, absolute tolerance = pass → it is a pass overall

In this case, the small absolute tolerance helps to overcome the limitation of a small relative tolerance.

The *Pass/Fail* column on the *Results* tab displays a `pass` status for the waveforms that fall within the specified tolerance value, and a `fail` status for the waveforms that fall outside the specified tolerance value. This allows you to validate only the waveforms with the `fail` status.

Time Tolerance Values

You can specify the maximum acceptable time difference (skew) between transition times on the reference and compared waveforms in the *Time Tolerance* field. For more details, refer to Waveform Compare Algorithms

Note the following:

- The default unit of time for the *Time Tolerance* field is seconds.
- If this field is left blank, a simple value comparison is done on the reference and compared waveforms.
- If a value is specified in this field, time and value comparison is done on the reference and compared waveforms.

Algorithm for Comparison

Waveform comparison can use the following three algorithms for waveform comparison:

- Asymmetrical
- Symmetrical, conservative
- Symmetrical, liberal

By default, ADE Assembler uses the asymmetrical algorithm. You can select any other algorithm from the *Comparison Algorithm* drop-down list.

For details on how these algorithms compare the waveforms, see Waveform Compare Algorithms.

© 2023 Cadence Design Systems, Inc.

Time Ranges to be Excluded from Compared Waveforms

You can specify the time ranges that should be excluded when validating each waveform. Only the time ranges that fall outside the specified time ranges will be validated for each waveform.

To specify the time ranges that should be excluded when comparing each waveform specify the time ranges in the *Time Exclude* field.

Use the following syntax to specify a comma-separated list of time ranges:

```
from:to[, from:to]
```

For example, specify `70n:72n,110n:112n` to exclude the time range starting from `70n` to `72` and `110n` to `112n` when validating each waveform.

Note: If the specified exclude time interval falls within a failure region, the failure count is increased by 1.

Parameters to Shift Data of a Compared Waveform Before Comparison

You can shift the data of compared waveforms on x or y axis before comparison.

To shift the data on x axis, specify an integer value in the *Time Delay* field. This adds a delay of the given number of seconds. A positive value shifts the data to the right and a negative value shifts the data to the left.

To shift the data on y axis, specify an integer value in the *Bias Level* field. A positive value shifts the data upwards by the given number of units and a negative value shifts the data downwards.

Specifying the Time Tolerance Values for Digital Signals

You can specify the maximum acceptable time difference (skew) between transition times on the reference and compared digital signals in the *Time Tolerance* field.

A time tolerance, when specified, checks both the timing and function of the digital signal even if there is functional non-equivalence. As a result, if the digital values are same and the timing change in digital values take place within the given time tolerance in the reference and compared signals, the timing and functional check passes. Otherwise, it fails.

Note the following:

- The default unit of time for the Time Tolerance field is seconds.

© 2023 Cadence Design Systems, Inc.

- The logic time tolerance value can be applied in any situation.

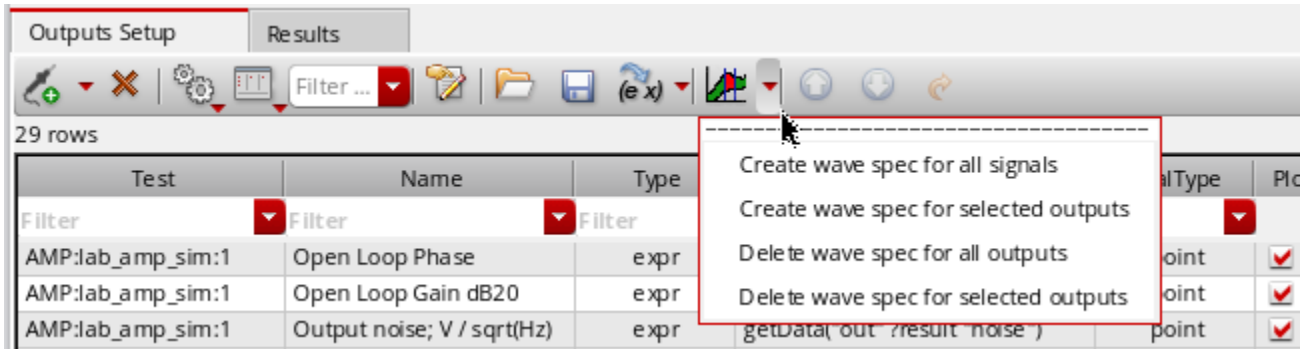
Filtering Glitches in Digital Signals

If there are small acceptable glitches in digital signals because of spurious noise or behavior, you can specify a time value (in seconds) in the *Glitch Filter* field to filter glitches that are equal to or smaller in width than the specified time value. The glitches are ignored when reference and compared digital signals are compared.

The glitch filter, when specified, filters out digital pulses faster than the entered time value before the validation process. Therefore, the glitch filter time should be set carefully to a value significantly smaller than the minimum signal pulse length.

Applying Waveform Options to Multiple Outputs Together

When you need to apply similar wave compare specifications to all or multiple outputs defined in the Outputs Setup pane, you can use the commands available in the ( ) drop-down list on the Outputs Setup pane toolbar.



- *Create wave spec for all signals*: Creates wave specifications for all the signals defined on the Outputs Setup tab. Outputs of other types, such as expr, opregion, and MATLAB, are ignored. This is the default action for the command on the toolbar.
- *Create wave spec for selected outputs*: Creates wave specifications for all the outputs for which you selected rows on the Outputs Setup tab. Ensure that you select outputs that return a waveform result. Output of type opregion, violation, area, and oppoint do not support wave specification.
- *Delete wave spec for all outputs*: Deletes all the wave specifications currently defined on the Outputs Setup tab.
- *Delete wave spec for selected outputs*: Deletes wave specifications for all the outputs for which you selected rows on the Outputs Setup tab.

© 2023 Cadence Design Systems, Inc.

Important Points to Note

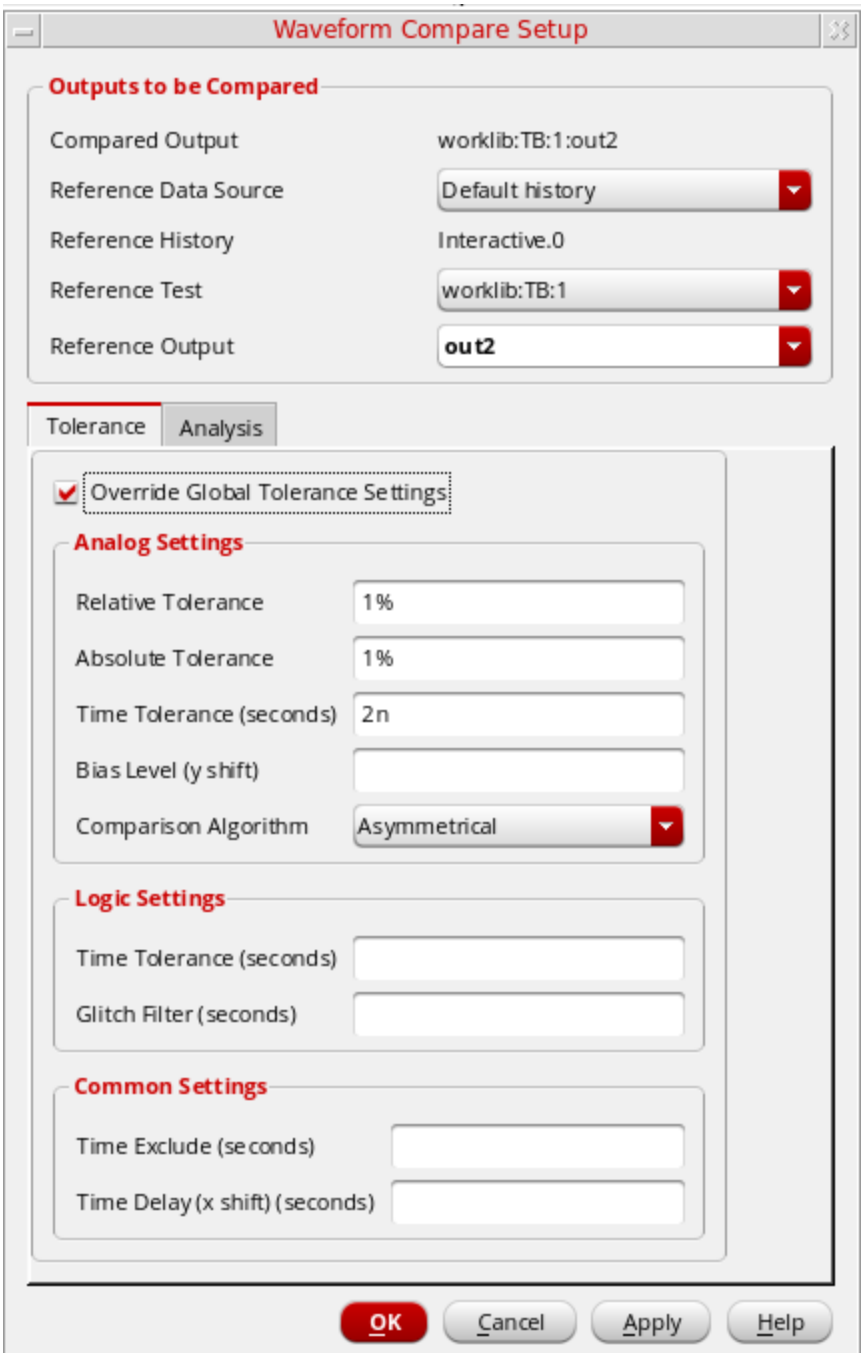
- The wave specifications created by the commands listed above use the settings defined in the Global Waveform Comparison options form.
- If required, you can modify the waveform comparison options individually for any output. For more details, refer to Specifying Local Options for Waveform Comparison.
- If the reference history does not contain any output with the same name as that of the output for which the wave specification is defined, the name of the reference output remains blank. You need to correct the output mapping in such cases. Otherwise, ADE Assembler reports errors while running a simulation.

Specifying Local Options for Waveform Comparison

By default, the options specified in the Global Waveform Comparison Options form are applied to all the waveforms. You can override the global options for specific signal outputs by specifying local options for wave specifications on those signals.

© 2023 Cadence Design Systems, Inc.

To specify local waveform comparison options for a specific signal output, click in the *Spec* column of that output and choose `wave` from the drop-down list. The Waveform Compare Setup form is displayed for that signal output.



The settings on the form are same as the global waveform compare options.

Note the following points related to these settings:

© 2023 Cadence Design Systems, Inc.

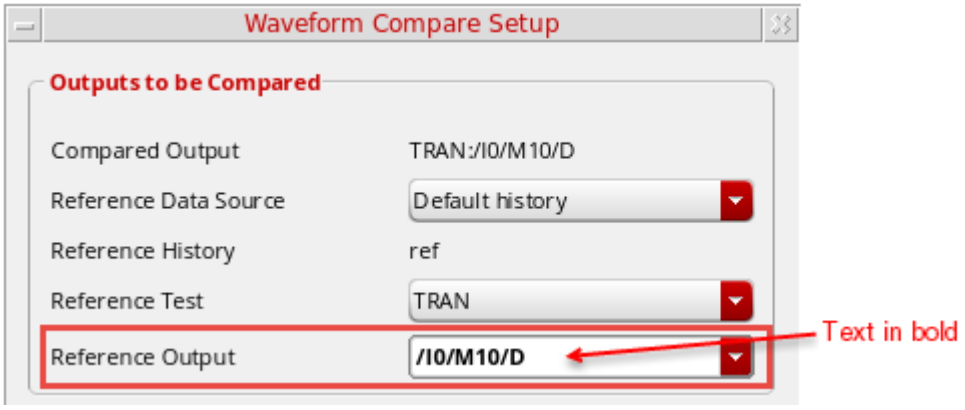
- The *Reference Data Source* field is set to `Default History`, which indicates that the default history set in the global options to be used here.

  If required, you can select other data sources from the drop-down list. The other fields in the *Outputs to be Compared* section of this form change accordingly.

  The other data sources can be:

  - `Specific history`: Use this to select a history other than that specified in global options. You also need to select a reference test and reference output from that history in the *Reference Test* and *Reference Output* drop-down lists, respectively.
  - `Current setup`: Use this when the reference output is another signal output from the current setup. In this case, the reference and compared data source is same. You need to select the name of reference output in the *Reference Output* drop-down list.

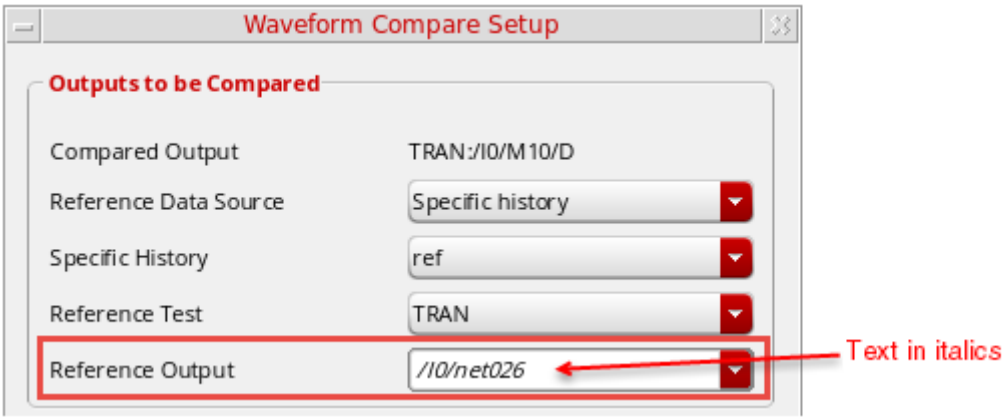
- If the *Reference Data Source* field is set to `Default History` or `Specific History`, the name of that history is displayed in the *Reference History* field.
- If the reference history contains a test with the same name as that of the current test for which you added a signal output, that is displayed in the *Reference Test* drop-down list.
- If the reference test contains a signal output with the same name as that of the compared output, it is displayed in the *Reference Output* drop-down list. The **bold** format of this value also indicates that the names of reference and compared waveforms are same.



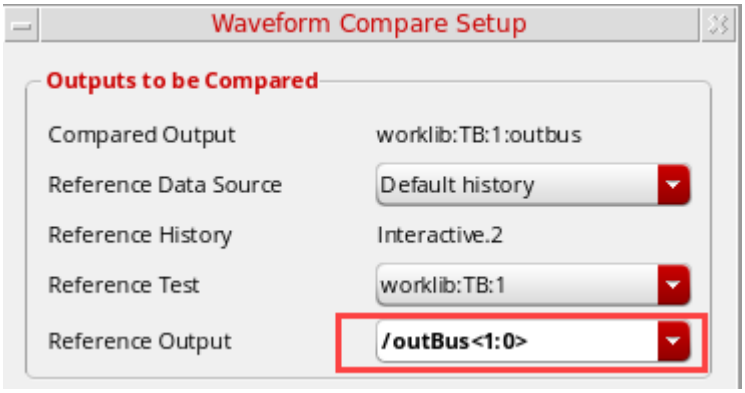
- If the reference test does not contain a signal output with the same name as that of the compared output, the *Reference Output* drop-down list is left blank. You can select an

© 2023 Cadence Design Systems, Inc.

appropriate signal name. The *italicized* format of this value indicates that the names of reference and compared waveforms are different.



For digital outputs, waveform comparison can be done for signals or buses. Therefore, you can choose a bus as a reference output, as shown below.



- The Tolerance tab shows the settings copied from the Global Waveform Comparison Options form. You need to change these settings for a specific output, select the *Override Global Tolerance Settings* check box. The fields for tolerance value settings are enabled and their values are deleted. You can edit the values as required.

  For details about the values that you can specify in these fields, refer to the following sections:

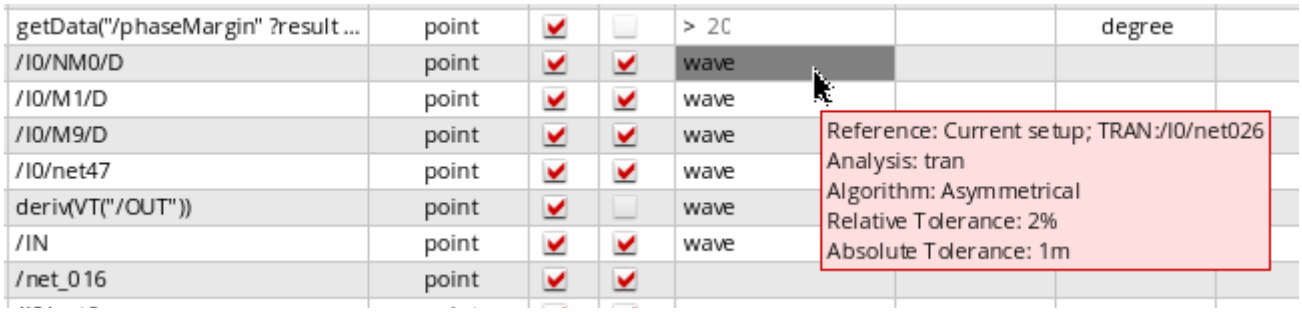
  - Maximum Acceptable Tolerance Values
  - Time Tolerance Values
  - Time Ranges to be Excluded from Compared Waveforms
  - Parameters to Shift Data of a Compared Waveform Before Comparison

© 2023 Cadence Design Systems, Inc.

- Algorithm for Comparison
- Specifying the Time Tolerance Values for Digital Signals
- Filtering Glitches in Digital Signals

Tip

Move the mouse over a wave spec to view its details in a tooltip.



Analyzing Waveform Comparison Results

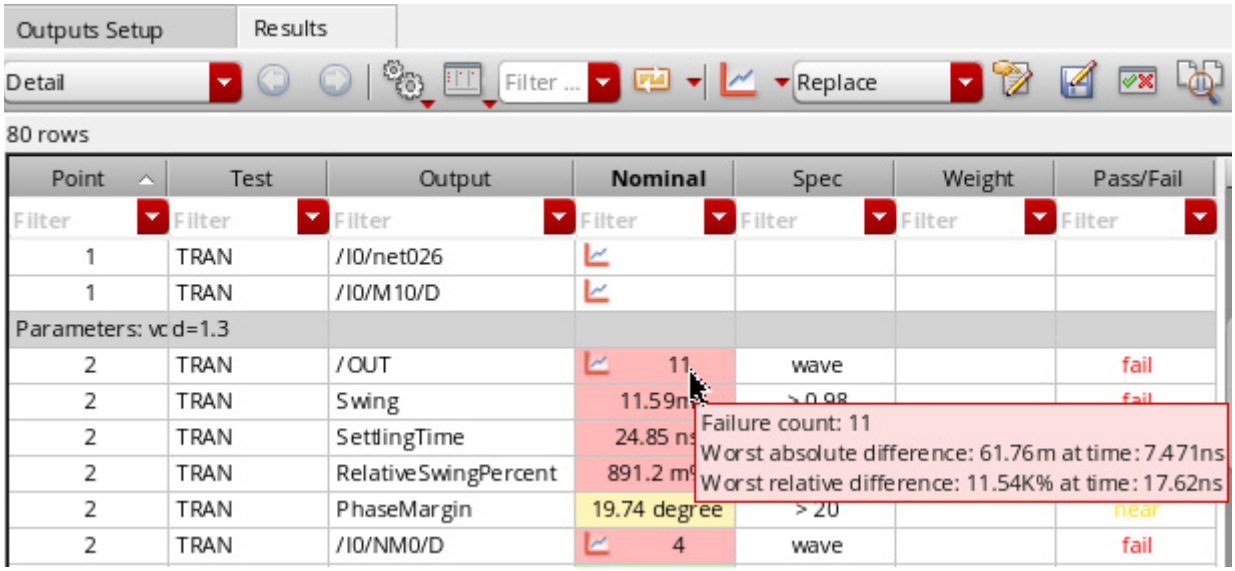
As described earlier, the pass/fail status of signal outputs on the *Results* tab shows the overall result of waveform comparison for each compared signal. In addition, the error count is shown in the result cell and its tooltip, which also shows the details of the maximum difference between the reference and compared waveform.

For example, in the figure shown below, the tooltip indicates the following points:

- The difference between the reference and compared waveforms was greater than the specified time tolerance value at `11` instances.
- The worst-case absolute difference was reported at time=`7.471ns` and the deviation was of `61.76m`.

© 2023 Cadence Design Systems, Inc.

- The worst-case relative deviation was reported at time=`17.62ns` and the deviation was of `11.54%`, which is higher than the tolerance value of `2%`.



Important Points to Note

- When time tolerance values are specified in the waveform compare settings, the tooltip shows only the failure count. It does not show the worst absolute difference and worst relative difference.
- If the specified exclude time interval falls within a failure region, the failure count is increased by 1.
- If the simulations are running across NFS, you can use the maxRDBSyncWait environment variable to increase the wait time to retrieve the results data before running comparison.

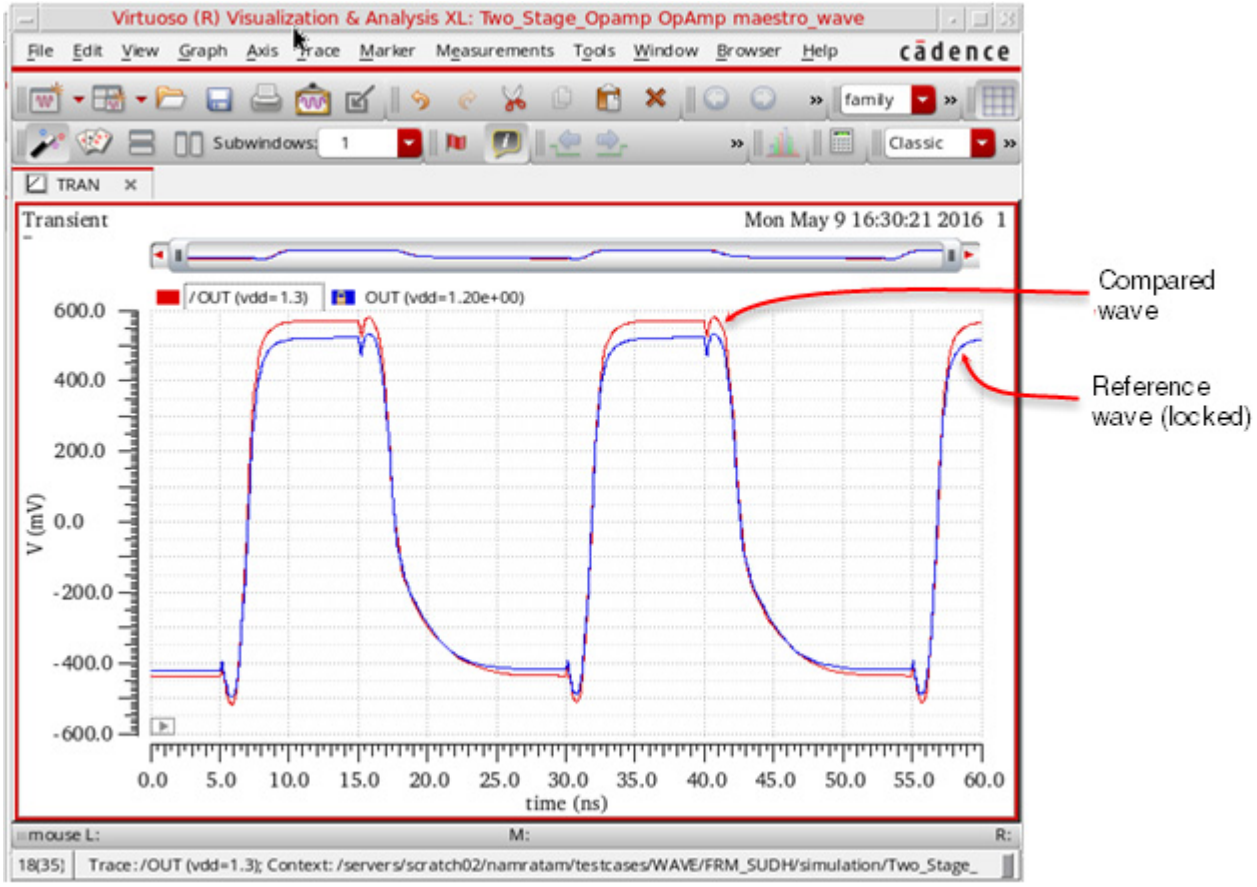
The tooltip display is different in case of logic signals. For more details, refer to Analyzing Waveform Comparison Results for Digital Signals.

You can do a detailed analysis of waveform comparisons by viewing the graphs plotted in the graph window.

To plot the result of a waveform comparison, right-click in a result cell and choose *Plot*. The reference and compared waveforms are plotted in the graph window. By default, both the waves are shown in the same color. You can change their colors for a better analysis.

© 2023 Cadence Design Systems, Inc.

If the reference and compare analog signals have different time ranges, the time ranges for which only one signal is available are ignored during comparison.



Important

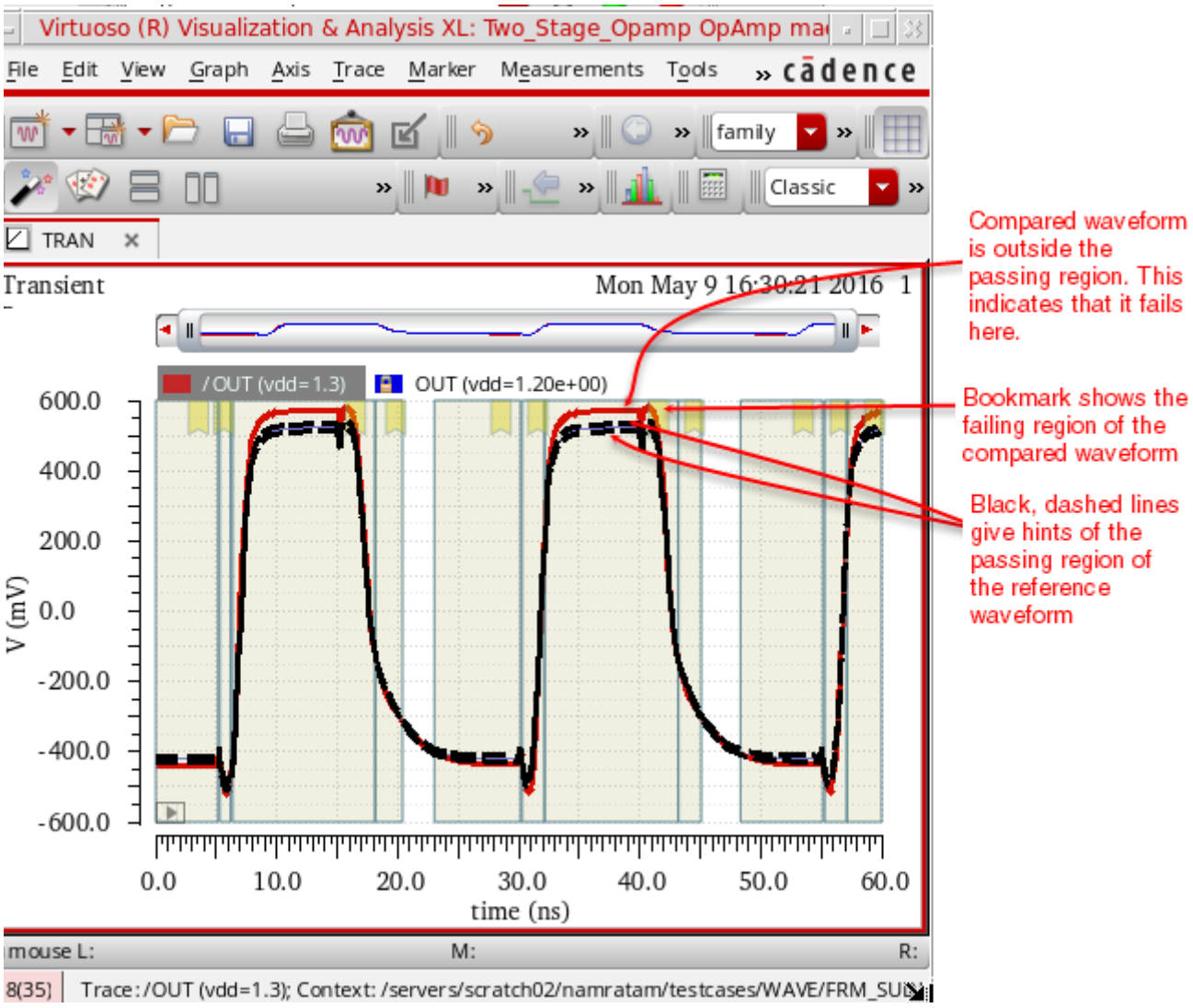
A comparison graph with both the reference and compared waveforms is shown only when the *Spec Markers* check box on the ADE Assembler Plotting/Printing Options form is selected. It is recommended to keep this check box selected while working with waveform specifications.

Click any point on the compared waveform. As shown in the figure given below, two black, dashed lines are shown to indicate the hints of tolerance region of the reference waveform that is calculated using the waveform comparison options. The points for these lines are calculated as:

- The value of the reference waveform at a particular point + y_tolerance
- The value of the reference waveform at a particular point - y_tolerance

© 2023 Cadence Design Systems, Inc.

In addition, region bookmarks are also shown to highlight the failing regions of the compared waveform.



Note: If the failure region falls in between the specific exclude time interval, the error count is increased by 1.

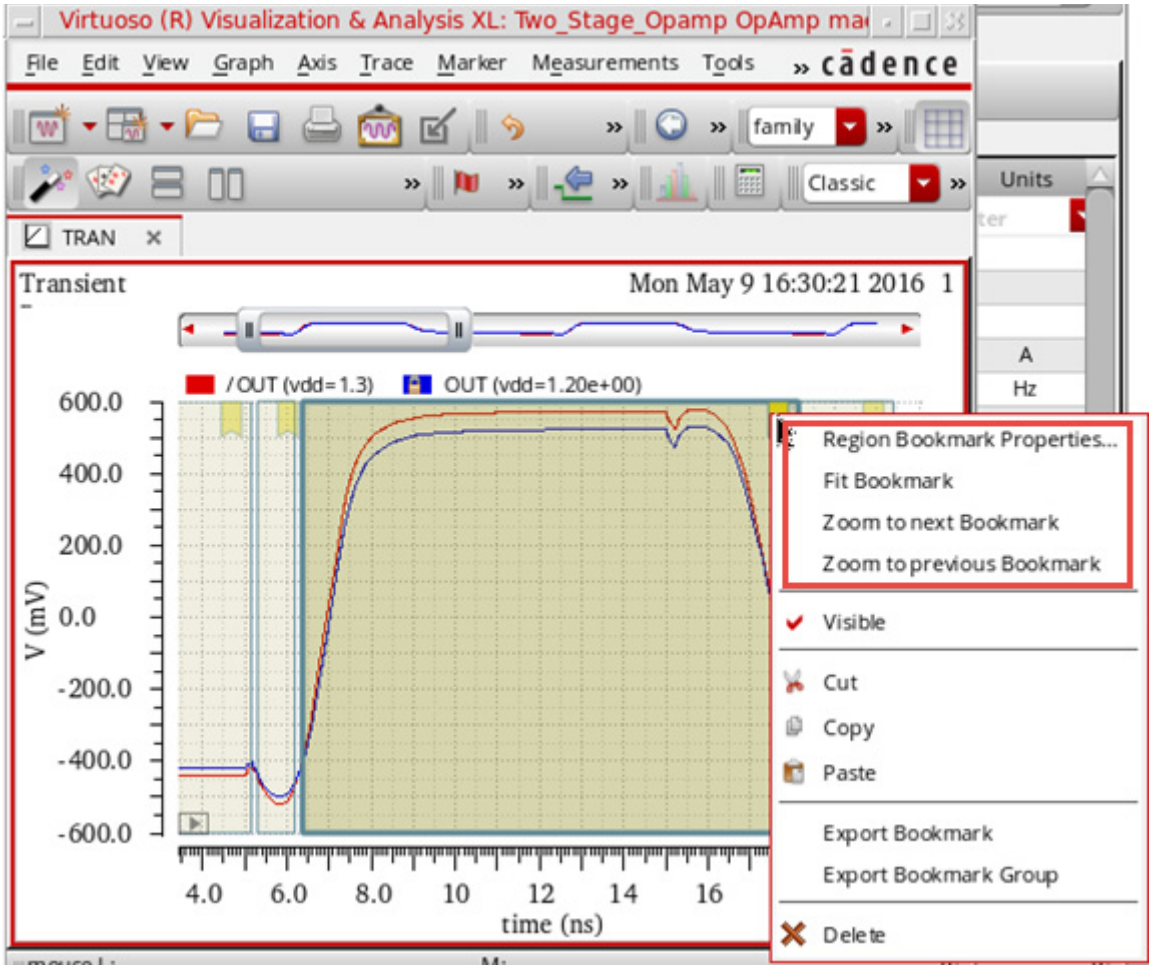
Important

The tolerance region markers provide only a guidance of the region where a result may pass, but they are not necessarily 100% accurate. The pass or fail status of a result depends on the comparison algorithm. The tolerance markers provide a good estimate, but there can be a few corner cases where the marks show different

© 2023 Cadence Design Systems, Inc.

results than the pass or fail status set by the algorithm. For example, it is possible to a see a point that falls outside the tolerance region is marked as pass by the algorithm.

Zoom into a smaller area to view the details either by using the zoom slider on top of the graph or by using the right-click context menu of a region bookmark.



Tip

Tolerance region markers and region bookmarks are the properties of spec markers applied to the compared waveform. By default, these properties are visible only when you select the compared waveform trace on the graph. However, you can change their default visibility mode by setting these properties using the Spec Marker Properties for Specification form or by using the commands in the graph legend of the compared waveform. For more details, refer to Visibility Settings for Specification Markers.

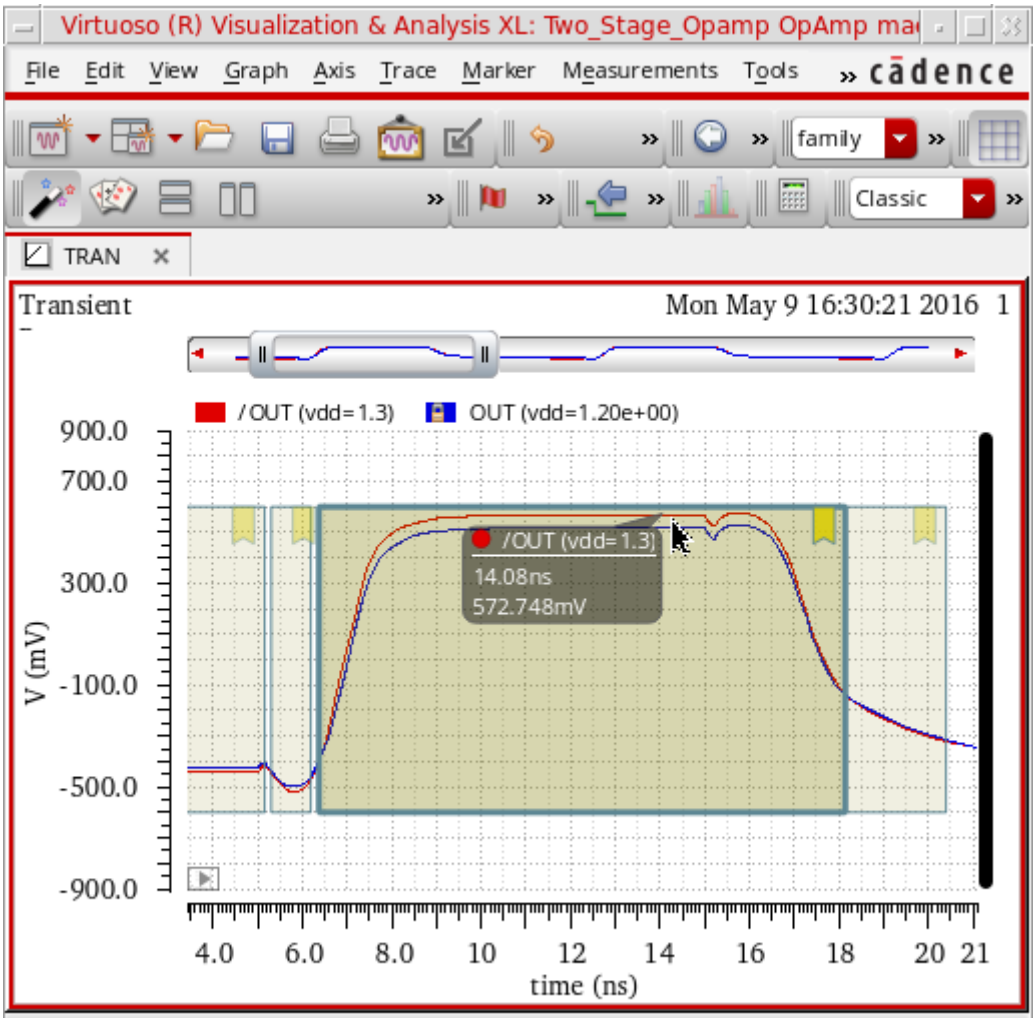
© 2023 Cadence Design Systems, Inc.

The *Fit Bookmark* command in the context menu zooms in the graph to show the failing region of the compared waveform.

Tip

Use the `Zoom to next Bookmark` and `Zoom to previous Bookmark` commands for quick transition to the next or previous failing regions of the compared waveform. You can also use `Shift + N` and `Shift + P` bindkeys.

Move the mouse over the trace to view the values in the tracking cursor.



Based on your analysis, you can make the required changes in your design to improve the results.

© 2023 Cadence Design Systems, Inc.

Important

If you make changes in the tolerance values in the waveform comparison options, re-evaluate the results to view the revised waveforms.

Visibility Settings for Specification Markers

This section describes the properties that you can set to customize the visibility of tolerance markers and region bookmarks on the plots for waveform comparison.

For this:

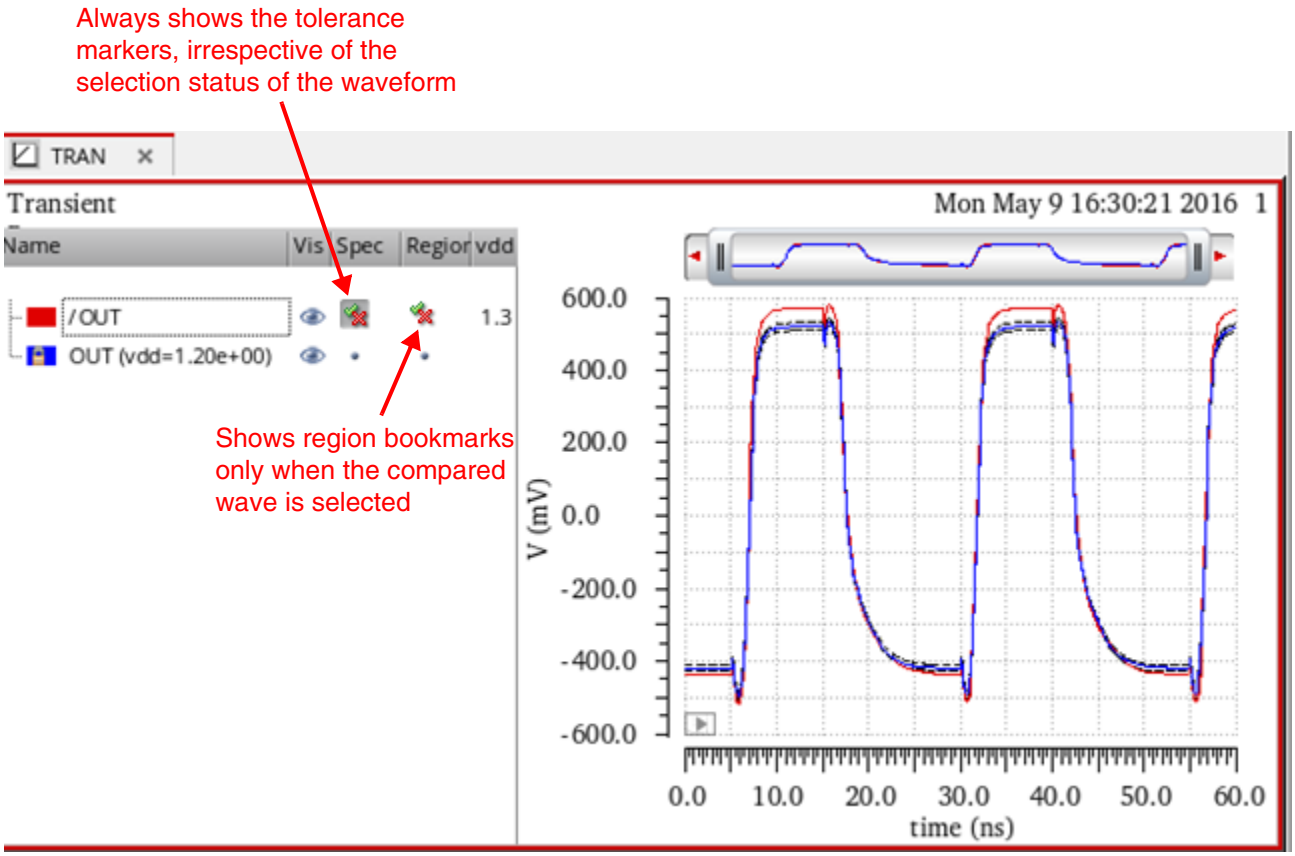
1. Ensure that the graph legend is visible to the left of graph. You can do this by setting the *Legend Position* option on the Graph Properties form to `left`.
2. Set the visibility setting of tolerance markers by changing the icon in the *Spec* column of the legend.
3. Set the visibility setting of region bookmarks by changing the icon in the *Region* column of the legend.

   Possible values that you can set in the *Spec* and *Region* columns in the graph legend are:

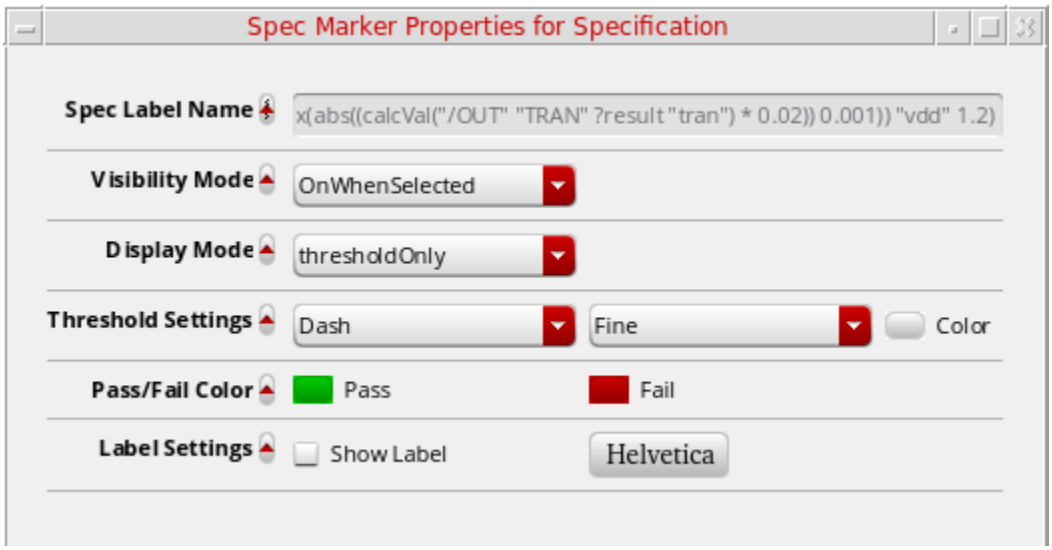
   - `On` ( ): Always show
   - `Off` ( ): Never show
   - `OnWhenSelected` ( ): Shows only when the compared waveform is selected

© 2023 Cadence Design Systems, Inc.

The following figure shows how these properties are set in the graph legend:



You can also set the spec properties on the Spec Marker Properties for Specification form.



© 2023 Cadence Design Systems, Inc.

All the spec marker properties that you can set on this form are:

- *Visibility Mode*: Specifies when to make the spec markers visible. By default, this property is set to `OnWhenSelected`, which implies that the spec markers for a waveform are shown only when the waveform is selected.

  Related environment variable: `viva.specMarker visibilityMode`

- *Display Mode*: Specifies what to display for the pass and fail markers. These markers use the colors specified by the *Pass/Fail Color* property.

  Set this property to:

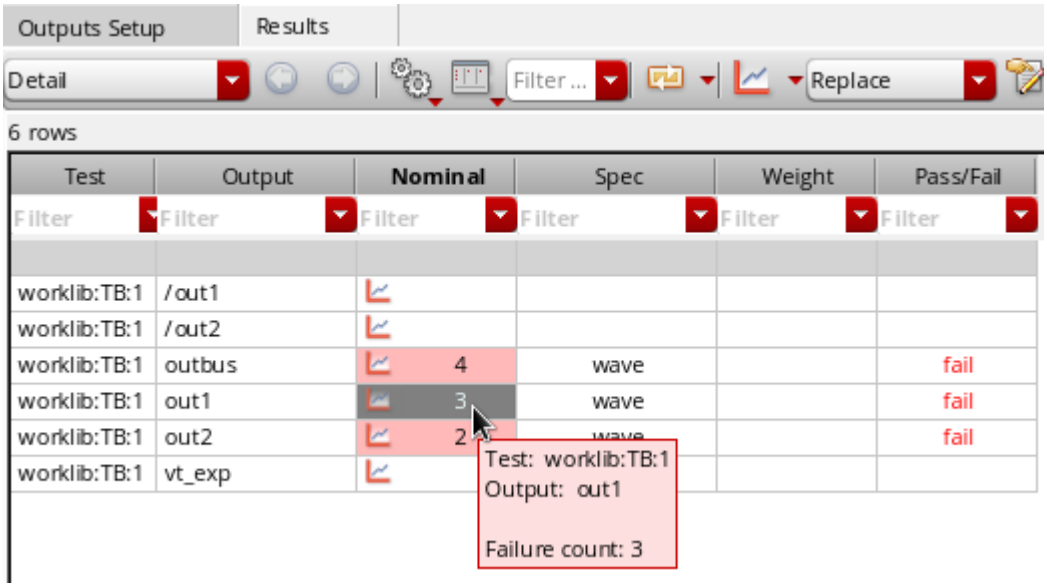
  - `none` to never show the pass and fail markers
  - `pass` to show only the pass markers
  - `fail` to show only the fail markers
  - `both` to always show the pass and fail markers
  - `thresholdOnly` to show only the threshold lines between the pass and fail markers (This is the default value)
  - `pointsOnly` colors the individual points green or red instead of using filled areas for pass/fail. This is exclusively used for histograms and QQplots.

- *Threshold Settings*: Specifies line style and color for the tolerance markers.
- *Pass/Fail Color*: Specifies the colors to be used to highlight the pass and fail marker areas.

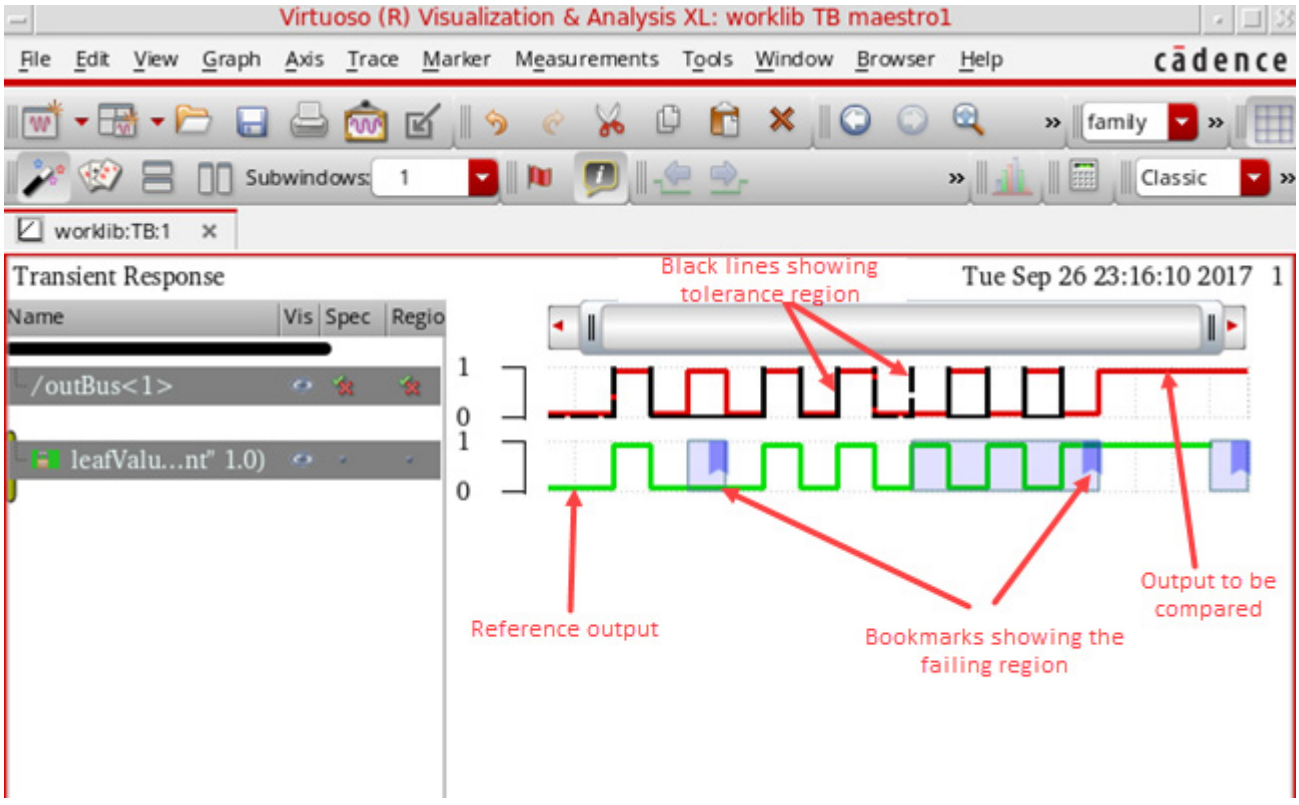
© 2023 Cadence Design Systems, Inc.

Analyzing Waveform Comparison Results for Digital Signals

For logic or digital signals, the tooltip on the Results tab shows only the failure count.

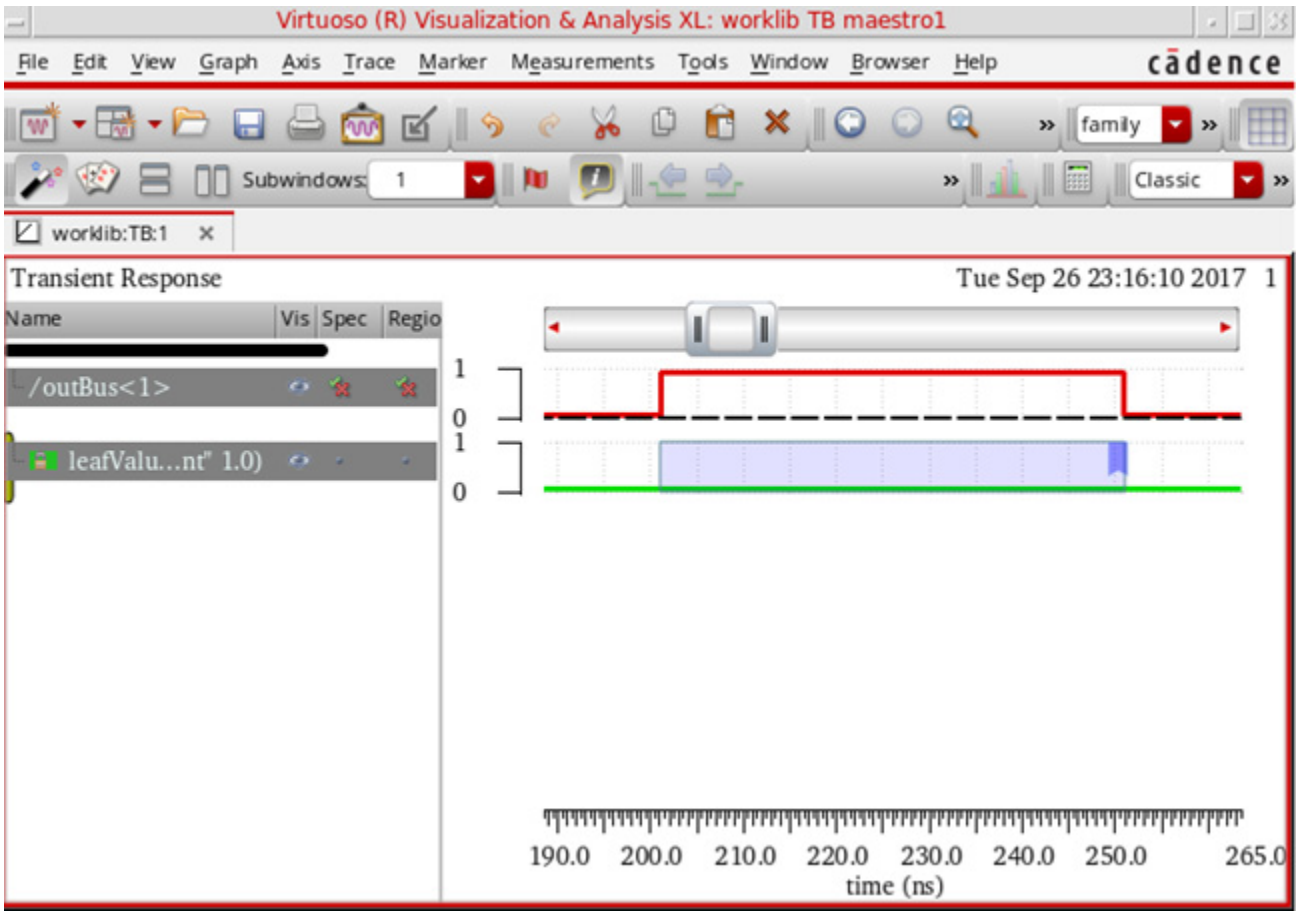


Click the plot icon to plot the graph in Virtuoso Visualization and Analysis XL window.



© 2023 Cadence Design Systems, Inc.

For a clear view of the failing region, use the *Fit Bookmark* command in the context menu of the compared waveform.



Also see: Visibility Settings for Specification Markers

You can now analyze the failing areas and make modifications in the active setup or the waveform comparison settings to resolve the issues.

© 2023 Cadence Design Systems, Inc.

Saving Waveform Comparison Results in a Datasheet

The datasheet saved from ADE Assembler includes the details of wave specifications and resulting values. The error count for a particular result is in the `Failures: x` format, as shown below.

Setup Summary

Library	Two_Stage_Opamp
Cell	OpAmp
View	maestro_wave

Results Summary

Specification details

Result

	Test	Calculation	Expression	Target	Minimum Value	Maximum Value
TRAN						
	TRAN	/OUT	/OUT	Reference: reference: TRAN:/OUT Analysis: tran Algorithm: Asymmetrical Relative Tolerance: 3% Absolute Tolerance: 1m	Failures: 4	Failures: 4
	TRAN	Swing	(value(VT("/OUT") 2.5e-08) - value(VT("/OUT") 1e-09))	> 0.98	6.818m	6.818m
	TRAN	SettlingTime	settlingTime(VT("/OUT") 0 t 2.5e-08 t 10 nil "time")	< 8n	24.84 ns	24.84 ns
	TRAN	RelativeSwingPercent	((Swing / VAR("vdd")) * 100)	> 75	568.2 m%	568.2 m%
	TRAN	PhaseMargin	getData("/phaseMargin" ?result "stb_margin")	> 20	21.83 degree	21.83 degre
	TRAN	/I0/NM0/D	/I0/NM0/D	Reference: reference: TRAN:/I0 /NM0/D Analysis: tran Algorithm: Asymmetrical	Failures: 2	Failures: 2

© 2023 Cadence Design Systems, Inc.

Waveform Compare Algorithms

ADE Assembler uses the following algorithms for waveform comparison:

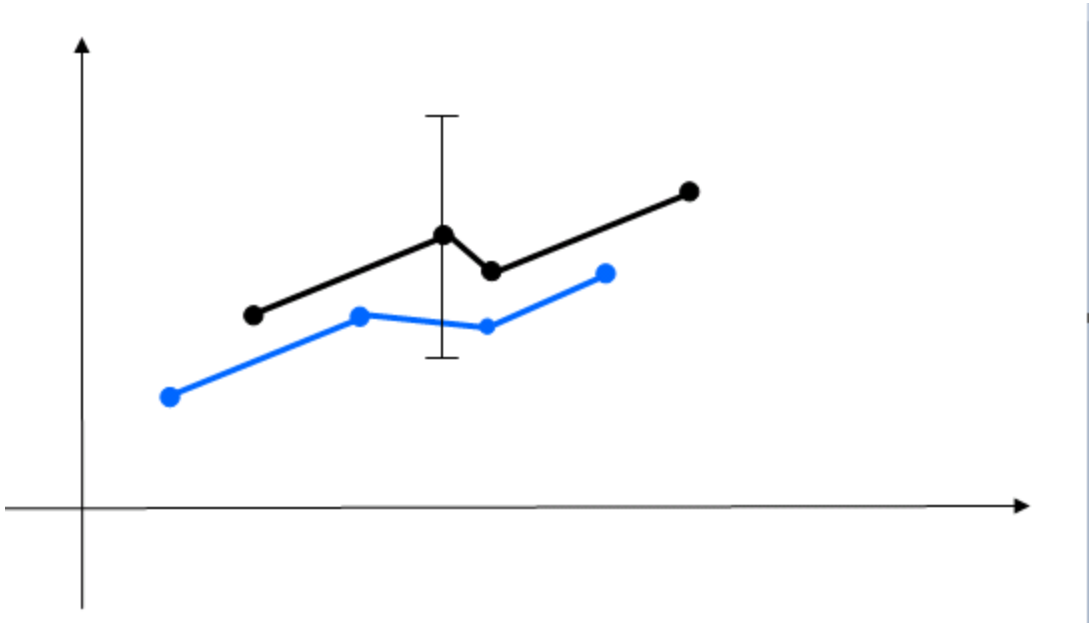
- Asymmetrical Comparison (default)
- Symmetrical (Conservative)
- Symmetrical (Liberal)

It is important to understand these algorithms so that you can make an appropriate choice for the waves you need to compare.

Asymmetrical Comparison

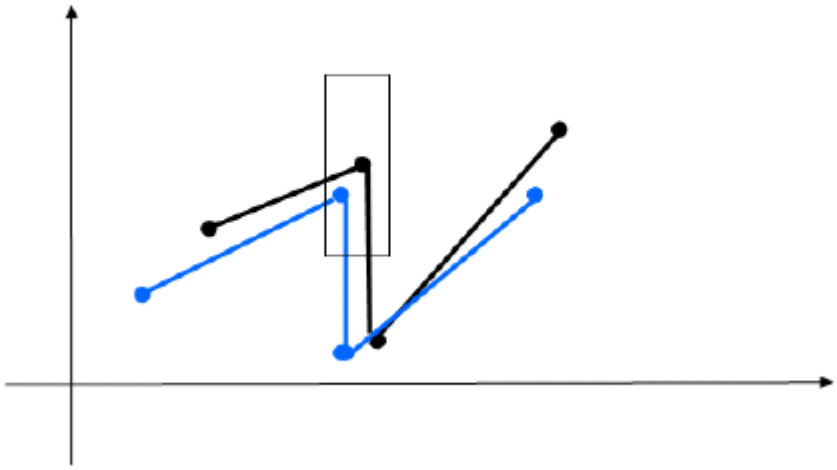
This is the default algorithm used for waveform comparison in ADE Assembler. This shows a better performance as compared to the other algorithms and is sufficient for most of the cases.

For each data point of the reference waveform, an acceptance region is calculated based on the absolute and relative tolerance specified. Note the upper black waveform and the vertical acceptance region for the second data point in the graph in the figure shown below. If the compared waveform is within the region, the point is marked as passed, else it is a failure. This procedure is repeated for all the reference data points.



© 2023 Cadence Design Systems, Inc.

In addition, it uses time tolerance, if provided, for time and value comparison. Time tolerance specifies an absolute time difference that is acceptable between the two analog waveforms. An acceptance region is calculated based on the absolute and relative value tolerance and the provided time tolerance. This defines a rectangular region, as shown in the figure below.



If the compared waveform is within the region, the point is marked as passed, else it fails.

Note: When the time tolerance value is not given (left blank), an acceptance region is calculated based on the absolute and relative tolerance values.

Tip

When asymmetrical algorithm is used, it is recommended to define the more ideal waveform signal out of the two as reference, while the waveform signal with glitches or over- or undershoots should be defined as the compared waveform.

Limitations of Asymmetrical Algorithms

Some of the other limitations of the asymmetrical algorithm are as follows:

- If the compared and references waveforms are swapped, the results might be different.
- The comparison is done only at time points of the reference waveform.
  - If the compare waveform changes significantly between two time points of the reference waveform, the algorithm might not detect problem in this area.

© 2023 Cadence Design Systems, Inc.

- If the time points in a reference waveform are too less as compared to the compare waveform, the remaining parts of the compared waveform are ignored. However, if the compared waveform is shorter, you will get errors for the parts where there is a reference signal but no compared signal.

- The relative tolerance is calculated based on the reference value, if this value is close to zero, the relative tolerance is almost zero too. Therefore, a reasonable value for the absolute tolerance is highly recommended.

Symmetrical (Conservative)

In this algorithm, the comparison of waveforms is done twice—first, the reference waveform is compared with the compared waveform and then, the compared waveform is compared with the reference waveform. An area is marked as failed if any one of the runs fails at the same time.

Note: This comparison process takes more time because the comparison is being done twice.

Symmetrical (Liberal)

Similar to the conservative approach, in this algorithm too, the comparison of waveforms is done twice, but an area is marked as failed if both the runs fail at the same time.

Note: This comparison process takes more time because the comparison is being done twice.

© 2023 Cadence Design Systems, Inc.

Defining Operating Region Specifications

Operating region specifications define expressions that ensure that your devices are operating in a desired region (saturation, triode, and so on).

For more information, see the following topics:

- Setting Up Operating Region Specifications on page 1378
- Saving Operating Region Expressions to a File on page 1399
- Loading Operating Region Expressions from a File on page 1401
- Modifying Operating Region Specifications on page 1395
- Deleting Operating Region Specifications on page 1402
- Disabling and Enabling Operating Region Specifications on page 1402
- Viewing Operating Region Violations on page 1408
- Re-evaluating Operating Region Expressions on page 1411

Copying a Specification

To copy a specification, do the following:

1. In the *Spec* column on the Outputs Setup tab of the Outputs pane, select the specification.
2. Press *Ctrl+C* to copy the specification.
3. Click in the *Spec* column next to the expression for which you want to paste the specification.
4. Press *Ctrl+V* to paste the specification.

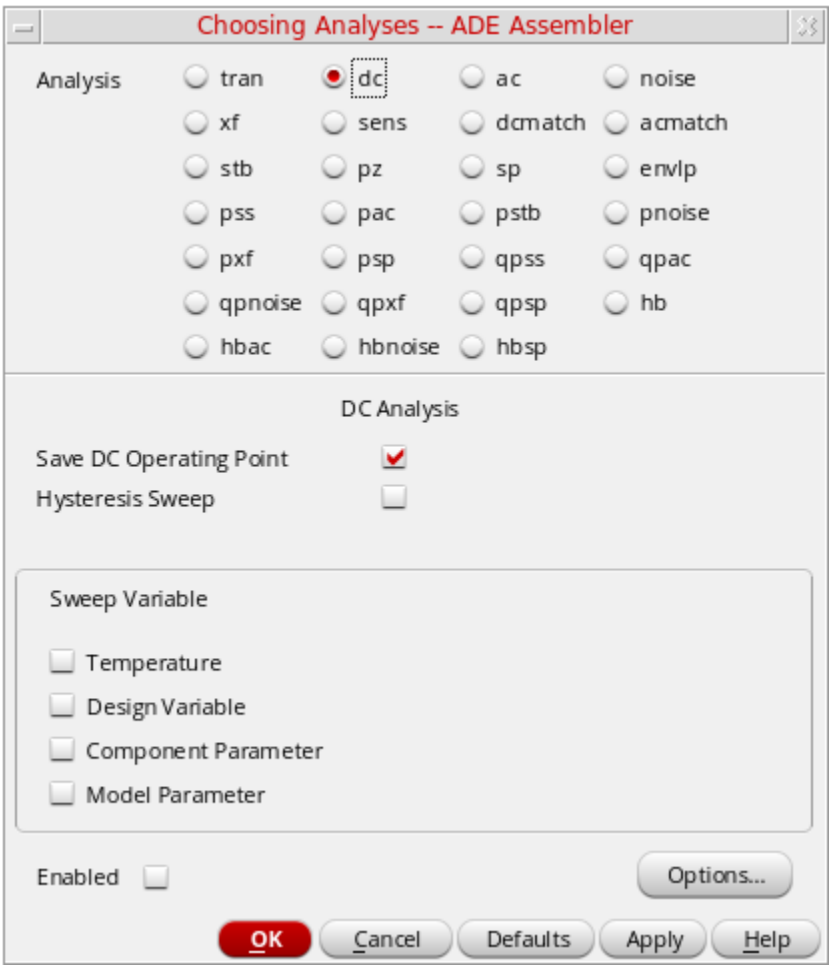
Note: To copy a specification from one test to the other, both the tests should have same the schematic/design.

Setting Up Operating Region Specifications

To set up an operating region specification for a test, do the following:

© 2023 Cadence Design Systems, Inc.

1. Ensure that a DC analysis with option to save DC operating point information is set up for the test as shown below:

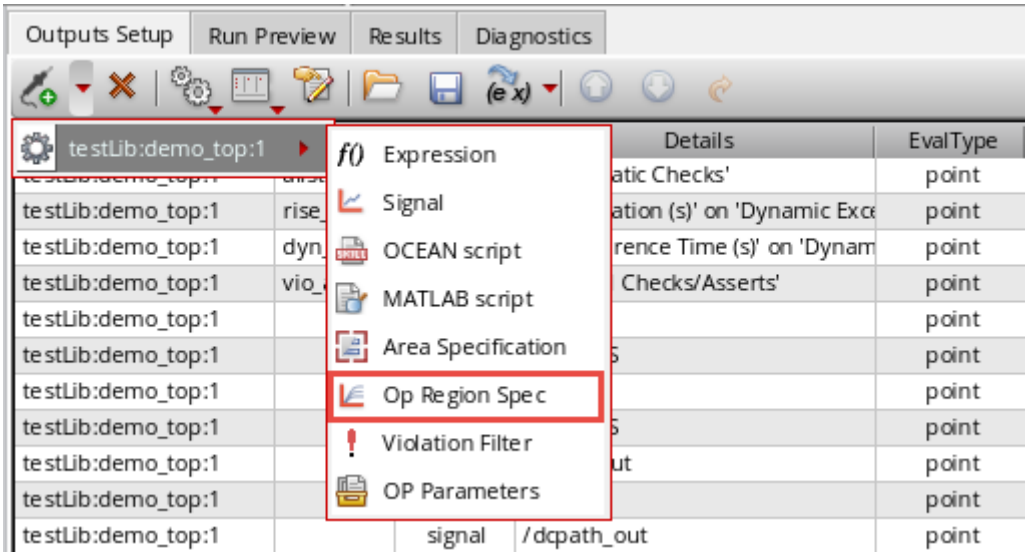


   For more information about setting up a DC analysis, see Adding an Analysis.

2. On the Outputs Setup tab, click the *Add new output* toolbar button, to view the drop-down list of available tests.

© 2023 Cadence Design Systems, Inc.

3. In the drop-down list, select a test and choose *Op Region Spec*.

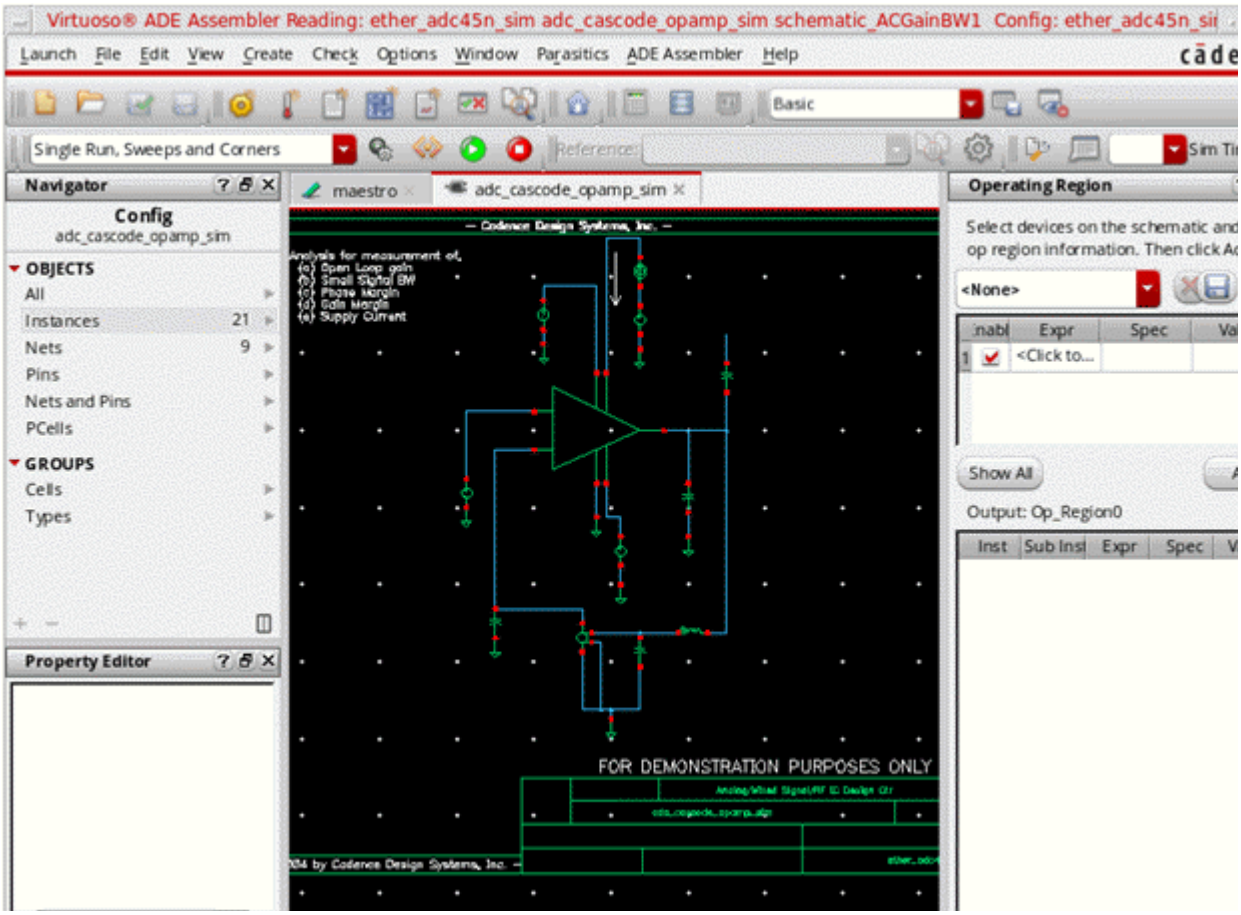


Tip

You can also right-click a test name in the Outputs Setup tab and choose *Add Op Region Spec* to set up an operating region specification for that test.

© 2023 Cadence Design Systems, Inc.

The schematic for the test is displayed with the Operating Region and Navigator assistant panes docked.



4. Specify operating region expressions using the Operating Region assistant pane or the Operating Region Specification form.

   The Operating Region assistant pane enables you to quickly specify expressions while the Operating Region Specification form provides advanced methods for specifying expressions.

   Note: You can specify multiple operating region specifications for a single test.

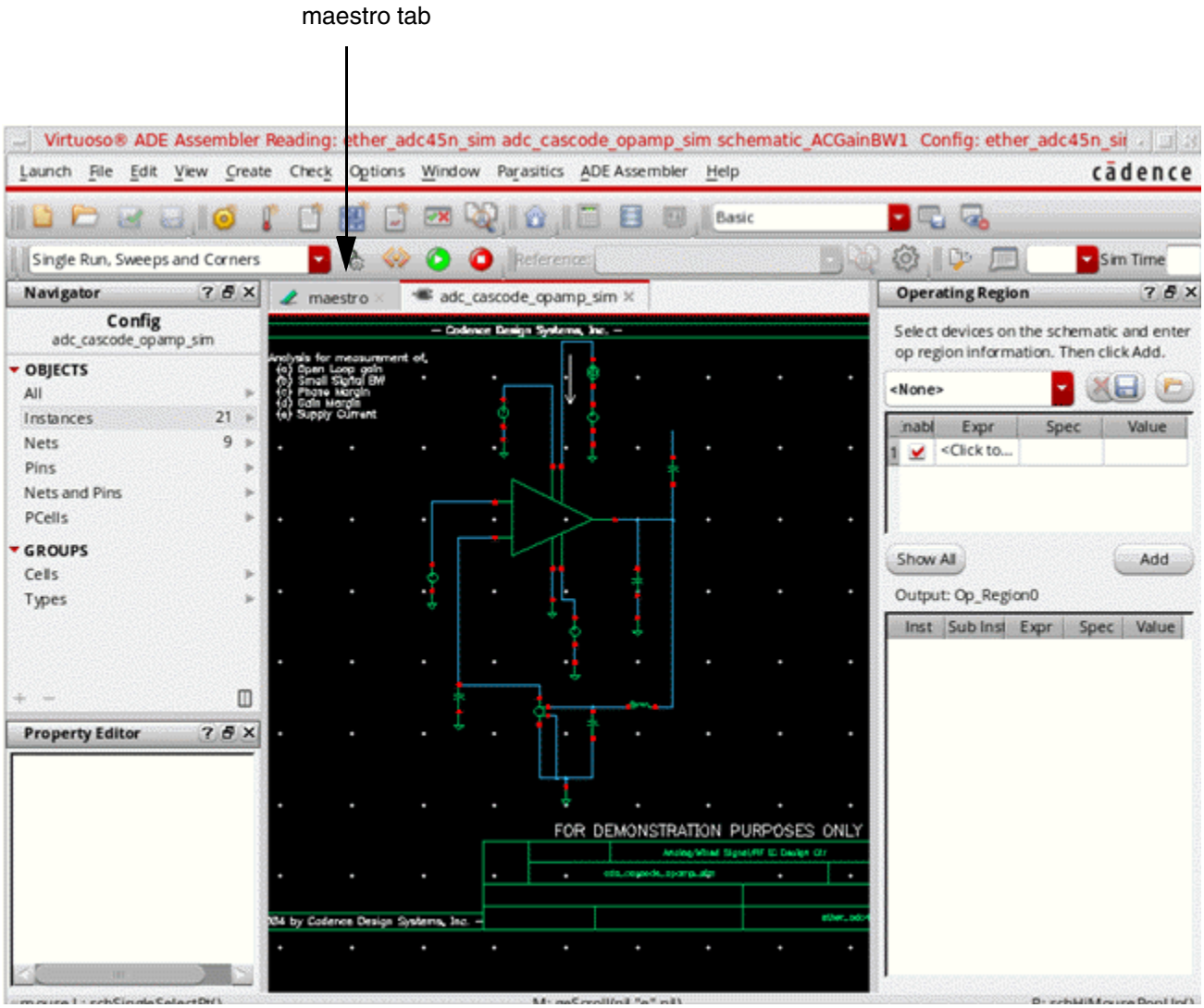
   For more information, see the following topics:

   - Specifying Operating Region Expressions Using the Operating Region Assistant Pane on page 1383
   - Specifying Operating Region Expressions on page 1388
   - Viewing Operating Region Violations on page 1408

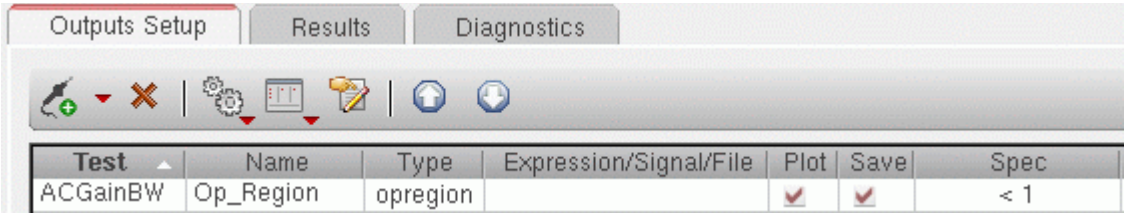
© 2023 Cadence Design Systems, Inc.

- Re-evaluating Operating Region Expressions on page 1411

5. Click the adexl tab when you are done.



The operating region specification for the test is displayed in the Outputs Setup tab as shown below:



© 2023 Cadence Design Systems, Inc.

Specifying Operating Region Expressions Using the Operating Region Assistant Pane

To specify operating region expressions using the Operating Region assistant pane, do the following:

1. On the schematic or in the Navigator assistant pane, select the instance for which you want to specify operating region expressions.

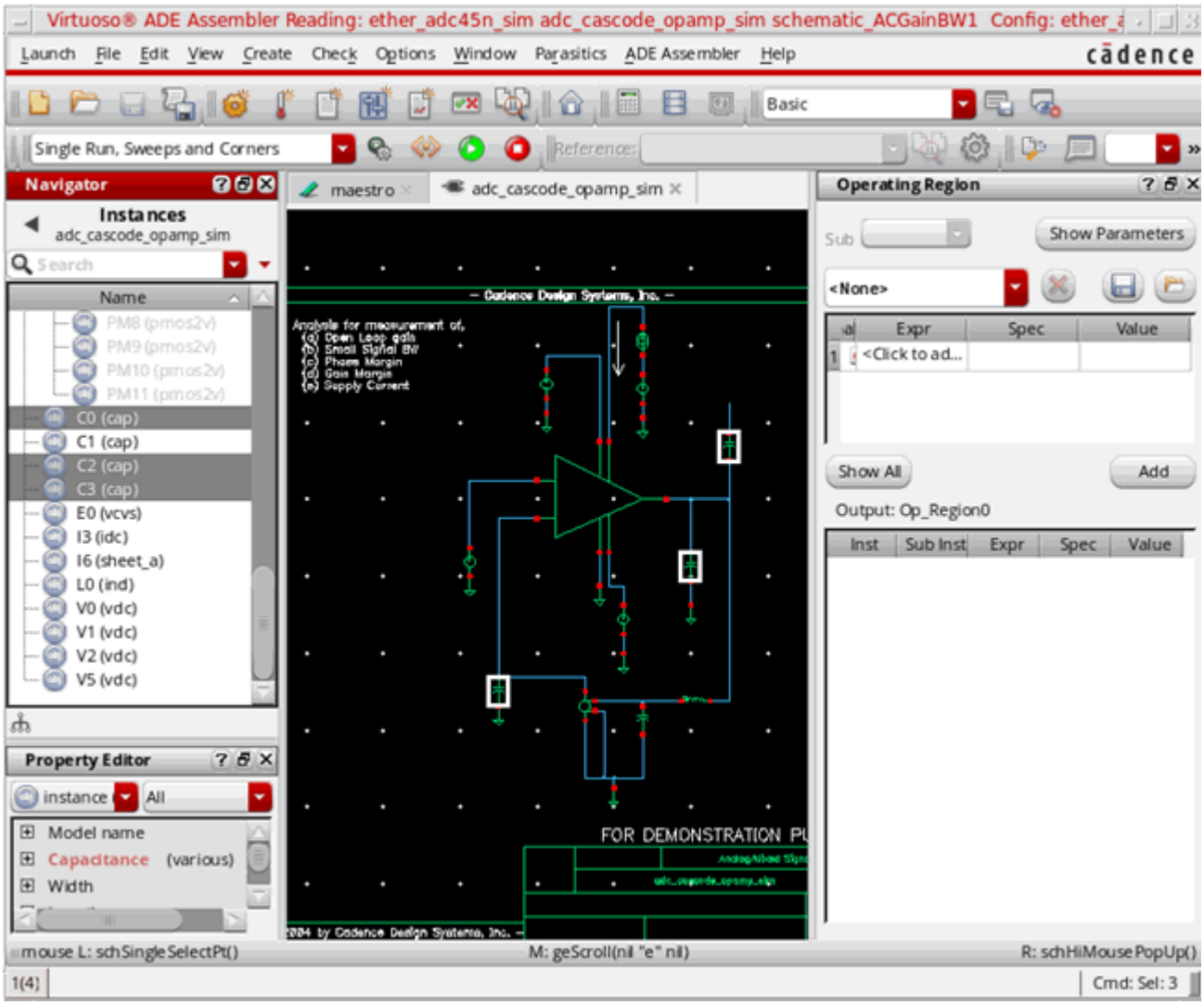
   You can also select more than one instance of the same type to simultaneously specify operating region expressions for all of them.

   - To select more than one instance at a time on the schematic, do one of the following:
     - Hold down the *Shift* key and click instances.
     - Click and drag the mouse over the instances you want to select.

       All the instances that are within the yellow bounding box that appears are included in the selection.
   - To select more than one instance at a time in the Navigator assistant pane, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection) and click the next instance to add it to the selection set.

© 2023 Cadence Design Systems, Inc.

For example, in the following figure, four instances of the `nmos2v` device are selected in the Navigator assistant pane.



2. Specify operating region expressions for the selected instances by doing the following in the Operating Region assistant pane:

   a. (Optional) If the selected instance has subcircuit instances, select the subcircuit instance for which the operating region expression is to be applied from the *SubInst* drop-down list.

   b. In the *Expr* column on the Operating Region assistant, click where it says *<Click to add>* and enter an operating region expression. For example:

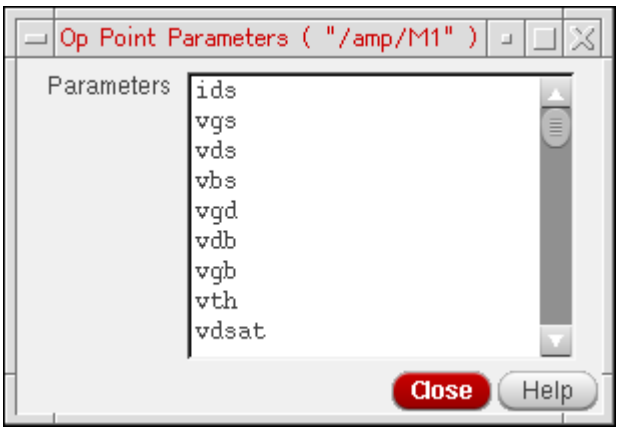
      ```
      vgs - vth
      ```

      Note: To view the list of operating point parameters that you can use in expressions,

© 2023 Cadence Design Systems, Inc.

click the *Show Parameters* button on the Operating Region assistant.

Note: The operating point parameters for the selected instances are displayed in the Op Point Parameters form.



Identify the parameters from this list and create expression in the *Expr* column of the Operating Region assistant.

c. Click in the *Spec* column and select the specification for the expression.

d. Click in the *Value* column and specify the target value for the specification.

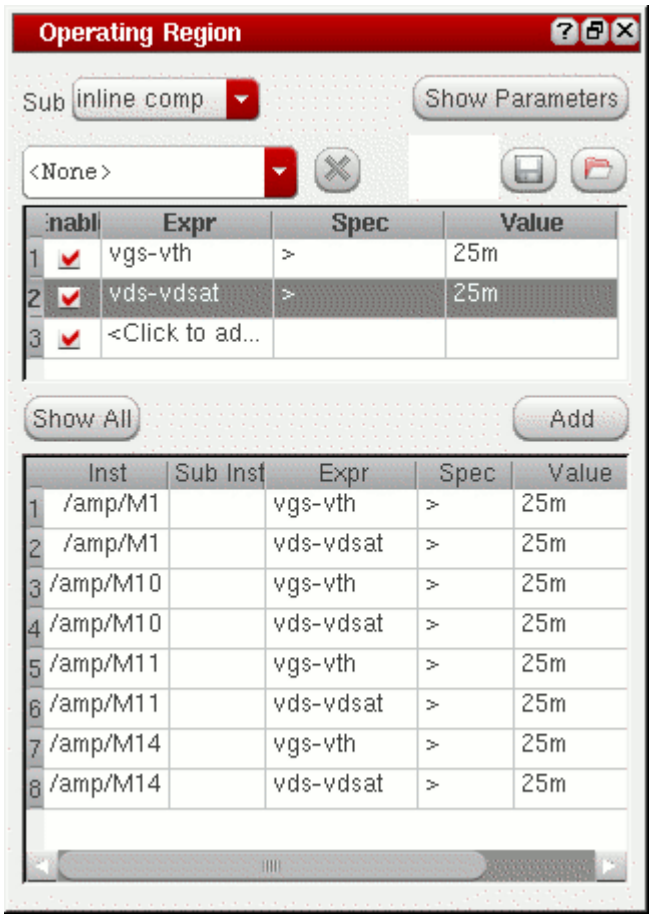
© 2023 Cadence Design Systems, Inc.

For example, in the following figure, two expressions have been specified for the selected instances.

3. By default, the specified expressions are enabled. Ensure that the expressions you want to add for the selected instances are enabled, and the expressions you do not want to add for the selected instances are disabled.
   - To enable an expression, select the check box next to it in the *Enable* column.
   - To disable an expression, deselect the check box next to it in the *Enable* column.
4. Click *Add*.

© 2023 Cadence Design Systems, Inc.

The enabled expressions are added for the selected instances and displayed in the table at the bottom of the Operating Region assistant pane as shown below:



Note: The Operating Region assistant form displays only the operating region expressions specified for the instances that are currently selected on the schematic or in the Navigator assistant pane. However, if no instance is selected, expressions for all the instances are displayed in the Operating Region assistant. Alternatively, you can click the *Show All* button to view all the operating region expressions specified for the test in the Operating Region Specification form. For more information about using the Operating Region Specification form, see Modifying Operating Region Specifications on page 1395.

See also:

- Saving Operating Region Expressions to a File on page 1399
- Loading Operating Region Expressions from a File on page 1401
- Copying and Pasting Operating Region Expressions on page 1398
- Deleting Operating Region Expressions on page 1398

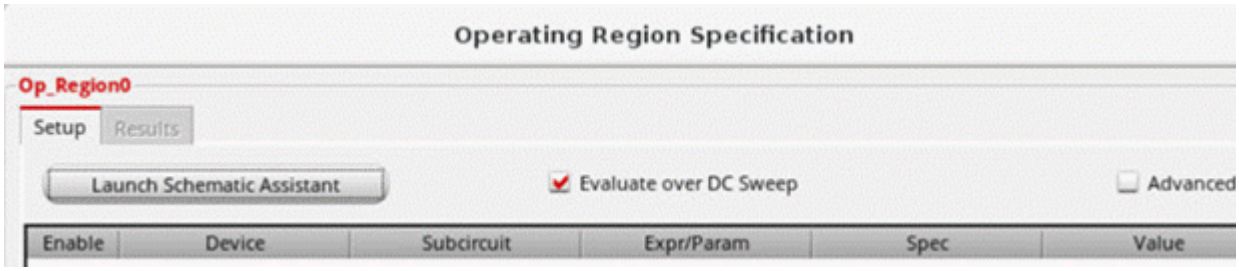
© 2023 Cadence Design Systems, Inc.

Specifying Operating Region Expressions

You can use the advanced view of the Operating Region Specification form to specify operating region expressions. To do this:

1. Click *Show All* on the Operating Region assistant pane.

   The Operating Region Specification form appears.



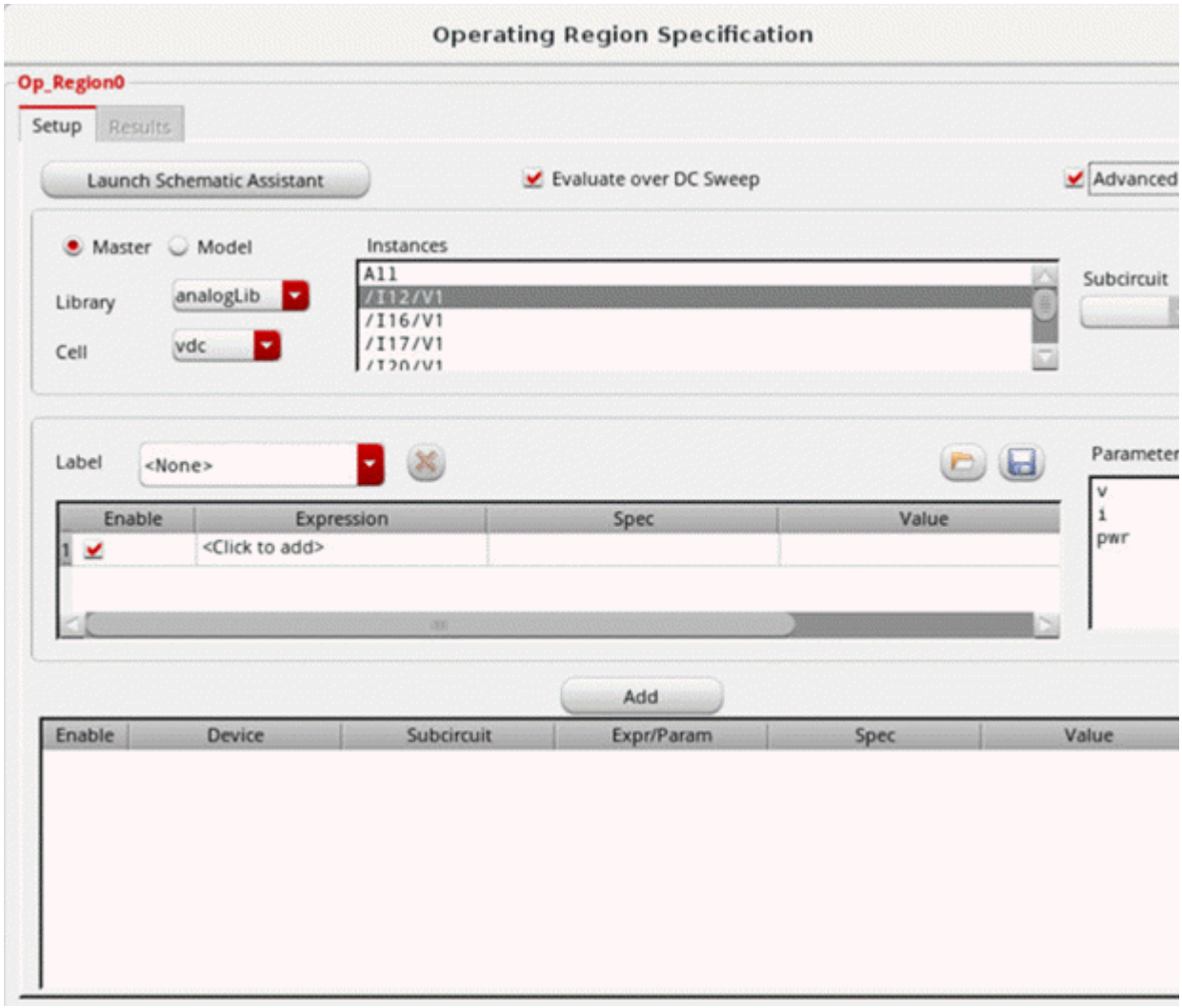
2. Select *Evaluate over DC Sweep* to evaluate the specification over the DC sweep specified through the Choosing Analyses form for DC analysis.

   Selecting this check box adds the option `enable_dcsweep_op_info=yes` to the netlist.

   Note: This check box is available from Spectre 20.1 ISR15 and Spectre 21.1. ISR6.

© 2023 Cadence Design Systems, Inc.

3. Select the *Advanced* check box to display the advanced view of the form.



© 2023 Cadence Design Systems, Inc.

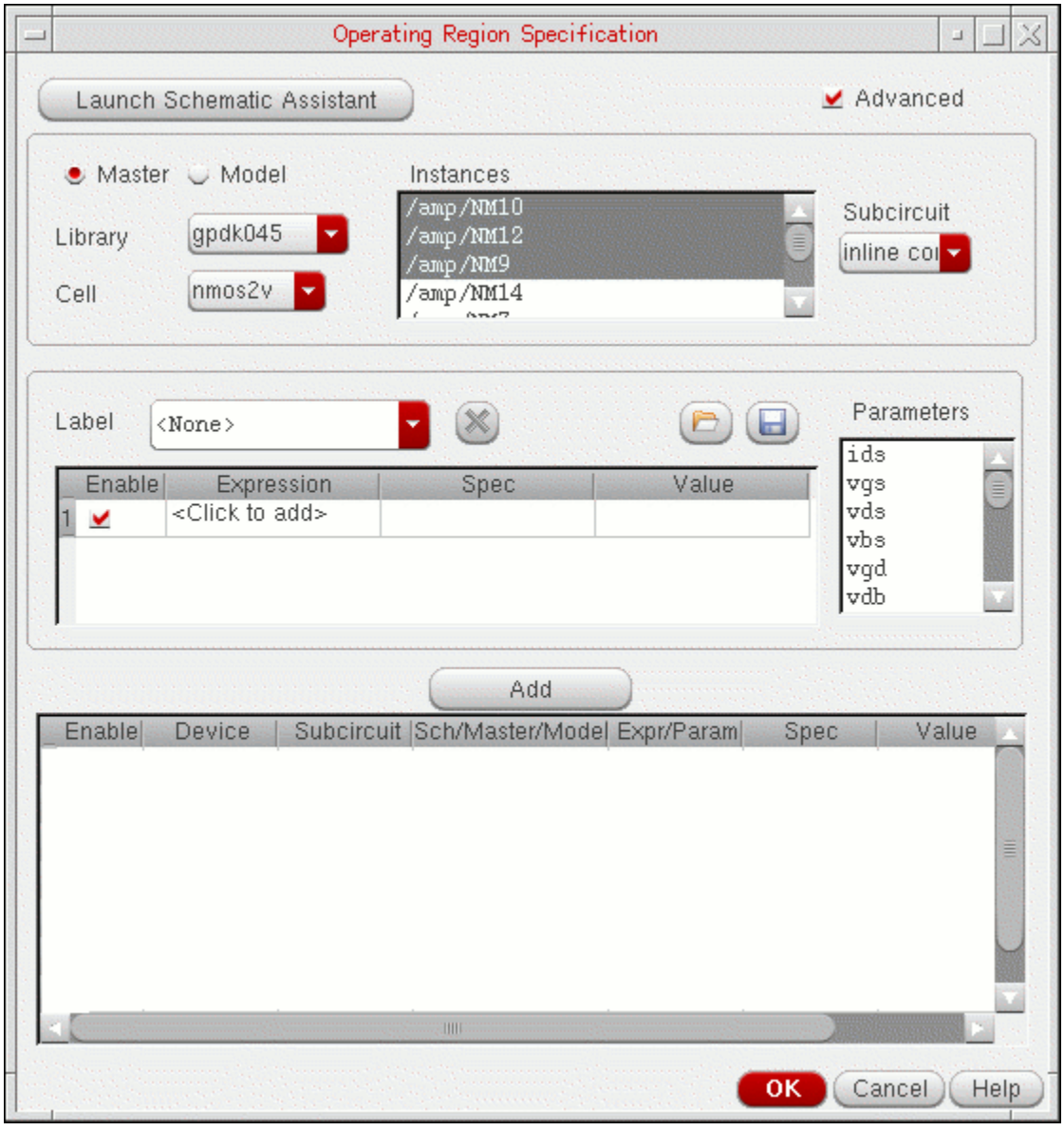
4. Select the instances for which you want to add operating region expressions by doing one of the following:

Select	To
Master	Select instances based on the library in which they exist. Then, do the following to select instances: 1. Use the *Library* and *Cell* drop-down lists to select the library and cell in which cellviews for instances in the design for the test exist. All the instances of the cellviews in the design for the test are displayed in the *Instances* list box. 2. In the *Instances* list box, select the instances for which you want to add operating region expressions. To select more than one instance at a time, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection) and click the next instance to add it to the selection set. 3. (Optional) If the selected instance has subcircuit instances, select the subcircuit instance for which the operating region expression is to be applied from the *Subcircuit* drop-down list.
Model	Select instances based on the models specified for the instances. Then, do the following to select instances: 1. From the *Model* drop-down list, select a model. All the instances in the design for the test for which the model is specified are displayed in the *Instances* list box. 2. In the *Instances* list box, do one of the following: ❑ Select *All* to add operating region expressions to all instances for which the model is specified. ❑ Select the specific instances for which you want to add operating region expressions. To select more than one instance at a time, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection) and click the next instance to add it to the selection set.

© 2023 Cadence Design Systems, Inc.

Tip

The list of operating point parameters that you can use in expressions for the selected instances are displayed in the *Parameters* field.



5. Specify operating region expressions for the selected instances by doing the following:

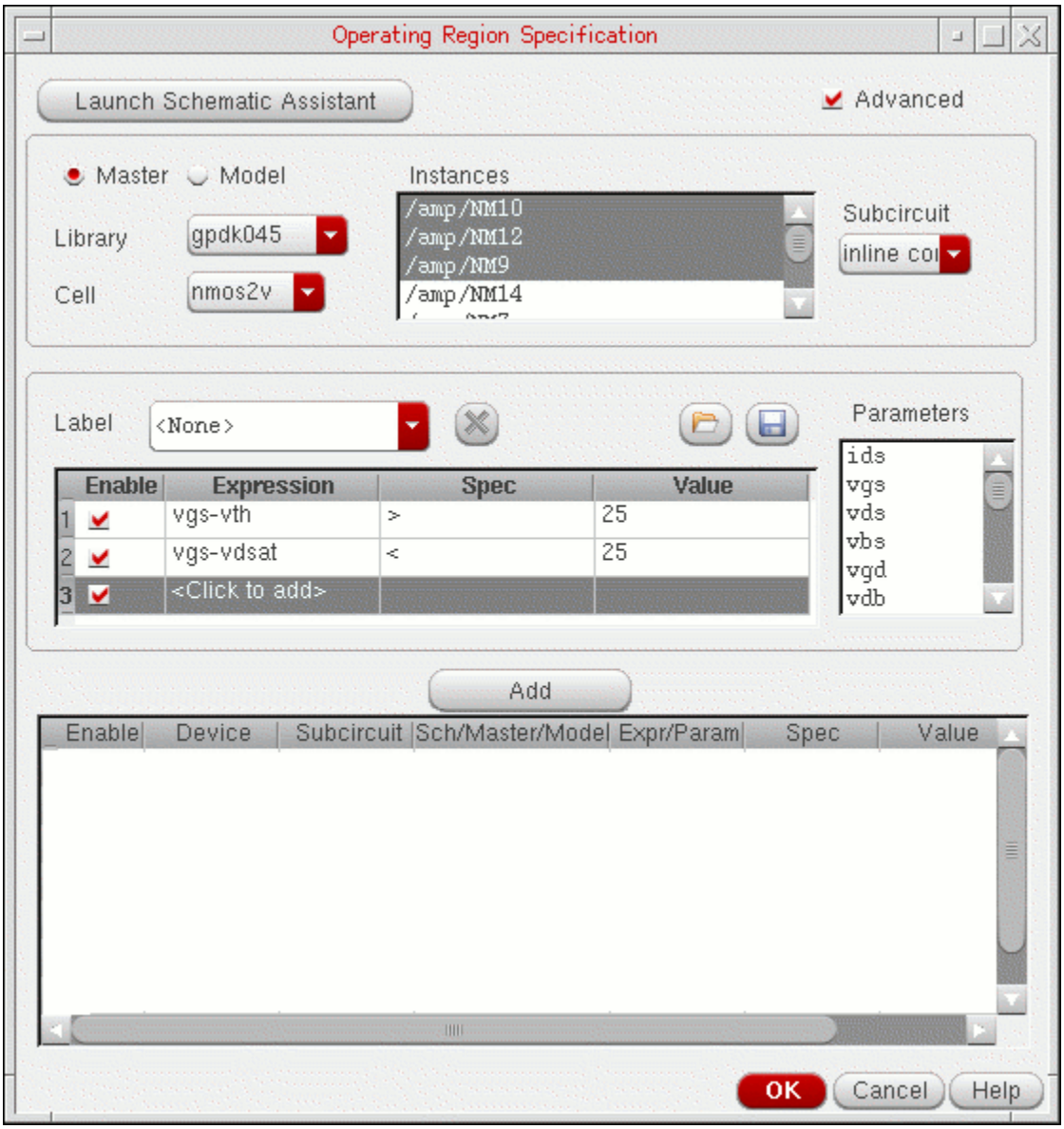
© 2023 Cadence Design Systems, Inc.

a. In the *Expression* column, click where it says *<Click to add>* and enter an operating region expression.

b. Click in the *Spec* column and select the specification for the expression.

c. Click in the *Value* column and specify the target value for the specification.

For example, in the following figure, two expressions have been specified for the selected instances.



© 2023 Cadence Design Systems, Inc.

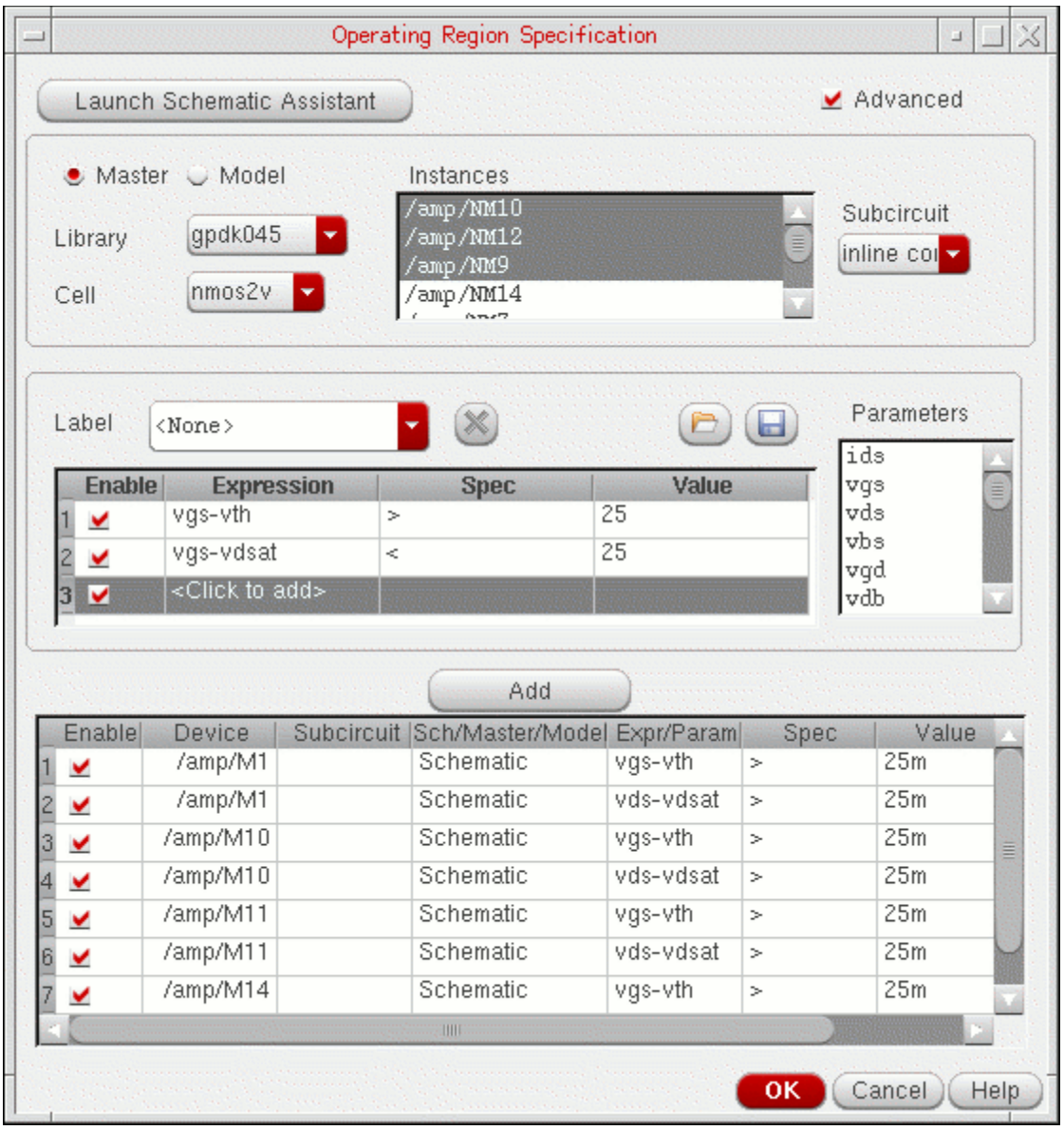
6. By default, the specified expressions are enabled. Ensure that the expressions you want to add for the selected instances are enabled, and the expressions you don't want to add for the selected instances are disabled.

   - To enable an expression, select the check box next to it in the *Enable* column.
   - To disable an expression, deselect the check box next to it in the *Enable* column.

7. Click *Add*.

   The enabled expressions are added for the selected instances and displayed in the table at the bottom of the Operating Region Specification form as shown below.

© 2023 Cadence Design Systems, Inc.



See also:

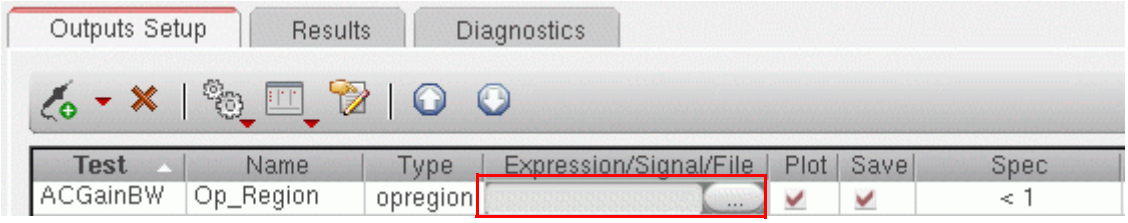
- Saving Operating Region Expressions to a File on page 1399
- Loading Operating Region Expressions from a File on page 1401
- Disabling and Enabling Operating Region Expressions on page 1398
- Copying and Pasting Operating Region Expressions on page 1398
- Deleting Operating Region Expressions on page 1398

© 2023 Cadence Design Systems, Inc.

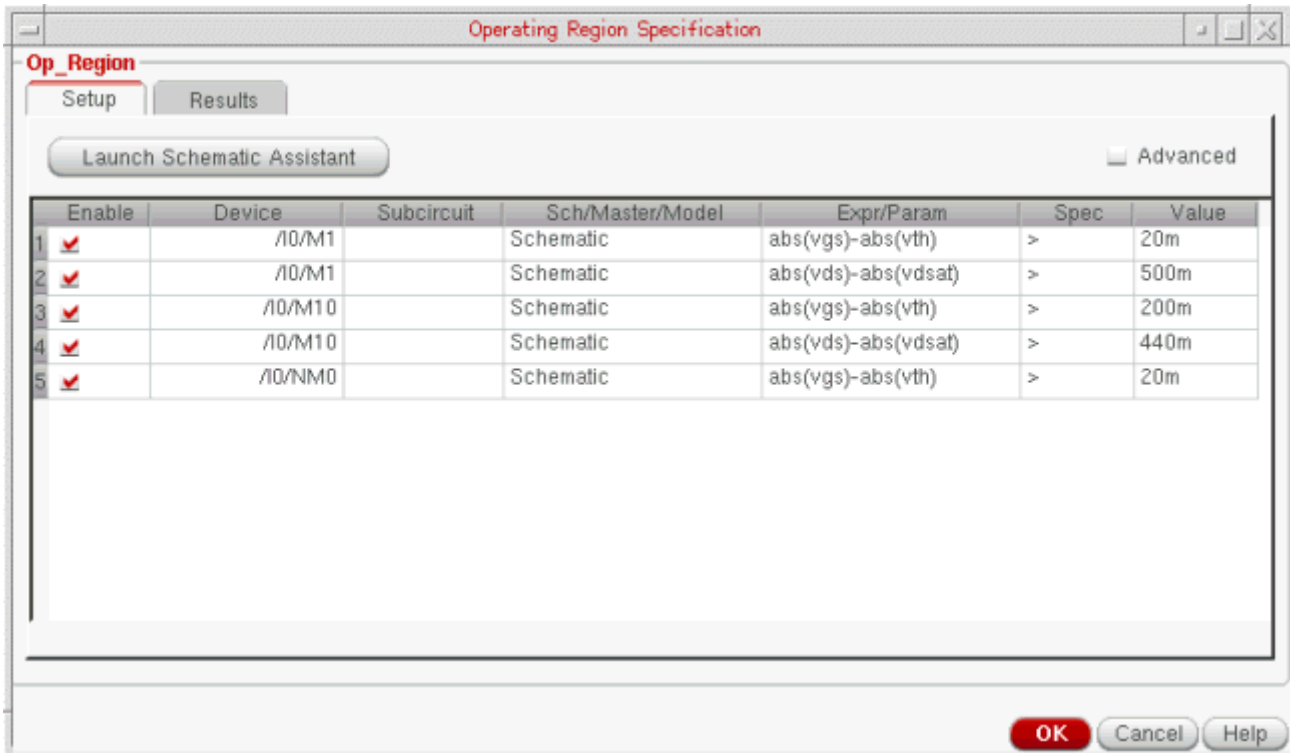
Modifying Operating Region Specifications

To modify an operating region specification, do the following:

1. In the Outputs Setup tab, double-click the *Details* field in the row for the operating region specification. An ellipse button appears in this field. Click the ellipse button.



   The Operating Region Specification form appears displaying the operating region expressions specified for the test.



© 2023 Cadence Design Systems, Inc.

You can use this form to perform the following tasks:

Task	Description
View operating region expressions	
Modify operating region expressions	To modify an expression, double-click in the *Expr/Param*, *Spec* or *Value* columns and make the required changes. If you need to use a common expression for multiple parameters, you can modify expressions for their rows together. For more details, see Modifying Multiple Expressions Together on page 1397.
Copy and paste operating region expressions	For more information, see Copying and Pasting Operating Region Expressions on page 1398
Delete operating region expressions	For more information, see Deleting Operating Region Expressions on page 1398
Enable or disable operating region expressions	For more information, see Disabling and Enabling Operating Region Expressions on page 1398
Click the *Launch Schematic Assistant* to open the schematic for the test with the Operating Region and Navigator assistant panes docked. You can use the Operating Region assistant pane to quickly add or modify operating region expressions for instances you select on the schematic or in the Navigator assistant pane. **Note:** The Operating Region assistant pane displays only the operating region expressions for the instances that are currently selected on the schematic or in the Navigator assistant pane.	For more information, see Specifying Operating Region Expressions Using the Operating Region Assistant Pane on page 1383

© 2023 Cadence Design Systems, Inc.

Task	Description
Click the *Advanced* check box to access a detailed user interface for adding or modifying operating region expressions	For more information, see Specifying Operating Region Expressions on page 1388

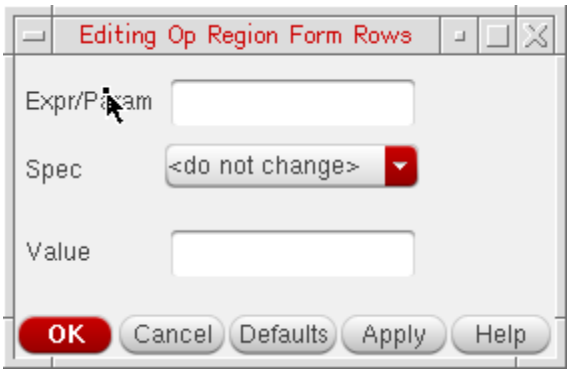
2. Modify the operating region expressions as required.
3. Click *OK*.

Modifying Multiple Expressions Together

To modify multiple operating region expressions together, do the following:

1. Hold down the *Ctrl* key and select more than one expressions.
2. Right-click and choose *Edit Selected Rows*.

   The Editing Op Region Form Rows form appears.



3. Enter an expression in the *Expr/Param* field.
4. Enter a spec value in the *Value* field.
5. Click *OK*.

The expression that you specified in this form is updated in the *Expr/Param* column of all the selected rows in the Operating Region Specifications form.

© 2023 Cadence Design Systems, Inc.

Copying and Pasting Operating Region Expressions

To copy an operating region expression, do the following:

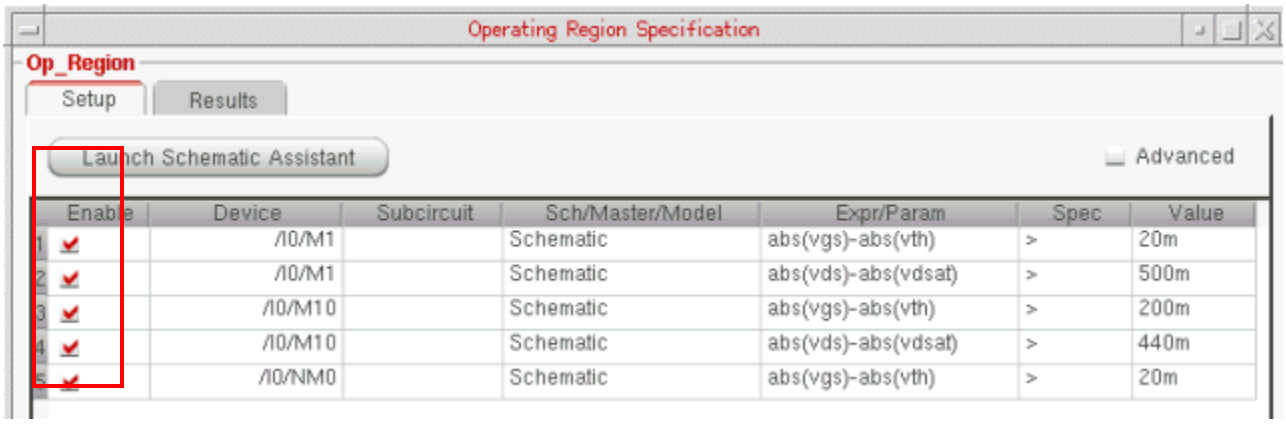
➡ Right-click the row for the expression you want to copy and select *Copy Expression*.

To paste an operating region expression, do the following:

➡ Right-click the row where you want to paste the expression and select *Paste*.

Disabling and Enabling Operating Region Expressions

The *Enable* column in the Operating Region Specification form indicates whether an operating region expression is enabled or disabled. By default, the operating region expressions you add are automatically enabled.



To disable an operating region expression, do the following:

➤ Deselect the check box next to the expression in the *Enable* column.

To enable an operating region expression, do the following:

➤ Select the check box next to the expression in the *Enable* column.

Deleting Operating Region Expressions

To delete an operating region expression, do the following:

➡ Right-click the row for the expression and choose *Delete*.

© 2023 Cadence Design Systems, Inc.

Saving Operating Region Expressions to a File

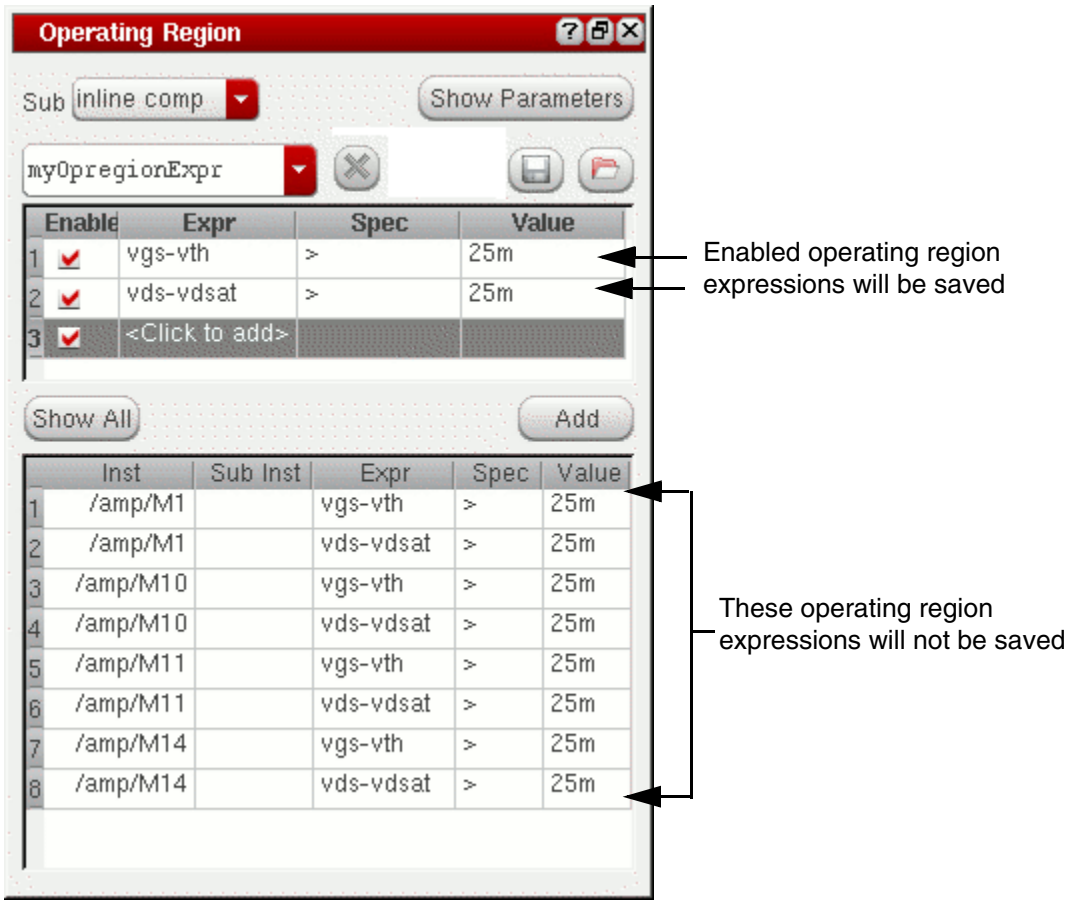
You can save the operating region expressions specified in the Operating Region assistant pane or the detailed view of the Operating Region Specification form to a file. You can then load the file to quickly add operating region expressions. For more information about loading operating region expressions, see Loading Operating Region Expressions from a File on page 1401.

To save operating region expressions to a file, do the following:

1. On the Operating Region assistant pane or the detailed view of the Operating Region Specification form, in the *Label* field, type a name for the specified set of operating region expressions.

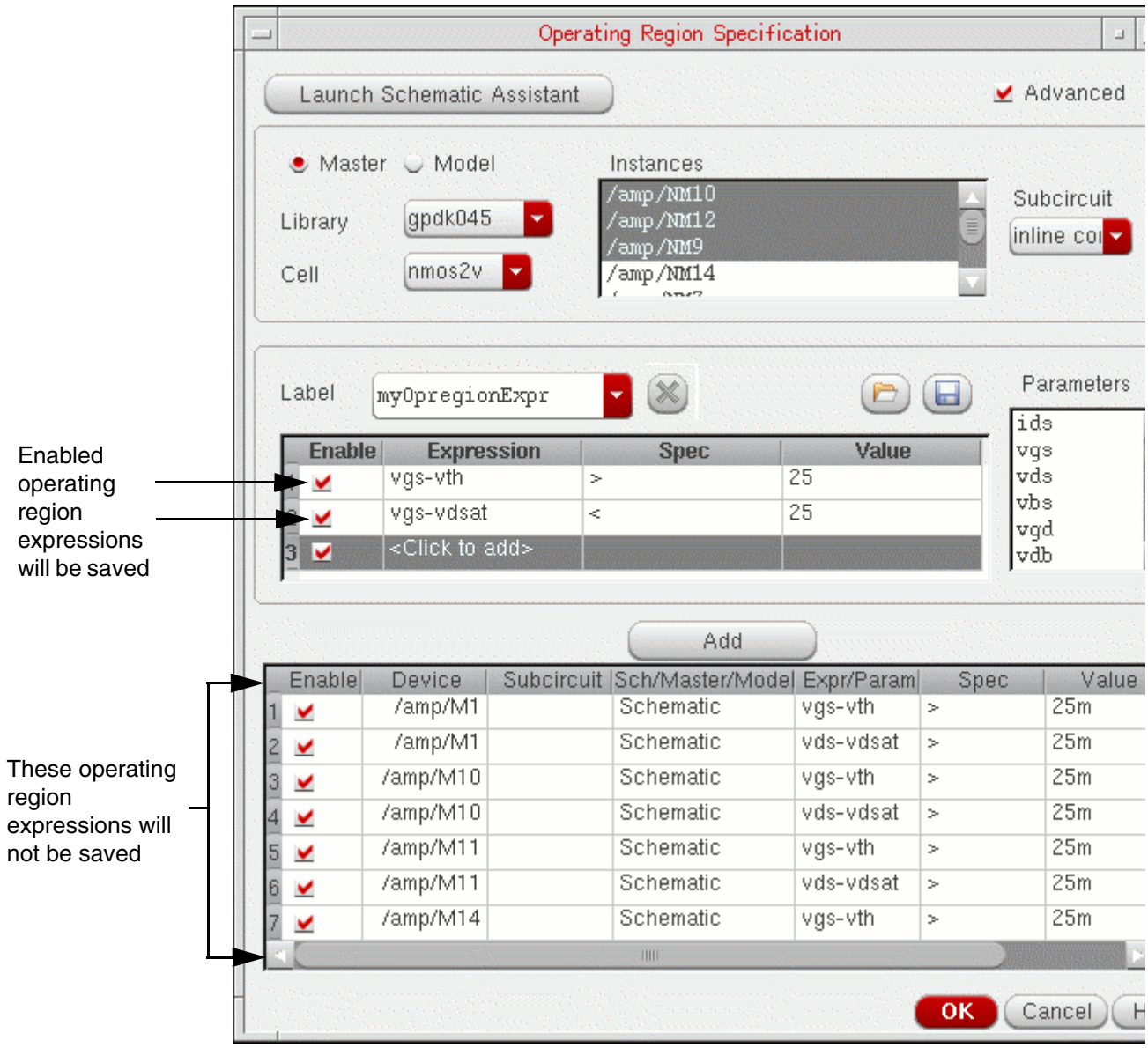
   For example, in the following figure, the two expressions specified in the Operating Region assistant pane will be saved with the name `myOpRegionExpr`.

Figure 22-1 Saving Expressions in Operating Region Assistant Pane



© 2023 Cadence Design Systems, Inc.

Figure 22-2 Saving Expressions in Operating Region Specification Form



2. Ensure that the expressions you want to save are enabled, and the expressions you don't want to save are disabled.

   - To enable an expression, select the check box next to it in the *Enable* column.
   - To disable an expression, deselect the check box next to it in the *Enable* column.

3. Click the *Save* button.

   The Select Operating Region Expression File form appears.

© 2023 Cadence Design Systems, Inc.

4. Specify the name and location of the file and click *Save*.

   The enabled expressions are saved in the file.

Important

If you specify name of an existing file, the tool prompts you to check if you want to replace the existing file or specify another file name. The tool does not append the expressions to the existing file.

Tip

You can concatenate the expressions specified in multiple files and save in a common file. In this case, when the file is loaded, all expressions saved with a common label name are displayed together.

Loading Operating Region Expressions from a File

You can load operating region expressions from a file into the Operating Region assistant pane and the detailed view of the Operating Region Specification form.

To load operating region expressions from a file, do the following:

1. Click the *Load* button in the Operating Region assistant pane or the detailed view of the Operating Region Specification form.

   The Select Operating Region Expression File form appears.

2. Select the file and click *Open*.

   The expressions in the file are displayed. The name specified for the set of expressions in the file is also displayed in the *Label* drop-down list.

Tip

The names specified for the sets of expressions in the files loaded during the current ADE Assembler session are displayed in the *Label* drop-down list. Therefore, during the current ADE Assembler session, you can select a name from the *Label* drop-down list to add expressions, instead of loading the corresponding file again.

© 2023 Cadence Design Systems, Inc.

Tip

Expressions saved in a tab- or comma-separated file can also be loaded in the Operating Region assistant pane or the Operating Region Specification form. In this case, if the expressions are not associated with a label name, the *Label* list displays `<none>`.

Deleting Operating Region Specifications

To delete an existing operating region specification, do the following:

- In the Outputs Setup tab, right-click the row for the operating region specification and choose *Delete Output*.
- In the Operating Region assistant pane or the detailed view of the Operating Region Specification form, select a label from the *Label* list and click *Delete*. All the expressions saved with the selected label name are deleted.

Disabling and Enabling Operating Region Specifications

To disable an operating region specification, do the following:

- In the *Plot* column on the Outputs Setup tab, deselect the check box next to the specification.
- Right-click an operating region specification and choose *Disable Plot*.

To enable an operating region specification, do the following:

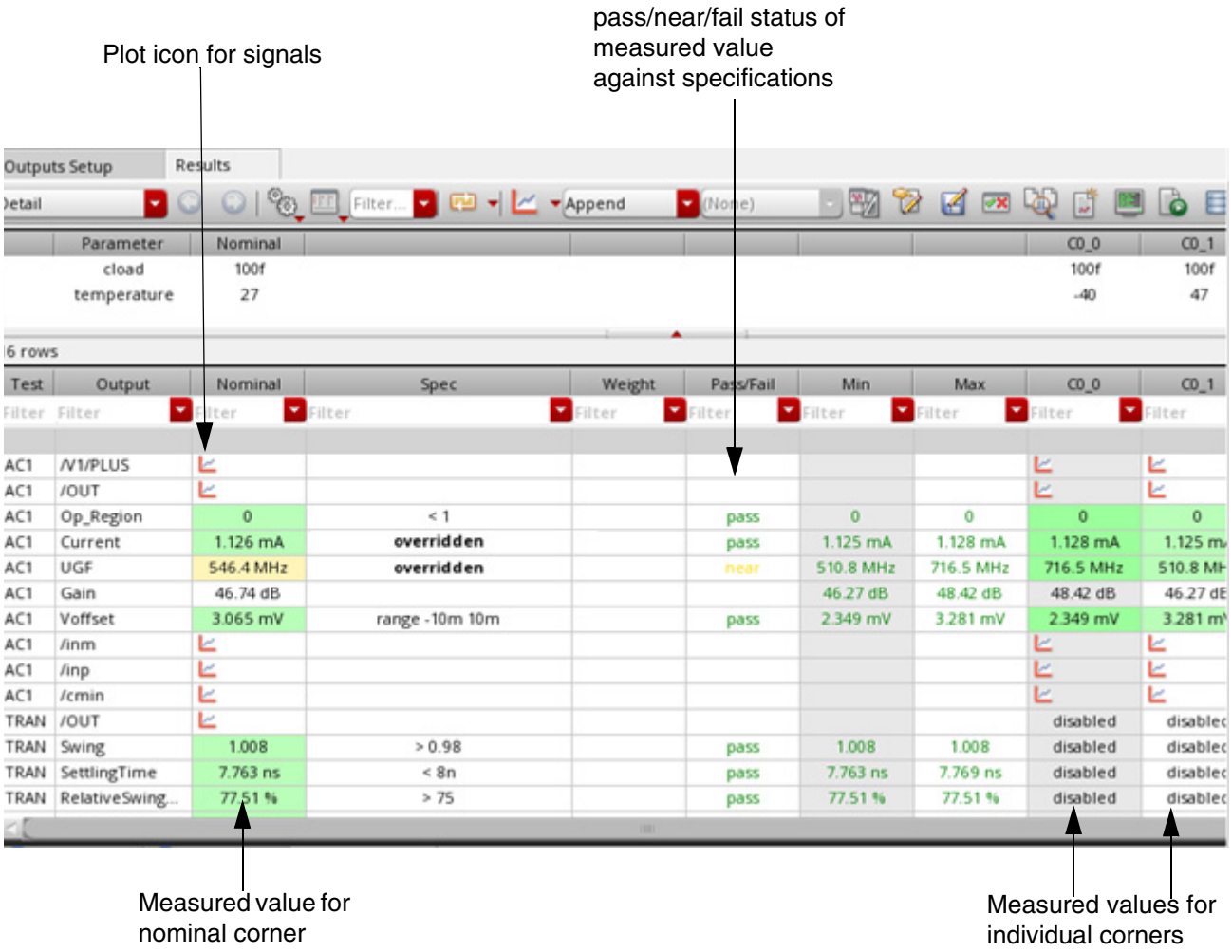
- In the *Plot* column on the Outputs Setup tab, select the check box next to the specification.
- Right-click an operating region specification and choose *Enable Plot*.

© 2023 Cadence Design Systems, Inc.

Viewing Specification Results in the Results Tab

When you run a simulation, the simulation results are displayed in the Results tab of the Outputs pane. For more information about running simulations, see Chapter 17, “Running Simulations.”

Figure 22-3 Display of Simulation Results in the Results Tab



© 2023 Cadence Design Systems, Inc.

Tip

After running a simulation, if you add or modify measurement expressions or specifications in the Outputs Setup tab, you can click the button on the Results tab to update the results displayed in the Results tab based on the new or modified expressions and specifications. For more information about using the button, see Re-Evaluation of Expressions and Specifications.

The results will also be updated based on the new or modified expressions and specifications if you click the button on the Results tab to plot across all points. For more information about using the button, see Refreshing Graphs.

The Results tab displays a plot icon for the signals that you have selected for plotting or saving on the Outputs Setup tab of the Outputs pane. For more information about selecting signals for plotting or saving, see Specifying Whether a Result Will Be Saved or Plotted.

For measurements defined in the Outputs Setup tab, the program measures the result against the performance specification and displays the following information for each simulation in the Results tab.

- Measured value for the nominal corner in the *Nominal* column.
- Measured value for each corner in the column for the corner.
- If a specification is specified as an expression, OCEAN script, or SKILL code, the evaluated value of the expression or code is compared with the output value to identify the *pass* or *fail* status. To view the evaluated specification value, move the mouse pointer over the *Spec* column on the *Results* tab. The evaluated value is displayed in a tooltip.
- *pass*, *near*, *fail* or *error* status for the measured value from each simulation in the *Pass/Fail* column.
  - *pass* means that all the measured values are within the limits defined by the specification.

    For example, in Figure 22-3, the measured value `506.7m` for the `Gain_commonMode` measurement has the *pass* status because it falls within the target value `<550m` for the measurement.
  - *near* means that one or more measured values are no more than 10% outside the target value of the specification.

    For example, in Figure 22-3, the measured value `94.24` for the `Gain_openLoop` measurement has the *near* status because it falls near (within 10% of) the target value `>95` for the measurement.

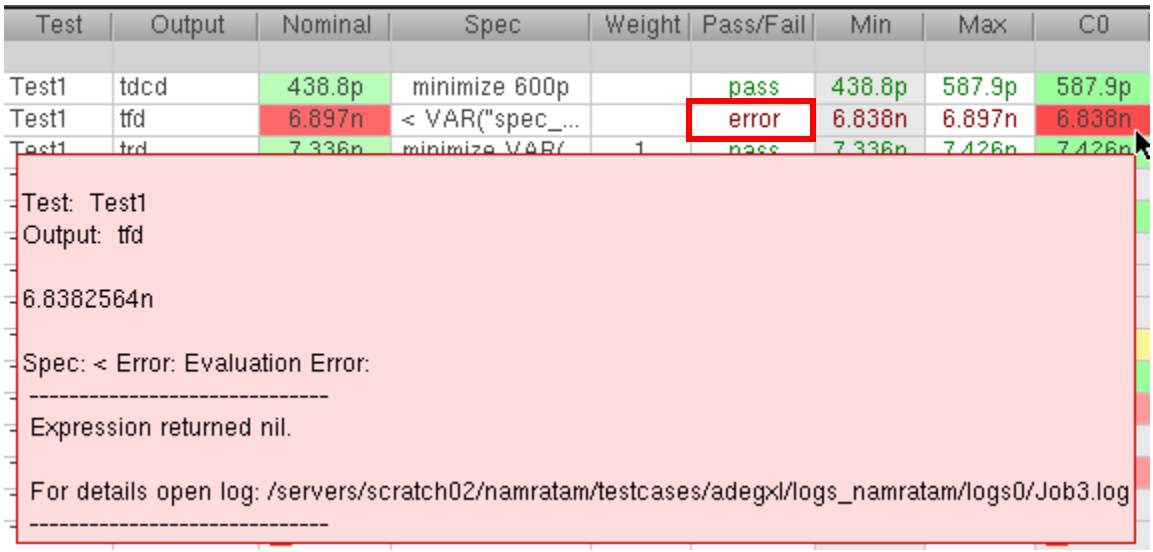
© 2023 Cadence Design Systems, Inc.

- *fail* means that one or more measured values are greater than 10% outside the target value of the specification.

  For example, in Figure 22-3, the measured value `118.4` for the `AC_gain_1KHz` measurement has the *fail* status because it falls outside the target value `>130` for the measurement.

- *error* means that the expression given in the Spec column for the output failed to evaluate.

  In this case, the result value is also displayed with a bright red background To view the error details, move the mouse over the result value in the corner column. The tooltip displays the details of the error, as shown in the figure given below.



The measured values with a *pass* status are displayed with a green background, those with a *near* status appear with a yellow background, and those with a *fail* status appear with a red background.

Note the following:

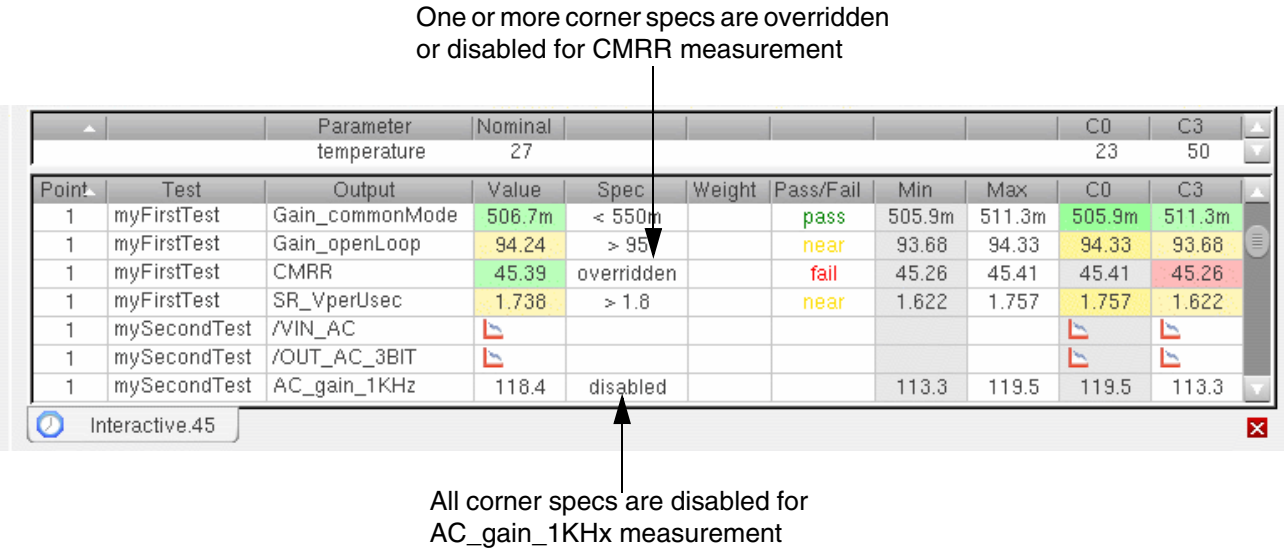
- The *Pass/Fail* column displays the *pass* status for a measurement only if the measured values for all the corners for the measurement have a pass status (displayed with a green background). For example, in 22-3, the `Gain_commonMode` measurement has the *pass* status because the measured values for the nominal corner (displayed in the *Nominal* column) and the corners *C0* and *C3* have a pass status.

- If the measured value for any corner has a *near* status (displayed with a yellow background), the *Pass/Fail* column displays the *near* status for the measurement.

© 2023 Cadence Design Systems, Inc.

- If the measured value for any corner has a fail status (displayed with a red background), the *Pass/Fail* column displays the *fail* status for the measurement.

- The text `overridden` displayed next to a measurement in the *Spec* column on the Results tab indicates that you have overridden, or disabled for one or more corner specifications for the measurement. For example, in the following figure, the text `overridden` next to the `CMRR` measurement in the *Spec* column indicates that you have overridden or disabled one or more corner specifications for the `CMRR` measurement.



Do the following to view the specification for the measurement and for each corner:

- To view the specification for the measurement, hover the mouse pointer over the text `overridden` in the *Spec* column.

  The specification for the measurement is displayed in a pop-up, as shown below:

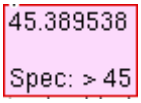
  Global Spec: > 45

- To view the specification for the nominal corner, hover the mouse pointer over the measured value for the nominal corner that is displayed next to the measurement in the *Nominal* column.

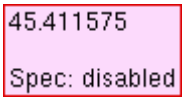
  To view the specification for a corner, hover the mouse pointer over the measured value for the corner that is displayed next to the measurement in the column for the corner.

© 2023 Cadence Design Systems, Inc.

The measured value and specification for the corner is displayed in a pop-up, as shown below:

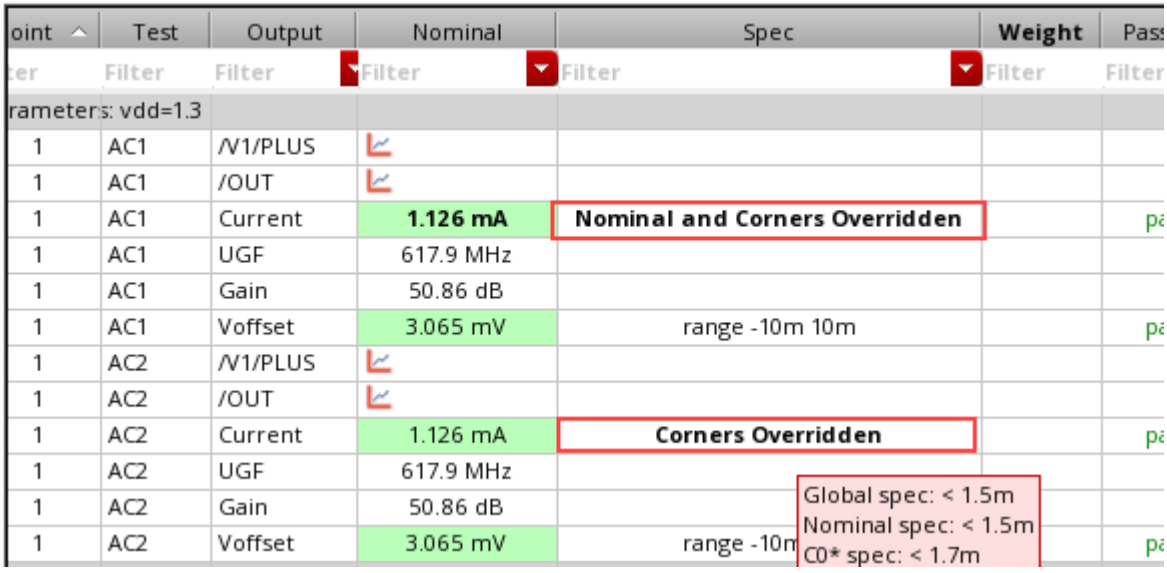


If a corner specification is disabled for the measurement, the pop-up displays the measured value and disabled status for the corner as shown below:



- When the showOverriddenValueForSpec environment variable is set to `t`, the specification overridden for nominal or specific corners are indicated by one of the following standard texts in the *Spec* column.
  - `Nominal Overridden`
  - `Corners Overridden`
  - `Nominal and Corners Overridden`

  As shown in the figure given below, you can move the mouse over these text values to view details of the global spec and the overridden specs in the tooltip.



- The text `disabled` displayed next to a measurement in the *Spec* column on the Results tab indicates that you have disabled all the corner specifications for the measurement. For example, in the results shown above, the text `disabled` next to the `AC_gain_1KHz`

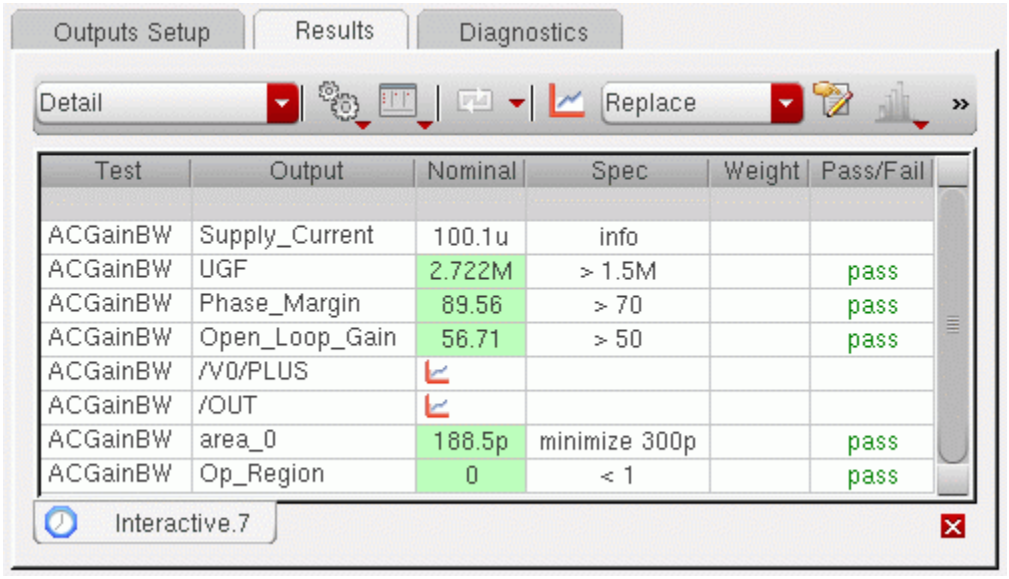
© 2023 Cadence Design Systems, Inc.

measurement in the *Spec* column indicates that you have disabled all the corner specifications for the `AC_gain_1KHz` measurement.

Viewing Operating Region Violations

When you run a simulation, ADE Assembler combines the individual device expressions in the operating region specification into a single output and calculates the total error. The total error is the sum of all the individual violation margins plus an offset of 1.

An operating region specification is met if the results for all its operating region expressions fall within their corresponding target values. The Results tab displays a value of `0` and a `pass` status if an operating region specification is met. For example, in the following figure, the operating region specification for the `ACGainBW` test has a `pass` status because its value is `0`.



A violation occurs when the result for any operating region expression in the specification falls outside its specified target value. In this case, the Results tab displays a value greater than `0` that is calculated as, shown below:

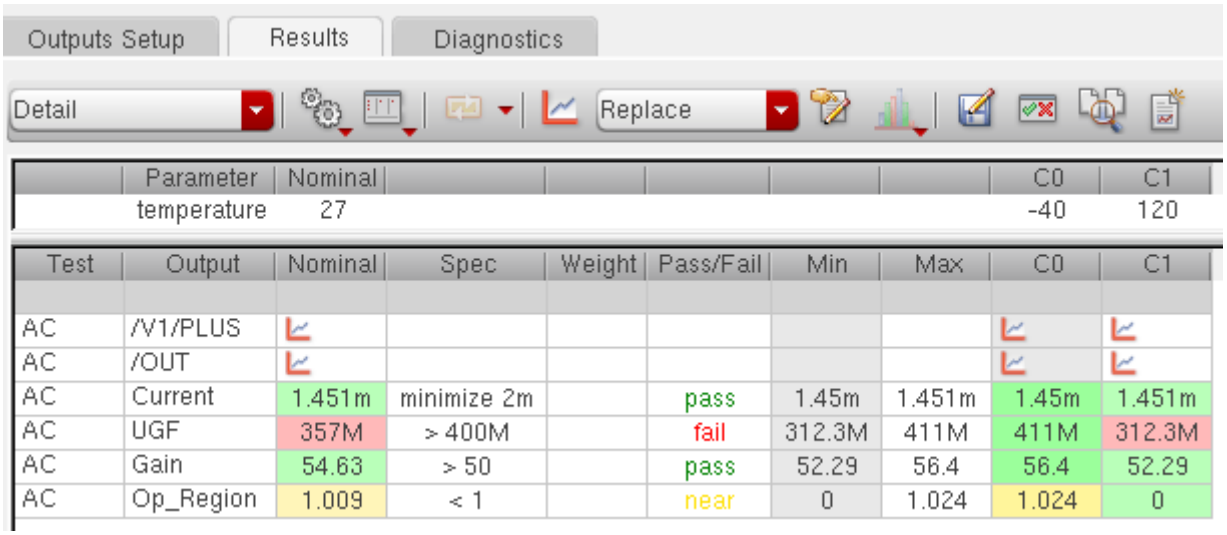
failed operating region result = 1 + sum of fail margin for failed expressions

Note: The operating region result is never between `0` and `1`.

A `near` or `fail` status depending on the range of deviation from the target spec value. For example, in the following figure, the operating region specification for the `AC` test,

© 2023 Cadence Design Systems, Inc.

`Op_Region`, has a `near` status because its value is slightly greater than `1` whereas the spec requires the value to be less than 1.



	Parameter	Nominal						C0	C1
	temperature	27						-40	120

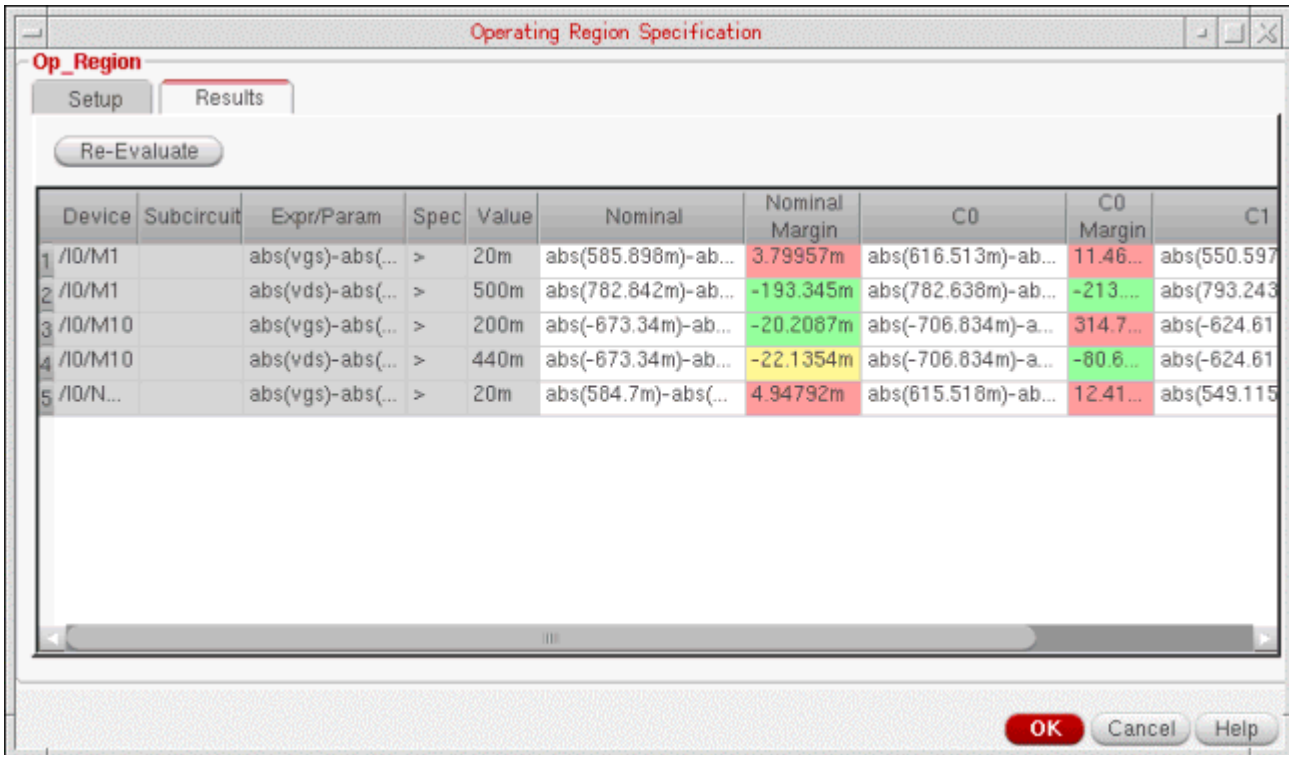
Test	Output	Nominal	Spec	Weight	Pass/Fail	Min	Max	C0	C1
AC	/V1/PLUS								
AC	/OUT								
AC	Current	1.451m	minimize 2m		pass	1.45m	1.451m	1.45m	1.451m
AC	UGF	357M	> 400M		fail	312.3M	411M	411M	312.3M
AC	Gain	54.63	> 50		pass	52.29	56.4	56.4	52.29
AC	Op_Region	1.009	< 1		near	0	1.024	1.024	0

To view operating region violations, do the following:

➡ In the Results tab of the Outputs pane, right-click the operating region specification and choose *Op. Region Violations*.

The Operating Region Specification form is displayed with the *Results* tab in view. This tab displays the results of all operating region expressions in a tabular format, as shown below:

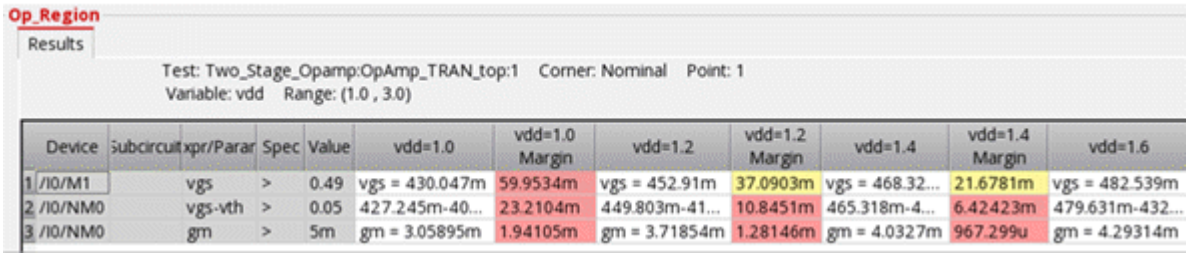
© 2023 Cadence Design Systems, Inc.



Two columns are shown for each corner—the column with corner name shows the result of expression evaluation for that corner and the *Margin* column next to it shows the margin by which the result deviates from the spec value. Depending on the difference between the spec and margin values, cells in the *Margin* column are colored as green, red, and yellow to highlight the `pass`, `fail`, or `near` status, respectively.

You can right-click the entries in this table and select *Expand Sweep* to view the DC sweep result values in a separate table.

This table displays the test name, corner name, point value, and the DC sweep variable information.



Note: This *Expand Sweep* option is available only when the *Evaluate over DC Sweep* check box is selected in the Setup tab.

© 2023 Cadence Design Systems, Inc.

Related Topics

Re-evaluating Operating Region Expressions on page 1411

Highlighting Operating Region Violations on the Schematic on page 1411

Unhighlighting Operating Region Violations on the Schematic on page 1412

Hiding and Showing a Single Column on page 1413

Hiding and Showing Multiple Columns on page 1413

Hiding and Showing Only Expression Columns or Margin Columns on page 1414

Re-evaluating Operating Region Expressions

Important

Make sure that you load the setup to active before re-evaluating outputs on the Outputs Setup tab. You cannot re-evaluate operating region expressions only by viewing results.

If required, you can edit the expressions that do not meet the specifications and re-evaluate results. For this, do the following in the Operating Region Specification form:

1. Open the *Setup* tab and add, edit, or delete expressions, as required.
2. Open the *Results* tab and click *Re-evaluate*.

The results for all the expressions are re-evaluated for all the corners and updated on the *Results* tab. If you get the desired results, click *OK* to save the changes in the operating region specification. Now, to see the result of change in operating region specification on the operating region output, click on the *Results* tab of ADE Assembler Outputs pane. All the expressions and operating region outputs are re-evaluated. The operating region output is now the result of new operating region specifications.

Highlighting Operating Region Violations on the Schematic

You can highlight the instance on the schematic for which an operating region expression has failed. This helps you to quickly correct the design or the operating region expression.

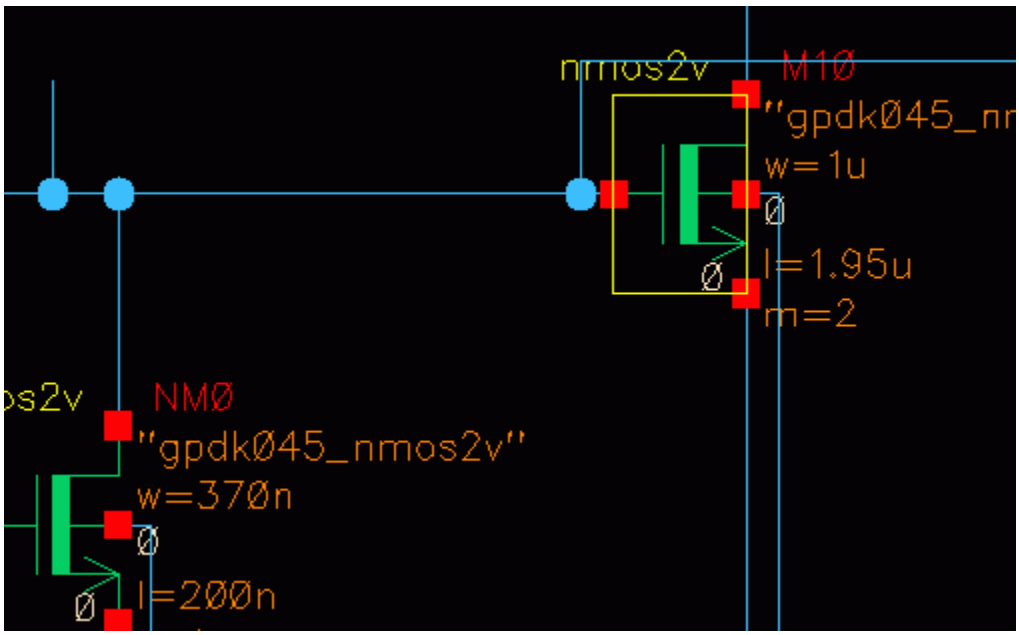
To highlight the instance for which an operating region expression has failed, do the following:

1. Right-click the row for which an expression has failed.

© 2023 Cadence Design Systems, Inc.

2. Choose *Highlight*.

   The instance for which the expression failed is highlighted in yellow on the schematic. For example, in the following figure, the instance `M10` for which an expression failed is highlighted on the schematic.



Tip

To highlight the instances corresponding to multiple failed expressions, hold down the *Shift* key (for contiguous selection) or the *Ctrl key* (for noncontiguous selection) and click the row for the next failed expression to add more expressions to the selection set, then click the *Highlight* button.

Note: In a hierarchical design, a block is highlighted in yellow if an operating region expression has failed for an instance in the block. When you descend into the block, the instance for which the expression has failed is also highlighted in yellow.

Unhighlighting Operating Region Violations on the Schematic

To clear the highlight from all the instances corresponding to the operating region violations on the schematic, do the following:

1. Right-click the row for which an expression has failed.
2. Choose *Clear*.

© 2023 Cadence Design Systems, Inc.

Hiding and Showing a Single Column

You can hide and show the required columns in the *Setup* or *Results* tab of the Operating Region Specification form.

To hide a column that is currently displayed, do the following:

1. Right-click a column name to display a context-sensitive menu.

   A tick mark next to a column name in the context-sensitive menu indicates that the column is currently displayed.

2. Choose the name of the column you want to hide.

To show a column that is currently not displayed,

➡ Right-click a column name and choose the name of the column you want to display from the context-sensitive menu.

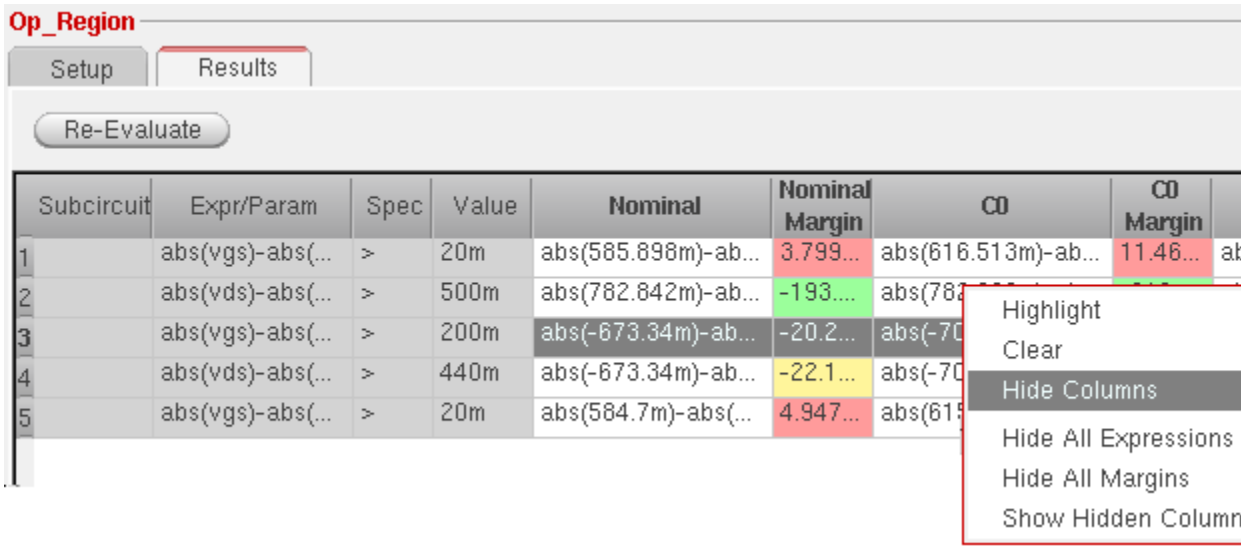
Hiding and Showing Multiple Columns

You can hide and show the required columns in the *Setup* or *Results* tab of the Operating Region Specification form.

To hide multiple columns that are currently displayed, do the following:

1. Hold down the Ctrl key and click in any cell in the columns that you want to hide.

2. Right-click in any of the selected cells and choose *Hide Columns*.



© 2023 Cadence Design Systems, Inc.

All the selected columns are hidden.

To display all the columns that are currently hidden,

➡ Right-click in any cell and choose *Show Hidden Columns*.

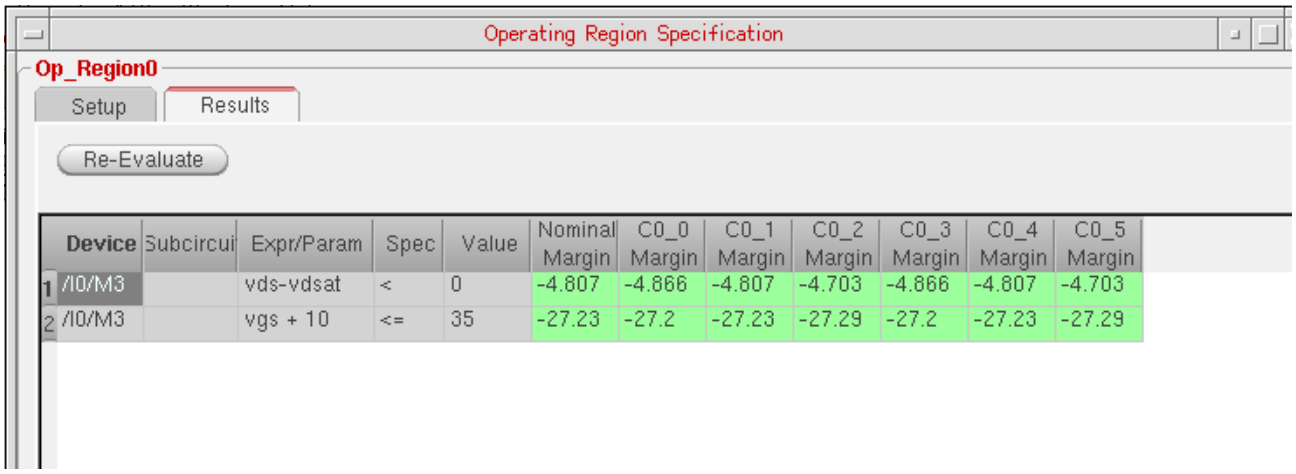
All the hidden columns become visible on the Results tab.

Hiding and Showing Only Expression Columns or Margin Columns

If multiple columns are displayed on the *Results* tab of the Operating Region Specification form, you can show only expression columns or margin columns. This helps in showing only one type of results at a time.

To hide all expression columns and to show only the margin columns:

➡ Right-click in any of the results cells and choose *Hide All Expressions*. All expression columns are hidden and only the margin columns appear, as shown below.



	Device	Subcircui	Expr/Param	Spec	Value	Nominal Margin	C0_0 Margin	C0_1 Margin	C0_2 Margin	C0_3 Margin	C0_4 Margin	C0_5 Margin
1	/I0/M3		vds-vdsat	<	0	-4.807	-4.866	-4.807	-4.703	-4.866	-4.807	-4.703
2	/I0/M3		vgs + 10	<=	35	-27.23	-27.2	-27.23	-27.29	-27.2	-27.23	-27.29

Tip

An alternate way to hide multiple columns is to right-click any column header and clear the check box for columns to be hidden. However, the *Hide All Margins* or *Hide All Expressions* commands provide quick options to hide multiple columns.

To unhide the hidden margin columns, right-click in any cell and choose *Show Hidden Columns*.

© 2023 Cadence Design Systems, Inc.

Working with the Specification Summary

The Spec Summary form provides a mechanism for managing and organizing simulation results for different purposes:

- Create a spec summary for design reviews
- Generate a summary report of pass/fail results for measurements
- Display results for quick inspection of multiple simulation runs
- Verify a design against specifications
- Export the spec summary to HTML or CSV format files for printing purposes.

For more information, see the following topics:

- Viewing the Spec Summary on page 1416
- Saving a Spec Summary on page 1419
- Opening a Spec Summary on page 1420
- Adding a Specification to the Spec Summary on page 1420
- Deleting a Specification from the Spec Summary on page 1421
- Changing the History Item from which Results are Displayed on page 1421
- Updating the Spec Summary with the Latest Results on page 1422
- Recreating the Spec Summary from the Results in the Active Results Tab on page 1423
- Viewing the Detailed Results for Specifications on page 1423
- Plotting the Results for Specifications on page 1425
- Exporting a Spec Summary to a HTML or CSV File on page 1425
- Sorting Data in the Spec Summary Form on page 1426
- Hiding and Showing Columns in the Spec Summary Form on page 1426

© 2023 Cadence Design Systems, Inc.

Viewing the Spec Summary

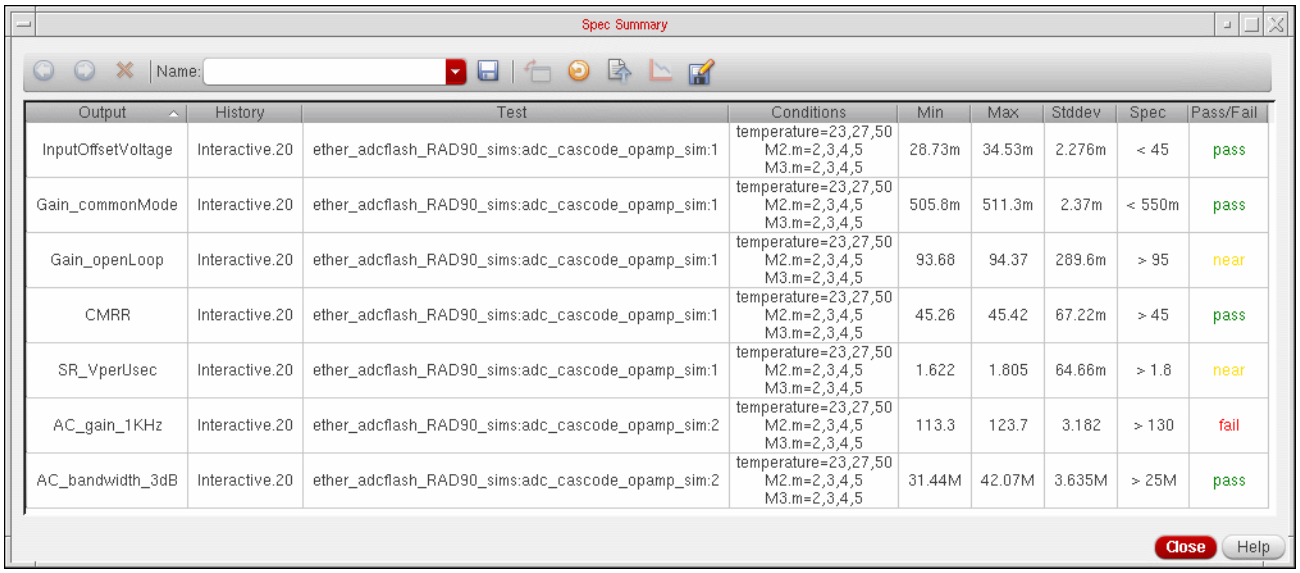
To view the specification (spec) summary, do the following on the Results tab of the Outputs pane:

Note: You can view the spec summary only for the results of a Single Run, Sweeps and Corners or Monte Carlo Sampling simulation run. You cannot view the spec summary for the results of a Global Optimization, Local Optimization, Size Over Corners or Sensitivity Analysis simulation run.

➤ Do one of the following on the currently active Results tab for a Single Run, Sweeps and Corners or Monte Carlo simulation run:

- ❑ Choose *Create – Spec Summary*.
- ❑ Click the toolbar button on the ADE Assembler window.
- ❑ Click the button on the Results tab.

The spec summary for the results in the currently active Results tab is displayed in the Spec Summary form.



Output	History	Test	Conditions	Min	Max	Stddev	Spec	Pass/Fail
InputOffsetVoltage	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:1	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	28.73m	34.53m	2.276m	< 45	pass
Gain_commonMode	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:1	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	505.8m	511.3m	2.37m	< 550m	pass
Gain_openLoop	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:1	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	93.68	94.37	289.6m	> 95	near
CMRR	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:1	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	45.26	45.42	67.22m	> 45	pass
SR_VperUsec	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:1	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	1.622	1.805	64.66m	> 1.8	near
AC_gain_1KHz	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:2	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	113.3	123.7	3.182	> 130	fail
AC_bandwidth_3dB	Interactive.20	ether_adcflash_RAD90_sims:adc_cascode_opamp_sim:2	temperature=23,27,50 M2.m=2,3,4,5 M3.m=2,3,4,5	31.44M	42.07M	3.635M	> 25M	pass

The toolbar in the Spec Summary form is described in the following table:

© 2023 Cadence Design Systems, Inc.

Table 22-2 Spec Summary Toolbar

Icon	Name	Description
	Back to Summary View	Switches from the detail view to the summary view. For more information, see Viewing the Detailed Results for Specifications on page 1423.
	Show Detail View	Displays the detailed results for the selected specifications in the detail view. For more information, see Viewing the Detailed Results for Specifications on page 1423.
	Delete	Deletes the selected specifications. For more information, see Deleting a Specification from the Spec Summary on page 1421.
	Save Spec Summary	Saves the spec summary. For more information, see Saving a Spec Summary on page 1419.
	Transpose Table	Displays the results in the detail view in horizontal or vertical format. For more information, see Viewing the Detailed Results for Specifications on page 1423.
	Update with Latest Results Data	Updates the spec summary with the latest results from the history item selected for each specification. For more information, see Updating the Spec Summary with the Latest Results on page 1422.
	Recreate for the Current History Results	Recreates the spec summary using the results in the active Results tab. For more information, see Recreating the Spec Summary from the Results in the Active Results Tab on page 1423.
	Plot Signals	Plots the results for the selected row. For more information, see Plotting the Results for Specifications on page 1425.

© 2023 Cadence Design Systems, Inc.

Icon	Name	Description
	Export to CSV or HTML File	Exports a spec summary to a HTML or comma-separated values (CSV) file For more information, see Exporting a Spec Summary to a HTML or CSV File on page 1425.

The columns in the Spec Summary form are described in the following table:

Table 22-3 Spec Summary Columns

Column	Description
Output	Displays the names of the outputs for which the specifications are evaluated.
History	Displays the names of the history items from which the results of outputs are displayed.
Test	Displays the names of the tests for which the results of outputs are displayed.
Conditions	Displays the values of each swept parameter as a list or range of values. For example, if a parameter `p` is swept through a list of values `x`, `y`, and `z`, then the *Conditions* column displays the values as `p=x,y,z` If a parameter `p` is swept through a range of values, the *Conditions* column displays the values as *`p=startValue:increment:stopValue`*
Min	Displays the minimum measured value for each output.
Max	Displays the maximum measured value for each output.

© 2023 Cadence Design Systems, Inc.

Column	Description
Std Dev	Displays the standard deviation of the measured values for each output. You can hover the mouse pointer over a standard deviation value to view the following information in a pop-up: ■ The number of simulations for the output expression. ■ The mean of the measured values for all simulations for the output expression. ■ The variance in the measured values. **Note:** For Single Run, Sweeps and Corners, this is the population standard deviation. For other run modes, this is the sample standard deviation.
Spec	Displays the specifications for outputs.
Pass/Fail	Displays a *pass*, *near* or *fail* status for each specification. ■ *pass* means that all the measured values are within the limits defined by the specification. ■ *near* means that one or more measured values are no more than 10% outside the target value of the specification. ■ *fail* means that one or more measured values are greater than 10% outside the target value of the specification. To investigate the measured values further, see Viewing the Detailed Results for Specifications.

Saving a Spec Summary

To save a spec summary in the Spec Summary form, do the following:

1. In the *Name* field, type a name for the spec summary.
2. Click the button.

The spec summary is saved to the `documents` directory of the maestro view and displayed in the *Documents* tree on the Data View pane. For more information about working with documents, see Chapter 23, “Working with Documents and Datasheets.”

Note: The spec summary file is saved with the `.specsummary` extension.

© 2023 Cadence Design Systems, Inc.

Opening a Spec Summary

To open an existing spec summary, do one of the following:

- In the *Name* drop-down list on the Spec Summary form, select the spec summary.
- In the *Documents* tree on the Data View pane, double-click the spec summary.

The spec summary is displayed in the Spec Summary form.

Deleting a Spec Comparison

To delete an existing spec summary, do the following:

➡ In the *Documents* tree on the Data View pane, right-click the spec summary you want to delete and choose *Delete*.

The spec summary will not be displayed in the *Name* drop-down list in the Spec Summary form.

Adding a Specification to the Spec Summary

You can add a specification from the results in the active Results tab or from the results of a history item.

Note: You can add specifications only from the results of a Single Run, Sweeps and Corners or Monte Carlo simulation run. You cannot add specifications from the results of a Global Optimization, Local Optimization, Size Over Corners or Sensitivity Analysis simulation run.

To add a specification from the results in the active Results tab for a Single Run, Sweeps and Corners or Monte Carlo simulation run, do the following:

➤ Right-click a row in the Results tab of the Outputs pane and choose *Add to Spec Summary*.

The specification is added to the spec summary.

Note: You cannot add an output of type signal to the spec summary.

To add a specification from the results of a history item for a Single Run, Sweeps and Corners or Monte Carlo simulation run, do the following:

1. Display the results for the history item. For more information, see Working with History Checkpoints.

© 2023 Cadence Design Systems, Inc.

2. On the Results tab for the history item, right-click a specification and choose *Add to Spec Summary*.

Tip

To add multiple specifications, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection), right-click and choose *Add to Spec Summary*.

Deleting a Specification from the Spec Summary

To delete a specification from the spec summary, do the following.

➤ Select the specification and click the button.

To delete multiple specifications, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection) and click the button.

Changing the History Item from which Results are Displayed

The *History* column in the Spec Summary form displays the names of the history items from which the results of the specifications are displayed. You can change the history item for a specification to display the results from that history item.

Note: You can change only to a history item for a Single Run, Sweeps and Corners or Monte Carlo simulation run. You cannot change to a history item for a Global Optimization, Local Optimization, Size Over Corners or Sensitivity Analysis simulation run.

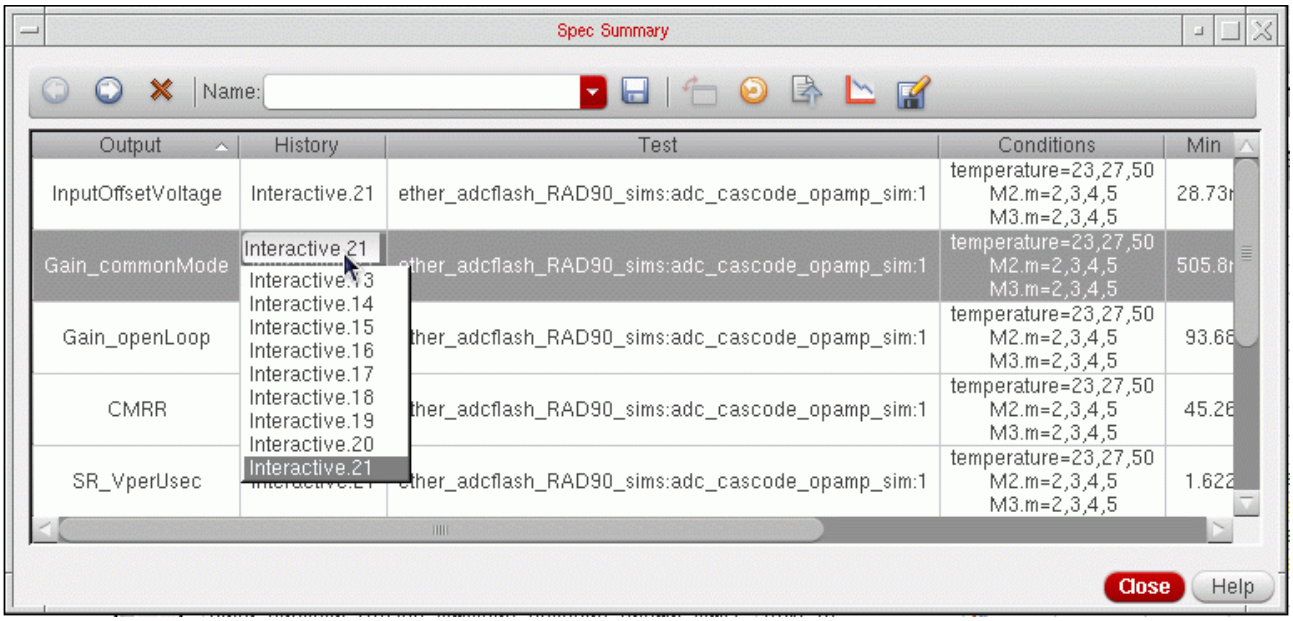
To change the history item for a single specification, do the following:

1. In the *History* column, click the name of the history item for the specification.

   A button with the name of the history item appears.

© 2023 Cadence Design Systems, Inc.

2. Click the button and select a different history item.



The results for the specification from the selected history item are displayed in the spec summary.

To change the history item for multiple specifications, do the following:

1. Hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection) and click the name of a history item for any specification in the selection.

   A button with the name of the history item appears.

2. Click the button and select a different history item.

   The results for the specifications from the selected history item are displayed in the spec summary.

Updating the Spec Summary with the Latest Results

To update the spec summary with the latest results from the history item selected for each specification, do the following:

➤ Click the button.

Note: If the history item selected for a specification does not exist, the row for the specification is highlighted in red color. You can select a different history item for the specification (see Changing the History Item from which Results are Displayed on

© 2023 Cadence Design Systems, Inc.

page 1421) or delete the specification from the spec summary (see Deleting a Specification from the Spec Summary on page 1421).

Recreating the Spec Summary from the Results in the Active Results Tab

To clear the current spec summary view and display all the specifications from the active Results tab, do the following.

➤ Click the button.

Note: You can recreate the spec summary only from the results of a Single Run, Sweeps and Corners or Monte Carlo simulation run. You cannot recreate the spec summary from the results of a Global Optimization, Local Optimization, Size Over Corners or Sensitivity Analysis simulation run.

Viewing the Detailed Results for Specifications

To view the detailed results for a specification, do the following:

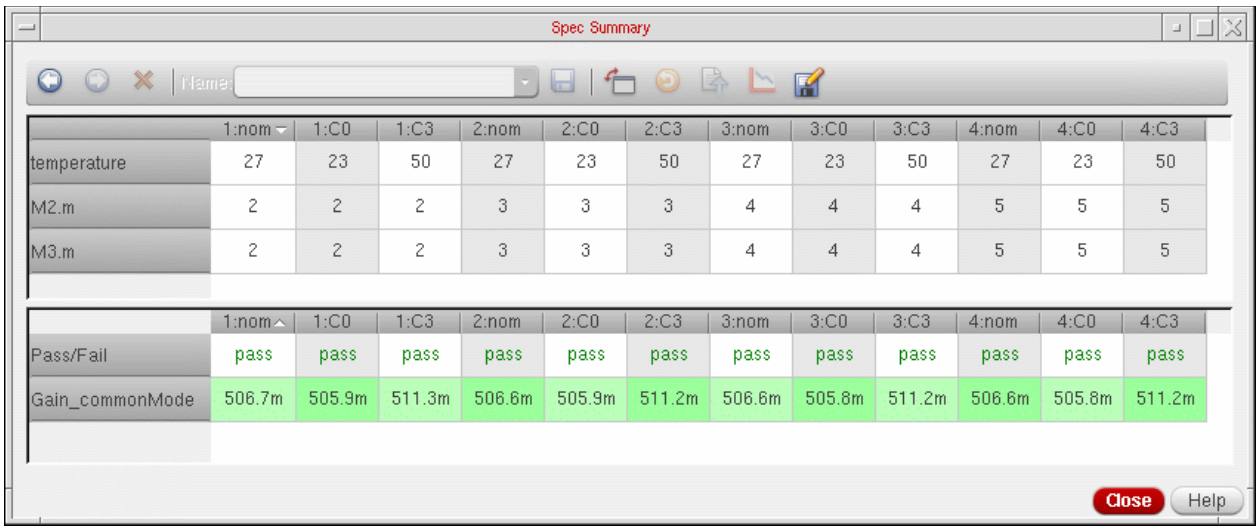
➤ Select a specification and click the button.

To view the detailed results for multiple specifications, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection) and click the button. Ensure that the specifications you select have the same conditions.

The detailed results are displayed in the detail view. You can click the button to view the detailed results in horizontal or vertical format.

© 2023 Cadence Design Systems, Inc.

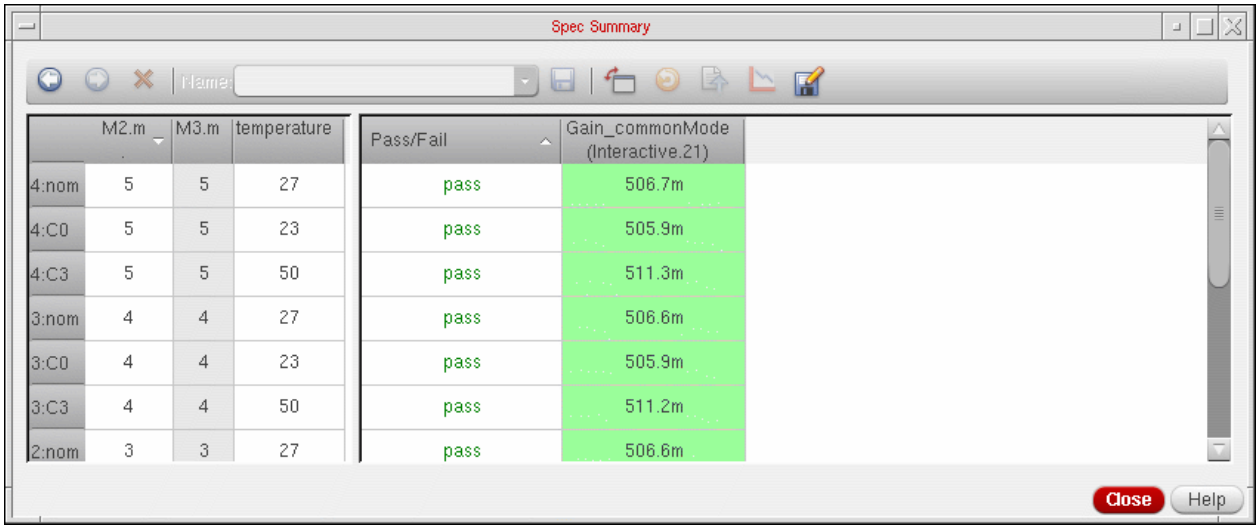
Figure 22-4 Detailed Results in Horizontal Format



	1:nom	1:C0	1:C3	2:nom	2:C0	2:C3	3:nom	3:C0	3:C3	4:nom	4:C0	4:C3
temperature	27	23	50	27	23	50	27	23	50	27	23	50
M2.m	2	2	2	3	3	3	4	4	4	5	5	5
M3.m	2	2	2	3	3	3	4	4	4	5	5	5

	1:nom	1:C0	1:C3	2:nom	2:C0	2:C3	3:nom	3:C0	3:C3	4:nom	4:C0	4:C3
Pass/Fail	pass	pass	pass	pass	pass	pass	pass	pass	pass	pass	pass	pass
Gain_commonMode	506.7m	505.9m	511.3m	506.6m	505.9m	511.2m	506.6m	505.8m	511.2m	506.6m	505.8m	511.2m

Figure 22-5 Detailed Results in Vertical Format



	M2.m	M3.m	temperature	Pass/Fail	Gain_commonMode (Interactive.21)
4:nom	5	5	27	pass	506.7m
4:C0	5	5	23	pass	505.9m
4:C3	5	5	50	pass	511.3m
3:nom	4	4	27	pass	506.6m
3:C0	4	4	23	pass	505.9m
3:C3	4	4	50	pass	511.2m
2:nom	3	3	27	pass	506.6m

The detail view displays the following information:

- Values of swept parameters for each simulation.
- *pass*, *near or fail* status for each simulation.
  - *pass* means that the measured value is within the limits defined by the specification.
  - *near* means that the measured value is no more than 10% outside the target value of the specification.

© 2023 Cadence Design Systems, Inc.

- *fail* means that the measured value is greater than 10% outside the target value of the specification.

- Measured values from each simulation.

  The measured values with a *pass* status are displayed with a green background, those with a *near* status appear with a yellow background, and those with a *fail* status appear with a red background.

To go back to the summary view, do the following:

➤ Click the button.

Plotting the Results for Specifications

To plot the results for the specifications in the summary view, do the following:

➤ Select a specification and click the button.

To plot the results for multiple specifications, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection) and click the button.

To plot the results in the detail view, do the following:

1. Click the button to display the results in the detail view in the vertical format.

   See Detailed Results in Vertical Format figure on page 1424 for an example of the detail in the vertical format.

2. Select a row in the table on the left-hand side and click the button.

   To plot the results for multiple rows, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontiguous selection) and click the button.

Exporting a Spec Summary to a HTML or CSV File

To export a spec summary to a HTML or comma-separated values (CSV) file, do the following:

1. Click the button.

   The Export Results form appears.

2. In the *File name* field, type a file name with the `.html`, `.htm` or `.csv` extension.

3. Click *Save*.

© 2023 Cadence Design Systems, Inc.

The program exports the results to the file you specified. By default, the file is saved in the `documents` directory of the maestro view and displayed in the *Documents* tree on the Data View pane. For more information about working with documents, see Chapter 23, “Working with Documents and Datasheets.”

Note: If you click the button in the detail view, the resulting HTML or CSV file contains the contents of the detail view and summary view.

Sorting Data in the Spec Summary Form

To sort data in the Spec Summary form, do the following:

➤ Click the name of the column based on which you want to sort the data.

Hiding and Showing Columns in the Spec Summary Form

To hide a column, do the following:

➤ Right-click the name of any column and select the name of the column you want to hide.

To display a hidden column, do the following:

➤ Right-click the name of any column and select the name of the column you want to display.

© 2023 Cadence Design Systems, Inc.

23

Working with Documents and Datasheets

ADE Assembler allows you to add documents that you want to refer to when working with the design. For example, you can add the design specifications document for your design, so that you can refer to it when working with the design. The added documents are displayed in the *Documents* directory in the Data View pane. You can double-click a document to open it in its associated program.

Another important feature in ADE Assembler is the ability to save the setup details and results in the form of datasheets. These datasheets can be saved in .html format and viewed in Internet browsers.

See the following topics for more information:

- Adding Documents on page 1427
- Opening Documents on page 1428
- Removing Documents on page 1428
- Saving Documents on page 1429
- Creating a Datasheet for a History on page 1430
- Displaying Customized Waveform Images in the Datasheet on page 1442
- Opening a Datasheet on page 1443
- Customizing the Datasheet Format and Structure on page 1443

Adding Documents

You can add documents in the HTML, CSV, PDF and text documents in ADE Assembler.

To add a document, do the following:

1. Choose *Create – Document*.

© 2023 Cadence Design Systems, Inc.

Tip

Alternatively, you can click where it says *Click to add document* in the Document directory on the Data View assistant pane.

The Choose Documents to be Added form appears.

2. Select the document you want to add and click *Open*.

   The document is displayed in the *Documents* directory in the Data View pane.

Important

The documents you add are copied to your maestro view. If you update a document at its original location, ensure that the latest version of the document is added in ADE Assembler.

Opening Documents

To open a document, do the following:

➤ In the *Documents* tree on the Data View pane, double-click the document.

The document is opened in its associated program. For example, if you open a PDF file, it is opened in Adobe Acrobat.

The default application or program associated with different file types cannot be specified in Virtuoso. You can use the window manager settings for your operating system to change the default association. For example, in KDE, you can use the *Settings–File Associations* command.

Removing Documents

Caution

There is no undo for this action.

To remove a document, do the following:

➤ In the *Documents* tree on the Data View pane, right-click the document you want to remove and choose *Delete*.

The program deletes the selected document.

© 2023 Cadence Design Systems, Inc.

Note: When a document is removed, it is only deleted from your maestro view. It is not deleted from the location from which you added the document.

Saving Documents

You can save a document displayed in the *Documents* tree on the Data View pane under a different name or at a different location.

To save a document under a different name or at a different location, do the following:

1. In the *Documents* tree on the Data View pane, right-click the document you want to save under a different name or at a different location and choose *Save As*.

   The Save Document As form appears.
2. Select the directory in which you want to save the file.
3. In the *File name* field, enter the name under which you want to save the document.
4. Click *Save*.

© 2023 Cadence Design Systems, Inc.

Working with Datasheets

- Creating a Datasheet for a History on page 1430
- Displaying Customized Waveform Images in the Datasheet on page 1442
- Opening a Datasheet on page 1443
- Customizing the Datasheet Format and Structure on page 1443

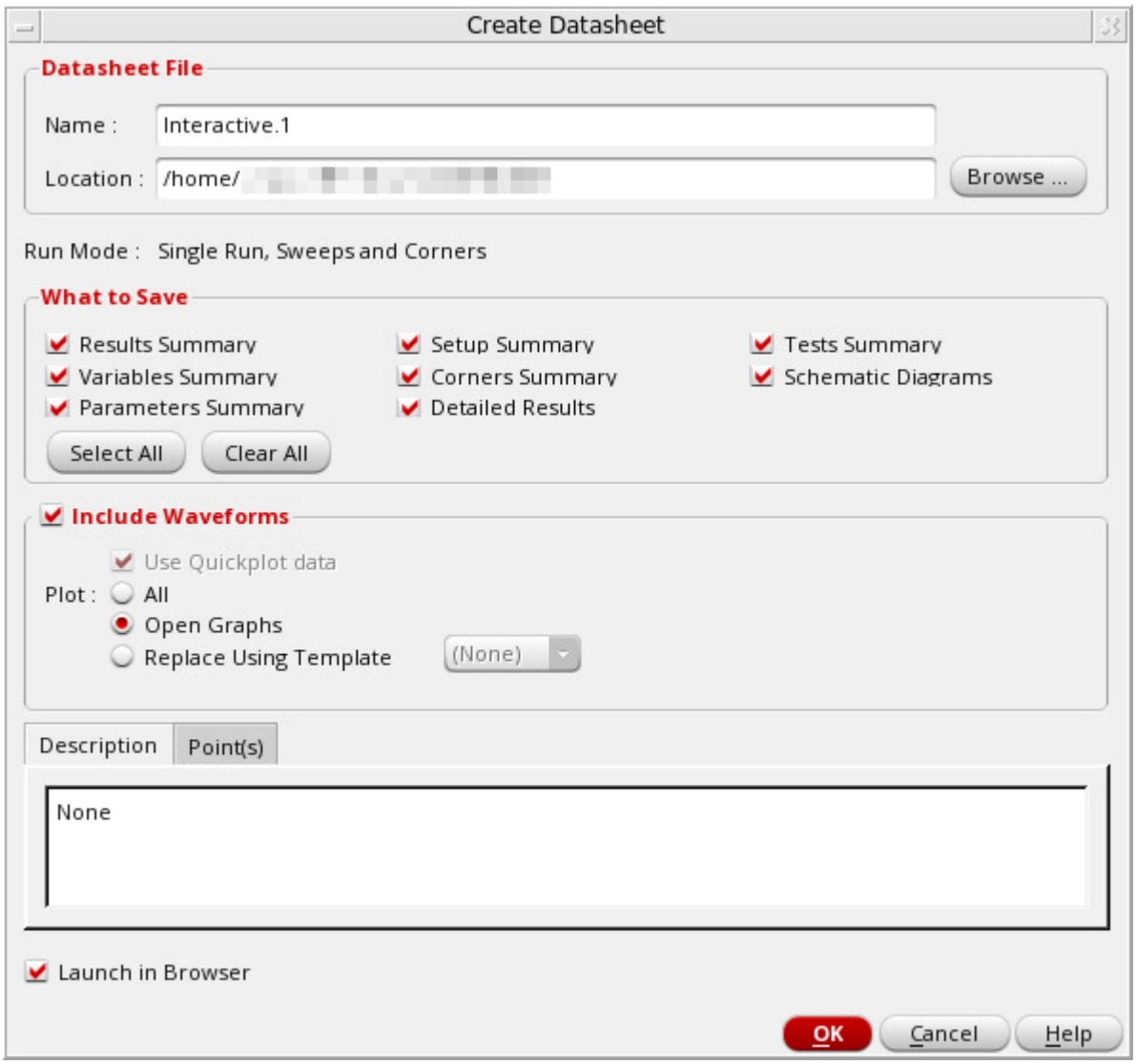
Creating a Datasheet for a History

To create a datasheet for a history checkpoint, do the following:

1. In the History tab of the Data View pane, right-click the history for which you want to create a datasheet and choose *Create Datasheet*.

© 2023 Cadence Design Systems, Inc.

The Create Datasheet form appears.



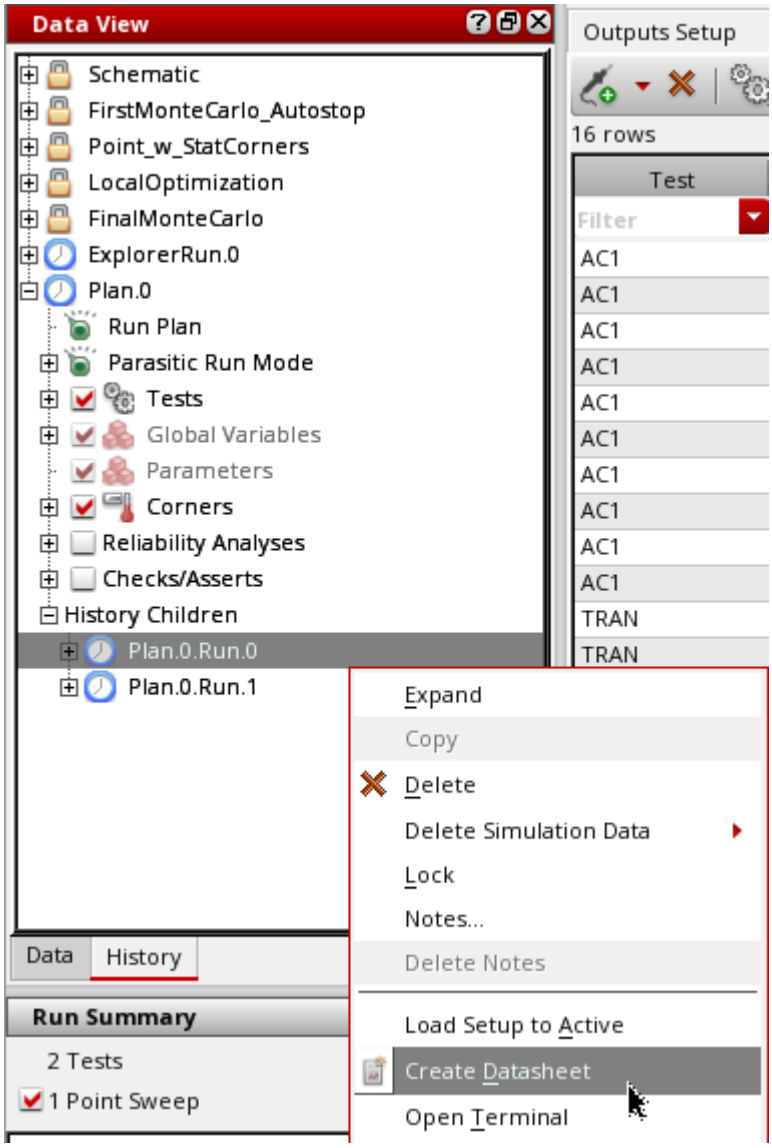
Note the following:

- You can also choose *Create – Datasheet* or click on the Results toolbar to open the Create Datasheet form.
- You cannot create a datasheet for a check point when simulations are in progress for that checkpoint.
- For group histories, you can create a datasheet at the group level or for a specific child history.

© 2023 Cadence Design Systems, Inc.

To create a datasheet at the group history level, right-click the group history and choose *Create Datasheet*. Datasheets for all child histories in that group are saved in separate html files. In addition, a datasheet containing links to all the datasheets created for child histories is created for the group level.

To create a datasheet at the level of a child history, right-click the name of the child history and choose *Create Datasheet*.



2. In the *Name* field, type a name for the datasheet.

   Note: By default, the name of the datasheet is the name of the checkpoint for which you are creating the datasheet. For example, if you are creating a datasheet for a checkpoint named `Interactive.17`, the default name of the datasheet will be `Interactive.17`.

© 2023 Cadence Design Systems, Inc.

3. In the *Location* field, type the directory path where you want the program to write your datasheet; or do the following:

   a. Click *Browse*.

   b. On the form that appears, navigate to and select the directory where you want the program to write your datasheet.

   c. Click *Open*.

   Note the following:

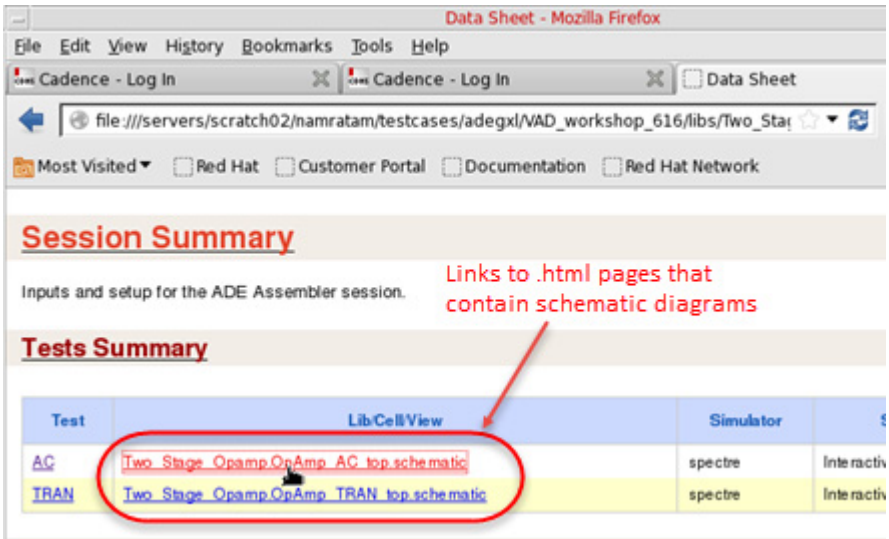
   - By default, the datasheet is created in the `documents` directory of the maestro view.
   - If you do not specify the absolute path to a directory in the *Location* field, the directory will be created in the current working directory. For example, if you run the `virtuoso` command from `/net/designs/myDesign` and type `myDatasheets` in the *Location* field, datasheets will be created in the `/net/designs/myDesign/myDatasheets` directory.

4. (Optional) Select or clear the following check boxes:

Check Box	Description
Results Summary	When selected, this option writes the results summary information to the datasheet. The results summary information is displayed in the following sections in the datasheet: ■ *Results Summary* section in the main datasheet page ■ *Results Summary* section in the datasheet page for each test The text *Various* in the *Target* column in the *Results Summary* section indicates that you have overridden or disabled one or more corner specifications for a measurement. For more information, see Disabling and Enabling Corner Specifications on page 1339. For information about the colors used in the *Minimum Value* and *Maximum Value* columns in the *Results Summary* section, see Colors Used to Display Status of Measured Values and Specifications in Datasheets on page 1441.

© 2023 Cadence Design Systems, Inc.

Check Box	Description
Variables Summary	When selected, this option writes the variables summary information to the datasheet, including names and values of variables. The variables summary information is displayed in the *Variables* section in the main datasheet page.
Corners Summary	When selected, this option writes the corners summary information to the datasheet, including the list of corners enabled or disabled for each test, values of parameters and design variables at corners, and the model groups added for each corner. The corners summary information is displayed in the *Corners* section in the main datasheet page.
Schematic Diagrams	When selected, this option saves the schematic diagrams in the datasheet. These diagrams are not displayed on the main datasheet page. Instead, a link to the diagram for each test is added to the *Tests Summary* section. When you click the hyperlink for the saved schematic diagram, a .html page saved by Virtuoso Schematic Editor is displayed. Diagrams of the schematic of the design and the devices instantiated in it are displayed on this page. The top of the page shows a Table of Contents (TOC) and a graphical TOC. Click the links in the TOC to view a particular schematic diagram.



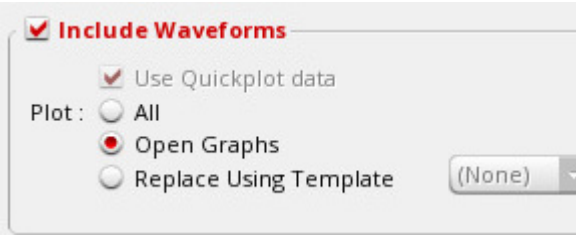
© 2023 Cadence Design Systems, Inc.

Check Box	Description
Tests Summary	When selected, this option writes the test summary information to the datasheet, including the test name, test design, simulator, and state information. The tests' summary information is displayed in the *Tests Summary* section in the main datasheet page.
Parameters Summary	When selected, this option writes the parameters' summary information to the datasheet, including the names and values of parameters. Parameters' summary information is displayed in the *Parameters* section in the main datasheet page.
Detailed Results	When selected, this option writes the detailed results information for each test to the datasheet, including the parameter values for each run (sweeps and corners) and the output values for each measurement expression. The detailed results information for each test is displayed in the *Detailed Results* section in the datasheet page for that test. For information about the colors used in the *Value* and *Target* columns in the *Outputs* sub-section of the *Detailed Results* section, see Colors Used to Display Status of Measured Values and Specifications in Datasheets on page 1441. **Note:** The *Target* column is displayed for a measurement in the *Outputs* sub-section of the *Detailed Results* section only if you have overridden or disabled a corner specification for the measurement. For more information about overriding or disabling corner specifications, see Working with Specifications on page 1329.

© 2023 Cadence Design Systems, Inc.

Check Box	Description
Launch in browser	When selected, this option launches the datasheet in a web browser. The datasheet is displayed in the browser specified in the *Web Browser* field of the User Preferences form. The default value of the *Web Browser* field is `firefox`. You can also specify the default browser to be used by setting the following environment variable in the your `.cdsenv` file before starting the Virtuoso Design Environment: `ui webBrowser string "`*`browserName`*`"` where `browserName` is the name of the browser executable. The default value for this environment variable is `firefox`. **Note:** Ensure that your web browser supports XML and XSLT.

5. Select the *Include Waveforms* check box to display waveform images in the datasheet files. This check box is selected by default.



- Select the *Use Quickplot data* check box if you want to use quick plot data to print waveforms in the datasheet. This option helps quickly print waveforms in the datasheet against regular plot. When this check box is selected, quick plot is used for transient data and regular plot is used for non-transient data. The default setting of this check box is determined by the value of the useQPDataToCreateDatasheet environment variable.
- In the *Plot* field,
  - Select *All* if you want to plot all the graphs in the Virtuoso Visualization and Analysis XL graph window and include them in the datasheet.

  Note: The *Use Quickplot data* check box field is available only when you select the *All* option.

© 2023 Cadence Design Systems, Inc.

- Select *Open Graphs* to include graphs that are currently open in the graph window. When this option is selected, the *Use Quickplot data* check box field is not available.
- Select *Replace Using Template* if you want to plot the graphs from a plotting template, which you saved earlier, and include them in the datasheet. You can then select the plotting template from the drop-down list displayed next to it. When this option is selected, the *Use Quickplot data* check box field is not available.

Note the following:

- If you select the *Open Graphs* option, but no graph is plotted in the graph window, then the plot all command is run to plot all the graphs. These graphs are then printed in the datasheet.
- If you select the *All* option, but some waveforms are already plotted in the graph window, then new waveforms are plotted based on the mode you have selected in the *Plotting Mode* drop-down list on the *Results* tab. The available plotting modes are: append, replace, new window, and new subwindow.

The waveform images are saved as Portable Network Graphic (.png) image files in the datasheet directory. For more information about the location of the datasheet directory, see Location of Datasheet Files on page 1440.

See also: Displaying Customized Waveform Images in the Datasheet on page 1442

6. By default, the waveform images in the results directory for a history item are displayed in the datasheet. If waveform images are not available for a history item, they are automatically created and displayed in the datasheet.
7. (Optional) On the Points tab, do one of the following:

Select	To
Design Point ID	Include results information only for a specific design point in the datasheet. Type the ID of the design point in the *Design Point ID* field. **Note:** You can find the ID for a design point from the *Point* column in the Results tab of the Outputs pane.
All Design Points	Include results information for all the design points in the datasheet.

© 2023 Cadence Design Systems, Inc.

8. (Optional) On the Description tab, enter a description for the datasheet.
9. Click *OK*.

The program writes the datasheet information you requested. If you selected the *Launch in browser* check box, ADE Assembler launches the datasheet in the default HTML browser, as shown below.

Datasheet for run Interactive.5

Source and Contents

The run mode is: Single Run, Sweeps and Corners.

- Setup Summary
- Results Summary
- Session Summary
  - Tests Summary
  - Variables
  - Corners
    - Tests/Corners Matrix
    - Model Groups
- Detailed Results
  - AC

Setup Summary

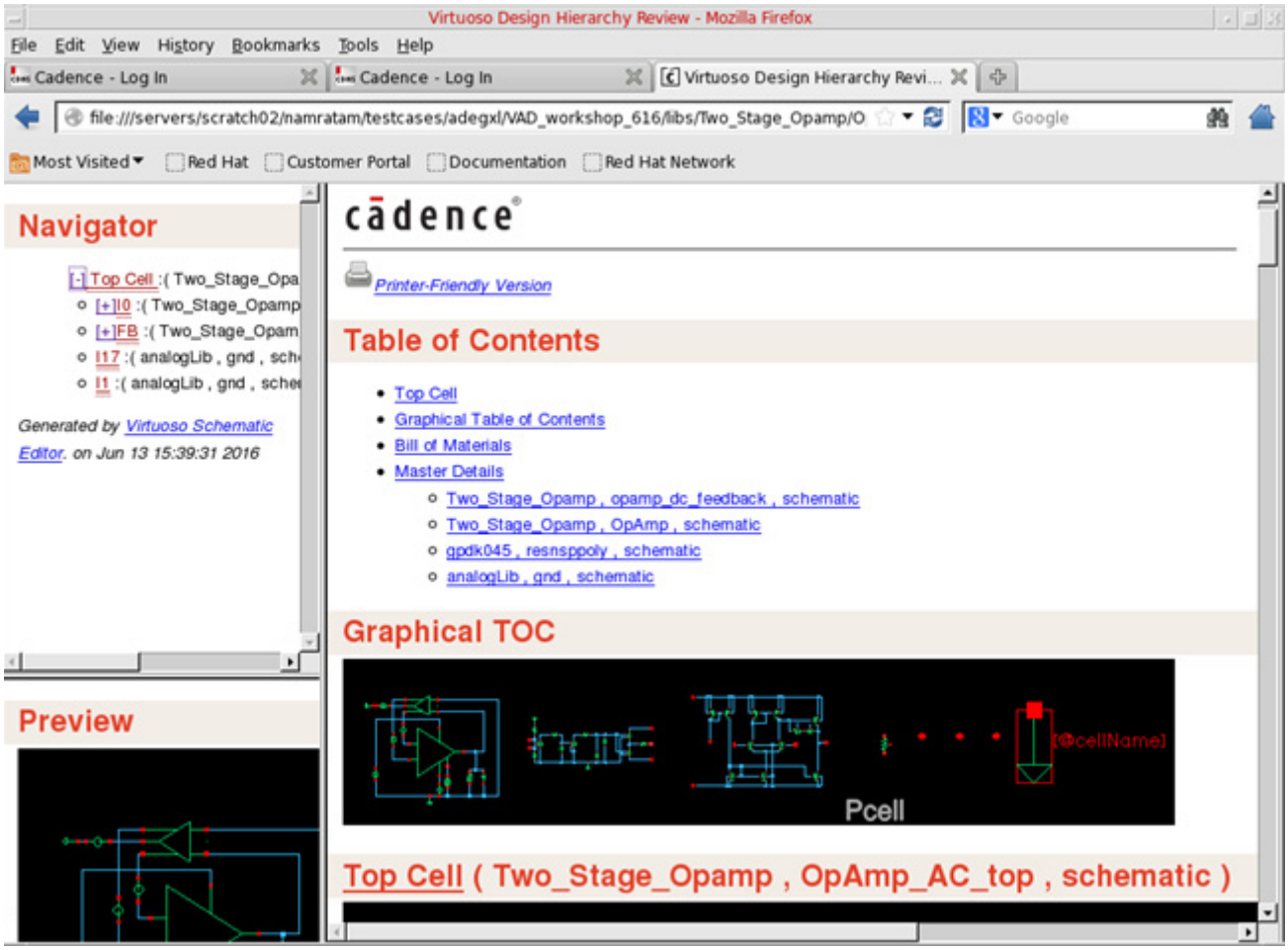
Library	Two_Stage_Opamp
Cell	OpAmp_AC_top
View	maestro_sweeps-corners

Results Summary

	Test	Calculation	Expression	Target	Minimum Value
AC					
	AC	Current	abs(IDC("/V1/PLUS"))	< 1.5m	1.086m
	AC	UGF	cross(dB20(mag(VF("/OUT"))) 0 1 "falling")	> 460M	439.7M
	AC	Gain	value(dB20(mag(VF("/OUT"))) 1)	> 44	44.87
	AC	Voffset	(VDC("/inm") - VDC("/inp"))	range -10m 10m	2.947m
	AC	CMRR	(value(dB20(mag(VF("/OUT"))) 1) - value(db(getData("/Vcm" ?result "xf")) 1))		42.72
	AC	GainVsupply1	value(db(getData("/V1" ?result "xf")) 1)		-6.982
	AC	PSRR	(value(dB20(mag(VF("/OUT"))) 1) - value(db(getData("/V1" ?result "xf")) 1))		47.69
	AC	GainCm	value(db(getData("/Vcm" ?result "xf")) 1)		-1.286
	AC	bandwidthAcrossCorners	bandwidth(dB20(mag(VF("/OUT"))) 10 "low")		Waveform

© 2023 Cadence Design Systems, Inc.

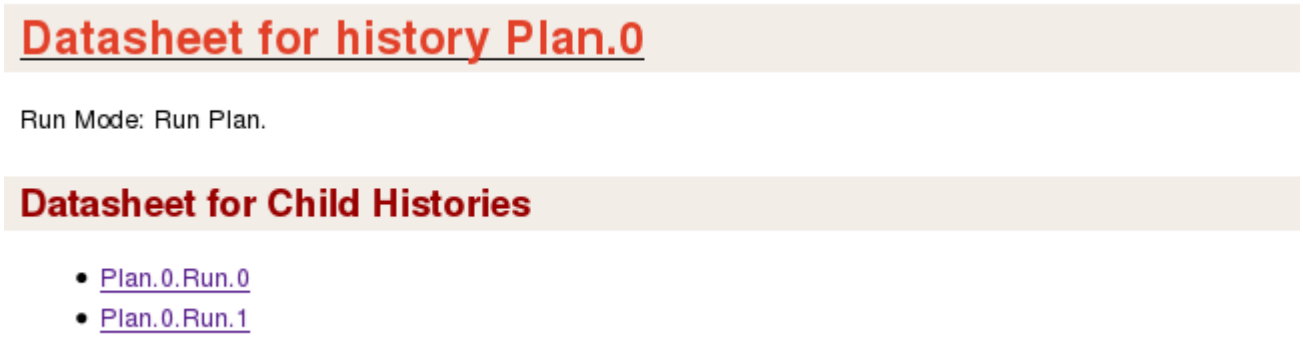
If the links to schematics are added in the *Tests Summary*, the schematic diagrams are saved as part of the datasheet, as shown below.



Use the navigator to traverse through the design hierarchy, or the Table of Contents to go to a particular schematic master.

© 2023 Cadence Design Systems, Inc.

The datasheet saved for a group history, for example, a history saved for a run plan, contains links to all child histories in that group run.



Location of Datasheet Files

The datasheet files are saved in a directory that has the same name as the name of the datasheet. For example, if you specified the name of the datasheet as `myDatasheet` in the Create Datasheet form, the datasheet files are saved in the `myDatasheet` directory in the location specified in the Create Datasheet form.

The datasheet directory contains the following files:

Table 23-1 Datasheet Files

File Name	Description
<datasheet_name>.xml	The main datasheet file. For example, if you specified the name of the datasheet as `myDatasheet` in the *Name* field, the main datasheet file will have the name `myDatasheet.xml`.

© 2023 Cadence Design Systems, Inc.

File Name	Description
<datasheet_name>_<test_name>.xml	Datasheet file for a test that is created if the *Detailed Results* or *Waveforms* check boxes are selected. For example, if you specified the name of the datasheet as myDatasheet in the *Name* field, and you have two tests named AC1 and TRAN1, the following two datasheet files that contain the detailed results information for each test are created if the *Detailed Results* or *Waveforms* check boxes are selected: ■ myDatasheet_AC1.xml ■ myDatasheet_TRAN1.xml
dsImg*.png	Portable Network Graphic (.png) files for the waveform images displayed in datasheets.

Colors Used to Display Status of Measured Values and Specifications in Datasheets

The following colors are used as the background color for cells in the *Minimum Value* and *Maximum Value* columns in the *Results Summary* section to display the status of measured values:

Cell Background Color	Description
Green	Indicates a pass status—all the measured values are within the limits defined by the specification.
Yellow	Indicates a near status—one or more measured values are no more than 10% outside the target value of the specification.
Red	Indicates a fail status—one or more measured values are greater than 10% outside the target value of the specification.

The following colors are used as the background color for cells in the *Value* and *Target* columns in the *Outputs* sub-section in the *Detailed Results* section to display the status of

© 2023 Cadence Design Systems, Inc.

measured values and corner specifications:

Cell Background Color	Description
Green (in *Value* column)	Indicates a pass status for the measured value—the measured value is within the limits defined by the specification.
Yellow (in *Value* column)	Indicates a near status for the measured value—the measured value is no more than 10% outside the target value of the specification.
Red (in *Value* column)	Indicates a fail status for the measured value—the measured value is greater than 10% outside the target value of the specification.
Green (in *Target* column)	Indicates a pass status for the corner specification—the measured value is within the limits defined by the specification.
Yellow (in *Target* column)	Indicates a near status for the corner specification—the measured value is no more than 10% outside the target value of the specification.
Red (in *Target* column)	Indicates a fail status for the corner specification—the measured value is greater than 10% outside the target value of the specification.
Gray (in *Target* column)	Indicates that a corner specification is disabled.

Displaying Customized Waveform Images in the Datasheet

If the setup in a history item is the same as the active setup, you can display customized waveform images for the history item by doing the following:

1. On the Results tab of the Outputs pane, click the button.

   Waveforms are displayed in waveform windows.
2. Customize the waveforms as required in the waveform window.
3. Create a datasheet without closing the waveform windows in which you customized the waveforms.

The customized waveforms are added to the datasheet.

© 2023 Cadence Design Systems, Inc.

Note: You can modify the resolution of the waveform images in the datasheet by overriding the `axlCustomSaveGraphImageArguments` SKILL function.

Opening a Datasheet

To open a datasheet, do one of the following:

- In the *Documents* tree on the Data View pane, double-click the HTML file for the datasheet.

  The HTML file for the datasheet has the same name as the name of the datasheet. For example, if you specified the name of the datasheet as `myDatasheet` in the Create Datasheet form, the HTML file for the datasheet will have the name `myDatasheet.html`.

- Open the `.xml` files in the datasheet directory in a web browser.

  For more information about the `.xml` files in the datasheet directory, see Location of Datasheet Files on page 1440.

Customizing the Datasheet Format and Structure

The following stylesheet files are used to control the format and structure of datasheets. These files are located at `<your_install_dir>/share/cdssetup/adexl`.

File Name	Description
datasheetMainDoc.xsl	XSLT stylesheet that controls the structure of the main datasheet file.
datasheetTestDoc.xsl	XSLT stylesheet that controls the structure of the datasheet files that contain the detailed results information for each test.
datasheet.css	CSS stylesheet that controls the formatting of the main datasheet file and the datasheet files that contain the results information for each test.

To customize the structure of the main datasheet page, do the following:

1. Copy the `datasheetMainDoc.xsl` file in the `<your_install_dir>/share/cdssetup/adexl` directory to a different location.

   Rename the file, if required.

© 2023 Cadence Design Systems, Inc.

2. Open the file in a text editor and make the required changes.
3. Use the `adexl.datasheet mainDocXSLFile` environment variable to specify the location of the file. For more information about this environment variable, see mainDocXSLFile.

To customize the structure of the datasheet pages for tests, do the following:

1. Copy the `datasheetTestDoc.xsl` file in the *<your_install_dir>*`/share/cdssetup/adexl` directory to a different location.

   Rename the file, if required.
2. Open the file in a text editor and make the required changes.
3. Use the `adexl.datasheet testDocXSLFile` environment variable to specify the location of the file. For more information about this environment variable, see testDocXSLFile.

To customize the formatting of the datasheet files, do the following:

1. Copy the `datasheet.css` file in the *<your_install_dir>*`/share/cdssetup/adexl` directory to a different location.

   Rename the file, if required.
2. Open the file in a text editor and make the required changes.
3. Use the `adexl.datasheet CSSFile` environment variable to specify the location of the file. For more information about this environment variable, see CSSFile.

© 2023 Cadence Design Systems, Inc.

24

Circuit Optimization

You can parameterize a design for sizing and optimization using variables. In addition, you can also use a parameterization flow in ADE Assembler. To use this flow, you can parameterize a design and set specifications, then perform a local or global optimization and backannotate the results of the optimization.

This chapter covers the following tasks:

- Parameterizing the Design
- Setting Up Specifications
- Running Optimization
- Sizing Over Corners
- Performing Manual Tuning

© 2023 Cadence Design Systems, Inc.

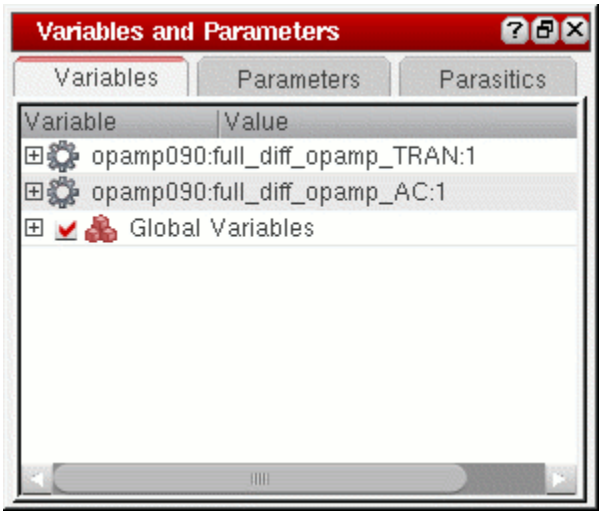
Parameterizing the Design

When you parameterize the design for optimization, you capture the critical device relationships in the circuit. For example, the input transistors of an opamp must be matched, and the transistors of a current mirror must be ratioed. In order to run optimization, you must define a range of legal values for optimization for each transistor that is the “master” in a matching relationship.

To begin parameterizing the design.

➤ Choose *Window — Assistants — Variables and Parameters*.

The Variables and Parameters pane appears.



Matching Devices and Device Properties

You have three options for matching. You can match all device properties for a device, or you can match only specific properties. You can also ratio-match. When you’re finished specifying device relationships, you can define legal values for device properties.

Matching Devices

To match devices:

1. Select the devices you want to match in the schematic.

   The devices are listed on the upper half of the Parameters tab of the Variables and Parameters assistant pane.

© 2023 Cadence Design Systems, Inc.

2. In the Parameters tab, select the device you want as the “master device.” The master device is the device that all the other selected devices will match.
3. Click the *Match Parameters* button.

   The matched parameters appear in the lower half of the Parameters tab of the Variables and Parameters assistant pane.

You can now modify the values of matched parameters to vary them during the optimization process.

Matching Device Properties

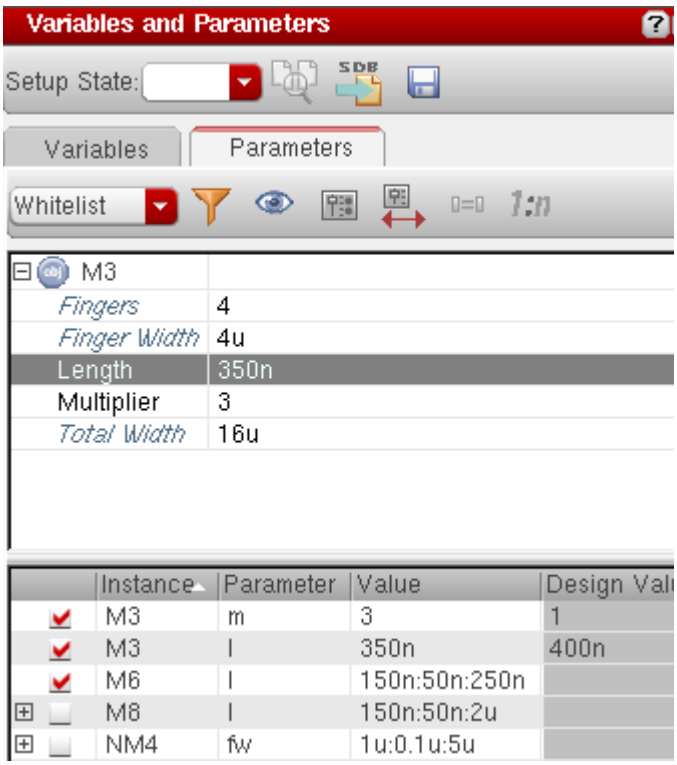
To match specific device properties:

1. In the schematic, click the devices for which you want to match properties.

   As you click, each device and its parameters appear on the upper half of the Parameters tab of the Variables and Parameters assistant pane.
2. Ensure that *Default* is selected in the drop-down list on the upper half of the Parameters tab of the Variables and Parameters assistant pane.
3. Click the + sign next to a device with whose properties you want to match the properties of other devices.

© 2023 Cadence Design Systems, Inc.

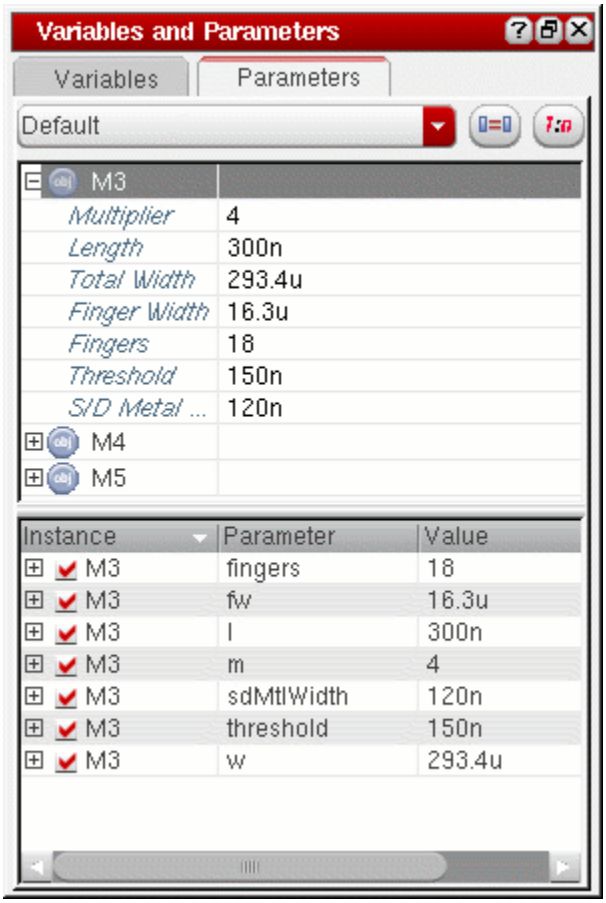
The list of properties for the device appears, as well as the schematic value of each property.



4. Select the property you want to match.
5. Click the *Match Parameters* button.

© 2023 Cadence Design Systems, Inc.

The matched parameters appear in the lower half of the Parameters tab.



You can now modify the values of matched parameters to vary them during the optimization process.

Ratio-Matching Device Properties

To ratio-match specific device properties:

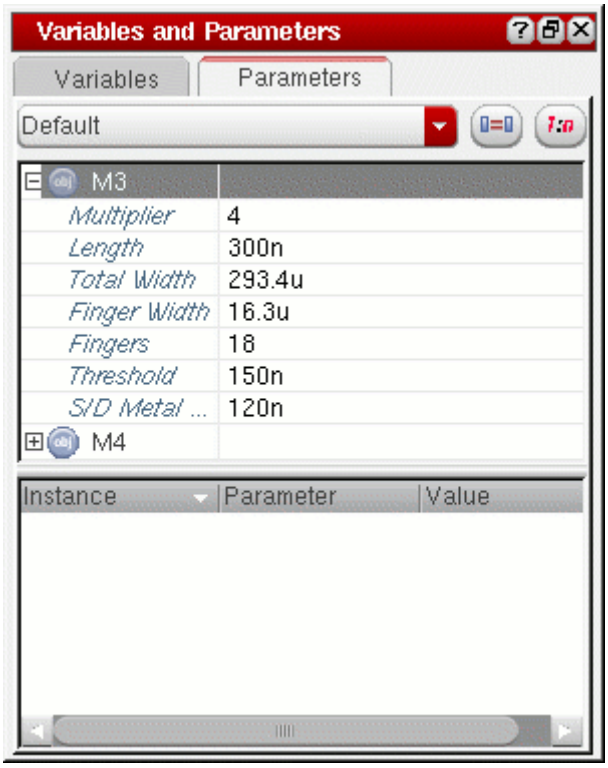
1. In the schematic, click the devices for which you want to ratio-match properties.

   As you click, each device and its parameters appear on the upper half of the Parameters tab of the Variables and Parameters assistant pane.

2. Ensure that *Default* is selected in the drop-down list on the upper half of the Parameters tab of the Variables and Parameters assistant pane.

© 2023 Cadence Design Systems, Inc.

3. Click the + sign next to a device with whose parameters you want to ratio-match the parameters of other devices.



4. Select the parameters you want to ratio-match.
5. Click the *Ratio Matched Parameters* button.

   ADE Assembler automatically ratios the parameters based on the lowest value.

   The ratio-matched parameters appear in the lower half of the Parameters tab.

You can now modify the values of ratio-matched parameters to vary them during the optimization process.

Defining Values for Optimization

To define a legal range of values for devices:

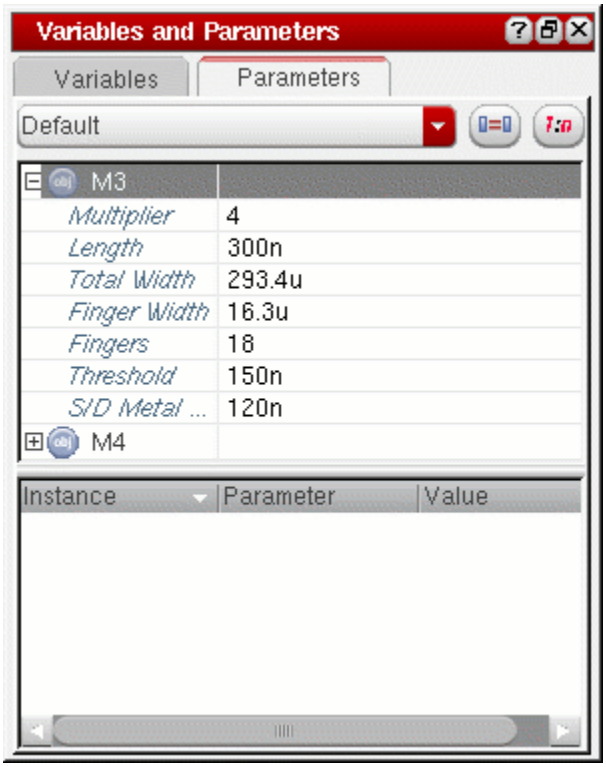
1. In the schematic, click the devices for which you want to specify values.

   As you click, each device and its parameters appear on the upper half of the Parameters tab of the Variables and Parameters assistant pane.

© 2023 Cadence Design Systems, Inc.

2. Ensure that *Default* is selected in the drop-down list on the upper half of the Parameters tab of the Variables and Parameters assistant pane.
3. Click the + sign next to a device whose parameters you want to modify for optimization.

   The list of parameters for the device appears, as well as the schematic value of each property.



4. Click the value for the parameters you want to modify and type a range or list of values. Specify ranges in the format `MIN:STEP:MAX`. You can separate a list of values with either commas or spaces.

   The modified parameters appear in the lower half of the Parameters tab.

© 2023 Cadence Design Systems, Inc.

Setting Up Specifications

When you set specifications on outputs, you establish the performance specifications that ADE Assembler must meet during synthesis. These specifications are used as benchmarks for candidate circuit quality during optimization.

You can set up general specifications, area constraint specifications and operating region specifications. For more information, see the following topics:

- General Specifications
- Device Area Constraint Specifications

General Specifications

For each specification, you specify an objective, or performance constraint, then specify a target value for that specification.

Objective Types

There are five types of objectives:

- open-ended
- close-ended
- range
- tolerance
- info

An open-ended specification is when you choose either minimize (min) or maximize (max), then specify an ideal value. For an open-ended specification, ADE Assembler tries to continually improve upon the ideal value. So, if the ideal value for the *ugf* specification is maximize to 1e+8, 3e+8 is better than 2e+8.

A close-ended specification is when you specify a target value, then specify that the resulting specification must be greater than (>) or less than (<) that target value. For a close-ended specification, ADE Assembler attempts to meet the target value, and exceeding the target is no better than meeting the target. So, if the target value for *ugf* is >1e+8, 3e+8 is no better than 2e+8.

© 2023 Cadence Design Systems, Inc.

You can also specify a range specification, in which you specify both an upper and a lower boundary for the target value. For a tolerance (tol) specification, you specify a target value and an allowable percentage deviation.

The optimizer treats the range and tolerance specs as closed-ended unless all the specifications are met. When all the specifications are met, range and tolerance are treated as open-ended. When all the specifications are met and the `all specs met` stopping criteria has not been specified, the optimizer continues to prefer the values closer to the midpoint of the range or tolerance target value.

You can specify an info specification when you want to see the value of a measurement, but do not want that specification to affect sizing.

Design specifications might include specifying an open loop gain maximized with an acceptable value of 60dB, or a settling time < 15ns.

Tip

You can also specify area goals. For more information, see Device Area Constraint Specifications.

Specification Weights

ADE Assembler supports weighing on specifications, that is, the ability to put particular emphasis on one or more specifications. You may use weighing if you have certain important specifications, and it is imperative that ADE Assembler meet those specifications, even at the expense of other specifications. Or, if you have run ADE Assembler without any specification weights, and you have one particular specification that is not being met, you can put emphasis on that specification in order to ensure that ADE Assembler meets it.

To set specification weighing, you specify a positive integer as the “weight.” ADE Assembler multiplies the cost function for that specification by the integer specified for the weight. Because this specification is now “x” times more important than other specifications, it changes the way ADE Assembler searches the design space. ADE Assembler will put particular stress on meeting that specification, even at the expense of specifications weighted at “1.” This discrepancy is especially true of specifications that are minimized or maximized, because ADE Assembler is already working harder to optimize the value for that specification.

In general, ADE Assembler is able to find a working solution without specification weighing. As a result, Cadence advises that you run ADE Assembler first with all specifications weights set to “1.” If ADE Assembler is repeatedly unable to meet a particular specification, you can put more emphasis on meeting that specification using specification weighing.

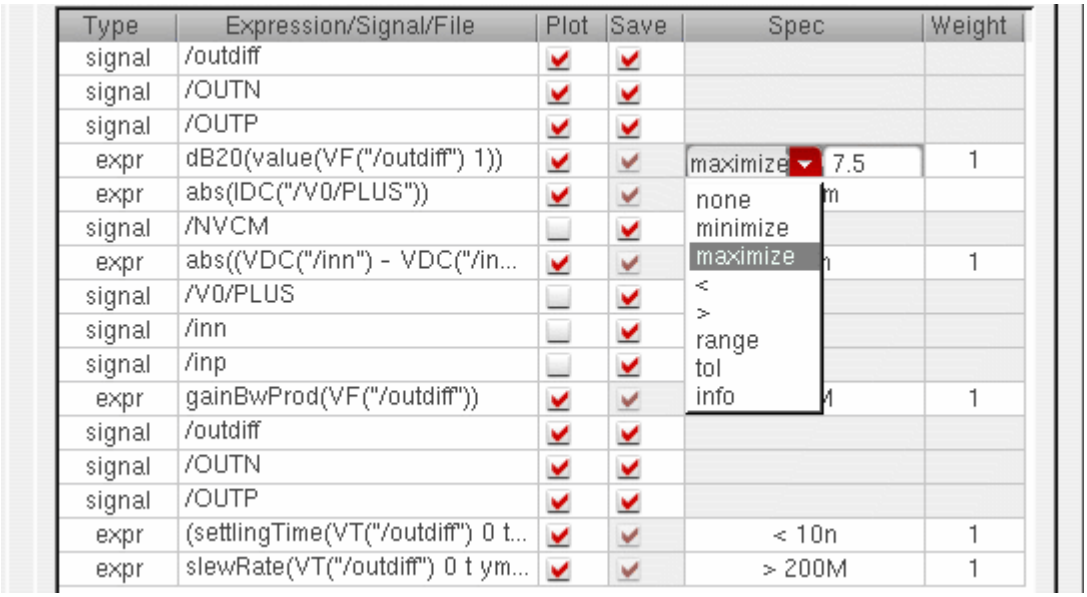
© 2023 Cadence Design Systems, Inc.

Setting Up Specifications

To set up specifications:

1. In the Outputs Setup tab, double-click the *Spec* column in the row for the output for which you want to set up specifications.

   A drop-down list of objectives appears.



2. Choose an objective from the drop-down list.

   For more information on the types of objectives, see Objective Types.

3. Select the field next to the drop-down list.

4. Enter a target value in the field.

5. (Optional) Specify a weight for the specification in the Weight column.

   For more information on weights, see Specification Weights.

6. Repeat step 1 through step 5 for all outputs for which you want to set specifications.

© 2023 Cadence Design Systems, Inc.

Maximizing Operating Region Specification

To maximize an operating point specification:

- Create the expression as normal ADE Assembler output specification.
- Perform the optimization.

Note: In this case, the special operating region type of specification is not used because the individual expressions are considered to pass the specification when the constraint is met. For example, if you define an expression, `vgs-vth maximize 20m`, it is considered pass for all the values above `20m`.

The special operating region specification performs the following tasks:

- Provides a single specification that determines pass or fail of operating region.
- Folds a long list of individual expressions into a single specification.
- Provides methods to simplify adding operating region constraints to multiple devices.
- Selects devices on schematic and apply constraints to multiple devices at the same time or selects devices by using master or model.
- Provides reusable constraints, which are a set of named expressions that can be loaded from file and applied to selected devices

If you want to maximize the operating region for a small number of devices, you can create expressions as normal output specifications. For a large number of devices, you can create close-ended expressions as part of the special operating region specification.

© 2023 Cadence Design Systems, Inc.

Device Area Constraint Specifications

You can specify area constraint specifications for use in optimization runs. For each device, you can either manually specify an area formula, or use a CDF parameter to set default area formulas for devices. For more information about using a CDF parameter to set default area formulas, see Setting Default Area Formulas.

Tip

You can add multiple area constraint specifications for a test if you want to set different specifications for some devices.

For more information, see the following topics:

- Setting Default Area Formulas
- Setting Up Area Constraint Specifications
- Modifying an Area Constraint Specification
- Copying a Device Formula
- Including Devices in the Area Constraint Specification
- Excluding Devices from the Area Constraint Specification
- Deleting Devices with Area Formulas

Setting Default Area Formulas

You can specify default area formulas for devices using the `AreaFormula` CDF parameter.

If you want to use a different CDF parameter name for specifying default area formulas, specify the parameter name using the *`defAreaProp`* environment variable in one of the following ways:

- In the CIW, enter the following:

  ```
  envSetVal("adexl.gui" "defAreaProp" 'string "<myParamName>")
  ```

- In your `.cdsenv` file, enter the following:

  ```
  adexl.gui defAreaProp string "<myParamName>"
  ```

For more information about CDF parameters, see the Component Description Format User Guide.

© 2023 Cadence Design Systems, Inc.

Setting Up Area Constraint Specifications

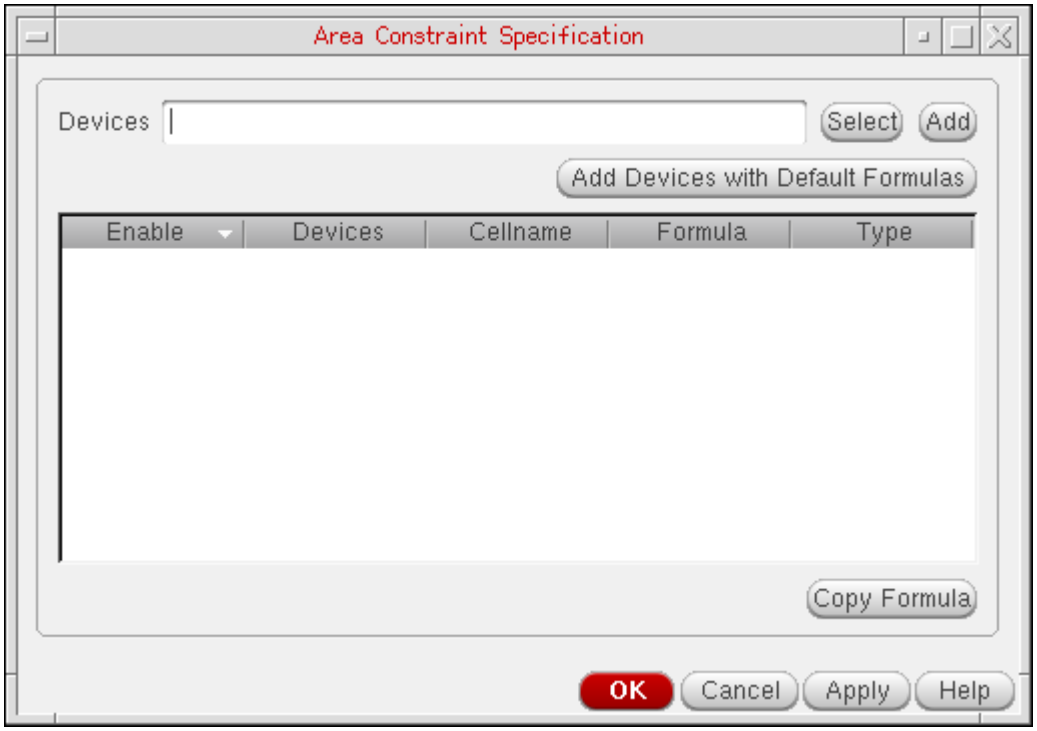
To set up area constraint specifications, do the following:

1. On Outputs Setup tab, click the *Add new output* button.
2. In the drop-down list, select a test and choose *Area Specification*.

Tip

You can also right-click a test name in the Outputs Setup tab and choose *Add Area Specification* to set up area constraint specifications for that test.

The Area Constraint Specification form appears.



3. Specify a device name for which you want to add an area formula:
   - Select a device in the schematic. To do this, click the *Select* button to display the schematic window. Select the device in the schematic window and press the *Esc* key. The device name is displayed in the *Devices* field.
   - Enter the device name in the *Devices* field.
4. Click *Add* to add the device to the list.
5. Select the *Formula* field for the device and enter an area formula.

© 2023 Cadence Design Systems, Inc.

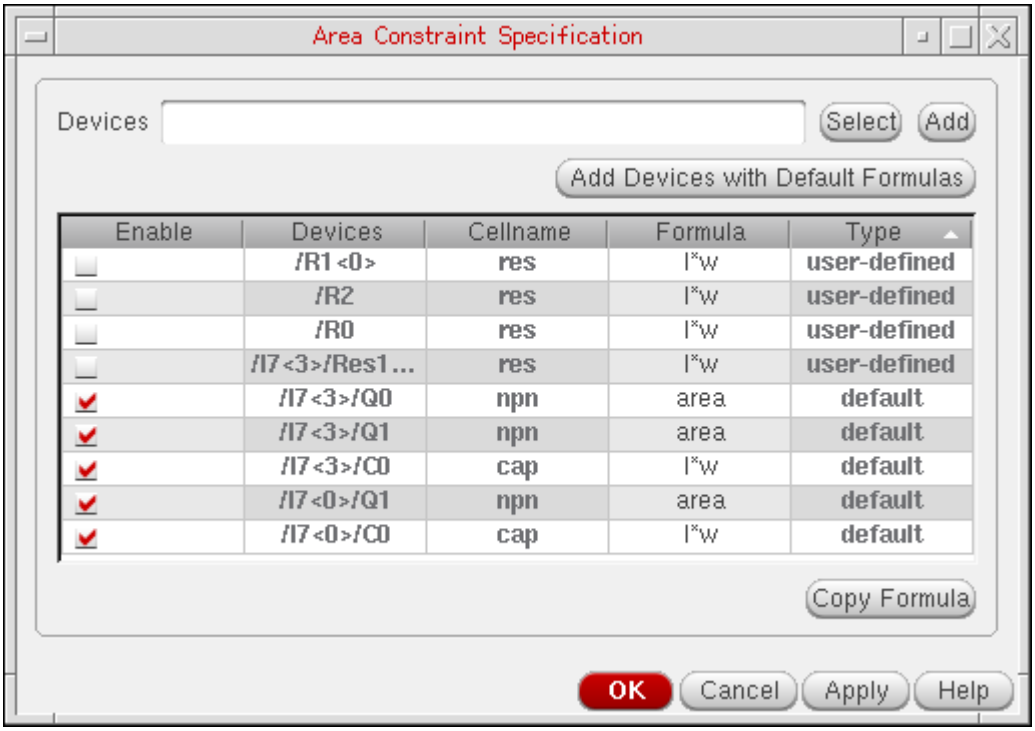
You must enter an arithmetic combination of valid CDF parameters that are specified for the device.

Tip

You can use the Copy Formula button to copy area formulas from one device to another.

6. Repeat step 3 through step 5 for any other devices you want to include in the area constraint specification.
7. Click the *Add Devices with Default Formulas* check box to add all instances in the design for which default formulas are specified using the `AreaFormula` CDF parameter. For more information, see Setting Default Area Formulas.

   Note: You can modify any default formula by selecting the *Formula* field and typing a new area formula.



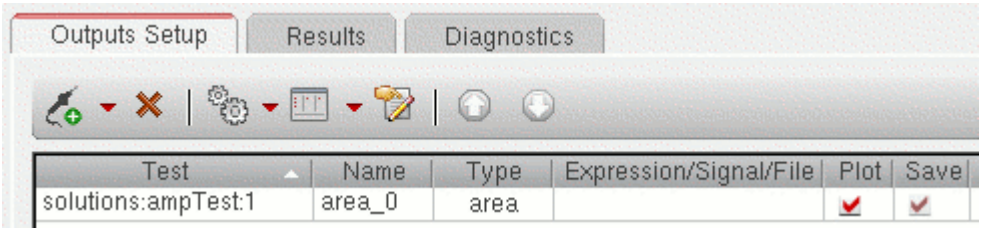
8. To include a device in the area constraint specification, select the *Enable* check box for the device. De-select the *Enable* check box for the devices you do not want to include in the area constraint specification.

   Note: All devices with default area formulas are included by default, while any devices for which you manually entered area formulas are not.

© 2023 Cadence Design Systems, Inc.

9. Click *OK*.

   The area constraint specification is displayed in the Outputs Setup tab.



   The area constraint specification is assigned the name `area_`*`seqNum`*, where *`seqNum`* is `0` (zero) for the first area constraint specification you add. You can double-click the *Name* field and modify the name.

10. Set up specifications for the area constraint specification. For more information, see Setting Up Specifications.

Modifying an Area Constraint Specification

To modify an existing area constraint specification:

1. In the Outputs Setup tab, double-click the *Details* field for the area constraint specification.

   The Area Constraint Specification form appears.

2. Modify the area constraint specification as required.

Copying a Device Formula

To copy the area formula from one device to another:

1. Select the device from which you are copying the formula.
2. Control-click the device to which you want to copy the formula.
3. Click *Copy Formula*.

Including Devices in the Area Constraint Specification

➡ To include a device in the area constraint specification, select the *Enable* check box.

To include more than one device at once:

© 2023 Cadence Design Systems, Inc.

1. Select the devices you want to include.
2. Right-click and choose *Enable*.

Tip

You can enable all devices by right-clicking and choosing *Enable All*.

Excluding Devices from the Area Constraint Specification

➡ To exclude a device from the area constraint specification, deselect the *Enable* check box.

To exclude more than one device at once:

1. Select the devices you want to exclude.
2. Right-click and choose *Disable*.

Tip

You can disable all devices by right-clicking and choosing *Disable All*.

Deleting Devices with Area Formulas

To delete devices and their area formulas:

1. Select one or more devices.
2. Right-click and choose *Delete*.

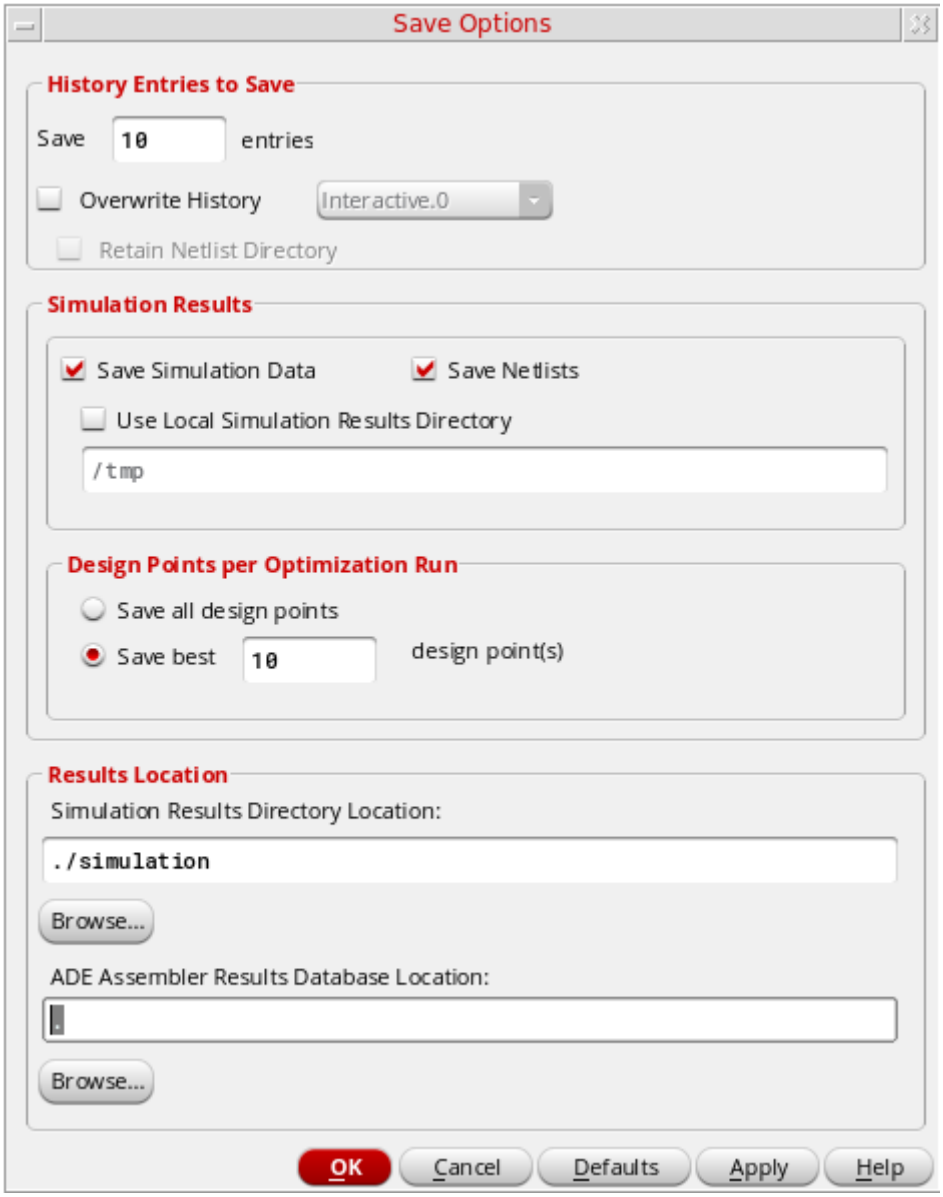
© 2023 Cadence Design Systems, Inc.

Specifying How Much Optimization Data to Save

To specify the number of design points for which you want to save simulation data, do the following:

1. In the ADE Assembler session window, choose *Options – Save*.

   The Save Options form appears.



© 2023 Cadence Design Systems, Inc.

2. In the *Design Points per Optimization Run* group box, select one of the following radio buttons:

Radio Button	Description
Save all design points	Saves data from all design points
Save best	Saves data from the specified number of best design points

3. (Optional) In the *Results DB* group box, either type the directory path where you want the program to write your setup and results database information in the *Results DB location* field, or do the following:

   a. Click *Browse*.

   b. On the form that appears, navigate to and select the directory where you want the program to write your setup and results database information.

   The program writes setup and results database information to *libraryName*/*cellName*/adexl in the specified directory.

   If you do not specify a setup and results database location, the program writes this information to *libraryName*/*cellName*/adexl for the current test. If your design library is set up as read-only, you can use this field to specify a writable location.

4. Click *OK*.

   The program applies the settings you specified.

See also

- Data Points—Definition
- Design Points—Definition
- Specifying Results Database Location

© 2023 Cadence Design Systems, Inc.

Data Points—Definition

A data point represents one simulation run with one set of parameter values and corners setup. For example, if you sweep *CAP* (a global variable) from `600p` to `800p` with a step value of `3`, you will have three data points. If you sweep more than one variable, each unique combination of values constitutes one data point. If you have two corners for temperature at `0` and `30` in addition to the *CAP* sweep through three values, you will have six data points:

Data Point	*CAP*	Corner Temperature
1	`600p`	`0`
2	`700p`	`0`
3	`800p`	`0`
4	`600p`	`30`
5	`700p`	`30`
6	`800p`	`30`

Design Points—Definition

A design point consists of the set of data points that represents one sweep value across all corners. For example, if you have two corners for temperature at `0` and `30`, and a sweep of *CAP* (a global variable) through three values (`600p`, `700p`, `800p`), each sweep across both corners constitutes a design point.

Design Point	*CAP*	Corner Temperatures
1	`600p`	`0`
	`600p`	`30`
2	`700p`	`0`
	`700p`	`30`
3	`800p`	`0`
	`800p`	`30`

© 2023 Cadence Design Systems, Inc.

How Top Ten Best Design Points are Calculated Using Running Over Corners?

The optimizer treats range and tolerance specs as closed-ended unless all specs are met. If all specs are met, range and tolerance are treated as open-ended and optimization attempts to find values that are closer to the target yield.

The design point that has the yield value closer to the target yield is considered as the best point and placed at the top of the table. For example, consider the following case where target yield is 1.443, point A has yield value=1.267, and point B has yield value=1.265. In this case, point A is the best point because it is closer to the target than point B.

Points are also ordered in terms of the total error generated for all specs for each point. The point that has lesser error value is considered as the best point.

Total error in point A = 1.332
Total error in point B = 1.452

Clearly, point A is better than point B. You can calculate the total error of all the specs in a point by summing the absolute value of the result minus the target.

Points for which tolerance spec is passed are treated as equally good points unless all the specs are passed. You can compute the total error considering only those values for which the tolerance specs fail. You can also check how many specs are failed for each point. If some points have more failures as compared to the others, then those points are placed towards the end of the best points list.

For example, below list shows the total number of failures and total error for 10 design points that are arranged in best to worst order:

1. Number of failures = 4, Total sum of failing difference from target = 6.09u
2. Number of failures = 4, Total sum of failing difference from target = 9.06u
3. Number of failures = 5, Total sum of failing difference from target = 5.65u
4. Number of failures = 5, Total sum of failing difference from target = 5.773u
5. Number of failures = 5, Total sum of failing difference from target = 5.787u
6. Number of failures = 5, Total sum of failing difference from target = 5.799u
7. Number of failures = 5, Total sum of failing difference from target = 5.844u
8. Number of failures = 5, Total sum of failing difference from target = 5.849u
9. Number of failures = 5, Total sum of failing difference from target = 5.882u
10. Number of failures = 5, Total sum of failing difference from target = 5.936u

© 2023 Cadence Design Systems, Inc.

Notice that the first two points have a larger total, but they have 4 failures as compared to other points that have 5 failures. It has been concluded from this example that number of passes overrides the total number of errors while calculating the best design points.

Error Conditions

ADE Assembler shows error messages in the following situations:

- NFS file intermittent problem.
- File permission changes to read-only when write permission was expected/required.
- File has been deleted manually.

The errors displayed are:

```
ERROR: Encountered the following problem with the results database engine:
SQL statement:
Error code: 1
Description: unable to open database file.
The results database uses SQL statements internally.
Contact Cadence Customer Support to report the problem.
ERROR: Received a database engine error: unable to open database file All runs have
been stopped because ADE Assembler can no longer guarantee the correctness of
results written to the results database. The results tab corresponding to this run
has been closed. Common reasons for this error:
1. Physical disk full or quota limit reached.
2. The results database file has been deleted or is no longer writable.
3. File system internal I/O error.
ERROR: ADE Assembler session has recovered from a serious error: 'unable to open
database file' and may become unstable. Save your work and restart the session.
```


© 2023 Cadence Design Systems, Inc.

Running Optimization

You have three options for optimization:

Local optimization runs the schematic design, then deterministically improves that design. Local optimization is useful if:

- You have a starting design that you want to improve.
- A design meets specifications at the nominal corner but not across all corners.
- Models in a PDK have changed just enough to cause the design to no longer meet all specifications.

Local optimization searches the design space only around the specified point, so ADE Assembler may not locate the best point possible to meet the design specifications. In addition, local optimization does not guarantee small perturbations to your variable values.

Unlike local optimization, global optimization efficiently searches over all of the variables defined to find a good solution for your design.

Tip

If you have a large number of corners, you may want to run Size Over Corners instead of running global or local optimization. For more information, see Sizing Over Corners.

Note: To perform an optimization, ensure that the design meets the following requirements:

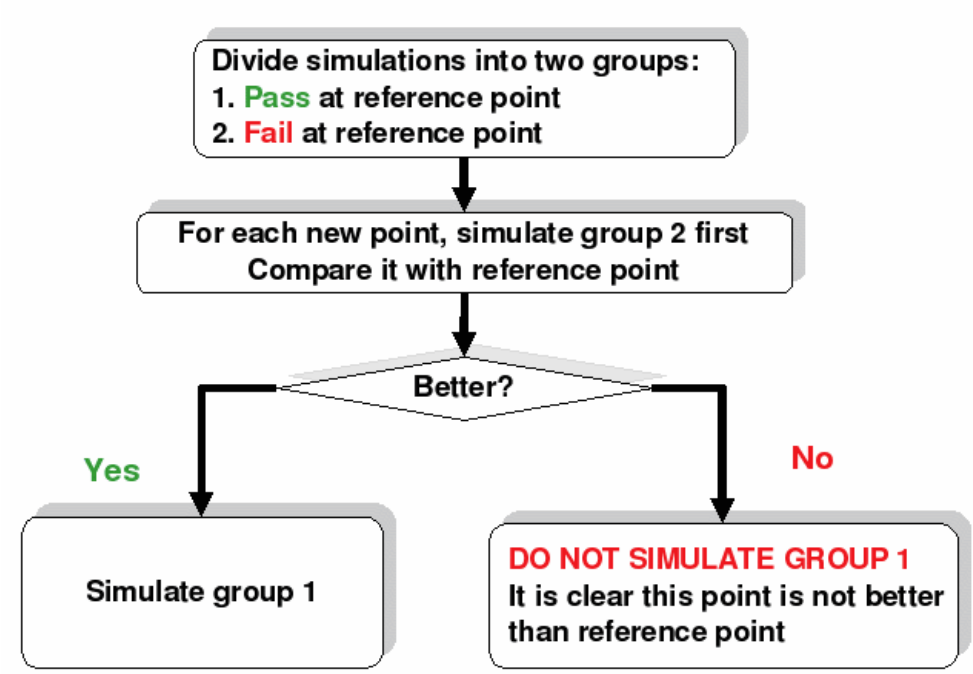
- Specification targets are defined
- Specification output expressions are scalar results and do not include waveforms.

Understanding Conditional Evaluation

As ADE Assembler is running the simulations for each design point, it may encounter a point where the first simulation run is not as good as the current best design point. In this case, ADE Assembler will not run any remaining simulations in order to more quickly and effectively reach a solution. This type of evaluation is called conditional evaluation, and may result in only partially-evaluated points.

© 2023 Cadence Design Systems, Inc.

When sizing a design with conditional evaluation set, ADE Assembler follows the following procedure:



The “No” case results in a partially-evaluated point.

Algorithms of Local Optimization

You can perform local optimization by using any of the four algorithms: *Brent-Powell*, *Hooke-Jeeves*, *BFGS*, and *Conjugate Gradient*. You can consider the following points while making a choice of an algorithm to be used to run local optimization:

- The *BFGS* and *Conjugate Gradient* algorithms are gradient-based algorithms. These methods do well when gradient information is available, that is, when specification measurements return more than two or three states and they do not result in evaluation errors.

  BFGS uses second order Hessian Matrix to search local optimum, which is much more efficient than Conjugate Gradient if the performance space is closer to quadratic. Therefore, it is recommended to use the BFGS algorithm in cases where performance gradient can be calculated and design variables are naturally continuous.

- The *Brent-Powell* algorithm searches in a fine grid around the starting point. Use the Brent-Powell option if your starting point is already close (e.g. only a few specifications not met). With this type, ADE Assembler should find the local minimum around the

© 2023 Cadence Design Systems, Inc.

selected point, but it may also quit once that local minimum is found and then miss a more optimal solution in the local design space. Because it only runs one point at a time, ADE Assembler is slow with this option, and it runs the least number of points on average.

- The type *Hooke-Jeeves* algorithm searches in bigger steps, but it can miss a local minimum that might have been found by the *fine* option. Use the *Hooke-Jeeves* option if your time is limited and the starting point is not as close. With this option, ADE Assembler is less likely to quit prematurely, but it also may miss a local minimum that the *Brent-Powell* option would have found. With the *Hooke-Jeeves* option, ADE Assembler runs more points, but it is faster on multiple computers than when running with the *Brent-Powell* option because of parallel points.

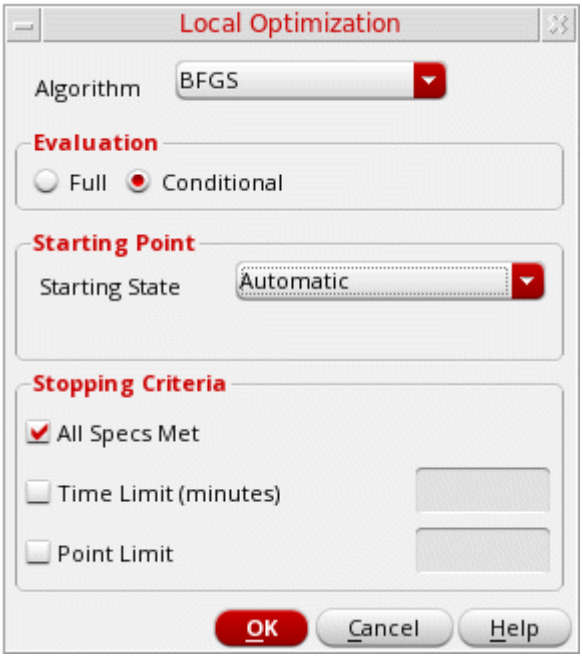
© 2023 Cadence Design Systems, Inc.

Running a Local Optimization

To run local optimization:

1. From the *Select a Run Mode* drop-down list on the Run toolbar, choose *Local Optimization*.
2. Click *Simulation Options* to specify optimization options.

   The Local Optimization Options form appears.



3. From the *Algorithm* drop-down list, select the algorithm for local optimization.

   The following algorithms are supported:

   - Conjugate Gradient
   - Brent-Powell
   - Hooke-Jeeves
   - BFGS

4. Select an evaluation type by selecting one of the following radio buttons:

   - Full
   - Conditional

© 2023 Cadence Design Systems, Inc.

For more information on conditional evaluation, see Understanding Conditional Evaluation.

5. In the *Starting State* drop-down list, select a setup state that defines a set of fixed values for each global variable or device parameter that uses a range of values in the active setup. This drop-down list can contain the following values:

   - `Automatic`: It refers to the default way in which the values for parameters and global variables are set for a starting point. The parameters with range values are replaced with their values in the schematic. Parameters with scalar values or a reference to another parameter remain unchanged. Global variables with range values are replaced with the mid-point of the given range.

     If the value of a parameter in the schematic is out of the optimization value range, the actual value used for the starting point is changed to a value from the optimization range. For example, if a parameter is set to a range of `100u:10u:200u`, but the value of this parameter in the schematic is `20u`, the starting point would be set to `20u`. However, for the actual optimization run, the value of this parameter would be set to `100u`.

     When you select this value, ADE Assembler internally creates a state named `automaticStartingPoint`, which replaces each parameter and global variable range as described above. `automaticStartingPoint` is visible in the Load State form, *Setup States* tree in the Data View assistant, and the *Setup State* field in the Variables and Parameters assistant. ADE Assembler does not allow you to save any changes in `automaticStartingPoint` because they would be overridden whenever you choose `Automatic` as a starting state. To reuse and customize `automaticStartingPoint`, you can load the state, modify the parameters or global variables and save those in another state, which you can choose in the *Starting State* drop-down list for the next optimization run.

   - Names of the setup states you created, if any, to save the values of variables and parameters. For details, refer to Saving Parameters and Global Variables to a Saved State.

     Note: A saved stated to be used as a starting point should not have sweep values in variables and parameters.

   - A setup state named `ref_point_migrated`, which was created if your cellview earlier used a reference point. If it exists, it is automatically selected.

6. If desired, select the criteria for the duration of time for which the local optimization should run. You can select one or more of the following:

   - To run only until all goals are met, select the *All Specs Met* check box.

© 2023 Cadence Design Systems, Inc.

- To set a time limit for the run, select the *Time Limit* check box and enter a value in minutes.
- To set a limit in the number of points run, select the *Point Limit* check box and enter the number of points.

7. Click *OK*.
8. Click the *Run Simulation* button on the Run toolbar to optimize the circuit.

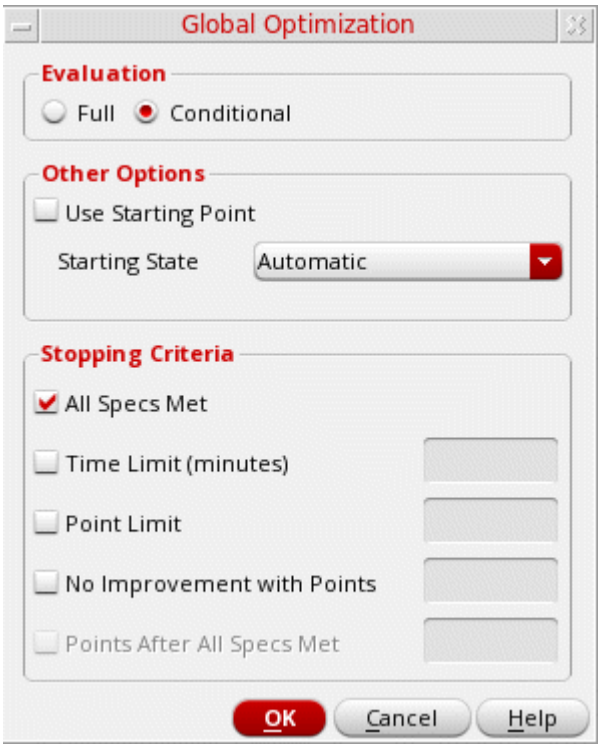
© 2023 Cadence Design Systems, Inc.

Running a Global Optimization

To run a global optimization:

1. From the *Select a Run Mode* drop-down list on the Run toolbar, choose *Global Optimization*.
2. Click *Simulation Options* to specify optimization options.

   The Global Optimization form appears.



3. Select an evaluation type by selecting one of the following radio buttons:
   - ❑ Full
   - ❑ Conditional

   For more information on conditional evaluation, see Viewing the Variable Data from Optimization Results.
4. If you want to specify a starting point for the simulation run, select the *Use Starting Point* check box in the *Other Options* panel.

© 2023 Cadence Design Systems, Inc.

5. In the *Starting State* drop-down list, select a setup state that defines a set of fixed values for each global variable or device parameter that uses a range of values in the active setup. For more details, refer to Starting State.

6. If desired, select criteria for the length of time local optimization should run.

   You can select one or more of the following:

   - To run only until all goals are met, select the *All Specs Met* check box.
   - To set a time limit for the run, select the *Time Limit* check box and enter a value in minutes.
   - To set a limit in the number of points run, select the *Point Limit* check box and enter the number of points.
   - To stop sizing when no improvement is seen for a certain number of points, select the *No Improvement with Points* check box and enter the number of points.
   - To continue exploring the design space for a better solution even all specifications are met, select the *Points After All Specs Met* check box and enter the number of points to explore.

Tip

You cannot select both the *All Specs Met* and *Points After All Specs Met* check boxes, as these two options are mutually exclusive.

7. Click *OK*.

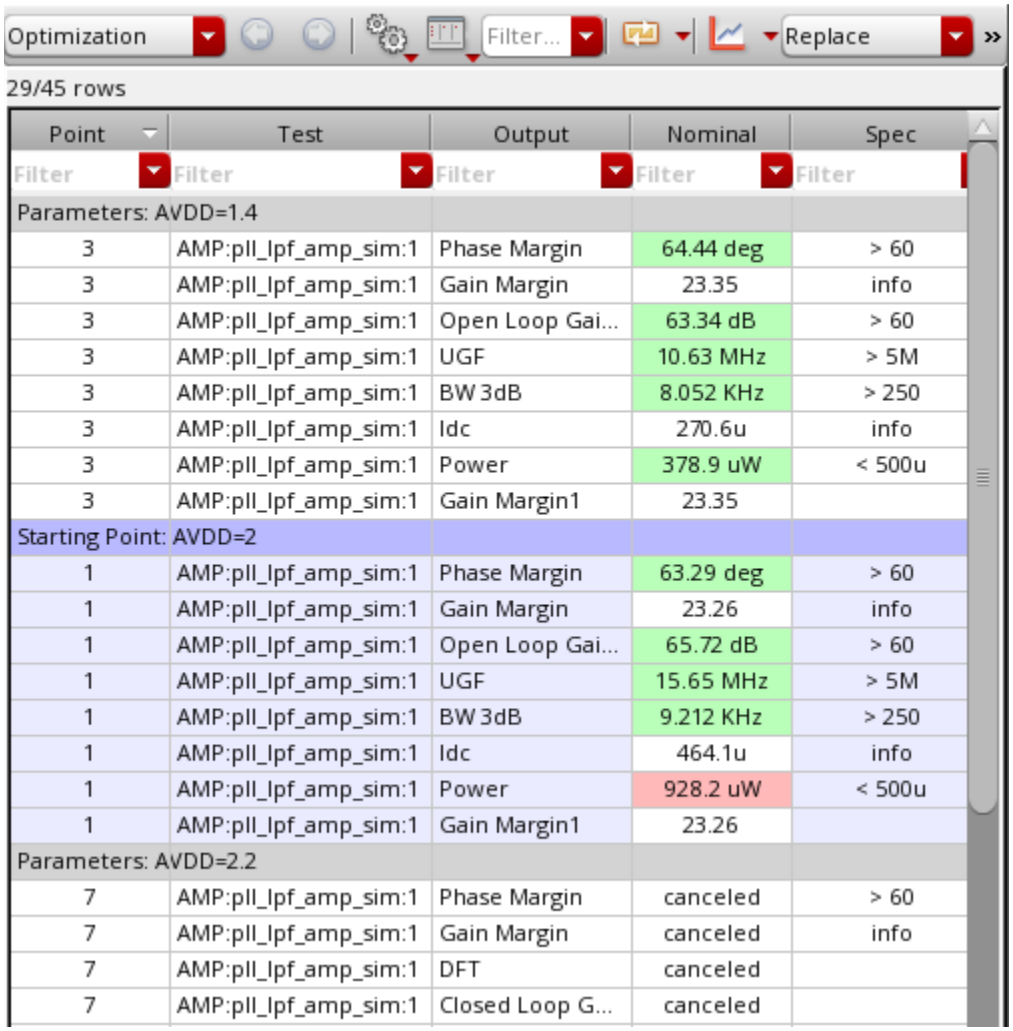
8. Click the *Run Simulation* button on the Run toolbar to optimize the circuit.

© 2023 Cadence Design Systems, Inc.

Viewing the Variable Data from Optimization Results

The *Optimization* results view displays all the possible design points for which simulation can be run. The starting point is highlighted in blue and the simulation continues for the remaining points until the stopping criteria is met. After the stopping criteria is met, the simulation for the remaining points is canceled.

In the following example, the design point with `AVDD=2` is the starting point. Next, ADE Assembler runs simulation for the point with `AVDD=1.4` where the stopping criteria is met, and therefore, the simulation for the point with `AVDD=2.2` is canceled.



Point	Test	Output	Nominal	Spec
Filter	Filter	Filter	Filter	Filter
Parameters: AVDD=1.4				
3	AMP:pll_lpf_amp_sim:1	Phase Margin	64.44 deg	> 60
3	AMP:pll_lpf_amp_sim:1	Gain Margin	23.35	info
3	AMP:pll_lpf_amp_sim:1	Open Loop Gai...	63.34 dB	> 60
3	AMP:pll_lpf_amp_sim:1	UGF	10.63 MHz	> 5M
3	AMP:pll_lpf_amp_sim:1	BW 3dB	8.052 KHz	> 250
3	AMP:pll_lpf_amp_sim:1	Idc	270.6u	info
3	AMP:pll_lpf_amp_sim:1	Power	378.9 uW	< 500u
3	AMP:pll_lpf_amp_sim:1	Gain Margin1	23.35	
Starting Point: AVDD=2				
1	AMP:pll_lpf_amp_sim:1	Phase Margin	63.29 deg	> 60
1	AMP:pll_lpf_amp_sim:1	Gain Margin	23.26	info
1	AMP:pll_lpf_amp_sim:1	Open Loop Gai...	65.72 dB	> 60
1	AMP:pll_lpf_amp_sim:1	UGF	15.65 MHz	> 5M
1	AMP:pll_lpf_amp_sim:1	BW 3dB	9.212 KHz	> 250
1	AMP:pll_lpf_amp_sim:1	Idc	464.1u	info
1	AMP:pll_lpf_amp_sim:1	Power	928.2 uW	< 500u
1	AMP:pll_lpf_amp_sim:1	Gain Margin1	23.26	
Parameters: AVDD=2.2				
7	AMP:pll_lpf_amp_sim:1	Phase Margin	canceled	> 60
7	AMP:pll_lpf_amp_sim:1	Gain Margin	canceled	info
7	AMP:pll_lpf_amp_sim:1	DFT	canceled	
7	AMP:pll_lpf_amp_sim:1	Closed Loop G...	canceled	

The optimization results view hides certain outputs, for example, expressions resulting in waveforms, which are not related to the spec. In this case, it shows the number of output rows displayed out of the total number of output rows, `29/45` in this example, above the table.

© 2023 Cadence Design Systems, Inc.

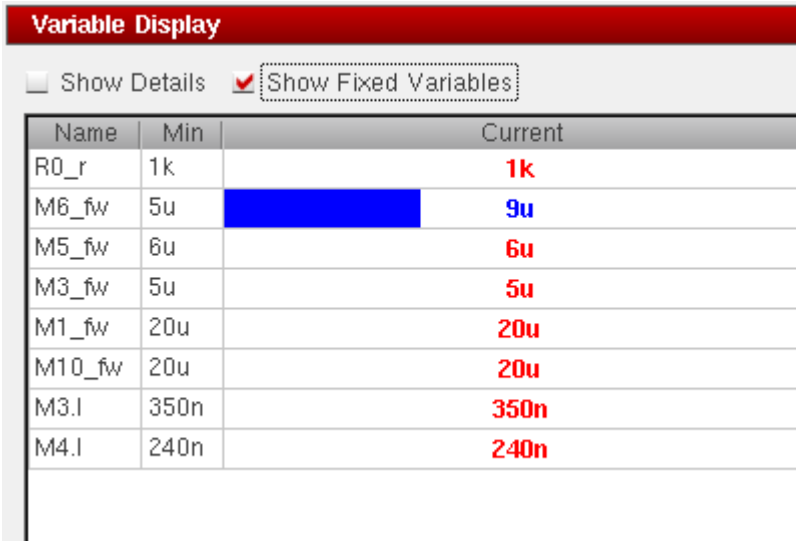
The Variable Display assistant also displays, for a selected design point, the minimum and maximum values for each variable. During optimization, this assistant shows the values for the current best point found, while after optimization, the assistant displays the values for the best point found during the optimization process.

In addition, ADE Assembler also displays the current variable values for the selected point, and, through a status bar, shows how far from the minimum and maximum values the current value is. A min or max value in **red** indicates that the current value is pushing the minimum or maximum and may need adjustment for synthesis to be completely successful.

To display the Variable Display Assistant:

➤ Choose *Window – Assistants – Variable Display*.

The Variable Display Assistant appears.



Tip

You can view more details about the variable, including the lib, cell, and view, by selecting the *More Details* check box.

© 2023 Cadence Design Systems, Inc.

Sizing Over Corners

While you can always size with all corners enabled using local or global optimization, it may not be efficient to do so with a large number of corners.

The Sizing Over Corners feature is an intelligent algorithm designed to optimize the test benches over a large number of corners. It does this by running a single point over all user-defined corners, determining the worst-case corners, using the worst-case corners for optimization and improving those corners on each iteration.

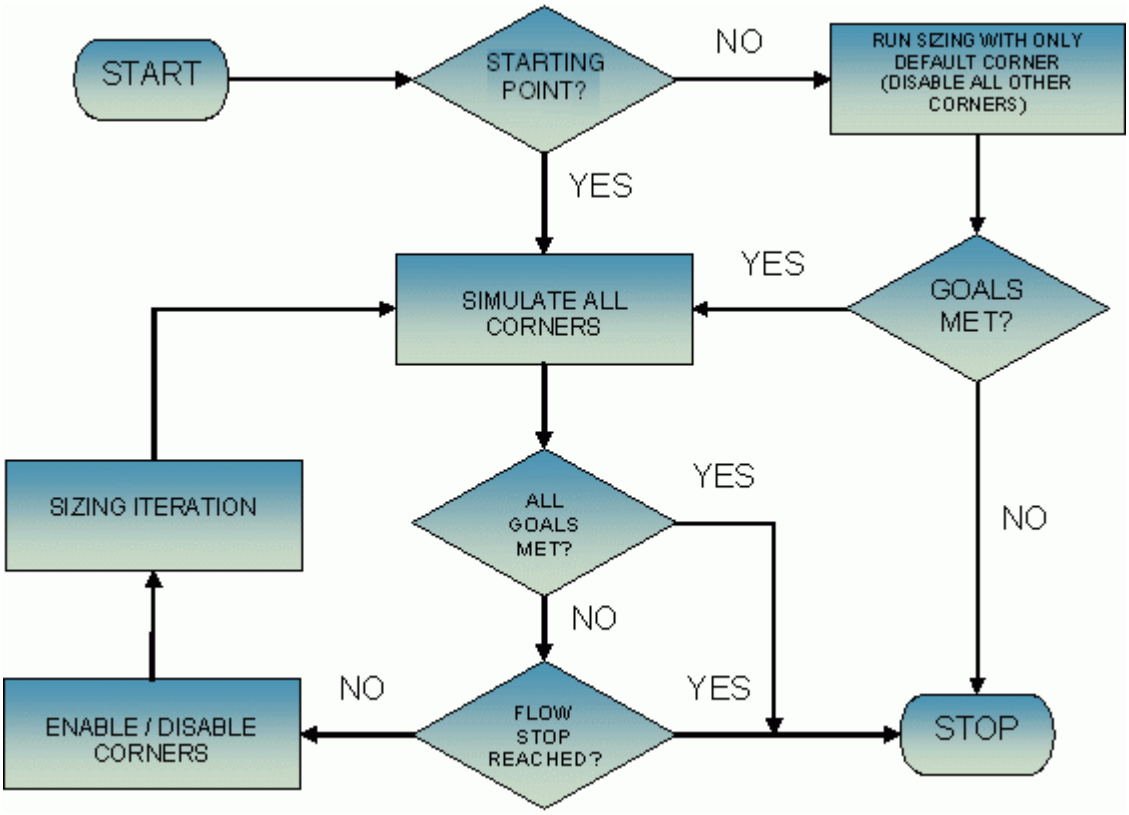
You can either provide a starting point, or ADE Assembler performs an initial global optimization on a specific corner and uses its best point as the starting point for the run. ADE Assembler simulates all corners to determine the worst case performance corners for each goal. ADE Assembler then resizes, taking into account the identified worst case performance corners. This process repeats until the stopping criteria for your size over corner run are met—either until all goals are met, or until it has run through the specified number of iterations or time limit.

Note: If the run stops due to the number of iterations or time limit being met, a final corner simulation is run if the last history children item for the run is related to sizing/optimization.

Figure 24-1 illustrates the sizing over corners flow.

© 2023 Cadence Design Systems, Inc.

Figure 24-1 Flow for Sizing Over Corners



To size over corners, do the following:

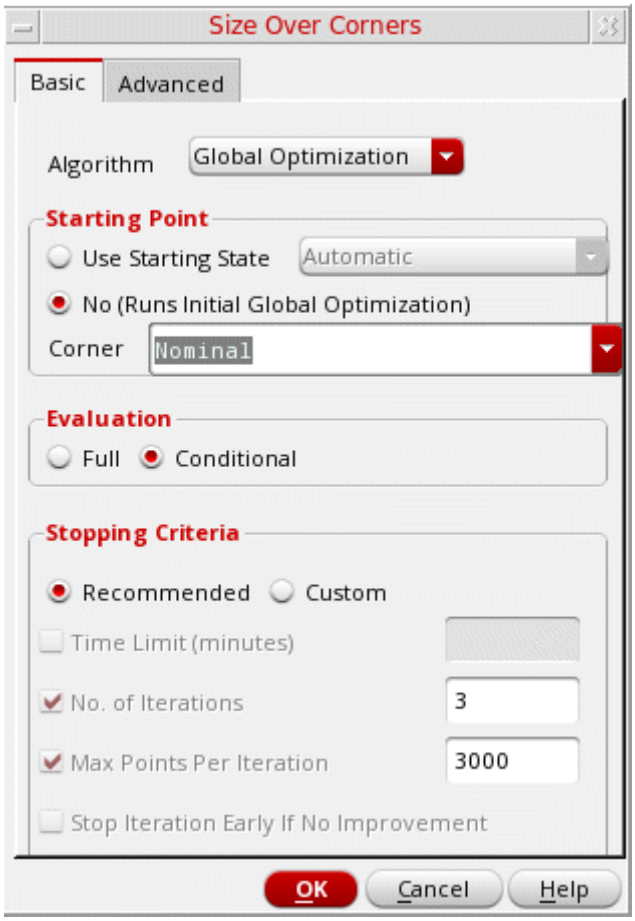
1. From the *Run* menu, select *Size Over Corners*.

Tip

Alternatively, select *Size Over Corners* in the *Select a Run Mode* drop-down list on the Run toolbar, then click the *Simulation Options* button on the Run toolbar.

© 2023 Cadence Design Systems, Inc.

The Size Over Corners form appears.



2. From the *Algorithm* drop-down list on the Basic tab, select the algorithm for sizing over corners.

   The default algorithm is *Global Optimization*.

3. In the *Starting Point* panel, select one of the following options to specify a starting point for the simulation run:

   - *Use Starting State*—Select this option if you have created a setup state and want to use that as the starting point for the run. To use the setup state as the starting point for the optimization, select a setup state that defines a set of fixed values for every global variable or parameter that defines a range of values in the active setup.

     For information about the possible values of this list, refer to Starting State.

   - *No (Runs Initial Global Optimization)*—Select this option if you do not have a starting state to be used as a starting point for sizing. When this option is selected, ADE Assembler performs an initial global optimization on the corner specified in the

© 2023 Cadence Design Systems, Inc.

Corner drop-down list and uses its best point as the starting point for the run. If you select *No (Runs Initial Global Optimization)*, ensure that:

- The specified corner is not disabled in the Run Summary pane.

  For more information, see the Disabling and Enabling the Nominal Corner.

- For tests that are enabled in the Data View pane, the specified corner is not disabled in the Corners Setup form.

  For more information, see Disabling and Enabling the Nominal Corner for Specific Tests.

4. In the *Evaluation* group box, select one of the following evaluation types for the sizing run:

   - Full
   - Conditional

   For more information on conditional evaluation, see Understanding Conditional Evaluation.

5. Select the *Recommended* option under *Stopping Criteria* to use the recommended criteria for the length of time size over corners should run. The recommended criteria are:

   - Three sizing iterations
   - 3000 points run per iteration

   If you want to modify these defaults, select the *Custom* option, then select one or more of the following stopping criteria:

   - To set a time limit for the run, select the *Time Limit (minutes)* check box and enter a value in minutes.
   - To specify the number of sizing iterations, select the *No. of Iterations* check box and enter the number in the field.
   - To specify the maximum number of points processed per iteration, select the *Max Points per Iteration* check box and enter the number of points in the field.
   - To stop the process early if the sizing results in no improvement, select the *Stop Iteration Early if No Improvement* check box. This option is applied to each optimization iteration. This stopping criteria is similar to the *No Improvement with Points* stopping criteria for the Global Optimization run mode for which the user also specifies the number of points. In the case of iterative run modes, the number of points is calculated as `Max Points per Iteration / 3`.

© 2023 Cadence Design Systems, Inc.

Note: If all goals are met prior to the stopping criteria being reached, ADE Assembler will stop the sizing run.

6. (Optional) By default, the Size Over Corners run is automatically stopped when all the goals are met. If the test outputs have minimize or maximize specs defined for them, you can choose to automatically run the local optimization after the best design point is found. This helps in getting better results for these types of specifications.

   For this, open the Advanced tab.

   Select the *Run Local Optimization After All Spec Met* option. This specifies that local optimization needs to be run after all the specs are met. The other options on this tab are used by the local optimization run. If required, change the default values.

7. Click *OK* to save the changes and close the Size Over Corners form.

8. Click the *Run Simulation* button on the Run toolbar.

© 2023 Cadence Design Systems, Inc.

ADE Assembler runs through iterations of sizing and corner sweeping until your stopping criteria are met.

© 2023 Cadence Design Systems, Inc.

Performing Manual Tuning

While parameterizing a design, you can tune your design by varying the values of parameters, running multiple simulations and then comparing results. You can do this by using the Manual Tuning run mode.

Some of the important points related to the Manual Tuning run mode are as follows:

- You run the Single Run, Sweeps and Corners, Monte Carlo, Global Optimization, and Local Optimization run modes multiple times.

  Currently, the Manual Tuning run mode is not supported in the Improve Yield, Sensitivity Analysis, High Yield Estimation, Create Worst Case Corners, Size Over Corners run modes.

- Every time you tune the design parameters and run a simulation, results are appended to the same history checkpoint. This is as compared to the other run modes in which the results of every simulation run are saved in a new checkpoint.

- You can compare the results data in a single table and plot specifications across design points run in different simulations. Therefore, manual tuning, when used with design parameterization, helps you tune your design without changing the schematic.

To run manual tuning:

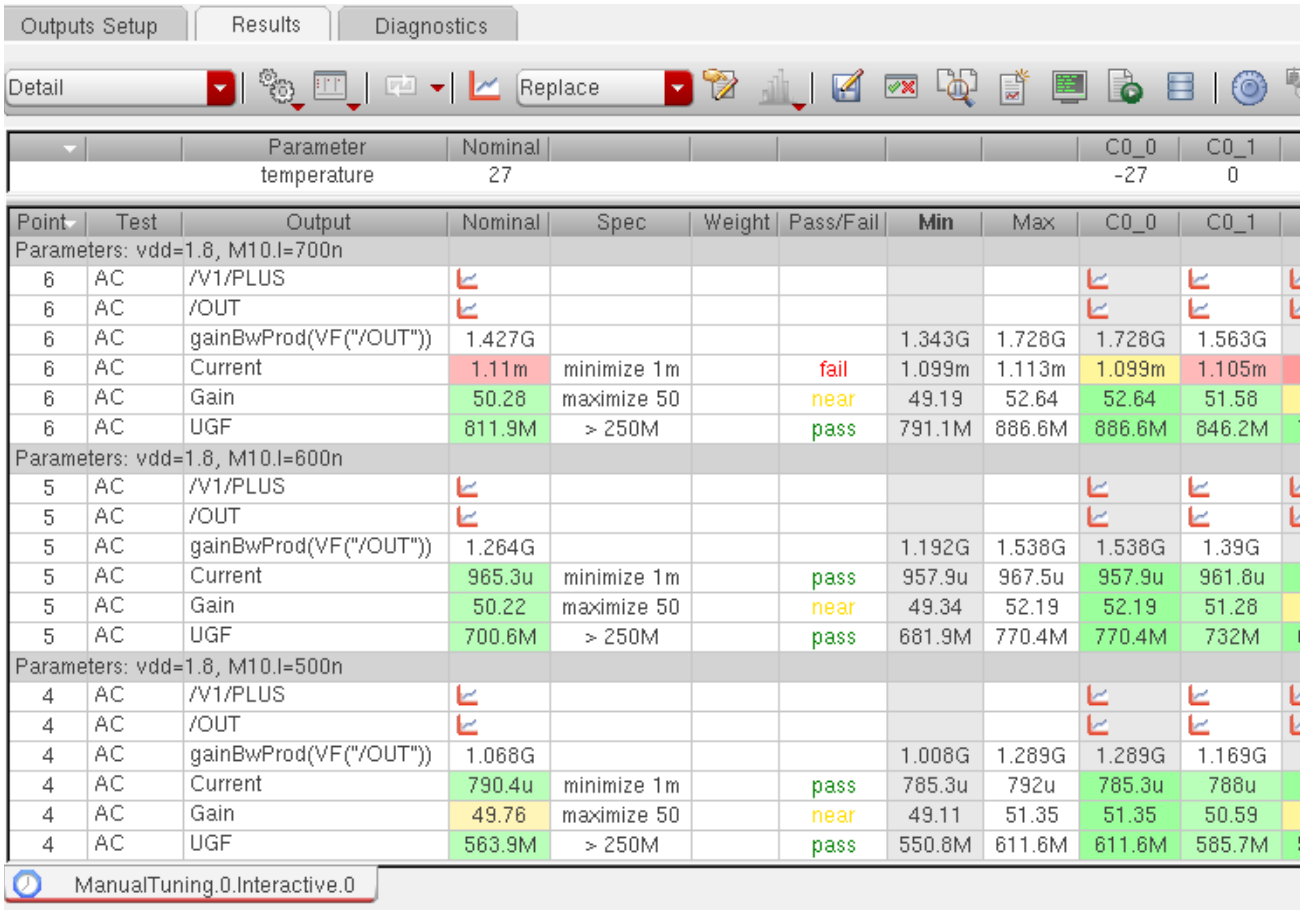
1. In the ade assembler view of your design, from the *Select a Run Mode* drop-down list on the Run toolbar, choose *Manual Tuning*.
2. Click the *Run Simulation* button on the Run toolbar.

   Note: The color of the *Run Simulation* button changes to yellow . This indicates that the Manual Tuning run mode has started.

   Now, you can tune or vary the values of design parameters and run multiple simulations to obtain the desired results.
3. In the Data View or the Variables and Parameters assistant, specify values for the design parameters.
4. From the *Select a Run Mode* drop-down list on the Run toolbar, choose *Single Run, Sweeps and Corners*.
5. Click the *Run Simulation* button on the Run toolbar to start the *Single Run, Sweeps and Corners* run.

   The simulation results are displayed on the *Results* tab, as shown below.

© 2023 Cadence Design Systems, Inc.



		Parameter	Nominal						C0_0	C0_1	
		temperature	27						-27	0	

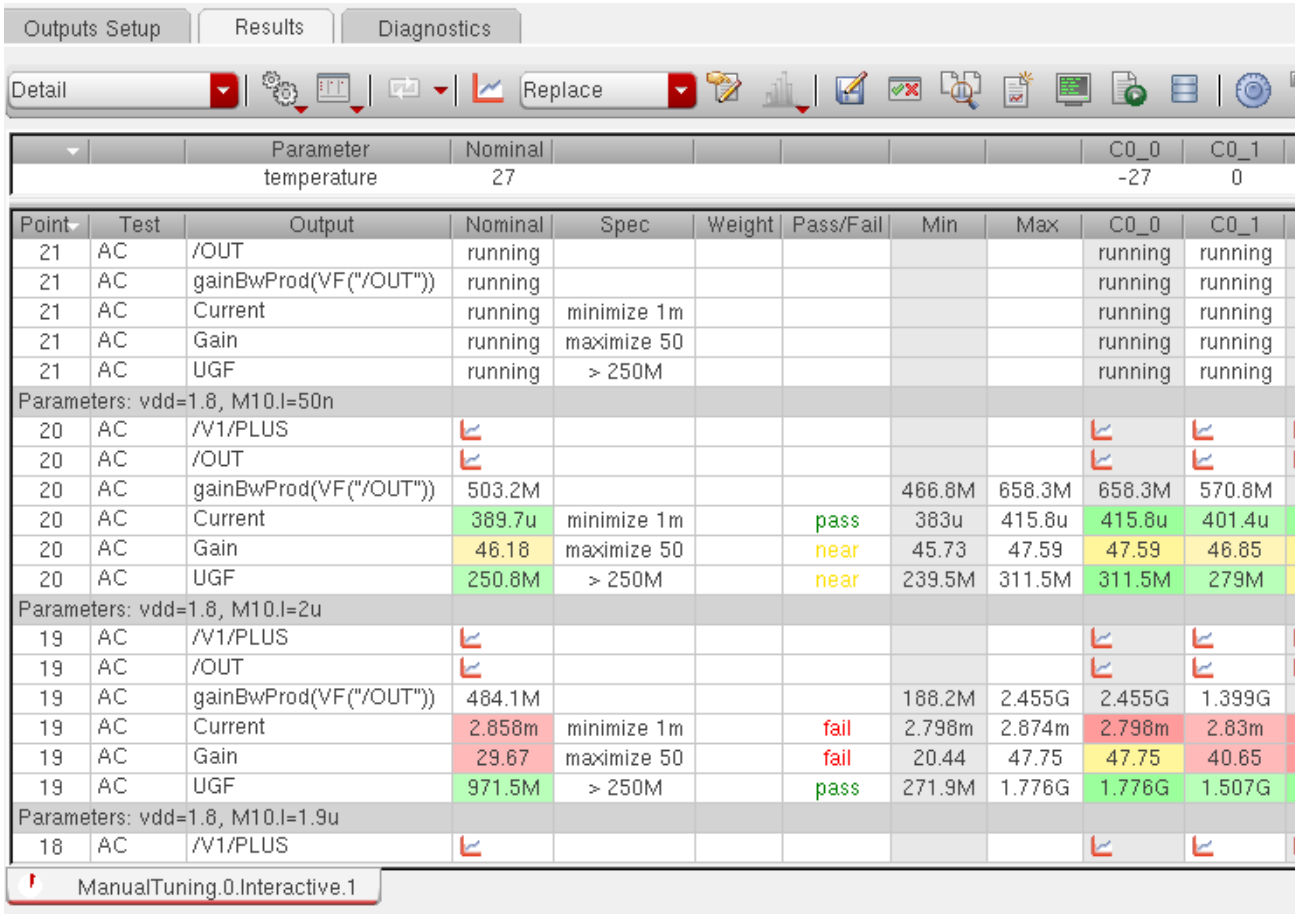
Point	Test	Output	Nominal	Spec	Weight	Pass/Fail	Min	Max	C0_0	C0_1	
Parameters: vdd=1.8, M10.l=700n											
6	AC	/V1/PLUS									
6	AC	/OUT									
6	AC	gainBwProd(VF("/OUT"))	1.427G				1.343G	1.728G	1.728G	1.563G	
6	AC	Current	1.11m	minimize 1m		fail	1.099m	1.113m	1.099m	1.105m	
6	AC	Gain	50.28	maximize 50		near	49.19	52.64	52.64	51.58	
6	AC	UGF	811.9M	> 250M		pass	791.1M	886.6M	886.6M	846.2M	
Parameters: vdd=1.8, M10.l=600n											
5	AC	/V1/PLUS									
5	AC	/OUT									
5	AC	gainBwProd(VF("/OUT"))	1.264G				1.192G	1.538G	1.538G	1.39G	
5	AC	Current	965.3u	minimize 1m		pass	957.9u	967.5u	957.9u	961.8u	
5	AC	Gain	50.22	maximize 50		near	49.34	52.19	52.19	51.28	
5	AC	UGF	700.6M	> 250M		pass	681.9M	770.4M	770.4M	732M	
Parameters: vdd=1.8, M10.l=500n											
4	AC	/V1/PLUS									
4	AC	/OUT									
4	AC	gainBwProd(VF("/OUT"))	1.068G				1.008G	1.289G	1.289G	1.169G	
4	AC	Current	790.4u	minimize 1m		pass	785.3u	792u	785.3u	788u	
4	AC	Gain	49.76	maximize 50		near	49.11	51.35	51.35	50.59	
4	AC	UGF	563.9M	> 250M		pass	550.8M	611.6M	611.6M	585.7M	

Note: The default name of the Results tab indicates the interactive run number in the Manual Tuning run mode. For example, in the figure shown above, the first single, sweep, corner run in the manual tuning run mode is named as `ManualTuning.0.Interactive.0`.

6. If required, change the value of parameters to tune your design and again click the *Run Simulation* button on the Run toolbar to start a new *Single Run, Sweeps and Corners* simulation run.

   Note: The results of the subsequent simulation runs are appended on top of the Results tab already open for the previous run, as shown below.

© 2023 Cadence Design Systems, Inc.



In the other run modes, results of every next simulation run are displayed on a new tab. Also, the name of the Results tab has also changed to `ManualTuning.0.Interactive.1`.

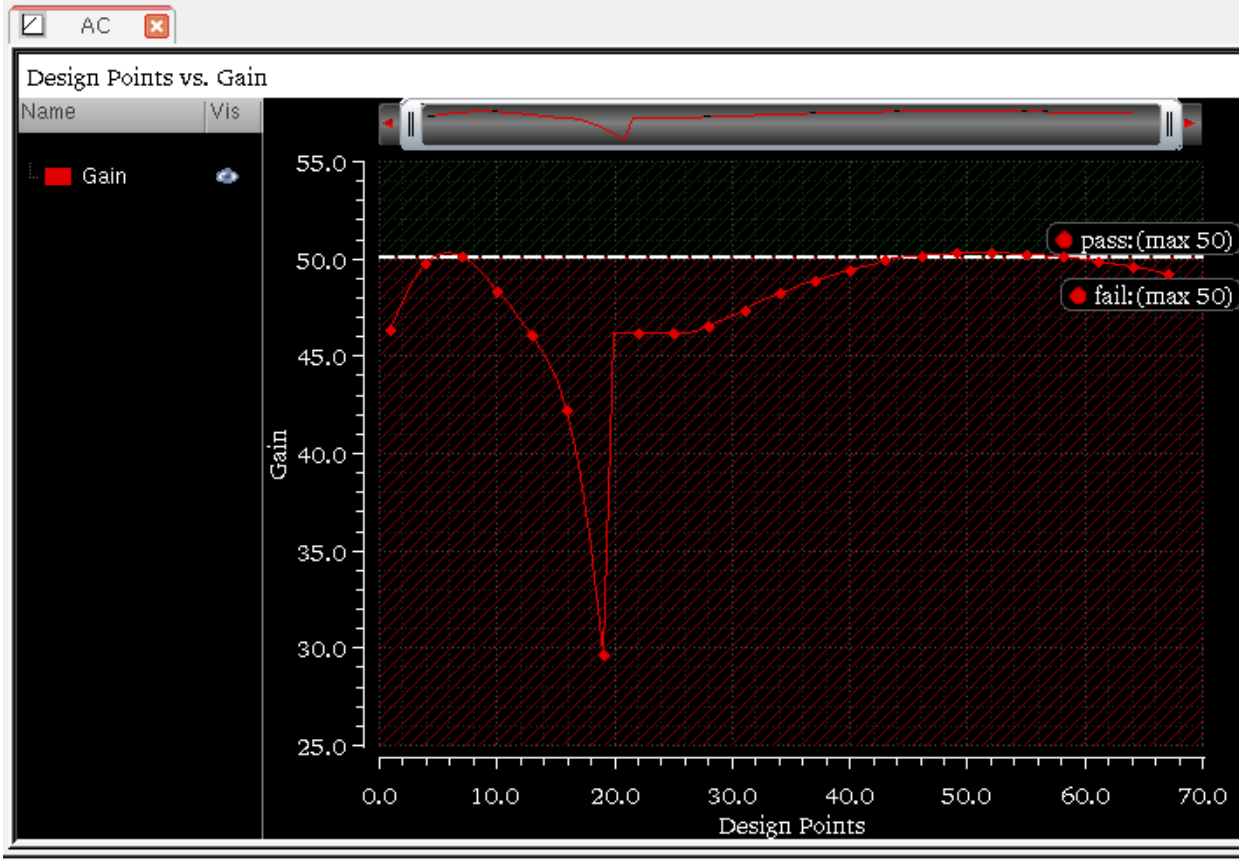
At the end of the second simulation run, the Results tab contains all the design points run during the `Interactive.0` and `Interactive.1` runs.

7. Compare the results of the two simulation runs.

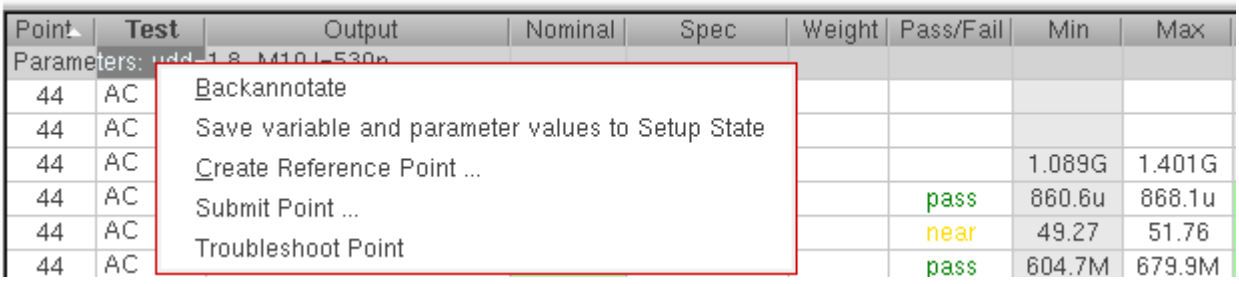
   You can also plot the results of all the simulations run in a manual tuning run. For that, in the Results tab, right-click a specification and choose *Plot Across Design Points*.

   Note: When you plot results across different design points, data is plotted for all the points of the two runs, as shown below.

© 2023 Cadence Design Systems, Inc.



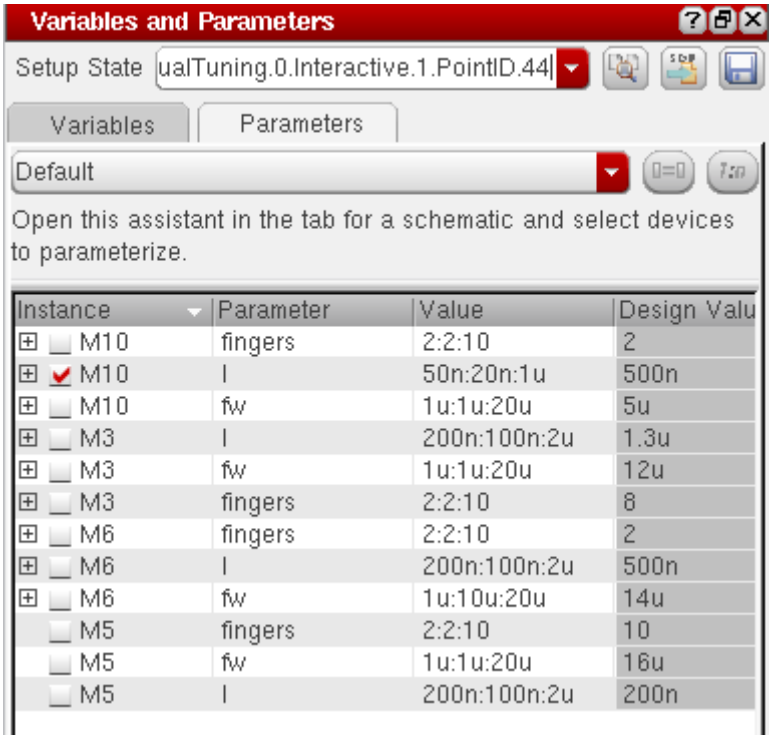
8. (Optional) If the desired results are not obtained, you might choose to further tune the design parameters and run more simulations till you get the best design point.
9. After the desired results are obtained, sort the design points to view the best or the worst point. If the best point meets the specifications, you can save the values used for that point so that they can be reused later. To do this, right-click the gray row on top of that design point and choose *Save variable and parameter values to Setup State*.



ADE Assembler creates a state by using the names of the simulation run and the design point ID. For example, for the design point 44 shown above, the tool creates a state

© 2023 Cadence Design Systems, Inc.

named `ManualTuning.0.Interactive.1.PointID.44` and saves the values of variables and design parameters in that state.



Note: You can use this saved state in the future to load the values of variables and parameters. For more details, see Working with Global Variables.

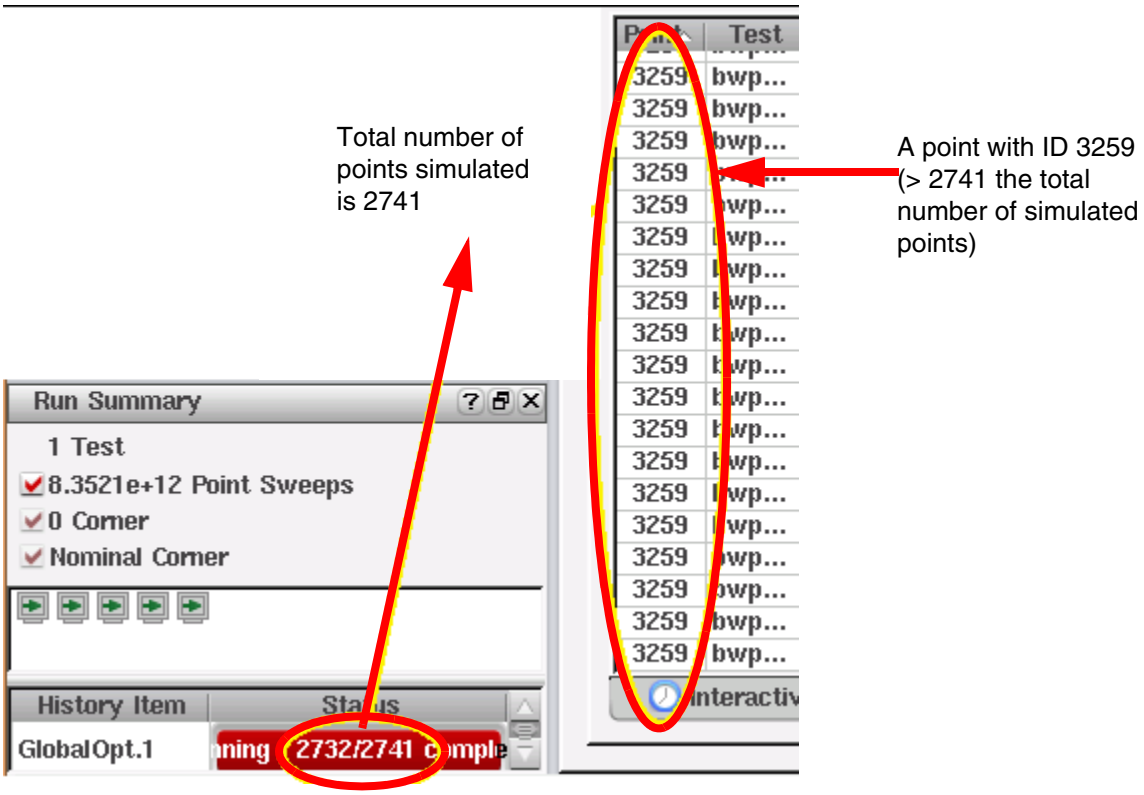
10. Click *Stop Simulation* on the *Run* toolbar to exit the Manual Tuning run mode.

Important Notes

- If you do not exit the Manual Tuning run mode and close the ADE Assembler GUI, the currently running Manual Tuning is not stopped. When you open a new ADE Assembler session, the Run Simulation button is yellow in color and you can continue with the same Manual Tuning run that was running earlier. If you want that the currently running Manual Tuning run should stop when the ADE Assembler GUI is closed, set the `stopManualTuningOnSessionExit` environment variable to `t`.
- The results of Manual Tuning are saved in a single database. You can view the results of all the child histories of a manual tuning run together by using the *View Results* command in the right-click menu of a parent or child Manual Tuning history.

© 2023 Cadence Design Systems, Inc.

- It is not possible to delete a specific child history or the simulation data for a specific child history. Therefore, on the Data View pane, the right-click menu commands, *Delete* and *Delete Simulation Data*, are not available for the child histories of a Manual Tuning run.
- The Design Points field displays a count of unique design points that are visited by the optimizer. With multiple tests and corners, multiple simulations are run for each design point. During optimization, the optimizer assigns a unique point ID numbers to all the design points. The optimizer may visit the design point or set of design parameters twice or more. In this case, the optimizer assigns a new ID number to the design point every time it is visited. This is called a cache-hit because the results for this point already exist in the database and simulator does not need to re-simulate this design point. When the design points are visited more than once, you may notice a greater point ID number than the total number of unique design points that are simulated (see the figure below). In case of a single test and no corners, the number of simulations equals the count of design points.



© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

25

Sensitivity Analysis

Sensitivity analysis helps in finding out the sensitivity of specifications to variables, device parameters, statistical parameters and DC operating point parameters. This chapter provides an overview of sensitivity analysis and describes how to run this analysis and view results. It also explains how to create worst case corners based on the results of sensitivity analysis.

For more details, refer to the following sections:

- Generating Sensitivity Data on page 1490
- Viewing Sensitivity Analysis Results on page 1524
- Working with Sensitivity Data on page 1530
- Plotting Sensitivity Analysis Results on page 1553
- Saving the Sensitivity Data on page 1570

© 2023 Cadence Design Systems, Inc.

Generating Sensitivity Data

You can generate and view data on the sensitivity of specifications to variables, device parameters, statistical parameters and DC operating point parameters.

Note: Sensitivity Analysis can also be performed on the simulation runs for the extracted and layout views.

Variation of Design Variables and Device Parameters

ADE Assembler varies each variable or parameter one-factor-at-a-time (OFAT). To calculate sensitivity, the tool perturbs the variable or parameter value around its nominal value, while leaving other variables as fixed.

For design and PVT variation, ADE Assembler provides the following methods:

- OFAT 3-level: Specify three values for a variable while varying one factor at a time. For example, 1.9u, 2.0u, 2.1u with nominal value equal to 2.0u. If, instead of three points, you specify a range of values, the tool chooses the following values of the variable for simulation:
  - one step below the nominal value
  - nominal value
  - one step above the nominal value

  For example, if the variable range is 1u:0.1u:5u and the nominal value is 3.6u, the three values used by the tool are: 3.5u 3.6u, and 3.7u.
- OFAT Sweep: Specify more than three values for a variable while varying one factor at a time. For example, if you specify the variable range as 200n:50n:400n, and the nominal value as 250n, the variable values used for simulation are: 200n, 250n, 300n, 350n, and 400n.
- Hammersley: Select this method to capture both nonlinear relationships and interactions among the design variables. The OFAT methods vary one factor at a time, whereas Hammersley generates samples varying more than one variable at a time. You can post process the Hammersley sampling results in the Variance Contribution analysis.

  Note: Statistical variables cannot be varied with Hammersley sampling. Run Monte Carlo simulations for statistical analysis.

© 2023 Cadence Design Systems, Inc.

Variation of Statistical Parameters

For statistical variation, ADE Assembler varies each statistical parameter one-factor-at-a-time at 3 levels (values). When the statistical parameter is modeled with normal or log normal distribution, ADE Assembler uses the following values for simulation:

- N sigma below the mean value
- mean value
- N sigma above the mean value

When the statistical parameter is modeled with uniform distribution, ADE Assembler chooses values based on the percentage of range.

Sometimes the variable whose sensitivity is being determined is at its boundary of possible values (as specified by the variable range and the nominal value). In this case, ADE Assembler cannot determine a higher value (if the current value is at the upper boundary) or lower value (if at lower boundary) for computing its sensitivity. In this case, the tool uses only two points for calculating the sensitivity and the correlation will always be 1 or -1, since two points will always form a straight line. However, the regression values (for this variable) are still applicable (since they represent the slope of the line).

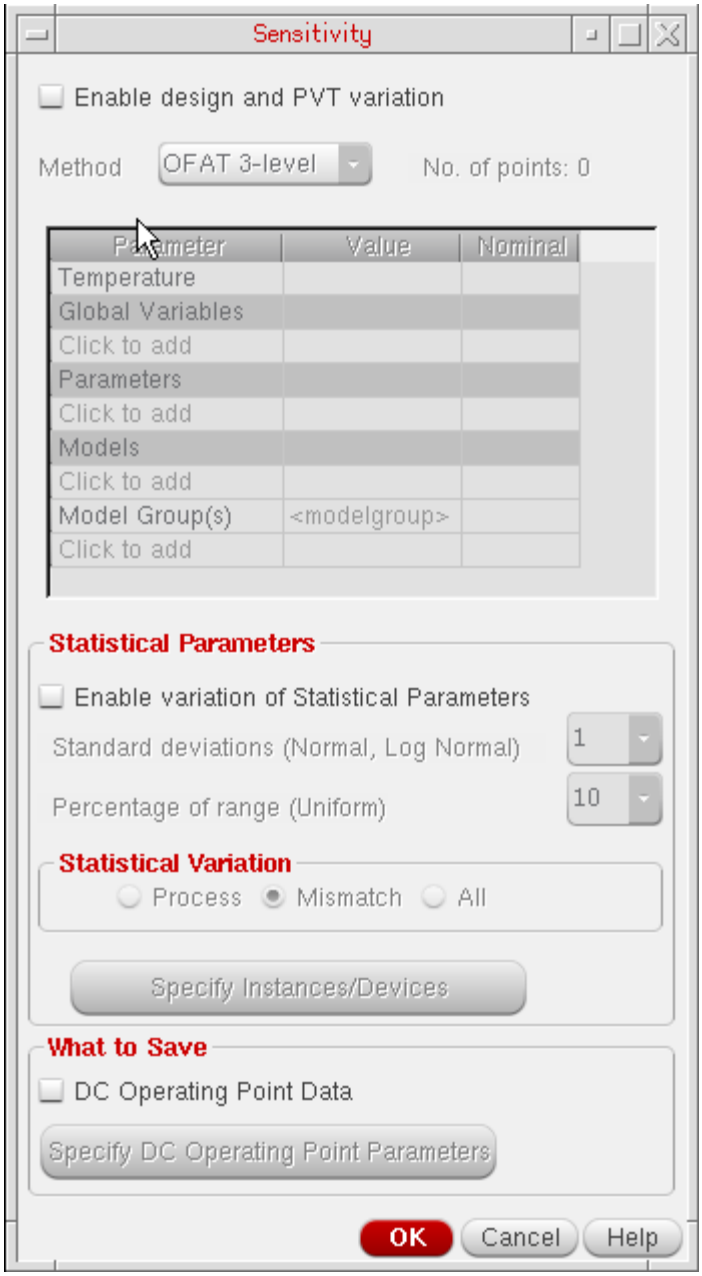
For boundary conditions where there are only two points available for simulation, the correlation is +1 or -1, depending upon whether the regression coefficient is positive or negative. However, it does not reflect the true meaning of correlation because you need to specify at least three points for calculating the correlation coefficients.

To generate and view the sensitivity data:

1. Select *Sensitivity Analysis* in the *Select a Run Mode* drop-down list on the Run toolbar, then click *Simulation Options* on the Run toolbar.

   The Sensitivity form appears, as shown below.

© 2023 Cadence Design Systems, Inc.



2. Select the *Enable Design and PVT Variation* check box if you want to vary temperature, global variables, device parameters, or model files for sensitivity analysis.

   Ensure that if you select this check box, you have specified at least one variable or parameter in the Variables and Parameters assistant. The variables or parameters that you have included in this form appear with a ~~strikethrough~~ in the Variables and Parameters assistant and the Data View pane.

© 2023 Cadence Design Systems, Inc.

Also ensure that you disable all other variables or parameters that have sweep values in the Variables and Parameters assistant and are not overridden in the Sensitivity form. This is because Sensitivity Analysis does not consider variables with sweep values in the Variables and Parameters assistant and displays an error.

Then, do the following:

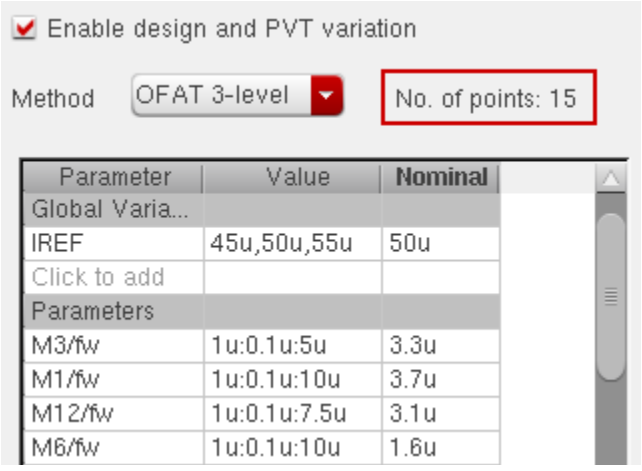
a. In the *Method* field, specify the method to be used to vary global variable and parameter values. You can select any one of the following two values:

- `OFAT 3-level`
- `OFAT sweep`
- `Hammersley`

For more details on these methods, refer to Variation of Design Variables and Device Parameters.

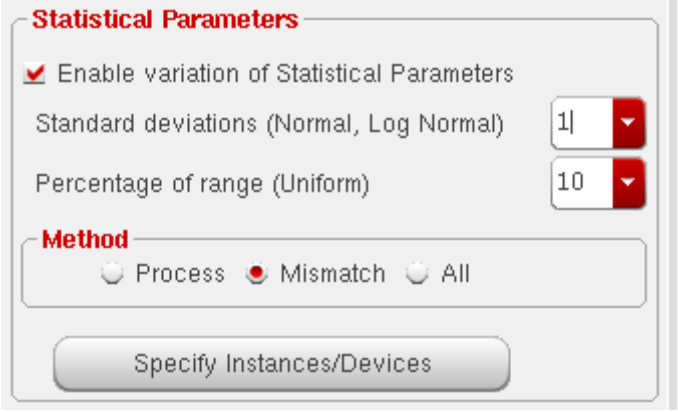
b. In the table given below the *Method* field, select the design variables, parameter, or model files that you want to vary. For more details, refer to Specifying Variables, Parameters, or Model Files to be Varied for Sensitivity Analysis.

c. Depending on the range of values, the nominal value, and the value in the *Method* list, the tool calculates the total number of data points for the sensitivity analysis and displays the count in the *No. of Points* label to the right of the *Method* field, as shown below:



3. Select the *Enable variation of Statistical Parameters* check box if you want to vary statistical process and mismatch parameters.

© 2023 Cadence Design Systems, Inc.



Then, do the following:

a. In the *Standard deviations (Normal, Log Normal)* field, specify the number of standard deviations for statistical parameters with normal or log normal distribution.

b. In the *Percentage of Range (Uniform)* field, specify the percentage range by which statistical parameters with uniform distribution need to be varied.

Note: The value specified must be a number greater than `0` and no more than `50`.

c. In the *Method* group box, select one of the following statistical variations:

Process	for process statistical variations
Mismatch	for per-instance statistical variations
All	for both process and per-instance statistical variations

d. (Optional) In step c above, if you select *Mismatch* or *All*, by default, the tool varies mismatch statistical parameters for all devices and instances in the design. Click *Specify Instances/Devices* to select specific instances and devices for which you want to analyze the impact of statistical variations. For more details, refer to Selecting Instances and Parameters for Mismatch Variation.

4. (Optional) By default, you cannot view the sensitivity data for DC operating point parameters because the data is not saved in the results database. To save and view sensitivity data for specific DC operating point parameters, do the following:

© 2023 Cadence Design Systems, Inc.

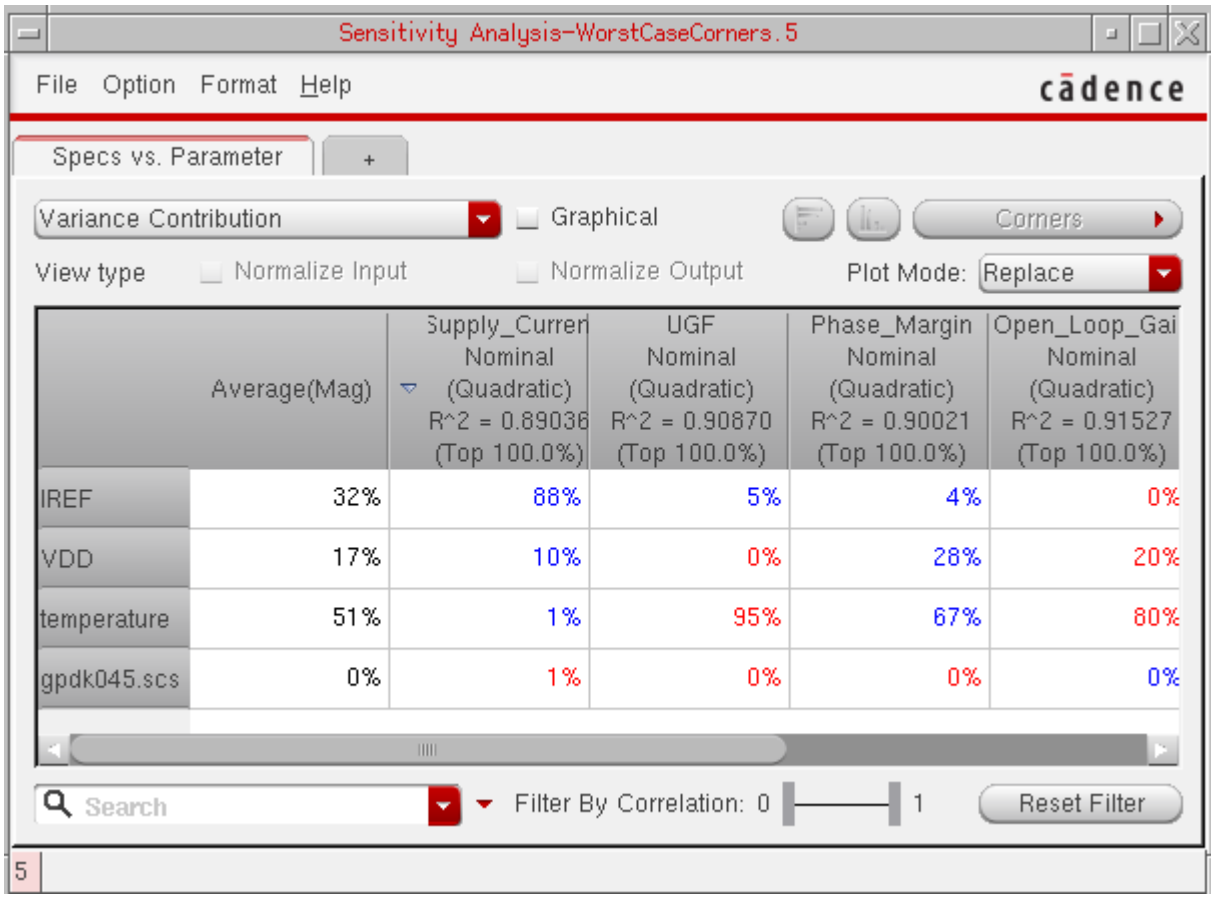
a. Select the *DC Operating Point Data* check box.

b. Click the *Specify DC Operating Point Parameters* button to specify the DC operating point parameters for which you want to save and view sensitivity data. For more information, see Saving Sensitivity Data for DC Operating Point Parameters.

5. Click *OK* to save the changes and close the Sensitivity form.

6. Click the *Run Simulation* button on the Run toolbar.

   After the run is complete, the tool displays the simulation results on the Results tab. In addition, it also opens the **Sensitivity Analysis** window that displays the sensitivity data for different specs or parameters. By default, this window displays the Variance Contribution results in percentage format, as shown below.



You can view the data in other formats. For more details, refer to Working with Sensitivity Data.

To display the maximum and minimum columns in the Sensitivity Analysis results table, right-click the left-most column heading and choose *Max* or *Avg* or *Min*. The Average

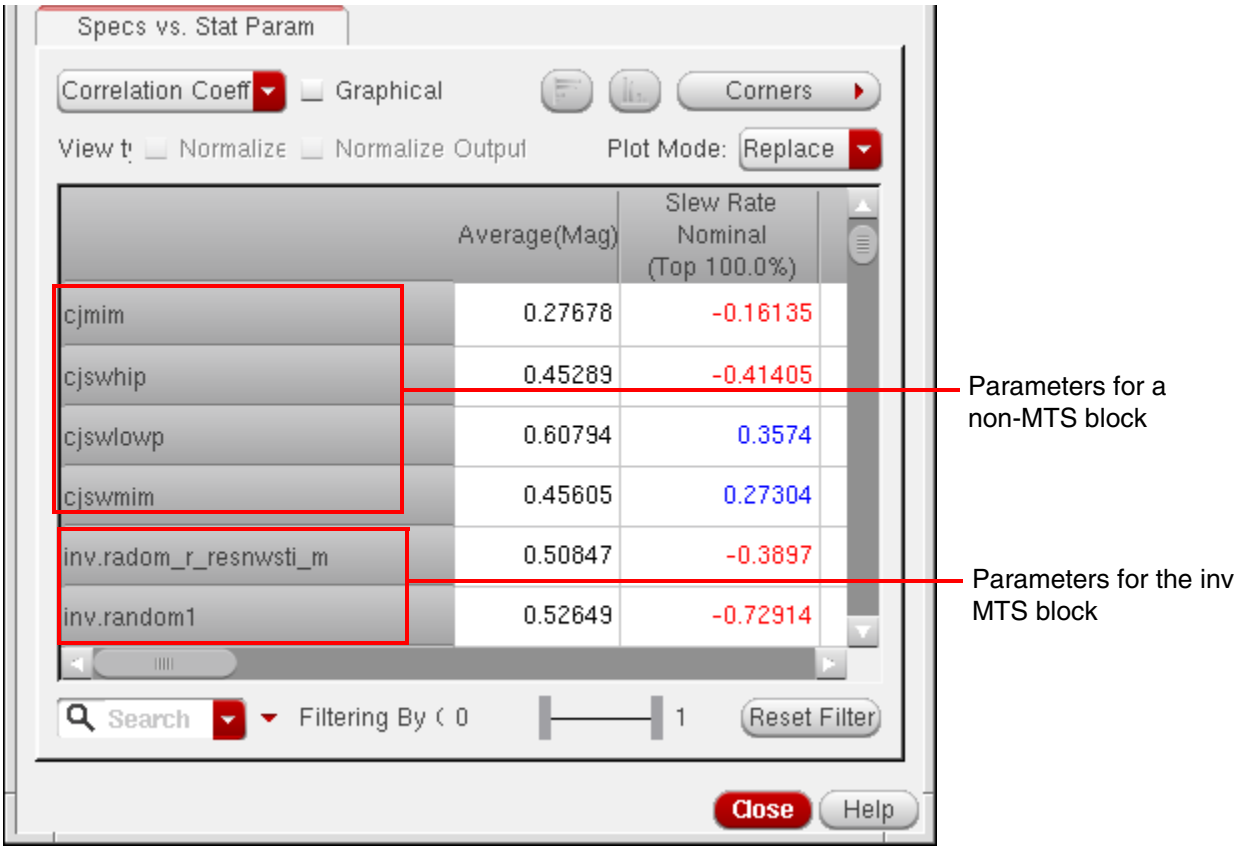
© 2023 Cadence Design Systems, Inc.

column is displayed by default. Also, the mismatch contribution tab displays only Max column.

Support for Multi-Technology Simulations in Sensitivity Analysis

If you have enabled multi-technology simulations for a test, in the Sensitivity Analysis results window, the parameter names for MTS blocks are prefixed with the name of their corresponding block.

For example, in the sensitivity analysis results displayed below, the names of parameters for the `inv` MTS block are prefixed with `inv`.



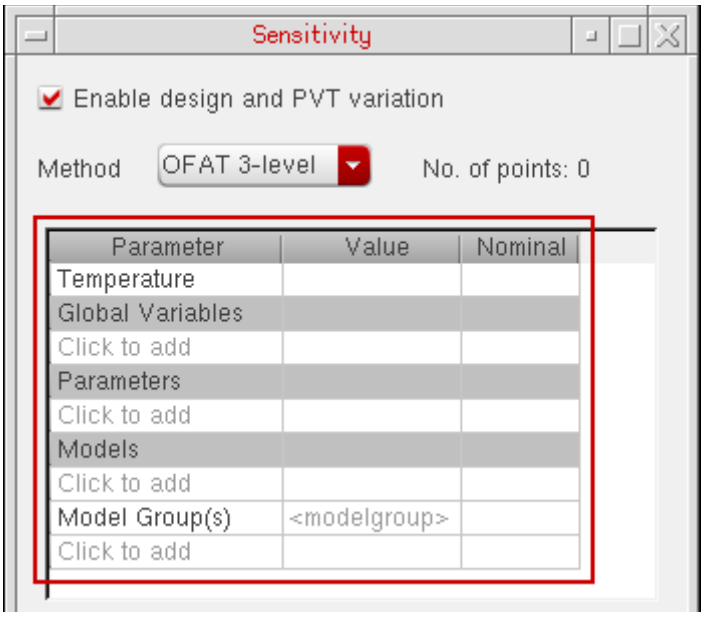
© 2023 Cadence Design Systems, Inc.

Related Topics

- Specifying Variables, Parameters, or Model Files to be Varied for Sensitivity Analysis
- Selecting Instances and Parameters for Mismatch Variation
- Viewing Sensitivity Analysis Results
- Working with Sensitivity Data
- Plotting Sensitivity Analysis Results
- Highlighting Associated Devices in the Schematic
- Saving the Sensitivity Data

Specifying Variables, Parameters, or Model Files to be Varied for Sensitivity Analysis

You can specify design variables, parameters, or model files that you want to vary for sensitivity analysis in the table given on the Sensitivity form.



In this table, you can select the name of a variable and specify values for which you want to vary the variable and a nominal value for it.

To vary temperature:

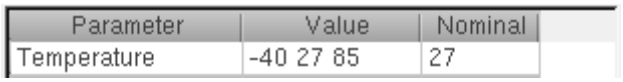
© 2023 Cadence Design Systems, Inc.

1. Double-click the *Value* cell for `Temperature` and specify a range or a set of values for which you want to vary temperature.
2. Double-click the *Nominal* cell for temperature and open the drop-down list.

   All the values specified in the *Value* cell are displayed in the list.
3. Select a nominal value from this list.

   Alternatively, you can get the value of temperature from the test. For this, right-click the *Nominal* cell for `Temperature` and choose *Get Value from Test*. This command copies the value of temperature from the test.

   Note: If the value taken from the test does not exist in the *Value* cell, the tool automatically adds the nominal value to the existing values. For example, if you specify -40 and 85 as two values for temperature and get the nominal value from test, which is 27, the tool adds 27 to the value list.

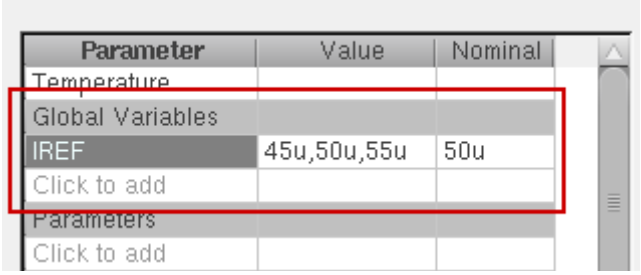


To select global variables to be varied:

1. Click *Click to add* in the Global Variables section in the table.

   A list of global variables given in the Data View pane is displayed in the cell.
2. From the list, select a global variable that you want to vary.

   The tool gets the sweep values and the nominal value specified for the variable in the Variables and Parameters assistant and displays it in the *Value* and *Nominal* cells, respectively, as shown below.



   If required, you can change the range or nominal value for a variable in this table. For more details, refer to Changing the Parameter Value and Changing the Nominal Value.

To select design parameters to be varied:

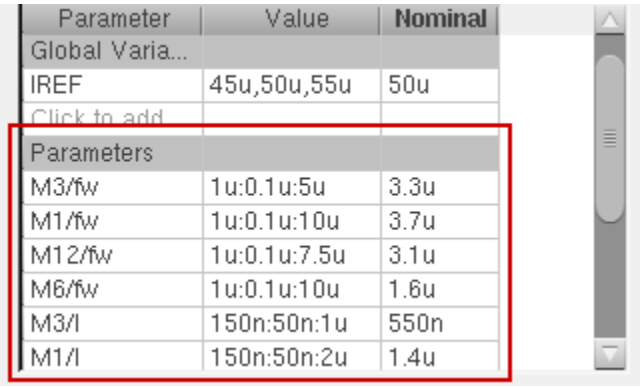
1. Click *Click to add* in the Parameters section in the table.

© 2023 Cadence Design Systems, Inc.

A list of parameters already defined in the Variables and Parameters assistant is displayed in the cell.

2. From the list, select the parameter that you want to vary.

   The tool reads the range of values and design value for the selected parameter from the Variables and Parameters assistant and displays it in the *Value* and the *Nominal* column, respectively, as shown below.



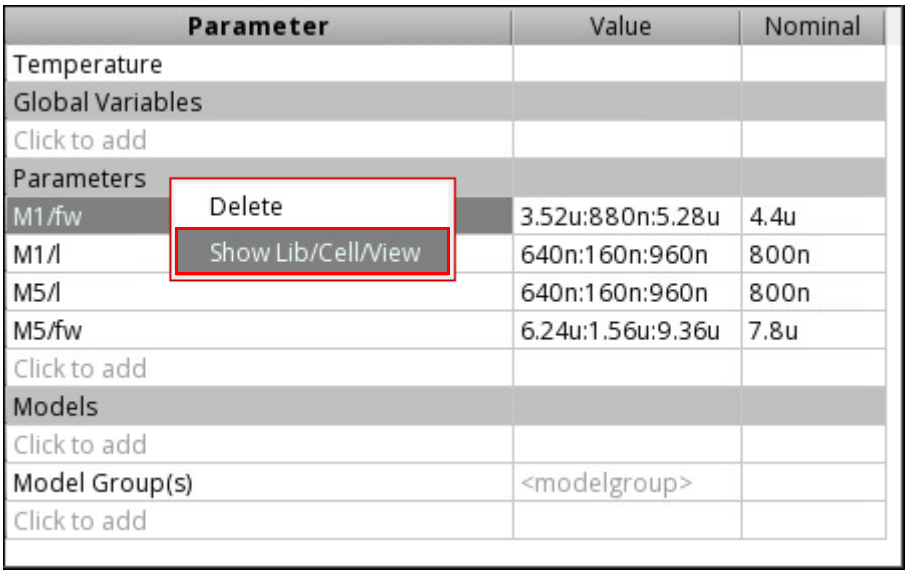
Parameter	Value	Nominal
Global Varia...		
IREF	45u,50u,55u	50u
Click to add		
Parameters		
M3/fw	1u:0.1u:5u	3.3u
M1/fw	1u:0.1u:10u	3.7u
M12/fw	1u:0.1u:7.5u	3.1u
M6/fw	1u:0.1u:10u	1.6u
M3/l	150n:50n:1u	550n
M1/l	150n:50n:2u	1.4u

If required, you can change the range or nominal value for a variable in this table. For more details, refer to Changing the Parameter Value and Changing the Nominal Value.

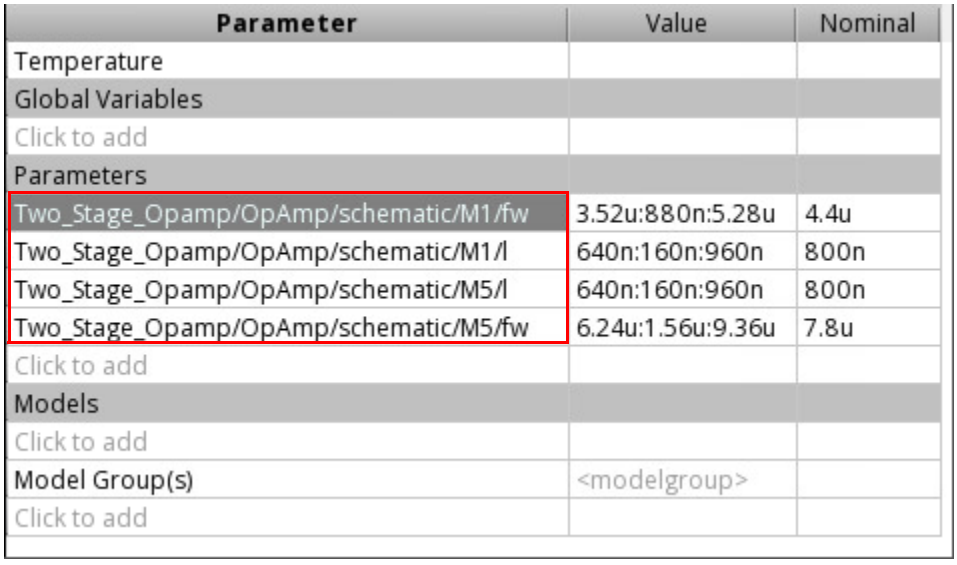
© 2023 Cadence Design Systems, Inc.

Tip

By default, only the names of the device parameters are displayed in the *Parameters* column of the *Parameter* table. To view the complete path (`<Library>/<Cell>/<View>/<parameter_name>`) to the device parameters, right-click either the *Parameters* header or the parameter name in the cell, and then select *Show Lib/Cell/View*.



The complete paths to the device parameters are displayed in the *Parameters* column, as shown in the following figure.



Alternatively, you can move the mouse over the row of a parameter to view the library/cell/view information for the device parameter.

© 2023 Cadence Design Systems, Inc.

To select model files to be varied:

1. Click *Click to add* in the Models section in the table.

   The Add/Edit Model Files form is displayed.

2. Browse and select a model file for which you want to vary different sections.

   The name of the file appears in the cell.

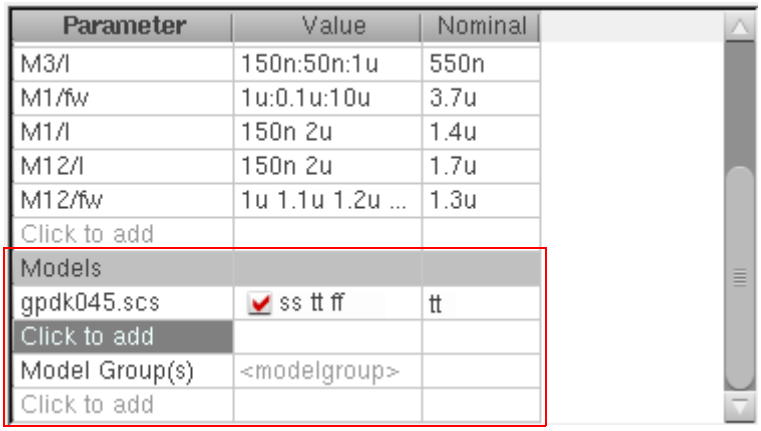
3. Double-click in the *Value* cell.

   A list of sections defined in the specified model file appears.

4. Select a section that you want to add to the list.

   You can select multiple sections that you want to vary for the model file.

5. Double-click in the *Nominal* cell and select name of a section that you want to use as a nominal value for the selected model file, as shown below.



Similarly, you can select model groups that you want to vary for the simulation.

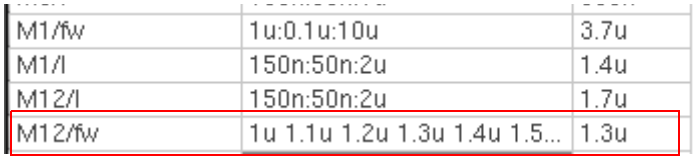
Important

You must define your models so that they respond to the statistical variations you choose. For a Spectre circuit simulator example on how to define your models, see Specifying Parameter Distributions Using Statistics Blocks in the *Spectre Circuit Simulator User Guide*.

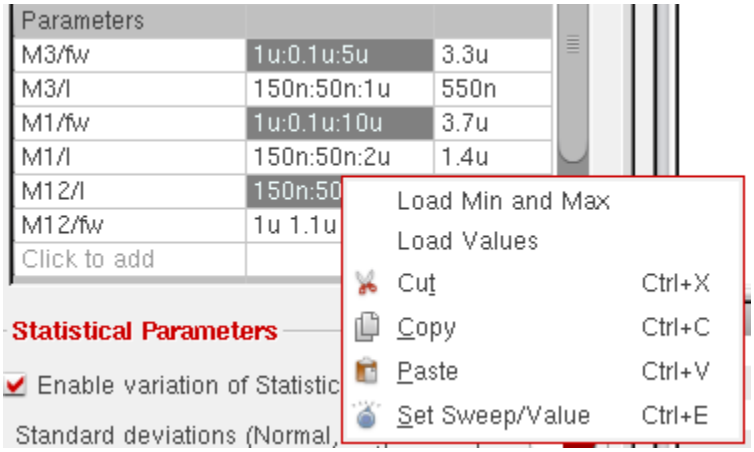
© 2023 Cadence Design Systems, Inc.

Changing the Parameter Value

To change the values for a parameter, double-click in the *Value* cell for the variable or parameter and edit the value. You can either specify a range or specify a list of space-separated values, as shown below.



Alternatively, you can select one or more values and right-click to display the shortcut menu:



From this menu, choose:

- *Load Values* to load the values defined in the Data View or the Variables and Parameters assistant
- *Load Min and Max* to load only the minimum and maximum values defined in the Data View or the Variables and Parameters assistant.
- *Set Sweep/Value* to edit the exclusion/inclusion list.

Changing the Nominal Value

By default, the nominal value for each variable or parameter is taken from the design, but you can modify it.

To change the nominal value for a variable or parameter:

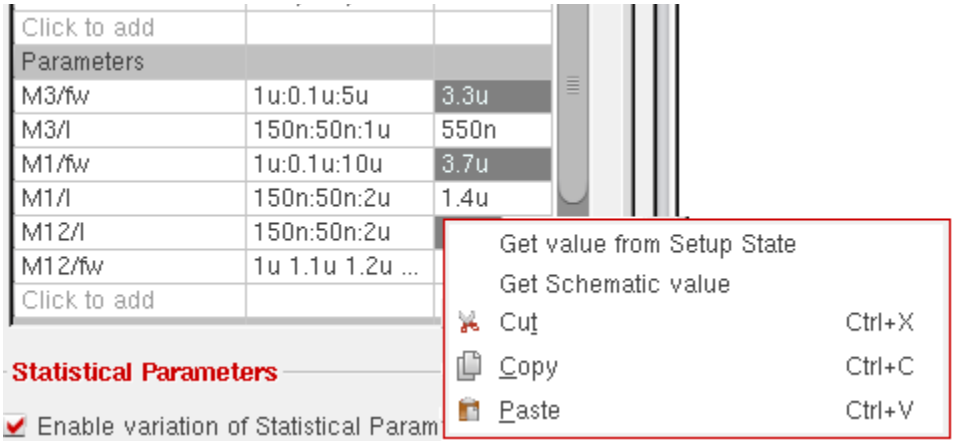
1. Double-click the value.

© 2023 Cadence Design Systems, Inc.

A list of possible nominal values calculated from its sweep range is displayed.

2. Select a value from the list or type any other value.

Alternatively, select one or more cells in the *Nominal* column and right-click to display the context-sensitive menu:



Choose one of the following commands to use values from a saved state or schematic:

- *Get value from Setup State* to get the nominal value from a saved setup state.

  The Get Nominal Values from Setup State form is displayed. Select the name of a saved state from the *State Name* list. Select *Parameters* in the *What to Load* section. This option copies the values of all the parameters from the selected saved state. Click *OK* to close the form.

- *Get Schematic value* to get the nominal value from the schematic.

Important points to Note

- It is not essential to specify a nominal value when the `Hammersley` variation method is selected. A nominal value is necessary for the `OFAT` methods.
- If the variation method is `OFAT 3-level`, a list of all possible nominal values calculated by using the range of values is displayed.
- If the variation method is `OFAT Sweep`, all sweep values appear in the list. You can select any value from this or type a different value.
- A valid nominal value for a parameter is any value from within the range or sweep set corresponding to the parameter. If you specify a nominal value outside the specified range or the sweep set, the tool does not accept the value and makes the cell blank.

© 2023 Cadence Design Systems, Inc.

- If the value taken from the setup state or schematic does not fall in the given value range or sweep set, the tool automatically expands the range or adds the schematic value to the sweep set.
- For global variables, the context-sensitive menu displays the *Get value from Test* command instead of *Get Schematic value*. Use this command to copy the value of a variable or parameter from the test.

Selecting Instances and Parameters for Mismatch Variation

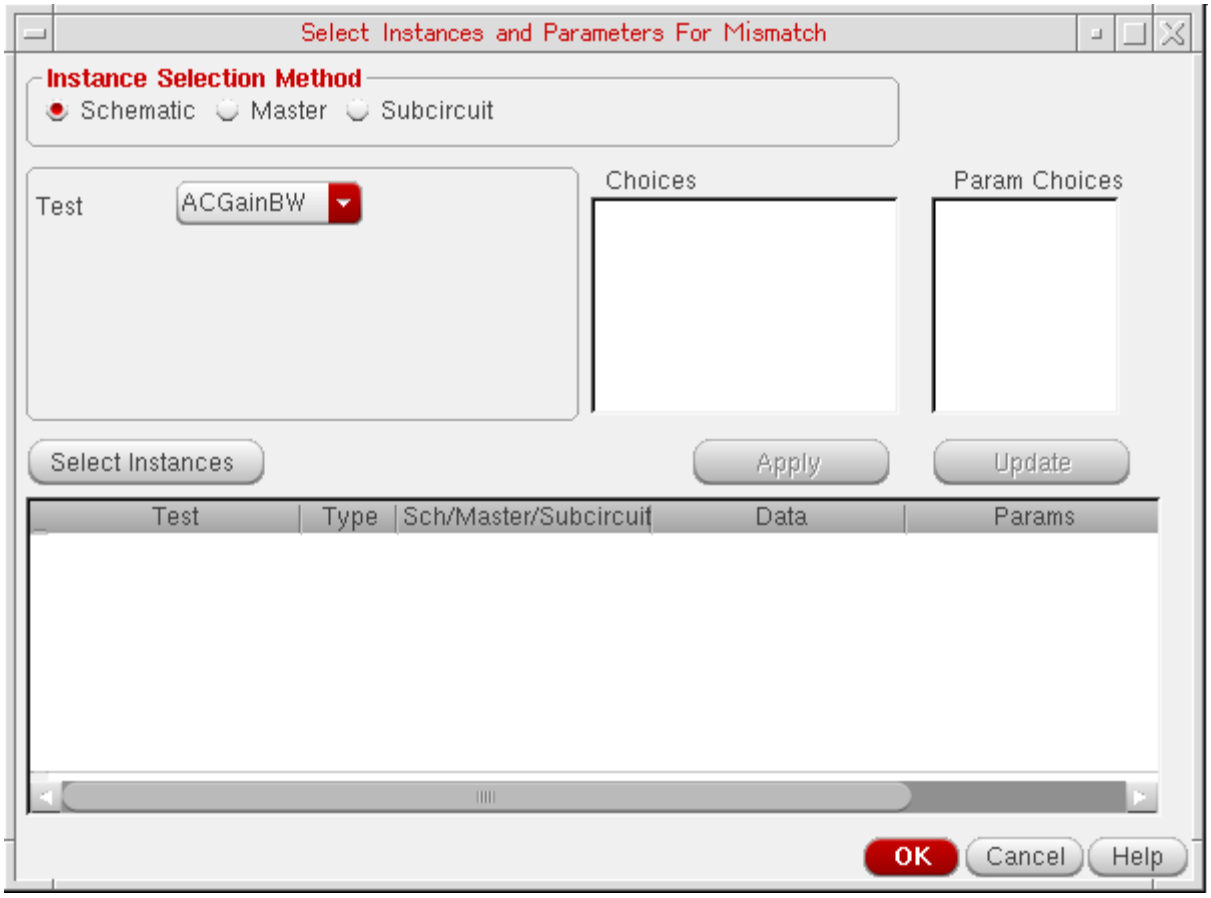
By default, statistical analysis analyzes mismatch variations for all devices and instances in the design. If the number of instances is large, time to run simulation is high. You can limit mismatch variation analysis to run for a selected set of instances or devices. For this, click *Specify instances/Devices* on the Sensitivity options form.

The tool netlists the design for the enabled tests and extracts data on the instances modeled by statistical mismatch.

Note: Netlist is created only when you are selecting instances for a set of tests for the first time. When you click *Specify instances/Devices* for consecutive selections, netlist is created for only those designs/tests that were changed after the previous selections.

After the dummy simulations are run successfully, the Select Instances and Parameters for Mismatch form is displayed, as shown in the following figure.

© 2023 Cadence Design Systems, Inc.



Only those instances that are modeled with mismatch variation will be listed in the *Choices* lists.

To select instances and parameters, do the following:

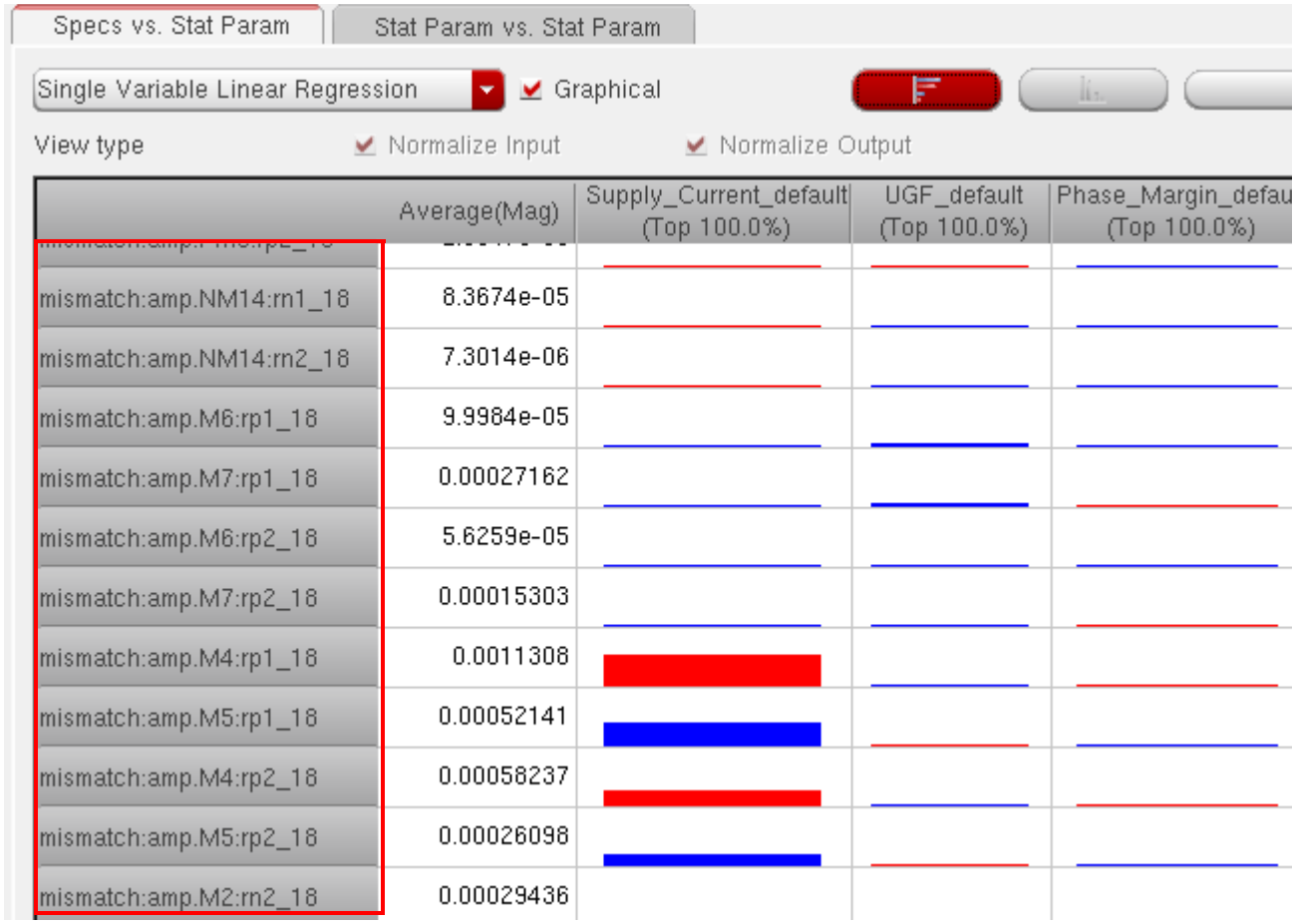
1. In the *Instance Selection Method* section, select a method by which you want to select instances. You can select instances in any of the following three ways:

   - By selecting instances on schematic. For this, select the *Schematic* option in the *Instance Selection Method* section. For more details, refer to Selecting Instances from Schematic on page 1506.
   - By selecting instances of master cellviews. For this, select the *Master* option in the *Instance Selection Method* section. For more details, refer to Selecting Instances from Master Cellviews on page 1508.
   - By selecting subcircuit instances. For this, select the *Subcircuits* option in the *Instance Selection Method* section. For more details, refer to Selecting Instances from Subcircuits on page 1509.

© 2023 Cadence Design Systems, Inc.

2. Click *OK* to save the changes and return to the Sensitivity options form.

Next time when you run sensitivity analysis, the mismatch variation is run for only the selected instances. For example, in the following figure, the variation is limited to only selected instances and parameters.



Selecting Instances from Schematic

To select instances from schematic, do the following:

1. In the *Instance Selection Method* section, select the *Schematic* option.
2. In the *Test* list, select the name of test for which you want to open the schematic view.

   Note: Select *All Tests* if you want to select instances for all the tests that are enabled in the *Data View* pane.
3. Click *Select Instances*. The schematic view for the selected test is opened in a new tab.

© 2023 Cadence Design Systems, Inc.

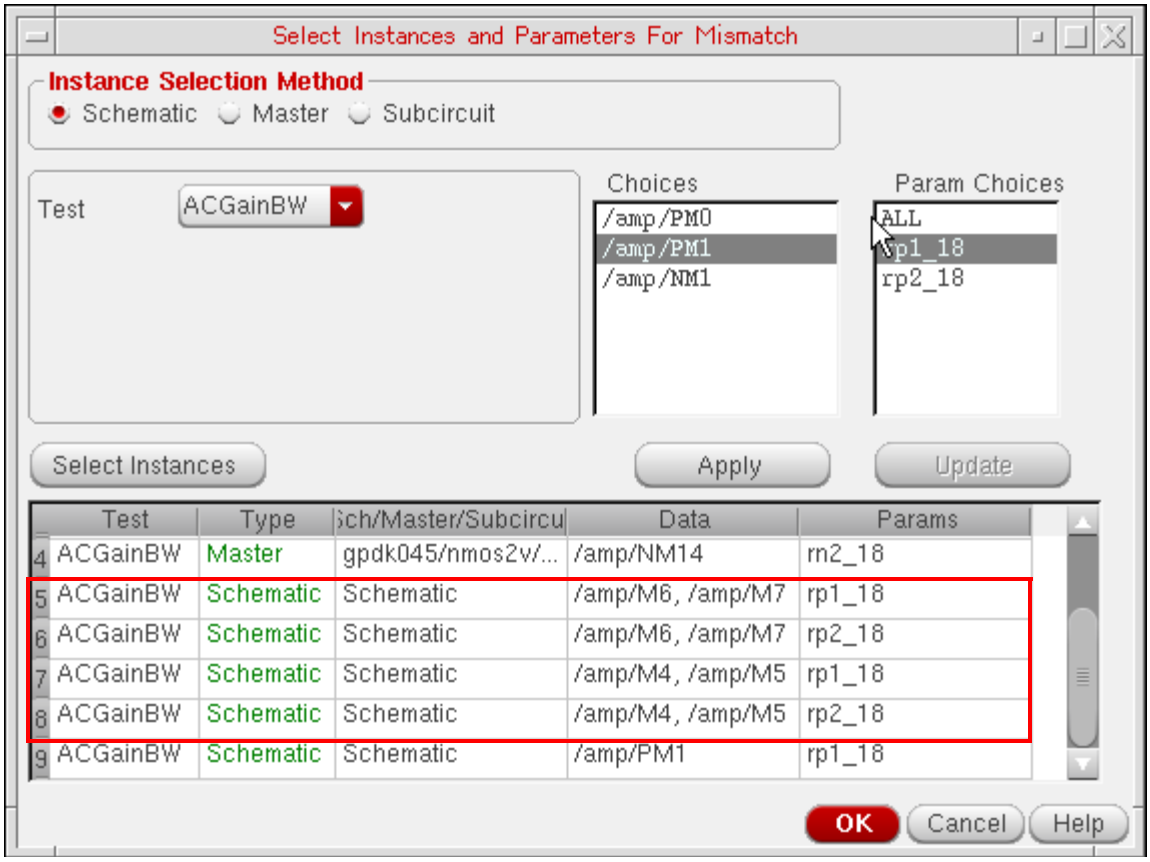
4. On the schematic tab, hold the *Shift* key and select instances that you want to include for mismatch variation.
5. Press the *Esc* key when done.

   The list of selected instances that are modeled with statistical mismatch variation is displayed in the *Choices* list.

   Note: The selected instances that are not modeled with statistical mismatch variation will not be added to the *Choices* list. Instead, an error message 1692 is displayed with the names of all such instances.
6. In the *Choices* list, select an instance.

   modeled parameters corresponding to the selected instances are displayed in the *Param Choices* list.
7. In the *Param Choices* list, select `ALL` or specific parameters that you want to vary for mismatch variation.
8. Click *Apply* to save the selections as rows in the tabular list, as shown in the following figure.

© 2023 Cadence Design Systems, Inc.



Selecting Instances from Master Cellviews

To select instances from cellviews, do the following:

1. In the *Instance Selection Method* section, select the *Master* option.

   List of all the libraries, cells, and views are displayed for the selected test.

2. Select the library, cell, and view name for which you want to select instances.

   For the selected cellview, the list of instances that are modeled with statistical mismatch variation are displayed in the *Choices* list.

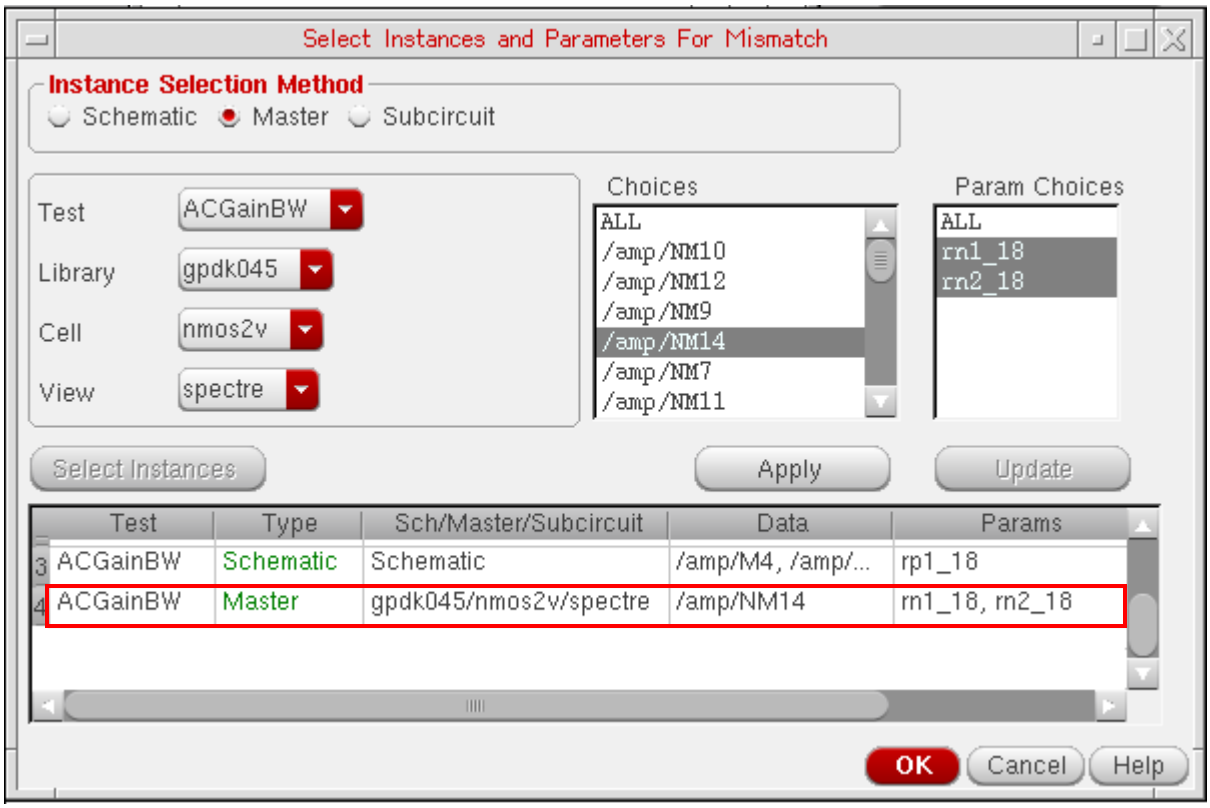
3. In the *Choices* list, select one or more instances.

   The modeled parameters corresponding to the selected instances are displayed in the *Param Choices* list.

4. In the *Param Choices* list, select `ALL` or specific parameters that you want to vary.

© 2023 Cadence Design Systems, Inc.

5. Click *Apply* to save the selections as rows in the tabular list, as shown in the following figure.



Selecting Instances from Subcircuits

To select instances from subcircuits, do the following:

1. In the *Instance Selection Method* section, select *Schematic*.
2. In the *Test* list, select the name of test for which you want to open the schematic view.

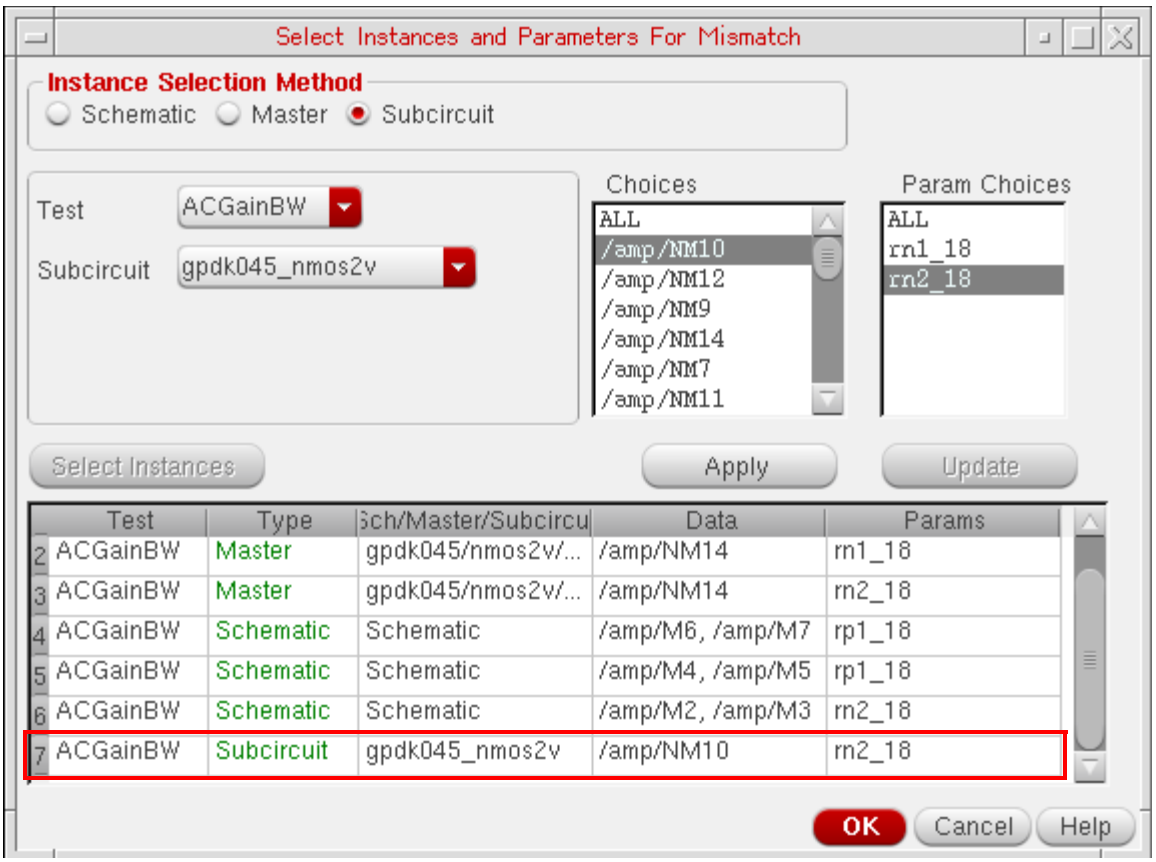
   Note: Select *All Tests* if you want to select instances for all the tests that are enabled in the *Data View* pane.
3. In the *Subcircuit* list, choose the subcircuit whose instances you want to select.

   All the instances in the selected subcircuit that are modeled with statistical mismatch variation are displayed in the *Choices* list.
4. In the *Choices* list, select one or more instances.

   The modeled parameters corresponding to the selected instances are displayed in the *Param Choices* list.

© 2023 Cadence Design Systems, Inc.

5. In the *Param Choices* list, select `ALL` or specific parameters that you want to vary.
6. Click *Apply* to save the selections as rows in the tabular list, as shown in the following figure.



Tip

To delete any selection from the tabular list, select the required row(s), right-click and choose *Delete*.

Saving Sensitivity Data for DC Operating Point Parameters

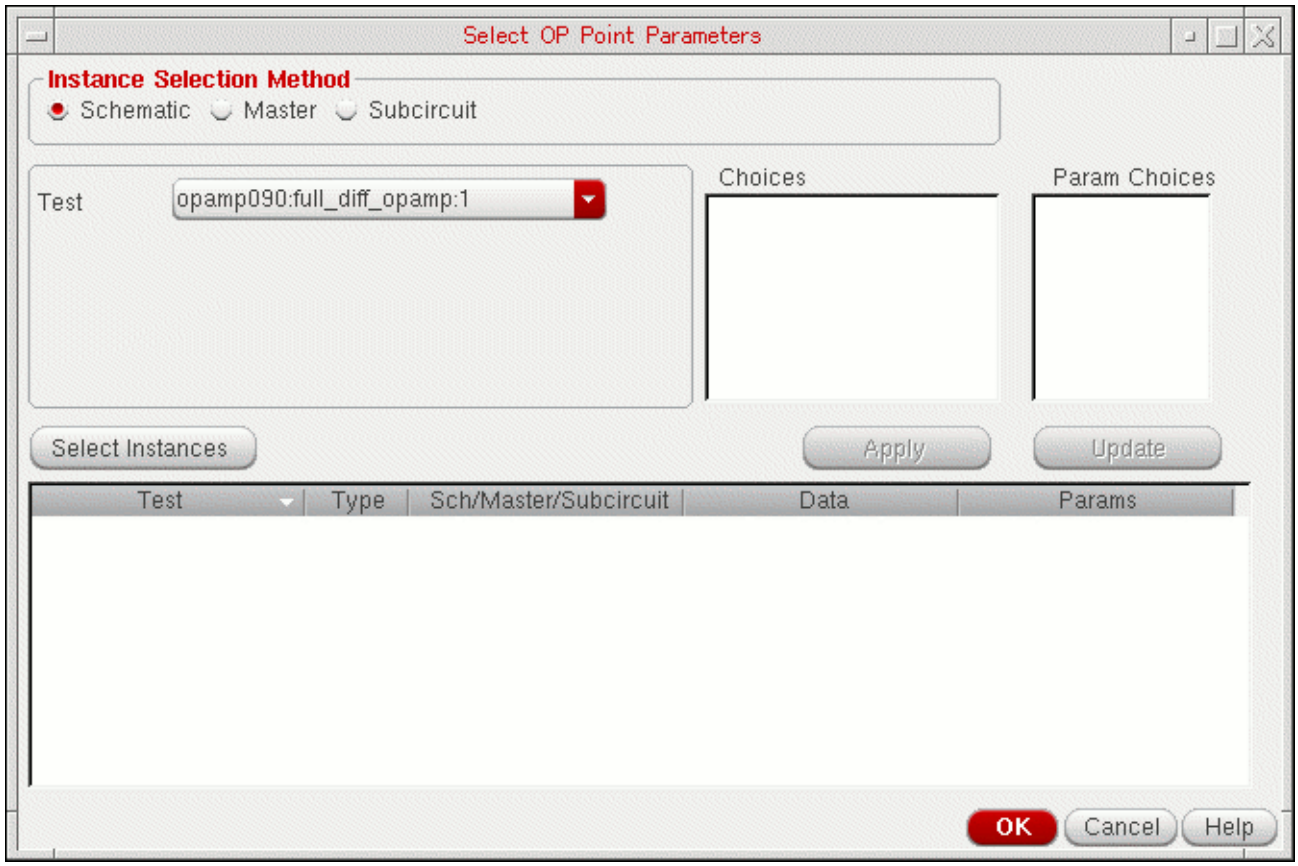
By default, you cannot view the sensitivity data for DC operating point parameters because the data is not saved in the results database. The Select OP Point Parameters form allows you to specify the DC operating point parameters for which you want to save and view sensitivity data. For more information about viewing sensitivity data, see Viewing Sensitivity Analysis Results on page 1524.

© 2023 Cadence Design Systems, Inc.

To specify the DC operating point parameters for which you view sensitivity data, do the following:

1. On the Sensitivity form click the *Specify DC Operating Point Parameters* button.

   The Select OP Point Parameters form appears.



2. Do one of the following:

Select	To
Schematic	Add DC operating point parameters by selecting instances on the schematic. For more information, see Adding DC Operating Point Parameters of Schematic Instances on page 1512.
Master	Add DC operating point parameters of instances of cellviews. For more information, see Adding DC Operating Point Parameters of Cellview Instances on page 1516.

© 2023 Cadence Design Systems, Inc.

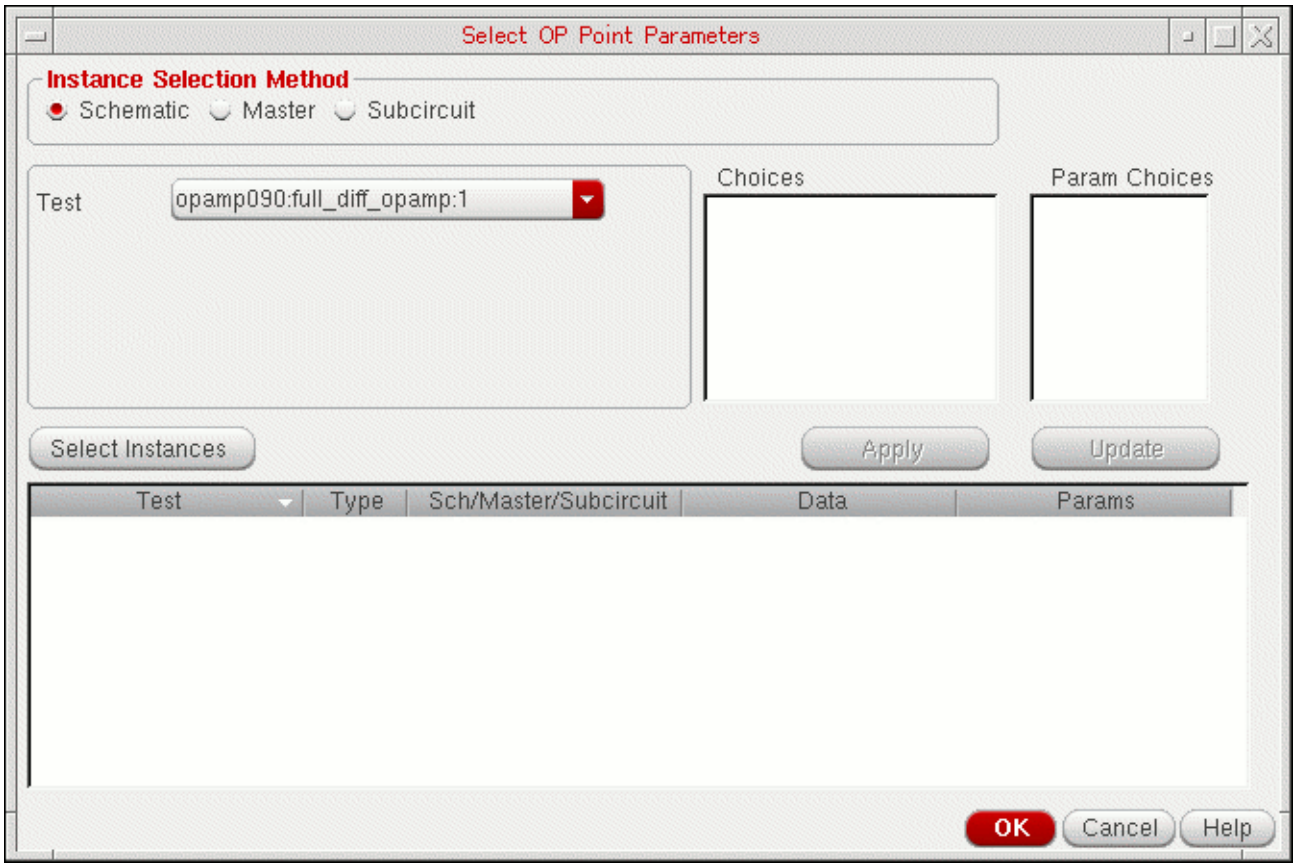
Select	To
Subcircuit	Add DC operating point parameters of subcircuit instances. For more information, see Adding DC Operating Point Parameters of Subcircuit Instances on page 1520.

3. Click *OK* to save the changes and return to the Sensitivity options form.

Adding DC Operating Point Parameters of Schematic Instances

To add DC operating point parameters by selecting instances on the schematic, do the following:

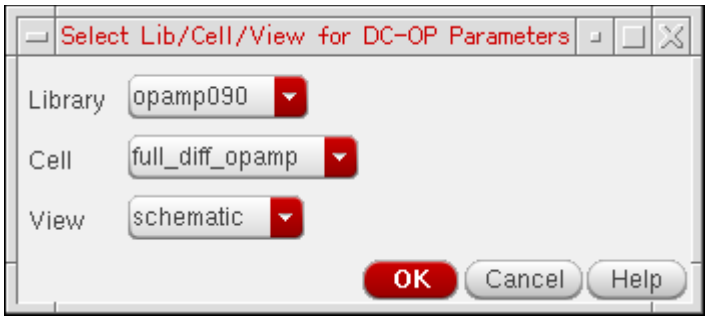
1. Select the *Schematic* option.



2. In the *Test* drop-down list, choose the test in which you want to select instances.

© 2023 Cadence Design Systems, Inc.

To select instances for all the tests that are enabled in the Data View pane, choose *All Tests* in the *Test* drop-down list. If you choose *All Tests*, the Select Lib/Cell/View for DC-OP Parameters form appears.



Do the following to open the schematic view from which you want to select instances for all the tests that are enabled in the Data View pane.

a. Use the *Library*, *Cell* and *View* drop-down lists to select the schematic view.

Note: The *Library*, *Cell* and *View* drop-down lists display only the cellviews that are used in the designs for all the tests.

b. Click *OK*.

The selected schematic view is opened in a new tab.

3. If a test name is selected in the *Test* drop-down list, click the *Select Instances* button. The schematic for the test is opened in a new tab.

4. In the schematic, select one or more instances.

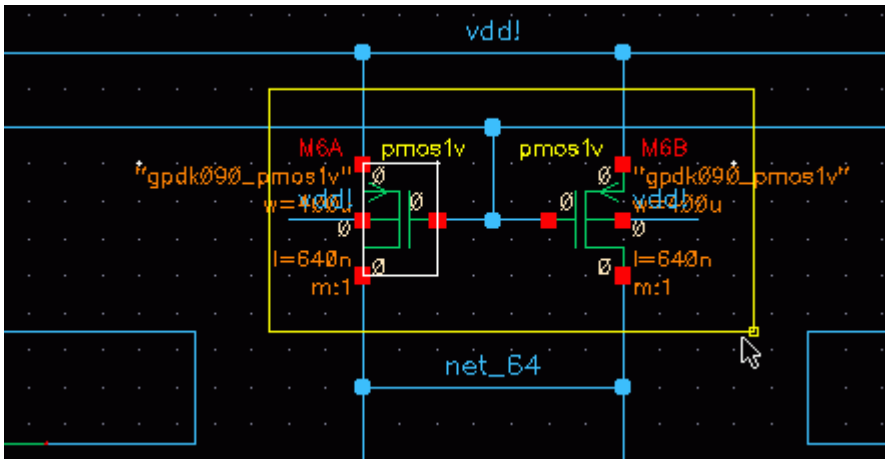
To select more than one instance at a time, do one of the following:

- Hold down the *Shift* key and click the instances.
- Click and drag the mouse over the instances you want to select.

  All the instances that are within the yellow bounding box that appears are included in the selection.

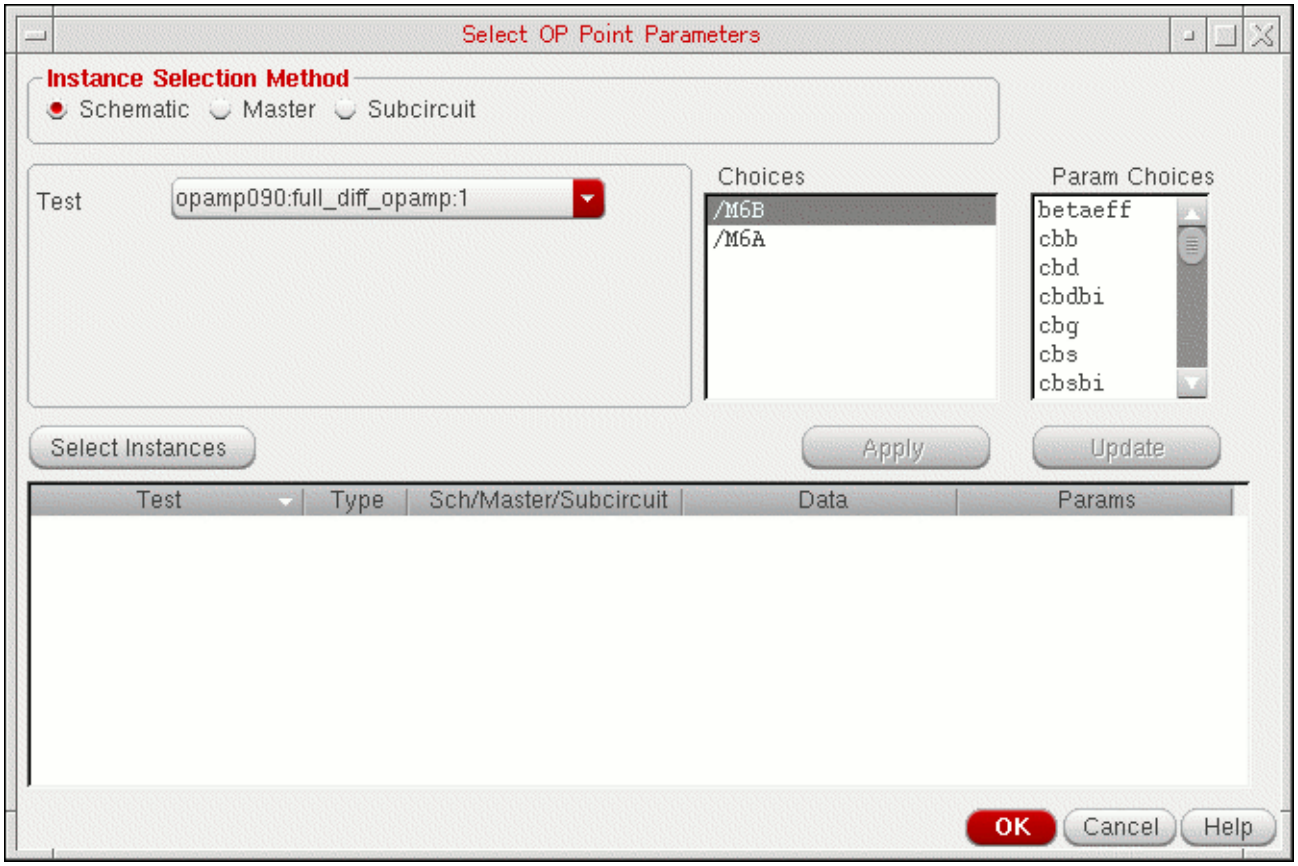
© 2023 Cadence Design Systems, Inc.

In the following example, instances `M6A` and `M6B` that are within the yellow bounding box are included in the selection.



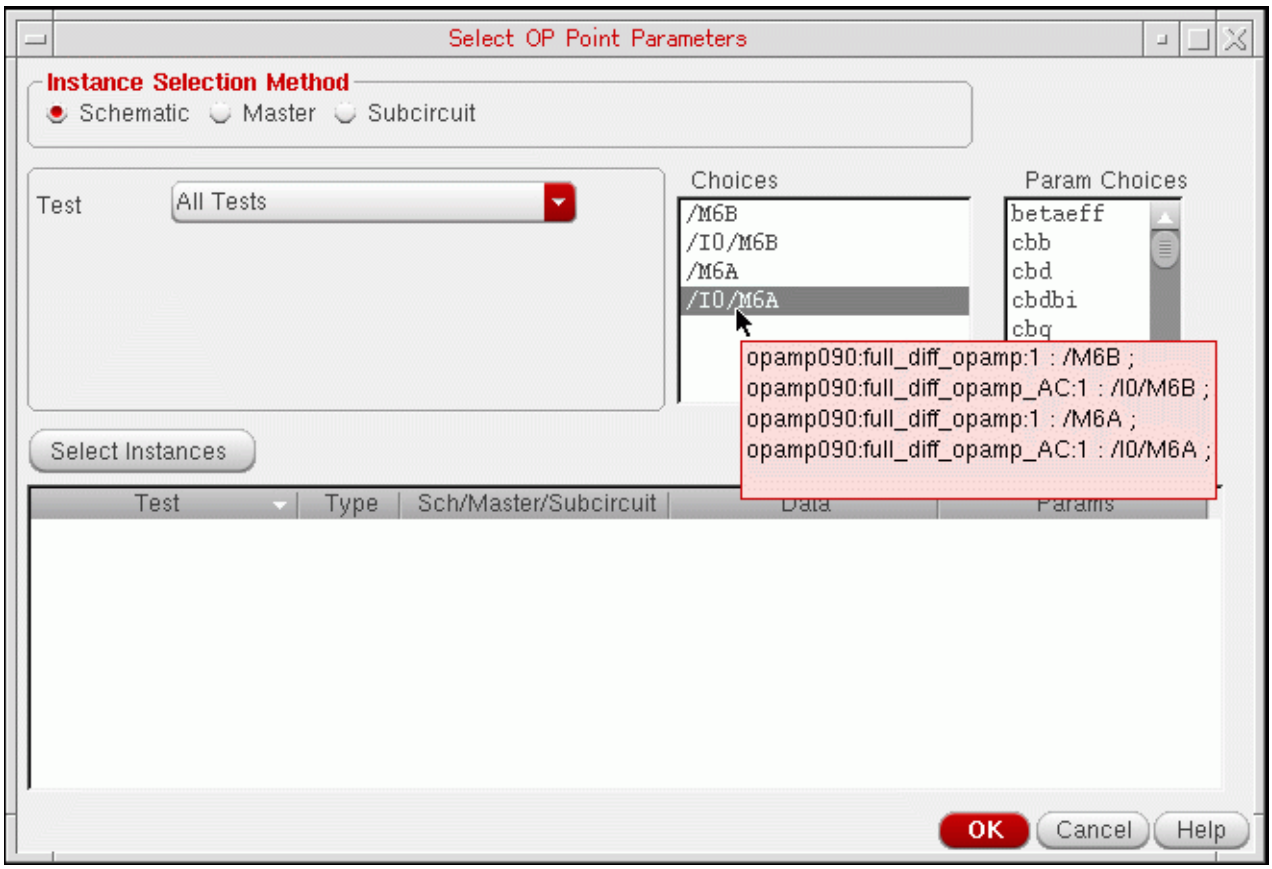
5. Press the *Esc* key when you are done.

   The selected instances are displayed in the *Choices* field.



© 2023 Cadence Design Systems, Inc.

Note: If *All Tests* is selected in the *Test* drop-down list, the instance names of the selected instances in the designs for all the tests are displayed in the *Choices* field. You can place the mouse pointer in the *Choices* field to view the instance names of the selected instances in the design for each test.



6. In the *Choices* field, select the instance for which you want to add DC operating point parameters.

   The DC operating point parameters for the instance are displayed in the *Param Choices* field.

Tip

If the same DC operating point parameter exists in more than one instance, you can simultaneously add the parameter for all the instances by selecting those instances in the *Choices* field. To select multiple instances, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontinguous selection) and click the next instance to add it to the selection set.

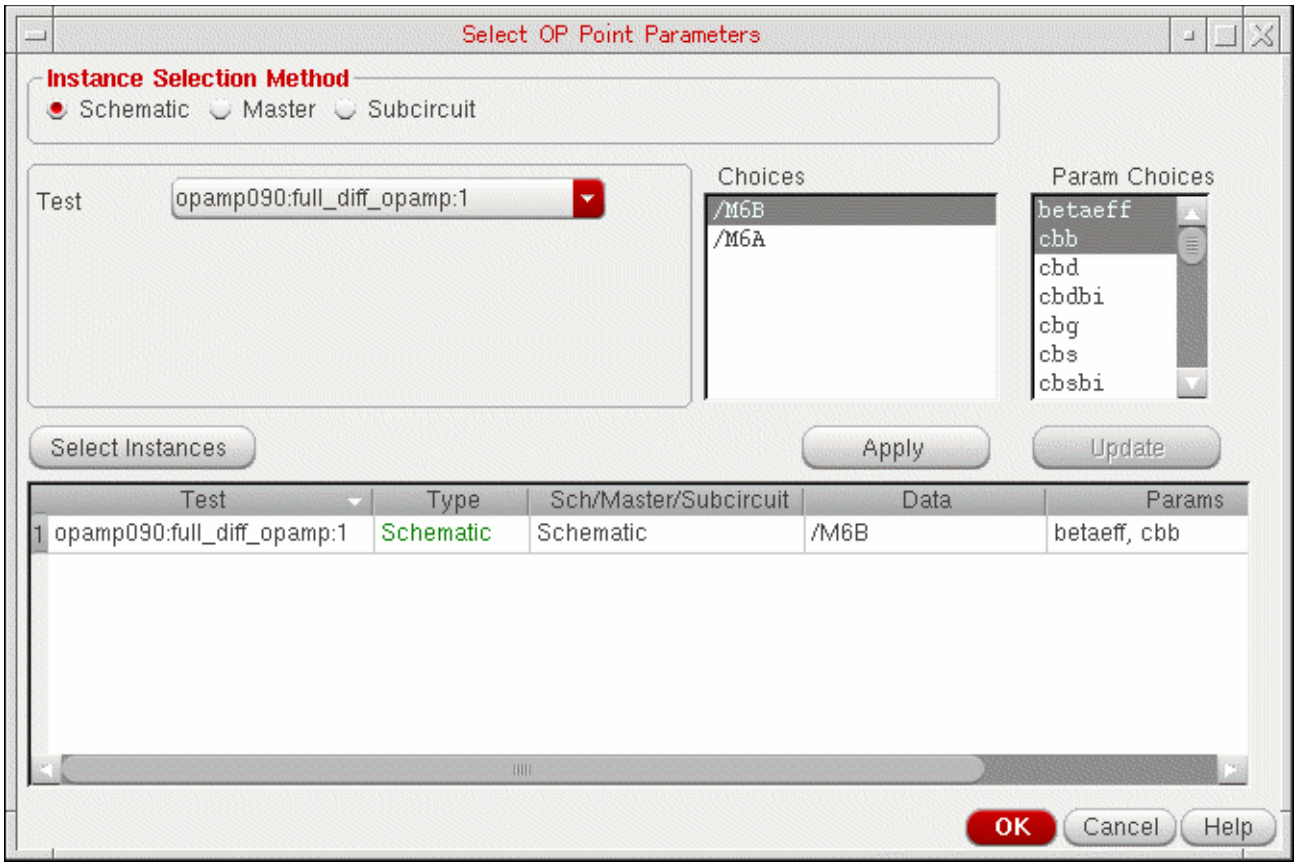
7. In the *Param Choices* field, select the DC operating point parameter you want to add.

© 2023 Cadence Design Systems, Inc.

To select multiple DC operating point parameters, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontinguous selection) and click the next parameter to add it to the selection set.

8. Click *Apply*.

   The selected parameters are added for the test. For example, in the following figure, the DC operating point parameters `betaeff` and `cbb` of instance `M6B` are added for the `opamp090:full_diff_opamp:1` test.



See also:

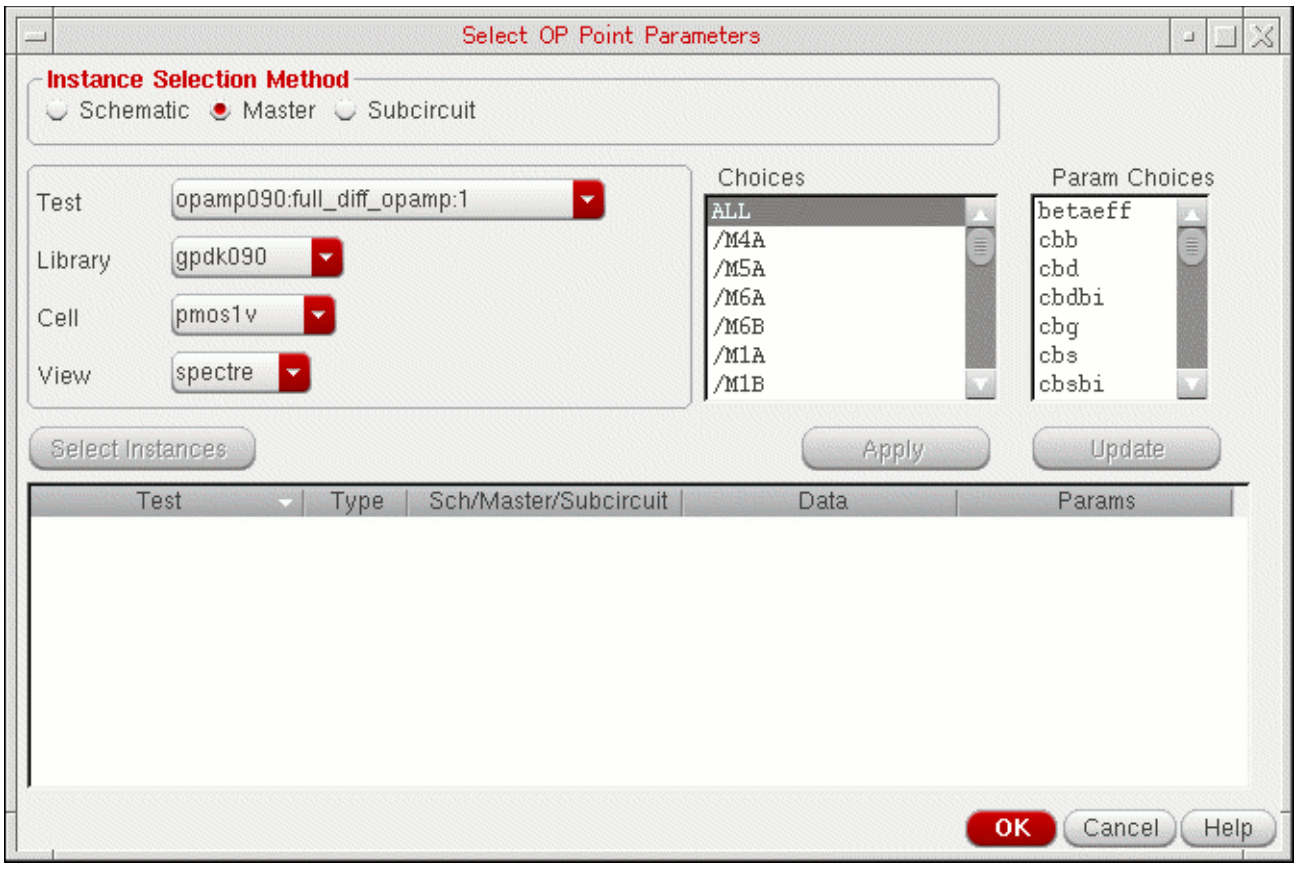
- Replacing DC Operating Point Parameters on page 1523
- Deleting DC Operating Point Parameters on page 1523

Adding DC Operating Point Parameters of Cellview Instances

To add DC operating point parameters of instances of cellviews, do the following:

© 2023 Cadence Design Systems, Inc.

1. Select the *Master* option.



2. In the *Test* drop-down list, choose the test for which you want to select instances of cellviews.

   To select instances for all the tests that are enabled in the Data View pane, choose *All Tests* in the *Test* drop-down list.

3. Use the *Library*, *Cell* and *View* drop-down lists to select the library, cell and view in which the cellview exists.

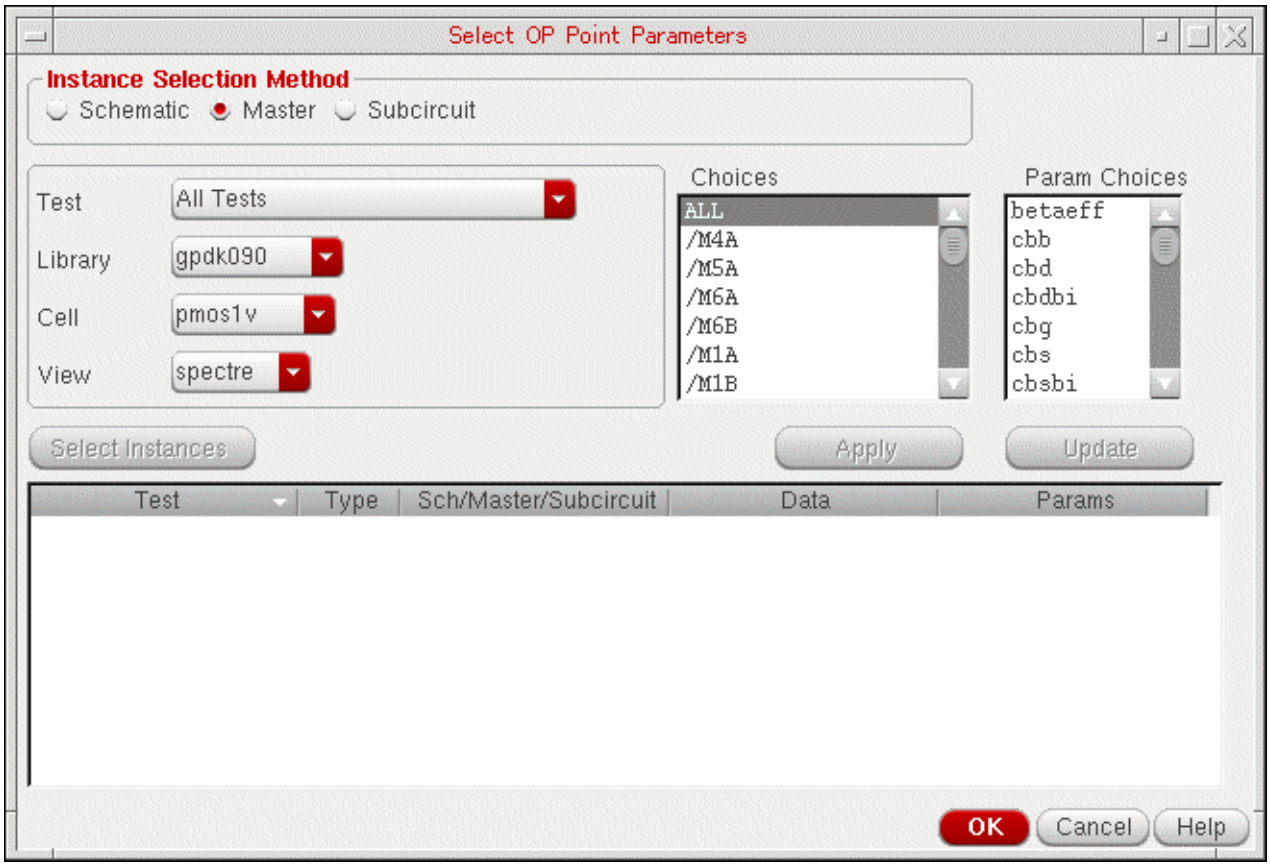
   All the instances of the cellview in the design for the test are displayed in the *Choices* field.

   Note the following:

   - The *Library*, *Cell* and *View* drop-down lists display only the libraries and cells for the cellviews that have DC operating point parameters.
   - If *All Tests* is selected in the *Tests* drop-down list, the *Choices* field displays the instance names of the cellview in the designs for all the tests that are enabled in the Data View pane. For example, in the following figure, the *Choices* field displays the

© 2023 Cadence Design Systems, Inc.

instance names of the `spectre` cellview of the `pmos1v` cell in the designs for all the tests.



Note: You can place the mouse pointer in the *Choices* field to view the instance name of the cellview in the design for each test.

4. In the *Choices* field, select the instance for which you want to add DC operating point parameters.

   The DC operating point parameters for the instance are displayed in the *Param Choices* field.

   Note the following:

   - To add DC operating point parameters for all instances of the cellview, select *ALL* in the *Choices* field.
   - If the same DC operating point parameter exists in more than one instance, you can simultaneously add the parameter for all the instances by selecting those instances in the *Choices* field. To select multiple instances, hold down the *Shift* key (for

© 2023 Cadence Design Systems, Inc.

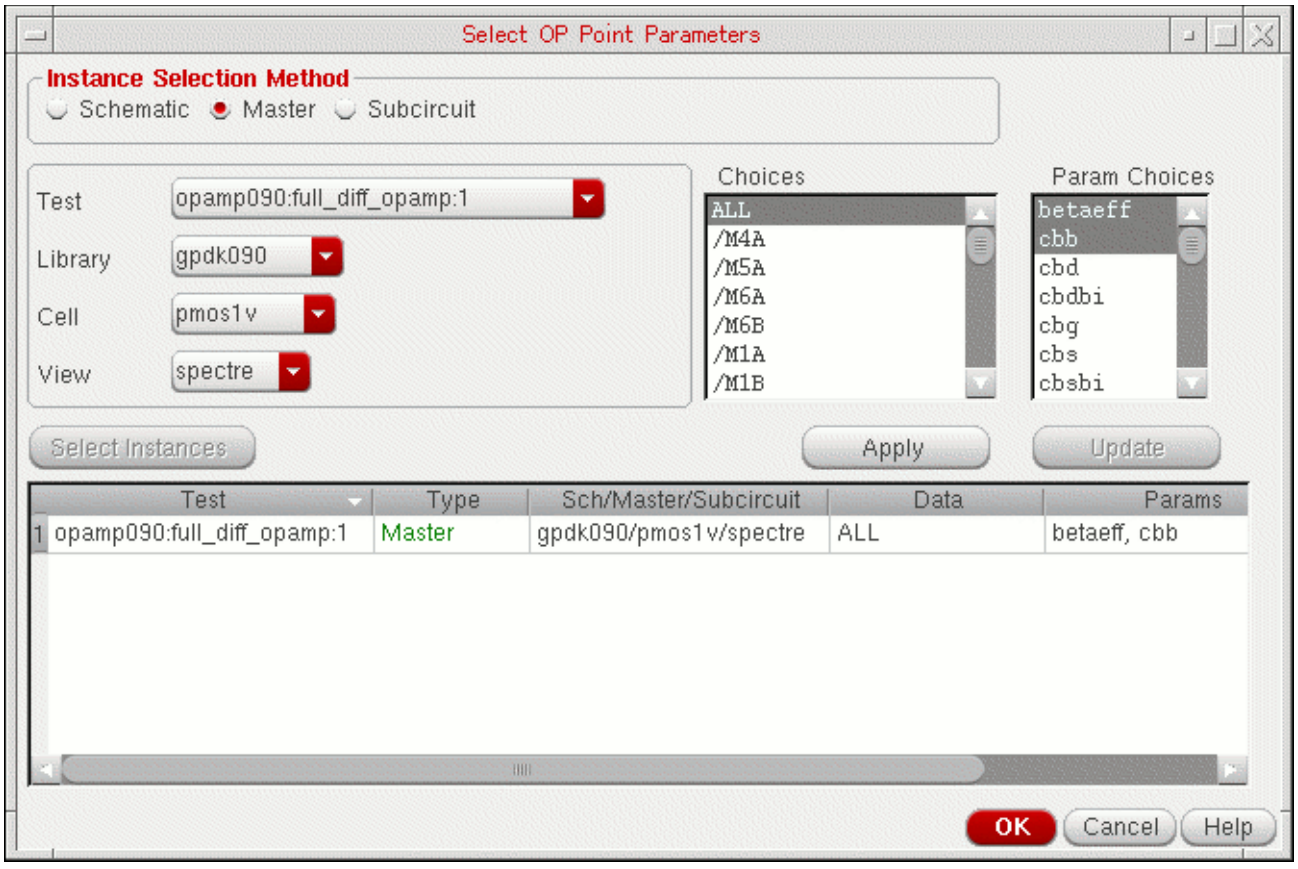
contiguous selection) or the *Ctrl* key (for noncontinguous selection) and click the next instance to add it to the selection set.

5. In the *Param Choices* field, select the DC operating point parameter you want to add.

   To select multiple DC operating point parameters, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontinguous selection) and click the next parameter to add it to the selection set.

6. Click *Apply*.

   The selected parameters are added for the test. For example, in the following figure, the DC operating point parameters `betaeff` and `cbb` on all instances of the `spectre` cellview of the `pmos1v` cell are added for the `opamp090:full_diff_opamp:1` test.



See also:

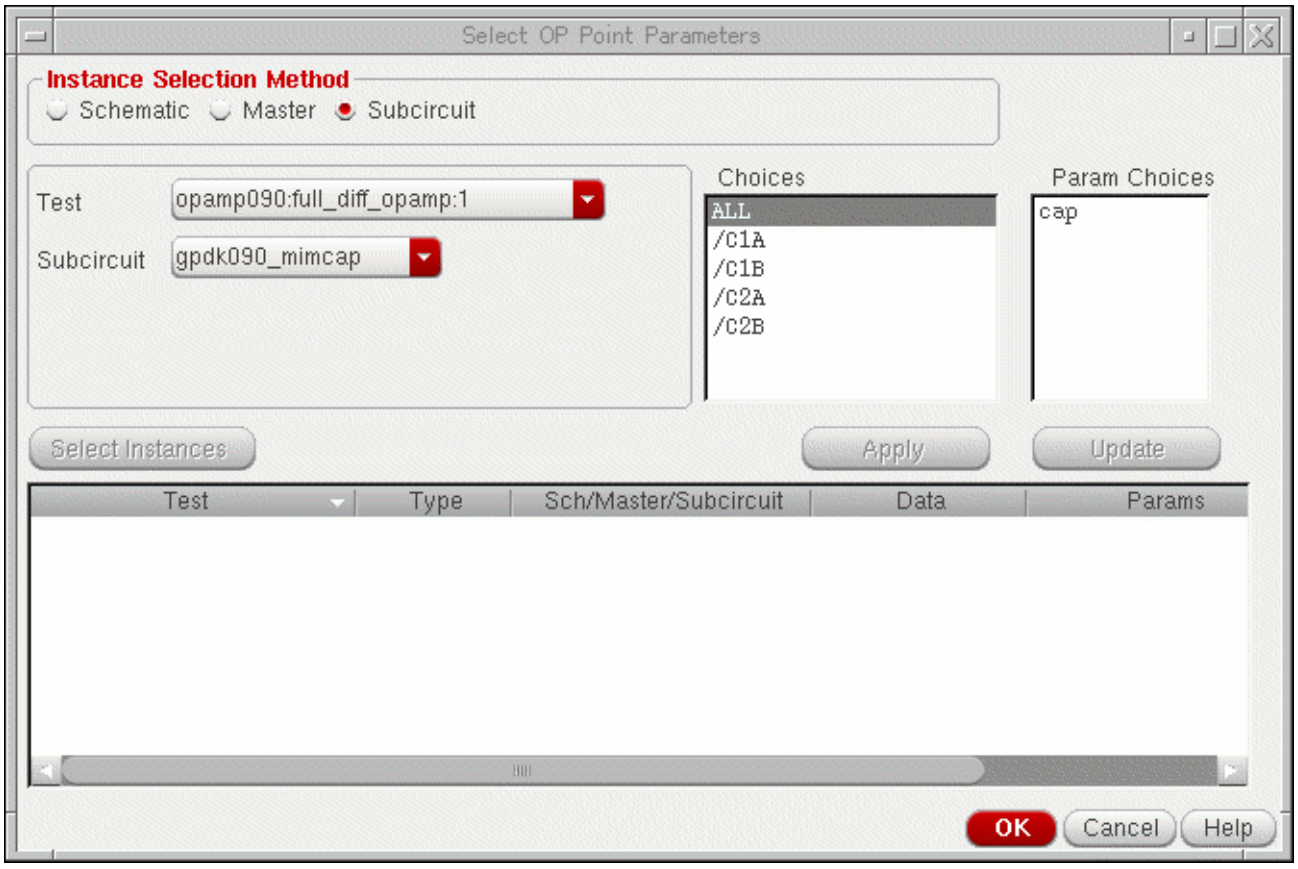
- Replacing DC Operating Point Parameters on page 1523
- Deleting DC Operating Point Parameters on page 1523

© 2023 Cadence Design Systems, Inc.

Adding DC Operating Point Parameters of Subcircuit Instances

To add DC operating point parameters of subcircuit instances, do the following:

1. Select the *Subcircuit* option.



2. In the *Test* drop-down list, choose the test for which you want to select subcircuit instances.

   To select subcircuit instances for all the tests that are enabled in the Data View pane, choose *All Tests* in the *Test* drop-down list.

3. In the *Subcircuit* drop-down list, choose the subcircuit whose instances you want to select.

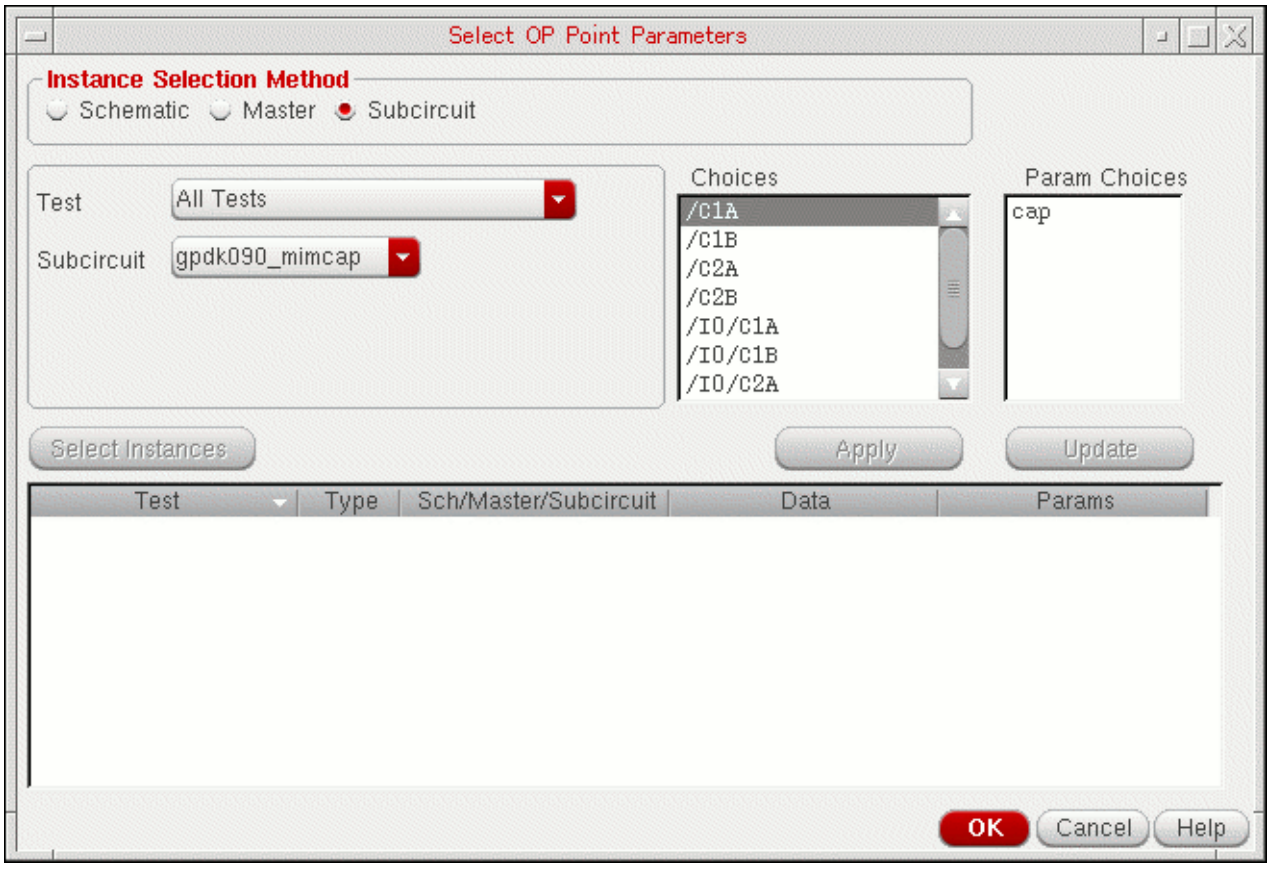
   All the instances of the subcircuit in the design are displayed in the *Choices* field.

   Note the following:

   - The *Subcircuit* drop-down list displays only the subcircuits that have DC operating point parameters.

© 2023 Cadence Design Systems, Inc.

- If *All Tests* is selected in the *Tests* drop-down list, the *Choices* field displays the instance names of the subcircuit in the designs for all the tests. For example, in the following figure, the *Choices* field displays the instance names of the `ampn` subcircuit in the designs for all the tests.



  You can place the mouse pointer in the *Choices* field to view the instance name of the subcircuit in the design for each test.

4. In the *Choices* field, select the instance for which you want to add DC operating point parameters.

   The DC operating point parameters for the instance are displayed in the *Param Choices* field.

© 2023 Cadence Design Systems, Inc.

Tip

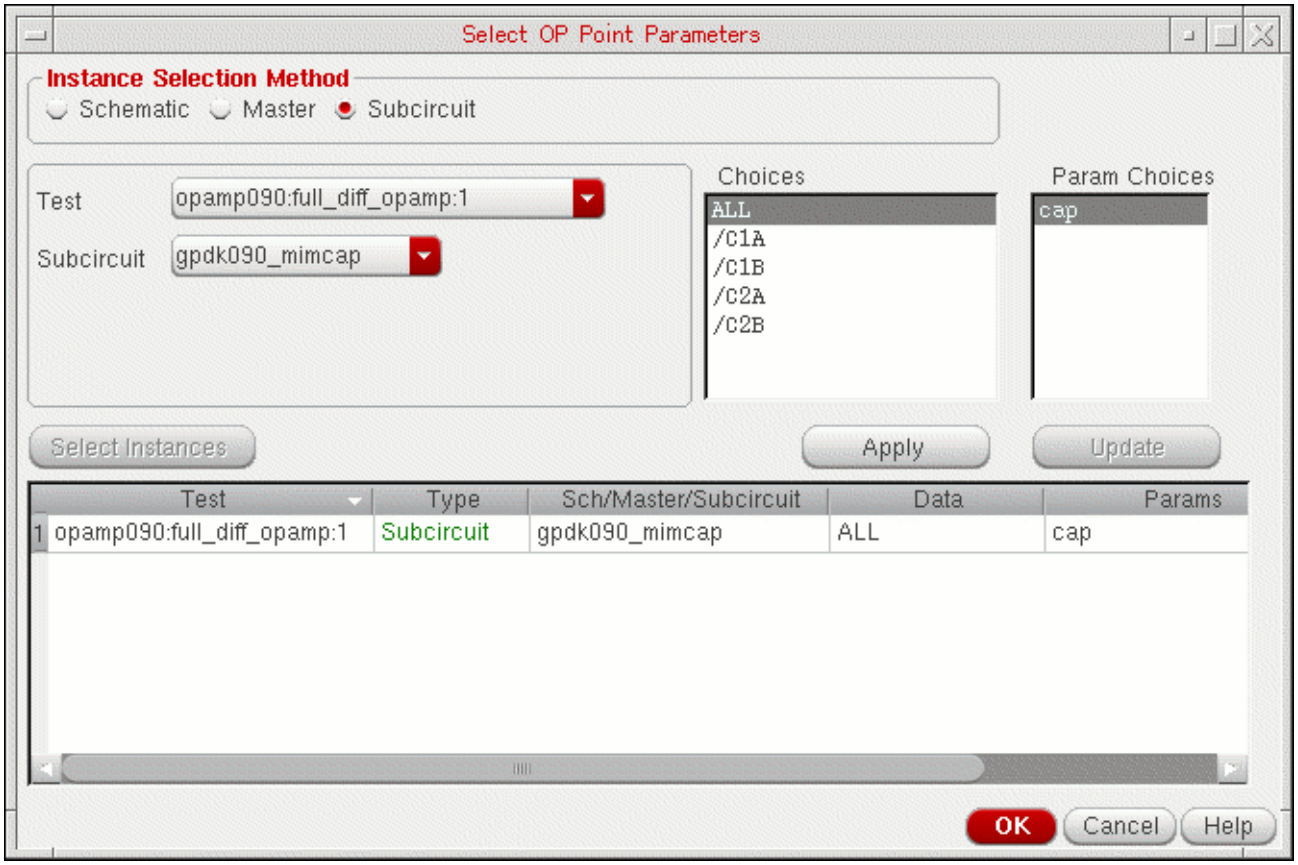
If the same DC operating point parameter exists in more than one instance, you can simultaneously add the parameter for all the instances by selecting those instances in the *Choices* field. To select multiple instances, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontinguous selection) and click the next instance to add it to the selection set.

5. In the *Param Choices* field, select the DC operating point parameter you want to add.

   To select multiple DC operating point parameters, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontinguous selection) and click the next parameter to add it to the selection set.

6. Click *Apply*.

   The selected parameters are added for the test. For example, in the following figure, the DC operating point parameter `cap` on all instances of the `gpdk090_mimcap` subcircuit are added for the `opamp090:full_diff_opamp:1` test.



© 2023 Cadence Design Systems, Inc.

See also:

- Replacing DC Operating Point Parameters on page 1523
- Deleting DC Operating Point Parameters on page 1523

Replacing DC Operating Point Parameters

To replace one DC operating point parameter with another, do the following:

1. Select the row for the parameter.
2. Select a parameter in the *Param Choices* field.
3. Click the *Update* button.

   The parameter in the row is replaced with the parameter selected in the *Param Choices* field.

Deleting DC Operating Point Parameters

To delete DC operating point parameters, do the following:

➡ Right-click the row for a parameter and choose *Delete*.

To delete multiple parameters, hold down the *Shift* key (for contiguous selection) or the *Ctrl* key (for noncontinguous selection), click the rows for the parameters you want to delete, then right-click and choose *Delete*.

To delete all parameters, do the following:

1. Right-click the row for a parameter and choose *Select All*.
2. Right-click the row for a parameter and choose *Delete*.

© 2023 Cadence Design Systems, Inc.

Viewing Sensitivity Analysis Results

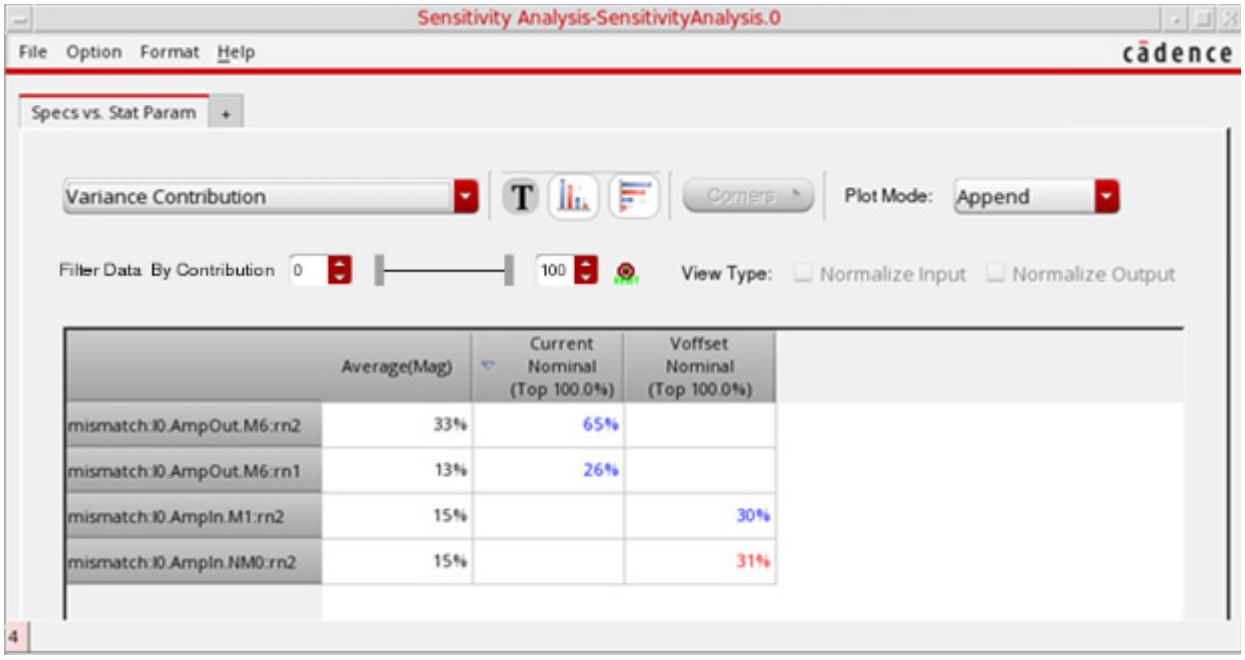
After the Sensitivity Analysis or the Monte Carlo run is complete, the tool automatically opens the Sensitivity Analysis window with sensitivity analysis results displayed in it.

To view sensitivity data for a sensitivity analysis run:

➡ Click *Sensitivity Results* on the *Results* tab.

Alternatively, on the *History* tab of the Data View pane, right-click the history of a Sensitivity Analysis or Monte Carlo run and choose *Sensitivity Results*.

The Sensitivity Analysis window appears, displaying results of sensitivity analysis.



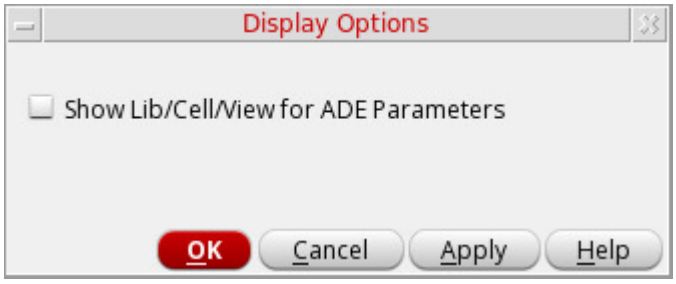
Viewing the Library, Cell, and View Information of the Device Parameters

To view the information about the library cellviews of the device parameters, do the following:

1. In the *Sensitivity Analysis* window, choose *Option – Display*.

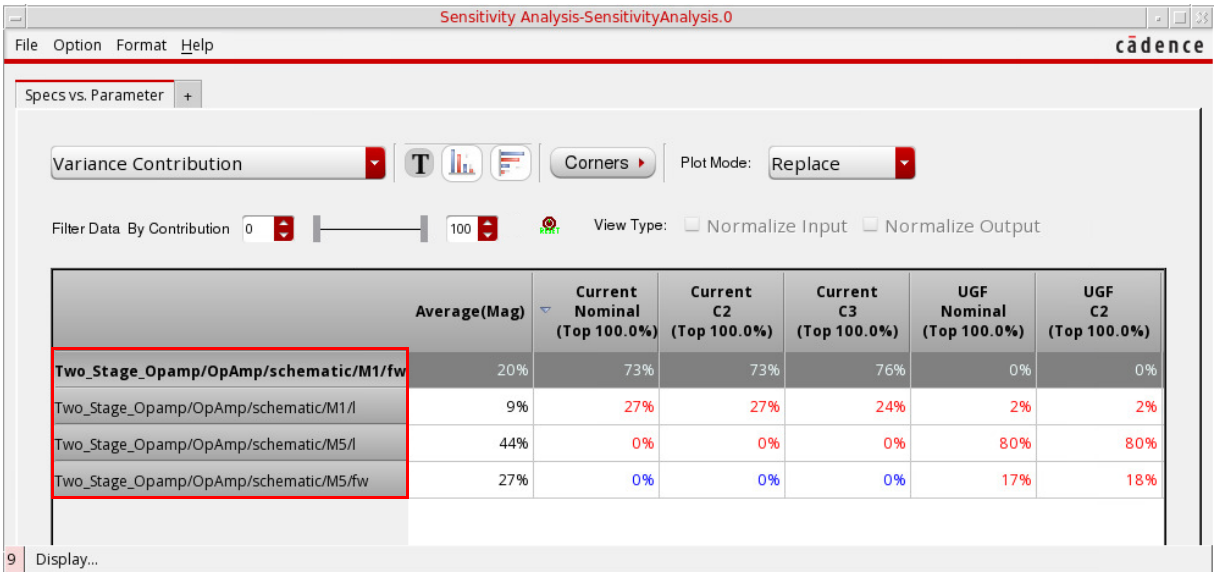
© 2023 Cadence Design Systems, Inc.

The *Display Options* form is displayed.



2. Select the *Show Lib/Cell/View for ADE Parameters* check box, and then click *OK*.

   The library/cell/view information about the device parameters are displayed.



Reading Data in the Sensitivity Analysis window

In the Sensitivity Analysis window, sensitivity data is displayed in a tabular format, where each data cell shows the sensitivity of a spec or parameter in the corresponding column to the spec or parameter displayed in the corresponding row. The specs or parameters displayed in the inputs and outputs depend on the selected tab.

By default, the data shows sensitivity of each specification with respect to the available device and statistical parameters. This view is governed by the *Specs vs. Parameter* tab. To know more about the other available tabs, refer to Viewing Sensitivity Data in Different Tabs.

© 2023 Cadence Design Systems, Inc.

If the analysis was run on various corners, a column is added for each corner condition. For example, in the figure shown above, data for the `gainBwProd` spec is shown for all the three corners, `nominal`, `C0_0`, and `C0_2`.

Note: If you disable any test for a corner in the **Corners Setup** form, specification columns for that test and corner combination are not shown in the Sensitivity Analysis window.

There are various ways in which you can customize the way in which the sensitivity data is displayed. For more details on how to customize the view, refer to Working with Sensitivity Data.

For details on how to plot sensitivity results, refer to Plotting Sensitivity Analysis Results.

Viewing Sensitivity Data in Different Tabs

The Sensitivity Analysis window provides the following tabs for viewing sensitivity data:

- Specs vs. Specs
- Specs vs. Parameter
- Specs vs. Stat Param
- Specs vs. DC Ops
- DC Ops vs. Parameter
- DC Ops vs. Stat Param
- Stat Param vs. Stat Param

In these tabs, the input factor is displayed in the rows and the output is displayed in the columns. For example, in the Specs vs. Parameter tab shown in Figure on page 1524, parameters are input factors and specifications are the output. Therefore, parameters are displayed in the rows and specifications are displayed in the columns. For information about working with the data in the tabs, see Working with Sensitivity Data on page 1530.

Note: The text `N.A.` in a cell in the Sensitivity Analysis window indicates that the sensitivity value could not be calculated because of some reason. You can place the mouse pointer on the cell to view the reason in a tooltip.

© 2023 Cadence Design Systems, Inc.

These tabs are described below:

Tab Name	Description
Specs vs. Specs	Displays the sensitivity of each specification with respect to other specifications. Do the following to display the tab: **1.** From the drop-down list, choose *Specification vs. Specification*. **2.** Click the + button to display the Specs vs. Specs tab.
Specs vs. Parameter	Displays the sensitivity of each specification with respect to device and statistical parameters. Do the following to display the tab: **1.** From the drop-down list, choose *Specification vs. Parameter*. **2.** Click the + button to display the Specs vs. Parameter tab. **Note:** The sensitivity of statistical parameters is displayed in this tab only if sensitivity data for statistical parameters exists in the results database. See also, Hiding or Showing Statistical Parameters on page 1550.
Specs vs. Stat Param	Displays the sensitivity of each specification with respect to statistical parameters. Do the following to display the tab: **1.** From the drop-down list, choose *Specification vs. Statistical Parameter*. **Note:** The *Specification vs. Statistical Parameter* option is available only if sensitivity data for statistical parameters exists in the results database. **2.** Click the + button to display the Specs vs. Stat Param tab.

© 2023 Cadence Design Systems, Inc.

Tab Name	Description
Specs vs. DC Ops	Displays the sensitivity of each specification with respect to DC operating point parameters. Do the following to display the tab: **1.** From the drop-down list, choose *Specification vs. DC OP Point Parameter*. **Note:** The *Specification vs. DC OP Point Parameter* option is available only if sensitivity data for DC operating point parameters exists in the results database. **2.** Click the + button to display the Specs vs. DC Ops tab.
DC Ops vs. Parameter	Displays the sensitivity of each DC operating point parameter with respect to device and statistical parameters. Do the following to display the tab: **1.** From the drop-down list, choose *DC OP Point Parameter vs. Parameter*. **Note:** The *DC OP Point Parameter vs. Parameter* option is available only if sensitivity data for DC operating point parameters exists in the results database. **2.** Click the + button to display the DC Ops vs. Parameter tab. **Note:** The sensitivity of statistical parameters is displayed in this tab only if sensitivity data for statistical parameters exists in the results database. See also, Hiding or Showing Statistical Parameters on page 1550.

© 2023 Cadence Design Systems, Inc.

Tab Name	Description
DC Ops vs. Stat Param	Displays the sensitivity of each DC operating point parameters with respect to statistical parameters. Do the following to display the tab: **1.** From the drop-down list, choose *DC OP Point Parameter vs. Statistical Parameter*. **Note:** The *DC OP Point Parameter vs. Statistical Parameter* option is available only if sensitivity data for DC operating point parameters and statistical parameters exists in the results database. **2.** Click the + button to display the DC Ops vs. Stat Param tab.
Stat Param vs. Stat Param	Displays the sensitivity of each statistical parameter with respect to other statistical parameters. Do the following to display the tab: **1.** From the drop-down list, choose *Statistical Parameter vs. Statistical Parameter*. **Note:** The *Statistical Parameter vs. Statistical Parameter* option is available only if sensitivity data for statistical parameters exists in the results database. **2.** Click the + button to display the Stat Param vs. Stat Param tab.

© 2023 Cadence Design Systems, Inc.

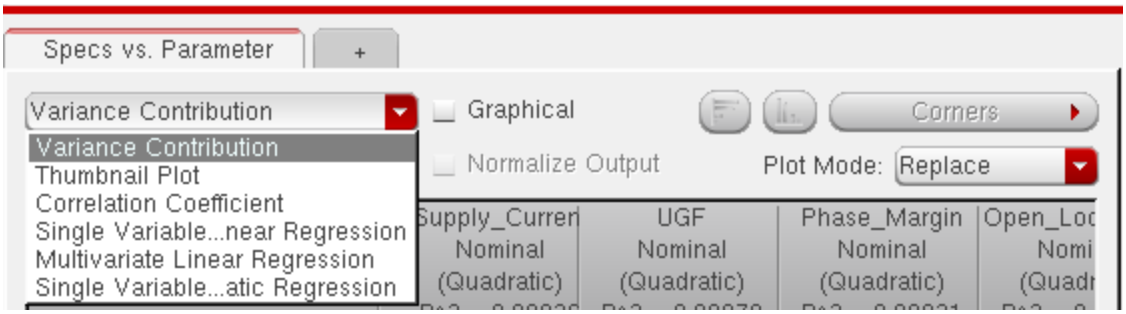
Working with Sensitivity Data

This following topics describe how you can work with the data in the Sensitivity Analysis window:

- Changing the Type and Format of Data To View on page 1530
- Changing the Data View Type on page 1544
- Filtering Data Based on Correlation Coefficient on page 1545
- Filtering Data Based on Search Criteria on page 1547
- Sorting Data on page 1548
- Hiding Corner Data on page 1549
- Hiding Specification Data on page 1549
- Hiding or Showing the Detailed Results for Corners on page 1549
- Hiding or Showing the Detailed Results for Specifications on page 1550
- Hiding or Showing Statistical Parameters on page 1550
- Showing All Hidden Data on page 1550
- Highlighting Associated Devices in the Schematic on page 1550
- Viewing Results in Multiple Sensitivity Analysis Windows on page 1551
- Changing Number Format on page 1551

Changing the Type and Format of Data To View

By default, the Sensitivity Analysis window shows the variance contribution data in percentage format. You can also view an overview of the data in thumbnail form or view the correlation and regression data by selecting an option from the drop-down list.



© 2023 Cadence Design Systems, Inc.

The Sensitivity Analysis window shows both correlation and regression data in the percentage format. This Sensitivity Analysis window is displayed when you analyze data by using the one-factor-at-a-time (OFAT) method.

When you analyze data during a Monte Carlo or Hammersley sampling run, you will notice an additional format, *Multivariate Linear Regression*, in the drop-down list.

In the one-factor-at-a-time (OFAT) method, a single variable linear regression model is followed, which means that an individual independent variable is created for each row in the sensitivity table. However, in the multivariate linear regression method, one common model is applied to all the rows in the sensitivity table.

To change the format of data to the graphical format:

➤ Select the *Graphical* checkbox next to the drop-down list at the top of a tab.

For more details on different types and formats of data in the Sensitivity Analysis window, see:

- Viewing Correlation Data on page 1533
- Viewing Correlation Data in Graphical Format on page 1534
- Viewing Thumbnail Plots on page 1536
- Viewing Regression Data on page 1538
- Viewing Normalized Regression Data on page 1540
- Viewing Regression Data in Graphical Format on page 1543

Viewing Variance Contribution Data

Variance contribution uses a similar method as used by mismatch contribution (refer to Mismatch Contribution Analysis).

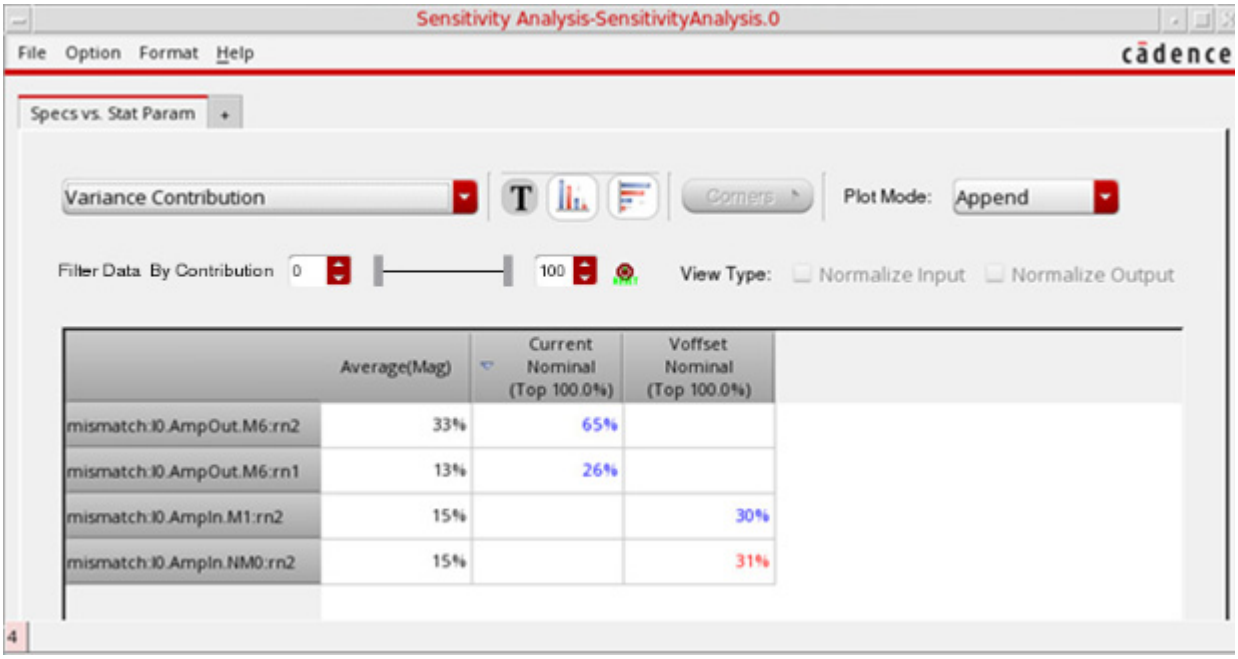
Mismatch contribution is applied to Monte Carlo results, where as variance contribution is used primarily to view the Sensitivity Analysis results; however, it can also be applied to Single run, Sweeps and Corners runs, Local Optimization, and Global Optimization runs.

To view the variance contribution data:

➤ Choose *Variance Contribution* from the drop-down list at the top of a tab.

© 2023 Cadence Design Systems, Inc.

The variance contribution data appears as shown in the figure below. By default, this data is displayed in the percentage format.



The first column in the results table lists the variables, and the *Average(Mag)* column displays the average of percentage values of the variance contribution for each specification. Each cell in the other columns displays the variance contribution values. Positive values are displayed in blue and negative values are displayed in red.

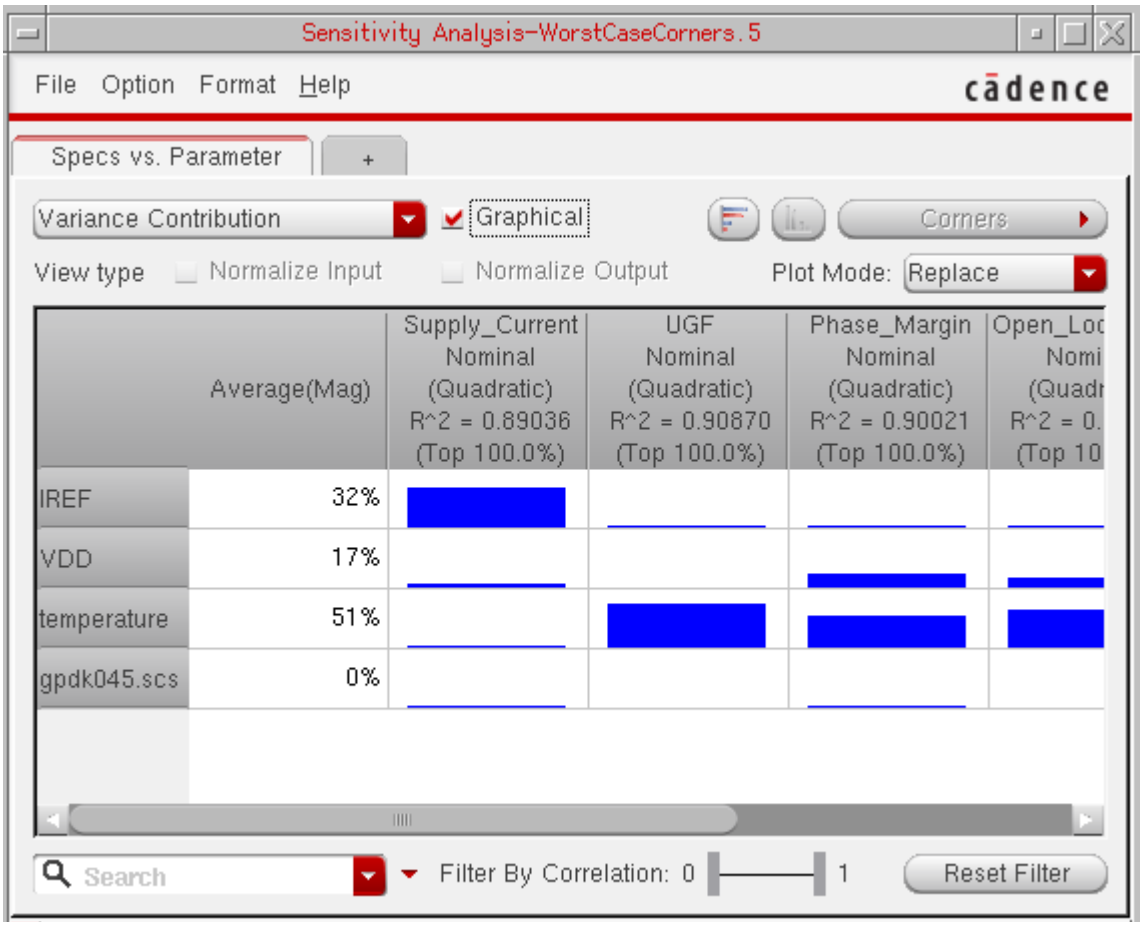
You can also filter sensitivity analysis results data by contribution values by typing the specific contribution values in the *Filter Data by Contribution* boxes.

Viewing Variance Data in Graphical Format

When the *Graphical* check box is selected, the variance contribution data is displayed as shown in the following figure.

© 2023 Cadence Design Systems, Inc.

Figure 25-1 Variance Data Displayed in the Graphical Format



The first column lists the variables, and the *Average(Mag)* column displays the average of the fit values for each sensitivity in the row. Each cell in the other columns in the table displays a symbol that portrays:

- whether the relationship is positive or negative
- how tightly the data fits to a line

Viewing Correlation Data

Correlation data tells how correlated each goal is to that variable. If the correlation is close to 1 (or -1), then they are highly correlated. However, if the correlation is 0, then the goal and variable are not correlated. The correlation value is always between -1 and 1.

To view correlation data:

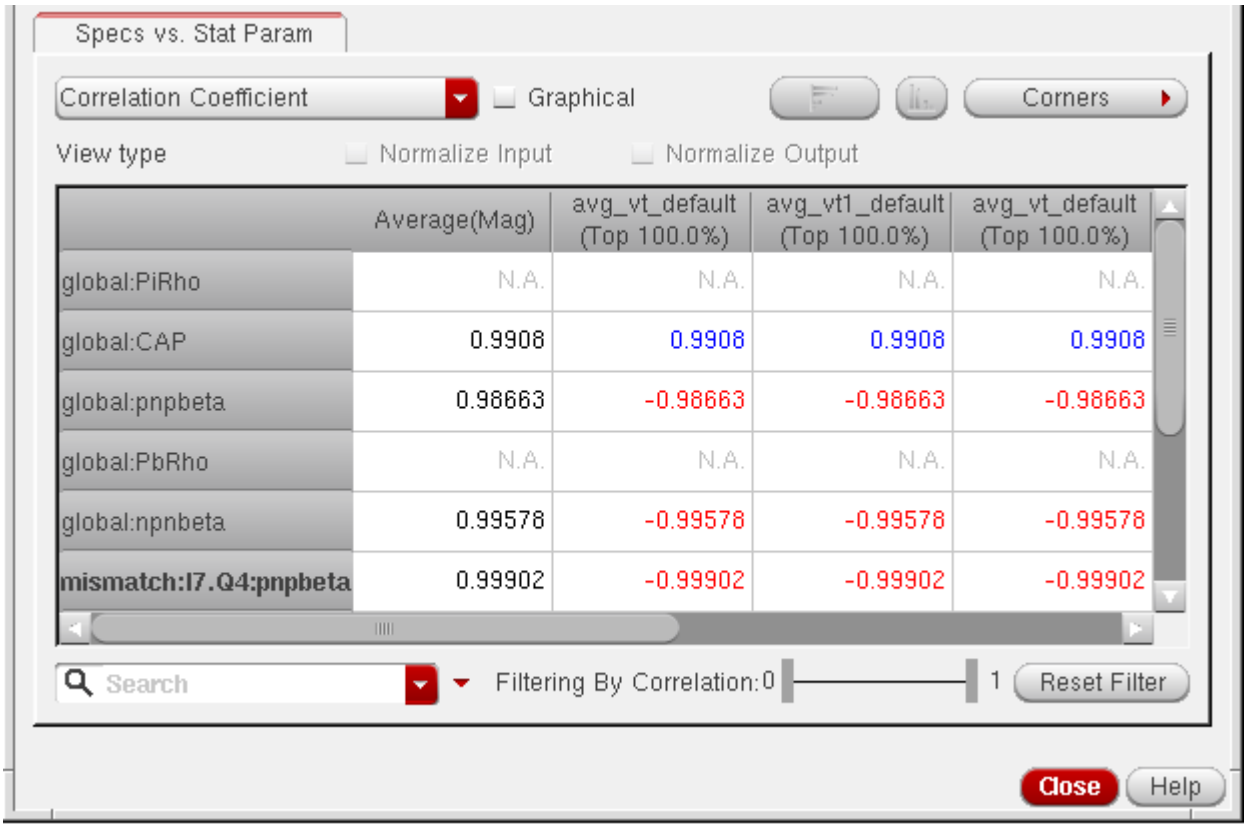
© 2023 Cadence Design Systems, Inc.

➤ Choose `Correlation Coefficient` from the drop-down list at the top of a tab.

The correlation data appears. By default, this data is in the graphical format. For more details on this, refer to Viewing Variance Data in Graphical Format on page 1532. To view the correlation data in numeral format, clear the *Graphical* checkbox.

Correlation data in numeral format is displayed as shown in the following figure.

Figure 25-2 Correlation Data in Sensitivity Analysis window



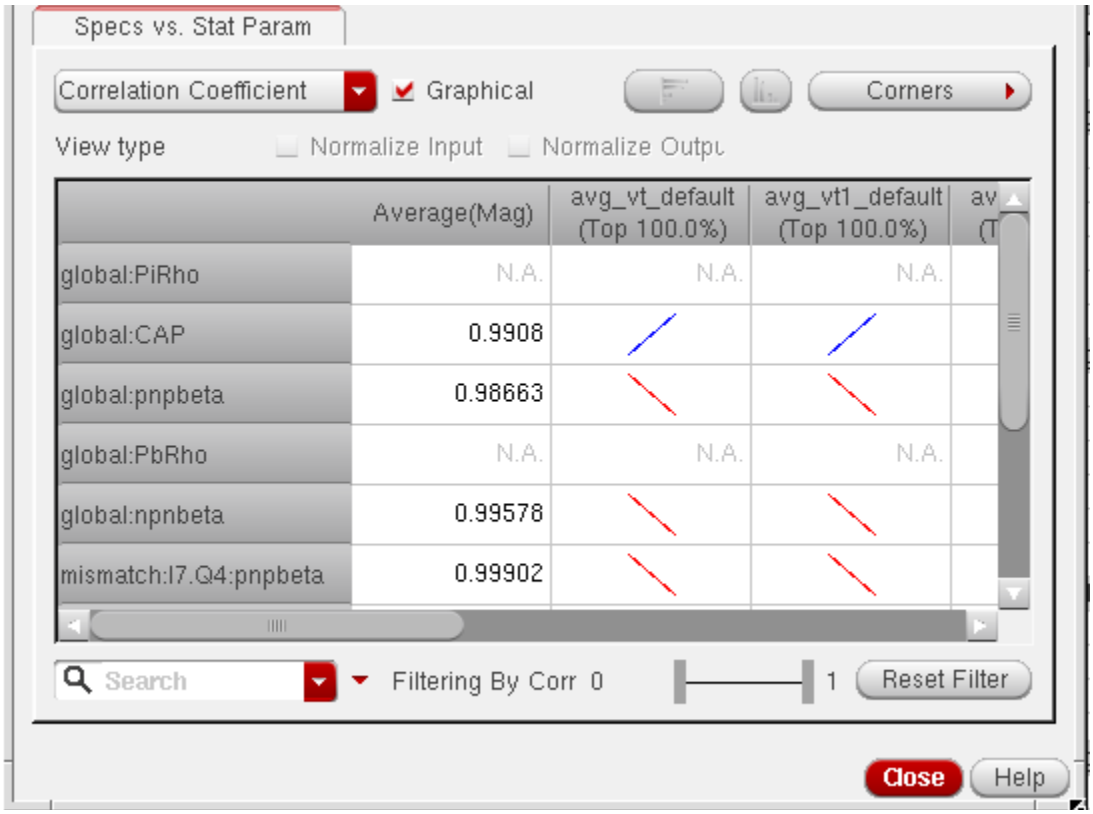
The first column lists the variables, and the *Average(Mag)* column displays the average of absolute values of correlation coefficients for each specification. Each cell in the other columns in the table displays the correlation values. Positive correlation values are displayed in blue color and negative values are displayed in red color.

Viewing Correlation Data in Graphical Format

When the *Graphical* checkbox is selected, the correlation data is displayed as shown in the following figure.

© 2023 Cadence Design Systems, Inc.

Figure 25-3 Correlation Data Displayed in the Graphical Format



The first column lists the variables, and the *Average(Mag)* column displays the average of the fit values for each sensitivity in the row. Each cell in the other columns in the table displays a symbol that portrays:

- whether the relationship is positive or negative
- how tightly the data fits to a line

The polarity of the relationship is indicated by the direction of the slope of the line. A blue line that slopes from the lower left to upper right is positive. A red line that slopes from the upper left to the lower right is negative.

Positive Negative

© 2023 Cadence Design Systems, Inc.

The tool illustrates how tightly the data fits to a line using the width and intensity of the shape. The narrower and brighter the line, the tighter the fit.

Tighter Fit Looser Fit

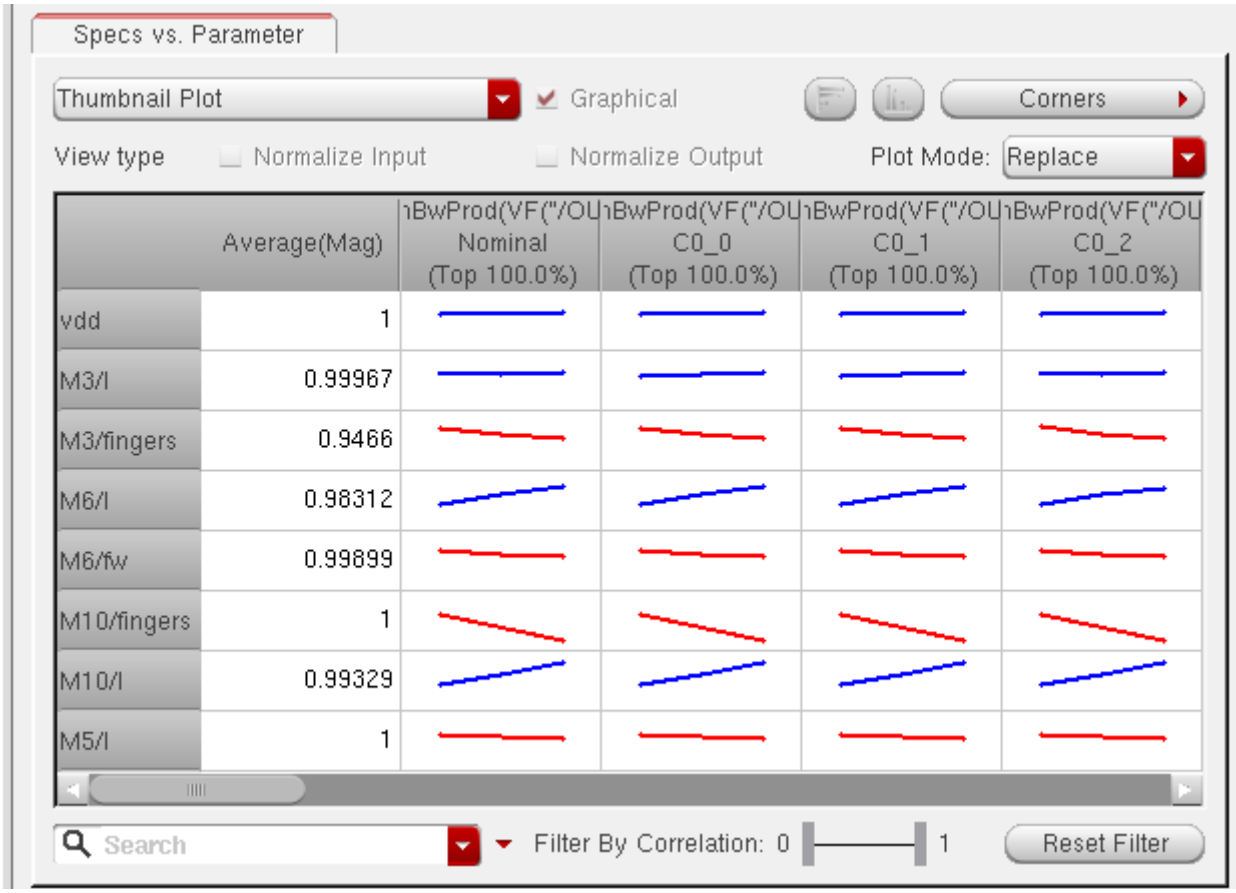
Viewing Thumbnail Plots

The default graphical view is the correlation data, which shows the positive or negative relation between the inputs and outputs. However, it does not show the view of exact relationship between the two. To show a better graphical representation of the relationship between the inputs and outputs, you can choose to show thumbnail images of the plots of the Sensitivity Analysis or Monte Carlo run results.

To show thumbnail plots, choose `Thumbnail Plots` from the drop-down list at the top of a tab. With this option, thumbnail plots of the results data are displayed in each cell, as shown in the following figure.

© 2023 Cadence Design Systems, Inc.

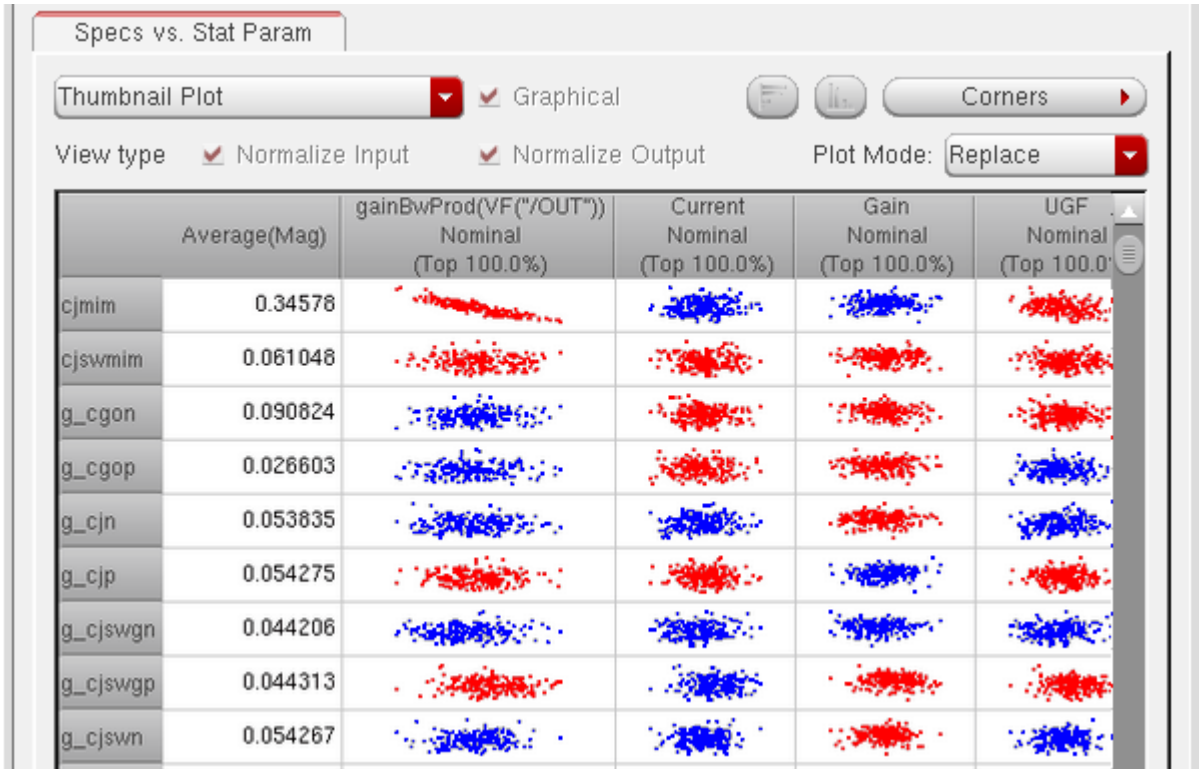
Figure 25-4 Thumbnail Plots with Line Graphs



By default, if the waveform contains ten points or less, a line plot is displayed, as shown in the figure above. For more than ten points, the thumbnail shows scatter plots, as shown in the following figure.

© 2023 Cadence Design Systems, Inc.

Figure 25-5 Thumbnail Plots with Scatter Graphs



The color of thumbnail plots depends on the sign of correlation coefficients. For positive correlation coefficients, the plots are displayed in blue and for negative values, they are displayed in red.

The thumbnail plot view gives you a view of all the plots at the same time. Click any thumbnail to view the plot in the Virtuoso Visualization and Analysis XL window.

Viewing Regression Data

The regression data shows the best fit line or the regression line that gives an output value which is close to the spec value.

The Sensitivity Analysis window allows you to view the following types of regression data.

- The single variable linear regression is the slope of the line between goal values and variable values that best fits the data points. Thus, a change in the variable multiplied by the regression co-efficient is the change that can be expected in the goal value.

  Note: The least square method is used to display the best fit line.

© 2023 Cadence Design Systems, Inc.

- The single variable quadratic regression uses a quadratic equation to fit the goal values with respect to the variable values. The coefficients from this option can be considered when the correlation values are close to zero and cannot be modelled by a straight line.
- The multivariate regression gives an estimate of the linear relationship between the output and all inputs.

To view regression data, choose one of the following from the drop-down list at the top of a tab:

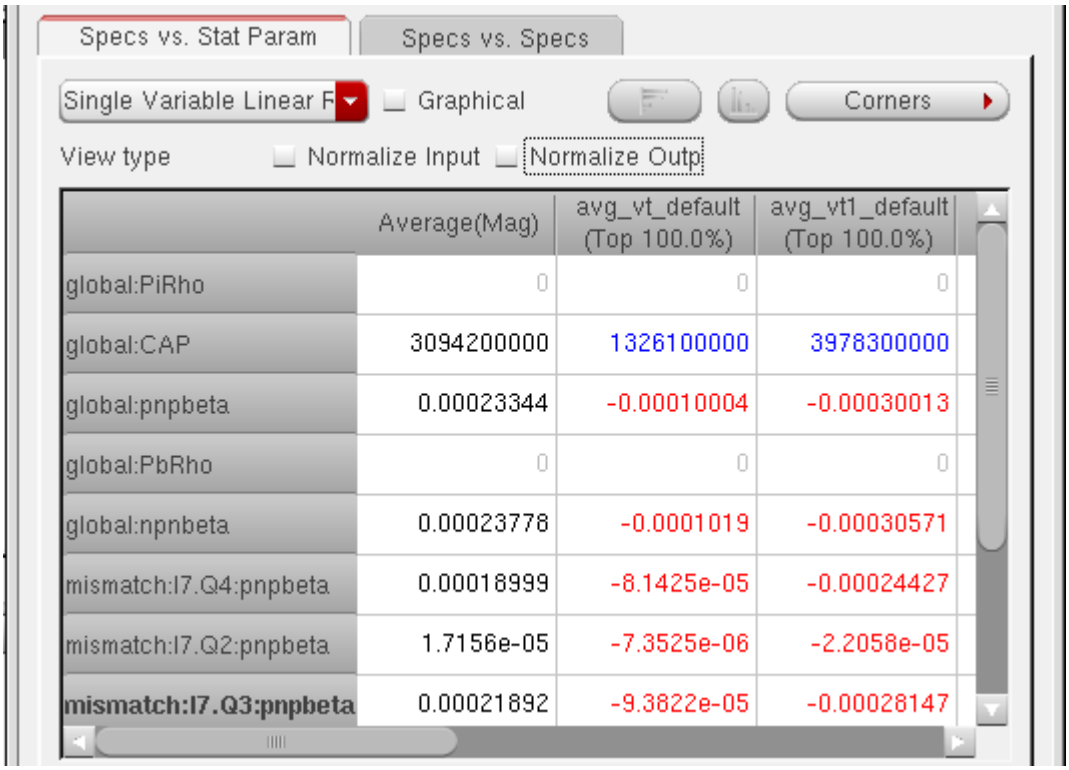
- `Single Variable Linear Regression` to view single variable linear regression data.
- `Single Variable Quadratic Regression` to view single variable quadratic regression data.
- `Multivariate Linear Regression` to view multivariate linear regression data.

  By default, the regression data appears in the graphical format. For more details, refer to Viewing Regression Data in Graphical Format on page 1543.

  Clear the *Graphical* checkbox to view the data in numeral format. The regression data appears as shown in the following figure.

© 2023 Cadence Design Systems, Inc.

Figure 25-6 Regression Data in Sensitivity Analysis window



	Average(Mag)	avg_vt_default (Top 100.0%)	avg_vt1_default (Top 100.0%)
global:PiRho	0	0	0
global:CAP	3094200000	1326100000	3978300000
global:pnpbeta	0.00023344	-0.00010004	-0.00030013
global:PbRho	0	0	0
global:npnbeta	0.00023778	-0.0001019	-0.00030571
mismatch:I7.Q4:pnpbeta	0.00018999	-8.1425e-05	-0.00024427
mismatch:I7.Q2:pnpbeta	1.7156e-05	-7.3525e-06	-2.2058e-05
mismatch:I7.Q3:pnpbeta	0.00021892	-9.3822e-05	-0.00028147

The first column lists the variables, and the *Average(Mag)* column displays the average of absolute values of regression coefficients for each specification. Each cell in the other columns in the table displays the regression values. Positive regression values are displayed in blue color and negative values are displayed in red color.

Viewing Normalized Regression Data

Normalized regression is the value of the regression coefficient calculated when either or both the input and the output parameter values are normalized.

- When only input parameters are normalized, we say it is input normalized regression coefficient.
- When only output parameters are normalized, we say it is output normalized regression coefficient.
- When both input and output parameters are normalized, we say it is input-output normalized regression coefficient.

ADE Assembler uses the following equations to normalize the data:

© 2023 Cadence Design Systems, Inc.

- To normalize input values for design variables and statistical variables:

  `Xnorm = 2*(X-Xmin)/(Xmax-Xmin)`

- To normalize input values for Monte Carlo analysis:

  `Xnorm = (X-Xmean)/Xstd`

- To normalize output values for design variables and statistical variables:

  `Ynorm = (Y - Ymean)/Ymean`, if `Ymean` != 0

Normalized regression coefficients allow comparing the results between different specs for the same parameter (output normalized), same specs for different parameters (input normalized), and also between different parameter/spec combinations (input-output normalized).

To view regression data:

1. Choose *Single Variable Linear Regression*, *Single Variable Quadratic Regression*, or *Multi Variate Linear Regression* from the drop-down list at the top of a tab.

   The regression data appears.

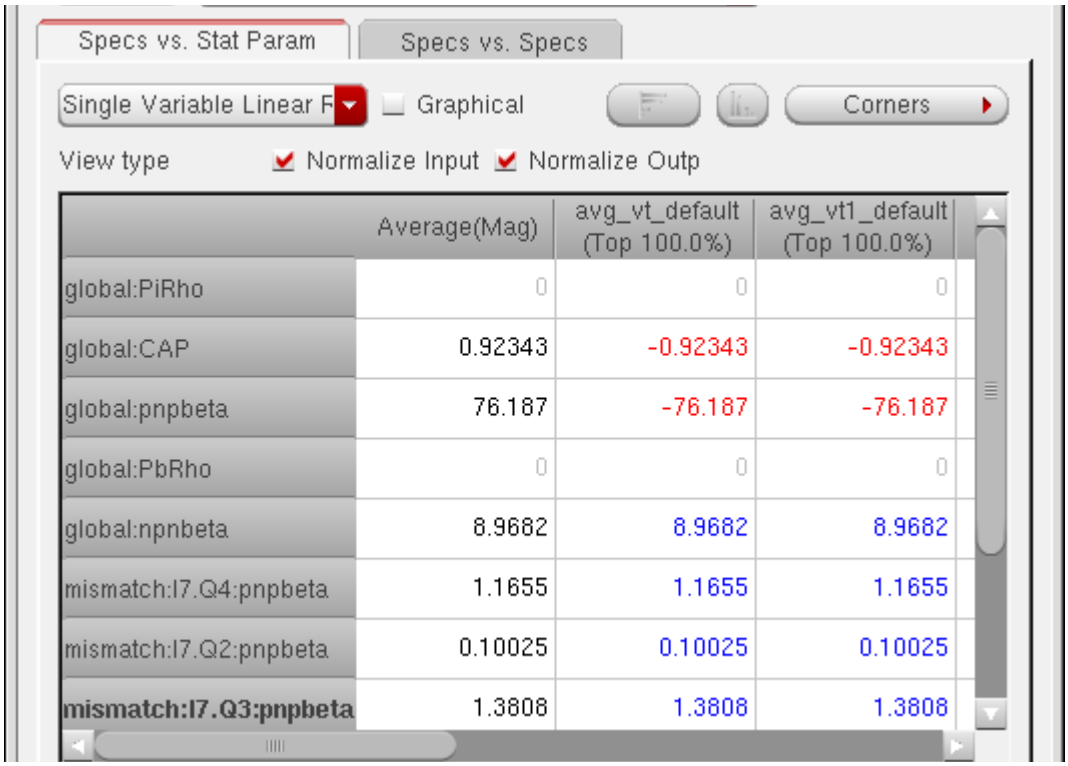
2. Do one of the following:

   - To normalize the input data, select the *Normalize Input* check box.
   - To normalize the output data, select the *Normalize Output* check box.
   - To normalize both the input and the output data, select the *Normalize Input* and *Normalize Output* check boxes.

   The regression data appears.

© 2023 Cadence Design Systems, Inc.

Figure 25-7 Regression Data in Sensitivity Analysis window



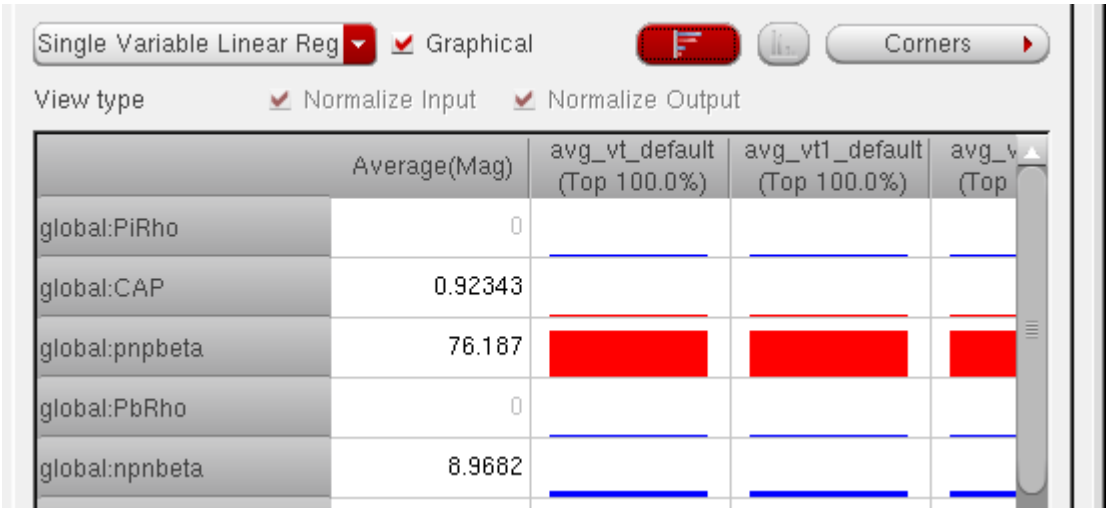
	Average(Mag)	avg_vt_default (Top 100.0%)	avg_vt1_default (Top 100.0%)
global:PiRho	0	0	0
global:CAP	0.92343	-0.92343	-0.92343
global:pnpbeta	76.187	-76.187	-76.187
global:PbRho	0	0	0
global:npnbeta	8.9682	8.9682	8.9682
mismatch:I7.Q4:pnpbeta	1.1655	1.1655	1.1655
mismatch:I7.Q2:pnpbeta	0.10025	0.10025	0.10025
mismatch:I7.Q3:pnpbeta	1.3808	1.3808	1.3808

The first column lists the variables, and the *Average(Mag)* column displays the average of absolute values of normalized regression coefficients for each specification.

© 2023 Cadence Design Systems, Inc.

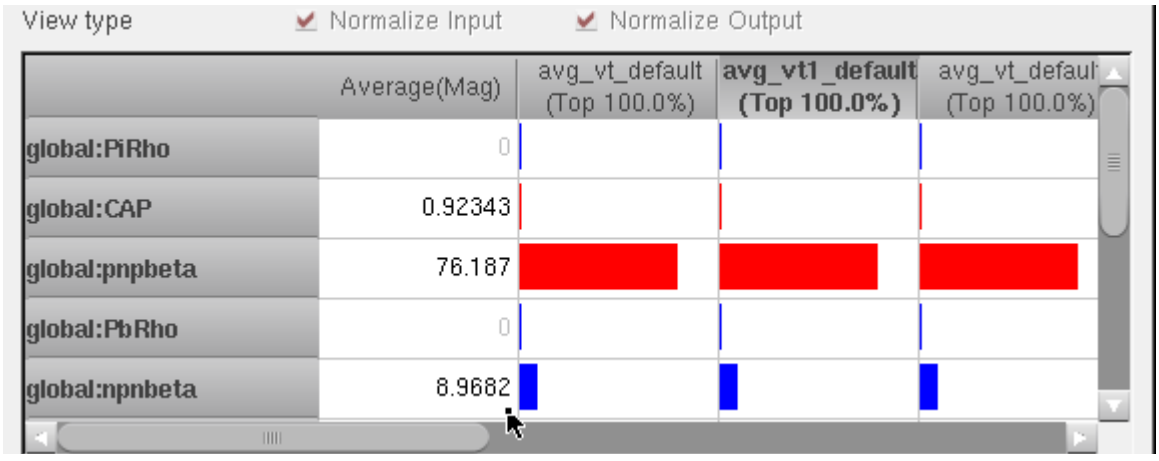
Viewing Regression Data in Graphical Format

In graphical format, the regression data is shown as bars. By default, the data is displayed as vertical bars, as shown in the following figure.



Note: Here, positive regression values are displayed in blue color and negative values are displayed in red color.

To display the regression data as horizontal bars, click the button next to the *Graphical* checkbox. The graphical data is changed as shown in the following figure.



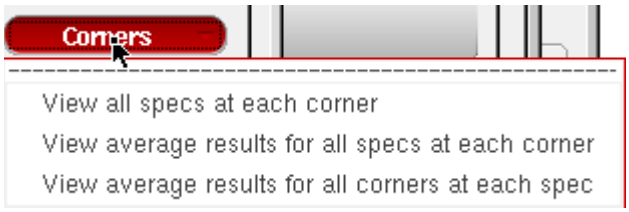
Important

When you view data in graphical format, the *Normalize Input* and *Normalize Output* options are automatically disabled. In addition, these two options are automatically selected when you are viewing single variable or multivariate linear regression data. For correlation data, these two options are not used.

© 2023 Cadence Design Systems, Inc.

Changing the Data View Type

A view type specifies the data that you want to view in inputs and outputs. You can specify the data view type by using the following list at top right of the tab.



The available view types and their descriptions are listed in the following table.

View Type	Description
View all specs at each corner	View detailed results for all corners and specs. This is the default view.
View average results for all specs at each corner	View average results of all specs at each corner.
View average results for all corners at each spec	View average results at all corners for each spec.

You can plot the average results displayed by the view types listed in the table given above.

See also:

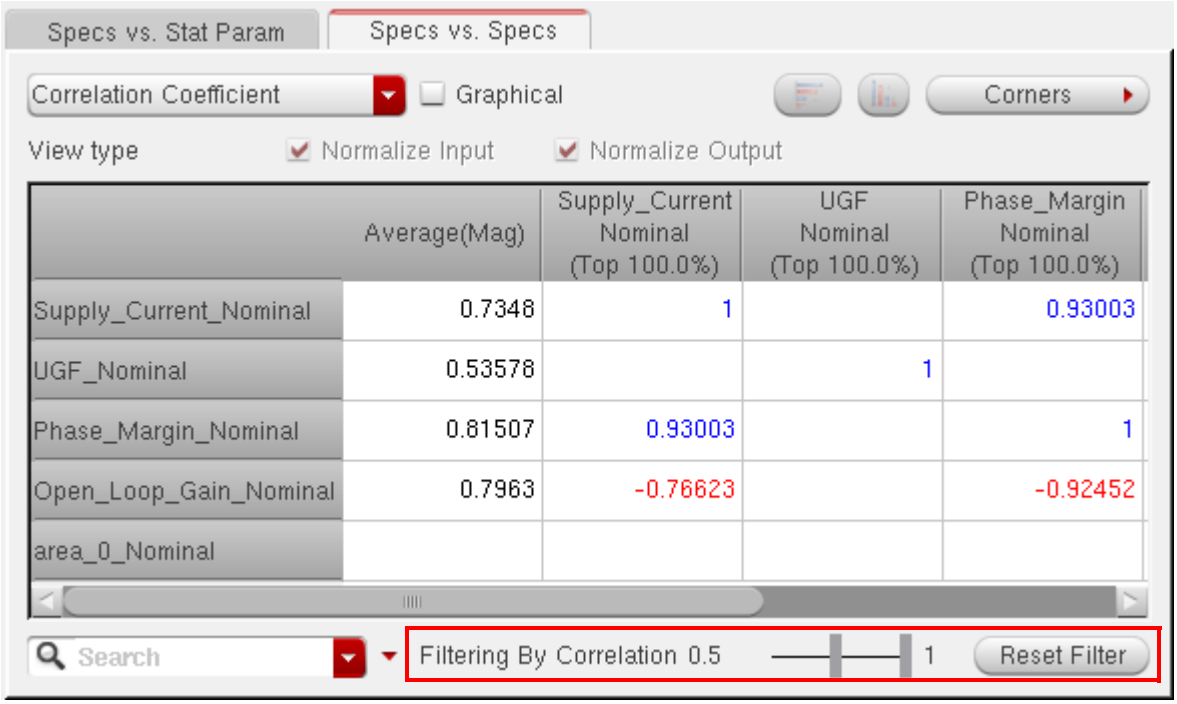
- Hiding Corner Data on page 1549
- Hiding Specification Data on page 1549
- Hiding or Showing the Detailed Results for Corners on page 1549
- Hiding or Showing the Detailed Results for Specifications on page 1550
- Hiding or Showing Statistical Parameters on page 1550
- Showing All Hidden Data on page 1550

© 2023 Cadence Design Systems, Inc.

Filtering Data Based on Correlation Coefficient

With the *Filter by Correlation* slider bars, you can choose to filter out relationships that are in a range of two given correlation values. By default, all the results in the range of `0` and `1` are displayed.

You can move the slider bars to specify the correlation filter range. For example, if you move the slider bars as shown below, the tool displays data with correlation coefficient values between 0.5 and 1.0.



Filtering Data Based on Columns

To find out the most or least sensitive parameters, you can apply a filter on a column. This will filter out parameter rows based on the value of the column data. When the correlation table is displayed, the filter is applied to the correlation data. When the regression table is displayed, the filter is applied to the regression data.

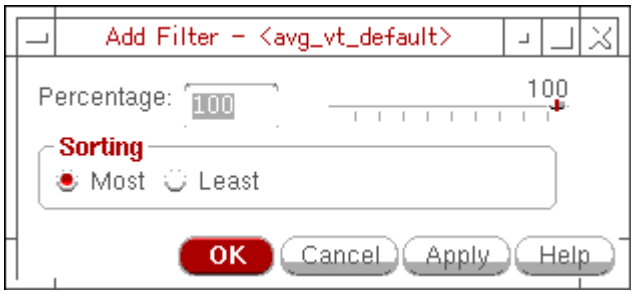
To filter data based on a column:

1. Select a column based on which you want to filter the data.
2. Right-click and choose *Add Filter*.

   The Add Filter form is displayed.

© 2023 Cadence Design Systems, Inc.

Figure 25-8 Add Filter Form

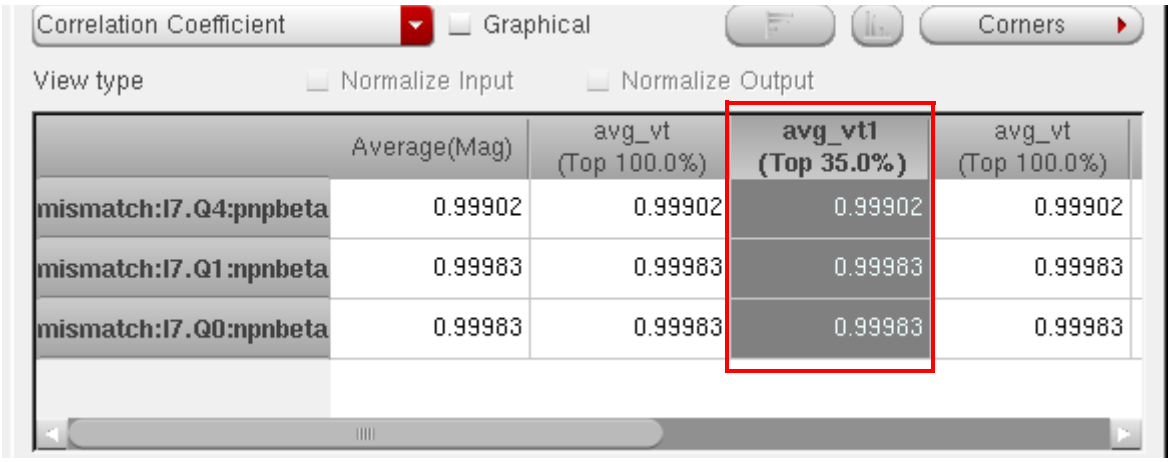


3. Specify a percentage value.

   Alternatively, use the percentage slider to specify a percentage of parameters to be filtered.

4. In the *Sorting* section, select any one of the two options, *Most* or *Least*, to display a specific percentage of top or bottom rows according to the sensitivity of parameters.

   For example, if you specify `35` as the percentage value and `Most` as the sorting criteria, the tool will display the top 35% of rows that show strong correlation, as shown in the following figure.



	Average(Mag)	avg_vt (Top 100.0%)	**avg_vt1 (Top 35.0%)**	avg_vt (Top 100.0%)
mismatch:I7.Q4:pnpbeta	0.99902	0.99902	0.99902	0.99902
mismatch:I7.Q1:npnbeta	0.99983	0.99983	0.99983	0.99983
mismatch:I7.Q0:npnbeta	0.99983	0.99983	0.99983	0.99983

Note: You can apply separate filters on multiple columns. In that case, the tool filters data from the entire row set for each column condition separately and then displays the row intersection (rows that are common in all resulting row sets).

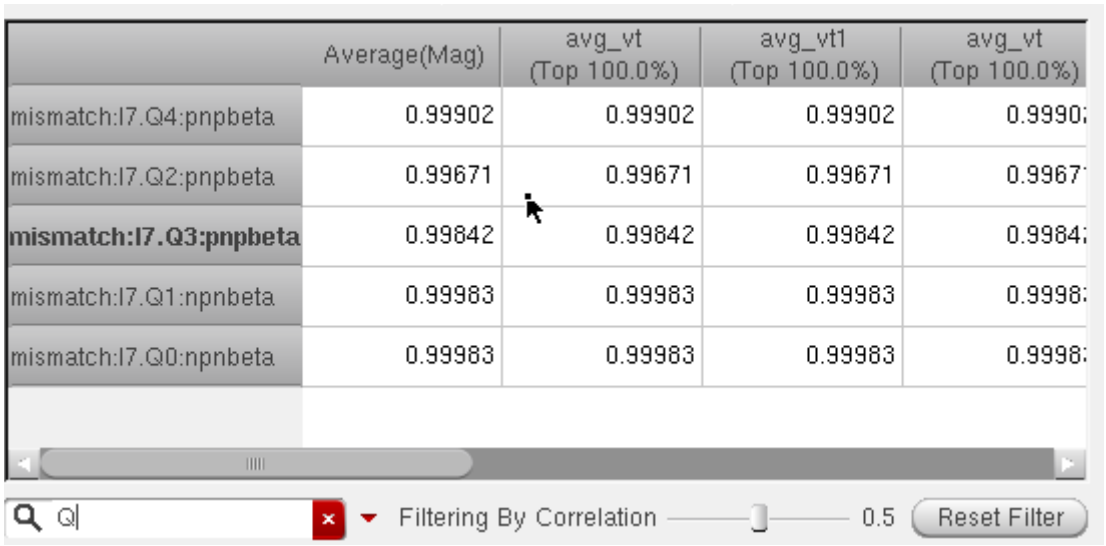
To remove the filters on columns:

- Right-click in a column and choose *Remove Filter*.
- Click *Reset Filter* on the Sensitivity Analysis window. This removes all filters applied on the form.

© 2023 Cadence Design Systems, Inc.

Filtering Data Based on Search Criteria

The *Search* field allows you to filter the results in the Sensitivity Analysis window to view only the results for specific parameters. For example, if you type a string Q in the *Search* field, the form updates automatically to show only the results for the parameters whose names start with the string Q, as shown in the following figure.

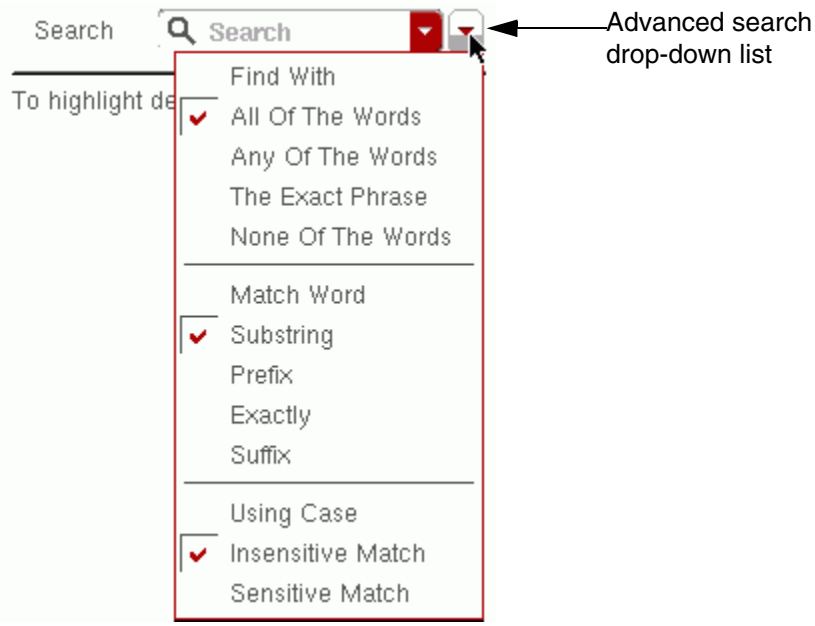


To clear the search, click the button in the *Search* field.

Previous filters used in the current session are accessible from the drop-down list in the Search field.

© 2023 Cadence Design Systems, Inc.

The *Advanced* search drop-down list next to the *Search* field provides commonly used search options to refine your search results.



For more information on the *Search* field, see The Search Toolbar in the *Virtuoso Schematic Editor XL User Guide*.

Resetting Filters

To reset all the filters applied on sensitivity results, click *Reset Filter* next to the *Filter by Correlation* slider bar.

The tool removes all the applied filters and displays all the data for that view.

Sorting Data

To sort data in a row or a column in the Sensitivity Analysis window:

- Right-click a row or column heading and choose any of the following commands, as required:
  - *Sort by value*: This command sorts all the data based on the value in each cell.
  - *Sort by absolute value*: This command sorts all the data based on the absolute value in each cell in the selected column or row.

© 2023 Cadence Design Systems, Inc.

Data is always sorted based on normalized sensitivity regression coefficients.

Hiding Corner Data

You can do the following to hide corner data:

- To hide the data for a specific corner, right-click the column for the corner and choose *Hide Corners*.
- To hide the data for all corners other than a specific corner, right-click the column for the corner you don't want to hide and choose *Hide Other Corners*.

To show all hidden data, right-click and choose *Show All*.

Hiding Specification Data

You can do the following to hide specification data:

- To hide the data for a specification, right-click the column for the specification and choose *Hide Specs*.
- To hide the data for all specifications other than a specific specification, right-click the column for the specification you don't want to hide and choose *Hide Other Specs*.

To show all hidden data, right-click and choose *Show All*.

Hiding or Showing the Detailed Results for Corners

When you are viewing average results for all specs at each corner, you can do the following to hide the detailed results for corners:

- To hide the detailed results for a corner, right-click the column for the corner and choose *Collapse Selected Corners*.
- To hide the detailed results for all corners, right-click and choose *Collapse All Corners*.

You can do the following to show the detailed results for corners in the Corner View:

- To show the detailed results for a corner, right-click the column for the corner and choose *Expand Selected Corners*.
- To show the detailed results for all corners, right-click and choose *Expand All Corners*.

© 2023 Cadence Design Systems, Inc.

Hiding or Showing the Detailed Results for Specifications

When you are viewing average results for all corners at each spec, you can do the following to hide the detailed results for specifications:

- To hide the detailed results for a specification, right-click the column for the specification and choose *Collapse Selected Specs*.
- To hide the detailed results for all specifications, right-click and choose *Collapse All Specs*.

You can do the following to show the detailed results for specifications in the Spec view:

- To show the detailed results for a specification, right-click the column for the specification and choose *Expand Selected Specs*.
- To show the detailed results for all specifications, right-click and choose *Expand All Specs*.

Hiding or Showing Statistical Parameters

Statistical parameters are displayed in the Specs vs. Parameter tab and the DC Ops vs. Parameter tab if the *Enable variation of Statistical Parameters* check box is selected in the Sensitivity form.

To hide the statistical parameters, right-click and choose *Hide Statistical Parameters*.

To show all hidden data, right-click and choose *Show All*.

Showing All Hidden Data

To show all hidden data, right-click and choose *Show All*.

Highlighting Associated Devices in the Schematic

When you right-click the parameters in the Sensitivity Analysis window and choose *Highlight on Schematic*, the associated device(s) in the schematic are highlighted.

Note: For Monte Carlo runs, you can highlight only the devices associated with mismatch parameters.

© 2023 Cadence Design Systems, Inc.

Viewing Results in Multiple Sensitivity Analysis Windows

If you want to view the results of Sensitivity Analysis in different view modes and with different filtering conditions at the same time, you can open the same results in multiple windows. Next, you can apply different view settings and filters in different windows and compare the results.

To open the results in multiple windows, perform the following steps:

1. In the *Data View*, right-click a history run and choose *Sensitivity Results*.

   A new Sensitivity Analysis window is opened with the results of the selected history displayed in it.

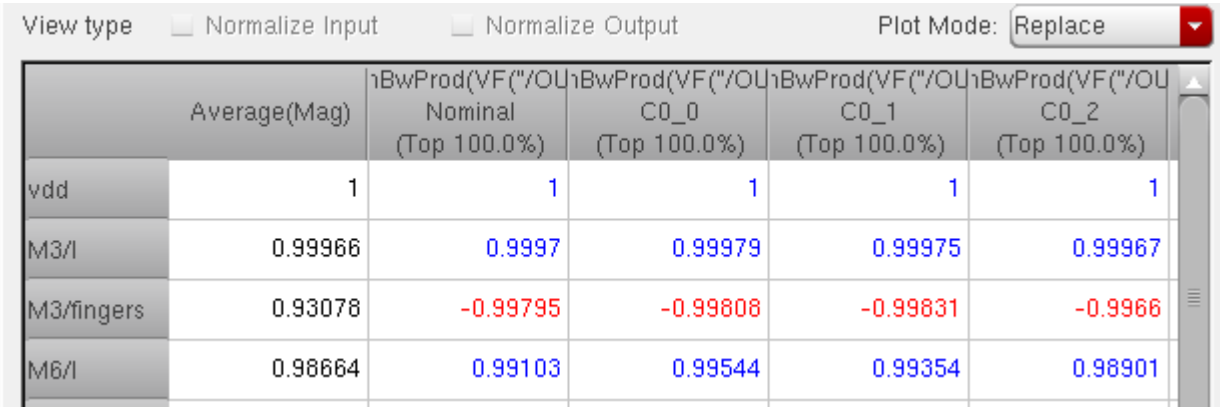
2. If required, customize the view of results as per your requirements.

3. In the Sensitivity Analysis window, choose *File — New Window*.

   A new Sensitivity Analysis window is opened and the same results are displayed in it. The data view settings in the new window are same as those in the first window. However, you can customize these, as required. Now, the same results are visible in two different windows with different view settings.

Changing Number Format

By default, in numeral format, the number of significant figures used to display data is not fixed. In the results shown below, the data appears in different significant digits.



	Average(Mag)	ıBwProd(VF("/OU Nominal (Top 100.0%)	ıBwProd(VF("/OU C0_0 (Top 100.0%)	ıBwProd(VF("/OU C0_1 (Top 100.0%)	ıBwProd(VF("/OU C0_2 (Top 100.0%)
vdd	1	1	1	1	1
M3/l	0.99966	0.9997	0.99979	0.99975	0.99967
M3/fingers	0.93078	-0.99795	-0.99808	-0.99831	-0.9966
M6/l	0.98664	0.99103	0.99544	0.99354	0.98901

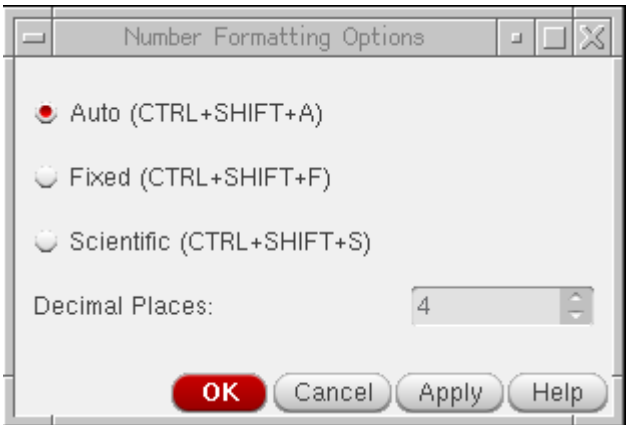
If required, you can set a fixed number of significant digits in which you want to display the numeral data. For this, you can use the **Number Formatting Options** form.

To change the number format:

1. In the Sensitivity Analysis window, choose *Format — Numbers*.

© 2023 Cadence Design Systems, Inc.

The **Number Formatting Options** form is displayed. The default format is *Auto*.



2. Select *Fixed* to display the numbers in a fixed point format.

   Note: Similarly, you can select *Scientific* to display the numbers in scientific format.

3. In the *Decimal Places* field, specify the number of decimal places that you want to show for each number in the results.

   Note: Decimal places can be set only upto a maximum of six digits.

4. Click *Apply* to apply the changed format.

In the results shown below, the data appears in the fixed format with four decimal places.

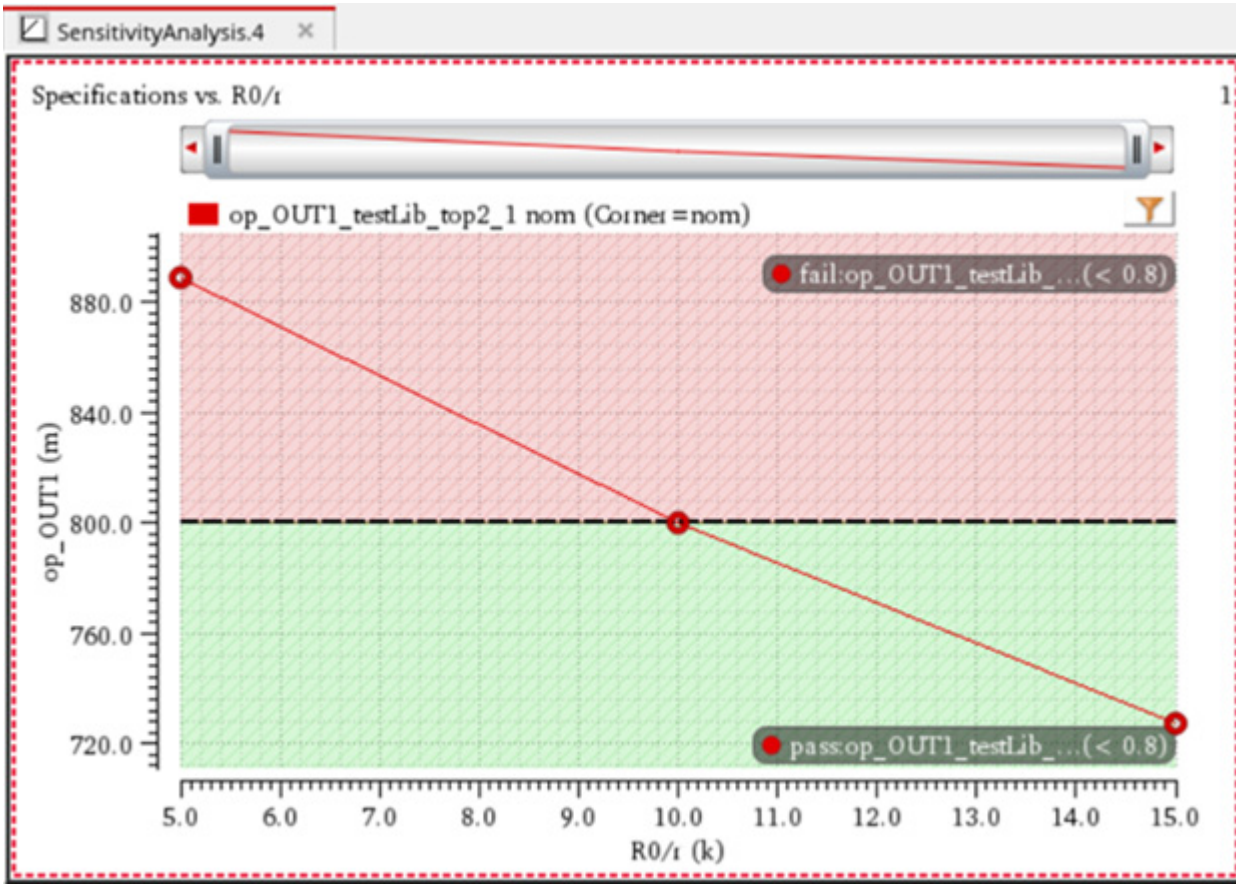
	Average(Mag)	ıBwProd(VF("/OU Nominal (Top 100.0%)	ıBwProd(VF("/OU C0_0 (Top 100.0%)	ıBwProd(VF("/OU C0_1 (Top 100.0%)	ıBwProd(VF("/OU C0_2 (Top 100.0%)
vdd	1.0000	1.0000	1.0000	1.0000	1.0000
M3/l	0.9997	0.9997	0.9998	0.9998	0.9997
M3/fingers	0.9308	-0.9980	-0.9981	-0.9983	-0.9966
M6/l	0.9866	0.9910	0.9954	0.9935	0.9890
M6/fw	0.9992	-0.9996	-0.9995	-0.9996	-0.9996

© 2023 Cadence Design Systems, Inc.

Plotting Sensitivity Analysis Results

The sensitivity analysis results can be plotted in the Virtuoso Visualization and Analysis XL window. The data can be plotted in three ways:

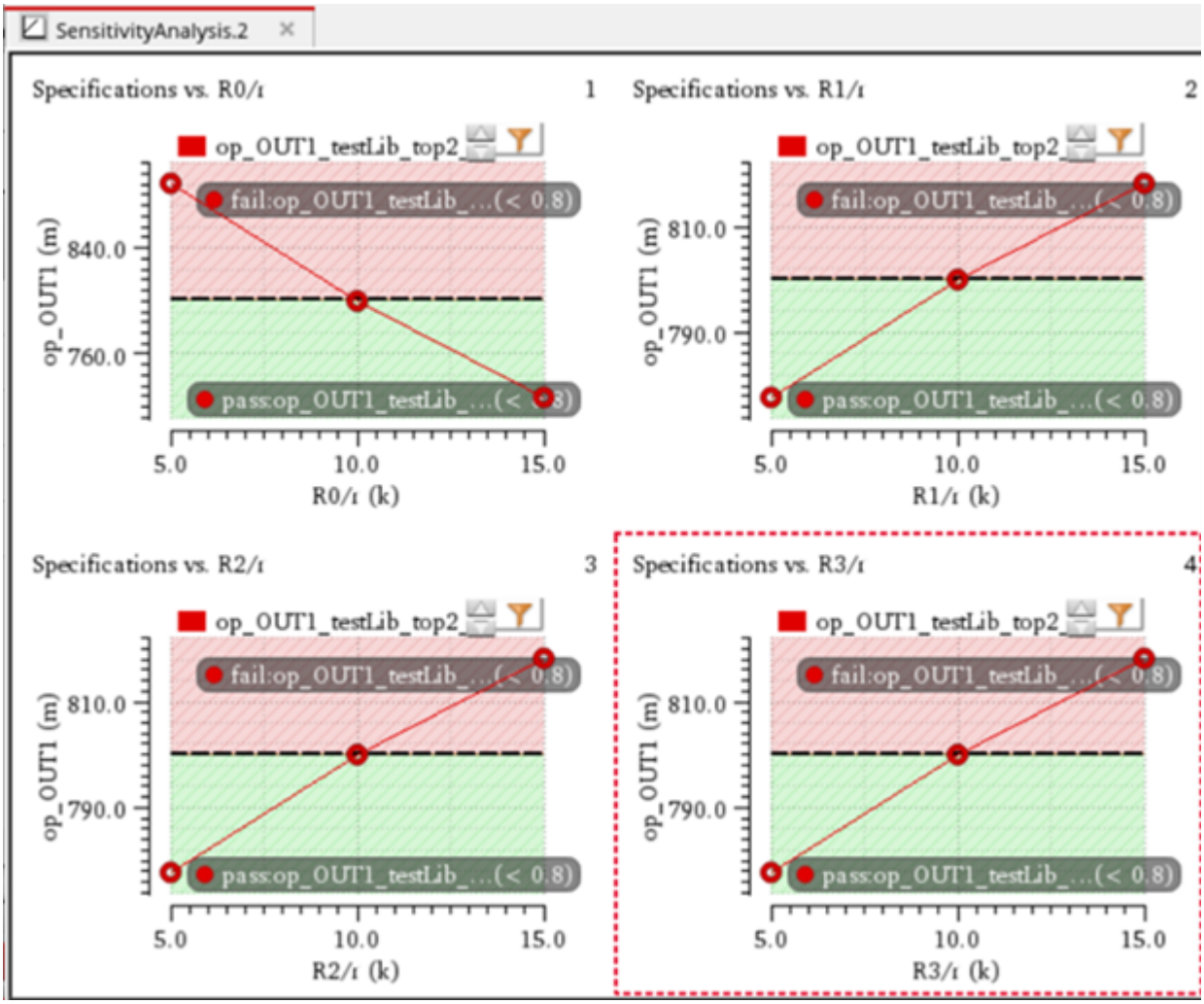
- Double-click any data cell in the Sensitivity Analysis results window. The correlation and regression data in the plotted graph exhibits the sensitivity of the selected specification to the corresponding parameter, as shown below.



- Select two or more parameter cells for a single spec column, right-click, and choose the *Plot Across Selected Parameters* command. This displays the sensitivity of multiple

© 2023 Cadence Design Systems, Inc.

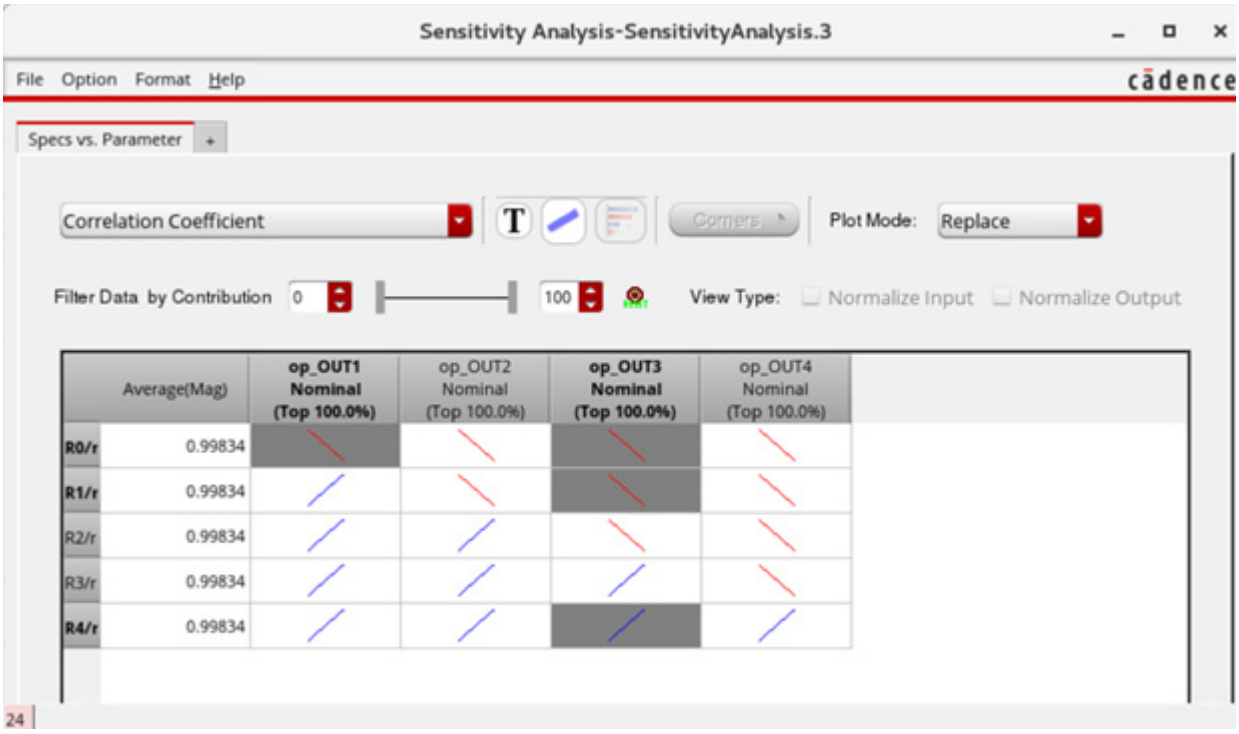
selected parameters to the specification represented by the corresponding column, as shown in the following figure:



➡ Select two or more data cells in different spec columns, right-click, and choose the *Plot Across Parameters* command to display the sensitivity of the selected parameters to the selected specifications.

© 2023 Cadence Design Systems, Inc.

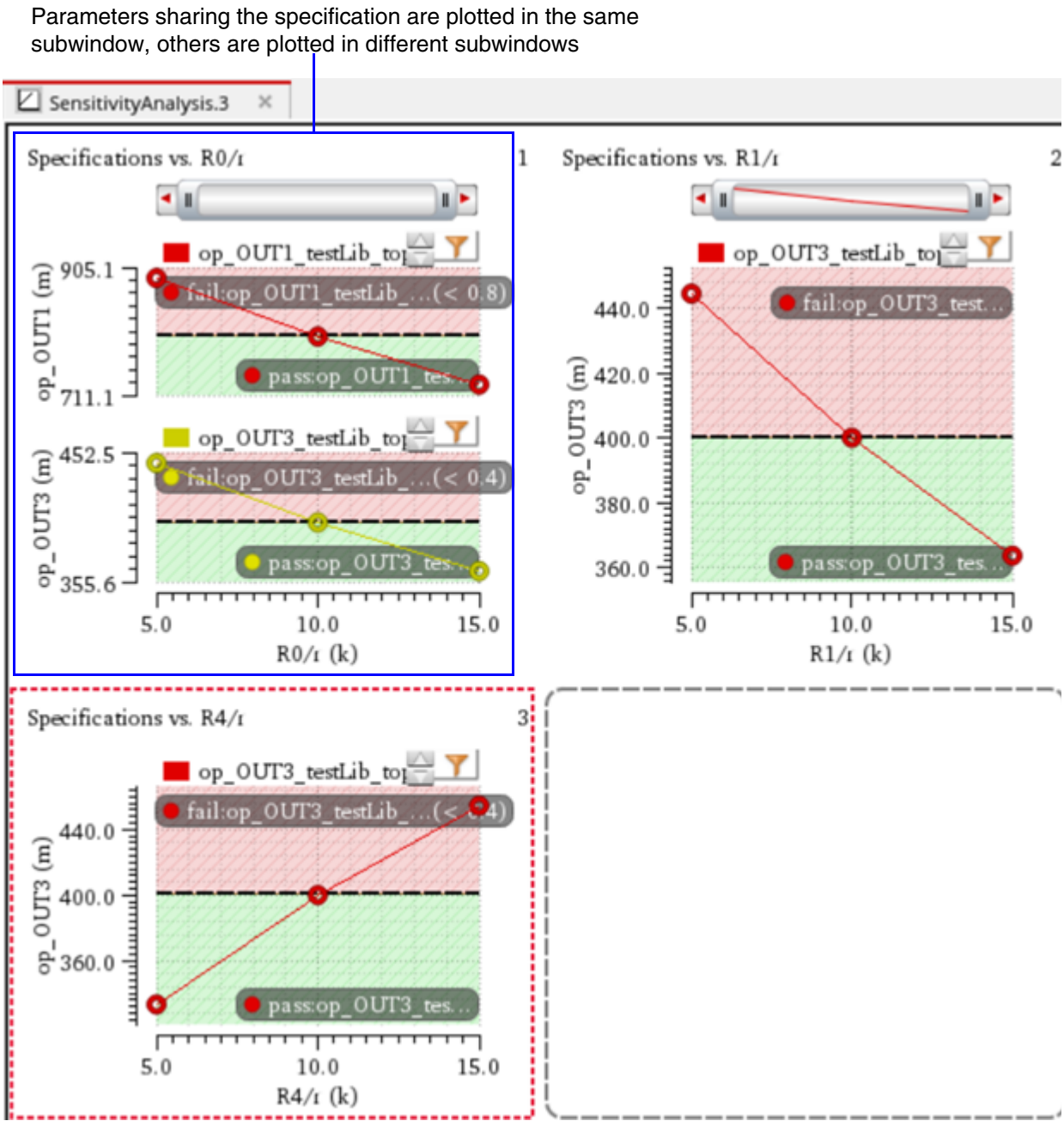
For example, select cells for different specs and parameters, as shown in the following figure:



Now, right-click and choose *Plot Across Parameters.*

© 2023 Cadence Design Systems, Inc.

In this case, the result for each parameter and spec combination is plotted in a separate strip, as shown in the figure below.



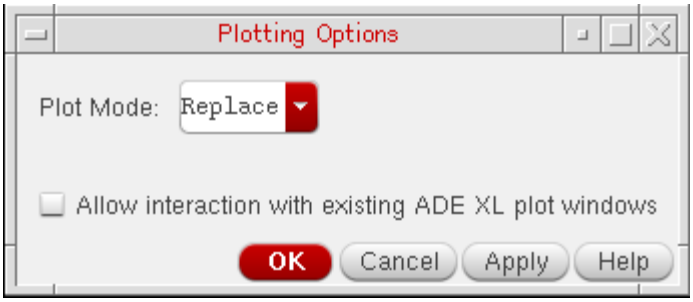
If the Virtuoso Visualization and Analysis XL window is already open for the selected history, by default, a new plot always replaces the previous plot. However, you can specify different plotting modes depending on your data analysis requirements. For more details, refer to the following sections:

© 2023 Cadence Design Systems, Inc.

- Setting the Plotting Mode
- Displaying the ADE XL and Sensitivity Analysis Results in the Same Graph
- Plotting Average Sensitivity Analysis Results
- Displaying Spec Markers in Sensitivity Analysis Plots

Setting the Plotting Mode

You can set the plotting mode for the sensitivity analysis results graph by using the Plotting Options form.



In the Sensitivity Analysis Results window, choose *Options – Plotting*. The Plotting Options form is displayed. In this form, the *Plot Mode* field specifies the plotting mode of the graph. By default, the default plotting mode is set to `Replace`.

You can select any one of the following plotting modes:

- Replace
- Append
- New SubWin
- New Win

Tip

You can also set the plot mode by using the *Plot Mode* drop-down list on the Sensitivity Analysis results window.

Replace Plot Mode

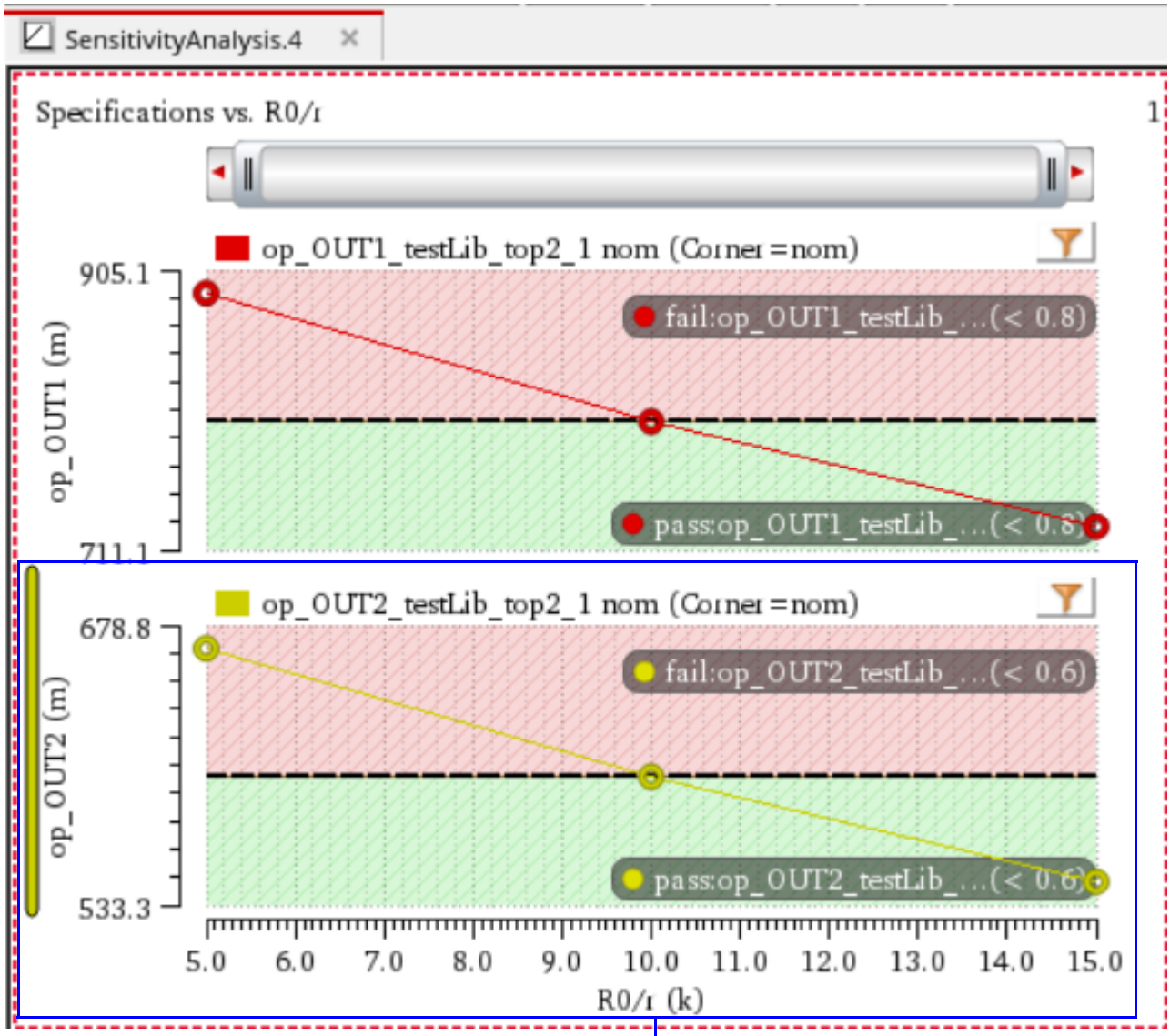
In this mode, the new plot always replaces an existing graph, if any. Otherwise, a new window appears that displays the graph.

© 2023 Cadence Design Systems, Inc.

Append Plot Mode

In this mode, if a graph is already open for the selected history, the new plot is appended to the existing graph. Otherwise, the new plot is displayed in a new window.

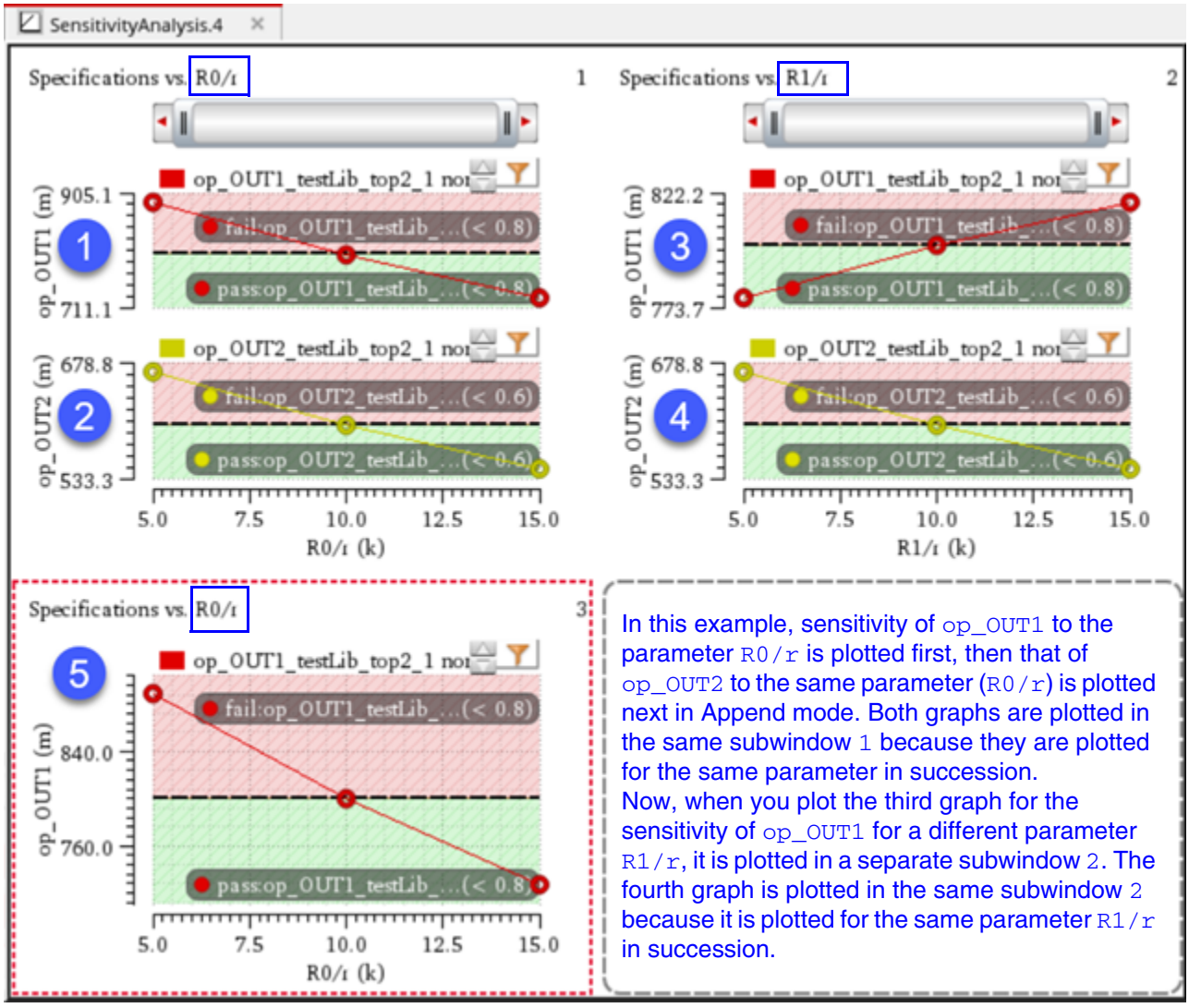
Consider an example. A graph is already open for the selected history and displays the sensitivity of `op_OUT1` to the parameter `R0/r`. Next, if you set the plot mode to *Append* and plot the sensitivity of `op_OUT2` to the parameter `R0/r`, the new plot is appended to the existing graph, as shown below.



New plot is appended to the existing graph in a new strip

© 2023 Cadence Design Systems, Inc.

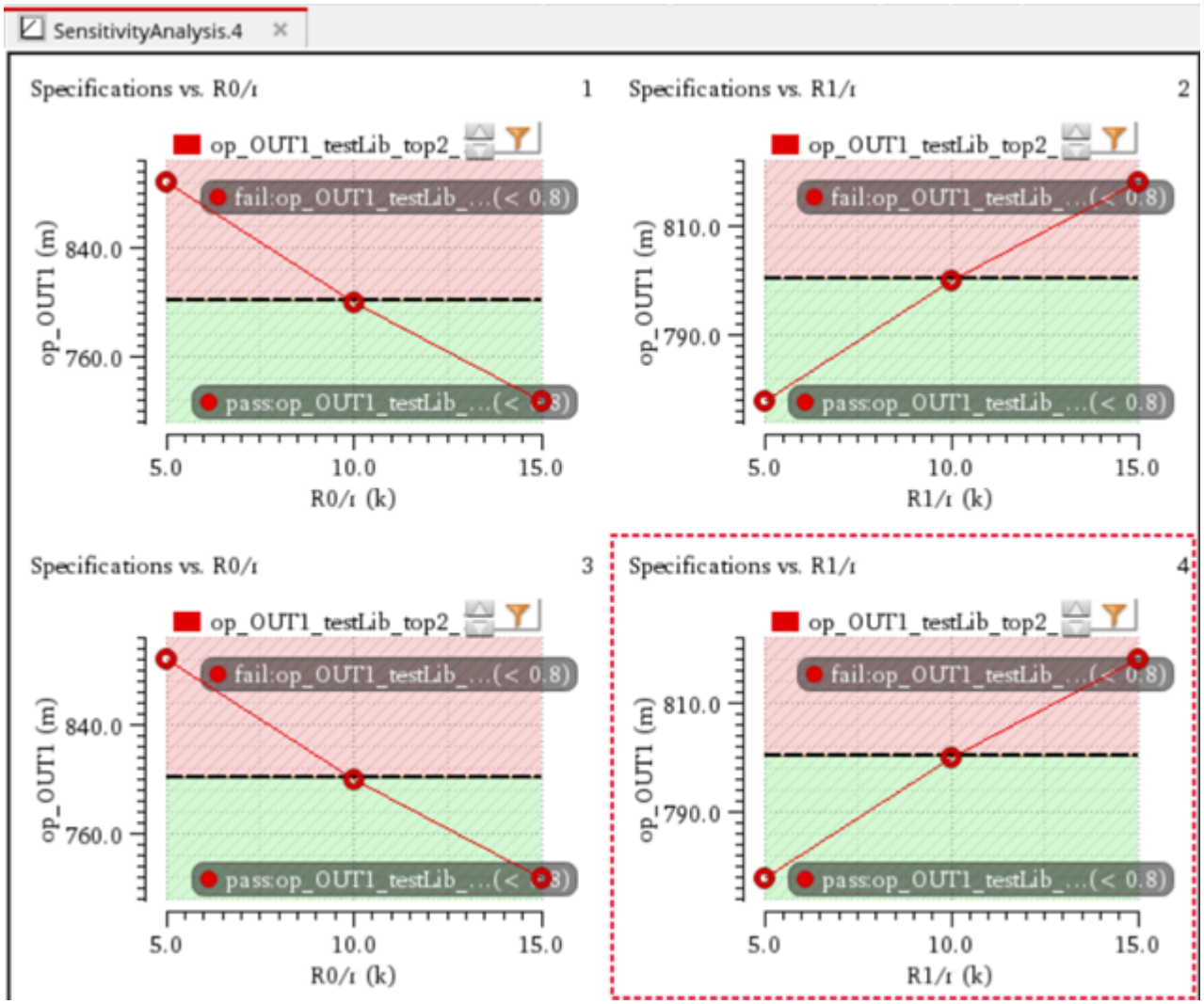
Note: For the plot to append to the existing graph, it is important that you plot another graph for the same parameter in succession.



The tool does not retain plots with different x axes on the same graph. In the example shown above, if you plot the results for a different parameter, the graph will be plotted in a new

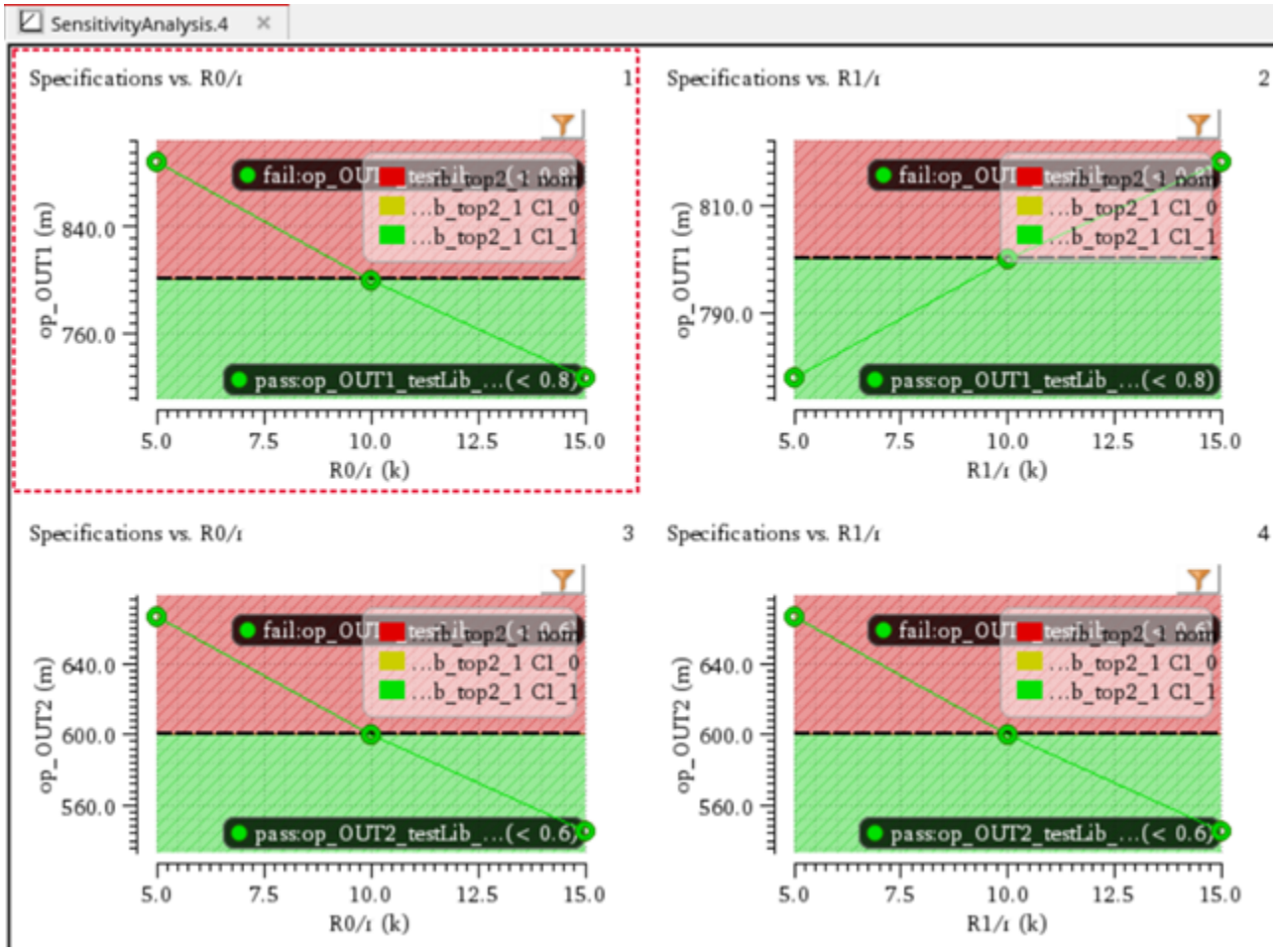
© 2023 Cadence Design Systems, Inc.

subwindow. If you select multiple results for different parameters and choose *Plot Across Parameters*, the graphs are plotted in one subwindow per graph.



© 2023 Cadence Design Systems, Inc.

If the view type is '*View average results for all corners at each spec*', graphs are overlaid by corners, one spec per subwindow.

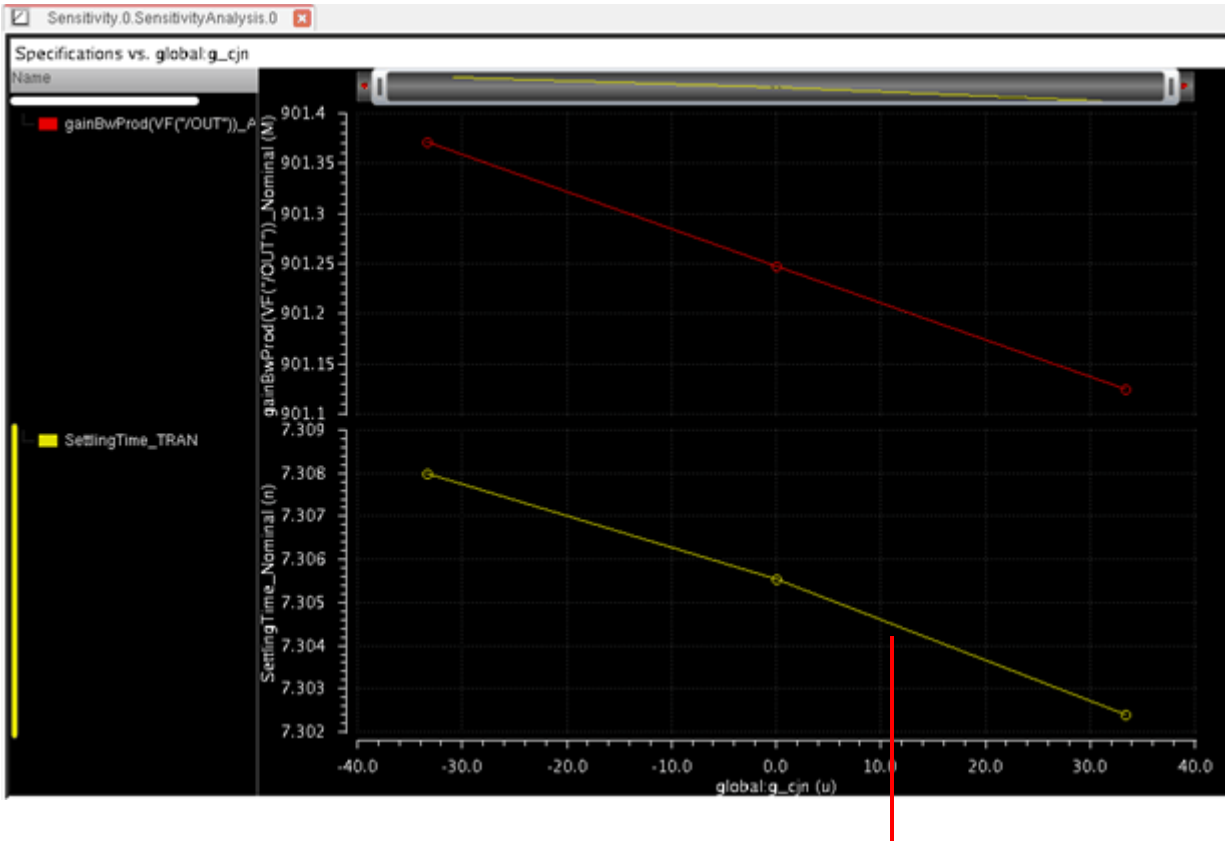


Exceptions to the Append Plot Mode

When the Plot Mode is set to Append, in the following two cases, the new plot is displayed in a new strip instead of being appended to the existing graph:

- If the new plot is for the same parameter but for a different specification as that of the graph plotted earlier, the new plot is displayed in a new strip instead of being appended to the existing graph because of common x-axis parameter, as shown below.

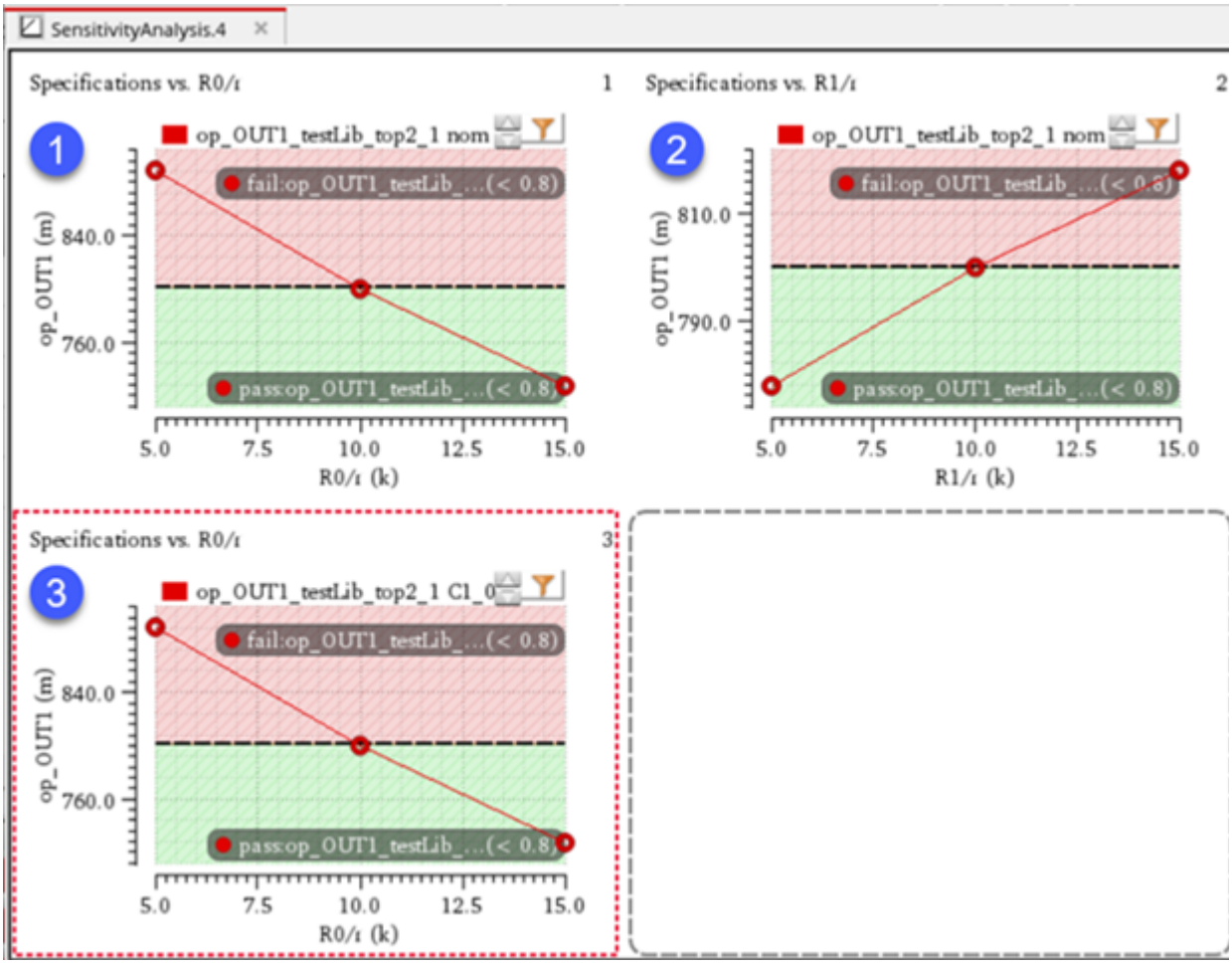
© 2023 Cadence Design Systems, Inc.



- If you plot multiple parameters for the same specifications, all graphs are plotted in different subwindows, and next, if you plot the results for new parameters, a new subwindow is created for each additional graph because the x-axes refer to different

© 2023 Cadence Design Systems, Inc.

parameters. Only when different specifications are plotted for the same parameter successively, they are appended to strips.



Note: When plotting is done across different parameters, the data is normalized by using the following equations:

- For design variables and statistical variables:

  ```
  Xnorm = 2*(X-Xmin)/(Xmax-Xmin)
  ```

- For Monte Carlo analysis:

  ```
  Xnorm = (X-Xmean)/Xstd
  ```

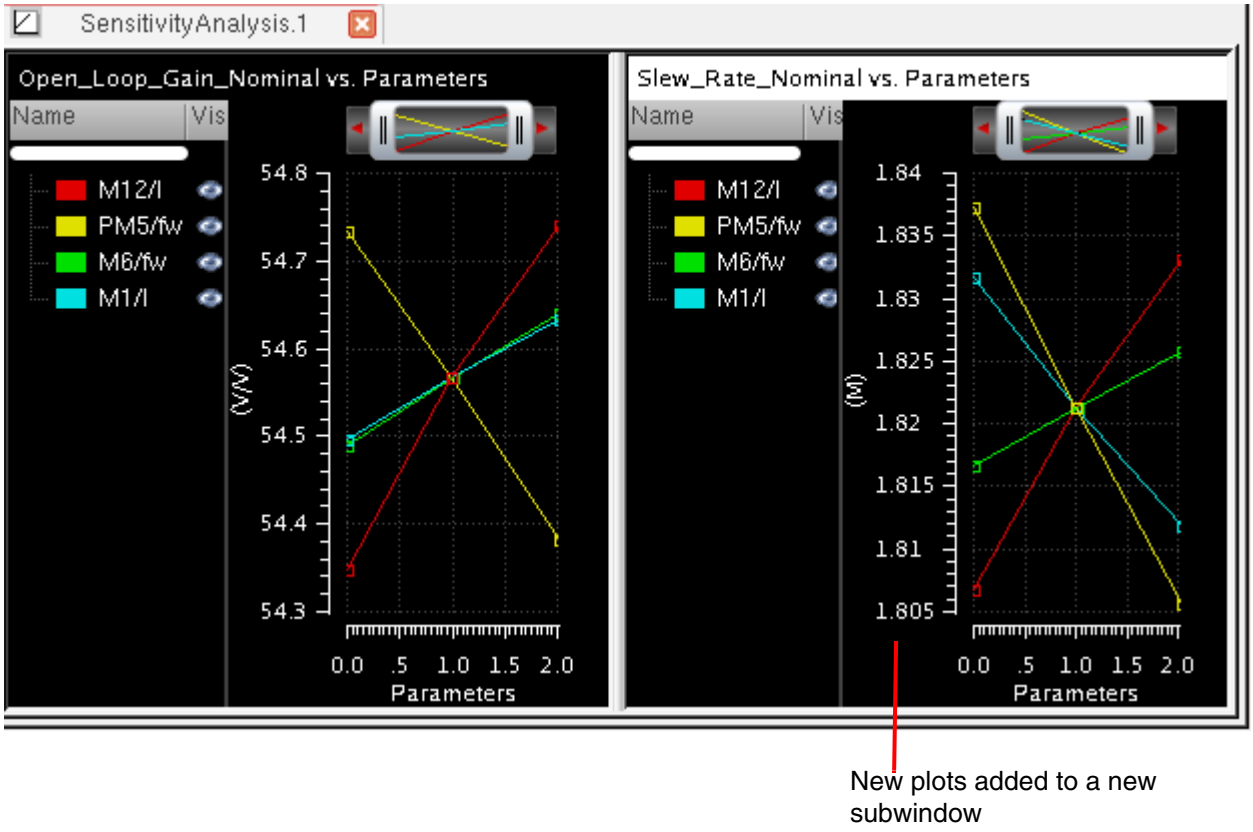
New SubWin Plot Mode

In this plot mode, if a graph is already open for the selected history, the new plot is added to a new subwindow in the existing graph.

© 2023 Cadence Design Systems, Inc.

Consider an example. You have plotted the sensitivity data for the `Open_Loop_Gain` specification to parameters `M12/I`, `PM5/fw`, `M6/fw`, and `M1/I`.

Next, you want to plot and compare the sensitivity data for `Slew_Rate` to the same set of parameters. If you set the plot mode to `New SubWin` and plot data for these parameters, the graph appears as shown below.



New Win Plot Mode

In this plot mode, for every new plot, a window is added to the existing Virtuoso Visualization and Analysis XL window, as shown below.

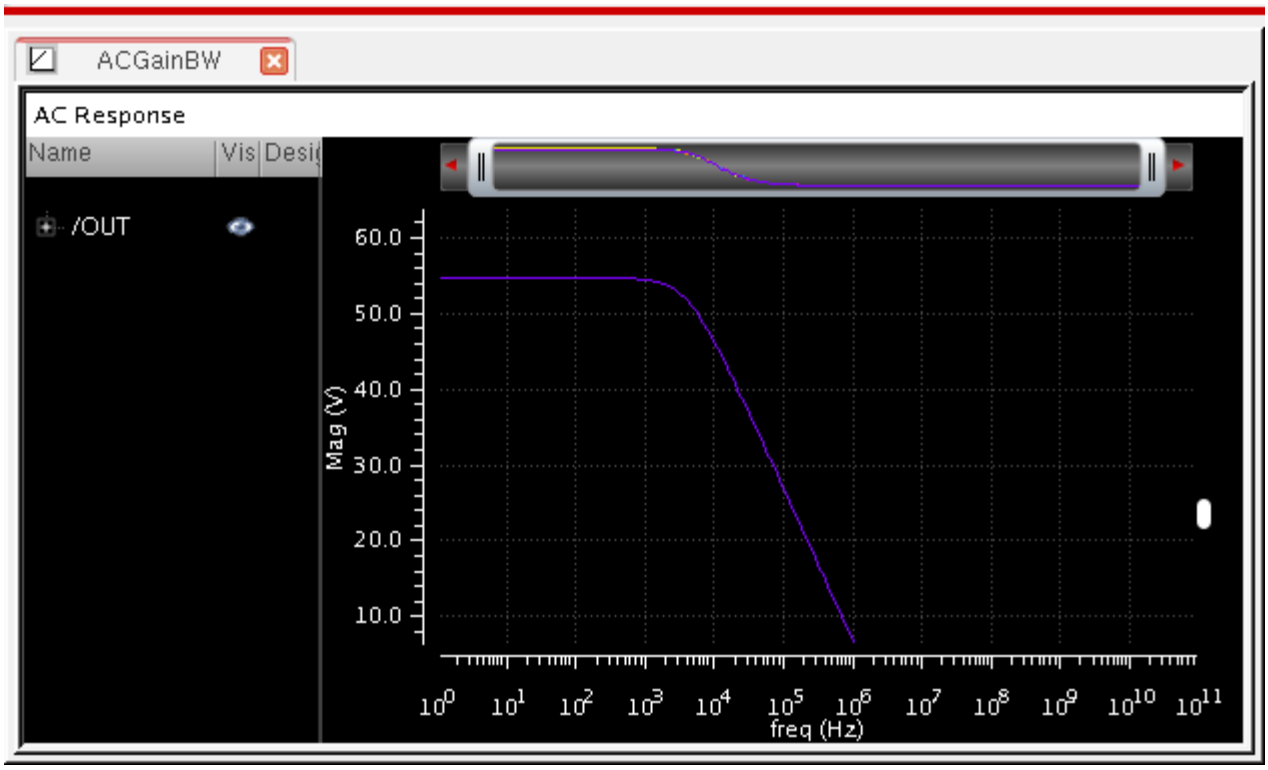
© 2023 Cadence Design Systems, Inc.

Displaying the ADE XL and Sensitivity Analysis Results in the Same Graph

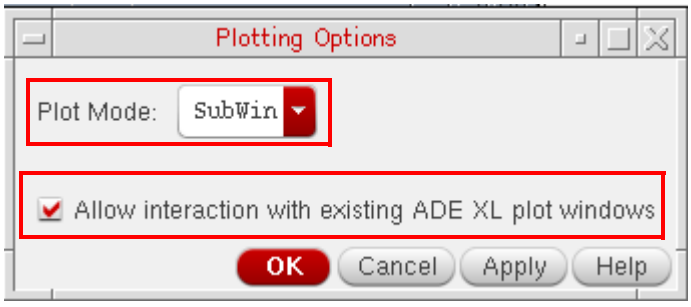
If you have already plotted results from the ADE XL window, you can choose to plot the sensitivity analysis results in the same graph. To do this, select the *Allow interaction with existing ADE XL plot windows* check box in the Plotting Options form.

For example, when you plot an output, `ACGainBW`, from the ADE XL results tab, the graph appears as shown below.

© 2023 Cadence Design Systems, Inc.

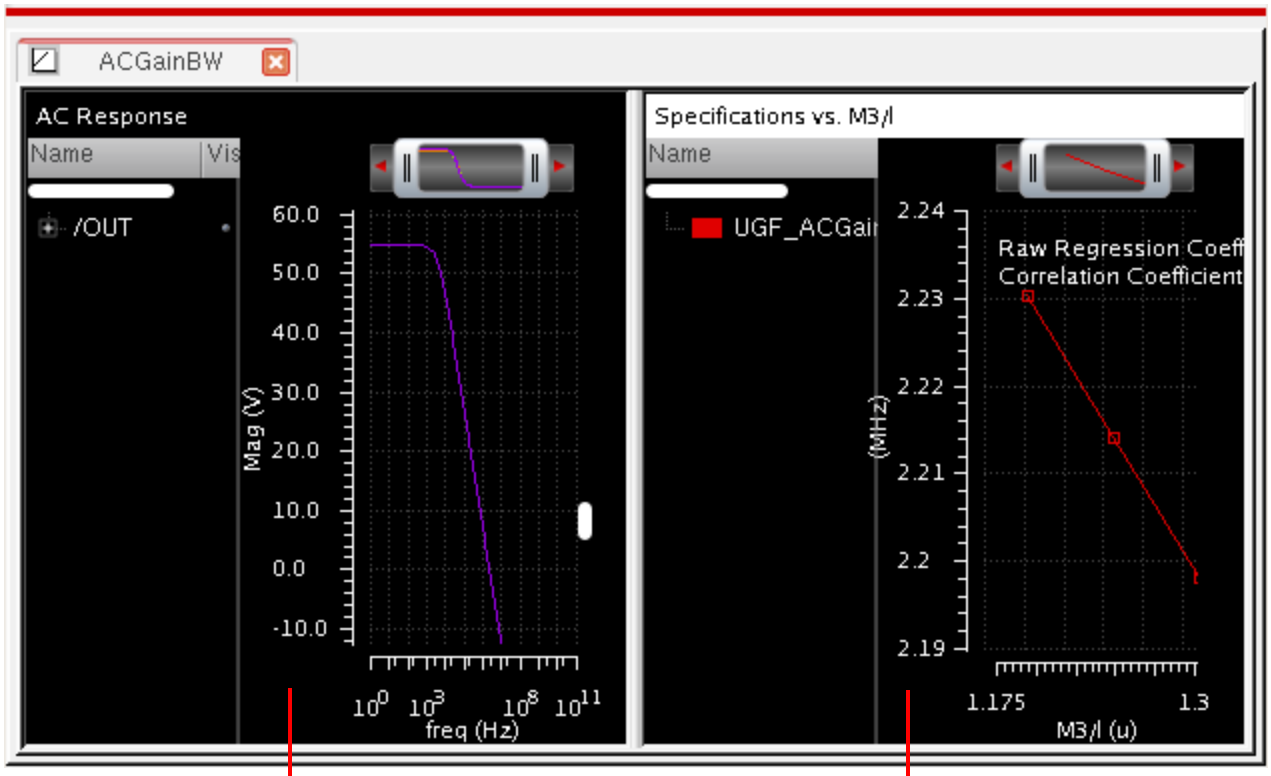


Now, if you want to plot the sensitivity results in a new subwindow of this graph to show sensitivity of `UGF_ACGainBW` to the variable `M3/I`, set the plotting options as shown below:



Now, if you plot the sensitivity results, a new subwindow is added as shown below.

© 2023 Cadence Design Systems, Inc.



Graph plotted from the ADE XL Results tab

New subwindow added to the same graph to display sensitivity results

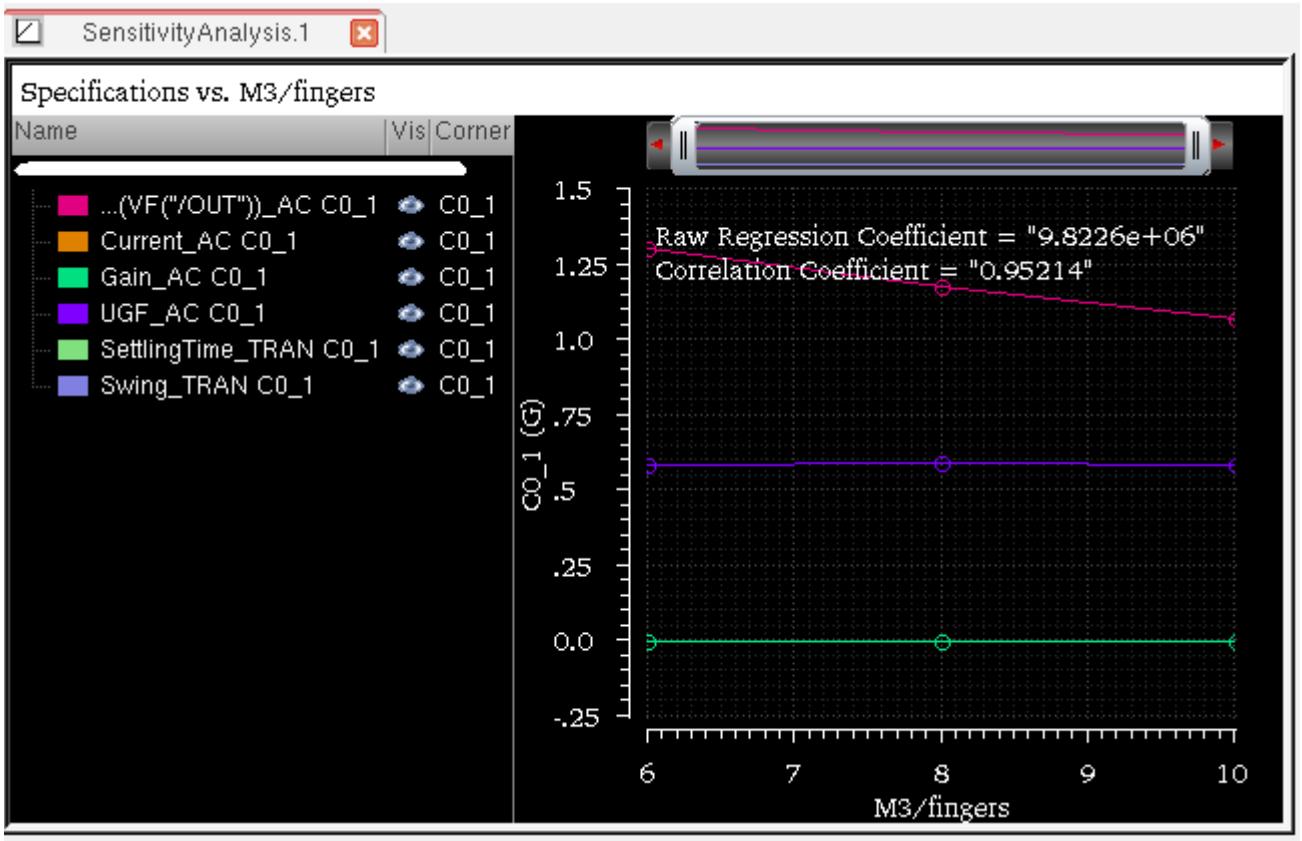
Note: The *Allow interaction with existing ADE XL plot windows* check box is not considered when the plot mode is set as *Append*. If this check box is selected and the *Plot Mode* is set to *Append*, the plots for the sensitivity data are not appended to the existing graph. Instead, the sensitivity data is plotted in a new window.

Plotting Average Sensitivity Analysis Results

If you change the default Sensitivity Analysis results view and display average results of all the corners for each spec or results of all specs for each corner, you can double-click in a cell to show all the plots.

For example, if you display average results of all corners at each spec, when you double-click in a cell, the plots are displayed as shown below.

© 2023 Cadence Design Systems, Inc.



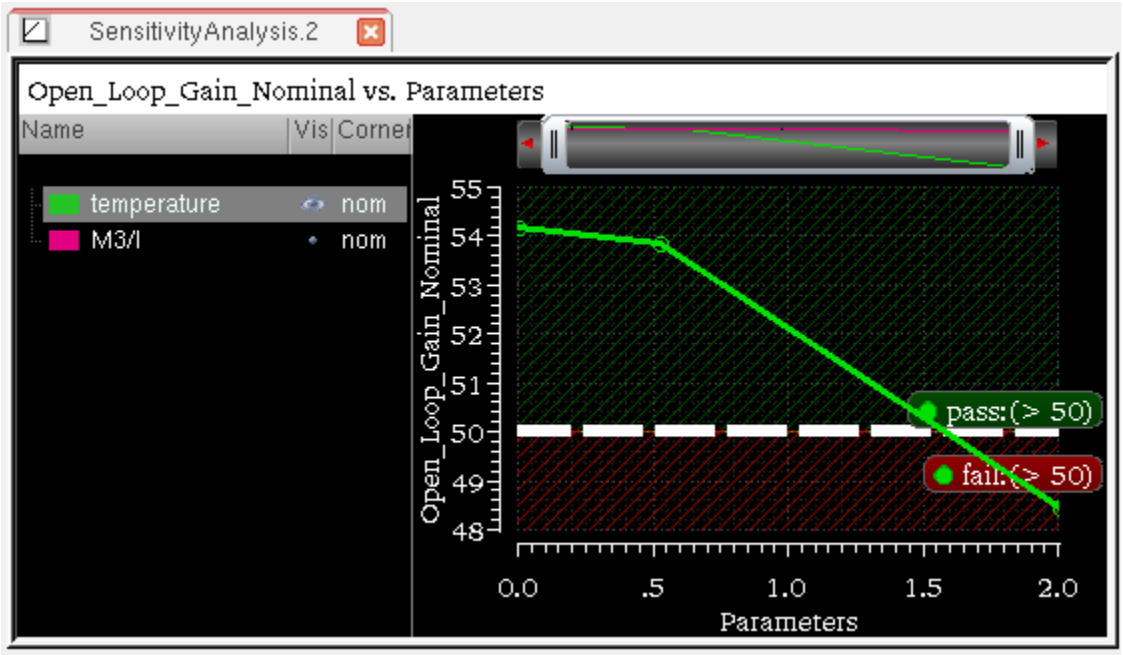
Similarly, you can plot average results of all corners at each spec.

Displaying Spec Markers in Sensitivity Analysis Plots

Similar to the ADE XL graphs, you can display spec markers in the graphs plotted for Sensitivity Analysis results. For this, on the Results tab of ADE XL, choose *Options — Plotting/Printing*. On the **ADE XL Plotting/Printing Options** form, select *Spec Markers.* When this option is selected, the graphs displayed from the Results tab of the Output pane show spec markers, as shown below.

© 2023 Cadence Design Systems, Inc.

The specs vs. parameter graphs displayed from the Sensitivity Analysis window also show spec markers, as shown below.



© 2023 Cadence Design Systems, Inc.

Saving the Sensitivity Data

You can save the sensitivity data to a text file:

1. Choose *File – Export Results to CSV*, then do one of the following:
   - Choose *Current Tab* to save only the data in the current tab.
   - Choose *All Tabs* to save the data in all the tabs.
2. In the Export Results form, specify a path and file name.
3. Click *OK*.

Highlighting a Device on Schematic

While analyzing data in Sensitivity Analysis table, you can highlight a device on the schematic to view its exact location. This helps in identifying and modifying a device in schematic to improve the variance data, if required.

To highlight a device on schematic:

➡ Right-click in a row and choose *Highlight on Schematic*.

Alternatively, press bindkey `H`.

© 2023 Cadence Design Systems, Inc.

26

Worst Case Corners

A corner is a combination of variables or process models that define a scenario in which you want to measure the performance of your design. For more information about corners, see Simulating Corners.

When you work with large simulation data, defining corner conditions results in hundreds or thousands of possible corner combinations to run. Therefore, running simulations over all corners becomes time consuming process and also requires large computation resources. Therefore, Worst Case Corners helps to reduce the overall project time devoted to simulation and still provide a high level of accuracy.

By identifying Worst Case Corners, the problem is reduced to a subset of worst case conditions. These worst case corners can then be used in verification and design. In addition, running simulations over a reduced set of corners facilitates design changes to be validated quickly.

This chapter covers the following topics to describe how to use the Worst Case Corner analysis in ADE Assembler:

- Prerequisites for Setting Up the Worst Case Analysis on page 1571
- Specifying the Worst Case Corner Analysis Setup on page 1572
- Running Worst Case Corners on page 1583
- Validating Worst Case Corners on page 1584
- Creating Worst Case Corners Automatically on page 1588
- Viewing Worst Case Corners on page 1590

Prerequisites for Setting Up the Worst Case Analysis

Before you create worst case corners, perform the following steps:

1. On the *Run* toolbar, in the *Select a Run Mode* drop-down list, select *Worst Case Corners.*

© 2023 Cadence Design Systems, Inc.

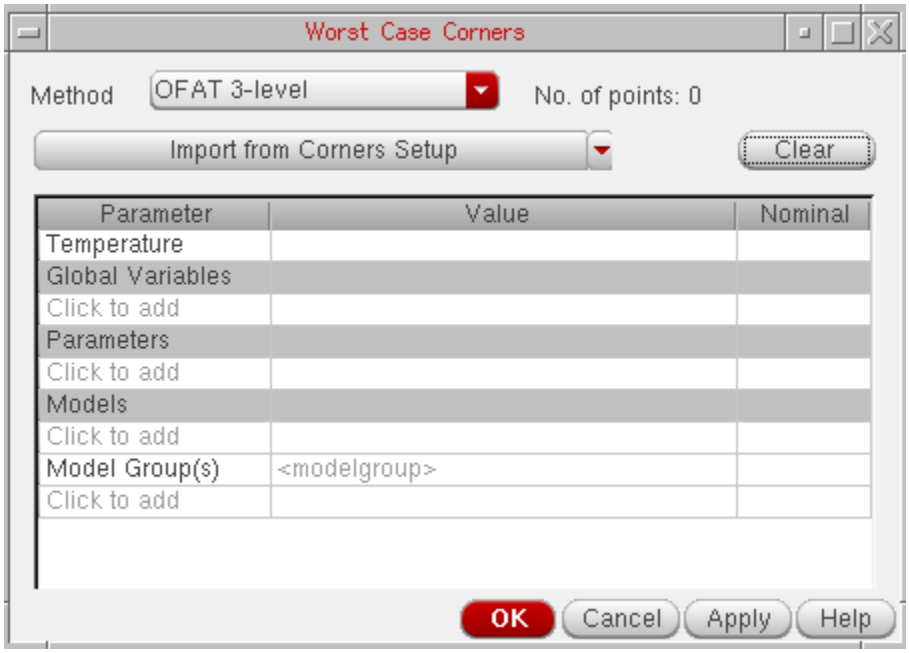
2. If you want to vary variables, ensure that you have specified variables in the Variables and Parameters assistant.
3. On the *Outputs Setup* tab, select the required spec and ensure that for each spec, you specify an objective type and a target value. Worst case corners are created only for the relevant specs.

   Note: If you specify `tolerance (tol)` or `range` objective types for the spec you choose, two worst case corners are created. For all other objective types, one worst case corner is created.

Specifying the Worst Case Corner Analysis Setup

After you select the run mode, click the *Simulation Options* button to specify the worst case corner setup.

The Worst Case Corners form appears as shown in the figure below.



In the Worst Case Corners form, specify the settings that you want to use while running the sensitivity analysis to identify the worst case corners. In this form, you can specify the Temperature, variables, and models that you want to vary and also specify the method by which you want to vary them.

The various form fields are explained in detail in the following sections:

© 2023 Cadence Design Systems, Inc.

- Choosing a Method on page 1573
- Importing Values from the Corners Setup on page 1578
- Specifying Temperature on page 1579
- Specifying Global Variables on page 1580
- Specifying Parameters on page 1580
- Specifying Model and Model Groups on page 1581

Choosing a Method

In the *Method* field, specify the method (algorithm) by which you want to vary the variables. Depending upon the method you specify in this field, the tool, decides how many and which combination of variables are to be simulated.

Following are the five available methods:

- `OFAT 3-level` - In this method, the tool varies each factor for three values. For example, the values of temperature can be specified as `-40`, `27`, `85` with the nominal value equal to `27`. If, instead of three points, you specify a range of values, the tool chooses the following three values of the variable for simulation:
  - minimum value in the range
  - nominal value
  - maximum value in the range

  For example, if the Temperature range is `-40:10:120` (`-40` in steps of `10` degrees to maximum value of `120`) and the nominal value is `27`, the three values used by the tool are: `-40`, `27`, `120`.
- `OFAT Sweep` - In this method, the tool varies each factor for the specified sweep values. You can either specify sweep values as a range in the `min:step:max` syntax or as a set of space-separated values.
- `Central Composite Design` - This method is similar to OFAT 3-level, but more accurate in the case of interactions among the corner variables. In addition to the points simulated by OFAT 3-level, Central Composite Design simulates the max and min values together for every pair of variables. This method does not simulate all model section values and simulates only first, nominal, and last values. Choose a different method if you want to take all model sections into account.

© 2023 Cadence Design Systems, Inc.

- `2^K Factorial` - This method simulates all possible combinations of the min and max variable values. Use this method to capture interactions of variables at their extreme values.
- `Full Factorial` - This method simulates all possible combinations.
- `Automatic` - This method calculates the worst case corners automatically without manually specifying methods. For more details, see Creating Worst Case Corners Automatically on page 1588.

The first three algorithm methods, `OFAT 3-level`, `OFAT Sweep`, and `Central Composite Design`, are termed as Response Surface Methods (RSM), which means in these methods, a polynomial model is created and values for each corner variable is chosen based on the fact whether the specified value fits into the model. The worst case corner is the combination of variable values that are not simulated. As a result, after the worst case corner is created, the tool performs the corner validation for these methods.

For the `2^K Factorial` and `Full Factorial` methods, the tool determines the worst case corner from the simulated combination values. The combination that gives the worst result for each spec is identified as the worst case corner.

When you create a statistical corner from a given spec and then run Worst Case Corner simulation using OFAT, Central Composite Design, or Factorial methods to create worst case corner for other variables, then the worst case corner is created only for the spec for which the statistical corner was created. When the automatic method is used, the worst case corner is created for each enabled PVT or statistical corners.

The following table outlines the comparison between these five methods that helps you choose an appropriate method to get the desired results:

Methods	Accuracy	Simulation # (M parameters with N1, N2,...,Nm levels)	Example Simulation # (10 parameters with 3 level each)	Limitations
OFAT 3-level	Quadratic and non-interacting variables	`2*M+1`	`21`	Not accurate for strong non-linear or interacting variables

© 2023 Cadence Design Systems, Inc.

OFAT Sweep	Non-linear variables	`N1+N2+...+NM-M+1`	`21`	Not accurate for interacting variables
Central Composite Design	Quadratic and variable interactions	`2*M+2*C(M,2)+1`	`111`	Not accurate for strong nonlinear variables
2^K Factorial	Linear and interacting variables	`2^M`	`1024`	Not accurate for non-linear variables
Full Factorial	Most accurate	`N1*N2*...*NM`	`59049`	Long simulation time for large number of variables

Example

To better understand these method, consider the following example:

Suppose you have three variables, `var1`, `var2`, and `var3`, with the following values:

```
Var1: Lo, Nom, Hi

Var2: Lo, Nom, Hi

Var3: Val1, Val2, No, Val3, Val4
```

OFAT 3-Level

If you run the `OFAT 3-Level` method, the table below describes the combination of variables that are simulated for this run.

Point	Var1	Var2	Var3
1	`Nom`	`Nom`	`Nom`
2	`Nom`	`Nom`	`Val1`
3	`Nom`	`Nom`	`Val4`
4	`No`	`Lo`	`Nom`
5	`No`	`Hi`	`Nom`

© 2023 Cadence Design Systems, Inc.

6	Lo	Nom	Nom
7	Hi	Nom	Nom

OFAT Sweep

If you run the `OFAT Sweep` method, the table below describes the combination of variables that are simulated for this run.

Point	Var1	Var2	Var3
1	Nom	Nom	Nom
2	Nom	Nom	Val1
3	Nom	Nom	Val2
4	Nom	Nom	Val3
5	Nom	Nom	Val4
6	Nom	Lo	Nom
7	Nom	Hi	Nom
8	Lo	Nom	Nom
9	Hi	Nom	Nom

Central Composite Method

If you run the `Central Composite Design` method, the table below describes the combination of variables that are simulated for this run.

Point	Var1	Var2	Var3
1	Nom	Nom	Nom
2	Hi	Nom	Nom
3	Lo	Nom	Nom
4	Nom	Hi	Nom
5	Nom	Lo	Nom

© 2023 Cadence Design Systems, Inc.

6	Nom	Nom	Val4
7	Nom	Nom	Val1
8	Hi	Hi	Nom
9	Lo	Lo	Nom
10	Hi	Nom	Val4
11	Lo	Nom	Val1
12	Nom	Hi	Val4
13	Nom	Lo	Val1

2^K Factorial

If you choose the `2^K FActorial` method, the table below describes the combination of variables that are simulated for this run

Point	Var1	Var2	Var3
1	Lo	Lo	Val1
2	Lo	Lo	Val4
3	Lo	Hi	Val1
4	Lo	Hi	Val4
5	Hi	Lo	Val1
6	Hi	Lo	Val4
7	Hi	Hi	Val1
8	Hi	Hi	Val4

Full Factorial

The `Full Factorial` method includes the 45 combinations of variables. This is the most accurate method.

© 2023 Cadence Design Systems, Inc.

Determining the Number of Data Points

The *No of Points* label to the right of the *Method* field displays the total number of data points for which simulation is to be run. The tool calculates these data points based on the algorithm selected in the *Method* drop-down list, nominal value and the range of parameter values. Normally, the tool dynamically changes the value of this label.

Importing Values from the Corners Setup

In the tabular format given in the centre of the form, specify the variables that you want to vary during the analysis. If the corners are already defined in the Corners Setup form, you can use the *Import from Corner Setup* command to directly import all or selected variables from the existing setup. When you click this button, the following two options appear:

- *Enabled Corners*—Select this option to import only enabled corners from the Corner Setup form.
- *All Corners*—Select this option to import all the corners that are present in the Corner Setup form.

Note: The *Import from Corner Setup* command follows the local corner selection, which means if corners are globally disabled in Run Summary, but enabled in the Corner Setup form, these corners are also imported in the Worst Case Corner form.

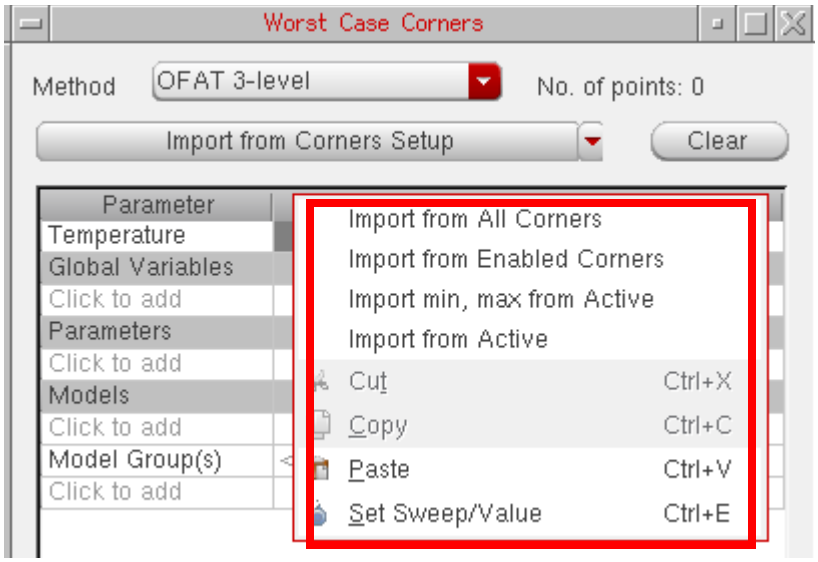
Tip

At any point, you can click the *Clear* button to clear all the content of the Worst Case Corner form.

© 2023 Cadence Design Systems, Inc.

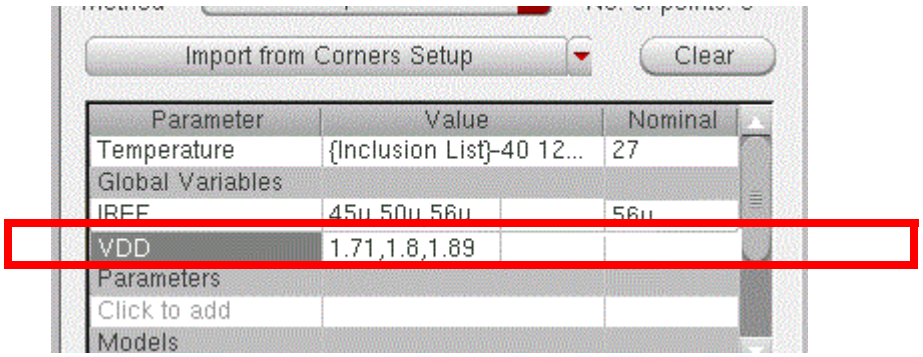
Specifying Temperature

To vary temperature, specify values for temperature in the Value cell. To import the value of temperature from all the corners, right-click in the *Value* column and choose *Import from All Corners* (refer the figure below).



Type a nominal temperature value in the *Nominal* cell for temperature or double-click in the cell to open the drop-down list. All the values specified in the *Value* cell are displayed in the list. Select a nominal temperature value from this list. The nominal value should be in the given value list. If you type a nominal temperature value that is not in the specified value list, that value is not accepted by the form.

Alternatively, you can get the value of temperature from the test. For this, right-click in the *Nominal* cell for temperature and choose *Get Value from Test*. If the value taken from the test does not exist in the *Value* cell, the tool automatically creates an inclusion list to add the nominal value to the existing values.



© 2023 Cadence Design Systems, Inc.

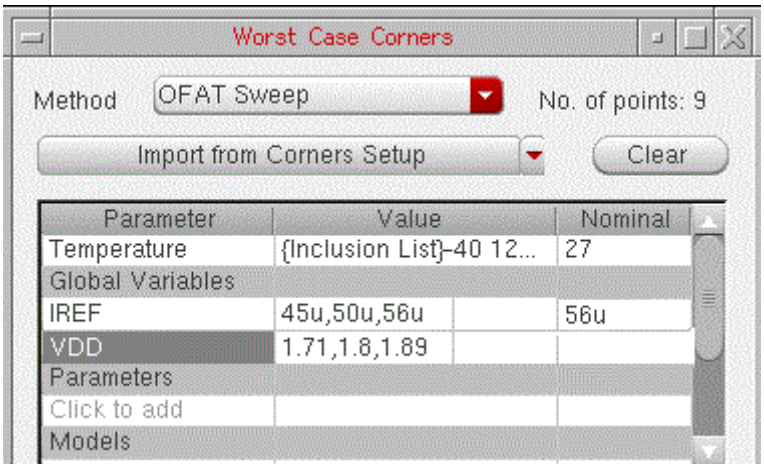
Specifying Global Variables

To specify global variable values:

➡ In the *Global Variables* field, click `Click to add`.

A drop-down list containing variables is displayed. Select the required variables from this list. The tool reads the range of values from the active setup in ADE Assembler and displays them in the *Value* column. The tool also reads the value of that variable from the test and if that value is one of the values in the *Value* column, displays the same in the *Nominal* cell.

Note: If you want to add all variables in the setup in a single step, select the `ALL` option from the drop-down list that appears when clicking `Click to add`.



For example, in the above figure, the value for *`IREF`* in the test is `56u`, which is there in the value list. Therefore, the tool automatically fills that value in the *Nominal* cell. However, for *`VDD`*, the value in the test is `2.0`, which is not in the given value list. Therefore, no value is displayed in the *Nominal* column. In this case, you can choose to get the nominal value either from the test or a setup state. For that, right-click in the *Nominal* column and choose an appropriate option. If the value taken from any of these sources does not exist in the *Value* cell, the tool automatically creates an inclusion list to add the nominal value to the existing set of values in the *Value* cell.

Note: If required, you can edit the value list in the *Value* cell. The tool dynamically calculates the number of points and changes the value in the *No. of points* label.

Specifying Parameters

To specify parameters:

➡ In the *Parameters* section, click `click to add`.

© 2023 Cadence Design Systems, Inc.

A drop-down list containing the parameters is displayed. Select the required parameters from this list. Similar to the global variables, the tool reads the range of values and design value for the selected parameters from the Variables and Parameters assistant and displays them in the *Value* and the *Nominal* column. Similar to the global variables, if the nominal value is not one of the values in the *Value* column, the *Nominal* cell is blank. You need to fill in that value.

Note: If there is only a single value in the *Value* column, the same value is filled in the Nominal cell as well. However, if required, you can change the nominal value. In that case, the tool creates an inclusion list with the values from both the *Value* and *Nominal* columns.

To change the parameter values, double-click in the *Value* column and edit the values by specifying a range or a list of space-separated values. Alternatively, you can select one or more values and right-click to display the shortcut menu and choose:

- *Import from All Corners* to import values defined for all corners in the Corner Setup form.
- Import from Enabled Corners to import values from defined for the enabled corners in the Corner Setup form.
- *Import Min, Max from Active* to load only the minimum and maximum values defined in the Data View or the Variables and Parameters assistant.
- Import from Active to load the values defined in the Data View or the Variables and Parameters assistant.
- Set *Sweep/Value* to edit the exclusion/inclusion list.

If required, you can change the nominal value for variables. The default nominal value is taken from the values of the parameter in the design. To change the nominal value for a variable or parameter, double-click in the *Nominal* cell for the variable or parameter. A list of possible nominal values is displayed. You can either select a value from the list or edit the value.

Note: If you specify a nominal value outside the specified range or the sweep set, the tool does not accept the value and makes the cell blank. A valid nominal value for a parameter is any value from within the range or sweep set corresponding to the parameter.

Specifying Model and Model Groups

To specify model files:

➡ Click `Click to add` in the Model section in the table.

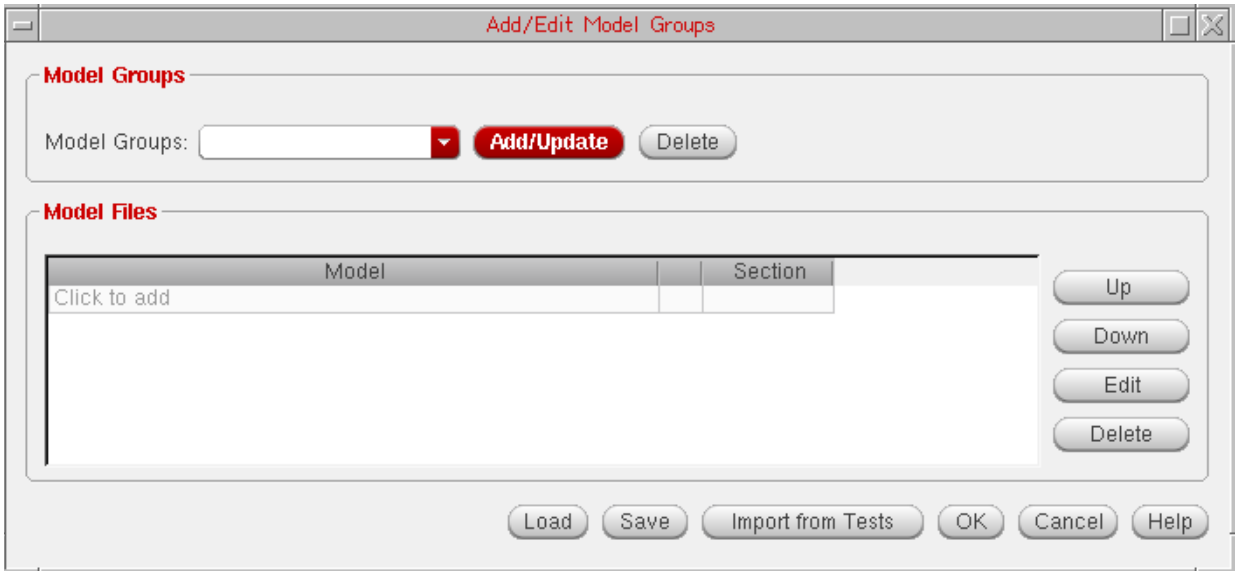
The *Add/Edit Model* Files form appears, in which you can specify the model files.

To specify model groups:

© 2023 Cadence Design Systems, Inc.

➡ Click `Click to add` in the Model Group(s) section in the table.

The *Add/Edit Model Group* form appears, as shown in the figure below:



- ❑ In the *Model Groups* drop-down, select the model groups that you want to vary for the simulation.
- ❑ In the *Model Files* panel, browse and select a model file for which you want to vary different sections. The name of the file appears in the cell.
- ❑ Select a section that you want to add to the list. You can select multiple sections that you want to vary for the model file.
- ❑ Click the *Import from Tests* button to import the model files directly from the test.
- ❑ Click *OK*.

For more information about the fields in this form, see Adding Model Files to a Corner.

To add a nominal value, double-click in the *Nominal* cell and select name of a section that you want to use as a nominal value for the selected model file.

Note: It is also possible to specify a model file without selecting any section. In this case, the model file specified in the Worst Case Corners form is used instead of the model file specified with the test.

The setup required for creating worst case corners is complete.

© 2023 Cadence Design Systems, Inc.

Running Worst Case Corners

Important

The worst case corners cannot be run with swept variables. Therefore, either disable sweeps or ensure that all the swept variables are overridden in the Worst Case Corners form.

After the worst case corner variables and methods have been specified in the Worst Case Corner form, click the *Run Simulation* button on the *Run* toolbar to run the simulation for the specified corners.

Below is a step-wise description about the basic tasks that are performed when a worst case corner simulation is run and the worst case corners are evaluated. These steps help you find out whether the design meets the specification across all corners before running a large number of simulations.

1. In the first step, the worst case corners are created by generating and simulating the design samples for the specified set of parameter values. You can use different algorithms to generate the worst case corners. For more information about the available algorithms, see Choosing a Method.
2. Then, the simulation is run for the specified parameter and values. If you have chosen an RSM algorithm in the Worst Case Corner form, the tool generates the predicted results and validates the obtained simulation results against the predicted results. For more information about the results validation, see Validating Worst Case Corners.
3. Now, the simulated results generated from the previous step are compared with the predicted result expressions. If the generated results do not match the predicted results, you can use a different algorithm and rerun the worst case corner simulation to achieve the desired results. This makes the process faster because you are not required to run all the simulations again if you use an RSM algorithm.
4. The above steps are then repeated to predict the values for all remaining corners. All the expressions are evaluated and then the simulation is run for all the combination sets.

When the Full Factorial method is used to create worst case corners, the simulation first checks if all specifications have been failed in the run, instead of only checking the test status because it can have partially failed specs too. Those specifications are ignored for which all the points have been failed and a message is displayed in the run log. In this case, the first failed point is considered as the worst case point.

Related environment variable: warnWhenWCCSimsExceed

© 2023 Cadence Design Systems, Inc.

Validating Worst Case Corners

If you have selected an RSM algorithm, such as OFAT 3-Level, OFAT Sweep, or Central Composite Design, the tool generates a set of predicted results and then validates these results against the results obtained from the actual simulation run.

Note: The validation step does not take place when you have only one variable, such as `temp`.

Refer to the following sections to know more about how the worst case corners are validated:

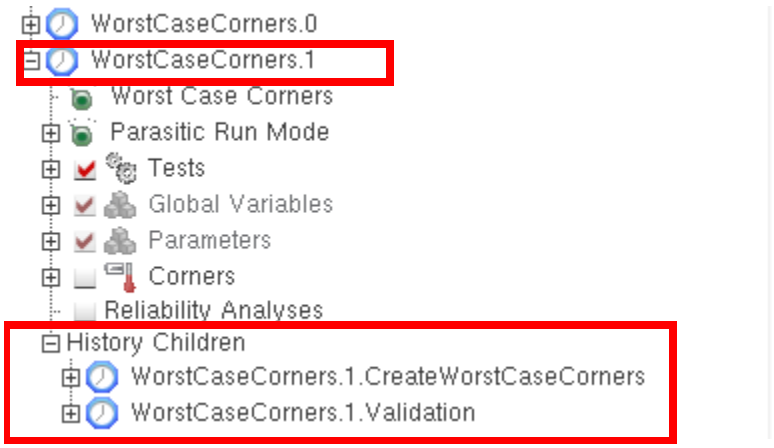
- Viewing Validation Results Using Run History
- Viewing Validation Results Using Log File

Viewing Validation Results Using Run History

To view the history information for a particular simulation run, click the *History* tab. The histories of the previous simulation run are displayed in the Data View assistant. The history name for a particular simulation run includes the name of the run mode.

Alternatively, you can click the history tabs displayed at the bottom of the Results window.

To view the history for a previous worst case corner run, expand *WorstCaseCorners.1* as shown in the figure below.



If you selected an RSM method, such as OFAT 3-level, OFAT Sweep, or Central Composite, the worst case corner simulation run creates a group run with child histories. To view child histories, expand the *History Children* section as shown in the figure above. The following two histories are displayed under this section:

- *WorstCaseCorners.1.CreateWorstCaseCorners*

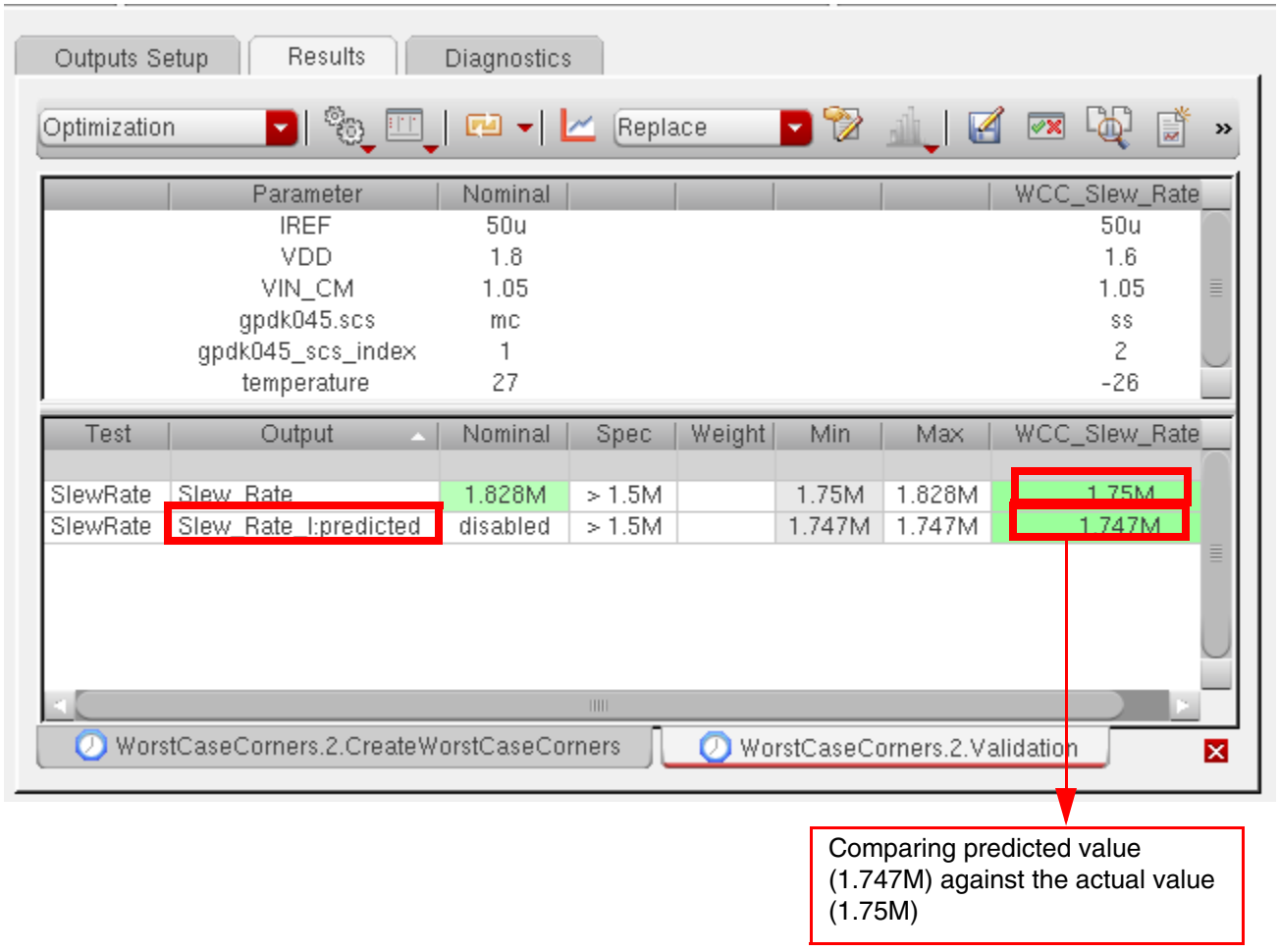
© 2023 Cadence Design Systems, Inc.

- *WorstCaseCorners.1.Validation*

In the first child run, the samples are generated and simulated to create the worst case corner. This run also generates the predicted result expressions for each corner if an RSM algorithm is used.

In the second child run, the worst case corners are simulated and validated against the predicted results. The validation is performed only if you have selected a RSM algorithm in the Worst Case Corners form.

In the second child history tab, the predicted worst case corner value is calculated for all corners and is displayed for each test in a separate row in the results window. The figure below shows the predicted and generated corners for `SlewRate`.

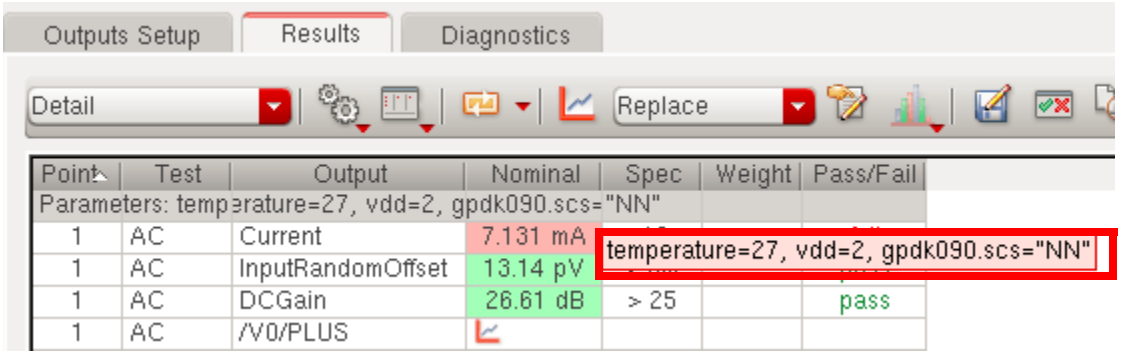


In this figure, the *Output* column displays two values for the Test, *SlewRate*, `Slew_Rate` and `Slew_Rate_I:predicted`.

© 2023 Cadence Design Systems, Inc.

In the next step, the simulated value for the test that has been generated after simulating the worst case corners is compared against the corresponding predicted value, and then the difference between these two values is calculated. If the test output includes a range spec, two worst case corners are created, one for the high value range and the other for the low value range. Similarly, two predicted results values are generated for high and low value ranges, respectively.

When you select to display the results in the *Detail* and *Detail-Transpose* view, a tooltip appears when you move the mouse pointer on the parameter values.

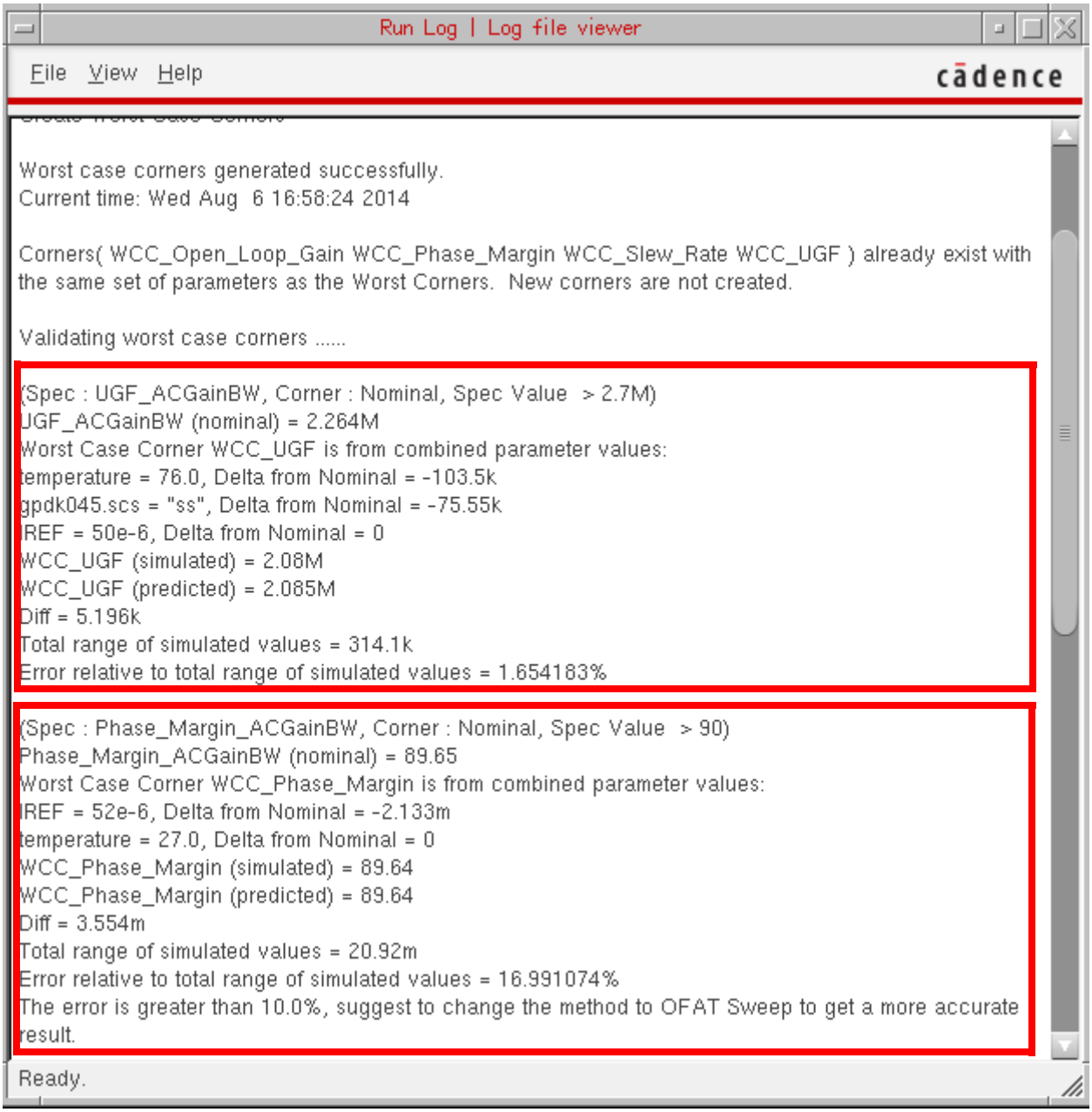


Viewing Validation Results Using Log File

A log file is created while a simulation for the worst case corners is running. This log file maintains a record of all the tasks that the tool performs during the worst case corner run (as shown in the figure below). This log file displays the difference between the actual simulated

© 2023 Cadence Design Systems, Inc.

value and the predicted value and the error relative to the total range of simulated values, which provides you a hint about whether the worst case corner run was successful.



If the relative error is greater than `10%` of the total range of simulated values, the tool displays a message in the log file that suggests you to use another alogrithm to get more accurate worst case corners. For example, in the figure above, the tool suggests you to use the OFAT Sweep algorithm to get the optimal results. You can then change the method in the Worst Case Corners form and rerun the simulation.

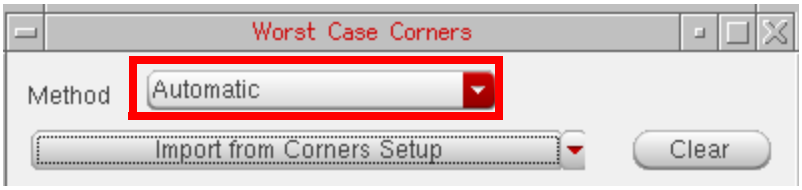
© 2023 Cadence Design Systems, Inc.

The approach described above is followed for all set of corners.

Creating Worst Case Corners Automatically

The `Automatic` method enables you to find worst case corners without manually specifying the algorithm to generate accurate worst case corners. It accurately and efficiently identifies worst cases corners for a wide range of applications. If you do not know which algorithm you need to apply to find worst case corners, it is recommended that you use the Automatic method, which finds all the worst case corners efficiently and generates worst case corners accurately by running fewer simulations.

To generate worst case corners using this method, select the `Automatic` option from the *Method* drop-down list in the *Worst Case Corner* form.



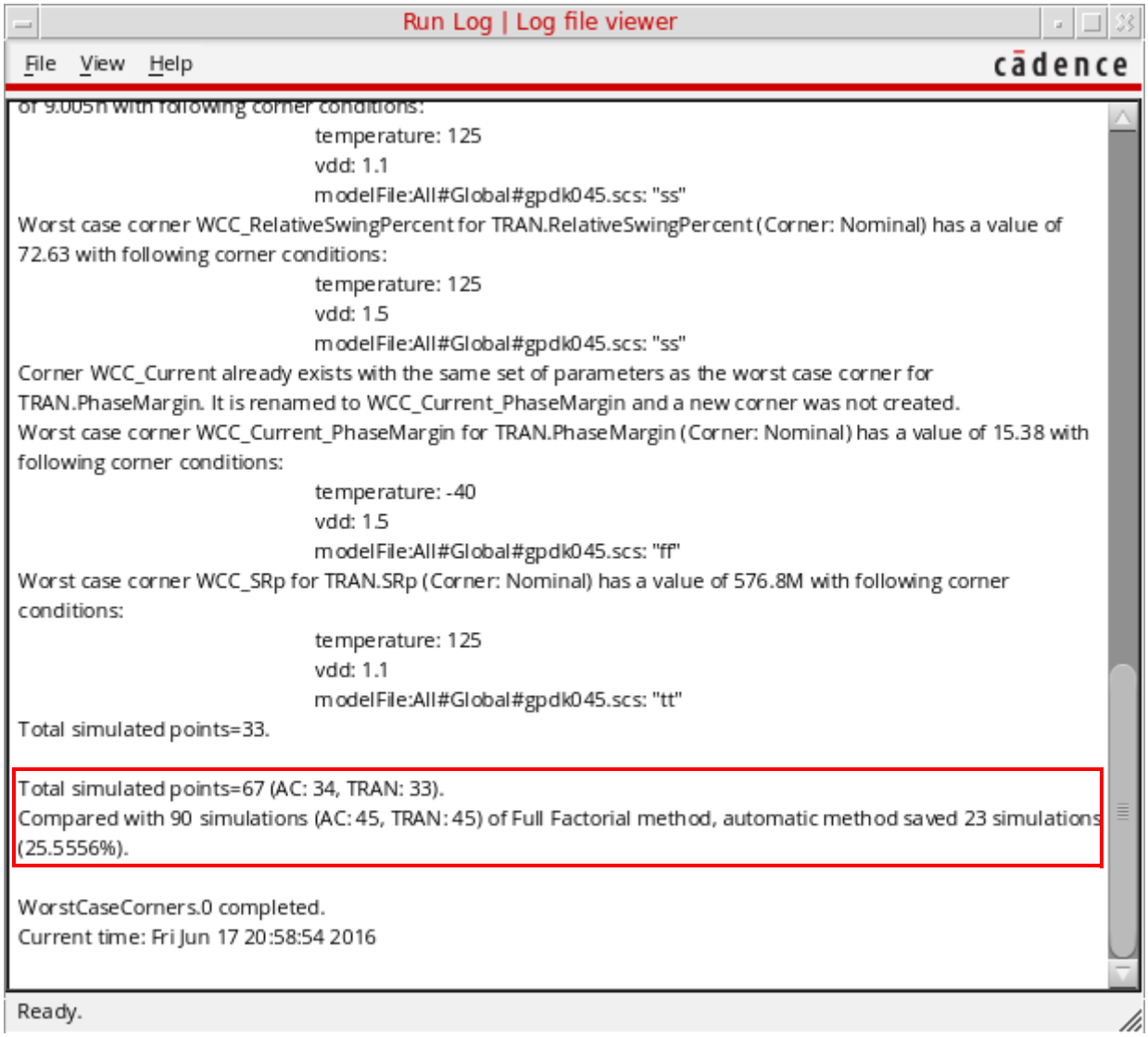
When you use this method, the tool performs the following tasks:

1. Runs OFAT sweep method.
2. Runs fractional factorial design on the important variables based on OFAT results.
3. Runs local optimization based on the worst case corner from OFAT and fractional factorial samples.

The worst case corners are generated and displayed in the Corners Setup form. For more details about different ways of viewing worst case corners, see Viewing Worst Case Corners.

© 2023 Cadence Design Systems, Inc.

The figure below displays the run log generated after the simulation run is complete.



For each test enabled in the setup, the run log displays the details of the worst case corners created and the total number of points simulated during the run.

From the information displayed at the end of the run log, you can determine that the `Automatic` method required a much lesser number of simulation runs as compared to the `Full Factorial` method.

When the method used is `Automatic`, you can also create worst case corners for a given specification by using the Sensitivity Analysis results. These results are generated after the Worst Case Corners simulation run is complete.

© 2023 Cadence Design Systems, Inc.

In the Sensitivity Analysis window, right-click in a cell and choose *Create Worst Corners*. The worst case corners are created for the selected specification and displayed in the *Corner Setup* form. This option can be used to re-create worst case corners when the previously generated corners are deleted.

Viewing Worst Case Corners

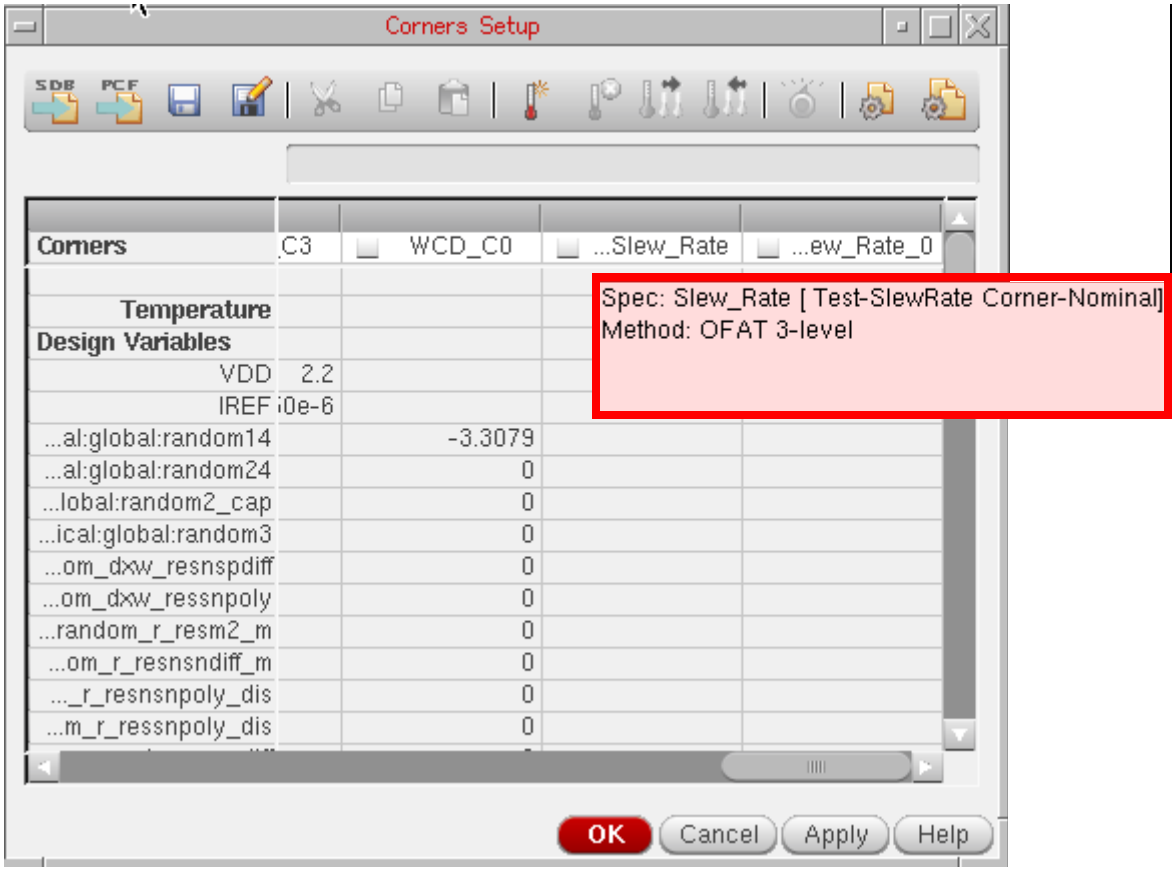
Worst case corners are created for every specification and are indicated by the prefix `WCC`. You can view worst case corners by doing one of the following:

- Viewing Worst Case Corners in the Corners Setup Form
- Viewing Worst Case Corners in the Data View
- Viewing Results in Sensitivity Analysis Window

© 2023 Cadence Design Systems, Inc.

Viewing Worst Case Corners in the Corners Setup Form

In the Corners Setup form, a tooltip displays the spec and method that has been used to create the worst case corners. Move the cursor on a worst case corner name to view the tooltip, as shown in the figure below.



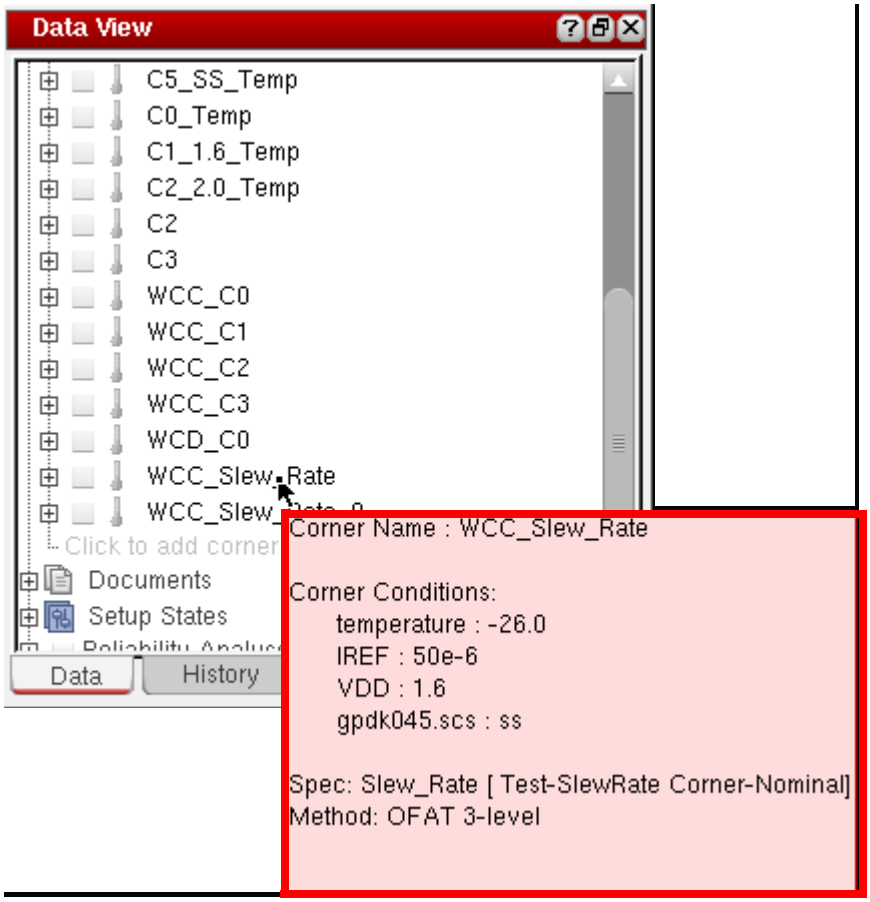
Viewing Worst Case Corners in the Data View

To view the worst case corners from the Data View assistant pane, follow these steps:

1. In the Data View assistant pane, click the *Data* tab.
2. Expand the *Corners* tree.
3. Point to a worst case corner name, such as `WCC_SlewRate`.

© 2023 Cadence Design Systems, Inc.

A tooltip, as shown in the figure below, displays the spec and method used for creating the worst corners.

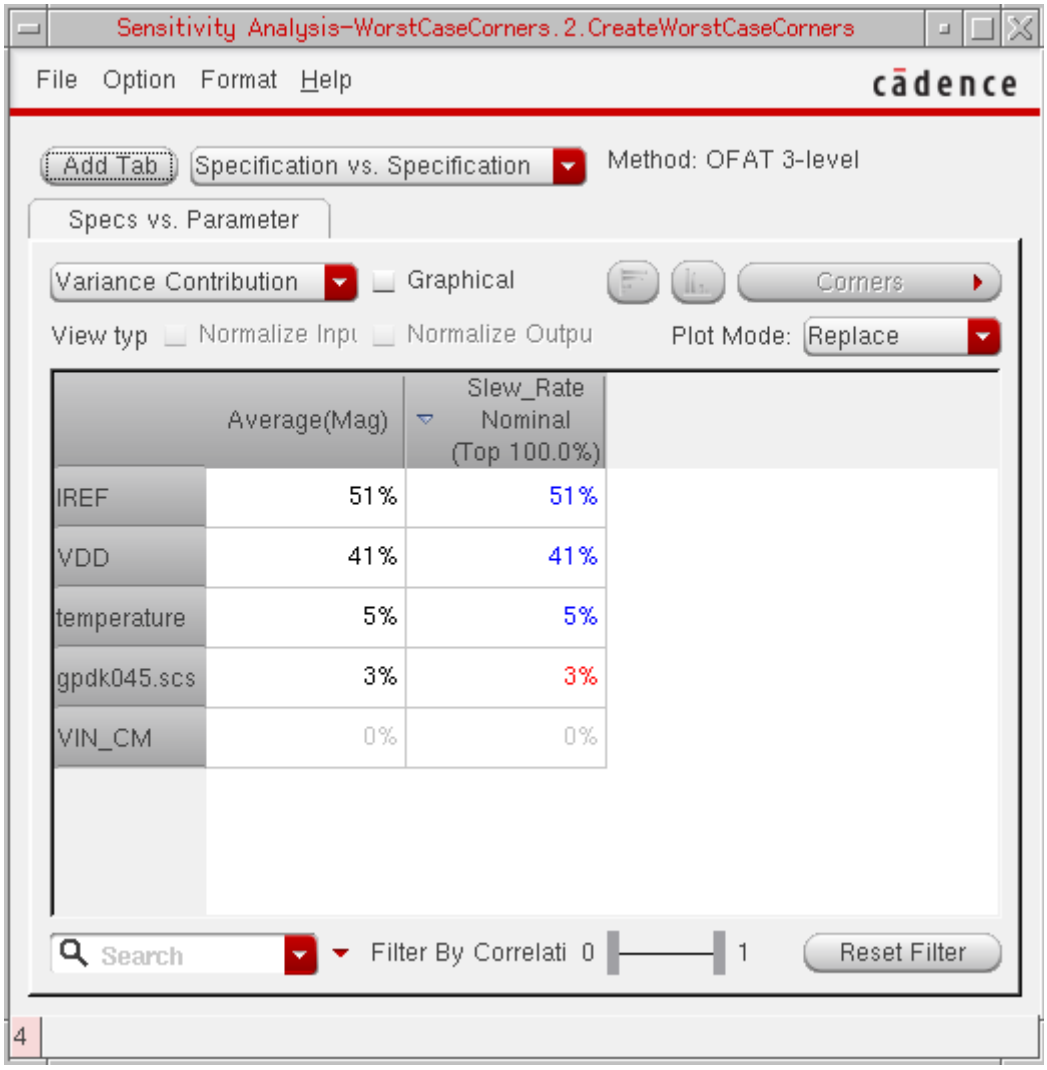


Viewing Results in Sensitivity Analysis Window

The Sensitivity Analysis window opens up automatically when the simulation run is complete. This window displays the results generated from the actual simulation data. The results,

© 2023 Cadence Design Systems, Inc.

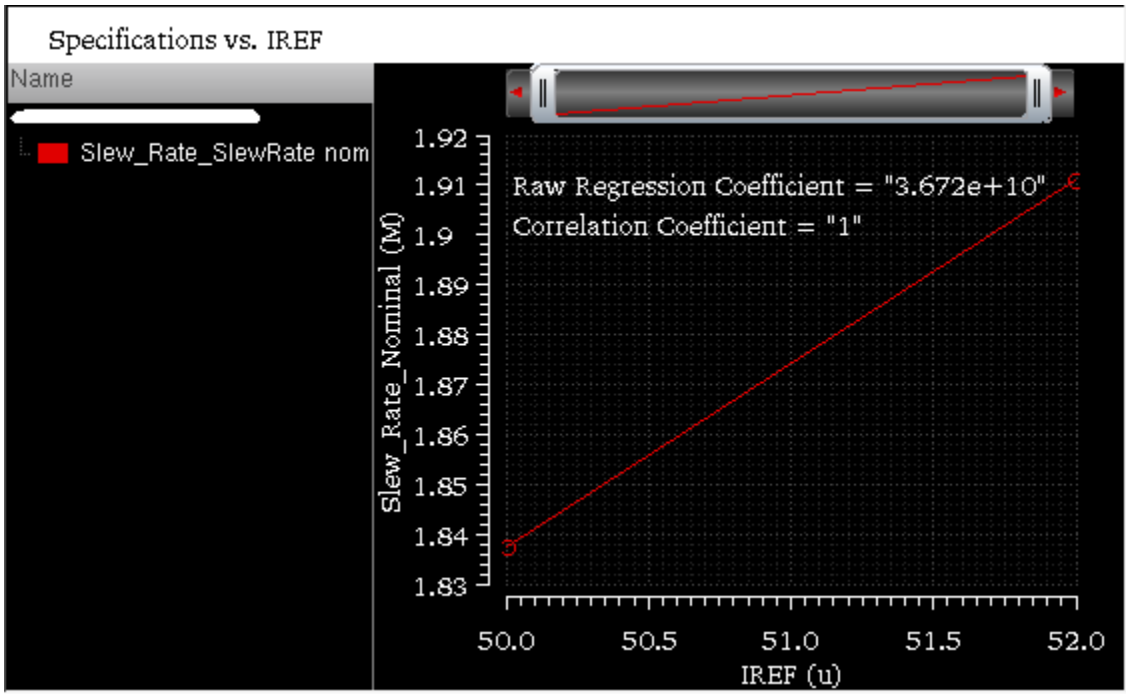
which are the relationship between each measurement and the parameter that you change, are displayed in a separate row. By default, the result is displayed in the percentage format.



When you double-click a value displayed in the cell, the Virtuoso Visualization and Analysis XL window appears displaying the plots for the selected measurement value. For example, if you double-click the *IREF* and *Slew_Rate Nominal* cell, the following graph shown in the

© 2023 Cadence Design Systems, Inc.

below figure appears. This graph displays the regression and correlation coefficient values for the selected measurement results.



Specifying the Number of Best Design Points to be Saved

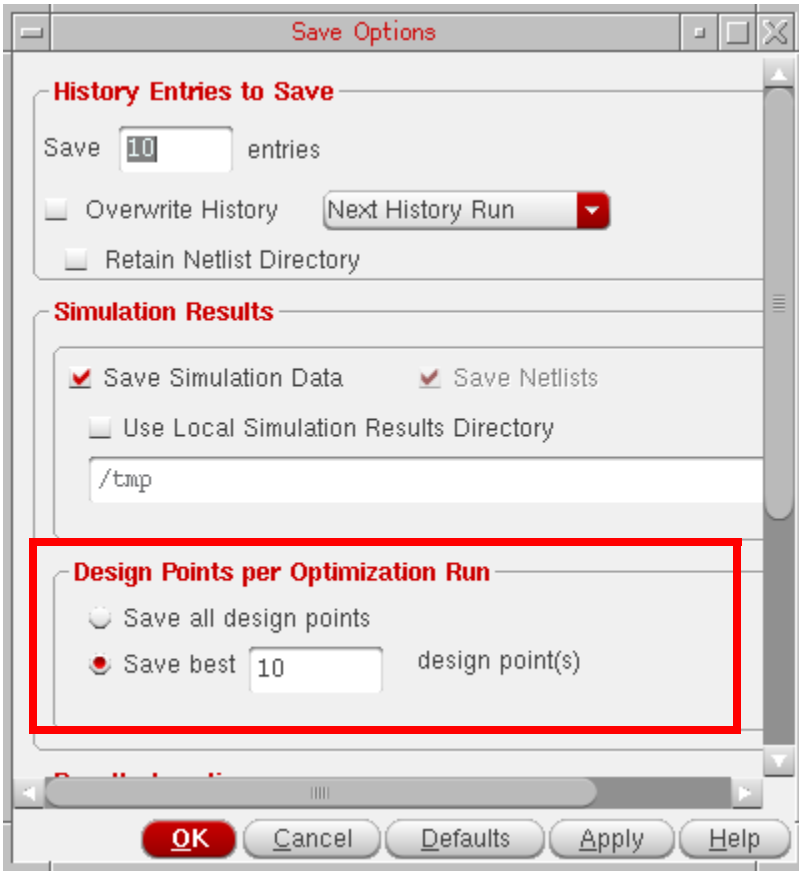
When you specify the number of design points that are to be saved during a optimization run, the same setting is applied to the worst case corners run. This means only the specified number of best design points are saved in the database after the run is complete and not all the design points.

To specify the number of best design points that are to be saved during a optimization or worst case corners run:

➥ Choose *Options – Save*. The *Save Options* form appears.

© 2023 Cadence Design Systems, Inc.

In this form, in the *Design Points per Optimization Run* section, select the *Save best design point(s)* check box and specify the number of best design points to be saved.



Alternatively, to save all the design points in the database, select the *Save all design points* option.

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

27

Multi-Technology Designs and Locally Scoped Models and Options

You can use multi-technology simulation (MTS) when designing high-speed interfaces between integrated circuits (ICs) to see the effects of signals traveling between two or more chips. You can also use MTS for custom IC system-in-package (SiP) designs. SiP designs are fully functional systems or subsystems in an IC package. MTS allows you to combine passive components with ICs fabricated using different process technologies while minimizing parasitic loading.

Your MTS design must consist of a top-level schematic that contains two or more blocks, each with its own associated set of process definitions, model libraries, simulation states, and other setup files. You cannot use MTS with a non-hierarchical schematic because each different technology must be self-contained. You must instantiate device instances from different technologies in different cell views so that the program can map model information appropriately.

Note the following:

- Different MTS setups can be stored for different tests.
- MTS settings are saved in the ADE test setup with the maestro view. If any existing MTS settings are found in the config view, they are automatically migrated to the ADE view and removed from the config view provided that the `MTSremoveOldConfigSetupAfterMigration` environment variable is set to `true`.
- You can setup MTS options only when the design in your test is a config view. If you change the design to a schematic, the MTS settings are not retained even if the `asimenv.retainStateSettings` environment variable is set to either `design`, `all`, or `yes`.
- If you change the simulator, the MTS settings are disabled. To enable the MTS settings again, open the MTS Options form and select the *Enable locally scoped models and options (MTS)* check box.

© 2023 Cadence Design Systems, Inc.

Video

You can watch a video demonstration for this feature at Multi-Technology Simulation (MTS) with Local Scoped Models. Also, read the related blog at Virtuoso Video Diary: Multi-Technology Simulation—The Good has Changed for Better.

Related Topics

Enabling Locally Scoped Models and Options

Enabling Local Scoping for a Cell

Specifying Locally Scoped Models for a Cell

Specifying Local Simulator Options for a Cell

Enabling Local Scoping for an Instance

Specifying Locally Scoped Models for an Instance

Specifying Local Simulator Options for an Instance

retainStateSettings

© 2023 Cadence Design Systems, Inc.

Enabling Locally Scoped Models and Options

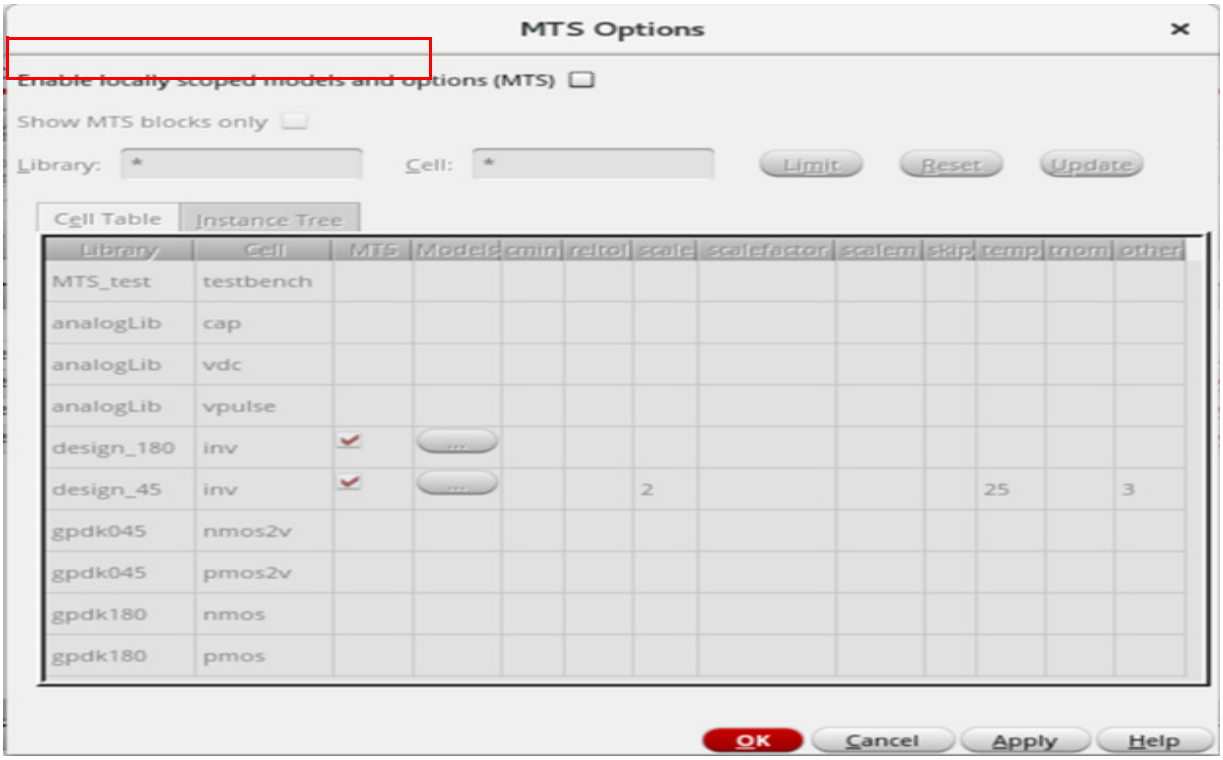
Important

You cannot enable locally scoped models and options (MTS options) for third-party simulators.

To enable MTS options in your test setup, do the following:

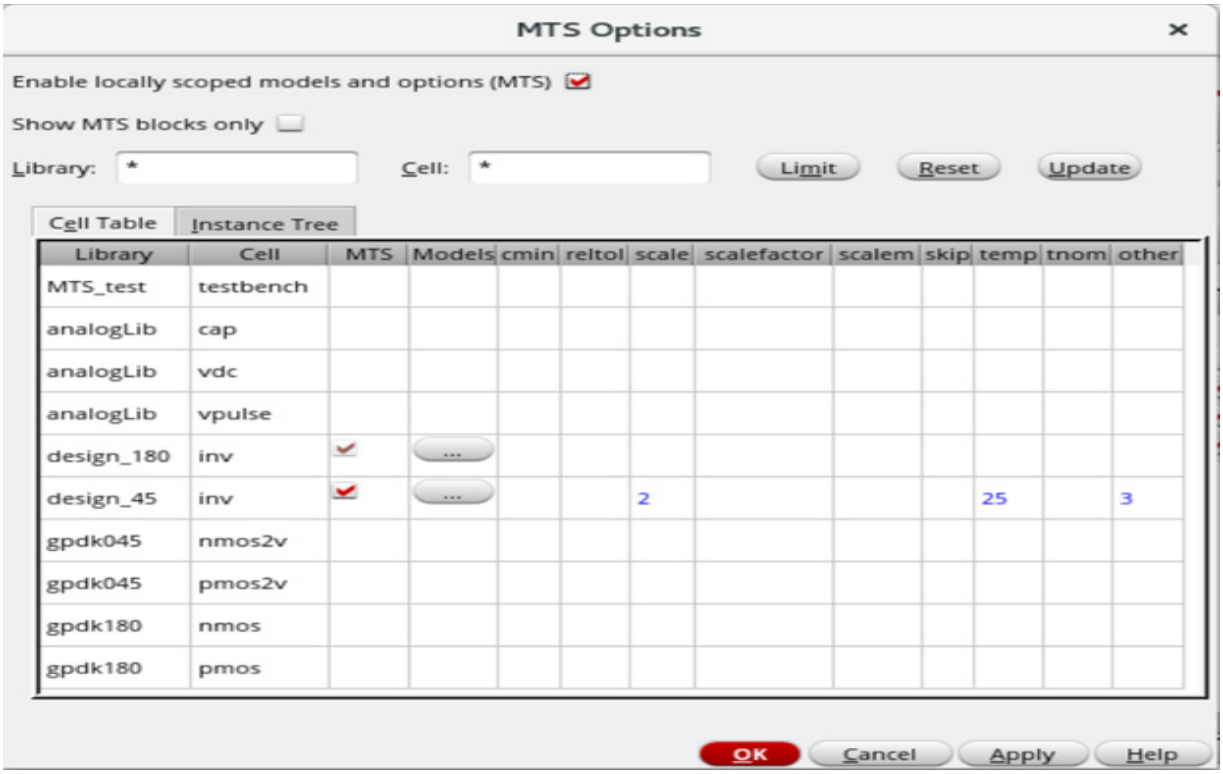
1. In the *Data View* assistant, right-click the test or analysis name, and then choose *MTS Options*.

   The MTS Options form appears.



© 2023 Cadence Design Systems, Inc.

2. Select the *Enable Locally scoped models and options (MTS)* check box to enable the MTS options for the selected test.



3. Click *OK* to save the changes.

After enabling the MTS options in your test setup, you can enable local scoping at the following levels in your design hierarchy:

- Cell
- Instance

Important Points to Note

- For Spectre, AMS, and UltraSim simulators, local scoping can be enabled at both these levels– the cell and instance levels.
- For third-party simulators, you cannot enable MTS options in your test setup.

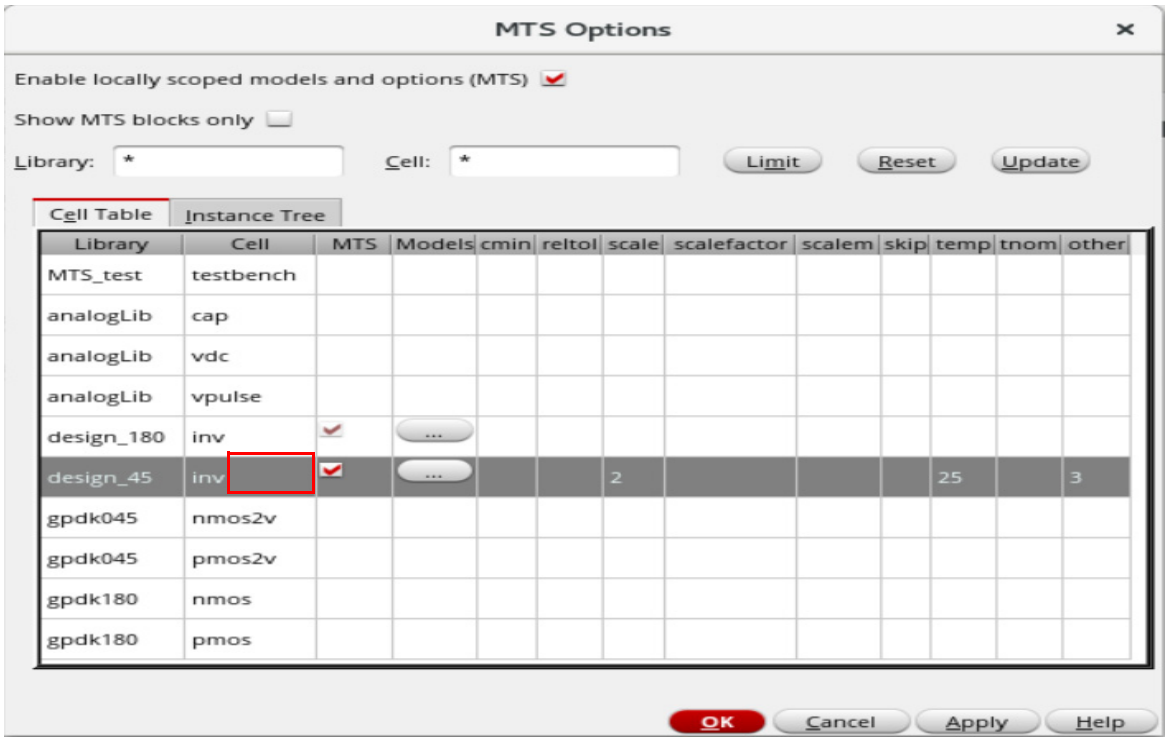
© 2023 Cadence Design Systems, Inc.

Enabling Local Scoping for a Cell

Note: Before enabling local scoping at the cell level, ensure that you have enabled local scoping in your test setup. See the Enabling Locally Scoped Models and Options section for more information.

To enable local scoping for a cell, do the following:

1. In the MTS Options form, click the *Cell Table* tab to view a list of the library and cells in your test setup.
2. Click the check box in the *MTS* column for the cell that you want to enable for local scoping, as shown in the following figure.



3. Click *OK* to save the changes.

   The selected cell is now enabled for local scoping.

   You can now specify the model libraries that will be locally scoped at the selected cell. See the Specifying Locally Scoped Models for a Cell section for more information.

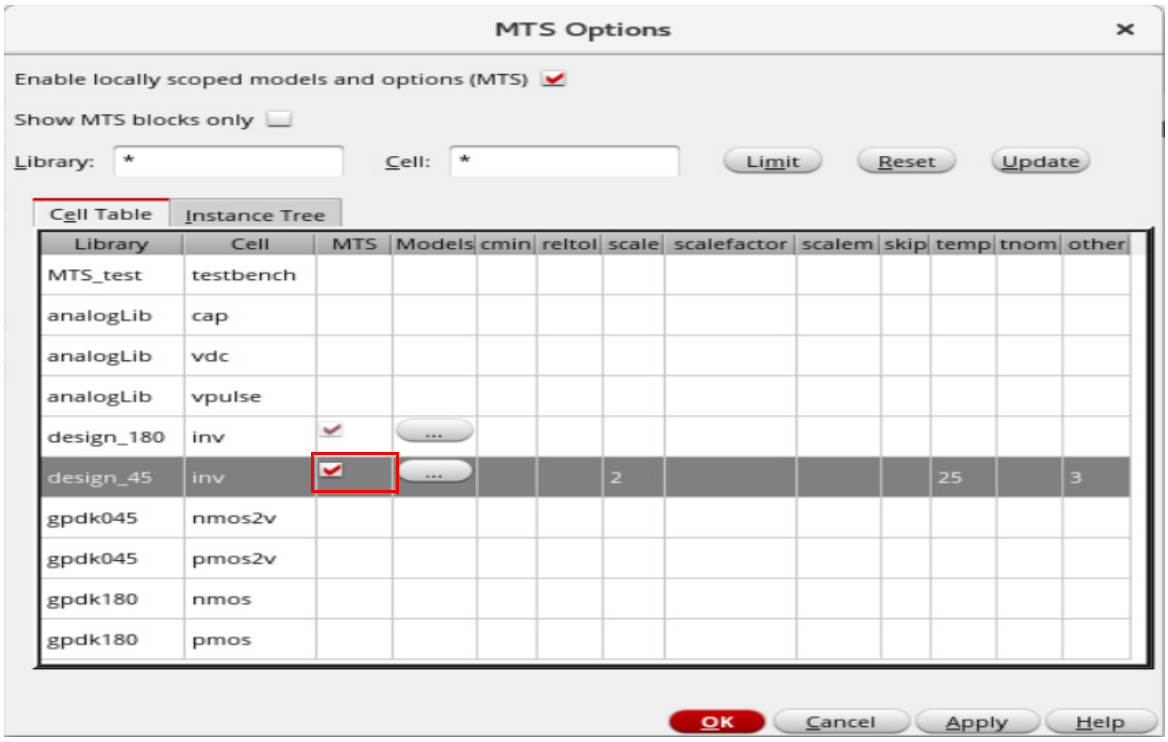
   You can also specify various process parameters that you want to include locally in the simulation. See the Specifying Local Simulator Options for a Cell section for more information.

© 2023 Cadence Design Systems, Inc.

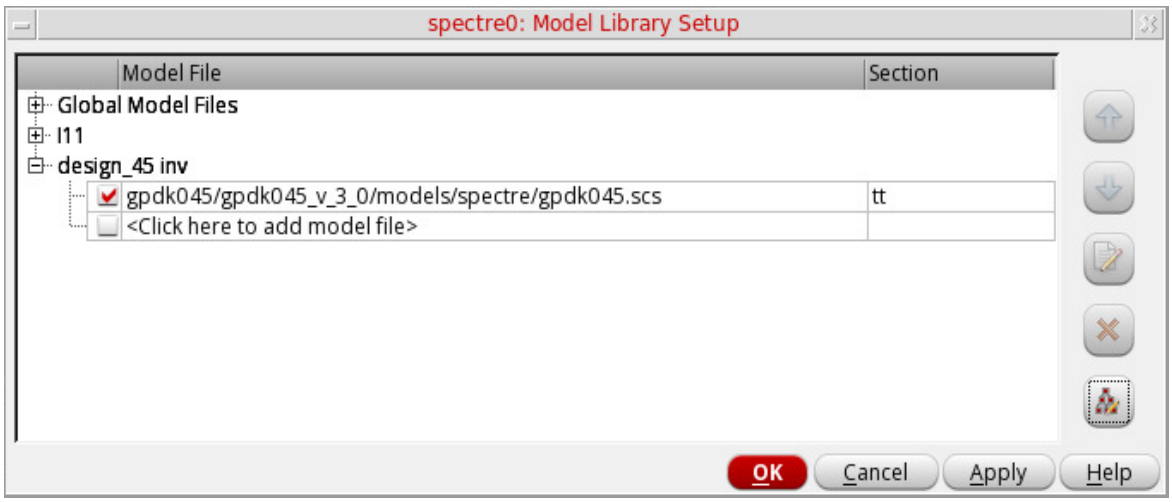
Specifying Locally Scoped Models for a Cell

To specify the models that will be locally scoped at the cell level, do the following:

1. On the *Cell Table* tab, in the row for the cell for which you have enabled local scoping, click the ellipsis (…) button the *Models* column, as shown in the following figure.



The Model Library Setup form appears, as shown in the following figure.



2. Follow the instructions described in the "Specifying Model Libraries" section to select the model files.

© 2023 Cadence Design Systems, Inc.

Specifying Local Simulator Options for a Cell

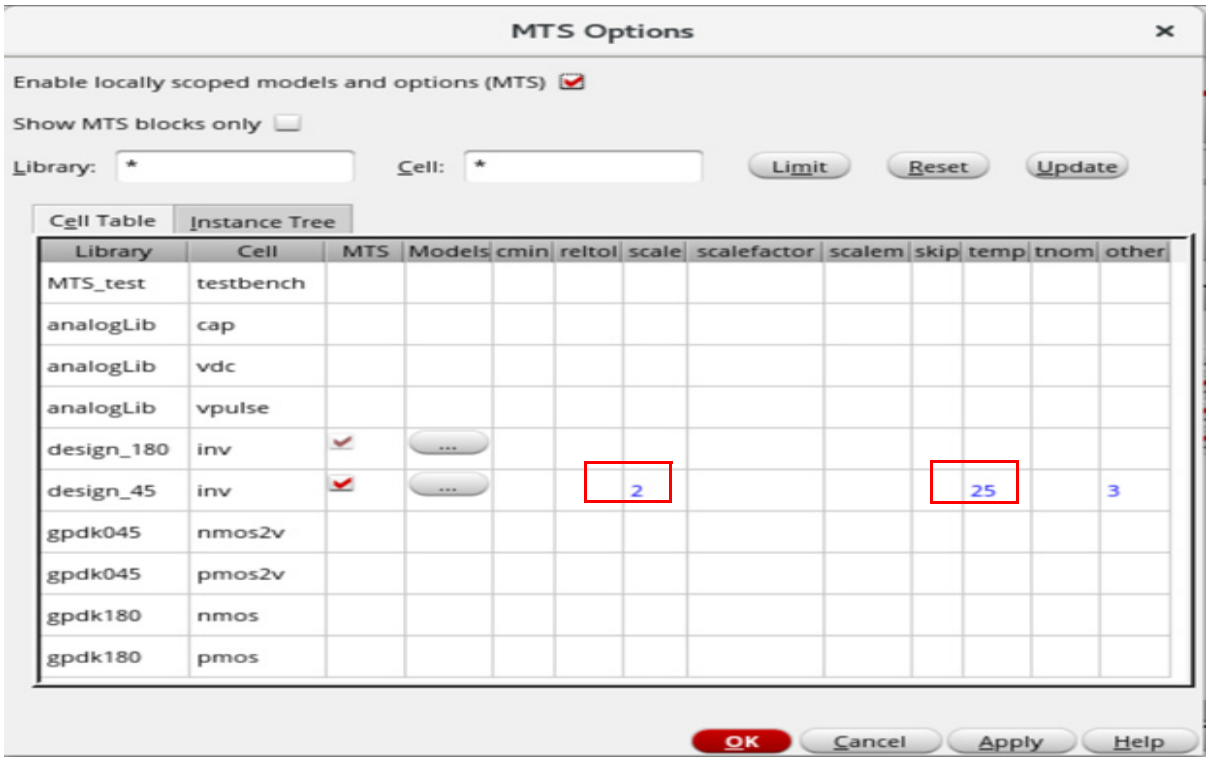
Important

You can specify local simulator options only for the cells that are enabled for local scoping. See the Enabling Local Scoping for a Cell section for more information.

To specify the various process parameters, such as `tnom`, `scale`, `scalefactor`, `temp`, and `scaleem`, that you want to include in the simulation for a cell, do the following:

1. On the *Cell Table* tab, click in the column for the simulator option you want to specify, and then type a value.

   The following figure shows an example where the local simulator options, `scale` and `temp`, are specified for the `inv` cell.



   Note: You can also specify a `VAR` expression in the *scale*, *scalem*, *scalefactor*, *temp*, and *tnom* columns. The variable name specified in the `VAR` expression is added to the *Design Variables* section in the test setup.

2. Click *OK* to save the changes.

© 2023 Cadence Design Systems, Inc.

Important point to note:

- All the options marked with an asterisk (*) in the Spectre help (`spectre -help options`) are available as columns, such as, `reltol`, `tnom`, and `scaleem`, in the MTS Options form. If you want to include these options in the simulation locally, you can specify their values in the respective columns.
- In addition to the currently supported local options, if Spectre supports new local options in future, you can include them too in the simulation locally by specifying their values in the `other` column.

© 2023 Cadence Design Systems, Inc.

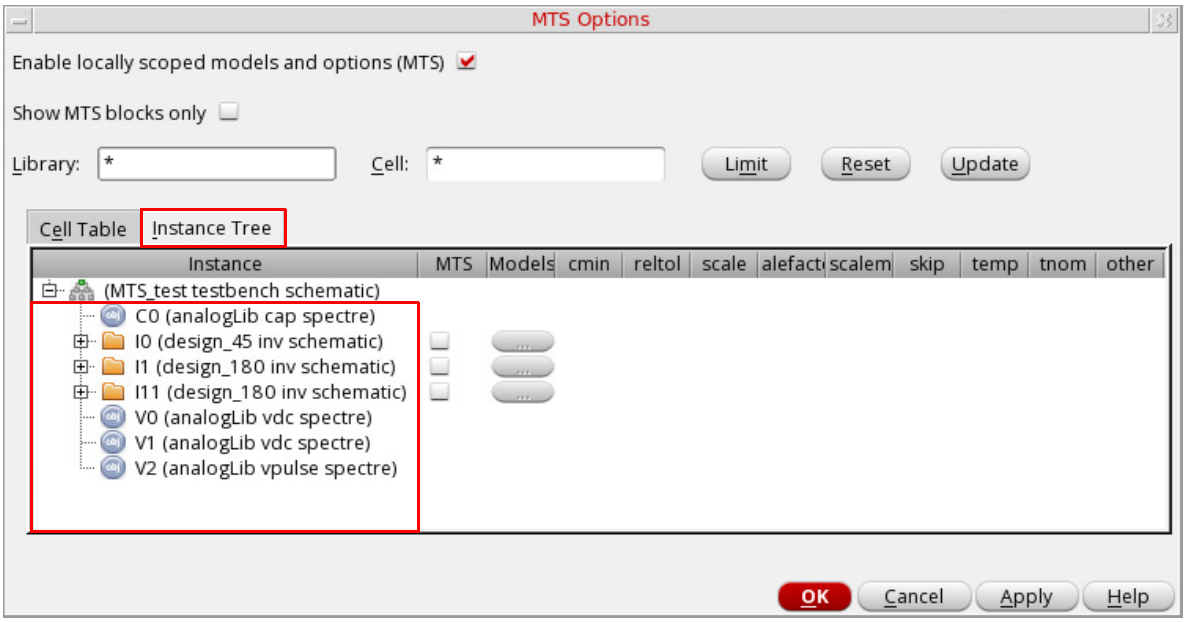
Enabling Local Scoping for an Instance

Important

Before enabling local scoping at the instance level, ensure that you have enabled it in your test setup. See the Enabling Locally Scoped Models and Options section for more information.

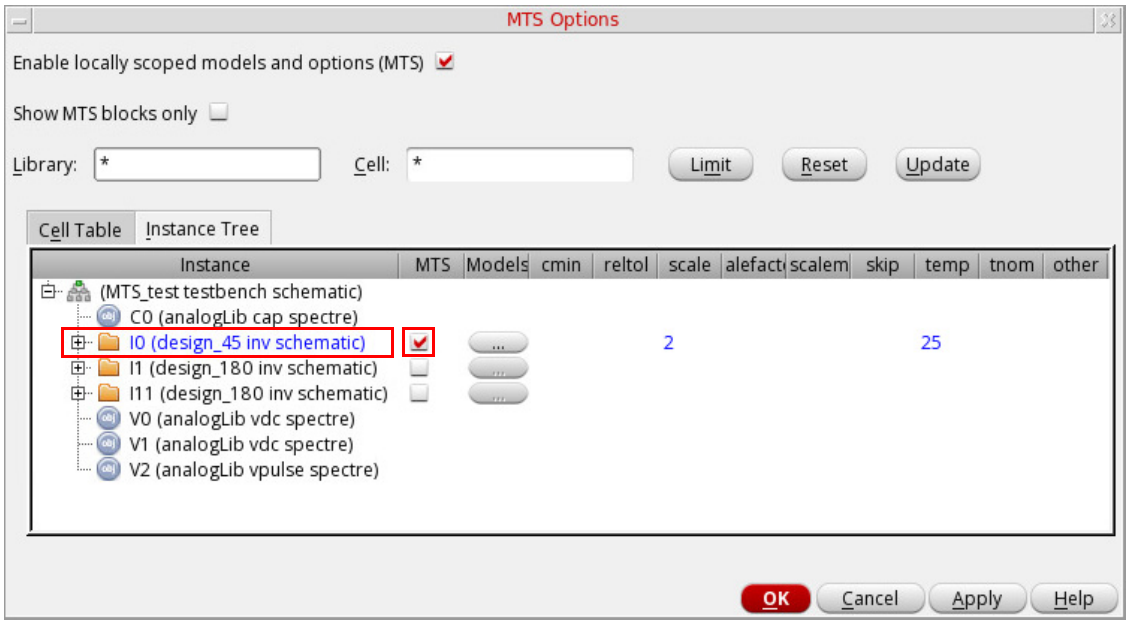
To enable local scoping for an instance, do the following:

1. In the *MTS Options* form, click the *Instance Tree* tab to view an alphabetical list of the instances available in your design.



© 2023 Cadence Design Systems, Inc.

2. Select the check box in the *MTS* column for the instance that you want to enable for local scoping.



Note: Instances that appear in blue indicate that local scoping has been enabled for them.

3. Click *OK* to save the changes.

You can now specify the model libraries that will be locally scoped for the selected instance. See the Specifying Locally Scoped Models for an Instance section for more information.

You can also specify various process parameters that you want to include locally in the simulation. See the Specifying Local Simulator Options for an Instance section for more information.

© 2023 Cadence Design Systems, Inc.

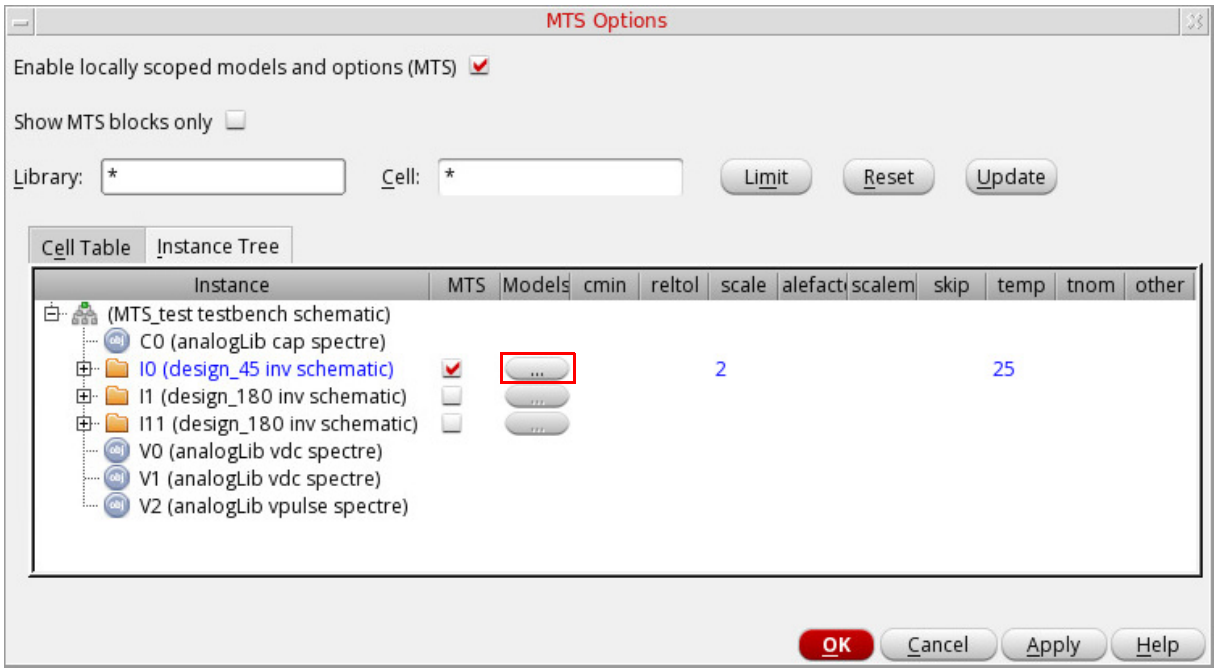
Specifying Locally Scoped Models for an Instance

Important

You can specify locally scoped models only for the instances that are enabled for local scoping. See the Enabling Local Scoping for an Instance section for more information.

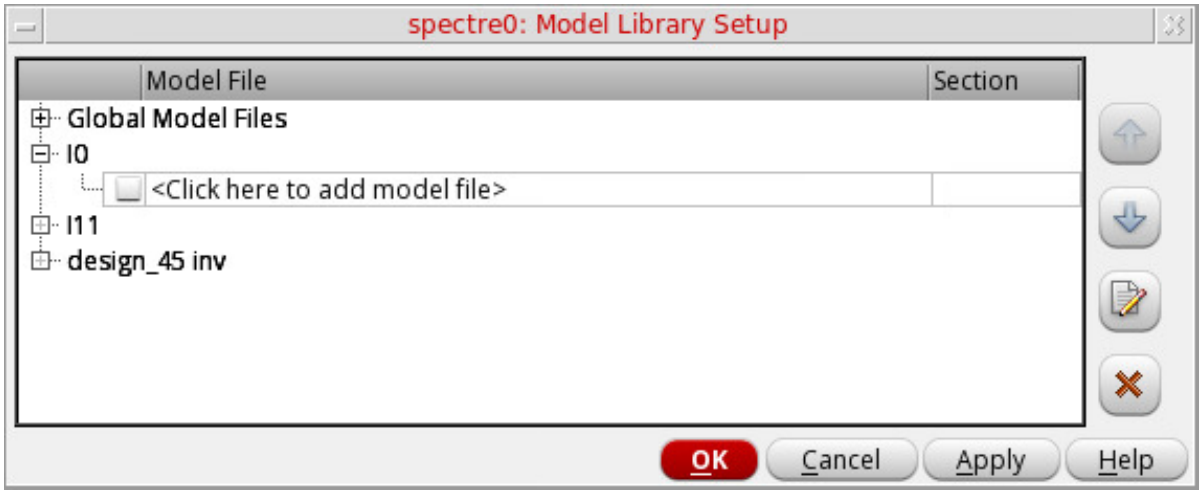
To specify the models that will be locally scoped for an instance, do the following:

1. On the *Instance Tree* tab, click the ellipsis (...) button in the *Models* column for the instance that is enabled for local scoping, as shown in the following figure. Such instances appear in blue.



© 2023 Cadence Design Systems, Inc.

The Model Library Setup form appears, as shown in the following figure.



2. Follow the instructions described in the Specifying Model Libraries section to select the model files.

© 2023 Cadence Design Systems, Inc.

Specifying Local Simulator Options for an Instance

Important

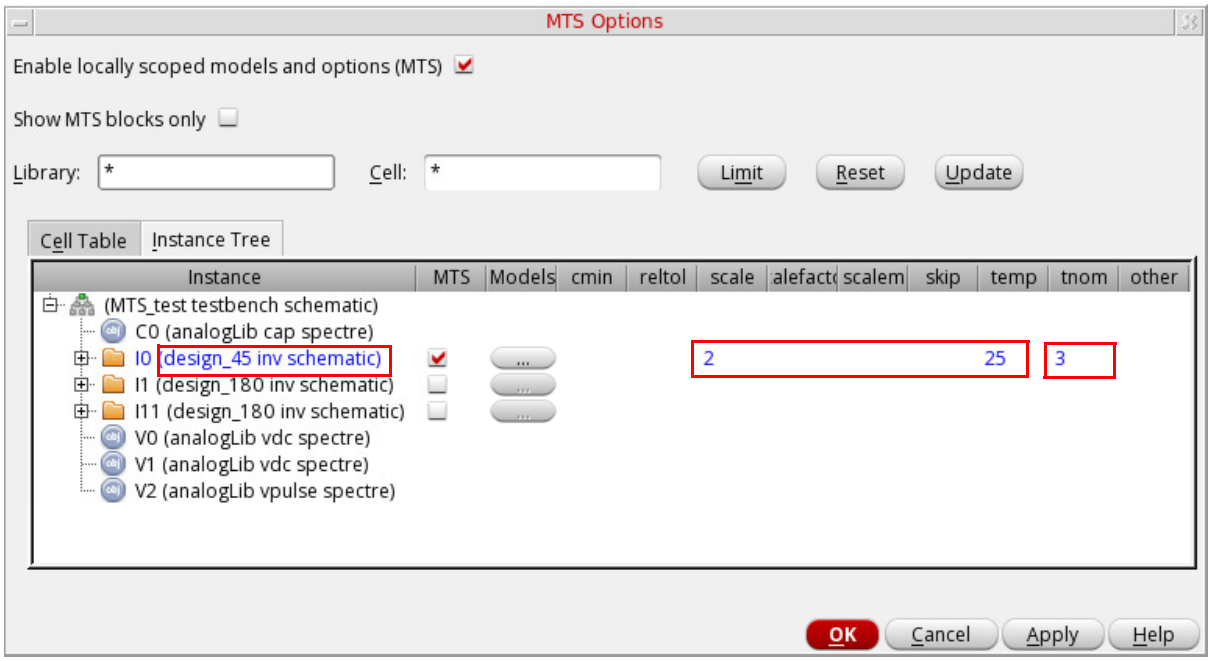
You can specify local simulator options only for the instances that are enabled for local scoping. See the Enabling Local Scoping for an Instance section for more information.

To specify the various process parameters, such as `tnom`, `scale`, `scalefactor`, `temp`, and `scaleem`, that you want to include in the simulation for an instance locally, do the following:

1. On the *Instance Tree* tab, click in the column for the simulator option you want to specify, and then type a value.

   Note: If the local simulator options are specified at the cell level, they are copied to the instances from their cells, provided that the local scoping is enabled at both the cell and the instance levels.

   The following figure shows an example where the local simulator option, `tnom`, is specified at the `I0` instance and `scale` and `temp` are copied from the `inv` cell that contains the `I0` instance in the design hierarchy.



   Note: You can also specify a `VAR` expression in the *scale*, *scalem*, *scalefactor*, *temp*, and *tnom* columns. The variable name specified in the `VAR` expression is added to the *Design Variables* section in the test setup.

2. Click *OK* to save the changes.

© 2023 Cadence Design Systems, Inc.

Disabling Local Scoping

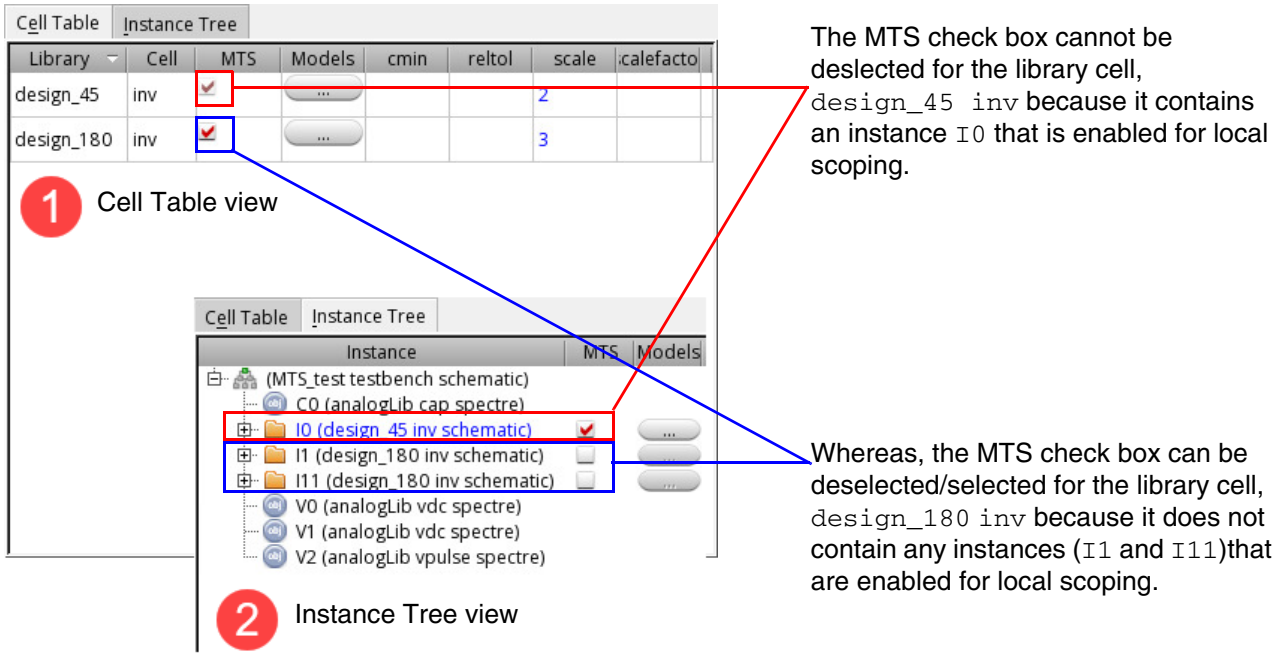
Perform the following steps to disable local scoping for a cell or an instance:

- In the MTS Options form, on the *Cell Table* tab or the *Instance Tree* tab, deselect the *MTS* check box next to a cell or an instance for which you want to disable the local scoping.

Important Point to Note

You cannot disable the local scoping for a cell if the local scoping is enabled for any of its instances. For such library cells the *MTS* check box appears dimmed and cannot be deselected.

In the following example, a side-by-side view of the *Cell Table* and *Instance Tree* views is shown.



© 2023 Cadence Design Systems, Inc.

Limiting the View by Library or Instance

You can limit what appears on the MTS Options form by library name, cell name, or both.

Note: You can reset the filter to view all cells in all libraries by clicking *Reset*.

To limit the view of design components on the MTS Options form to only those in a particular library or set of libraries, do the following:

1. In the *Library* field, type a filtering string.
2. Click *Limit*.

   Only those design libraries that match the filtering string are displayed in the *MTS Options* form.

To limit the view of design components on the *MTS Options* form to a particular cell or set of cells, do the following:

1. In the *Cell* field, type a filtering string.
2. Click *Limit*.

   Only those cells that match the filtering string are displayed in the MTS Options form.

To limit the view by library name and by cell name, do the following:

1. In the *Library* field, type a filtering string.
2. In the *Cell* field, type a filtering string.
3. Click *Limit*.

   Only those cells that match the filtering string in the *Cell* field contained in those libraries that match the filtering string in the *Library* field are displayed in the MTS Options form.
4. Select the *Show MTS blocks only* check box to display only MTS blocks in the *Cell Table* and *Instance Tree* tabs. The MTS blocks are indicated with a check mark in the *MTS* column of the *Cell Table* and *Instance Tree*.
5. Click the *Reset* button to clear the filtering and show all libraries and cellviews.
6. Click the *Update* button view only the blocks that are selected as MTS blocks.

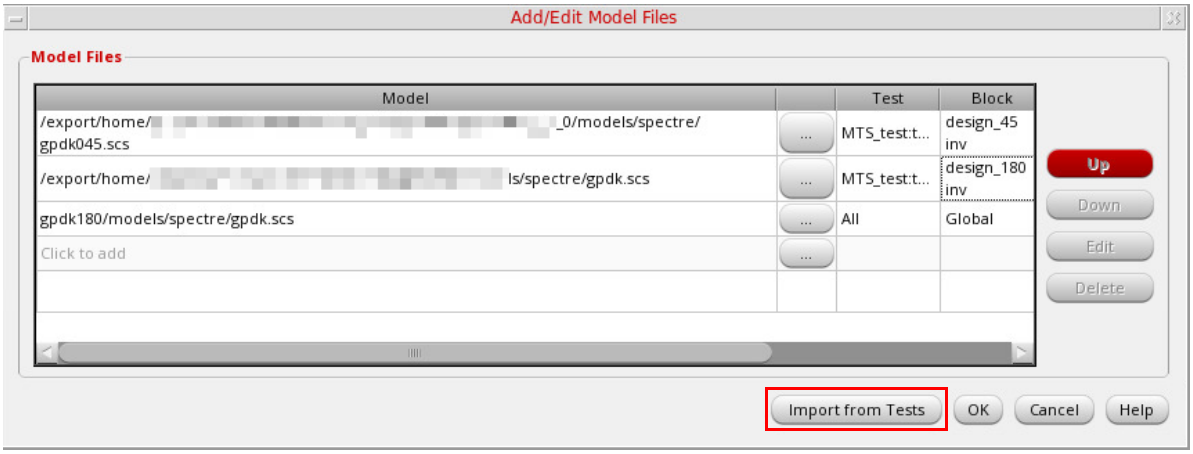
© 2023 Cadence Design Systems, Inc.

Specifying Test and Block Information when Adding Model Files to Corners

To specify a particular test to which you want to apply a model file or a particular section of a model file when adding model files for corners analysis, do the following:

1. Follow the instructions for adding model files to corners but do not click *OK*.

   The model file you added appears on the *Add/Edit Model Files* form.



2. In the *Test* column, double-click and, from the drop-down menu, select a particular test for which you want to use the model file.

   By default, the program uses the model file for all tests.

3. In the *Block* column, double-click and, from the drop-down menu, select a particular cell or instance which you want to apply the model file.

   By default, the program uses the model library for the entire design: *Global*.

4. Click *OK*.

Note: Alternatively, you can click *Import from Tests* in the *Add/Edit Model Files* form to import the models files from your test setup.

© 2023 Cadence Design Systems, Inc.

Detecting Context Symbol/Name Collisions

As a context developer, you need to check for conflicts that arise when you define a function or variable in more than one context (name collisions).

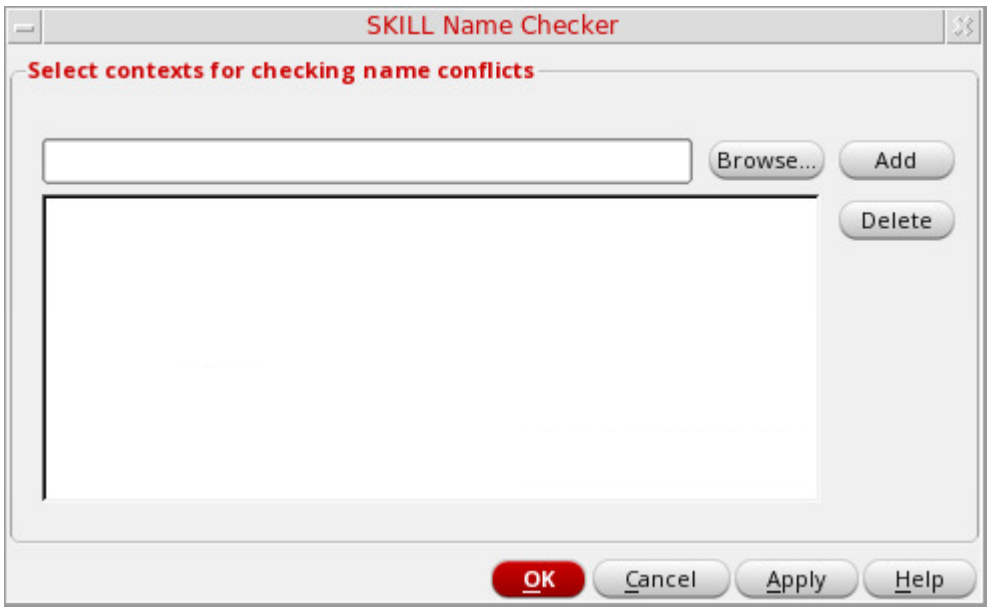
To check for name collisions in loaded contexts, do the following:

1. In the CIW, choose *Tools – Conversion Tool Box*.

   The Conversion Tool Box appears.

2. Click *Check SKILL Name Conflicts*.

   The SKILL Name Checker form appears.



3. Click *Browse*.

   A browser window appears.

4. For each context file you want to add to the check, do the following:

   a. Navigate to and select a valid context file and click *Apply*.

      The file name appears in the field on the SKILL Name Checker form.

   b. On the SKILL Name Checker form, click *Add*.

      The file name moves from the field to the list area.

5. On the SKILL Name Checker form, click *OK*.

© 2023 Cadence Design Systems, Inc.

Duplicate definitions appear in the SKILL Name Checker Results window as follows:

`symbol '`*`functionName`* `already defined by context "`*`contextName`*`"`

where *`functionName`* is the name of the function and *`contextName`* is the first context in which the program found *`functionName`*.

© 2023 Cadence Design Systems, Inc.

Multi-Technology Simulations Using DSPF File

Consider that your design has two blocks, `A` and `B` designed with different process technologies. The block `B` further contains blocks, `C`, `D`, and `E` inside it. You have enabled local scoping at the cell level for the block `B`.

Now, if you want to run a multi-technology simulation only for the sub-block `C` of the block `B`, along with the block `A`, you first need to create a `.scs` file for the block `C` with the following contents:

```
//DSPF File
include "<path to model file>" section=tttt
dspf_include "<path to the spf file of the block C>" extra_port=true bus_delim="<>
[]"
```

You then have to include this `.scs` file instead of the model file in the MTS Options form before running the multi-technology simulation.

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

A

Migrating ADE L or ADE XL Cellviews to ADE Assembler

ADE Assembler provides support to import and reuse the setup from ADE L or ADE XL. The following sections provide details on how to migrate states or cellviews saved using these tools into ADE Assembler:

- Migrating an ADE L State to ADE Assembler
- Migrating an ADE XL Setup to ADE Assembler
- Reusing Menu Customization from ADE XL
- Migrating an Ocean script from ADE XL to ADE Assembler

Video

You can view a video demonstration for this feature at Migrating ADE L/XL Setup to ADE Assembler.

© 2023 Cadence Design Systems, Inc.

Migrating an ADE L State to ADE Assembler

To migrate setup details from ADE L to ADE Assembler, save a state in ADE L, if not already available, and load it in ADE Assembler. You can do this in any of the following ways:

- Loading an ADE L State for a Test in ADE Assembler
- Migrating an ADE L State to ADE Assembler Using SKILL

Loading an ADE L State for a Test in ADE Assembler

To migrate an ADE L state to ADE Assembler, create a placeholder test in the maestro cellview in ADE Assembler and then, perform the following steps:

1. In the Data View pane of the ADE Assembler window, expand the *Tests* tree to view the names of all the tests contained in it.
2. Right-click the test for which you need to migrate an ADE L state and select *Load State*.

© 2023 Cadence Design Systems, Inc.

The Loading State -- ADE Assembler form is displayed, as shown below.

3. Depending on how the state was saved in ADE L, select *Directory* or *Cellview* radio button for *Load State Option*.

© 2023 Cadence Design Systems, Inc.

4. Select the details of the source directory or cellview state in the *Directory Options* or *Cellview Options* group box.
5. In the *What to Save* group box, select the check boxes for the components you want to copy to the test in ADE Assembler.
6. Click *OK*.

All the selected settings are copied to the test in ADE Assembler.

Also see: MTSremoveOldConfigSetupAfterMigration

Migrating an ADE L State to ADE Assembler Using SKILL

You can save an ADE L state as a cellview or a state file, and then migrate it to a maestro cellview using the maeMigrateADELStateToMaestro SKILL function.

The following example code migrates an ADE L state from a cellview state, `spectre_state1`:

```
maeMigrateADELStateToMaestro("Two_Stage_Opamp" "OpAmp_AC_top" "spectre_state1" )

=> ("Two_Stage_Opamp" "OpAmp_AC_top" "maestro")
;; the function returns the library, cell, and view name of the new maestro view
```

In the above example, the default name `maestro` was used to create a new cellview. This cellview will be available in the Library Manager and you can open it in ADE Assembler using the *Open With* command in the pop-up menu.

The following example code migrates an ADE L state from a state file, `AC_state1`, saved in the `./libs/Two_Stage_Opamp/OpAmp/adexl/test_states/` directory:

```
maeMigrateADELStateToMaestro("Two_Stage_Opamp" "OpAmp_AC_top" "AC_state1"
?maestroView "maestro1" ?migrateFrom 'directory ?statePath
"./libs/Two_Stage_Opamp/OpAmp/adexl/test_states/")
=> ("Two_Stage_Opamp" "OpAmp_AC_top" "maestro1")
```

The new cellview, `maestro1`, will be available in the Library Manager.

Related Topics

MTSremoveOldConfigSetupAfterMigration

© 2023 Cadence Design Systems, Inc.

Migrating an ADE XL Setup to ADE Assembler

You can migrate setup details from an ADE XL setup to ADE Assembler in any of the following three ways:

- Opening an ADE XL Cellview in ADE Assembler
- Importing an ADE XL Setup in an Existing Cellview in ADE Assembler
- Loading an ADE XL Setup Using SKILL

Important

If you rename the reference library in Library Manager after migrating an ADE XL cellview to ADE Assembler, ensure that the `maestro.cpupdtr keepHistoryData` environment variable is set to `t`. If this environment variable is set to `nil`, all copied histories are removed from the newly created maestro cellview.

Opening an ADE XL Cellview in ADE Assembler

Note: You can read a blog related to this feature at Virtuosity: Opening Old ADE States and Views with ADE Explorer and ADE Assembler.

When you open an adexl cellview in ADE Assembler, Virtuoso creates a new maestro cellview and migrates the setup. To migrate an adexl cellview into a new maestro cellview, perform the following steps:

1. In the Library Manager window, select the source library and cell to view the list of available views.
2. Right-click the source ADE XL cellview that you want to migrate and choose *Open With*.

   The Migrate ADE XL View form is displayed. The default name for the new cellview is `maestro`.
3. If required, change the name for the cellview.
4. Select *Exclude History* if you do not want to copy run histories from the ADE XL view to the new maestro view in ADE Assembler.
5. Click *OK* to close the Migrate ADE XL View form.

   A new cellview is created and opened in the ADE Assembler environment window. The setup details from the adexl cellview are copied to the maestro cellview and shown in the user interface. The histories and setup states are also copied to the new maestro view.

© 2023 Cadence Design Systems, Inc.

Important

If you are opening a read-only adexl cellview, it will become editable after migration to ADE Assembler.

Also see: MTSremoveOldConfigSetupAfterMigration

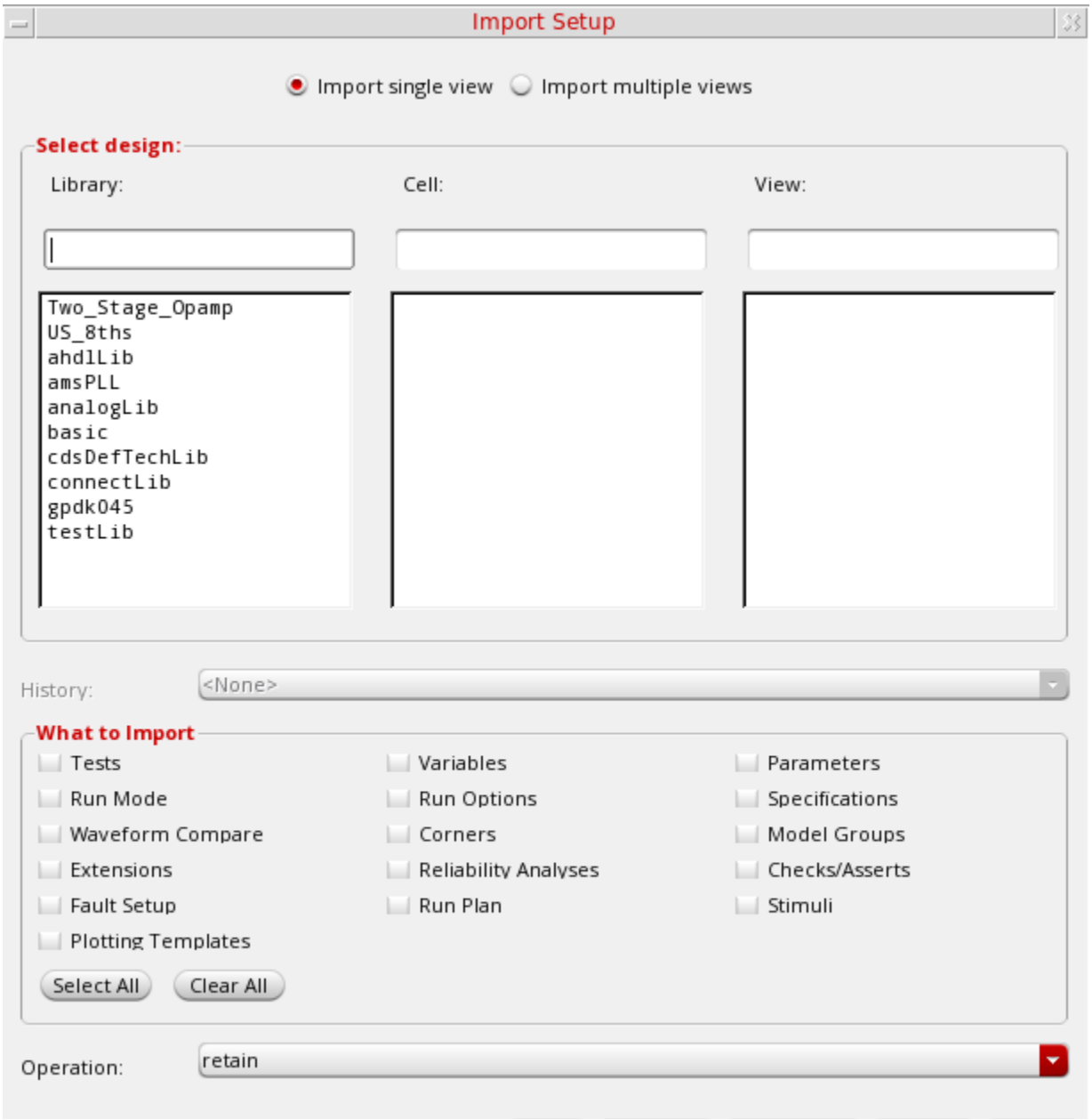
Importing an ADE XL Setup in an Existing Cellview in ADE Assembler

If you have already created a new maestro cellview in ADE Assembler, you can import the setup from an adexl cellview. For this, open the maestro cellview and perform the following steps:

1. In the ADE Assembler environment window, choose *File – Import*.

© 2023 Cadence Design Systems, Inc.

The Import Setup form is displayed.



You can use this form to import setup from one or more ADE XL or ADE Assembler cellviews. Select appropriate options on this form to specify the names of cellviews and the settings to be imported.

2. By default, the *Import single cellview* option is selected. This indicates that setup can be imported from a single cellview selected on the form. If you want to import multiple cellviews, select the *Import multiple cellview* option.

© 2023 Cadence Design Systems, Inc.

3. Select the names of library, cell, and adexl view from which you need to import the setup details.

   The names of histories and states available in the selected adexl view are displayed in the *History* drop-down list. A standard entry, `Active`, points to the active setup details in that view.

4. Select the name of source history or state from the *History* drop-down list.
5. Ensure that the check boxes for the settings you want to import are selected.
6. Depending on whether you want to retain, overwrite, or merge the existing settings of the maestro cellview, select an operation from the *Operation* drop-down list.
7. Click *OK* to import the setup and close the form.

Important

ADE Assembler copies setup states from the source adexl or maestro view only when the setup is imported from a single view. In case of multiple views, setup states are not imported because there could be conflicts due to common names of setup states in different views.

Also see: MTSremoveOldConfigSetupAfterMigration

Loading an ADE XL Setup Using SKILL

You can migrate setup from an adexl view to a maestro cellview using the maeMigrateADEXLToMaestro function in CIW or a SKILL code.

You can execute this function in the following format:

```
maeMigrateADEXLToMaestro("<adexl-library-name>" "<adexl-cell-name>"
"<adexl-view-name>" ?maestroLib "<lib-name>" ?maestroCell "<cell-name>"
?maestroView "<maestro-view-name>")
```

A new maestro cellview is available in Library Manager and can be opened in ADE Assembler. It contains all the setup details, histories, and setup states copied from the adexl view.

Important

ADE Assembler does not provide support for device checks. Therefore, if the imported ADE XL setup contained device checks created using the Device Check Specification form, they will not be loaded in the ADE Assembler setup. You would need to create a new setup using `Checks/Asserts` in the Data View pane. For more details, refer to Performing Circuit and Device Checks.

© 2023 Cadence Design Systems, Inc.

Reusing Menu Customization from ADE XL

ADE Assembler and ADE XL share the same menus. Therefore, menu customization can be copied from ADE XL to ADE Assembler, as described below.

Importing Menu Customization From a Trigger in .cdsinit

If you have defined a trigger in `.cdsinit` to customize ADE XL menus, you can register the same trigger callback for the `assembler` application type.

For example, if you have defined a callback, `CCSadexlExampleMenuTrigger`, to create a custom menu for ADE XL, you can register the same for ADE Assembler by using the correct application name, as shown below.

```
(deRegUserTriggers "assembler" nil 'CCSadexlExampleMenuTrigger)
```


© 2023 Cadence Design Systems, Inc.

Migrating an Ocean script from ADE XL to ADE Assembler

To migrate an Ocean Script from ADE XL to ADE Assembler,

1. Start the Virtuoso Design Environment.

   ```
   virtuoso &
   ```

2. Load the Ocean script generated using ADE XL by using the SKILL `load` command in CIW.

   ```
   load("<name-of-OCEAN-script-file>")
   ```

   Messages about the progress of your script appear in the CIW.

Note the following:

- If you wish to view the results of the Ocean run, in the *history* tab of the *Data View* Assistant, right click the related history and click *View Results*.
- On loading the Ocean script, the related ADE XL view will get migrated to a maestro view and the script will be run. The name of this maestro view will be in the following format.

  ```
  maestro_<ADE-XL-View-Name>_ocean_migrate
  ```

  You can then open the migrated maestro view and view the results of the Ocean run.

© 2023 Cadence Design Systems, Inc.

Job Policy Settings in Migrated Cellviews

When you migrate an ADE L state to a maestro celview, ADE Assembler uses the default job policy because the setup in ADE L is different than that in ADE Assembler.

When you migrate an adexl cellview to a maestro cellview in ADE Assembler, the job policy in ADE XL is used as the job policy for simulation in ADE Assembler, but the job control mode is changed to LSCS, the default mode in ADE Assembler. For netlisting, the maestro cellview uses the default job policy.

The following table describes how the settings are configured in a migrated cellview.

Migrated from	Job policy in the pource cellview	Simulation job policy in the maestro cellview	Netlisting job policy in the maestro cellview
ADE L	No policy	Maestro Default	Netlisting Default
ADE XL	ADE XL Default	Maestro Default	Netlisting Default
	Custom policy	Custom policys	Netlisting Default

© 2023 Cadence Design Systems, Inc.

© 2023 Cadence Design Systems, Inc.

B

ADE Assembler Environment Variables

This appendix provides a list of variables that you can use to control the characteristics of the ADE Assembler environment or the simulations run from this environment.

The environment variables for ADE Assembler have been divided in the following partitions:

- maestro.license
- maestro.assembler
- maestro.caa
- maestro.calcVal
- maestro.commonTestSetup
- maestro.debug
- maestro.distribute
- maestro.faultsimulation
- maestro.gui
- maestro.logging
- maestro.lscs
- maestro.plotting
- maestro.monte
- maestro.results
- maestro.script
- maestro.setupdb
- maestro.simulation
- maestro.sla

© 2023 Cadence Design Systems, Inc.

- maestro.test
- maestro.waveSpec
- maestro.mts
- adexl.algorithm
- adexl.setupdb
- adexl.test
- adexl.simulation
- adexl.distribute
- adexl.monte
- adexl.historyNamePrefix
- adexl.icrpStartup
- adexl.results
- adexl.gui
- adexl.cpupdtr
- adexl.datasheet
- asimenv.startup
- adexl.plotting
- asimenv.plotting
- asimenv

© 2023 Cadence Design Systems, Inc.

maestro.assembler

- defaultOutputColumns
- outputSetupFilteringNumOutputThreshold

© 2023 Cadence Design Systems, Inc.

defaultOutputColumns

```
maestro.assembler defaultOutputColumns string columnNameList
```

Description

Specifies the default list of columns to be shown on the Outputs Setup tab in ADE Assembler.

GUI Equivalent

None

Examples

```
envGetVal("maestro.assembler" "defaultOutputColumns")
```

```
envSetVal("maestro.assembler" "defaultOutputColumns" 'string "\"Test\" \"Name\" \"Type\" \"Details\" \"EvalType\" \"Plot\" \"Save\" \"Spec\" \"Weight\" \"Units\" \"Digits\" \"Notation\" \"Suffix\"")
```

Related Topics

Selecting Output Signals on the Schematic

© 2023 Cadence Design Systems, Inc.

outputSetupFilteringNumOutputThreshold

```
maestro.assembler outputSetupFilteringNumOutputThreshold int 0
```

Description

Specifies the threshold value for the number of rows that should exist in the output setup to show the filters. If the number of rows is more than this threshold, filters are displayed on the Outputs Setup tab.

Possible values: A positive integer value between `0` and `99999`.

When the variable is set to `0`, filters are always displayed. Set this value to any value other than `0` to specify the minimum number of rows on the Output Setup tab for the filters start appearing. Setting this to a large value closer to `99999` does not show filters.

Default value: `0`

GUI Equivalent

None

Examples

```
envGetVal("maestro.assembler" "outputSetupFilteringNumOutputThreshold")
envSetVal("maestro.assembler" "outputSetupFilteringNumOutputThreshold" 'int 0)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

maestro.caa

- enableDochecklimitFromDataView
- exportBatch
- refreshViolationPerCorner

© 2023 Cadence Design Systems, Inc.

enableDochecklimitFromDataView

```
maestro.caa enableDochecklimitFromDataView boolean { t | nil }
```

Description

Automatically synchronizes the *dochecklimit* setting on the *Checks* tab of the Simulator Options form when you change the *Checks/Asserts* setting in the Data View assistant.

Enabling this variable saves an extra step of enabling the *dochecklimit* simulation option, which is mandatory to run the checks and asserts simulations.

Default value is `nil`.

GUI Equivalent

None

Example

```
envGetVal("maestro.caa" "enableDochecklimitFromDataView")
envSetVal("maestro.caa" "enableDochecklimitFromDataView" 'boolean t)
```

Related Topics

Setting up Checks and Asserts Analysis

© 2023 Cadence Design Systems, Inc.

exportBatch

`maestro.caa exportBatch int` *`batchSize`*

Description

Specifies the batch size to be used while exporting the Checks/Asserts results to a CSV file.

Default value is `10000`.

GUI Equivalent

None

Example

```
envGetVal("maestro.caa" "exportBatch")
envSetVal("maestro.caa" "exportBatch" 'int 15000)
```

Related Topics

Exporting Checks/Asserts Results

© 2023 Cadence Design Systems, Inc.

refreshViolationPerCorner

```
maestro.caa refreshViolationPerCorner boolean { t | nil }
```

Description

Specifies whether to refresh the violations in the `Checks/Asserts` results view after the simulation for each corner is complete.

By default, the `Checks/Asserts` results view shows the results status as `disabled` until the simulation is complete for all points and corners. Setting this variable displays the updated violations data incrementally after the results of each corner are available in the database. Set this variable to `t` if you are running simulations for large designs and the simulation for each corner takes a few hours to complete.

Default value is `nil`.

GUI Equivalent

None

Example

```
envGetVal("maestro.caa" "refreshViolationPerCorner")
envSetVal("maestro.caa" "refreshViolationPerCorner" 'boolean t)
```

Related Topics

Running Simulations and Viewing Violations

© 2023 Cadence Design Systems, Inc.

maestro.calcVal

- debug

© 2023 Cadence Design Systems, Inc.

debug

```
maestro.calcVal debug int { 0 | 1 | 2 | 3 | 4 | 5 }
```

Description

Specifies the level of detail to be printed for `calcVal` expressions in the CIW and the log files. Depending on your debug requirements and the information you want read, you can specify an appropriate level.

The various levels set by this environment variable are:

- `0`: At this level, the tool does not print any messages for `calcVal` expressions.
- `1`: This is the default level at which the tool prints a message indicating which `calcVal` expression has incorrect values for arguments. This is the most basic information required to debug and correct the `calcVal` expression. For example, if an expression failed because some of the points did not have values specified for the `?matchParams` argument, the messages show the details of those points.
- `2`, `3`: In addition to the messages printed at level `1`, the tool prints more details for an error. For example, if the values for some points did not match the values specified for `?matchParams`, at level 2 the tool also prints the details of points for which the values match. The additional messages are prefixed with `CALCVAL(2)` or `CALCVAL(3)` indicating the level for which reporting is being done.
- `4`, `5`: These levels are for detailed debugging in special cases. It is recommended to set the variable to these values only when suggested by the Cadence Support team.

Depending on where the `calcVal` expressions are evaluated, you must set this variable in `.cdsinit`, `.vdsinit`, or CIW. For consistency, it is advised to set the same value for this variable in both `.cdsinit` and `.vdsinit`.

GUI Equivalent

None

Example

```
envGetVal("maestro.calcVal" "debug")
envSetVal("maestro.calcVal" "debug" 'int 2)
```


© 2023 Cadence Design Systems, Inc.

Related Topics

calcVal

© 2023 Cadence Design Systems, Inc.

maestro.commonTestSetup

- defaultComponentsdefaultComponents

© 2023 Cadence Design Systems, Inc.

defaultComponents

`maestro.commonTestSetup defaultComponents string` *`comma-separated-componentNames`*

Description

Specifies the default list of components to be displayed in the Multi-Test Editor in ADE Assembler.

Possible value: A string value containing a space-separated list of column names enclosed in double quotes.

Default value: `"Design Variables,adeInfo,analyses,convergence,environmentOptions,faultRules,faults,graphicalStimuli,mdlOptions,modelSetup,mtsSetup,opPoints,simulationFiles,simulatorOptions,subckts,turboOptions"`

GUI Equivalent

None

Example

```
envGetVal("maestro.commonTestSetup" "defaultComponents")
envSetVal("maestro.commonTestSetup" "defaultComponents" 'string
"adeInfo,analyses,convergence,environmentOptions,faultRules,faults,graphicalStimuli,mdlOptions,modelSetup,mtsSetup,opPoints,outputs,simulationFiles,simulatorOptions,subckts,turboOptions,outputs")
```

Related Topics

Editing Multiple Tests in ADE Assembler

© 2023 Cadence Design Systems, Inc.

maestro.debug

- beanstalk
- mainVirtuoso
- netlisting
- simAndEval

© 2023 Cadence Design Systems, Inc.

beanstalk

`maestro.debug beanstalk int` *`debugLevel`*

Description

Specifies the level of detail to be printed to the beanstalk monitor logs for simulations that are run using LSCS job control mode. Depending on your debug requirements and the information you want to read, you can specify an appropriate level.

- `0`: This is the default level at which the tool prints basic error, information, or warning messages. No debugging information is displayed.
- `1`: At this level, the tool prints some information required for debugging, which includes the timing information along with the basic messages. For example, the start and end time of a beanstalk process. There is no impact on the performance of the tool.
- `2`: In addition to the information printed at level 1, the tool prints the entire user-level debugging information for an error. This also includes the status change messages for a process. There might be some impact on the performance of the tool.
- `3`: At this level, the beanstalk monitor is enabled and the tool prints detailed debugging information suitable for advanced users. There might be significant impact on the performance of the tool.

To avoid any performance impact, it is recommended to reset this variable to `0` once you have finished debugging the simulations.

GUI Equivalent

Command:	*Tools – Debug*
Field:	*Enable verbose logging in job logs and CDS log*

Examples

```
envGetVal("maestro.debug" "beanstalk")
envSetVal("maestro.debug" "beanstalk" 'int 1)
```

Related Topics

mainVirtuoso

netlisting

© 2023 Cadence Design Systems, Inc.

simAndEval

Debugging with ADE Diagnostics

© 2023 Cadence Design Systems, Inc.

mainVirtuoso

`maestro.debug mainVirtuoso int` *`debugLevel`*

Description

Specifies the level of detail to be printed to the Virtuoso logs for simulations that are run using LSCS job control mode. Depending on your debug requirements and the information you want to read, you can specify an appropriate level.

- `0`: This is the default level at which the tool prints basic error, information, or warning messages. No debugging information is displayed.
- `1`: Prints some information required for debugging, which includes the timing information along with the basic messages. For example, the start and end time of a Virtuoso process. There is no impact on the performance of the tool.
- `2`: In addition to the information printed at level 1, the tool prints the entire user-level debugging information for an error. This also includes the status change messages for a process. There might be some impact on the performance of the tool.
- `3`: At this level, the tool prints detailed debugging information suitable for advanced users. There might be significant impact on the performance of the tool.

To avoid any performance impact, it is recommended to reset this variable to `0` once you have finished debugging the simulations.

GUI Equivalent

Command:	*Tools – Debug*
Field:	*Enable verbose logging in job logs and CDS log*

Examples

```
envGetVal("maestro.debug" "mainVirtuoso")
envSetVal("maestro.debug" "mainVirtuoso" 'int 1)
```

Related Topics

beanstalk

netlisting

© 2023 Cadence Design Systems, Inc.

simAndEval

Debugging with ADE Diagnostics

© 2023 Cadence Design Systems, Inc.

netlisting

`maestro.debug netlisting int` *`debugLevel`*

Description

Specifies the level of detail to be printed to the netlister logs for simulations that are run using LSCS job control mode. Depending on your debug requirements and the information you want to read, you can specify an appropriate level.

- `0`: This is the default level at which the tool prints basic error, information, or warning messages. No debugging information is displayed.
- `1`: Prints some information required for debugging, which includes the timing information along with the basic messages. For example, the start and end time of a netlisting process. There is no impact on the performance of the tool.
- `2`: In addition to the information printed at level 1, the tool prints the entire user-level debugging information for an error. This also includes the status change messages for a process. There might be some impact on the performance of the tool.
- `3`: At this level, the tool prints detailed debugging information suitable for advanced users. There might be significant impact on the performance of the tool.

To avoid any performance impact, it is recommended to reset this variable to `0` once you have finished debugging the simulations.

GUI Equivalent

Command:	*Tools – Debug*
Field:	*Enable verbose logging in job logs and CDS log*

Examples

```
envGetVal("maestro.debug" "netlisting")
envSetVal("maestro.debug" "netlisting" 'int 1)
```

Related Topics

beanstalk

mainVirtuoso

© 2023 Cadence Design Systems, Inc.

simAndEval

Debugging with ADE Diagnostics

© 2023 Cadence Design Systems, Inc.

simAndEval

`maestro.debug simAndEval int` *`debugLevel`*

Description

Specifies the level of detail to be printed to simulation job logs and expression evaluator logs for simulations that are run using LSCS job control mode. Depending on your debug requirements and the information you want to read, you can specify an appropriate level.

- `0`: This is the default level at which the tool prints basic error, information, or warning messages. No debugging information is displayed.
- `1`: Prints some information required for debugging, which includes the timing information along with the basic messages. For example, the start and end time of a netlisting process. There is no impact on the performance of the tool.
- `2`: In addition to the information printed at level 1, the tool prints the entire user-level debugging information for an error. This also includes the status change messages for a process. There might be some impact on the performance of the tool.
- `3`: At this level, the tool prints detailed debugging information suitable for advanced users. There might be significant impact on the performance of the tool.

To avoid any performance impact, it is recommended to reset this variable to `0` once you have finished debugging the simulations.

GUI Equivalent

Command:	*Tools – Debug*
Field:	*Enable verbose logging in job logs and CDS log*

Examples

```
envGetVal("maestro.debug" "simAndEval")
envSetVal("maestro.debug" "simAndEval" 'int 1)
```

Related Topics

beanstalk

mainVirtuoso

© 2023 Cadence Design Systems, Inc.

netlisting

Debugging with ADE Diagnostics

© 2023 Cadence Design Systems, Inc.

maestro.distribute

- commandJobTerminateTimeoutBeforeKill
- continueICRPNumRetriesToWaitForGUIRecoveredResDB
- inferCommandStatusFromProxy
- maxNetlistingJobsForLocal
- showErrorForNonExistingJobPolicy

© 2023 Cadence Design Systems, Inc.

jobCleanupTimeoutBeforeKill

`maestro.distribute jobCleanup TimeoutBeforeKill int` *`timeSeconds`*

Description

Specifies the time for which ADE Explorer or ADE Assembler should wait before killing an ICRP job when a connection with an ICRP is lost or after 20 seconds have elapsed since it sent a command to an ICRP to clean up and exit.

By default, there is no wait time, and ADE Explorer or ADE Assembler kills the job immediately after the events mentioned above occur. When LSF is used as the job distribution utility, this may result in an incorrect reporting of status of the LSF job containing the ICRP as `"killed by owner"` whereas the process might have exited successfully. This is because in addition to the ICRP, an LSF job contains some more services, such as cdsNameServer, clsbd, osFSLockD and cdsXvnc. If the exit of any of these services remains pending before ADE Explorer or ADE Assembler kills the job, the status is returned as `"killed by owner"`.

To avoid this incorrect reporting of status, set this variable to a positive value between 1 and 3600 seconds. ADE Explorer or ADE Assembler will wait for the specified time before killing an ICRP job.

Default value: `0`

GUI Equivalent

None

Examples

```
envGetVal("maestro.distribute" "jobCleanup TimeoutBeforeKill")
envSetVal("maestro.distribute" "jobCleanupTimeoutBeforeKill" 'int 20)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

commandJobTerminateTimeoutBeforeKill

`maestro.distribute commandJobTerminateTimeoutBeforeKill int` *`timeSeconds`*

Description

Specifies the time in seconds for which ADE Assembler or ADE Explorer should wait between sending the signal `TERM` and sending the signal `KILL` when it decides to kill a command job. The signals `TERM` and `KILL` are operating system level signals.

Suppose you are using LSF and specify a command such as `bsub -K` or `bsub -Is`. ADE Assembler or ADE Explorer kills the command job by first sending the signal `TERM` to the running job. It then sends the signal `KILL` to ensure that the job is killed. You can use `commandJobTerminateTimeoutBeforeKill` to configure this time interval. For example, if you configure the timeout to `5` secs, ADE Assembler or ADE Explorer will send signal `TERM` and wait for `5` secs before sending the signal `KILL`. This prevents the premature kill of the job by the signal `KILL` when it is handling the signal `TERM`. A premature kill leads to the LSF command job reporting a status `"Exits with exit code <number>"` instead of `"<job>_killed by owner"`.

To avoid this incorrect reporting of status, set this variable to a positive value between `1` and `3600` seconds. ADE Explorer or ADE Assembler will wait for the specified time before sending the signal `KILL` to kill a command job.

Default value: `1`

GUI Equivalent

None

Examples

```
envGetVal("maestro.distribute" "commandJobTerminateTimeoutBeforeKill")
envSetVal("maestro.distribute" " commandJobTerminateTimeoutBeforeKill" 'int 1)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

continueICRPNumRetriesToWaitForGUIRecoveredResDB

`maestro.distribute continueICRPNumRetriesToWaitForGUIRecoveredResDB int` *`numTimes`*

Description

Specifies the number of times ADE Assembler checks for a notification about the successful copy of data from /tmp to NFS before continuing with the simulation after Virtuoso closes abruptly.

When `continueICRPRunOnAbruptGUIExit` is set to `t`, ADE Assembler continues the simulation run even if Virtuoso closes abruptly or unexpectedly. Before continuing the simulation, ADE Assembler copies data from /tmp to NFS and waits for the copy to complete. If the copy of data to NFS is complete within the number of times specified by `continueICRPNumRetriesToWaitForGUIRecoveredResDB` (at an interval of one second), the simulation is continued. If the copy of data to NFS is not complete within the given number of retry checks, the simulation is stopped.

Note: This variable is applicable only when RDBCreatePolicy is set to `"NFSDisk"`.

Possible Values: A positive integer value.

Default value: `20`

GUI Equivalent

None

Examples

```
envGetVal("maestro.distribute" "continueICRPNumRetriesToWaitForGUIRecoveredResDB"
)
envSetVal("maestro.distribute" "continueICRPNumRetriesToWaitForGUIRecoveredResDB"
'int 20)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

inferCommandStatusFromProxy

```
maestro.distribute inferCommandStatusFromProxy cyclic {"Always" | "Never" |
     "GuessFromCommand"}
```

Description

Specifies whether ADE Explorer or ADE Assembler should consider the jobs for ICRP or LSCS job control modes to be interactive or not so as to infer their status from a local shell or a proxy process.

Possible values:

- `"Always"` : Specifies that the job is always interactive. Use this value when you are sure that every job would be interactive because if it is not, job distribution might not work correctly.
- `"Never"` : Specifies that the job is never interactive.
- `"GuessFromCommand"` : The tool will treat the jobs to be interactive only when interactive flags or commands are given. To know about the interactive flags that you can use, refer to Specifying a Command for DRMS.

The tool guesses the known interactive flags or commands only for LSF, SGE and Network Computer. If you have any other DRMS, The tool will not be able to understand whether the jobs are interactive or not. In such case, set this variable to "Always".

Default Value: `"GuessFromCommand"`

GUI Equivalent

None

Examples

```
envGetVal("maestro.distribute" "inferCommandStatusFromProxy")
envSetVal("maestro.distribute" "inferCommandStatusFromProxy" 'cyclic "Never")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

maxNetlistingJobsForLocal

```
maestro.distrubute maxNetlistingJobsForLocal int maxJobs
```

Description

Specifies the maximum number of jobs allowed to be set for the netlisting services to be used when the distribution method is set to `Local` or `Remote-Host`. A warning message is displayed if you set a value higher than the value specified by this variable in the *Max Jobs* field on the *Netlisting* tab.

Note: This variable is used only for the `LSCS` job control mode.

Possible values: A positive integer withing the range of 1 and 500 sets the maximum number of jobs allowed to be set for the netlisting services.

Default value: `50`

GUI Equivalent

None

Example

```
envGetVal("maestro.distrubute" "maxNetlistingJobsForLocal int 50")
envSetVal("maestro.distrubute" "maxNetlistingJobsForLocal" 'int 50)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

showErrorForNonExistingJobPolicy

```
maestro.distribute showErrorForNonExistingJobPolicy boolean { t | nil }
```

Description

Specifies whether to show an error message when the user-specified job policy is not found. The default is `t`, which means that an error message is displayed.

GUI Equivalent

None

Examples

```
envGetval("maestro.distribute" "showErrorForNonExistingJobPolicy")
envSetVal("maestro.distribute" "showErrorForNonExistingJobPolicy" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

maestro.faultsimulation

- defaultConfidenceLevel
- defaultBridgeFaultRes
- defaultOpenFaultRes
- defaultStuckAtFaultRes
- dropUndetectedSimErrorsInFaultDroppingFlow
- handleErrorsAs
- handleSimulationErrorsAs
- suppressAddToFaultGroupsPrompt
- warnWhenSimsExceed

© 2023 Cadence Design Systems, Inc.

defaultConfidenceLevel

`maestro.faultsimulation defaultConfidenceLevel float` *`defaultConfidenceLevel`*

Description

Specifies the default resistance for bridge faults. This value is displayed in the *Confidence Level* field of the Sampling and Stop Criteria Settings form.

GUI Equivalent

None

Examples

```
envGetval("maestro.faultsimulation" "defaultConfidenceLevel")
envSetVal("maestro.faultsimulation" "defaultConfidenceLevel" 'float "97")
```

Related Topics

Specifying Fault Sampling Options and Auto Stop Criteria

© 2023 Cadence Design Systems, Inc.

defaultBridgeFaultRes

`maestro.faultsimulation defaultBridgeFaultRes string` *`t_defaultResValue`*

Description

Specifies the default resistance for bridge faults. This value is displayed in the *Bridge Resistance (ohm)* field of the Fault Rules form.

GUI Equivalent

None

Examples

```
envGetval("maestro.faultsimulation" "defaultBridgeFaultRes")
envSetVal("maestro.faultsimulation" "defaultBridgeFaultRes" 'string "10")
```

Related Topics

Fault Rules Form

© 2023 Cadence Design Systems, Inc.

defaultOpenFaultRes

`maestro.faultsimulation defaultOpenFaultRes string` *`t_defaultResValue`*

Description

Specifies the default resistance for open faults. This value is displayed in the *Open Resistance (ohm)* field of the Fault Rules form.

GUI Equivalent

None

Examples

```
envGetval("maestro.faultsimulation" "defaultOpenFaultRes")
envSetVal("maestro.faultsimulation" "defaultOpenFaultRes" 'string "1G")
```

Related Topics

Fault Rules Form

© 2023 Cadence Design Systems, Inc.

defaultStuckAtFaultRes

`maestro.faultsimulation defaultStuckAtFaultRes string` *`t_defaultResValue`*

Description

Specifies the default resistance for stuck-at faults. This value is displayed in the *Bridge Resistance (ohm)* field of the Fault Rules form.

GUI Equivalent

None

Examples

```
envGetval("maestro.faultsimulation" "defaultStuckAtFaultRes")
envSetVal("maestro.faultsimulation" "defaultStuckAtFaultRes" 'string "100")
```

Related Topics

Fault Rules Form

© 2023 Cadence Design Systems, Inc.

dropUndetectedSimErrorsInFaultDroppingFlow

```
maestro.faultsimulation dropUndetectedSimErrorsInFaultDroppingFlow boolean { t |
     nil }
```

Description

Specifies how to handle simulation errors in the fault dropping flow while merging results of multiple fault simulations. By default, all undetected simulation errors are dropped and not merged.

GUI Equivalent

None

Examples

```
envGetval("maestro.faultsimulation" "dropUndetectedSimErrorsInFaultDroppingFlow")
envSetVal("maestro.faultsimulation" "dropUndetectedSimErrorsInFaultDroppingFlow"
'boolean t)
```

Related Topics

Merging Results from Multiple Fault Simulations

handleSimulationErrorsAs

© 2023 Cadence Design Systems, Inc.

handleErrorsAs

```
maestro.faultsimulation handleErrorsAs cyclic { "undetected" | "detected" |
     "ignore" }
```

Description

Specifies how to consider all fault expression evaluation errors in fault simulation results. By default, they are considered as undetected.

Possible value:

- `undetected`: All fault expression evaluation errors are included in the count of undetected faults.
- `detected`: All fault expression evaluation errors are included in the count of detected faults.
- `ignore`: All fault expression evaluation errors are ignored. They are not included in the count of detected or undetected errors.

GUI Equivalent

None

Examples

```
envGetval("maestro.faultsimulation" "handleErrorsAs")
envSetVal("maestro.faultsimulation" "handleErrorsAs" 'cyclic "undetected")
```

Related Topics

Viewing Summary of Fault Simulation Results

© 2023 Cadence Design Systems, Inc.

handleSimulationErrorsAs

```
maestro.faultsimulation handleSimulationErrorsAs cyclic { "undetected" |
     "detected" | "ignore" }
```

Description

Specifies how to handle faults that have simulation errors caused by convergence issues. By default, they are considered as undetected.

Possible value:

- `undetected`: All simulation errors caused by convergence issues are included in the count of undetected faults. This is the default value.
- `detected`: All simulation errors caused by convergence issues are included in the count of detected faults.
- `ignore`: All simulation errors caused by convergence issue are ignored. They are not included in the count of detected or undetected errors.

GUI Equivalent

None

Examples

```
envGetval("maestro.faultsimulation" "handleSimulationErrorsAs")
envSetVal("maestro.faultsimulation" "handleSimulationErrorsAs" 'cyclic
"undetected")
```

Related Topics

Viewing Summary of Fault Simulation Results

dropUndetectedSimErrorsInFaultDroppingFlow

© 2023 Cadence Design Systems, Inc.

suppressAddToFaultGroupsPrompt

```
maestro.faultsimulation suppressAddToFaultGroupsPrompt boolean { t | nil }
```

Description

Controls the display of a message box before adding a fault rule to a fault group if no fault list is generated for the selected fault rule or its fault list is not up to date.

Possible value are:

- `t`: Shows a message box when the fault list is not generated for the fault rule or it is not up to date.
- `nil`: Does not show any message box when the fault list is not generated or is not up to date. The tool always generates the fault list for the specified fault rule.

Default Value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.faultsimulation" "suppressAddToFaultGroupsPrompt")
envSetVal("maestro.faultsimulation" "suppressAddToFaultGroupsPrompt" 'boolean
nil)
```

Related Topics

Adding Faults to Fault Groups

© 2023 Cadence Design Systems, Inc.

warnWhenSimsExceed

`maestro.faultsimulation warnWhenSimsExceed int` *`maxSim`*

Description

Controls the maximum number of simulations allowed for the Fault Simulation run mode. If the number of simulations exceed this number, ADE Assembler shows an error.

Possible value is a positive integer within the range of `0` to `1000000`.

Default Value: `8000`

GUI Equivalent

None

Examples

```
envGetVal("maestro.faultsimulation" "warnWhenSimsExceed")
envSetVal("maestro.faultsimulation" "warnWhenSimsExceed" 'int 8000)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

maestro.gui

- alwaysEnableSaveButton
- autoExpandSetupStatesSubTree
- copytoRunPlanFromActiveSetup
- displayAgingModelFiles
- defaultAgingPlacementType
- defaultAgingModelFiles
- displayFixedTestVariables
- displayImplicitSignals
- defaultReliabilityMode
- detailTransposeViewShowTest
- enableManualRefresh
- enableWCCfromRunPlanInActiveSetup
- envVarCategoriesForDump
- hideStringResultsFailures
- ignoreLockHistoryForReliability
- manualResultsViewRefresh
- openExplorerOnTestDoubleClicked
- removeDuplicateCornersInResultMerge
- showCapacitanceView
- showOverriddenValueForSpec
- showHistoryPrefixToolbar
- showRedundantPointsinDT

© 2023 Cadence Design Systems, Inc.

alwaysEnableSaveButton

```
maestro.gui alwaysEnableSaveButton boolean { t | nil }
```

Description

By default, whenever there are unsaved changes, maestro cellviews have the *Save* button enabled in edit mode. However, in certain design management systems, you might observe that this button is not enabled in edit mode. In such scenarios, you can set this environment variable to `t` to always keep the *Save* button enabled.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "alwaysEnableSaveButton")
envSetVal("maestro.gui" "alwaysEnableSaveButton" 'boolean t)
```


© 2023 Cadence Design Systems, Inc.

autoExpandSetupStatesSubTree

```
maestro.gui autoExpandSetupStatesSubTree boolean { t | nil }
```

Description

Controls whether the *Setup States* tree in the Data View assistant of ADE Assembler is expanded while applying a filter.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "autoExpandSetupStatesSubTree")
envSetVal("maestro.gui" "autoExpandSetupStatesSubTree" 'boolean t)
```


© 2023 Cadence Design Systems, Inc.

copytoRunPlanFromActiveSetup

```
maestro.gui copytoRunPlanFromActiveSetup boolean { t | nil }
```

Description

Enables or disables the *Copy to Run Plan* command in the shortcut menu of global variables, design variables, parameters, and corners in the Data View assistant. You can use the *Copy to Run Plan* command to copy the selected objects from the active setup to one or all runs in the run plan.

The default value is `t`.

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "copytoRunPlanFromActiveSetup")
```

```
envSetVal("maestro.gui" "copytoRunPlanFromActiveSetup" 'boolean nil)
```

Related Topics

Modifying a Run in the Run Plan

© 2023 Cadence Design Systems, Inc.

displayAgingModelFiles

```
maestro.gui displayAgingModelFiles boolean { t | nil }
```

Description

Shows or hides *Aging Model Files* on the Reliability Analysis Setup form.

Note: The variable is effective only when you set it in `.cdsinit`.

Possible values are:

- `t`: Shows the *Aging Model Files* command on the Reliability Analysis Setup form.
- `nil`: Hides the *Aging Model Files* command on the Reliability Analysis Setup form.

GUI Equivalent

Aging Model Files on the Reliability Analysis Setup form

Examples

```
envGetVal("maestro.gui" "displayAgingModelFiles")
envSetVal("maestro.gui" "displayAgingModelFiles" boolean t)
```

Related Topics

Specifying the Reliability Analysis Setup

© 2023 Cadence Design Systems, Inc.

defaultAgingPlacementType

```
maestro.gui defaultAgingPlacementType cyclic { "Before Standard Model" | " After
     Standard Model" }
```

Description

Determines whether to print the aging-specific model files before or after the standard model files in the netlist.

Possible values:

- `Before Standard Model`: Prints the aging model file before the standard model file in the netlist.
- `After Standard Model`: Prints the aging model file after the standard model file in the netlist.

Default value: `After Standard Model`

GUI Equivalent

Aging/Self-Heating Model Setup form – *Placement of the Model Files in Netlist*

Examples

```
envGetVal("maestro.gui" "defaultAgingPlacementType" cyclic)
envSetVal("maestro.gui" "defaultAgingPlacementType" 'cyclic "After Standard
Model")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

defaultAgingModelFiles

```
maestro.gui defaultAgingModelFiles 'string
     list-of-path-to-aging-model-file-and-model-section
```

Description

Specifies the model files to be used as the default aging model files.

Possible values: A string containing the following values:

- Either the absolute path or the relative path to the model files, which you want to use as the default aging model files, are saved
- Model Section (optional)

Default value: `nil`

GUI Equivalent

None

Examples

To set a single model file as the default aging model file:

```
envSetVal("maestro.gui" "defaultAgingModelFiles")
```

To set multiple model files as the default aging model files:

```
envSetVal("maestro.gui" "defaultAgingModelFiles" 'string
"((\"<path_to_the_aging_model_file\"
\"<model_section>\")(\"<path_to_the_aging_model_file\" \"<model_section>\"))")
```


© 2023 Cadence Design Systems, Inc.

displayFixedTestVariables

```
maestro.gui displayFixedTestVariables boolean { t | nil }
```

Description

Controls the display of the fixed design variables in the gray headers on the *Results* tab when the values of the variables are fixed for some tests and swept for others.

Note: This environment variable is effective only when the *Fixed Parameters/Variables* command in the *Configure what is shown in the table* drop-down list on the toolbar of the *Results* tab is not selected.

Possible values are:

- `t`: Shows the values of the fixed design variables on the gray headers of a design point. This is the default value.
- `nil`: Hides the values of the fixed design variables. The variable is shown only in the header of only those design points for which it is swept.

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "displayFixedTestVariables")
envSetVal("maestro.gui" "displayFixedTestVariables" 'boolean nil)
```

Related Topics

Showing Variable and Parameter Values for Design Points on the Results Tab

© 2023 Cadence Design Systems, Inc.

displayImplicitSignals

```
maestro.gui displayImplicitSignals boolean { t | nil }
```

Description

Selects or deselects the *Implicit Signals* command in the *Configure what is shown in the table* drop-down list on the toolbar of the *Outputs Setup* tab.

Acceptable values:

- `t`: Selects the Implicit Signals command in the Configure what is shown in the table drop-down list on the toolbar of the Outputs Setup tab.
- `nil`: Deselects the Implicit Signals command in the Configure what is shown in the table drop-down list on the toolbar of the Output Setup tab.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "displayImplicitSignals")
envSetVal("maestro.gui" "displayImplicitSignals" 'boolean nil)
```


© 2023 Cadence Design Systems, Inc.

displayJobSummaryBar

```
maestro.gui displayJobSummaryBar boolean { t | nil }
```

Description

Controls the default display of job status in the Run Summary assistant.

Acceptable values:

- `t`: Displays the job status of simulations using the Job Summary view.
- `nil`: Displays the job status of simulations using the Job List view. You can manually switch to the Job Summary view.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "displayJobSummaryBar")
envSetVal("maestro.gui" "displayJobSummaryBar" 'boolean t)
```

Related Topics

Job Summary

© 2023 Cadence Design Systems, Inc.

defaultReliabilityMode

```
maestro.gui defaultReliabilityMode 'cyclic { "AgingScenarios"| "Reliability" }
```

Description

Specifies the reliability mode according to which UI fields are displayed on the Corners Setup form.

Possible value:

- `Reliability`: Displays the form fields that specify the test names for the fresh, stress, and aged run, and the variables to be used by each run. This mode is used for reliability analysis.
- `AgingScenarios`: Displays the form fields to specify the test names for the fresh, stress, and aged run, and the corner conditions for stress and aged tests. This mode is used to set up multiple reliability scenarios.

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "defaultReliabilityMode")
```

```
envSetVal("maestro.gui" "defaultReliabilityMode" 'cyclic "AgingScenarios")
```


© 2023 Cadence Design Systems, Inc.

detailTransposeViewShowTest

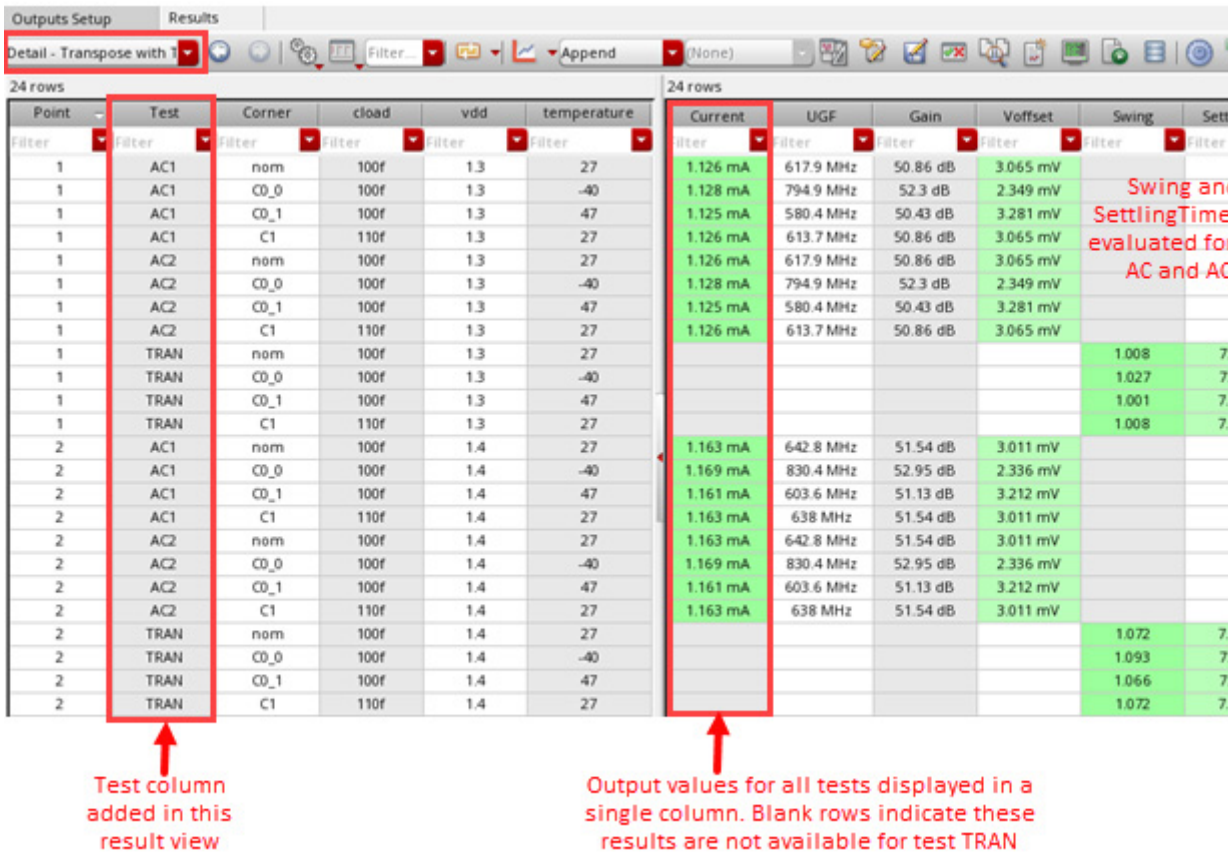
```
maestro.gui detailTransposeViewShowTest boolean { t | nil }
```

Description

Controls whether to show the `Detail - Transpose with Test` result view type in the *Select the results view* drop-down list on the *Results* tab in ADE Assembler. By default, this result view type is hidden. You can display it when you need to see the test name along with results in the `Detail - Transpose` format. The *Detail - Transpose with Test* result view is helpful for the purpose of pivot table creation.

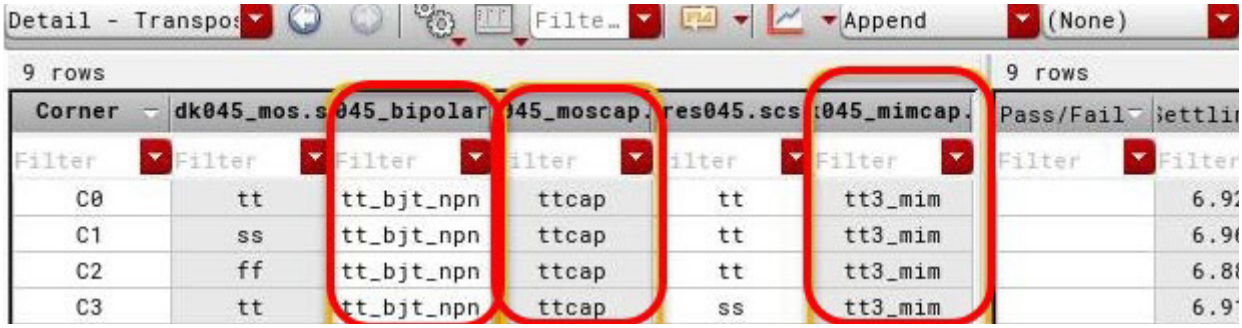
The `Detail - Transpose with Test` view is different from the `Detail - Transpose` view in the following ways:

- The Test column displays the test names.
- All common outputs that are saved for multiple tests are combined in a single column for each output.

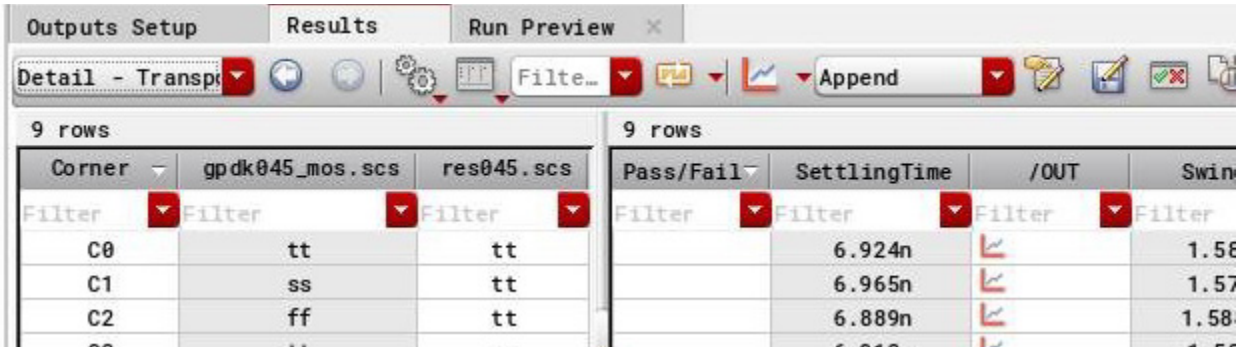


© 2023 Cadence Design Systems, Inc.

- The columns for corner variables that do not vary are hidden. For example, in the `Detail-Transpose` view shown below, values are not varying in the highlighted columns.



The `Detail-Transpose with Test` view hides these columns.



Note: This feature does not work when the nominal corner is enabled for simulation run.

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "detailTransposeViewShowTest")
envSetVal("maestro.gui" "detailTransposeViewShowTest" 'boolean t)
```

Related Topics

Exporting Results in Excel Format for Pivot Tables

© 2023 Cadence Design Systems, Inc.

defaultReliabilityMode

```
maestro.gui removeDuplicateCornersInResultMerge 'boolean { t | nil }
```

Description

Specifies whether to remove duplicate corners created by expanding the corner groups.

Possible value:

- `Reliability`: Displays the form fields that specify the test names for the fresh, stress, and aged run, and the variables to be used by each run. This mode is used for reliability analysis.
- `AgingScenarios`: Displays the form fields to specify the test names for the fresh, stress, and aged run, and the corner conditions for stress and aged tests. This mode is used to set up multiple reliability scenarios.

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "defaultReliabilityMode")
```

```
envSetVal("maestro.gui" "defaultReliabilityMode" 'cyclic "AgingScenarios")
```

Related Topics

Running the Reliability Simulation

© 2023 Cadence Design Systems, Inc.

enableManualRefresh

```
maestro.gui enableManualRefresh boolean { t | nil }
```

Description

Controls the state, enabled or disabled, of the *Manual Refresh* check box in the Run Options form when the check box has been enabled using the manualResultsViewRefresh environment variable.

Possible values are:

- `t`: Enables the *Manual Refresh* check box on the Run Options form
- `nil`: Disables the *Manual Refresh* check box on the Run Options form. This is the default value.

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "enableManualRefresh")
envSetVal("maestro.gui" "enableManualRefresh" 'boolean t)
```

Related Topics

manualResultsViewRefresh

© 2023 Cadence Design Systems, Inc.

enableWCCfromRunPlanInActiveSetup

```
maestro.gui enableWCCfromRunPlanInActive Setup boolean { t | nil }
```

Description

Specifies the default state, enabled or disabled, for the worst case corners copied from the run plan setup to the active setup in ADE Assembler.

Possible values are:

- `t`: Enables the worst case corners copied from the run plan setup to the active setup.
- `nil`: Enables the worst case corners copied from the run plan setup to the active setup.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "enableWCCfromRunPlanInActiveSetup")
envSetVal("maestro.gui" "enableWCCfromRunPlanInActiveSetup" 'boolean t)
```

Related Topics

Setting Up a Run Plan with the Worst Case Corners Run Mode

© 2023 Cadence Design Systems, Inc.

envVarCategoriesForDump

`maestro.gui envVarCategoriesForDump string` *`list-of-variable-categories`*

Description

Specifies a comma-separated list of product categories for which you want to write the current values of environment variables in a file to be used while debugging simulations. Only those variables that have been changed from their default values are written to the file.

By default, this variable is set to `"asimenv,auCore,adexl,maestro"`.

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "envVarCategoriesForDump")
envSetVal("maestro.gui" "envVarCategoriesForDump" 'string "asimenv,maestro")
```

Related Topics

Debug Utility Form

© 2023 Cadence Design Systems, Inc.

ignoreLockHistoryForReliability

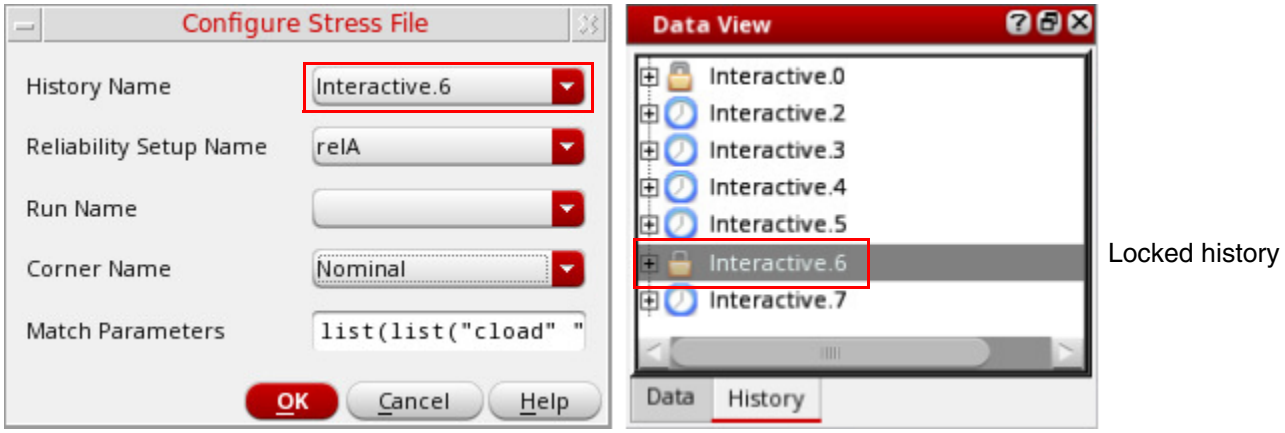
```
maestro.gui ignoreLockHistoryForReliability boolean { t | nil }
```

Description

Specifies whether to lock the history specified in the stress file configuration. You can use the Configure Stress File form to set up the stress file that you want to be reused in running an age simulation.

The default value is `t`, which indicates that the history used in the stress file configuration is not locked.

For example, if a history `Interactive.6` is specified in a stress file configuration and this environment variable is set to `nil`, the history `Interactive.6` is locked so that it cannot be renamed or deleted.



Locked history

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "ignoreLockHistoryForReliability")
```

```
envSetVal("maestro.gui" "ignoreLockHistoryForReliability" 'boolean nil)
```

Related Topics

Reusing the Stress File

© 2023 Cadence Design Systems, Inc.

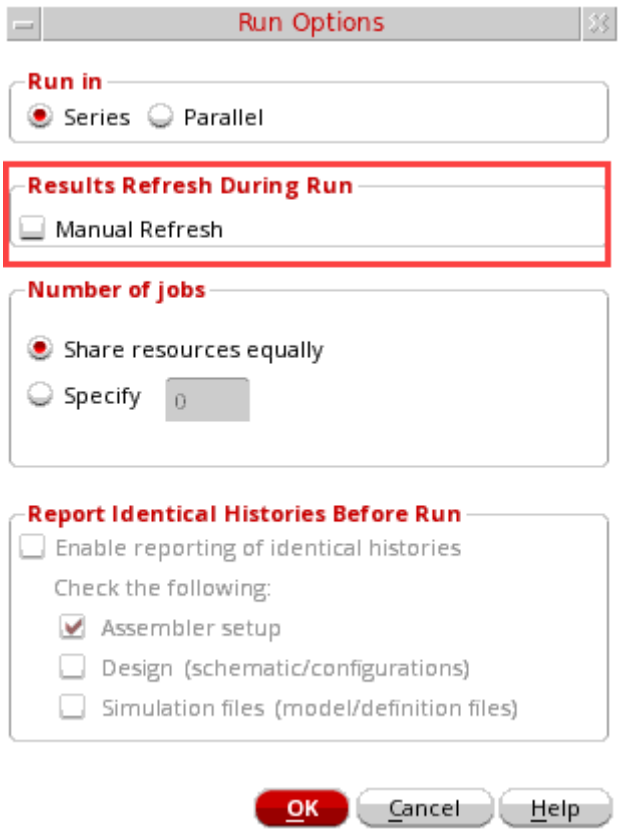
manualResultsViewRefresh

```
maestro.gui manualResultsViewRefresh 'boolean { t | nil }
```

Description

Enables or disables the feature that allows you to control the refresh of the `Yield` view in the *Results* tab while a simulation run is in progress. When the feature is enabled, the *Manual Refresh* check box is added to the Run Options form. You can use this check box to control the display of results on the *Results* tab while a simulation run is in progress, or wait until the run is complete or you manually refresh the tab.

When you select the *Manual Refresh* check box, the *Results* tab is not updated while the run is in progress. You can control the state of the *Manual Refresh* check box by using the enableManualRefresh environment variable.



© 2023 Cadence Design Systems, Inc.

A command, *Refresh to view results*, is added to the toolbar of the `Yield` view in the Results tab. You can use this command to show or update the results at any time during or after the run.

Note: The *Refresh to view results* command is not visible in any other result view.

Possible values:

- `t`: Displays the *Manual Refresh* check box on the Run Options form and sets it the default value, which is selected.
- `nil`: Hides the *Manual Refresh* check box on the Run Options form.

Default value: `nil`

Important

This environment variable works only when set in `.cdsinit`.

GUI Equivalent

None

Example

```
envGetVal("maestro.gui" "manualResultsViewRefresh")
envSetVal("maestro.gui" "manualResultsViewRefresh" 'boolean t)
```

Related Topic

© 2023 Cadence Design Systems, Inc.

openExplorerOnTestDoubleClicked

```
maestro.gui openExplorerOnTestDoubleClicked boolean { t | nil }
```

Description

Controls whether a test opens in ADE Explorer when the test name is double-clicked in the Data View assistant of ADE Assembler.

Possible values:

- `t`: Opens the test for editing in ADE Explorer when you double-click the test name in the Tests tree of the Data View assistant of ADE Assembler.
- `nil`: Does not open the test for editing in ADE Explorer. Instead, it enables renaming of the test.

You can use the down arrow ( ) icon to the left of the test name in the *Tests* tree to open the test in ADE Explorer.

Default value: t

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "openExplorerOnTestDoubleClicked")
envSetVal("maestro.gui" "openExplorerOnTestDoubleClicked" 'boolean t)
```

Related Topic

© 2023 Cadence Design Systems, Inc.

removeDuplicateCornersInResultMerge

```
maestro.gui removeDuplicateCornersInResultMerge boolean { t | nil }
```

Description

Specifies whether to remove duplicate corners from a reference history when conflicting corner names are found in expanded corner groups copied from the master history while merging simulation results.

Possible values are:

- `t`: Removes the corners with duplicate names found in the reference history.
- `nil`: Retains the duplicate corner from the reference history and renames the conflicting corner copied from the master history.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "removeDuplicateCornersInResultMerge")
envSetVal("maestro.gui" "removeDuplicateCornersInResultMerge" 'boolean t)
```

Related Topics

How does ADE Assembler merge variables and corners?

© 2023 Cadence Design Systems, Inc.

showCapacitanceView

```
maestro.gui showCapacitanceView boolean { t | nil }
```

Description

Controls whether to show the *Capacitance* results view in the drop-down list on the *Results* tab.

Possible Values:

- `t`: Shows the Capacitance results view for each history.
- `nil`: Does not show the Capacitance results view.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "showCapacitanceView")
envSetVal("maestro.gui" "showCapacitanceView" 'boolean t)
```

Related Topics

Capacitance results view

© 2023 Cadence Design Systems, Inc.

showOverriddenValueForSpec

```
maestro.gui showOverriddenValueForSpec boolean { t | nil }
```

Description

Controls whether to show the overridden values for specifications in the Detail and Detail - Transpose result views.

Possible values:

- `t`: If the specification values are overridden for corners, the *Spec* column in the results table shows the global spec and the overridden spec value, which is being used to calculate the pass or fail status of an output.
- `nil`: The *Spec* column in the results table shows the text `Corners overridden`, or `Nominal and Corners overridden` but does not show the global spec and the overridden spec value.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "showOverriddenValueForSpec" )
envSetVal("maestro.gui" "showOverriddenValueForSpec" 'boolean t)
```


© 2023 Cadence Design Systems, Inc.

showHistoryPrefixToolbar

```
maestro.gui showHistoryPrefixToolbar boolean { t | nil }
```

Description

Controls whether to show the *History Name* toolbar.

Possible values:

- `t`: Displays the *History Name* toolbar.
- `nil`: Displays the *History Name* toolbar.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "showHistoryPrefixToolbar" )
envSetVal("maestro.gui" "showHistoryPrefixToolbar" 'boolean nil)
```

Related Topics

Specifying a Name for Simulation History

© 2023 Cadence Design Systems, Inc.

showSchematicCrossSelectionHyperlinkInSimLog

```
maestro.gui showSchematicCrossSelectionHyperlinkInSimLog cyclic {"Auto" | "Always"
     | "Never"}
```

Description

Controls whether to show hyperlinks to instances and nets in the Spectre log file saved for a simulation run. You can use this variable to enable, disable, or selectively show hyperlinks depending on the size of the netlist or amap file.

Note: This variable does not affect the hyperlinks to include files that are always visible in the log file.

Possible values are:

- `Auto`: Hyperlinks are not shown in the Spectre log file if the size of the netlist or amap file is greater than 200M.
- `Always`: Hyperlinks are always shown in the Spectre log file. With this setting, you might observe performance issues with large netlist files.
- `Never`: Hyperlinks to instances or nets in the schematic are never shown in the Spectre log file.

Default value: `Auto`

GUI Equivalent

None

Examples

```
envGetVal("maestro.gui" "showSchematicCrossSelectionHyperlinkInSimLog" )
envSetVal("maestro.gui" "showSchematicCrossSelectionHyperlinkInSimLog" 'cyclic
"Älways")
```

Related Topics

Viewing the Simulation Results

© 2023 Cadence Design Systems, Inc.

maestro.logging

- databaseDir
- level
- maxNumberSimulatorNoteWarn

Important

All environment variables in the `maestro.logging` product category must be set in a .cdsenv or .cdsinit file before Virtuoso is launched. They cannot be changed mid-session using the `envSetVal` SKILL API in the CIW.

© 2023 Cadence Design Systems, Inc.

databaseDir

`maestro.logging databaseDir string` *`filepath-to-message-database`*

Description

Specifies the file path to be used to save the message database for the simulations run in LSCS job control mode. By default, this database is saved at the same location as the results database.

Default value: “”

GUI Equivalent

None

Example

```
envGetVal("maestro.logging" "databaseDir")
envSetVal("maestro.logging" "databaseDir" 'string "")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

level

```
maestro.logging level cyclic { "configFile" | "info" | "process" | "warn" | "error"
     }
```

Description

Specifies the severity level of messages to be saved in the message database file in LSCS job control mode. By default, ADE Assembler saves all information, process, warning, and error messages from all the components—netlist service, simulation monitor, Spectre, and vds—involved in running a simulation. For large simulations, saving all these messages can result in a huge file requiring a large disk space. Use this environment variable to save only specific types of messages to help you monitor the simulations. Other types of messages are not saved.

Possible values:

- `configFile`: Saves the messages if their levels match the levels specified in the config file for each component—netlist service, simulation monitor, Spectre, and vds—involved in running a simulation.
- `info`: Saves all information, process, warning, and error messages from all components.
- `process`: Saves only process, warning, and error messages from all components.
- `warn`: Saves only warning and error messages from all components.
- `error`: Saves only error messages from all components.

Default value: `configFile`

GUI Equivalent

None

Example

```
envGetVal("maestro.logging" "level")
envSetVal("maestro.logging" "level" 'cyclic "configFile")
```

Related Topic

© 2023 Cadence Design Systems, Inc.

maxNumberSimulatorNoteWarn

`maestro.logging maxNumberSimulatorNoteWarn int` *`maxNum`*

Description

Specifies the maximum number of information (note) and warning messages to be saved for each simulator. These messages are saved in the message database for LSCS job control mode.

Note: This variable controls only the count of messages reported by simulators, not those from other components—netlist service, simulation monitor, and vds.

Possible value: A positive integer value. When set to 0, no limit is applied and all messages from the simulators are saved.

Default value: `0`

GUI Equivalent

None

Examples

```
envSetVal("maestro.logging" "maxNumberSimulatorNoteWarn")
envSetVal("maestro.logging" "maxNumberSimulatorNoteWarn" 'int 0)
```

Related Topics

Message Viewer.

© 2023 Cadence Design Systems, Inc.

maestro.lscs

- defaultNetlistingJobPolicy
- defaultSimulationJobPolicy
- jobHealthCheckInterval
- preferNetlistingSameTest
- runBeforeSimulation
- useLSCSFor3rdPartySimulators

© 2023 Cadence Design Systems, Inc.

defaultNetlistingJobPolicy

`maestro.lscs defaultNetlistingJobPolicy string` *`jobPolicyName`*

Description

Specifies the default job policy for netlisting jobs in newly created maestro views.

The default value is `Netlisting Default`.

GUI Equivalent

None

Example

```
envGetVal("maestro.lscs" "defaultNetlistingJobPolicy")
envSetVal("maestro.lscs" "defaultNetlistingJobPolicy" 'string
"myNetlistingJobPolicy")
```


© 2023 Cadence Design Systems, Inc.

defaultSimulationJobPolicy

`maestro.lscs defaultSimulationJobPolicy string` *`jobPolicyName`*

Description

Specifies the default job policy for newly created maestro views.

The default value is `Maestro Default`.

GUI Equivalent

None

Example

```
envGetVal("maestro.lscs" "defaultSimulationJobPolicy")
envSetVal("maestro.lscs" "defaultSimulationJobPolicy" 'string "mySimuJobPolicy")
```


© 2023 Cadence Design Systems, Inc.

jobHealthCheckInterval

`maestro.lscs jobHealthCheckInterval integer` *`time-interval`*

Description

Configures the time interval after which an active Simulation Monitor job on a farm must access its corresponding token in the message queue that is maintained by the maestro session that started the job. If the job does not access the token after the specified time, the LSCS job control mode considers the job to be non-responsive and stops this job. It also removes the icon of this job from the Run Summary assistant.

If you set this variable to a value less than or equal to zero, ADE Explorer or ADE Assembler uses the default value, `60` seconds.

Default value: `60`

GUI Equivalent

None

Example

```
envGetVal("maestro.lscs" "jobHealthCheckInterval")
envSetVal("maestro.lscs" "jobHealthCheckInterval" 'integer 100)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

preferNetlistingSameTest

```
maestro.lscs preferNetlistingSameTest boolean { t | nil }
```

Description

Specifies whether a netlister service job should continue to create netlists for more points in the same test instead of picking up points from another test. After the netlist for all the available points of a test is complete, the job can start netlisting points from other tests in the cellview.

If your setup has multiple tests for which netlist creation can be done in parallel by multiple jobs, set this variable to `t`. By preferring to netlist the points from the same test, you can save the time required to reconfigure the job for another test.

If the netlist creation is to be done for a single test at a time, either because there is only one test or the setup for a test contains calcVal expressions dependent on other tests, set this variable to `nil`.

Default value is `t`.

GUI Equivalent

None

Example

```
envSetVal("maestro.lscs" "preferNetlistingSameTest" 'boolean nil)
```


© 2023 Cadence Design Systems, Inc.

runBeforeSimulation

`maestro.lscs runBeforeSimulation string` *`list-of-space-separated-queue-names`*

Description

Specifies a UNIX command or executable to be run by the Simulation Monitor service in LSCS job control mode before running a simulation.

Possible value: A string specifying the complete command to be run.

Default value: “”

Important

You must set this environment variable in the `.cdsenv` or `.cdsinit` file before Virtuoso is launched. You cannot modify it mid-session by using the `envSetVal` SKILL API in the CIW.

GUI Equivalent

None

Examples:

```
envGetVal("maestro.lscs" "runBeforeSimulation")
envSetVal("maestro.lscs" "runBeforeSimulation" 'string "cp netlist ./../../../../
netlist")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

useLSCSFor3rdPartySimulators

```
maestro.lscs useLSCSFor3rdPartySimulators boolean { t | nil }
```

Description

To use a third-party simulator in an LSCS simulation, set this environment variable to `t`. The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("maestro.lscs" "useLSCSFor3rdPartySimulators")
envSetVal("maestro.lscs" "useLSCSFor3rdPartySimulators" 'boolean t)
```


© 2023 Cadence Design Systems, Inc.

maestro.plotting

- reevaluateBeforePlotAll

© 2023 Cadence Design Systems, Inc.

reevaluateBeforePlotAll

```
maestro.script reevaluateBeforePlotAll boolean { t | nil }
```

Description

Specifies whether to re-evaluate the expressions for outputs before running the *Plot All* command.

Possible values are:

- `t`: ADE Assembler re-evaluates all output expressions before running the *Plot All* command. Use this option when you need to reflect the changes in output expressions.
- `nil`: Does not evaluate the output expressions before running the *Plot All* command. Use this option to improve the performance of the *Plot All* command when you are sure that there were no changes in the output expressions or you do not want to see the impact of any change.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("maestro.plotting" "boolean" nil)
envSetVal("maestro.plotting" 'boolean nil)
```

Related Topics

Plotting Across All Corners and Sweeps

© 2023 Cadence Design Systems, Inc.

maestro.monte

reliabilitymode

© 2023 Cadence Design Systems, Inc.

reliabilitymode

```
maestro.monte reliabilitymode cyclic { "singlestress" | "multistress" }
```

Description

Specifies the default value for the *Reliability Mode* drop-down list on the Monte Carlo options form.

Possible values:

- `singlestress`: Sets the value of the *Reliability Mode* drop-down list as `Single Stress (1+N)`.
- `multistress`: Sets value of the *Reliability Mode* drop-down list as `Multi Stress (N+N)`.

Default value: `singlestress`

GUI Equivalent

None

Examples

```
envGetVal("maestro.monte" "reliabilitymode")
envSetVal("maestro.monte" "reliabilitymode" 'cyclic "singlestress")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

maxPointsPerJob

`maestro.monte maxPointsPerJob integer` *`numMaxPointsPerJob`*

Description

Specifies the maximum number of points to be allocated to each job when a Monte Carlo simulation is run in LSCS job control mode. By default, this variable is set to `0` and the points are assigned automatically depending on the number of samples, tests, corners, and jobs. However, you can use this variable to specify the maximum number of points that can be allocated per job depending on the available resources.

Note: This variable is useful only for the new maestro cellviews that use LSCS job control mode. It does not affect the existing cellviews or setups.

Default value: `0`

GUI Equivalent

Monte Carlo options form – *Points per job – Group automatically*

Monte Carlo options form – *Points per job – Max*

Examples

```
envGetVal("maestro.monte" "maxPointsPerJob")
envSetVal("maestro.monte" "maxPointsPerJob" int 10)
```


© 2023 Cadence Design Systems, Inc.

maestro.results

- continueRunOnRdbFatalError
- defaultReliabilityFilterDirectory
- defaultRunStatisticItems
- enableRunStatisticInfoDisplay
- forceRdbStatusEnable
- forceRdbStatusIgnoreList
- forceRdbStatusTo
- keepTmpRdbForReevaluation
- postLayoutToSchematicInRelReport
- RDBCreatePolicy
- RDBTmpDirDiskQuotaLowerLimitInMegabytes
- RDBNFSDiskQuotaLowerLimitInMegabytes
- RDBTmpTouchInterval
- RDBSynchronous
- saveEvaluationErrorResults
- saveFingersForOpParams
- showDeviceCheckCumulativeEndTime
- sumDeviceFingerCurrents
- waiveViolationsByCorner

© 2023 Cadence Design Systems, Inc.

continueRunOnRdbFatalError

```
maestro.results continueRunOnRdbFatalError boolean { t | nil }
```

Description

Lets you continue the simulation run in case of a fatal disk error. By default, ADE Explorer or ADE Assembler stops the run if the disk on which results are saved becomes inaccessible during run. Set this environment variable to `t` if you want to skip the job that failed due to disk issues and continue the run with the remaining jobs.

Possible values are:

- `t`: Skips the job with disk error and continues the simulation run with the remaining jobs. However, in this case, some results that were saved on the disk with issues may not be accessible.
- `nil`: Stops the run in case of a disk error. This is the default value.

GUI Equivalent

None

Examples

```
envGetVal ("maestro.results" "continueRunOnRdbFatalError")
envSetVal ("maestro.results" "continueRunOnRdbFatalError" 'boolean t)
```


© 2023 Cadence Design Systems, Inc.

defaultReliabilityFilterDirectory

`maestro.results defaultReliabilityFilterDirectory string` *`pathToDirectory`*

Description

Specifies the path to the directory where custom reliability report filters are saved. The default value is `"."`, which indicates that custom filters are saved in the `.cadence` directory under the `dfII/maestro/reliability-filters` subdirectory.

GUI Equivalent

None

Example

```
envGetVal ("maestro.results" "defaultReliabilityFilterDirectory")
envSetVal ("maestro.results" "defaultReliabilityFilterDirectory" 'string "/home/
user/customFilter")
```

Related Topics

Creating a Custom Reliability Data Filter

© 2023 Cadence Design Systems, Inc.

defaultRunStatisticItems

`maestro.results defaultRunStatisticItems string` *`space-or-comma-separated-Items`*

Description

Specifies a space- or comma-separated list of information items to be displayed when the *Run Statistics* option is selected in the *Configure what is shown in the table* list in the *Results* tab

This variable can only be used if:

- The job policy mode for the maestro session is LSCS.
- The variable enableRunStatisticInfoDisplay is set to `t`.
- The specified simulator is Spectre or AMS.

Possible values:

- `memory`: Peak memory consumption for Spectre
- `threads`: Number of threads used by Spectre
- `jobId`: LSF job identifier. This value can only be specified if the distribution method for the job policy is LBS.
- `host`: The name of the host that the simulation is running on
- `nodes`: The node number in the Spectre inventory
- `startTime`: Start time of the simulation run
- `endTime`: End time of the simulation run
- `elapsedTime`: Time elapsed time for the Spectre simulation run
- `idinrun`: Simulation point ID in run
- `dspfSubstitution`: Subckt information related to DSPF substitution from the Spectre output file
- `tranIntrinsicElapsedTime`: Transient intrinsic elapsed time for the Spectre simulation run
- `cpuTime`: CPU time reported by Spectre
- `cpuUtil`: CPU utilization reported by Spectre
- `jobLog`: Name of the log file for a job used to run simulation

© 2023 Cadence Design Systems, Inc.

- `serviceLog`: Name of the netlister service log file
- `recommendedThreads`: Number of threads recommended Spectre X or APS for a simulation

Default value:
`memory,threads,recommendedThreads,jobId,host,nodes,startTime,endTime,elapsedTime,tranIntrinsicElapsedTime,cpuTime,cpuUtil`

GUI Equivalent

None

Example

```
envGetVal("maestro.results" "defaultRunStatisticItems")
envSetVal("maestro.results" "defaultRunStatisticItems" 'string
"memory,threads,jobId,host,nodes,startTime,endTime,elapsedTime")
```

Related Topics

Viewing the Run Statistics

enableRunStatisticInfoDisplay

© 2023 Cadence Design Systems, Inc.

enableRunStatisticInfoDisplay

```
maestro.results enableRunStatisticInfoDisplay boolean { t | nil }
```

Description

Adds the *Run Statistics* option to the *Configure what is shown in the table* list on the toolbar of the *Results* tab.

Note: This variable can be used only if the job policy mode for the maestro session is set to LSCS and the specified simulator is Spectre or AMS.

Possible values:

- `t`: The Run Statistics option gets added to the Configure what is shown in the table list on the toolbar of the Results tab.
- `nil`: No new options are added.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "enableRunStatisticInfoDisplay")
envSetVal("maestro.results" "enableRunStatisticInfoDisplay" 'boolean t)
```

Related Topics

defaultRunStatisticItems

© 2023 Cadence Design Systems, Inc.

forceRdbStatusEnable

```
maestro.results forceRdbStatusEnable boolean { t | nil }
```

Description

Shows or hides the *Change Results Status to Done* command in the context-sensitive menu of a history or the point results that have status other than those mentioned by forceRdbStatusIgnoreList.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "forceRdbStatusEnable")
envSetVal("maestro.results" "forceRdbStatusEnable" 'boolean nil)
```

Related Topics

Enabling Re-Evaluation for Unsuccessful Simulations

forceRdbStatusIgnoreList

forceRdbStatusTo

© 2023 Cadence Design Systems, Inc.

forceRdbStatusIgnoreList

`maestro.results forceRdbStatusIgnoreList string` *`comman-separated-list-of-status`*

Description

A comma-separated list of status to be ignored while forcefully setting the status by the *Change Results Status to Done* command.

Default value: `"done, pending, canceled, disabled"`

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "forceRdbStatusIgnoreList")
envSetVal("maestro.results" "forceRdbStatusIgnoreList" 'string "error, sim err, done, pending, canceled")
```

Related Topics

Enabling Re-Evaluation for Unsuccessful Simulations

forceRdbStatusEnable

forceRdbStatusTo

© 2023 Cadence Design Systems, Inc.

forceRdbStatusTo

`maestro.results forceRdbStatusTo cyclic` *`status-to-be-set`*

Description

Specifies the status to which the *Change Results Status to Done* command must change the status of point results that have status other than those specified by forceRdbStatusIgnoreList.

Possible values are `"error"`, `"sim err"`, `"pending"`, `"running"`, `"netlisting"`, `"netlisting finished"`, `"simulating"`, `"simulation finished"`, `"evaluating"`, `"done"`, `"partialDone"`, `"canceled"`, `"suspended"`, `"eval err"`, `"netl err"`, `"upld err"`, `"disabled"`, `"uploading netlist"`, `"netlist uploaded"`.

Default value: `"done"`

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "forceRdbStatusTo")
envSetVal("maestro.results" "forceRdbStatusTo" 'cyclic "evaluated")
```

Related Topics

Enabling Re-Evaluation for Unsuccessful Simulations

forceRdbStatusEnable

forceRdbStatusIgnoreList

© 2023 Cadence Design Systems, Inc.

keepTmpRdbForReevaluation

```
maestro.results keepTmpRdbForReevaluation boolean { t | nil }
```

Description

Specifies whether to retain the results database in the `/tmp` directory on the local host after the simulation run is complete. Retaining a copy in the /tmp directory enables fast data access for the display or reevaluation of results. In case of space constraints for `/tmp` directory, set this variable to `nil` so that the results database at /tmp are removed after the simulation run is complete. This variable is used only when the RDBCreatePolicy variable is set to `"TmpDir"`.

Possible value:

- `t`: After the simulation run is complete, the results database in /tmp directory is copied to the NFS disk. /tmp is retained to enable fast data access for the display or reevaluation of results.
- `nil` : After the simulation run is complete, the results database is copied to the NFS disk and removed from the temporary location. /tmp is immediately removed to free up the space.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "keepTmpRdbForReevaluation")
envSetVal("maestro.results" "keepTmpRdbForReevaluation" 'boolean nil)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

postLayoutToSchematicInRelReport

```
maestro.results postLayoutToSchematicInRelReport boolean { t | nil }
```

Description

Specifies whether to show instance names in post-layout format or in schematic format in the reliability report.

The default is `nil`, which indicates that instance names are shown in post-layout format in the reliability report.

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "postLayoutToSchematicInRelReport")
envSetVal("maestro.results" "postLayoutToSchematicInRelReport" 'boolean nil)
```

Related Topics

Viewing Reliability Report

© 2023 Cadence Design Systems, Inc.

RDBCreatePolicy

```
maestro.results RDBCreatePolicy cyclic {TmpDir | NFSDisk}
```

Description

Specifies a policy that defines the location where ADE Explorer or ADE Assembler needs to save the results database.

Possible values:

- `TmpDir`: Initially, ADE Explorer or ADE ADE Assembler saves the results database at a temporary location (/tmp directory) on the local host for faster read and write operations. After the simulation is complete, it copies the /tmp database to the NFS disk in small blocks at periodic time intervals of 50 seconds so that next time when you reload the history from the maestro view, it is opened from the NFS disk. By default, after the simulation run is complete, the results database created for this run in the /tmp directory is copied to the NFS disk and then immediately removed from /tmp. However, if you want to retain the results in /tmp for fast data access to display or reevaluate results, set the `keepTmpRdbForReevaluation` environment variable to `t`.

  The default temporary location is /tmp, but you can change it by setting the `TMPDIR` shell environment variable. For example, to change the location to tmp2, use the `setenv TMPDIR /tmp2` command before launching Virtuoso or add following line in the `.cdsinit` file:

  ```
  setShellEnvVar("TMPDIR=/tmp2")
  ```

- `NFSDisk` : ADE Explorer or ADE Assembler directly writes the results database on the NFS disk.

Default value: `TmpDir`

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "RDBCreatePolicy")
envSetVal("maestro.results" "RDBCreatePolicy" 'cyclic "TmpDir")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

RDBTmpDirDiskQuotaLowerLimitInMegabytes

```
maestro.results RDBTmpDirDiskQuotaLowerLimitInMegabytes int 2
```

Description

Specifies the minimum disk quota (in Megabytes) to be maintained for the /tmp directory where ADE Explorer or Assembler saves the results for faster read or write operations with the permanent results database.

Possible values: A positive integer value.

Default value: `100`

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "RDBTmpDirDiskQuotaLowerLimitInMegabytes")
envSetVal("maestro.results" "RDBTmpDirDiskQuotaLowerLimitInMegabytes" 'int 5)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

RDBNFSDiskQuotaLowerLimitInMegabytes

```
maestro.results RDBNFSDiskQuotaLowerLimitInMegabytes int 100
```

Description

Specifies the lower limit of NFS disk quota. If the available NFS disk space becomes lower than this limit, ADE Assembler suspends the current simulation and flags a warning message to free up some space and resume simulation.

Possible values: A positive integer

Default value: `100`

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "RDBNFSDiskQuotaLowerLimitInMegabytes")
envSetVal("maestro.results" "RDBNFSDiskQuotaLowerLimitInMegabytes" 'int 100)
```


© 2023 Cadence Design Systems, Inc.

RDBTmpTouchInterval

```
maestro.results RDBTmpTouchInterval int timeInterval
```

Description

Specifies the time interval, in seconds, at which the RDB file located in the /tmp directory must be touched by ADE Explorer or Assembler. This prevents a cleanup script running on the system from accidentally clearing the data in the /tmp file. This option is useful in case a simulation takes a long time and there are infrequent updates in the RDB.

To disable this feature, set the variable to `0`.

Note: This variable must be set only when `maestro.results RDBCreatePolicy` is set to `TmpDir`.

The default value is `3600`.

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "RDBTmpTouchInterval")
envSetVal("maestro.results" "RDBTmpTouchInterval" 'int 3500)
```

Related Topics

RDBCreatePolicy

© 2023 Cadence Design Systems, Inc.

RDBSynchronous

```
maestro.results RDBSynchronous cyclic {"Off" | "Normal" | "Full" | "Extra"}
```

Description

Specifies how the results database of ADE Explorer and ADE Assembler should commit transactions to the disk. The values of this variable define the extent of data safety in exceptional cases when it is not possible to complete the transaction in a normal way. For example, in case of a power failure or insufficient disk space to save data.

Possible values for this variable are:

- `Off`: The RDB file on disk remains in an invalid state for short periods of time due to OS caching of file updates. In case of a failure due to insufficient disk space or power failure, RDB may be left in an unreadable state.

  While reading or writing data, if the tool detects that the RDB file has been damaged or corrupted, it deletes the RDB file because it is unrecoverable.

  This value provides the best performance, but does not offer any protection against data loss in exceptional circumstances that can lead to data loss. If other options are selected, the data on disk will remain safe although there are chances of some data loss. This is the default value.

- `Normal`: The RDB file is kept in a valid state by flushing data to disk after each update. If the flush is unsuccessful, the database is rolled back to its previous state. There is still a small possibility of the RDB becoming unreadable after failure if the OS flushes data to disk in an order that is different than the requested order. This value provides a good balance between data safety and performance.

- `Full`: The RDB file is kept in a valid state even if data is flushed out of order. This value guarantees data safety, but that comes with performance degradation when the RDB is handling large results.

- `Extra`: Provides extra durability in the event of an update that is followed closely by a power loss. Similar to `Full`, this value guarantees data safety, but that comes with performance degradation when the RDB is handling large results.

The values of this variable are same as those used by the `synchronous` PRAGMA in SQLite.

GUI Equivalent

None

© 2023 Cadence Design Systems, Inc.

Examples

```
envGetVal("maestro.results" "RDBSynchronous")
envSetVal("maestro.results" "RDBSynchronous" 'cyclic "Full")
```


© 2023 Cadence Design Systems, Inc.

saveEvaluationErrorResults

```
maestro.results saveEvaluationErrorResults boolean { t | nil }
```

Description

Saves the simulation data for the points with evaluation errors.

The default value is `nil`.

GUI Equivalent

Command:	*Options – Save – Save Options form*
Field:	*Save Data for Eval Errors*

Examples

```
envGetVal("maestro.results" "saveEvaluationErrorResults")
envSetVal("maestro.results" "saveEvaluationErrorResults" 'boolean t)
```

Related Topics

Specifying Simulation Results Directory Location

© 2023 Cadence Design Systems, Inc.

saveFingersForOpParams

```
maestro.results saveFingersForOpParams boolean { t | nil }
```

Description

Enables or disables dumping of all fingers of the selected DSPF device instances in the save statement in netlist. When set to `t`, the tool uses wildcard characters to save the fingers.

Possible values are:

- `t`: The `save` statement in netlist contains wildcard characters to let the tool save the operating points for all fingers of the selected device instances.

  Example: `save i0.MM0*:gm`

- `nil`: The `save` statement in netlist does not contain any wildcard character. The tool saves a single finger of the selected device instance.

  Example: `save i0.MM0:gm`

Default value is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "saveFingersForOpParams")
envSetVal("maestro.results" "saveFingersForOpParams" 'boolean t)
```

Related Topics

Customizing Netlist Creation Using Environment Variable

© 2023 Cadence Design Systems, Inc.

showDeviceCheckCumulativeEndTime

```
maestro.results showDeviceCheckCumulativeEndTime boolean { t | nil }
```

Description

Adds the column *End Time* to the cumulative `Checks/Asserts` results view.

The default is `nil`.

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "showDeviceCheckCumulativeEndTime")
envSetVal("maestro.results" "showDeviceCheckCumulativeEndTime" 'boolean t)
```


© 2023 Cadence Design Systems, Inc.

sumDeviceFingerCurrents

```
maestro.results sumDeviceFingerCurrents boolean { t | nil }
```

Description

Enables or disables dumping of the sum total of terminal currents for all fingers of a multi-fingered device in the `save` statement in netlist.

Possible values are:

- `t`: The save statement in netlist contains the sum total of terminal current of all fingers of a multi-fingered device.
- `nil`: The save statement in netlist contains a separate terminal current value for one of the fingers of a multi-fingered device.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "sumDeviceFingerCurrents")
envSetVal("maestro.results" "sumDeviceFingerCurrents" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

waiveViolationsByCorner

```
maestro.results waiveViolationsByCorner boolean { t | nil }
```

Description

Allows to waive corner-specific violations in the Checks/Asserts results.

Possible values:

- `t`: Waives corner-specific violations.
- `nil`: Waives all relevant violations across all corners and design points.

Default values: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.results" "waiveViolationsByCorner")
envSetVal("maestro.results" "waiveViolationsByCorner" 'boolean nil)
```

Related Topics

Waiving Violations

© 2023 Cadence Design Systems, Inc.

maestro.script

- accessMode
- exitAfterFinished
- writeCallbackForSingleRun

© 2023 Cadence Design Systems, Inc.

accessMode

```
maestro.script accessMode cyclic { "a" | "r" }
```

Description

Defines the access mode to be used in the `maeLoadSetup` function call added to the run plan script saved from the Run Plan assistant in ADE Assembler. The access mode is set using the `?mode` argument of `maeLoadSetup`.

For example, when the value of the `accessMode` variable is set to `"r"`, the command in the saved script has `?mode` set to `"r"`, as shown below.

maeLoadSetup("libName" "cellName" "viewName" ?mode "r")

Possible values:

- `a`: The `?mode` argument of `maeLoadSetup` is set to `"r"`. That is, the setup will loaded in read mode.
- `r`: The `?mode` argument of `maeLoadSetup` is set to `"a"`. That is, the setup will loaded in append or edit mode.

Default value: `a`

GUI Equivalent

None

Examples

```
envGetVal("maestro.script" "accessMode")
envSetVal("maestro.script" "accessMode" 'cyclic "a")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

exitAfterFinished

```
maestro.script exitAfterFinished boolean { t | nil }
```

Description

Specifies if the following commands are to be added to the run plan script saved from the Run Plan assistant:

```
maeWaitUntilDone('All)
exit
```

These commands ensure that the script exits only after all the simulations have finished running.

Note: The commands given above are added only when the writeCallbackForSingleRun environment variable is also set to `t`.

Possible values:

- `t`: The commands mentioned above are to be added to the run plan script.
- `nil`: The commands are not added to the run plan script.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.script" "exitAfterFinished")
envSetVal("maestro.script" "exitAfterFinished" 'boolean t)
```

Related Topics

writeCallbackForSingleRun

© 2023 Cadence Design Systems, Inc.

writeCallbackForSingleRun

```
maestro.script writeCallbackForSingleRun boolean { t | nil }
```

Description

Specifies if a callback procedure needs to be included in the Run Plan script. When this variable is set to `t`, ADE Assembler makes the following changes in the script:

- Defines the `RunFinishedCallback` function that you can customize to include the statements you want to execute after the simulation is run

  ```
  ;; Post-simulation callback.  Called after each run.
  define( RunFinishedCallback(session runID)
  printf("Run ID %L has finished\n" runID)
  ```

- Adds a call to the `maeExportOutputView` function

  ```
  ;; Use maeExportOutputView() or maeWriteDatasheet() in to report the simulation
  results, or maeReadResDB() for custom-processing

  maeExportOutputView( ?fileName simplifyFilename( strcat("./"
  axlGetHistoryName(axlGetRunData(session runID)) ".csv")) ?view
  "Detail-Transpose")
  )
  ```

- Adds the `?callback` argument in the `maeRunSimulation` call

  ```
  maeRunSimulation(?run "Run.0" ?callback "RunFinishedCallback" ?waitUntilDone
  nil
  ```

Default value: `nil`

GUI Equivalent

None

Example

```
envGetVal("maestro.script" "writeCallbackForSingleRun")
envSetVal("maestro.script" "writeCallbackForSingleRun" 'boolean nil)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

maestro.setupdb

- allowLocalCornersInRunPlan
- autoDeleteMalformedLockedSDB
- syncWithSLA

© 2023 Cadence Design Systems, Inc.

allowLocalCornersInRunPlan

```
maestro.setupdb allowLocalCornersInRunPlan boolean { t | nil }
```

Description

Allows or disallows creation of local corners in a run plan.

Possible values:

- `t`: You can create local corners in a run plan.
- `nil`: You cannot create local corners in a run plan. If any local corner already exists in the run plan before you set the variable to `nil`, it is converted to a reference corner if a corner with the same name exits in the active setup. If no corner is found with the same name in the active setup, it is removed from the run plan too.Appropriate messages are displayed in the CIW providing information about these changes.

Note: The Change to Local Corner Command is disabled.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("maestro.setupdb" "allowLocalCornersInRunPlan" )
envSetVal("maestro.setupdb" "allowLocalCornersInRunPlan" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

autoDeleteMalformedLockedSDB

```
maestro.setupdb autoDeleteMalformedLockedSDB boolean { t | nil }
```

Description

Specifies whether to display a warning message when a malformed lock-state file, such as `maestro.sdb.cdslck` and `history.sdb.cdslck` is found.

Possible values are:

- `t`: The locked files are deleted automatically and a warning message is displayed in Virtuoso Studio CIW.
- `nil` : A pop-up message is displayed asking whether to delete the malformed locked files or to continue the operation after canceling the deletion.

The default is `t`, which means that the malformed lock files are deleted automatically without displaying a pop-up message.

GUI Equivalent

None

Examples

```
envGetval("maestro.setupdb" "autoDeleteMalformedLockedSDB")
envSetVal("maestro.setupdb" "autoDeleteMalformedLockedSDB" 'boolean nil)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

syncWithSLA

```
maestro.setupdb syncWithSLA cyclic { "Popup_Sync" | "Popup_NoSync" |
"Sync" | "NoSync" )
```

Description

Specifies the default action to be performed before running simulation in a scenario when the Data View and Setup Library assistants are out of sync.

- `Popup_Sync`: Default. In graphical mode, displays a message to let you choose whether you want to synchronize the Data View and Setup Library assistants before running the simulation. In non-graphical mode, it is considered the same as `Sync`.
- `Popup_NoSync`: In graphical mode, displays a message to let you choose whether you want to synchronize the Data View and Setup Library assistants before running the simulation. In non-graphical mode, it is considered the same as `NoSync`.
- `Sync`: Runs a simulation after synchronizing the Data View and Setup Library assistants without displaying any message.
- `NoSync`: Runs a simulation without synchronizing the Data View and Setup Library assistants and displaying any message.

GUI Equivalent

None

Examples

```
envGetVal("maestro.setupdb" "syncWithSLA")
envSetVal("maestro.setupdb" "syncWithSLA" 'cyclic "Popup_NoSync")
envSetVal("maestro.setupdb" "syncWithSLA" 'cyclic "Sync")
envSetVal("maestro.setupdb" "syncWithSLA" 'cyclic "NoSync")
```

Related Topics

Starting a Simulation

© 2023 Cadence Design Systems, Inc.

maestro.simulation

- allowVariableCheckingDuringConfigSweep
- behaviorOnPcellParamChangedByCallback
- checkDesignInIdenticalHistory
- checkSimFilesInReportIdenticalHistory
- checkSpectreStatus
- dspfNamePrefixSimInfo
- exprEvalMaxTries
- exprEvalMaxTries
- groupRunA
- ignoreAnalysisCheckInResourceEstimation
- ignoreRunModeCheckInResourceEstimation
- interactiveA
- maxRDBSyncWait
- minGroupSizeSplitAcrossIdleJobs
- reportChangesAfterComparingSetupWithHistories
- reportIdenticalHistoryIgnoreElementsList
- retainNetlistLinkForRerun
- reuseNetlistForRerun
- runModesForIdenticalHistoryCheck
- simulationServiceJobPolicy
- tempDirectory
- useReportIdenticalHistory
- useSelfCDFInPostLayoutNetlist
- waitToAcquireROLock
- warnWhenWCCSimsExceed

© 2023 Cadence Design Systems, Inc.

allowVariableCheckingDuringConfigSweep

```
maestro.simulation allowVariableCheckingDuringConfigSweep boolean { t | nil }
```

Description

Specifies whether to run checks to ensure that all design variables used in the simulation are added to all cellviews in config sweeps. Running this check helps in eliminating simulation errors due to missing variables in the swept designs.

Possible values: `t` and `nil`

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "allowVariableCheckingDuringConfigSweep")
envSetVal("maestro.simulation" "allowVariableCheckingDuringConfigSweep" 'boolean
t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

behaviorOnPcellParamChangedByCallback

```
maestro.simulation behaviorOnPcellParamChangedByCallback cyclic { "none" |
     "warning" | "none" }
```

Description

Specifies whether to show an error or warning message when the user-specified Pcell parameter value is out of the valid range returned by the validating CDF callback. By default, the variable is set to "`none`" and incorrect values of Pcell parameters are replaced with valid values during netlisting.

Possible values:

- `error`: Shows an error message for an invalid value of a Pcell parameter. The simulation is stopped so that you can change the parameter value and rerun the simulation.
- `warning`: Shows a warning message for an invalid value of a Pcell parameter. The tool automatically updates the parameter value to make it valid and continues with netlist creation.
- `none`: The tool automatically updates the parameter value to make it and continues with netlist creation. No error or warning message is printed.

Default value: `none`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "behaviorOnPcellParamChangedByCallback" )
envSetVal("maestro.simulation" "behaviorOnPcellParamChangedByCallback" 'cyclic
"none")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

checkDesignInIdenticalHistory

```
maestro.simulation checkDesignInIdenticalHistory boolean { t | nil }
```

Description

Specifies whether to check for any changes in the schematic or config view of the design while looking for identical histories.

Possible values:

- `t`: Enables checking for any changes in the design.
- `nil`: Disables checking for any changes in the design.

Default value: `nil`

GUI Equivalent

Run Options form – *Design (schematic/configurations)*

Examples

```
envGetVal("maestro.simulation" "checkDesignInIdenticalHistory" )
envSetVal("maestro.simulation" "checkDesignInIdenticalHistory" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

checkSimFilesInReportIdenticalHistory

```
maestro.simulation checkSimFilesInReportIdenticalHistory boolean { t | nil }
```

Description

Specifies whether to check for any changes in the simulation files, such as definition file and model file, while looking for identical histories.

Possible values:

- `t`: Enables checking for any changes in the simulation files.
- `nil`: Disables checking for any changes in the simulation files.

Default value: `nil`

GUI Equivalent

Run Options from – *Simulation files*

Examples

```
envGetVal("maestro.simulation" "checkSimFilesInReportIdenticalHistory")
envSetVal("maestro.simulation" "checkSimFilesInReportIdenticalHistory" 'boolean
t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

dspfNamePrefixSimInfo

`maestro.simulation dspfNamePrefixSimInfo string "`*`simulator-name`*`"`

Description

Specifies the simulator whose `namePrefix` property, specified in the *Simulation Information* tab of the Edit CDF form, is considered for adding name prefixes to devices in the netlist. This is useful when you want to match the schematic device names with the DSPF names.

The default is `""`, which means that no simulator-specific `namePrefix` is specified. It is recommended to provide a valid `namePrefix` so that the netlisted devices in the DSPF file have the correct prefix and the simulation does not fail.

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "dspfNamePrefixSimInfo")
envSetVal("maestro.simulation" "dspfNamePrefixSimInfo" 'string "auCDL")
envSetVal("maestro.simulation" "dspfNamePrefixSimInfo" 'string "auLvs")
envSetVal("maestro.simulation" "dspfNamePrefixSimInfo" 'string "hspiceD")
envSetVal("maestro.simulation" "dspfNamePrefixSimInfo" 'string "spectre")
```

Related Topics

Rules for Adding Prefixes to Instance Names

Prefixes in Instance Names in the DSPF File

useSelfCDFInPostLayoutNetlist

© 2023 Cadence Design Systems, Inc.

exprEvalMaxTries

```
maestro.simulation exprEvalMaxTries int 3
```

Description

Specifies the maximum number of tries to launch a new expression evaluator process when the process that was already evaluating an expression stops abruptly.

This variable is used only when the LSCS job control mode is used.

Possible values: A positive integer value.

Default value: `3`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "exprEvalMaxTries")
envSetVal("maestro.simulation" "exprEvalMaxTries" 'int 1)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

endNetlistingImmediatelyOnStop

```
maestro.simulation endNetlistingImmediatelyOnStop boolean { t | nil }
```

Description

Specifies whether to end all netlisting jobs immediately after the simulation run is stopped. By default, this variable is set to `t`, which means that if you forcefully stop a simulation run, the tool immediately stops all netlisting jobs associated with that run. This helps in saving netlisting resources occupied by large designs that would otherwise continue to netlist even if the related run is not active. The released resources can be used for the other active runs or histories.

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "endNetlistingImmediatelyOnStop")
envSetVal("maestro.simulation" "endNetlistingImmediatelyOnStop" 'boolean nil)
```

Related Topics

Setting up the Netlisting Job Policy for LSCS

© 2023 Cadence Design Systems, Inc.

groupRunA

```
maestro.simulation groupRunA boolean { t | nil }
```

Description

Enables or disables group submission of points to the available ICRP jobs.

Possible values:

- `t`: Enables group submission of points. ADE Assembler divides the points to be simulated into groups and submits those to the available ICRP jobs at a time. Instead of communicating with ADE Assembler for each point individually, an ICRP runs the simulation for all the points in the group together and returns the results for the group. This enhances the performance of the simulation run.
- `nil`: Disables group submission of points. ADE Assembler submits a single point to an ICRP at a time.

Note: When set to `t`, the points are grouped at the beginning of the simulation. Once the simulation is in progress, any updates made to the value of the *Max. Jobs* field in the Job Policy Setup form would not be reflected.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "groupRunA")
envSetVal("maestro.simulation" "groupRunA" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

ignoreAnalysisCheckInResourceEstimation

```
maestro.simulation ignoreAnalysisCheckInResourceEstimation boolean { t | nil }
```

Description

Specifies whether you want to be notified through a message box when the analysis specified in the setup is not supported for resource estimation by Spectre. The message box lets you decide whether to proceed with the default resource estimation values from the job policy or any other history.

Spectre currently supports resource estimation for only hb and tran analyses. If the setup contains an enabled analysis other than these two, ADE Explorer or ADE Assembler reuse values from an available history. If there is no history available, it uses the default memory and CPU value specified in the *Default Values* group on the *Resources* tab of the Job Policy Setup form.

Possible values:

- `t`: Does not show any message box. The tool automatically reuses the memory and CPU value from an available history or the Job Policy Setup form.
- `nil`: Shows a message box when the analysis specified in the setup is not supported for resource estimation by Spectre.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "ignoreAnalysisCheckInResourceEstimation")
envSetVal("maestro.simulation" "ignoreAnalysisCheckInResourceEstimation" 'boolean
t)
```

Related Topics

Setting up the Resource Estimation Options for LSCS

© 2023 Cadence Design Systems, Inc.

ignoreRunModeCheckInResourceEstimation

```
maestro.simulation ignoreRunModeCheckInResourceEstimation boolean { t | nil }
```

Description

Specifies whether you want to be notified through a message box when the current run mode in the setup is not supported for resource estimation by Spectre. The message box lets you decide whether to proceed with the default resource estimation values from the job policy or any other history.

Spectre does not support resource estimation for Improve Yield, Sensitivity Analysis, and Single Run, Sweeps and Corner with statistical corners. If any of these run modes is set in the current setup, ADE Explorer or ADE Assembler shows an error message and prompts you to confirm whether to proceed by reusing estimated resource values from an available history or the default values from the job policy. However, the tool does not change the values already set in the Job Policy Setup form. If you do not want the tool to show this prompt and to use the available data by default, set this environment variable to `nil`.

Possible values:

- `t`: Does not show any message box. The tool automatically reuses the memory and CPU value from the Job Policy Setup form.
- `nil`: Shows a message box when the current run mode in the setup is not supported for resource estimation by Spectre.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "ignoreRunModeCheckInResourceEstimation")
envSetVal("maestro.simulation" "ignoreRunModeCheckInResourceEstimation" 'boolean
t)
```

Related Topics

Setting up the Resource Estimation Options for LSCS

© 2023 Cadence Design Systems, Inc.

interactiveA

```
maestro.simulation interactiveA boolean { t | nil }
```

Description

Specifies whether the ICRPs need to submit points directly to a Spectre plugin. If netlist recreation and Spectre startup time is comparable to actual simulation time, significant performance improvement can be noticed when the new Spectre plugin is used.

This variable is applicable only for the Spectre simulator.

Possible values:

- `t`: Each ICRP creates an initial netlist for the design and submits all the assigned points along with variable changes to a Spectre plugin. The plugin updates the netlist for each point by using the information about the variables. After the simulation for all the points is complete, Spectre plugin returns the results together for those points to the ICRP. This saves the overhead of Spectre startup, license checks, and netlisting separately for each point.
- `nil`: Each ICRP handles simulation for a single point at a time. For each point, it starts a new session with Spectre, checks out a license, creates a netlist, and runs simulation. After the simulation for a point is complete, Spectre returns the results to the ICRP and kills the session. This involves an overhead of repetitive tasks for each point, and impacts the performance when a simulation run involves a large number of sweep points and corners.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "interactiveA")
envSetVal("maestro.simulation" "interactiveA" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

reportChangesAfterComparingSetupWithHistories

```
maestro.simulation reportChangesAfterComparingSetupWithHistories boolean { t | nil
     }
```

Description

Specifies whether ADE Assembler needs to print a message in the log file to report the changes found in the active setup after comparing it with the setup of the saved histories.

Possible values:

- `t`: ADE runs a simulation if the current setup is different from the setup of the saved histories, and prints a message in the log file to report the changes found in the setup.
- `nil`: ADE runs a simulation, but no information about the changes in the setup is printed in the log file.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "reportChangesAfterComparingSetupWithHistories")
envSetVal("maestro.simulation" "reportChangesAfterComparingSetupWithHistories"
'boolean nil)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

reportIdenticalHistoryIgnoreElementsList

```
maestro.simulation reportIdenticalHistoryIgnoreElementsList string
     list-of-space-separated-element-or-setting
```

Description

Specifies the elements or settings that ADE Assembler should ignore while checking for identical histories.

Possible value: A string value containing a comma-separated list of settings to be ignored. You can specify any element name from the XML saved for the setup database or any environment variable used by ADE Assembler.

Default value: "`plottingoptions, sortVariableValues, overwritehistoryname`"

This default value indicates that ADE Assembler ignores any difference in these three settings:

- The plotting options specified on the Plotting/Printing Options form
- The value of the `sortVariableValues` environment variable
- The value of the `overwriteHistoryName` environment variable

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "reportIdenticalHistoryIgnoreElementsList")
```

```
envSetVal("maestro.simulation" "reportIdenticalHistoryIgnoreElementsList" 'string
"plottingoptions, sortVariableValues, overwritehistoryname")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

retainNetlistLinkForRerun

```
maestro.simulation retainNetlistLinkForRerun boolean { t | nil }
```

Description

When you use the *Rerun Unfinished/Error Points* command, this variable retains netlist links for the points that reuse results from a previous history. The netlist links point to the *designName*/*libName*/*cellName*/*results*/*maestro*/psf/*designName*/netlist/netlist location in the previous history. This helps in saving disk space used to save netlists.

When this variable is set to `nil` and you rerun an unfinished simulation, the points for which the results are being reused contain a copy of the netlist saved during the previous run instead of retaining the symbolic link to the netlist file. The new points simulated in the rerun contain a link to the newly created netlist for the current run.

Possible values are

- `t`: Retains links to the netlist of the previous history for the points that reuse results from that history.
- `nil`: Creates a copy of netlist in the netlist directory for the points that reuse results from that history. For other points that use the newly created netlist, a link to the psf/netlst directory of the new history is saved.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "retainNetlistLinkForRerun")
envSetVal("maestro.simulation" "retainNetlistLinkForRerun" 'boolean t)
```

Related Topics

Simulating Only Unfinished or Erroneous Points

© 2023 Cadence Design Systems, Inc.

reuseNetlistForRerun

```
maestro.simulation reuseNetlistForRerun boolean t | nil
```

Description

Specifies whether the tool needs to create a netlist before rerunning a simulation for unfinished or error points. By default, netlister recreates a design netlist for every rerun. Set this variable to reuse the design netlist from the original simulation during rerun.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "reuseNetlistForRerun")
envSetVal("maestro.simulation" "reuseNetlistForRerun" 'boolean t)
```

Related Topics

Simulating Only Unfinished or Erroneous Points

© 2023 Cadence Design Systems, Inc.

runModesForIdenticalHistoryCheck

`maestro.simulation runModesForIdenticalHistoryCheck string` *`list-of-queue-names`*

Description

Specifies the run mode for which the check to identify identical history needs to be run.

Possible values:

- `"SRSC" or \"Single Run, Sweeps and Corners\"`: Enables reporting of identical histories for the Single Run, Sweeps and Corners run mode.
- `"MC" or "\Monte Carlo Sampling\"`: Enables reporting of identical histories for the Monte Carlo run mode.
- `"GO" or "\Global Optimization\"`: Enables reporting of identical histories for the Global Optimization run mode.
- `"LO" or "\Local Optimization\"`: Enables reporting of identical histories for the Local Optimization run mode.
- `"SA" or "\Sensitivity Analysis\"`: Enables reporting of identical histories for the Sensitivity Analysis run mode.
- `"FA" or "\Feasibility Analysis\"`: Enables reporting of identical histories for the Feasibility Analysis run mode.

Default value: `""`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "runModesForIdenticalHistoryCheck")
envSetVal("maestro.simulation" "runModesForIdenticalHistoryCheck" 'string "SRSC GO LO")
```


© 2023 Cadence Design Systems, Inc.

simulationServiceJobPolicy

`maestro.simulation simulationServiceJobPolicy string` *`name-of-Job-Policy`*

Description

Specifies the name of the job policy to be used by the Simulation Manager process. By default, the policy for Simulation Manger is named as Simulation Manager Default and it is displayed in the *Job Policy Name* field of the of the Job Policy setup form.

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "simulationServiceJobPolicy")
envSetVal("maestro.simulation" "simulationServiceJobPolicy" 'string "SimMgrPol1")
```

Related Topics

Using Simulation Manager with LSCS

© 2023 Cadence Design Systems, Inc.

tempDirectory

`maestro.simulation tempDirectory string` *`path-to-directory`*
simulationServiceJobPolicy

Description

Specifies the path to a directory where a maestro cellview saves the temporary data required to check the health of the netlist service processes and simulation monitor processes for LSCS job control mode. This directory is also used to save the local results database or the results for Real-Time Tuning.

Use this variable to specify a directory path that always exists and can accommodate temporary data. It is recommended to ensure that this location is on a disk that can be accessed fast for read and write operations.

Default value: `/dev/shm`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "tempDirectory")
envSetVal("maestro.simulation" "tempDirectory" 'string "/user/myDir/localPath")
```

Related Topics

Real-Time Tuning in ADE Explorer

LSCS Architecture

© 2023 Cadence Design Systems, Inc.

useReportIdenticalHistory

```
maestro.simulation useReportIdenticalHistory boolean { t | nil }
```

Description

Specifies whether to enable or disable reporting of identical histories before running simulation.

Value values:

- `t`: enables reporting of identical histories
- `nil`: Disables reporting of identical histories

Default value: `nil`

GUI Equivalent

Run Options form – *Enable reporting of identical histories*

Examples

```
envGetVal("maestro.simulation" "useReportIdenticalHistory")
envSetVal("maestro.simulation" "useReportIdenticalHistory" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

useSelfCDFInPostLayoutNetlist

`maestro.simulation useSelfCDFInPostLayoutNetlist string` *`list-of-simulator-names`*

Description

Specifies the list of simulators from which the simulation-specific CDF properties are to be used to apply name prefixes to devices.

For example, you specify the following:

```
envSetVal("maestro.simulation" "useSelfCDFInPostLayoutNetlist" 'string "ams
hspiceD")
```

- If the current simulator is `ams`, the netlister will get the `namePrefix` value for `ams` from the *Simulation Information* section of the Edit CDF form.
- If the current simulator is `hspiceD`, the netlister will get the `namePrefix` value for `hspiceD` from the *Simulation Information* section of the Edit CDF form.
- If the current simulator is `auLvs`, the netlister will get the `namePrefix` value for `auCdl` from the *Simulation Information* section of the Edit CDF form.

The default is `""`, which means that the `namePrefix` value from a different simulator can be used. It is recommended to provide a valid `namePrefix` so that the netlisted devices in the DSPF file have the correct prefix and the simulation does not fail.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "useSelfCDFInPostLayoutNetlist")
envSetVal("maestro.simulation" "useSelfCDFInPostLayoutNetlist" 'string "ams")
envSetVal("maestro.simulation" "useSelfCDFInPostLayoutNetlist" 'string "ams
auLvs")
envSetVal("maestro.simulation" "useSelfCDFInPostLayoutNetlist" 'string "auCdl
auLvs hspiceD")
envSetVal("maestro.simulation" "useSelfCDFInPostLayoutNetlist" 'string "ams
spectre hspiceD")
```


© 2023 Cadence Design Systems, Inc.

Related Topics

Rules for Adding Prefixes to Instance Names

Prefixes in Instance Names in the DSPF File

dspfNamePrefixSimInfo

© 2023 Cadence Design Systems, Inc.

waitToAcquireROLock

`maestro.simulation waitToAcquireROLock int` *`timeSeconds`*

Description

Specifies the time period (in seconds) for which ADE Assembler needs to wait to obtain a lock on a read-only view to make any change in the setup or to run a simulation.

Possible values: An integer value between `1` and `100000`.

Default value: `300`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "waitToAcquireROLock")
envSetVal("maestro.simulation" "waitToAcquireROLock" 'int 300)
```

Related Topics

Working with Read-Only or Editable maestro Views

© 2023 Cadence Design Systems, Inc.

warnWhenWCCSimsExceed

`maestro.simulation warnWhenWCCSimsExceed int` *`maxSimulations`*

Description

Controls the maximum number of simulations for the Worst Case Corners run mode after which ADE Assembler shows a warning is displayed seeking confirmation to continue with the simulation run.

Possible values: A positive integer within the range of `0` and `1000000`.

Default value: `10000`

GUI Equivalent

None

Examples

```
envGetVal("maestro.simulation" "warnWhenWCCSimsExceed")
envSetVal("maestro.simulation" "warnWhenWCCSimsExceed" 'int 8000)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

maestro.sla

- autoUpdateTagForReference
- overwriteEntireCorner
- saveCompositeSetups

© 2023 Cadence Design Systems, Inc.

autoUpdateTagForReference

```
maestro.sla autoUpdateTagForReference boolean { t | nil )
```

Description

Adds the name of the referenced setup as a tag for the variable or corner in the Data View assistant. If the setup name has leading and trailing whitespaces, they are removed and intermediate whitespaces are replaced with an underscore character. For example, if the setup name is ' `Sweep Setup1` ', the tag added to the Data View is '`Sweep_Setup_1`'.

Clicking the context-menu command *Clear References* in the Data View assistant removes the references as well as the tags. Clicking *Sync with Reference Setup* adds the tags back to the variable or corner even if it has been removed.

The default is `nil`, which means that the referenced setup name is not added as a tag for the specified variable or corner.

GUI Equivalent

None

Examples

```
envGetVal("maestro.sla" "autoUpdateTagForReference")
envSetVal("maestro.sla" "autoUpdateTagForReference" 'boolean t)
```

Related Topics

Reusing the Setup from Data View and Setup Library Assistants

© 2023 Cadence Design Systems, Inc.

overwriteEntireCorner

```
maestro.sla overwriteEntireCorner boolean { t | nil }
```

Description

Overwrites the entire corner by using the entire corner as the smallest unit in the reference flow.

When set to `nil`, the variables, parameters, model files, and model groups are considered as independent units in the reference flow. This implies that you can have additional settings for corners in the Data View assistant.

The default is `t`.

GUI Equivalent

None

Examples

```
envGetVal("maestro.sla" "overwriteEntireCorner")
envSetVal("maestro.sla" "overwriteEntireCorner" 'boolean nil)
```

Related Topics

Reusing the Setup from Data View and Setup Library Assistants

© 2023 Cadence Design Systems, Inc.

saveCompositeSetups

```
maestro.sla saveCompositeSetups boolean { t | nil )
```

Description

Saves a combination of all verification spaces that have been used in a session during simulation as a new verification space. This verification space includes the following:

- a superset of all sweep setups
- a superset of all corner setups
- the first simulation setup

You can reuse the saved settings, when required.

The default is `nil`, which means that the combined setups are not saved.

GUI Equivalent

None

Examples

```
envGetVal("maestro.sla" "saveCompositeSetups")
envSetVal("maestro.sla" "saveCompositeSetups" 'boolean t)
```

Related Topics

Reusing the Setup from Data View and Setup Library Assistants

Running Implementation with Multiple Verification Spaces

© 2023 Cadence Design Systems, Inc.

maestro.test

- autoPromoteVarsToGlobal

© 2023 Cadence Design Systems, Inc.

autoPromoteVarsToGlobal

```
maestro.test autoPromoteVarsToGlobal boolean { t | nil }
```

Description

Controls whether design variables are automatically added as global variables on the Data View and on the Variables tab of the Variables and Parameters pane.

Possible values

- `t`: All design variables are automatically added as global variables in the Global Variables tree on the Data View and the Variables tab of the Variables and Parameters pane.
- `nil`: Disables the automatic addition of design variables as global variables.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.test" "autoPromoteVarsToGlobal")
envSetVal("maestro.test" "autoPromoteVarsToGlobal" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

maestro.waveSpec

- abstol
- biasShift
- disableChecks
- excludeTimes
- reltol
- timeTolerance
- timeDelay
- compareAlgorithm

© 2023 Cadence Design Systems, Inc.

abstol

`maestro.waveSpec abstol string` *`list-of-space-separated-queue-names`*

Description

Specifies the default value for the *Absolute Tolerance* field on the Global Waveform Compare Settings form.

GUI Equivalent

Global Waveform Compare Settings form – *Absolute Tolerance*

Examples

```
envGetVal("maestro.waveSpec" "abstol")
envSetVal("maestro.waveSpec" "abstol" 'string "2")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

biasShift

`maestro.waveSpec biasShift string` *`list-of-space-separated-queue-names`*

Description

Specifies the default value for the *Bias Level* field on the Global Waveform Compare Settings form.

Possible value: A positive integer value in string format.

Default value: “”

GUI Equivalent

Global Waveform Compare Settings form – *Bias level (y shift)*

Examples

```
envGetVal("maestro.waveSpec" "biasShift")
envSetVal("maestro.waveSpec" "biasShift" 'string "2")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

disableChecks

```
maestro.waveSpec disableChecks boolean { t | nil }
```

Description

Enables or disables checking of waveform specifications.

Possible values are:

- `t`: Enables checking of waveform specifications.
- `nil`: Disables checking of waveform specifications.

GUI Equivalent

None

Examples

```
envGetVal("maestro.waveSpec" "disableChecks")
envSetVal("maestro.waveSpec" "disableChecks" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

excludeTimes

`maestro.waveSpec excludeTimes string` *`list-of-space-separated-queue-names`*

Description

Specifies the default value for the *Time Tolerance* field on the Global Waveform Compare Settings form.

Possible value:

a comma-separated list of time ranges in the following format:

`"from:to[, from:to]"`

Default value: “”

GUI Equivalent

Global Waveform Compare Settings form – *Time Exclude (seconds)*

Examples:

```
envGetVal("maestro.waveSpec" "excludeTimes")
envSetVal("maestro.waveSpec" "excludeTimes" 'string "")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

reltol

`maestro.waveSpec reltol string` *`list-of-space-separated-queue-names`*

Description

Specifies the default value for the *Relative Tolerance* field on the Global Waveform Compare Settings form.

Possible values: A positive integer value in string format.

Default value: “”

GUI Equivalent

Global Waveform Compare Settings form – *Relative Tolerance*

Examples

```
envGetVal("maestro.waveSpec" "reltol")
envSetVal("maestro.waveSpec" "reltol" 'string "2")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

timeTolerance

`maestro.waveSpec timeTolerance string` *`list-of-space-separated-queue-names`*

Description

Specifies the default value for the *Time Tolerance* field on the Global Waveform Compare Settings form.

Possible value: A positive integer value in string format.

Default Value: “”

GUI Equivalent

Global Waveform Compare Settings form – *Time Tolerance (seconds)*

Examples

```
envGetVal("maestro.waveSpec" "timeTolerance")
envSetVal("maestro.waveSpec" "timeTolerance" 'string "2")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

timeDelay

`maestro.waveSpec timeDelay string` *`list-of-space-separated-queue-names`*

Description

Specifies the default value for the *Time Delay* field on the Global Waveform Compare Settings form.

Possible value: A positive integer value in string format.

Default value: “”

GUI Equivalent

Global Waveform Compare Settings form – *Time Delay (x shift)*

Example

```
envGetVal("maestro.waveSpec" "timeDelay")
envSetVal("maestro.waveSpec" "timeDelay" 'string "2")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

compareAlgorithm

`maestro.waveSpec compareAlgorithm string` *`list-of-space-separated-queue-names`*

Desription

Specifies the default value for the *Comparison Algorithm* field on the Global Waveform Compare Settings form.

Possible values:

- "`Asymmetrical`"
- "`Symmetrical,liberal`"
- "`Symmetrical,conservative`"

Default value: "`Asymmetrical`"

GUI Equivalent

Global Waveform Compare Settings form – Comparison Algorithm

Examples

```
envGetVal("maestro.waveSpec" "compareAlgorithm" )
envSetVal("maestro.waveSpec" "compareAlgorithm" 'string "Asymmetrical")
```

Related Topics

Waveform Compare Algorithms.

© 2023 Cadence Design Systems, Inc.

maestro.mts

- MTSremoveOldConfigSetupAfterMigration

© 2023 Cadence Design Systems, Inc.

MTSremoveOldConfigSetupAfterMigration

```
maestro.mts MTSremoveOldConfigSetupAfterMigration boolean { t | nil }
```

Description

Specifies whether to delete the existing MTS settings from the config view when an ADE L state is migrated to a maestro cellview.

Possible Values:

- `t`: Deletes the existing MTS settings during the migration.
- `nil`: Retains the existing config view during the migration.

Default value: `nil`

GUI Equivalent

None

Examples

```
envGetVal("maestro.mts" "MTSremoveOldConfigSetupAfterMigration")
envSetVal("maestro.mts" "MTSremoveOldConfigSetupAfterMigration" 'boolean nil)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

adexl.algorithm

- linearModelR2
- quadraticModelR2

© 2023 Cadence Design Systems, Inc.

linearModelR2

```
adexl.algorithm linearModelR2 int 90
```

Description

Specifies the R2 threshold for the linear model used in the reorder algorithms. A higher threshold leads to higher accuracy for the final linear model and therefore, takes more time to build. This variable is used when you run Monte Carlo for mismatch contribution analysis, yield verification with reorder samples, or for statistical corners creation using worst samples.

Possible values: A positive value between `80` and `99`.

Default value: `90`

GUI Equivalent

None

Examples

```
envGetVal("adexl.algorithm" "linearModelR2")
envSetVal("adexl.algorithm" "linearModelR2" 'int 90)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

quadraticModelR2

```
adexl.algorithm quadraticModelR2 int 90
```

Description

Specifies the R2 threshold for the quadratic model used in the reorder algorithms. A higher threshold leads to higher accuracy for the final algorithm model and therefore, takes more time to build.This variable is used when you run Monte Carlo for mismatch contribution analysis, yield verification with reorder samples, or for statistical corners creation using worst samples.

Possible values: integer - A positive integer between `80` and `99`.

Default value: `90`

GUI Equivalent

None

Examples

```
envGetVal("adexl.algorithm" "quadraticModelR2")
envSetVal("adexl.algorithm" "quadraticModelR2" 'int 90)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

adexl.setupdb

- loadSetupToActiveAlsoViewsResults
- saveDir
- percentageForNearSpec
- useNMPForMapping

adexl.test

- autoCopyCellviewVars
- checkForUnsavedViewsUponRun
- initiallyAddNameUniqifier

adexl.simulation

- autoDetectNetlistProcs
- createCompositeSimLogFileWhenSimCountFewerThan
- createRunLogForSweepsCorners
- createRunLogWhenSimsFewerThan
- checkInstanceBindings
- haltCurrentRunAfterPreRunTrigger
- ignoreDesignChangesDuringRun
- ignoredLibsForDUT
- includeStatementForNetlistInSimInputFile
- matlabExprReevalSearchUpdateLevel
- matlabStartTimeout
- matlabEnableIdleTimeout
- matlabIdleTimeout
- matlabResultTimeout

© 2023 Cadence Design Systems, Inc.

- moveConfigsToNetlistDir
- overrideNetlistProcDetection
- overwriteHistory
- overwriteHistoryName
- retainNetlistsOverwriteHistory
- saveBestNDesignPoints
- saveBestPointsStrategy
- saveLastNHistoryEntries
- saveNetlistData
- saveRawData
- setCurrentRunPostSimulation
- showErrorForNonExistingVariables
- showWarningForReferenceNetlist
- singleNetlistForAllPoints
- sortVariableValues
- stopManualTuningOnSessionExit
- warnWhenSimsExceed

© 2023 Cadence Design Systems, Inc.

matlabExprReevalSearchUpdateLevel

```
adexl.simulation matlabExprReevalSearchUpdateLevel int 0
```

Description

A MATLAB script can call another script, which can further call another script, and so on. You can use this variable to control the depth level up to which the scripts will be traversed to re-evaluate a MATLAB expression.

Possible values: integer - A positive integer value `0` and `10000`.

Default value: `0`

GUI Equivalent

None

Examples

```
envGetVal("adexl.simulation" "matlabExprReevalSearchUpdateLevel")
envSetVal("adexl.simulation" "matlabExprReevalSearchUpdateLevel" 'int 0)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

matlabStartTimeout

```
adexl.simulation matlabStartTimeout int 0
```

Description

The time, in seconds, for which ADE XL waits for MATLAB to start evaluation of an expression.

Note: This environment variable is not applicable when MATLAB is opened in the standalone mode.

Possible values: integer - A positive integer value between `1` and `10000`.

Default value: `60`

GUI Equivalent

None

Examples

```
envGetVal("adexl.simulation" "matlabStartTimeout")
envSetVal("adexl.simulation" "matlabStartTimeout" 'int 0)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

matlabEnableIdleTimeout

```
adexl.simulation matlabEnableIdleTimeout boolean {t | nil}
```

Description

Specifies whether the MATLAB engine can be kept running in an idle state after the evaluation is complete. This ensures that the MATLAB engine is available for the next evaluation, and the time to restart the engine is saved.

To specify the timeout duration, use `matlabIdleTimeout`.

- `t`: Even after the evaluation is finished, MATLAB engine continues to run in an idle state for the time duration specified through `matlabIdleTimeout`.

  This is the default.

- `nil`: MATLAB engine exits immediately after the evaluation.

GUI Equivalent

None

Examples

```
envGetVal("adexl.simulation" "matlabEnableIdleTimeout")
envSetVal("adexl.simulation" "matlabEnableIdleTimeout" 'boolean nil)
```

Related Topics

matlabIdleTimeout

© 2023 Cadence Design Systems, Inc.

matlabIdleTimeout

```
adexl.simulation matlabIdleTimeout int 0
```

Description

Specifies the maximum time (in seconds) for which the MATLAB engine can be kept running in an idle state. While evaluating a MATLAB expression for the first time in a session, ADE Explorer and ADE Assembler start the MATLAB engine in the background. After the evaluation is complete, the MATLAB engine is not closed so that it is available for the next evaluation and the time to restart the engine is saved. However, it is not recommended to keep the engine running in the idle state for a long time. Therefore, the MATLAB session is closed and its license is released if it remains in the idle state for the time specified by this environment variable.

Note: This environment variable is not applicable when MATLAB is opened in the standalone mode.

Possible values: integer - A positive integer value between `1` and `10000`.

Default value: `60`

GUI Equivalent

None

Examples

```
envGetVal("adexl.simulation" "matlabIdleTimeout")
envSetVal("adexl.simulation" "matlabIdleTimeout" 'int 0)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

checkInstanceBindings

```
adexl.simulation checkInstanceBindings boolean { t | nil }
```

Description

If config sweeps are used in the simulation setup, ADE Explorer or ADE Assembler check for instance bindings in the Hierarchy Editor. If the instance or occurrence bindings in the config view conflict with the config sweeps, the tool displays a message prompt with a list of those instances and seeking confirmation whether the tool should clear the bindings in the config view. When this variable is set to nil, this check is not run, but the netlist may not be correct in that case.

Possible values:

- `t`: Before running a simulation, ADE Explorer or ADE Assembler runs a check to identify instance or occurrence bindings in the config view that conflict with the config sweeps.
- `nil`: This check is not run.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("adexl.simulation" "checkInstanceBindings")
envSetVal("adexl.simulation" "checkInstanceBindings" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

adexl.distribute

- continueICRPRunOnAbruptGUIExit
- createUniqueLogsDirForICRPLogs
- defaultRunInParallel
- defaultPerRunNumJobs
- enableICRPReconnect
- generateJobFileOnlyOnError
- inferCommandICRPStatusFromProxy
- jobFileDir
- maxJobsIsHardLimit
- numRetriesOnError
- runTimeoutScalingStartsAfterSimCount
- runTimeoutScaleFactor
- useAllLingeringJobs
- useAsRunTimeout
- useSameProcess

adexl.monte

- applySaveOptionsToNetlist
- createStatisticalCornerType
- incrementalUpdate
- iterationUpdates
- minGroupSizeSplitAcrossIdleJobs
- numberOfPointsToView
- samplingMethod
- saveProcessOptionDefaultValue

© 2023 Cadence Design Systems, Inc.

- saveMismatchOptionDefaultValue
- saveSimulationData
- warnWhenSimsExceed

adexl.historyNamePrefix

- showNameHistoryForm
- initiallyAddHistoryNameUniquifier
- singleRunSweepsAndCorners
- monteCarloSampling
- WorstCaseCorners
- globalOptimization
- localOptimization
- improveYield
- highYieldEstimation
- sensitivityAnalysis
- feasibilityAnalysis
- manualTuning
- sizeOverCorners
- runPlan

© 2023 Cadence Design Systems, Inc.

runPlan

`adexl.historyNamePrefix runPlan string` *`list-of-space-separated-queue-names`*

Description

Specifies the default history name to be used for a run plan.

Possible values: A string specifying the history name.

Default value: `Plan`

GUI Equivalent

None

Examples

```
envGetVal("adexl.historyNamePrefix" "runPlan")
envSetVal("adexl.historyNamePrefix" "runPlan" 'string "Plan")
```


© 2023 Cadence Design Systems, Inc.

adexl.algorithm

- linearModelR2
- quadraticModelR2

© 2023 Cadence Design Systems, Inc.

linearModelR2

```
adexl.algorithm linearModelR2 int 90
```

Description

Specifies the R2 threshold for the linear model used in the reorder algorithms. A higher threshold leads to higher accuracy for the final linear model and therefore, takes more time to build. This variable is used when you run Monte Carlo for mismatch contribution analysis, yield verification with reorder samples, or for statistical corners creation using worst samples.

Possible values: A positive value between `80` and `99`.

Default value: `90`

GUI Equivalent

None

Examples

```
envGetVal("adexl.algorithm" "linearModelR2")
envSetVal("adexl.algorithm" "linearModelR2" 'int 90)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

quadraticModelR2

```
adexl.algorithm quadraticModelR2 int 90
```

Description

Specifies the R2 threshold for the quadratic model used in the reorder algorithms. A higher threshold leads to higher accuracy for the final algorithm model and therefore, takes more time to build.This variable is used when you run Monte Carlo for mismatch contribution analysis, yield verification with reorder samples, or for statistical corners creation using worst samples.

Possible values: A positive integer between `80` and `99`.

Default value: `90`

GUI Equivalent

None

Examples

```
envGetVal("adexl.algorithm" "quadraticModelR2")
envSetVal("adexl.algorithm" "quadraticModelR2" 'int 90)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

adexl.setupdb

- loadSetupToActiveAlsoViewsResults
- saveDir
- percentageForNearSpec
- useNMPForMapping

adexl.test

- autoCopyCellviewVars
- checkForUnsavedViewsUponRun
- initiallyAddNameUniqifier

adexl.simulation

- autoDetectNetlistProcs
- createCompositeSimLogFileWhenSimCountFewerThan
- createRunLogForSweepsCorners
- createRunLogWhenSimsFewerThan
- checkInstanceBindings
- haltCurrentRunAfterPreRunTrigger
- ignoreDesignChangesDuringRun
- ignoredLibsForDUT
- includeStatementForNetlistInSimInputFile
- matlabExprReevalSearchUpdateLevel
- matlabStartTimeout
- matlabIdleTimeout
- matlabResultTimeout
- moveConfigsToNetlistDir

© 2023 Cadence Design Systems, Inc.

- overrideNetlistProcDetection
- overwriteHistory
- overwriteHistoryName
- retainNetlistsOverwriteHistory
- saveBestNDesignPoints
- saveBestPointsStrategy
- saveLastNHistoryEntries
- saveNetlistData
- saveRawData
- setCurrentRunPostSimulation
- showErrorForNonExistingVariables
- showWarningForReferenceNetlist
- singleNetlistForAllPoints
- sortVariableValues
- stopManualTuningOnSessionExit
- warnWhenSimsExceed

© 2023 Cadence Design Systems, Inc.

matlabExprReevalSearchUpdateLevel

```
adexl.simulation matlabExprReevalSearchUpdateLevel int 0
```

Description

A MATLAB script can call another script, which can further call another script, and so on. You can use this variable to control the depth level up to which the scripts will be traversed to re-evaluate a MATLAB expression.

Possible values: integer - A positive integer value `0` and `10000`.

Default value: `0`

GUI Equivalent

None

Examples

```
envGetVal("adexl.simulation" "matlabExprReevalSearchUpdateLevel")
envSetVal("adexl.simulation" "matlabExprReevalSearchUpdateLevel" 'int 0)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

matlabStartTimeout

```
adexl.simulation matlabStartTimeout int 0
```

Description

The time, in seconds, for which ADE XL waits for MATLAB to start evaluation of an expression.

Note: This environment variable is not applicable when MATLAB is opened in the standalone mode.

Possible values: integer - A positive integer value between `1` and `10000`.

Default value: `60`

GUI Equivalent

None

Examples

```
envGetVal("adexl.simulation" "matlabStartTimeout")
envSetVal("adexl.simulation" "matlabStartTimeout" 'int 0)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

matlabIdleTimeout

```
adexl.simulation matlabIdleTimeout int 0
```

Description

Specifies the maximum time (in seconds) for which the MATLAB engine can be kept running in an idle state. While evaluating a MATLAB expression for the first time in a session, ADE Explorer and ADE Assembler start the MATLAB engine in the background. After the evaluation is complete, the MATLAB engine is not closed so that it is available for the next evaluation and the time to restart the engine is saved. However, it is not recommended to keep the engine running in the idle state for a long time. Therefore, the MATLAB session is closed and its license is released if it remains in the idle state for the time specified by this environment variable.

Note: This environment variable is not applicable when MATLAB is opened in the standalone mode.

Possible values: integer - A positive integer value between `1` and `10000`.

Default value: `60`

GUI Equivalent

None

Examples

```
envGetVal("adexl.simulation" "matlabIdleTimeout")
envSetVal("adexl.simulation" "matlabIdleTimeout" 'int 0)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

checkInstanceBindings

```
adexl.simulation checkInstanceBindings boolean { t | nil }
```

Description

If config sweeps are used in the simulation setup, ADE Explorer or ADE Assembler check for instance bindings in the Hierarchy Editor. If the instance or occurrence bindings in the config view conflict with the config sweeps, the tool displays a message prompt with a list of those instances and seeking confirmation whether the tool should clear the bindings in the config view. When this variable is set to nil, this check is not run, but the netlist may not be correct in that case.

Possible values:

- `t`: Before running a simulation, ADE Explorer or ADE Assembler runs a check to identify instance or occurrence bindings in the config view that conflict with the config sweeps.
- `nil`: This check is not run.

Default value: `t`

GUI Equivalent

None

Examples

```
envGetVal("adexl.simulation" "checkInstanceBindings")
envSetVal("adexl.simulation" "checkInstanceBindings" 'boolean t)
```

Related Topics

© 2023 Cadence Design Systems, Inc.

adexl.distribute

- continueICRPRunOnAbruptGUIExit
- createUniqueLogsDirForICRPLogs
- defaultRunInParallel
- defaultPerRunNumJobs
- enableICRPReconnect
- generateJobFileOnlyOnError
- inferCommandICRPStatusFromProxy
- jobFileDir
- maxJobsIsHardLimit
- numRetriesOnError
- runTimeoutScalingStartsAfterSimCount
- runTimeoutScaleFactor
- useAllLingeringJobs
- useAsRunTimeout
- useSameProcess

adexl.monte

- applySaveOptionsToNetlist
- createStatisticalCornerType
- incrementalUpdate
- iterationUpdates
- minGroupSizeSplitAcrossIdleJobs
- numberOfPointsToView
- samplingMethod
- saveProcessOptionDefaultValue

© 2023 Cadence Design Systems, Inc.

- saveMismatchOptionDefaultValue
- saveSimulationData
- warnWhenSimsExceed

adexl.historyNamePrefix

- showNameHistoryForm
- initiallyAddHistoryNameUniquifier
- singleRunSweepsAndCorners
- monteCarloSampling
- WorstCaseCorners
- globalOptimization
- localOptimization
- improveYield
- highYieldEstimation
- sensitivityAnalysis
- feasibilityAnalysis
- manualTuning
- sizeOverCorners
- runPlan

© 2023 Cadence Design Systems, Inc.

runPlan

`adexl.historyNamePrefix runPlan string` *`list-of-space-separated-queue-names`*

Description

Specifies the default history name to be used for a run plan.

Possible values: A string specifying the history name.

Default value: `Plan`

GUI Equivalent

None

Examples

```
envGetVal("adexl.historyNamePrefix" "runPlan")
envSetVal("adexl.historyNamePrefix" "runPlan" 'string "Plan")
```

Related Topics

© 2023 Cadence Design Systems, Inc.

adexl.icrpStartup

- binaryName
- defaultJobPolicy
- enableOutdirmessage URL ../anasimhelp/appA.html#refreshCDF
- showJobStdout
- showJobStderr
- showOutputLogOnError

adexl.results

- checksAssertsFiltersPath
- checkAssertsViewTool
- defaultBackAnnotationOption
- defaultResultsViewForSweepsCorners
- exportPreserveScalingFactors
- retainReferenceSimResults
- saveDir
- saveLocalPsfDir
- saveResDir
- saveResultsFromHistoryDir

adexl.gui

- continueJobsOnExitQuery
- copyMeasurementScripts
- copyPreRunScripts
- confirmReEvaluationWhen
- continueJobsOnExitQuery

© 2023 Cadence Design Systems, Inc.

- defaultCorners
- defaultCornerExportFileFormat
- defaultCornerImportFileFormat
- defaultParametersViewBy
- descendIntoSubcktForShowingInstOrNet
- detailtransposeViewShowDefault
- disableConstraintsRead
- disableNominalSimulation
- disableRunInReadOnly
- disableSimulationsDefault
- enableAutoRefreshSetupSummary
- enableAutoRefreshPointsTable
- enableMonteCarloForRunPreview
- forceShowAutomaticExpressions
- filterCDFParamsWithZeroOrNegativeOneDefValue
- headerAlignmentSide
- headerTruncationWidth
- LimitModelSections
- maxNotesLength
- maxNotesRowsDisplay
- mismatchPairs
- modelSectionFilterFunction
- numberOfBestPointsToView
- openDesignAccessMode
- openSchInWin
- openTerminalCommand
- outputTabsShowDefault

© 2023 Cadence Design Systems, Inc.

- pcfPrependBasePath
- reEvalOnlyMostRecentHistory
- reEvaluationAgeHoursThreshold
- reEvaluationMode
- reEvaluationRemovingOutputsThreshold
- reEvaluationWhenActiveAndHistoryTestsDiffer
- saveStateQuery
- sendOutputsToEEFilter
- setHistoryPrefixToSetupStateNameOnLoad
- setupFormDefaultEnabled
- setupFormDefaultLoadOperation
- significantDigits
- showSimLogForOnePointSim
- specComparisonMode
- testsShownInOutputsSetup
- toolbarButtonStyle
- yieldViewShowDefault
- zoomToProbedInstOrNet

adexl.cpupdtr

- copyResultsData

adexl.datasheet

- CSSFile
- customFiles
- mainDocXSLFile
- testDocXSLFile

© 2023 Cadence Design Systems, Inc.

- waveformFileExtension
- whatToSaveDefault

asimenv.startup

- copyDesignVarsFromCellview

adexl.oceanxl

- includeSimLogInJobLog

adexl.plotting

- histogramBins
- histogramType
- histogramQQPlot
- maxHistogramBins
- plotScalarExpressions
- plotSignals
- plotType
- plotWaveExpressions
- resultsCacheSize
- showHistogramDensity
- showHistogramDeviation
- showHistogramPoints
- showHistogramPercentMarkers

asimenv.plotting

- specMarker

© 2023 Cadence Design Systems, Inc.

asimenv

- allowSignalsExpressionInSameSubwindow

© 2023 Cadence Design Systems, Inc.